Bamboo 2.0

Documentation for Bamboo 2.0

edwin@atlassian.com (Jul 31, 2006)

smaddox (Apr 14, 2008)

Available Pages

- Bamboo Documentation Home
  - Bamboo Administrator's Guide
    - 01. Configuring Projects and Plans
      - 1.1 About Projects, Plans and Builds
      - 1.2 Creating a Plan
        - 1.2.1 Specifying a Plan's Details
        - 1.2.2 Specifying a Plan's Source Repository
          - CVS
          - Perforce
          - Subversion
        - 1.2.3 Specifying a Plan's Builder
        - Defining Global Variables
          - Using Global or Build-specific Variables
        - 1.2.4 Specifying a Plan's Capability Requirements
        - 1.2.5 Specifying a Plan's Build Artifacts
        - 1.2.6 Specifying a Plan's Notifications
        - 1.2.7 Specifying a Plan's Post Actions
          - Specifying Expiry for a Plan's Build Results
          - Specifying Labels for a Plan's Build Results
        - 1.2.8 Specifying a Plan's Permissions
      - 1.3 Editing a Plan
      - 1.4 Disabling or deleting a Plan
        - 1.4.1 Deleting a Build Result
        - 1.4.2 Deleting a Plan's Working Files
        - 1.4.3 Stopping an Active Build
      - 1.5 Moving a Plan to a different Project
      - 1.6 Renaming a Plan or Project
    - 02. Configuring Agents and Capabilities
      - 2.1 About Agents and Capabilities
      - 2.2 Creating a Local Agent
      - 2.3 Creating a Remote Agent
        - 2.3.1 Disabling Remote Agents Support
      - 2.4 Editing an Agent's Details
      - 2.5 Monitoring Agent Status
      - 2.6 Disabling or deleting an Agent
      - 2.7 Viewing an Agent
2.7.1 Viewing an Agent's Capabilities
2.7.2 Viewing the Plans that an Agent can build
2.7.3 Determining which Agents can build which Plans
2.7.4 Viewing an Agent's System Properties
2.8 Configuring Capabilities
2.8.1 Configuring a new Builder
2.8.1.1 Configuring an Agent-specific Builder Capability
2.8.1.2 Configuring a Shared Builder Capability
2.8.1.3 Viewing your Builder Capabilities
2.8.2 Configuring a new JDK
2.8.2.1 Configuring an Agent-specific JDK Capability
2.8.2.2 Configuring a Shared JDK Capability
2.8.2.3 Viewing your JDK Capabilities
2.8.3 Configuring a new Custom Capability
2.8.3.1 Configuring an Agent-specific Custom Capability
2.8.3.2 Configuring a Shared Custom Capability
2.8.4 Configuring a new Perforce Capability
2.8.5 Editing a Capability
2.8.6 Renaming a Capability
2.8.7 Deleting a Capability
2.8.8 Viewing the Agents & Plans Related to a Capability

3. Triggering a Build
3.1 About Build Triggering
3.2 Triggering a Build when Code is Updated
3.2.1 Polling the Repository for Code Changes
3.2.2 Triggering a Build on Code Check-in
3.3 Triggering a Build on Schedule
3.3.1 Scheduling a Single Daily Build
3.3.2 Specifying a Cron-based Schedule
3.4 Triggering a Build when another Build finishes
3.5 Triggering a Build Manually

4. Configuring Email and Instant Messaging Notifications
4.1 Enabling or disabling Notifications for a Plan
4.2 Configuring Bamboo to send SMTP Email
4.3 Configuring Bamboo to use Instant Messaging (IM)
4.3.1 Configuring Bamboo to use Google Talk for Instant Messaging

5. Managing Users and Security
5.01 Creating a User
5.02 Changing a User's Password or Details
5.03 Deleting or deactivating a User
5.04 Creating a Group
5.05 Deleting a Group
5.06 Adding Users to and removing them from Groups
• 5.07 Granting Plan Permissions to Users or Groups
• 5.08 Granting Global Permissions to Users or Groups
• 5.09 Granting Administration Rights to a User
• 5.10 Allowing Anonymous Users to access Bamboo
• 5.11 Enabling or disabling Contact Details Display
• 5.12 Enabling or disabling Public Signup
• 5.13 Working with External User Repositories
  • 5.13.1 Integrating Bamboo with Crowd
  • 5.13.2 Integrating Bamboo with LDAP
    • 5.13.2.1 Configuring the Caching of your LDAP Repository
• 06. Configuring Plugins
  • 6.1 About Bamboo Plugins
  • 6.2 Enabling the 'Auto-Favourites' Plugin
  • 6.3 Installing a new Plugin
• 07. Managing Data and Backups
  • 7.1 Locating Important Directories and Files
  • 7.2 Specifying Bamboo's Working Directory
  • 7.3 Viewing your Database Connection Details
  • 7.4 Moving your Bamboo Data to a different Database
  • 7.5 Optimising or Re-indexing Data
  • 7.6 Enabling Expiry of Build Results
  • 7.7 Specifying a Backup Schedule
  • 7.8 Exporting Data for Backup
  • 7.9 Importing Data from Backup
• 08. Configuring System Settings
  • 8.1 Viewing Bamboo's System Information
  • 8.2 Updating your Bamboo License Details
  • 8.3 Specifying Bamboo's Title
  • 8.4 Specifying Bamboo's URL
  • 8.5 Enabling GZIP Compression
  • 8.6 Enabling Bamboo's Remote API
  • 8.7 Enabling JIRA integration
• 09. Bamboo Security
  • Bamboo Security Advisory 2008-02-08 (Bamboo 2.0 Beta)
  • Securing your Remote Agents
• Appendix A. Embedding Bamboo into Other Applications
  • Javascript Widgets
    • All Plans & My Favourite Plans
    • Latest Builds
    • Latest Status of a Plan
    • My Latest Changes
    • Plan Summary Graphs
  • Bamboo Development Hub
• Bamboo API
• Bamboo Developer FAQ
• Bamboo Plugin Developer’s Guide
  • Build Complete Action Module
  • Builder Plugin Module
  • Build Processor Module
  • Getting Started
  • Index Reader Module
  • Notification Condition Module
  • Post Build Index Writer Module
  • Pre Build Action Module
  • Report Module
  • Repository Extensions
  • Source Repository Module
  • Web Item Module
  • Web Section Module
  • XWork Plugin Module
• Bamboo Plugin Tutorial
  • Tutorial 1 - Getting Started with a Simple Post Build Labeller
  • Tutorial 2 - Configurable Regex Labeller
• Changes for Bamboo 2.0
  • Build Process for 2.0
  • Post-Build Processing in Bamboo 2.0
  • Repository plugin changes in 2.0
  • Updates to the build processing plugins
• Setting up Bamboo Development Environment in IDEA
• Bamboo Installation & Upgrade Guide
• Bamboo Installation Guide
  • Bamboo EAR-WAR Installation Guide
  • Bamboo Remote Agent Installation Guide
  • Bamboo Standalone Installation Guide (Linux)
  • Bamboo Standalone Installation Guide (Mac)
  • Bamboo Standalone Installation Guide (Windows)
• Running the Setup Wizard
  • 1. Using Bamboo’s embedded HSQL database
  • 2. Connecting Bamboo to an external database
    • 2.1 MySQL 4.1 and 5.0
    • 2.2 Postgres 8+
    • 2.3 Oracle 9i and 10g
    • 2.4 Microsoft SQL Server
    • 2.5 Unsupported databases
• Bamboo Release Notes
  • Bamboo 1.0 Release Notes
1.1 Using the Bamboo Dashboard
1.2 Viewing Bamboo's Current Activity
1.3 Viewing your Latest Build Results
1.4 Working with Favourites
   1.4.1 Adding a Plan to your Favourites
   1.4.2 Removing a Plan from your Favourites
1.5 Displaying a Build Monitor
1.6 Viewing Bamboo's Agents

02. Working with Projects and Plans
   2.1 About Projects and Plans
   2.2 Viewing a Plan's Details
   2.3 Viewing a Plan's Activity Log

03. Working with Build Results
   3.1 About Builds and Build Results
   3.2 Viewing a Build Result
   3.3 Viewing the Code Changes that triggered a Build
   3.4 Viewing a Build's Artifacts
   3.5 Viewing a Build Log
   3.6 Viewing the Metadata for a Build Result
   3.7 Viewing the Clover Code-Coverage for a Build Result
   3.8 Viewing the JIRA Issues for a Build Result

04. Working with Tests
   4.1 Viewing Test Results for a Build
   4.2 Viewing a Test's History
   4.3 Viewing Test Statistics for a Plan

05. Reporting on Plan Trends
   5.1 Viewing Build Statistics for a Plan
   5.2 Generating Reports across multiple Plans
      'Build Activity per Plan' Report
      'Build Duration per Plan' Report
      'Clover Code Coverage per Plan' Report
      'Clover Lines of Code per Plan' Report
      'Number of Build Failures per Plan' Report
      'Number of Tests per Plan' Report
      'Percentage of Successful Builds per Plan' Report
      'Time to Fix per Plan' Report

06. Reporting on Author Trends
   6.1 Viewing Build Statistics for all Authors
   6.2 Viewing Build Results for an Author
   6.3 Generating Reports on selected Authors
      'Build Activity per Author' Report
      'Number of Build Failures per Author' Report
      'Number of Builds Broken per Author' Report
• 'Number of Builds Fixed per Author' Report
• 'Percentage of Successful Builds per Author' Report

• 07. Working with Comments
  • 7.1 About Comments
  • 7.2 Commenting about a Build Result
  • 7.3 Viewing Comments about a Build Result
  • 7.4 Viewing Code Check-in Comments

• 08. Working with Labels
  • 8.1 About Labels
  • 8.2 Labelling a Build Result
  • 8.3 Removing a Label from a Build Result
  • 8.4 Viewing Labelled Build Results
  • 8.5 Viewing Popular Labels

• 09. Subscribing to RSS Feeds
  • 9.1 Subscribing to an RSS Feed for All Build Results for All Plans
  • 9.2 Subscribing to an RSS Feed for Failed Builds for All Plans
  • 9.3 Subscribing to an RSS Feed for All Build Results for a Particular Plan
  • 9.4 Subscribing to an RSS Feed for Failed Builds for a Particular Plan
  • 9.5 Subscribing to an RSS Feed for Labelled Build Results

• 10. Working with Instant Messenger (IM) Notifications
  • 10.1 About Instant Messenger (IM) Notifications
  • 10.2 Labelling a Build Result via IM
  • 10.3 Commenting about a Build Result via IM

• 11. Editing your User Profile
  • 11.1 Changing your Password
  • 11.2 Changing your Notification Preferences
  • 11.3 Associating your Author Name with your User Profile

• Glossary
  • activity log
  • agent
  • agent-specific capability
  • artifact
  • author
  • build
  • build activity
  • build duration
  • builder
  • build log
  • build plan
  • build queue
  • build result
  • build telemetry
• capability
• child
• committer
• custom capability
• favourites
• global permission
• label
• local agent
• parent
• permission
• plan
• plan permission
• project
• queue
• reason
• remote agent
• requirement
• shared capability
• triggering
• watcher
• .bookmarks
• Diagrams
  • How are builds distributed to agents?
  • How do capabilities affect the distribution of builds to agents?
• __newreleaseBamboo
• TreeNavigation
Bamboo 2.0 has been released.

- Take a look at the features of Bamboo's [latest released version](#) and try it out!
- Read the [2.0 Release Notes](#).

<table>
<thead>
<tr>
<th>Installing/upgrading Bamboo 2.0</th>
<th>Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installation Guide</td>
<td>If you have a question about using Bamboo, please feel free to contact us at <a href="#">support</a>. You may also want to check out the mailing list forums:</td>
</tr>
<tr>
<td>Upgrade Guide</td>
<td>- Bamboo Announcements</td>
</tr>
<tr>
<td>Release Notes</td>
<td>- Bamboo General Forum</td>
</tr>
<tr>
<td>Download Bamboo</td>
<td>- Bamboo Developers Forum</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Using/administering Bamboo 2.0</th>
<th>Other handy links:</th>
</tr>
</thead>
<tbody>
<tr>
<td>User's Guide</td>
<td>- Knowledge Base</td>
</tr>
<tr>
<td>Administrator's Guide</td>
<td>- <a href="#">JIRA Issue Tracker &amp; Feature Requests for Bamboo</a></td>
</tr>
<tr>
<td>Bamboo Security</td>
<td></td>
</tr>
<tr>
<td>Development Hub</td>
<td>Bamboo Extensions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>About</th>
<th>Previous Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bamboo is a <strong>continuous integration</strong> (CI) server. Bamboo assists software development teams by providing:</td>
<td></td>
</tr>
<tr>
<td>- automated building and testing of software source-code status.</td>
<td></td>
</tr>
<tr>
<td>- updates on successful/failed builds.</td>
<td></td>
</tr>
<tr>
<td>- reporting tools for statistical analysis.</td>
<td></td>
</tr>
<tr>
<td>Download</td>
<td>Bamboo 1.2 Documentation Home</td>
</tr>
<tr>
<td>Documentation for the Bamboo 2.0 Beta is not available for download.</td>
<td>Bamboo 1.1 Documentation Home</td>
</tr>
<tr>
<td></td>
<td>Bamboo 1.0 Documentation Home</td>
</tr>
</tbody>
</table>

Recently Updated

by [Sarah Maddox](#) (9 minutes ago)

by [Bamboo 2.0](#) (13 minutes ago)

by [Bamboo Documentation Home](#) (16 minutes ago)
by Sarah Maddox (19 minutes ago)
Using Global or Build-specific Variables
by Sarah Maddox (22 minutes ago)
Bamboo Remote Agent Installation Guide
by Sarah Maddox (31 minutes ago)
Bamboo 2.0 Release Notes
by Andrew Lui (10 Apr)
by Ajay Sridhar (10 Apr)
Removing Coverage plug-in data from the Bamboo database
by Andrew Lui (10 Apr)
Bamboo 2.0 Upgrade Guide
by Oleksii Gnatkevych (09 Apr)
Re: 5.13.2 Integrating Bamboo with LDAP
Bamboo Administrator's Guide

This page last changed on Aug 09, 2007 by rosie@atlassian.com.

**About**

Bamboo is a continuous integration (CI) server. Bamboo assists software development teams by providing:

- automated building and testing of software source-code status.
- updates on successful/failed builds.
- reporting tools for statistical analysis.

The Bamboo Administrator's Guide provides information about configuring builds and administering your Bamboo system.

For more documentation please visit Bamboo Documentation Home.

**Download**

You can download the Bamboo documentation in PDF, HTML or XML formats.

**Search the Administrator's Guide**

**Getting Help**

Support Request | Feature Request | Forums | Knowledge Base

---

**Table of Contents**

01. Configuring Projects and Plans

- 1.1 About Projects, Plans and Builds
- 1.2 Creating a Plan
  - 1.2.1 Specifying a Plan's Details
  - 1.2.2 Specifying a Plan's Source Repository
    - CVS
    - Perforce
    - Subversion
  - 1.2.3 Specifying a Plan's Builder
    - Defining Global Variables
    - Using Global or Build-specific Variables
  - 1.2.4 Specifying a Plan's Capability Requirements
  - 1.2.5 Specifying a Plan's Build Artifacts
  - 1.2.6 Specifying a Plan's Notifications
  - 1.2.7 Specifying a Plan's Post Actions
    - Specifying Expiry for a Plan's Build Results
    - Specifying Labels for a Plan's Build Results
  - 1.2.8 Specifying a Plan's Permissions
- 1.3 Editing a Plan
- 1.4 Disabling or deleting a Plan
  - 1.4.1 Deleting a Build Result
  - 1.4.2 Deleting a Plan's Working Files
  - 1.4.3 Stopping an Active Build
- 1.5 Moving a Plan to a different Project
- 1.6 Renaming a Plan or Project

02. Configuring Agents and Capabilities

- 2.1 About Agents and Capabilities
- 2.2 Creating a Local Agent
- 2.3 Creating a Remote Agent
2.3.1 Disabling Remote Agents Support

2.4 Editing an Agent’s Details

2.5 Monitoring Agent Status

2.6 Disabling or deleting an Agent

2.7 Viewing an Agent

  2.7.1 Viewing an Agent’s Capabilities
  2.7.2 Viewing the Plans that an Agent can build
  2.7.3 Determining which Agents can build which Plans
  2.7.4 Viewing an Agent’s System Properties

2.8 Configuring Capabilities

  2.8.1 Configuring a new Builder
    2.8.1.1 Configuring an Agent-specific Builder Capability
    2.8.1.2 Configuring a Shared Builder Capability
    2.8.1.3 Viewing your Builder Capabilities
  2.8.2 Configuring a new JDK
    2.8.2.1 Configuring an Agent-specific JDK Capability
    2.8.2.2 Configuring a Shared JDK Capability
    2.8.2.3 Viewing your JDK Capabilities
  2.8.3 Configuring a new Custom Capability
    2.8.3.1 Configuring an Agent-specific Custom Capability
    2.8.3.2 Configuring a Shared Custom Capability
  2.8.4 Configuring a new Perforce Capability
  2.8.5 Editing a Capability
  2.8.6 Renaming a Capability
  2.8.7 Deleting a Capability
  2.8.8 Viewing the Agents & Plans Related to a Capability

03. Triggering a Build

  3.1 About Build Triggering
  3.2 Triggering a Build when Code is Updated
    3.2.1 Polling the Repository for Code Changes
    3.2.2 Triggering a Build on Code Check-in
  3.3 Triggering a Build on Schedule
    3.3.1 Scheduling a Single Daily Build
    3.3.2 Specifying a Cron-based Schedule
  3.4 Triggering a Build when another Build finishes
  3.5 Triggering a Build Manually

04. Configuring Email and Instant Messaging Notifications

  4.1 Enabling or disabling Notifications for a Plan
  4.2 Configuring Bamboo to send SMTP Email
  4.3 Configuring Bamboo to use Instant Messaging (IM)
    4.3.1 Configuring Bamboo to use Google Talk for Instant Messaging

05. Managing Users and Security

  5.01 Creating a User
  5.02 Changing a User’s Password or Details
  5.03 Deleting or deactivating a User
  5.04 Creating a Group
  5.05 Deleting a Group
  5.06 Adding Users to and removing them from Groups
  5.07 Granting Plan Permissions to Users or Groups
  5.08 Granting Global Permissions to Users or Groups
  5.09 Granting Administration Rights to a User
  5.10 Allowing Anonymous Users to access Bamboo
  5.11 Enabling or disabling Contact Details Display
  5.12 Enabling or disabling Public Signup
  5.13 Working with External User Repositories
    5.13.1 Integrating Bamboo with Crowd
    5.13.2 Integrating Bamboo with LDAP
      5.13.2.1 Configuring the Caching of your LDAP Repository
06. Configuring Plugins

- 6.1 About Bamboo Plugins
- 6.2 Enabling the 'Auto-Favourites' Plugin
- 6.3 Installing a new Plugin

07. Managing Data and Backups

- 7.1 Locating Important Directories and Files
- 7.2 Specifying Bamboo's Working Directory
- 7.3 Viewing your Database Connection Details
- 7.4 Moving your Bamboo Data to a different Database
- 7.5 Optimising or Re-indexing Data
- 7.6 Enabling Expiry of Build Results
- 7.7 Specifying a Backup Schedule
- 7.8 Exporting Data for Backup
- 7.9 Importing Data from Backup

08. Configuring System Settings

- 8.1 Viewing Bamboo's System Information
- 8.2 Updating your Bamboo License Details
- 8.3 Specifying Bamboo's Title
- 8.4 Specifying Bamboo's URL
- 8.5 Enabling GZIP Compression
- 8.6 Enabling Bamboo's Remote API
- 8.7 Enabling JIRA integration

09. Bamboo Security

- Bamboo Security Advisory 2008-02-08 (Bamboo 2.0 Beta)
- Securing your Remote Agents

Appendix A. Embedding Bamboo into Other Applications

- Javascript Widgets
  - All Plans & My Favourite Plans
  - Latest Builds
  - Latest Status of a Plan
  - My Latest Changes
  - Plan Summary Graphs
1. Configuring Projects and Plans

- 1.1 About Projects, Plans and Builds
- 1.2 Creating a Plan
  - 1.2.1 Specifying a Plan's Details
  - 1.2.2 Specifying a Plan's Source Repository
    - CVS
    - Perforce
    - Subversion
  - 1.2.3 Specifying a Plan's Builder
    - Defining Global Variables
    - Using Global or Build-specific Variables
  - 1.2.4 Specifying a Plan's Capability Requirements
  - 1.2.5 Specifying a Plan's Build Artifacts
  - 1.2.6 Specifying a Plan's Notifications
  - 1.2.7 Specifying a Plan's Post Actions
    - Specifying Expiry for a Plan's Build Results
    - Specifying Labels for a Plan's Build Results
  - 1.2.8 Specifying a Plan's Permissions
- 1.3 Editing a Plan
- 1.4 Disabling or deleting a Plan
  - 1.4.1 Deleting a Build Result
  - 1.4.2 Deleting a Plan's Working Files
  - 1.4.3 Stopping an Active Build
- 1.5 Moving a Plan to a different Project
- 1.6 Renaming a Plan or Project
1.1 About Projects, Plans and Builds

This page last changed on Dec 12, 2007 by alui.

• A Bamboo plan (or build plan) is the "recipe" for a build.

A plan defines: what gets built (i.e. the source-code repository); how the build is triggered; which builder to use; which agent capabilities are required for the build; what artifacts the build will produce; what tests to run; who will be notified of the build result; any labels with which the build result or build artifacts will be tagged; and who has permission to view and perform various actions on a plan and its build results.

Every plan belongs to a project.

• A project enables easy identification of plans that are logically related to each other, which is useful for instance when generating reports across multiple plans.

Each project has a Name (e.g. "CRM System") and a Key (e.g. "CRM"). The Project Key is prefixed to the relevant Plan Keys, e.g. the "CRM" project could have plans "CRM-TRUNK" and "CRM-BRANCH".

Note that creating a new project only requires defining the Project Name and Project Key, which is (optionally) done as part of the process of creating a new plan.

• A build is one execution of a plan.

Every build has a Build Number, which is appended to the relevant Plan Key to form the Build Key. For example, if a plan with the key "CRM-BRANCH" is executed for the seventeenth time, the build key will be "CRM-BRANCH-17".

Each plan's build results are stored in a subdirectory under your 'Build Directory' (see 7.1 Locating Important Directories and Files).
1.2 Creating a Plan

Only people with the 'Create' global permission can create a new build plan.

There are two ways of creating a new plan:

To create a new plan,

1. Click the 'Create Plan' link in the top navigation bar.
2. The 'Create a new plan' wizard will appear (see screenshot below). Click the nine sub-tabs to edit the following:
   - '1. Plan Details'
   - '2. Source Repository'
   - '3. Builder Configuration'
   - '4. Capability Requirements'
   - '5. Build Artifacts' (optional; can be completed now or later)
   - '6. Build Notifications' (optional; can be completed now or later)
   - '7. Post Actions' (optional; can be completed now or later)
   - '8. Permissions' (optional; can be completed now or later)
   - '9. Summary'
3. When you return to the Dashboard, your new plan (and new project, if applicable) will be displayed in the 'All Projects' list.

To copy an existing plan,

1. Click the 'Create Plan' link in the top navigation bar.
2. On the 'Plan Details' screen, select the check-box 'Clone an existing build plan?'
3. A list called 'Plan to clone' will be displayed, containing all existing plans for which you have the 'Clone' and/or 'Admin' plan permission. Select the plan you wish to copy.
4. Enter the required information in the screens listed above. On screens 2-8, appropriate information will be copied from the plan you selected; but you will need to complete all fields on screen 1.

Screenshot: 'Create a new plan' wizard — Page 1. 'Plan Details'
1.2.1 Specifying a Plan's Details

When creating a new plan, the first step is to specify the plan's details and the project to which it will belong.

A project is a collection of plans.

A project enables easy identification of plans that are logically related to each other, which is useful for instance when generating reports across multiple plans. Each project has a Name (e.g. "CRM System") and a Key (e.g. "CRM"). The Project Key is prefixed to the relevant Plan Keys, e.g. the "CRM" project could have plans "CRM-TRUNK" and "CRM-BRANCH".

To specify a plan's details,

1. Project — When you create a new plan, you can either add it to an existing project or create a new project. Either:
   • Select the appropriate project from the drop-down list;
   or:
   • Select 'New Project' and complete the following two fields:
     a. Project Name — Type a descriptive name (e.g. 'Issue Tracking Application') that will identify your project on the Dashboard and in reports.
     b. Project Key — Type a logical contraction of the Project Name (e.g. 'ITA'). The Project Key will be included in the plan's Build Results keys (e.g. 'ITA-MAIN-179'), so you may want to make it no longer than 3 or 4 characters. The Project Key must be unique within your Bamboo system.

2. Build Plan Name — Type a name that will identify the plan within its project (e.g. 'Main Build', 'Branch', 'Unit Tests', 'Acceptance Tests'). Note that the Build Plan Name, which is displayed throughout Bamboo, is always accompanied by its Project Name.

3. Build Plan Key — Type a logical contraction of the Build Plan Name. The Build Plan Key (e.g. 'MAIN') will be included in the plan's Build Results keys (e.g. 'ITA-MAIN-179'), so you may want to make it no longer than 3 or 4 characters.
   Note that the Build Plan Key only has to be unique within the project, that is, you could have a 'MAIN' plan in lots of different projects.

4. Click the 'Next' button to go to 1.2.2 Specifying a Plan's Source Repository.

Screenshot: 'Plan Details'

[Image of the Plan Details screen]
1.2.2 Specifying a Plan's Source Repository

For each plan, you need to specify what type of source-code repository the plan will use, where the repository is located, and what type of build strategy the plan will use. Please see:

- **CVS**
- **Perforce**
- **Subversion**

CVS, Subversion and Perforce are supported out-of-the-box. If you need to use a different type of repository, you can write a Source Repository Module plugin to enable Bamboo to connect to your repository.

Screenshot: 'Source Repository --- CVS'
For each plan, you need to specify what type of source-code repository the plan will use, where the repository is located, and what type of build strategy the plan will use.

To specify a CVS repository,

If you are creating a new plan, start at step 5.

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

4. The 'Configuration' tab will be displayed. Click the 'Source Repository' sub-tab.
5. The 'Source Repository' sub-tab will be displayed (see screenshot below). Complete the fields as follows:

CVS configuration

- 'Repository' — select 'CVS'.
- 'CVS Root' — Type the full path to your CVS repository root (e.g. ':pserver:me@cvs.atlassian.com:/cvsroot/atlassian'). Bamboo supports pserver, ext (ssh) and local repository access methods. Note that you can use global variables in this field (see Using Global or Build-specific Variables).
- 'Authentication Type' — Select either 'Password' or 'SSH'.
  - If you select 'Password', the following fields will appear:
    - 'Password' — (Optional) Type the password for your CVS repository.
    - 'Change Password' — (Will only appear after you have saved the plan) Select this check-box if you want to change the password that is used to access the CVS repository.
  - If you select 'SSH', the following fields will appear:
    - 'Private Key' — Type the absolute path of your SSH private key.
    - 'Passphrase' — Type the passphrase for your SSH private key.
    - 'Change Passphrase' — (Will only appear after you have saved the plan) Select this check-box if you want to change the password for your SSH private key.
- 'Quiet Period' — This setting is used to avoid starting a build while someone is in mid-checkin. Bamboo will only initiate a build for this plan when no more changes are detected within the Quiet Period following the last known change. Type the number of seconds Bamboo should wait. Please note that this parameter is mandatory for CVS, as CVS allows partial checkouts.
- 'Module' — Type the name of the CVS module that contains the source-code.
  - Currently Bamboo has limited support for CVS ampersand modules. To use an ampersand module, you will need to define a regular module with the same name as the ampersand module (since Bamboo expects there to be a directory with the specified checkout module name). For example:
    1. Create a module (e.g. allbuilds).
    2. Define an ampersand module with the same name. (The ampersand module can be empty.)
    3. In the 'Module' field, enter the following: allbuilds allbuilds &project2 &project2 &project3
- 'Version of Module' — Select either 'HEAD' or 'Branch/Tag'. If you select 'Branch/Tag', the following field will appear:
  - 'Branch name' — Type the relevant branch name or tag. Note that you can use global variables in this field (see Using Global or Build-specific Variables).

Common Repository Configuration

- 'Force Clean Build' (Optional) — You can force Bamboo to remove the source directory and check it out again prior to each build by selecting this option. Please note that this will greatly increase the time it takes to complete a build.
- 'Include/Exclude Files' — (Optional) You can specify a particular inclusion or exclusion pattern for file changes to be detected.
- 'File Pattern' — (Optional) The regular expression for file changes which you wish to include/exclude.
• 'Web Repository URL' --- (Optional) You can specify the URL of the plan's browsable repository. If you specify a Web Repository URL, then links to relevant files will be displayed in the 'Code Changes' section of a build result.
• 'Web Repository Module' — (Optional) The plan's repository name, if the above Web Repository URL points to multiple repositories.
• 'Build Strategy' — The default value, 'Poll the repository for changes', is a convenient option that requires no additional configuration. A number of other options are available; for details, please see 03. Triggering a Build. You can change the Build Strategy over time as required. The rest of the fields on this tab will vary depending on which Build Strategy you select.
• Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to 1.2.3 Specifying a Plan's Builder.

Screenshot: 'Source Repository'
For each plan, you need to specify what type of source-code repository the plan will use, where the repository is located, and what type of build strategy the plan will use.

If you wish to build plans on your server and remote agents using a Perforce repository, you need to specify the location of the Perforce P4 client application for your server and each remote agent using Perforce. These locations are set by specifying a mandatory local server Perforce capability for your server and agent-specific remote Perforce capabilities for each of your remote agents using Perforce. You will not be able to create plans that use a Perforce repository without specifying the shared local Perforce capability first.

Read more about configuring a Perforce capability.

To specify a Perforce repository,

If you are creating a new plan, start at step 5.

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

4. The 'Configuration' tab will be displayed. Click the 'Source Repository' sub-tab.
5. The 'Source Repository' sub-tab will be displayed. Complete the fields as follows:

Perforce Configuration

- 'Repository' — select 'Perforce'.
- 'Port' — Type either the port to which the Perforce client will connect, or the Perforce server itself. This is the Perforce P4PORT environment variable that tells Bamboo which p4d (Perforce server) to use.
- 'Client' — The name of the Perforce Client Workspace which Bamboo will use. The Client Workspace determines which portions of the depot are visible in your Workspace Tree.
- 'Depot' — The client view of the depot that contains the source-code files for this plan. This is typically in the form //<clientname>/<workspace_mapping>/... For details please see the Perforce User's Guide.
  - To build to a particular Perforce label, enter the following: //Repo/Project/dev/...@mylabel (where 'mylabel' is the label you wish to match).
  - Bamboo sets the client root to its working directory, which means that code will be checked out to the 'working directory/<workspace_mapping>' location. Please take note of this, when specifying the 'Artifact Copy Pattern' for your Build Artifacts.
- 'Username' — (Optional) The Perforce username that Bamboo will use when it accesses the server ('Port'). Leave this field blank if you want Bamboo to use the default Perforce user (i.e. the OS username).
- 'Password' — (Optional) Type the password required by the Perforce username (if applicable).
- 'Change Password' — (This field will only appear after you have saved the plan) Select this checkbox if you want to change the password that is used to access the Perforce repository.
- 'Let Bamboo manage your workspace' — This field indicates whether you want Bamboo to manage your workspace or not.

Please be aware of the following implications of letting/preventing Bamboo from managing your workspace:

If you let Bamboo manage your workspace,

- We recommend this configuration if your plans will be running on many different machines or different operating systems, as Bamboo sets the client root for you.
- Bamboo will make configuration changes to the Client Workspace to manage builds (e.g. Bamboo will modify the host and root). You need to ensure that you enter a Client Workspace in the 'Client' field that will be used solely for Bamboo.
Under this configuration, you should configure one client per plan to avoid conflicts when updating the client root.

If you do not let Bamboo manage your workspace,

- We recommend this configuration if you wish to reuse your client for several plans, as Bamboo will retrieve the client root directory from Perforce and use it to run builds.
- Under this configuration, you need to ensure that the client root directory exists on all machines that the plan will be built on.
- Please note that alternate roots does not currently work in Bamboo. See issue BAM-2377 for further details.

- 'Enable Quiet Period:' — Select this setting to set Quiet Period parameters for the build plan.
- 'Quiet Period:' — This field will only display if 'Enable Quiet Period:' has been selected. This setting is used to avoid starting a build while someone is in mid-checkin. Bamboo will only initiate a build for this plan when no more changes are detected within the Quiet Period following the last known change. Type the number of seconds Bamboo should wait.
- 'Maximum Retries:' — This field will only display if 'Enable Quiet Period:' has been selected. You can specify how many times Bamboo should check for new changes using the Quiet Period parameter before initiating a build. For example, if you have set the 'Quiet Period' to '10' seconds then Bamboo will check if a checkout has occurred in the last 10 seconds. If you have then specified 'Maximum Retries:' as '5', then Bamboo will perform this check five times before initiating the build, regardless of any activity during the Quiet Period of the last check.

Common Repository Configuration

- 'Force Clean Build' (Optional) — You can force Bamboo to remove the source directory and check it out again prior to each build by selecting this option. Please note that this will greatly increase the time it takes to complete a build.
- 'Include/Exclude Files' — (Optional) You can specify a particular inclusion or exclusion pattern for file changes to be detected.
- 'File Pattern' — (Optional) The regular expression for file changes which you wish to include/exclude.
- 'Web Repository URL' --- (Optional) You can specify the URL of the plan's browsable repository. If you specify a Web Repository URL, then links to relevant files will be displayed in the 'Code Changes' section of a build result.
- 'Web Repository Module' — (Optional) The plan's repository name, if the above Web Repository URL points to multiple repositories.
- 'Build Strategy' — The default value, 'Poll the repository for changes', is a convenient option that requires no additional configuration. A number of other options are available; for details, please see 03. Triggering a Build. You can change the Build Strategy over time as required. The rest of the fields on this tab will vary depending on which Build Strategy you select.
- Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to 1.2.3 Specifying a Plan's Builder.

Screenshot: 'Source Repository --- Perforce'
For each plan, you need to specify what type of source-code repository the plan will use, where the repository is located, and what type of build strategy the plan will use.

To specify a Subversion repository,

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon: 
4. The 'Configuration' tab will be displayed. Click the 'Source Repository' sub-tab.
5. The 'Source Repository' sub-tab will be displayed. Complete the fields as follows:

Subversion Configuration

- 'Repository' — select 'Subversion'.
- 'Repository URL' — The location of your Subversion repository (e.g. http://svn.collab.net/repos/svn/trunk). Note that you can use global variables in this field (see Using Global or Build-specific Variables).
- 'Username' — (Optional) The Subversion username (if any) required to access the repository.
- 'Authentication Type' — Select either 'Password' or 'SSH'.
  - If you select 'Password', the following fields will appear:
    - 'Password' — (Optional) Type the password required by the Subversion username (if applicable).
    - 'Change Password' — (Will only appear after you have saved the plan) Select this checkbox if you want to change the password that is used to access the Subversion repository.
  - If you select 'SSH', the following fields will appear:
    - 'Private Key' — Type the absolute path of your SSH private key.
    - 'Passphrase' — Type the passphrase for your SSH private key.
    - 'Change Passphrase' — (Will only appear after you have saved the plan) Select this checkbox if you want to change the password for your SSH private key.

Advanced Options
(The following fields will only display if you select the 'Enable Advanced Options' check-box.)

- 'Enable Quiet Period:' — Select this setting to set Quiet Period parameters for the build plan.
- 'Quiet Period:' — This field will only display if 'Enable Quiet Period:' has been selected. This setting is used to avoid starting a build while someone is in mid-checkin. Bamboo will only initiate a build for this plan when no more changes are detected within the Quiet Period following the last known change. Type the number of seconds Bamboo should wait.
- 'Maximum Retries:' — This field will only display if 'Enable Quiet Period:' has been selected. You can specify how many times Bamboo should check for new changes using the Quiet Period parameter before initiating a build. For example, if you have set the 'Quiet Period' to '10' seconds then Bamboo will check if a checkout has occurred in the last 10 seconds. If you have then specified 'Maximum Retries:' as '5', then Bamboo will perform this check five times before initiating the build, regardless of any activity during the Quiet Period of the last check.
- 'Detect Changes in Externals:' — (Optional) Select this check-box if your Subversion repository uses svn:externals to link to other repositories (note that your externals must be in the root of the checkout directory, not in a subdirectory). Please note that you only need to select this check box if you require Bamboo to detect changes in the externals. If your externals references a particular (static) revision, you do not need to check this box.

Common Repository Configuration

- 'Force Clean Build' (Optional) — You can force Bamboo to remove the source directory and check it out again prior to each build by selecting this option. Please note that this will greatly increase the time it takes to complete a build.
- 'Include/Exclude Files' — (Optional) You can specify a particular inclusion or exclusion pattern for file changes to be detected.
- 'File Pattern' — (Optional) The **regular expression** for file changes which you wish to include/exclude.
- 'Web Repository URL' --- (Optional) You can specify the URL of the plan's browsable repository. If you specify a Web Repository URL, then links to relevant files will be displayed in the 'Code Changes' section of a build result.
- 'Web Repository Module' — (Optional) The plan's repository name, if the above Web Repository URL points to multiple repositories.
- 'Build Strategy' — The default value, 'Poll the repository for changes', is a convenient option that requires no additional configuration. A number of other options are available; for details, please see [03. Triggering a Build](#). You can change the Build Strategy over time as required. The rest of the fields on this tab will vary depending on which Build Strategy you select.
- Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to [1.2.3 Specifying a Plan's Builder](#).

**Screenshot: 'Source Repository --- SVN'**

![Source Repository](image-url)
1.2.3 Specifying a Plan's Builder

When you configure a plan, you need to specify which builder should be used for the plan's builds. If you specify an Ant or Maven builder, you will also need to choose a JDK.

At least one builder and one JDK were configured automatically when you installed Bamboo. You can add more builders of different types as described in 2.8.1 Configuring a new Builder, and you can add more JDKs as described in 2.8.2 Configuring a new JDK. They will then appear in the 'Builder' drop-down list and the 'Build JDK' drop-down list as described below.

To specify a plan's builder and build resources,

1. If you are creating a new plan, start at step 5.
2. Click 'Home' to go to the Dashboard.
3. Click the 'All Plans' tab.
4. Locate the plan in the list and click this icon:
5. The 'Configuration' tab will be displayed. Click the 'Builder Configuration' sub-tab.
6. In the 'Builder' field, select the builder which Bamboo will use to build this plan.
7. The following fields are the ones that will vary depending on what type of Builder you select:
   - **Ant**:
     1. 'Build File' — Type the relevant filename (e.g. build.xml). You can include variables (see Using Global or Build-specific Variables).
     2. 'Target' — Specify the Ant target that you want Bamboo to execute each time the source code changes. For example: test (this will run the Ant target 'test'). You can also use '-D' to define one or more JVM parameters, e.g.: -Djava.awt.headless=true (this will pass the parameter 'java.awt.headless' with a value of 'true'). You can also include variables (see Using Global or Build-specific Variables).
     3. 'Build JDK' — Choose a JDK from the list.
   - **Maven**:
     1. 'Goal' — Specify the Maven goal that you want Bamboo to execute each time the source code changes. For example: clean test (this will run the Maven goal 'clean' followed by the Maven goal 'test'). You can also use '-D' to define one or more JVM parameters, e.g.: -Djava.awt.headless=true (this will pass the parameter 'java.awt.headless' with a value of 'true'). You can also include variables (see Using Global or Build-specific Variables).
     2. 'Build JDK' — Choose a JDK from the list.
   - **NAnt**:
     1. 'Build File' — Type the relevant filename (e.g. default.build). You can include variables (see Using Global or Build-specific Variables).
     2. 'Target' — Specify the NAnt target that you want Bamboo to execute each time the source code changes. For example: RegexDemo/RegexDemo.sln. You can also include variables (see Using Global or Build-specific Variables).
     3. 'Options' — Specify the NAnt command line options that you want to include (e.g. You can also include variables (see Using Global or Build-specific Variables).
   - **devenv.com**:
     1. 'Solution' — Type the name of the Visual Studio solution file that you want Bamboo to execute each time the source code changes. For example: run. You can also include variables (see Using Global or Build-specific Variables).
- 'Options' — Specify any devenv command line options that you want to include (e.g. /build Debug). You can also include variables (see Using Global or Build-specific Variables).

- Custom command:
  - 'Argument' — Specify the relevant argument to pass to the command. Note that arguments which contain spaces must be quoted. You can include variables (see Using Global or Build-specific Variables).

- Script:
  - 'Script' — Specify the location of the script file. This can be either relative to the repository root of the plan, or absolute. You can include variables (see Using Global or Build-specific Variables).
  - 'Argument' — Specify the relevant argument to pass to the script. Note that arguments which contain spaces must be quoted. You can include variables (see Using Global or Build-specific Variables).

8. 'System Environment Variables' — (Optional) Specify any additional* operating system environment variables you want to pass to your build (e.g. ANT_HOME=/tools/ant); use spaces to separate multiple variables. You can also include Bamboo global or build-specific variables (see Using Global or Build-specific Variables).

* i.e. additional to the existing environment variables (see 8.1 Viewing Bamboo's System Information for a list). Note that existing environment variables are automatically available to the builder, i.e. you don't need to specify them in the 'System Environment Variables' field.

9. 'Working Sub Directory' — (Optional) If you leave this field blank, Bamboo will look for the build files in the build root directory (which is assumed to be the build's Working Directory, as described in 7.1 Locating Important Directories and Files). You can override this option by specifying an alternative working directory (which must be a subdirectory of the root directory). For example, if your plan has a build script in a subdirectory, and the script needs to be run from within that subdirectory, you would type the name of that subdirectory in the 'Working Sub Directory' field.

10. 'The build has tests' — Select this check-box if you want Bamboo to gather test results data for each build result. (Note that Bamboo requires test results to be XML files that are compatible with JUnit XML format, unless you are using an NAnt builder, in which case the test results must be in NUnit XML format.) Choose one of the following:

- 'Test Results Directory' — Select this option if Bamboo should look in the Builder's standard test results directory.
- 'Specify custom results directories' — Select this option if the Builder will place generated test results in an alternative directory. The following field will appear:
  - 'Specify custom results directories' — Type the name of the test results directory (or multiple directories, separated by commas). You can also use Ant-style patterns such as */test-reports/*.xml. Please specify file path relative to your plan's root directory (e.g. /home/bamboouser/bamboo-home/xml-data/build-dir/MY_PLAN/), i.e. please do not specify an absolute path.

11. 'Clover output will be produced' — Select this check-box if you are running Atlassian Clover and want to view its code-coverage data from within Bamboo (see 3.7 Viewing the Clover Code-Coverage for a Build Result). The following field will be displayed:

- 'Clover XML Directory' — Specify the name of the directory (including path) where Bamboo will look for the XML report output file from Clover. Please specify file path relative to your plan's root directory (e.g. /home/bamboouser/bamboo-home/xml-data/build-dir/MY_PLAN/), i.e. please do not specify an absolute path.

12. Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to 1.2.4 Specifying a Plan's Capability Requirements.

Screenshot: 'Builder Configuration'
### Builder Configuration

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal</strong></td>
<td>clean test Obtained from user (set bamboo build number)</td>
</tr>
<tr>
<td><strong>Build JDK</strong></td>
<td>JDK 1.5</td>
</tr>
<tr>
<td><strong>System Environment Variables</strong></td>
<td>Optional: Any variable you want to pass to your build.</td>
</tr>
<tr>
<td><strong>Working Sub Directory</strong></td>
<td>Optional: Any additional directories you want to pass to your build.</td>
</tr>
</tbody>
</table>

**Where should Bamboo look for the test result files?**

- [ ] Test Results Directory: Log to the standard test results directory.
- [ ] Specify custom results directories. Where are the Bamboo tests located?

**Where should Bamboo look for the Oracle code-coverage output?**

- [ ] Oracle XML Directory: This is the path where Oracle code-coverage output will be stored. For more information visit Oracle Code Coverage.


Defining Global Variables

When configuring a plan, you may want to specify variables to be used in the build process. For details please see Using Global or Build-specific Variables.

Global variables are one type of variable that is available to you. Global variables are defined across your entire Bamboo instance, and have the same value for every plan that is built by Bamboo.

To add a new global variable,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Global Variables' link in the left navigation column.
3. This will display a list of variables that have been configured in Bamboo, with an empty field at the bottom of each column for adding a new variable.
4. In the 'Key' field, type the key that you will use to identify the variable.
5. In the 'Value' field, type the value of the variable.
6. Click the 'Save' button.

Screenshot: Global Variables

You can use this page to view, add and delete global variables. Global variables are available on every build run in Bamboo.

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>bamboo/Version</td>
<td>1.1</td>
<td>Save</td>
</tr>
</tbody>
</table>
Specifying Global or Build-specific Variables

When configuring a plan, you may want to specify variables to be used in the build process. There are two types of variables available to you:

- **Global variables** are defined across your entire Bamboo instance, and have the same (static) value for every plan that is built by Bamboo. You can define as many global variables as you wish (see [Defining Global Variables](#)).
- **Build-specific variables** are evaluated by Bamboo dynamically at build time. The source of a build-specific variable can either be a Bamboo property or one of the default plugins (assuming they have been enabled). The following build-specific variables are available by default:

<table>
<thead>
<tr>
<th>Build-specific variable</th>
<th>Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>buildKey</td>
<td>Bamboo property</td>
<td>The plan key for the build, e.g. BAM-MAIN</td>
</tr>
<tr>
<td>buildNumber</td>
<td>Bamboo property</td>
<td>The Bamboo build number, e.g. 123</td>
</tr>
<tr>
<td>custom.svn.revision.number</td>
<td>Plugin</td>
<td>(For Subversion only) The revision number</td>
</tr>
<tr>
<td>custom.cvs.last.update.time</td>
<td>Plugin</td>
<td>(For CVS only) The last updated timestamp</td>
</tr>
<tr>
<td>custom.p4.revision.number</td>
<td>Plugin</td>
<td>(For Perforce only) The change set number</td>
</tr>
</tbody>
</table>

The usage format for all global and build-specific variables is:

```text
${bamboo.<variable>}
```

Where can I use Global and Build-specific Variables?

Variables can be used in the following fields of your build plan:

- **Goal** (for Maven builders only)
  - see [1.2.3 Specifying a Plan's Builder](#)
- **Build File** (for Ant and NAnt builders only)
  - see [1.2.3 Specifying a Plan's Builder](#)
- **Target** (for Ant and NAnt builders only)
  - see [1.2.3 Specifying a Plan's Builder](#)
- **Options** (for NAnt builders only)
  - see [1.2.3 Specifying a Plan's Builder](#)
- **Script** (for Scripts only)
  - see [1.2.3 Specifying a Plan's Builder](#)
- **Argument** (for Scripts and Custom Commands only)
  - see [1.2.3 Specifying a Plan's Builder](#)
- **System Environment Variables**
  - see [1.2.3 Specifying a Plan's Builder](#)
- **Repository URL** (for Subversion repositories only)
  - see [1.2.2 Specifying a Plan's Source Repository](#)
- **CVS Root** (for CVS repositories only)
  - see [1.2.2 Specifying a Plan's Source Repository](#)
- **Branch name** (for CVS repositories only)
  - see [1.2.2 Specifying a Plan's Source Repository](#)

Maven example
For example, you may want your Maven 2 version to be determined by Bamboo. In Maven 2 pom.xml you may have:

```xml
<groupId>com.atlassian.boo</groupId>
<artifactId>boo-test</artifactId>
<packaging>jar</packaging>
<version>1.1.$(env.bambooBuildNumber)-SNAPSHOT</version>
```

You can then specify the following in the 'Goal' field of your build plan:

`clean package -DbambooBuildNumber=${bamboo.buildNumber}`

When the command runs, Bamboo will replace the `buildNumber` with the actual number (e.g. 1102), which will be passed to the underlying Maven build to use. The command will then produce a jar that looks like this: `boo-test-1.1.1102-SNAPSHOT.jar`.

### Using Capabilities as Variables

You can also specify a capability to be used in a similar way to a global variable.

The format of the capability should be as follows:

`${bamboo.capability.<capability_key>}`

For example,

- **Custom**
  ```
  `${bamboo.capability.<capability_key>`
  ```

- **JDK**
  ```
  `${bamboo.capability.system.jdk.<jdk_label>}`
  ```

- **Builder**
  ```
  `${bamboo.capability.system.builder.<builder_type>.<builder_label>}`
  ```
  e.g. `${bamboo.capability.system.build.maven.Maven}`

- **Perforce**
  ```
  `${bamboo.capability.system.p4Executable}`
  ```

If you click on a capability, the specific capability key will be contained in the URL.

Please note, the space characters in the URL will be replaced with '+' characters. We recommend that you do not use capability labels with space characters, if you wish to use them as variables. A possible solution for space characters is to format them with '${}' symbols, however, this does not work in all cases.

### What Capabilities Can I Use Where?

Global and Build-Specific Variables can be used in a specific fields of your build plan, as specified above. For capabilities,
• System Capabilities are available to all of these fields, (i.e. global and build-specific).
• Agent Capabilities (i.e. agent-specific and shared/server capabilities) are available only to the build-specific fields. (i.e. not available to Repository URL, CVS Root or Branch name.)

For example,

If you wanted to specify a system variable, but have it set to different values on each agent, do the following:

1. Set the following as a system environment variable field on the 'Builder' tab:

   $$\{\text{bamboo.capability.thatsystemvariable}\}$$

2. Specify the system environment variable as a custom capability on each of your agents, and set to the capability to the different values, as desired.
1.2.4 Specifying a Plan's Capability Requirements

A requirement is an agent capability required by a build plan. Together, capabilities and requirements control which agents can execute builds for particular plans. Each plan can only be built by agents whose capabilities meet the plan's requirements.

There are four types of capabilities in Bamboo:
- Builder capability — Every plan has one 'Builder' capability requirement.
- JDK capability — A plan has either one 'JDK' capability requirement or none, depending on the plan's 'Builder'.
- Custom capability — A plan can have multiple 'custom' capability requirements (or none).
- Perforce capability — A plan will have a 'Perforce' capability requirement automatically specified, if Perforce was selected as the source repository.

What are custom capabilities?

Custom capabilities can be used to control which build plans will be built by a particular agent. For example, if the builds for a particular plan should only run in a Windows environment, you could create a custom capability 'operating.system=WindowsXP' for the appropriate agent(s), and specify it as a requirement for this plan.

Note that before you can specify them in your build plan, you need to first define your custom capabilities in your Bamboo system (see 2.8.3 Configuring a new Custom Capability).

To specify a plan's Builder capability requirement,

Choose a value for the 'Builder' field on the plan's 'Builder Configuration' tab. Please see 1.2.3 Specifying a Plan's Builder for instructions.

To specify a plan's JDK capability requirement,

Choose a value for the 'Build JDK' field on the plan's 'Builder Configuration' tab. Please see 1.2.3 Specifying a Plan's Builder for instructions.

To specify a plan's custom capability requirements,

If you are creating a new plan, start at step 5.

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

4. The 'Configuration' tab will be displayed. Click the 'Plan Requirements' sub-tab.
5. The 'Plan Requirements' sub-tab will be displayed (see screenshot below), showing a list of all the plan's current capability requirements and a list of 'Capable Agents' (i.e. agents which meet the plan's requirements and are therefore capable of running a build for this plan).
6. If you have previously set up your custom requirement (e.g. on another project), you can select it from the 'Requirement' dropdown in the 'Add Extra Requirements' section and continue from step 9.
7. If you are setting up a new custom requirement, select 'New custom requirement' instead and complete the following steps.
8. In the 'Add Extra Requirement' section, in the 'Key' field, type the key of the capability which you want this plan to require.
9. In the 'Add Extra Requirement' section, in the drop-down box below the 'Key' field, select one of the following:
   - 'exists' — this plan can be built by any agent that has a custom capability with the same Key as the one you have just specified.
• 'equals' — this will display a text field for you to enter an exact value. This plan can be built by any agent that has a custom capability with the same Key and value.

• 'matches' — this will display a text field for you to enter a regular expression (for more information about regular expressions, please visit http://java.sun.com/j2se/1.4.2/docs/api/java/util/regex/Pattern.html#sum). This plan can be built by any agent that has a custom capability with the same Key as the one you have just specified, where the value matches this regular expression.

9. Click the 'Add' button to add your new custom capability to the plan's list of requirements.

   - The displayed list of 'Capable Agents' will be updated, as the plan can now only be built by agents which have a custom capability that meets the new custom requirement you have specified.

10. Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to 1.2.5 Specifying a Plan's Build Artifacts or click the 'Summarise' button and go to 1.2.9 Completing and Enabling a New Plan (hidden).

Screenshot: 'Edit Capability Requirements'

The 'Capability Requirements' tab (see screenshot above) shows all of a plan's capability requirements: Builder, JDK and custom capabilities. It also shows which agents meet the requirements (i.e. are capable of building the plan).

- To see what capabilities a particular agent has, see 2.7.1 Viewing an Agent's Capabilities
1.2.5 Specifying a Plan's Build Artifacts

A plan's artifacts are any reports, websites or files (e.g. JAR files) which you wish to keep after each build. Artifacts are copied to a subdirectory (/PLAN_KEY/download_data/) under your 'Build Directory' folder (see 7.1 Locating Important Directories and Files). Artifacts which you define in the plan are listed in each build result as User-defined artifacts (see 3.4 Viewing a Build's Artifacts in the Bamboo User's Guide).

To specify a plan's build artifacts,

1. If you are creating a new plan, start at step 5.
2. Click 'Home' to go to the Dashboard.
3. Click the 'All Plans' tab.
4. Locate the plan in the list and click this icon:
5. The 'Configuration' tab will be displayed. Click the 'Build Artifacts' sub-tab.
6. In the 'Artifact Label' field, specify the name which Bamboo will use to describe the artifact.
7. In the 'Artifact Copy Pattern' field, specify the name (or Ant file copy pattern) of the artifact(s) you want to keep.
8. In the 'Source Directory' field, specify the directory (including path) where Bamboo will look for your artifact.
9. Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to 1.2.6 Specifying a Plan's Notifications.

For example, if you want to keep the latest version of a JAR you have built, you could specify Artifact Copy Pattern to be '*/*.jar' and the Source Directory to be 'target'.

Screenshot: 'Build Artifacts'
1.2.6 Specifying a Plan's Notifications

You can specify which people will receive notifications about build results for a particular plan, and under what circumstances (known as 'Notification Triggers'), i.e.:

<table>
<thead>
<tr>
<th>Notification Trigger</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>'All Completed Builds'</td>
<td>Bamboo will send a notification whenever a build finishes for this plan, regardless of the build result. This trigger is recommended for any plans for which it is critical that people are always informed about the latest build activity. Many organisations start with this trigger, then change it later as they get more confidence in the continuous build process.</td>
</tr>
<tr>
<td>'Failed Builds And First Successful'</td>
<td>This trigger is generally suitable for the majority of plans. Bamboo will send a notification whenever:</td>
</tr>
<tr>
<td></td>
<td>• a build fails for this plan.</td>
</tr>
<tr>
<td></td>
<td>• the plan is 'fixed' (that is, the plan's latest build is successful and the previous build failed).</td>
</tr>
<tr>
<td>'After X Failed Builds'</td>
<td>This trigger enables you to specify the 'Number Of Failed Builds' after which Bamboo will send a notification. This is a useful way of limiting the number of notifications, if you are concerned about people receiving too many.</td>
</tr>
</tbody>
</table>

For each plan, you can specify different recipients for each Notification Trigger. Note also that recipients need not be people with Bamboo user accounts.

⚠️ Before you begin
You need to configure Bamboo's SMTP email and/or instant messaging capabilities before Bamboo can send notifications.

To enable notifications for a plan,

💡 If you are creating a new plan, start at step 5.

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

4. The 'Configuration' tab will be displayed. Click the 'Build Notifications' sub-tab.
5. A list of 'Notification Triggers' will be displayed (see screenshot below). Select the 'Notification Trigger' you wish to enable, then specify any or all of the following recipients:
   • 'Roles' — Select from the following:
     ° 'Committer' — A committer is the Bamboo user(s) who committed code to a particular build (i.e. someone who committed code after the previous build was checked out by Bamboo).
     ° 'Watcher' — A plan’s watchers are the Bamboo users who have marked this plan as one of their favourites.
   • 'Groups' — Type the names of the appropriate Bamboo group(s), separated by commas.
   • 'Users' — Type the usernames of the appropriate Bamboo users, separated by commas; or click the following icon to select from a list of users:

   • 'Email Addresses' — This is useful if you need to send email notifications to people who are not Bamboo users. Type the appropriate email addresses, separated by commas.
   • 'Instant Messaging Addresses' — This is useful if you need to send IM notifications to people who are not Bamboo users. Type the appropriate IM addresses, separated by commas. Note that if you specify a broadcast address (eg. ‘project-x@broadcast.chat.mycompany.com’), Bamboo will not know the context of related IM responses.
6. Click the 'Add' button.
7. Repeat steps 5 and 6 until you have added all the Notification Triggers that you wish to enable for this plan.
8. Click the 'Done' button if you are editing an existing plan; or if you are creating a new plan, either click the 'Next' button and go to 1.2.7 Specifying a Plan's Post Actions or click the 'Summarise' button and go to 1.2.9 Completing and Enabling a New Plan (hidden).

Each Bamboo user can choose whether to receive their notifications via email, IM, both or neither.

Screenshot: 'Build Notifications'
1.2.7 Specifying a Plan's Post Actions

For each plan, you can optionally specify actions that will occur after ('post') each build:

- what labels (if any) will be automatically applied to the plan's build results. For details please see Specifying Labels for a Plan's Build Results.
- this plan's specific schedule for deletion of build results (if different to the default). For details please see Specifying Expiry for a Plan's Build Results.

Screenshot: 'Post Actions'
Specifying Expiry for a Plan's Build Results

By enabling build expiry, you can choose how much build results data will be kept in your Bamboo system, and for how long it will be kept (e.g. for reporting purposes), before being automatically deleted.

If you disable build expiry, your build results will never be automatically deleted from Bamboo.

You can enable/disable build expiry for:

- all plans (see 7.6 Enabling Expiry of Build Results). This is generally the easiest way to manage your build expiry. Your settings will apply to all plans that do not have individual expiry settings.
- individual plans (as described below). You would generally only do this if there is a specific reason to keep/delete a particular plan's build results.

Note that you can also delete build results manually — see 1.4.1 Deleting a Build Result.

⚠️ If you enable build expiry, ensure that you back up your build results data before its expiry date is reached.

To enable expiry for a plan's build results,

1. If you are creating a new plan, start at step 5.
   1. Click 'Home' to go to the Dashboard.
   2. Click the 'All Plans' tab.
   3. Locate the plan in the list and click this icon:
   4. The 'Configuration' tab will be displayed. Click the 'Post Actions' sub-tab.
   5. The 'Post Actions' tab will be displayed. Under 'Build Expiry', select 'Override global build expiry configuration'.
   6. In the 'What should be expired' section, select what type of build results data you want to delete:
      - 'Artifacts' — select this if you want to delete all user-defined artifacts but keep all other build results data.
      - 'Build results' — select this if you want to delete all build results data (including artifacts).
   7. Choose one of these three methods for specifying how much data to keep:
      - To keep all build results up to a certain age,
        - In the 'Expiry period' field, specify the number of months/weeks/days for which you want to keep your build results, e.g. specify '24 months' to keep all build results for the last two years.
        - In the 'Minimum builds to keep' field, specify '0'.
      - To keep a certain number of build results per plan,
        - In the 'Expiry period' field, specify '0'.
        - In the 'Minimum builds to keep' field, specify the number of build results you want to keep, e.g. specify '50' to keep the latest 50 build results for each plan.
      - To keep all build results up to a certain age, and a certain number of build results per plan,
        - In the 'Expiry period' field, specify the number of months/weeks/days for which you want to keep your build results, e.g. specify '24 months' to keep all build results for the last two years.
        - In the 'Minimum builds to keep' field, specify the number of build results you want to keep, e.g. specify '50' to keep the latest 50 build results for each plan. (This means that, even if all of a plan's builds are over two years old, the last 50 build results will not be deleted.)
   8. In the 'Labels to keep' field, specify any labels for which you always want to keep labelled builds. (If you want to specify more than one label, use spaces to separate them.) For any label(s) that you specify, all builds that have a matching label will never be deleted, regardless of the method you followed in step 7 above.

⚠️ Note: builds can either be labelled:
   - manually, as described in 8.2 Labelling a Build Result in the Bamboo User's Guide; or
• automatically, as described in [Specifying Labels for a Plan’s Build Results](#) in the Bamboo Administrator’s Guide.

9. Click the ‘Save’ button if you are editing an existing plan; or if you are creating a new plan, either click the 'Next' button and go to [1.2.8 Specifying a Plan’s Permissions](#) or click the 'Summarise' button and go to [1.2.9 Completing and Enabling a New Plan (hidden)](#).

### Screenshot: 'Post Actions'

To disable expiry for a plan’s build results data,

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

![Icon](image)

4. The 'Configuration' tab will be displayed. Click the 'Post Actions' sub-tab.
5. The 'Post Actions' tab will be displayed. Under 'Build Expiry', deselect 'Override global build expiry configuration'.
6. Click the 'Save' button.
Specifying Labels for a Plan's Build Results

A label is a convenient way to tag and group build results that are logically related to each other. Labels can also be used to define RSS feeds and to control build expiry.

Labels can be applied to build results automatically, by specifying the label(s) in a build plan (note that only Bamboo administrators can do this). Labels can also be applied ad hoc to build results by Bamboo users.

To specify labels for a plan's build results,

If you are creating a new plan, start at step 5.

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

4. The 'Configuration' tab will be displayed. Click the 'Post Actions' sub-tab.
5. The 'Post Actions' tab will be displayed (see screenshot below).
6. In the 'Regex Pattern' field, you can either:
   - Specify a regular expression on which to match the log files. Labels will be applied to the build if the regular expression finds a match (see the examples below).
   - Leave this field blank to label every build.
7. In the 'Labels' field, type the word (or multiple words, separated by commas and/or spaces) with which the plan's build results are to be labelled.
8. Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, either click the 'Next' button and go to 1.2.8 Specifying a Plan's Permissions or click the 'Summarise' button and go to 1.2.9 Completing and Enabling a New Plan (hidden).

Screenshot: 'Post Actions'

Regex example:

'There are \d+ results'

In the above regex, '\d+' represents any number with one or more digits. ('\d' means 'any digit', and '+' means 'one or more times'. When combined, they mean 'any sequence of one or more digits'.)

Therefore, positive matches would include:

- 'There are 0 results'
- 'There are 123 results'

Regex example with multiple labels:
You can use "capturing groups" with Bamboo 1.2.1 or later to create different labels for different purposes.

For example,

- Enter the following into the 'Regex Pattern' field:

```plaintext
( PERFORMANCE_IMPROVED | PERFORMANCE_DETERIORATED )
```

- Enter the following into the 'Labels' field:

```plaintext
\1
```

These settings will label your builds with PERFORMANCE_IMPROVED if "PERFORMANCE_IMPROVED" appears in the build log, and PERFORMANCE_DETERIORATED if "PERFORMANCE_DETERIORATED" appears in the build log. If both strings appear in a log, then both labels are applied to the build.
1.2.8 Specifying a Plan's Permissions

This page last changed on Apr 07, 2008 by alui.

A plan permission is the ability to perform a particular operation in relation to a build plan. For each plan, different permissions can be granted to particular groups and/or users. The following plan permissions are available:

<table>
<thead>
<tr>
<th>Plan permission</th>
<th>Description</th>
<th>Can be granted to</th>
</tr>
</thead>
<tbody>
<tr>
<td>'View'</td>
<td>Permission to:</td>
<td>- a particular user&lt;br&gt;- a particular group&lt;br&gt;- all logged-in users&lt;br&gt;- anonymous users²</td>
</tr>
<tr>
<td></td>
<td>• view this plan's build results&lt;br&gt;• add comments or labels to this plan's build results¹</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>🔴 People who don't have the 'View' permission will not know that the plan exists.</td>
<td></td>
</tr>
<tr>
<td>'Edit'</td>
<td>Permission to view and edit this plan's configuration, except for the plan's permissions.</td>
<td>- a particular user&lt;br&gt;- a particular group&lt;br&gt;- all logged-in users&lt;br&gt;- anonymous users²</td>
</tr>
<tr>
<td>'Build'</td>
<td>Permission to:</td>
<td>- a particular user&lt;br&gt;- a particular group&lt;br&gt;- all logged-in users&lt;br&gt;- anonymous users²</td>
</tr>
<tr>
<td></td>
<td>• manually start/stop a build for this plan.&lt;br&gt;• enable/disable this plan from submitting builds to the queue.</td>
<td></td>
</tr>
<tr>
<td>'Clone'</td>
<td>Permission to copy this plan when creating a new plan. (Note: only users with the 'Create Plan' global permission can create new plans.)</td>
<td>- a particular user&lt;br&gt;- a particular group</td>
</tr>
<tr>
<td>'Admin'</td>
<td>Permission to:</td>
<td>- a particular user&lt;br&gt;- a particular group</td>
</tr>
<tr>
<td></td>
<td>• edit this plan's permissions.&lt;br&gt;• delete this plan's build results and working files.</td>
<td></td>
</tr>
</tbody>
</table>

¹ Only logged-in users (not anonymous users) can label or comment on a build result.
² Anonymous users cannot access Bamboo at all unless they have been granted the 'Access' global permission. See 5.10 Allowing Anonymous Users to access Bamboo.

Any user with the 'Admin' global permission automatically has all plan permissions for every plan.

The processes for granting and revoking plan permissions are described below. Note that, for ongoing ease of management, it is recommended that you grant permissions to groups rather than to individual users.

To grant plan permissions to a user,

If you are creating a new plan, start at step 5.

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

4. The 'Configuration' tab will be displayed. Click the 'Permissions' tab (see screenshot below).
5. In the 'Grant permission to' list at the bottom of the screen, select 'User'.

Document generated by Confluence on Apr 14, 2008 01:44
6. Type the username into the box, or click the following icon to select from a list of users:

Note that the assignment of permissions to LDAP users and groups in Bamboo is case sensitive. For instance, if the username of the LDAP user is 'Bob', you will need to type in 'Bob' (not 'bob' or 'BOB').

7. Click the 'Add' button.

8. The user will be added to the list of users on the 'Permissions' tab. Select the check-box for each permission that you wish to grant to this user.

9. Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to 1.2.9 Completing and Enabling a New Plan (hidden).

To grant plan permissions to a group,

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

4. The 'Configuration' tab will be displayed. Click the 'Permissions' tab (see screenshot below).
5. In the 'Grant permission to' list at the bottom of the screen, select 'Group'.
6. Type the group name into the box.

Note that the assignment of permissions to LDAP users and groups in Bamboo is case sensitive. For instance, if the name of the LDAP group is 'Dev', you will need to type in 'Dev' (not 'dev' or 'DEV').

7. Click the 'Add' button.

8. The group will be added to the list of groups on the 'Permissions' tab. Select the check-box for each permission that you wish to grant to this group.

9. Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to 1.2.9 Completing and Enabling a New Plan (hidden).

To grant plan permissions to all Bamboo users,

1. Click creating a new plan, start at step 5.
2. Click 'Home' to go to the Dashboard.
3. Click the 'All Plans' tab.
4. Locate the plan in the list and click this icon:

5. The 'Configuration' tab will be displayed. Click the 'Permissions' tab (see screenshot below).
6. Select the check-box for each permission that you wish to grant to all Bamboo users.
7. Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to 1.2.9 Completing and Enabling a New Plan (hidden).

To grant plan permissions to anonymous users,

1. Click creating a new plan, start at step 5.
2. Click 'Home' to go to the Dashboard.
3. Click the 'All Plans' tab.
4. The 'Configuration' tab will be displayed. Click the 'Permissions' tab (see screenshot below).
5. Locate 'Anonymous users' (under 'Other').
6. Select the check-box for each permission that you wish to grant to anonymous users.
7. Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to 1.2.9 Completing and Enabling a New Plan (hidden).
5. Select the check-box for each permission that you wish to grant to all anonymous users.
6. Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to 1.2.9 Completing and Enabling a New Plan (hidden).

To revoke plan permissions,

If you are creating a new plan, start at step 5. Click 'Home' to go to the Dashboard.
1. Click the 'All Plans' tab.
2. Locate the plan in the list and click this icon:
   ![Icon](image)

3. The 'Configuration' tab will be displayed. Click the 'Permissions' tab (see screenshot below).
4. Locate the relevant user/group/all logged-in users/anonymous users.
5. Deselect the check-box for each permission that you wish to revoke from the user/group/all users/anonymous users.
   
   If you deselect all permissions for a user or group, that user or group will disappear from the 'Permissions' tab for this plan.
6. Click the 'Save' button if you are editing an existing plan; or if you are creating a new plan, click the 'Next' button and go to 1.2.9 Completing and Enabling a New Plan (hidden).

Screenshot: Plan Permissions
1.3 Editing a Plan

To edit an existing plan,

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click the 'Edit' icon:

4. The 'Configuration' tab will be displayed (see below). Click the nine sub-tabs to edit the following:
   - 'Plan Details' — see 1.6 Renaming a Plan or Project.
   - 'Source Repository' — see 1.2.2 Specifying a Plan's Source Repository.
   - 'Builder Configuration' — see 1.2.3 Specifying a Plan's Builder.
   - 'Capability Requirements' — see 1.2.4 Specifying a Plan's Capability Requirements.
   - 'Build Artifacts' — see 1.2.5 Specifying a Plan's Build Artifacts.
   - 'Build Notifications' — see 1.2.6 Specifying a Plan's Notifications.
   - 'Post Actions' — see Specifying Labels for a Plan's Build Results and Specifying Expiry for a Plan's Build Results.
   - 'Dependencies' — see 3.4 Triggering a Build when another Build finishes.
   - 'Permissions' — see 1.2.8 Specifying a Plan's Permissions.

Screenshot: 'Configuration'
1.4 Disabling or deleting a Plan

Sometimes, for example if a plan’s latest build is broken and cannot be fixed quickly, you might need to temporarily stop the plan from being built. You can achieve this by disabling the plan, which will prevent it from submitting builds to the queue.

If a plan is no longer relevant, you have the option to completely delete it from your Bamboo system. To do this you will require the 'Admin' global permission. Note that deleting a plan will also delete all of the plan’s build results, artifacts, labels and comments.

To delete a build that is currently in progress, see 1.4.3 Stopping an Active Build.

To disable a plan,

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click the plan name.
4. This will display the 'Plan Summary'. Click the 'Build Actions' link (at the right of the page) and select 'Disable Plan':

   ![Build Actions](image)

To delete a plan,

Before you begin
If you need to keep a permanent record of the plan's build results, see 7.8 Exporting Data for Backup.

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Remove Plans' link in the left navigation column.
3. This will display a list of all plans in your Bamboo system. Select the check-box for the plan you wish to delete.
4. Click the 'Delete' button at the bottom of the list.
5. You will be prompted to confirm your deletion.

Screenshot: 'Remove Plans'

Remove Plans

Check the plans to be removed

- [ ] When deleting plans from Bamboo, build history is unrecoverable for the deleted plans.

Check one or more of the following plans to be deleted:
- Select all plans and projects
- Bamboo (Stable)
- HEAD (MAIN)
- Stable Branch (STAB)
- LDAP Integration Tests (LDAP)
- Acceptance tests (SAF/UNC)
1.4.1 Deleting a Build Result

This page last changed on Dec 12, 2007 by alui.

If a build result is no longer relevant, you have the option to completely delete it from your Bamboo system. Note that you can also automatically delete build results that reach a particular age — see 7.6 Enabling Expiry of Build Results.

To delete a build that is currently in progress, first see 1.4.3 Stopping an Active Build.

To delete a build result,

1. Go to the build result's plan. There are two ways to do this:
   a. Click 'Home' to go to the Dashboard, then click the 'All Plans' tab. Locate the plan in the list and click the plan name.
   OR:
   b. From the build result, click the plan name.
2. This will display the 'Plan Summary'. Click the 'Completed Builds' tab.
3. This will display a list of all completed build results for this plan (see screenshot below). Locate the relevant build result and click the 'Delete' icon:
4. The build result, and any artifacts created by the build, will be deleted.

Only people with the 'Admin' global permission or the 'Admin' plan permission can delete build results.

Screenshot: Build Results for a Plan

- Successful builds are green, failed builds are red.
- The build browser builds 10 tests in parallel.
- The average build time for recent builds is approximately 2 seconds.
- Bamboo builds everything whenever the CVS source code repository changes.
- Click for all builds or just the latest build.
1.4.2 Deleting a Plan's Working Files

If you want to ensure a clean check-out of your source code when Bamboo runs the next build for a particular plan, you will need to delete the plan's current Working Files.

To delete a plan's working files,

1. Click 'Home' to go to the Dashboard, then click the 'All Plans' tab.
2. Locate the plan in the list and click the plan name.
3. Click the 'Files' tab.
4. A list of the plan's working files will be displayed. Scroll down to the bottom of the screen and click the 'Delete all build files' icon:

Only people with the 'Admin' global permission or the 'Admin' plan permission can delete Working Files.
1.4.3 Stopping an Active Build

If you are running Bamboo on Windows, it may only be possible to stop an active build by going to Windows Task Manager and ending the relevant processes.

To stop an active build,

1. Click 'Home' to go to the Dashboard.
2. Locate the relevant plan on the 'All Plans' tab
   OR:
   Locate the relevant plan on the 'Current Activity' tab.
3. Click the 'Stop Build' icon:

To start a build on demand, see 3.5 Triggering a Build Manually.

To stop a plan from submitting builds to the queue, see 1.4 Disabling or deleting a Plan.
### 1.5 Moving a Plan to a different Project

A project enables easy identification of plans that are logically related to each other, which is useful for instance when generating reports across multiple plans. Each project has a Name (e.g. "CRM System") and a Key (e.g. "CRM"). The Project Key is prefixed to the relevant Plan Keys, e.g. the "CRM" project could have plans "CRM-TRUNK" and "CRM-BRANCH".

Moving a plan to a different project will therefore involve changing the plan's Project Key (as well as possibly the Plan Name and/or Plan Key), which will also change the build key for all of the plan's build results.

Moving a plan does not affect the plan's configuration, nor any comments or labels that have been applied to the plan's build results.

⚠️ **Before you begin**

- Note that moving a plan will require Bamboo to re-index all its data, so your Bamboo system may run slowly for a few minutes.
- It is recommended that you backup your Bamboo build results before you move a plan — see [7.8 Exporting Data for Backup](#).

To move a plan to a different project,

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Click the 'Administration' link in the top navigation bar.</td>
</tr>
<tr>
<td>2.</td>
<td>Click the 'Move Plans' link in the left navigation column.</td>
</tr>
<tr>
<td>3.</td>
<td>This will display the 'Move Build Plan Wizard'. Plans are listed under their project name, e.g. in Screenshot 1 below, the 'Clover Build' plan is listed under the 'Atlassian Config' project. Select the check-box for the plan you wish to move.</td>
</tr>
<tr>
<td>4.</td>
<td>Select the 'Destination Project' from the drop-down box at the bottom of the list.</td>
</tr>
<tr>
<td>5.</td>
<td>Click the 'Move' button.</td>
</tr>
<tr>
<td>6.</td>
<td>This will display the 'Configure New Plan Details' screen (see Screenshot 2 below).</td>
</tr>
<tr>
<td>7.</td>
<td>If the destination project already includes a plan with the same Plan Name, or if you want to change the Plan Name for some other reason, overtype the 'New Plan Name' field.</td>
</tr>
<tr>
<td>8.</td>
<td>If the destination project already includes a plan with the same Plan Key, or if you want to change the Plan Key for some other reason, overtype the 'New Plan Key' field.</td>
</tr>
<tr>
<td>9.</td>
<td>Click the 'Move' button.</td>
</tr>
</tbody>
</table>

Screenshot 1: 'Move Plans-Select Plans'
Move Build Plan Wizard

You can move a plan to another project with this wizard. Simply select the plan you want to move and the destination project. All builds stored in the selected plan will then be added to the new project and kept for the plan.

Select the builds that you want to move:

- Main Build (BUILD)
- Close Build (CLOVER)

Note: because we are changing plan keys, this operation requires some cleanup operations (e.g., indexing of all builds) and may take a few minutes.

Destination Project:

- Main Build (BUILD)

Choose new build keys and build names

The following plans will be moved to project Main Build. You can update the plan names and keys below.

- Close Build
- Main Build (BUILD)

Original Project	Original Name	New Name	Original Key	New Key
Main Build (BUILD)	Close Build	Close Build	CLOVER	CLOVER

Move	Cancel
1.6 Renaming a Plan or Project

To rename a plan or a project,

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click the 'Edit' icon:

4. The 'Configuration' tab will displayed. Click the 'Plan Details' sub-tab.
5. The 'Plan Details' sub-tab will be displayed (see below).
6. In the 'Project Name' field, type the project's new name.
7. In the 'Build Plan Name' field, type the plan's new name.
8. Click the 'Save' button.

Note that a plan's Project Key and Build Plan Key are not editable, but can be changed as described in 1.5 Moving a Plan to a different Project.

Screenshot: 'Edit Plan Details'
2. Configuring Agents and Capabilities

- 2.1 About Agents and Capabilities
- 2.2 Creating a Local Agent
- 2.3 Creating a Remote Agent
  - 2.3.1 Disabling Remote Agents Support
- 2.4 Editing an Agent's Details
- 2.5 Monitoring Agent Status
- 2.6 Disabling or deleting an Agent
- 2.7 Viewing an Agent
  - 2.7.1 Viewing an Agent's Capabilities
  - 2.7.2 Viewing the Plans that an Agent can build
  - 2.7.3 Determining which Agents can build which Plans
  - 2.7.4 Viewing an Agent's System Properties
- 2.8 Configuring Capabilities
  - 2.8.1 Configuring a new Builder
    - 2.8.1.1 Configuring an Agent-specific Builder Capability
    - 2.8.1.2 Configuring a Shared Builder Capability
    - 2.8.1.3 Viewing your Builder Capabilities
  - 2.8.2 Configuring a new JDK
    - 2.8.2.1 Configuring an Agent-specific JDK Capability
    - 2.8.2.2 Configuring a Shared JDK Capability
    - 2.8.2.3 Viewing your JDK Capabilities
  - 2.8.3 Configuring a new Custom Capability
    - 2.8.3.1 Configuring an Agent-specific Custom Capability
    - 2.8.3.2 Configuring a Shared Custom Capability
  - 2.8.4 Configuring a new Perforce Capability
  - 2.8.5 Editing a Capability
  - 2.8.6 Renaming a Capability
  - 2.8.7 Deleting a Capability
  - 2.8.8 Viewing the Agents & Plans Related to a Capability
2.1 About Agents and Capabilities

What is an agent?

An agent is a service that runs Bamboo builds. There are two types of agents:

- local agents run on the Bamboo server.
- remote agents run on computers other than the Bamboo server.

(Note: Local agents run in the server's process, i.e. in the same JVM as the server. Each remote agent runs in its own process, i.e. has its own JVM.)

What is a capability?

A capability is a feature of an agent. A capability can be a:

- builder (e.g. Maven)
- JDK
- custom capability (a key-value property which defines a particular characteristic of an agent, e.g. 'operating.system=WindowsXP' or 'fast.builds=true')
- Perforce (location of the P4 client application, if Perforce is being used as the source repository)

Capabilities can be defined specifically for an agent, or they can be shared between either all local agents or all remote agents. Note that the value of an agent-specific capability overrides the value of a shared capability of the same name (if one exists).

See 2.8 Configuring Capabilities.

How are capabilities used?

A requirement is an agent capability required by a build plan.

Together, capabilities and requirements control which agents can execute builds for particular plans. Each plan can only be built by agents whose capabilities meet the plan’s requirements. Matching can be specified as either a regular expression or an exact match. See 1.2.4 Specifying a Plan’s Capability Requirements.

How are builds distributed to agents?

How do capabilities affect the distribution of builds to agents?
2.2 Creating a Local Agent

An agent is a service that runs Bamboo builds. There are two types of agents:

- local agents run on the Bamboo server.
- remote agents run on computers other than the Bamboo server.

(Note: Local agents run in the server's process, i.e. in the same JVM as the server. Each remote agent runs in its own process, i.e. has its own JVM.)

One local agent was automatically created when you installed Bamboo.

To create a new local agent,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. This will display the 'Agents' screen, showing lists of all Local Agents and all Remote Agents that currently exist in your Bamboo system. Click the 'Add Local Agent' button.
4. The 'Add Local Agent' screen will be displayed (see screenshot below).
5. In the 'Name' field, type the name of the new agent. (Note that this name will be displayed on the Dashboard.)
6. In the 'Description' field, type a description of the new agent. (This description will only be displayed to administrators.)
7. Click the 'Add' button.
8. This will return you to the 'Agents' screen. Your new agent will appear in the 'Local Agents' list.

Note: new agents are enabled by default.
9. Your new local agent will inherit all local server capabilities that are defined in your Bamboo system. You can also define agent-specific capabilities (note that these will override shared capabilities) --- see:
   - 2.8.1.1 Configuring an Agent-specific Builder Capability
   - 2.8.2.1 Configuring an Agent-specific JDK Capability
   - 2.8.3.1 Configuring an Agent-specific Custom Capability

Your new agent will be able to run builds for all plans whose requirements are met by the agent's capabilities (see 1.2.4 Specifying a Plan's Capability Requirements).

Screenshot: 'Add Local Agent'

Add Local Agent

Enter a new unique name and a description for this local agent.

Agent Details

Name: ____________________________

Description: _____________________

Add Cancel
2.3 Creating a Remote Agent

Before you begin:

- Not sure whether to install a Remote Agent? See About Agents to understand how Remote Agents interact with your Bamboo server.
- Ensure that you have specified the Broker URL, as described in the Bamboo Setup Wizard and the Bamboo 2.0 Upgrade Guide.
- Do you have sufficient Agent licenses? See Bamboo licensing for details.
- Have you enabled the creation of Remote Agents, as described in 2.3.1 Disabling Remote Agents Support.
- Ensure that you have Java Runtime Environment 5.0 or later installed on the agent machine.

To install the Bamboo Remote Agent manually,

**Step 1. Download and install the Remote Agent**

1. Create a directory on the agent machine (e.g. `bamboo-agent-home`), to serve as the "Bamboo agent home" for the remote agent.
2. On your Bamboo server, click the 'Administration' link in the top navigation bar.
3. Click the 'Agents' link in the left navigation column.
4. This will display the 'Agents' screen, showing lists of all Local Agents and all Remote Agents that currently exist in your Bamboo system.
5. Click the 'Install Remote Agent' button.
6. The following screen will be displayed:

   ![Installing a Remote Agent](image)

   **Running a Remote Agent**

   Once installed, you can run the remote agent by executing the following command line from the directory containing the remote agent JAR file:

   ```
   ```

   (Note: You may wish to configure the remote agent machine to start the Bamboo remote agent automatically when the machine boots. Please consult your operating system documentation for instructions on how to do this.)

   **Customising the Remote Agent Home Directory (Optional)**

   The Bamboo remote agent uses an 'agent home directory' to store data on the agent machine. By default, when the remote agent starts up it will create a directory named 'bamboo-agent-home' within the home directory of the current user.

   If you would like the Bamboo remote agent to place the agent home directory in a different location, use the following command line (rather than the command line above) each time you start the Bamboo remote agent:

   ```
   java -jar `bamboo-agent-home`/remoteAgent.jar http://localhost:49154/agentServer/
   ```

   where `RemoatAgentHome` is the path to the agent home directory.

7. Click the 'DOWNLOAD Remote Agent JAR' button and save the JAR file to the directory you created in step 1.1.
8. Note the command under the heading 'Running a Remote Agent' for use in step 2 below.

**Step 2. Launch the Remote Agent**

Once installed, you can run the remote agent by executing the command line obtained in the previous step. This command will look something like the following:

```
```

You may wish to configure the remote agent machine to start the Bamboo remote agent automatically when the machine boots. Please consult your operating system documentation for instructions on how to do this.
You can also choose to run the remote agent with different command line parameters, to change where the remote agent stores its data or suppress the self-signed certificate of the server.

- Changing where the remote agent stores its data
  By default, the remote agent will store its data in a directory called `bamboo-agent-home`. If you wish to specify a different directory, add the following command line parameter:
  
  ```
  -Dbamboo.home=RemoteAgentHome
  ```

  where `RemoteAgentHome` is the path to the Bamboo agent home directory you created in step 1.1. Your command line will look something like this:
  
  ```
  ```

- Suppressing the self-signed certificate of the server
  If your Bamboo server uses SSL, the following instructions will also appear in the 'Running a Remote Agent' section:
  
  You will need to carry out either one of the two options listed above. The first option of suppressing the self-signed certificate is simple to execute, but will reduce the security of your configuration (as described in the instructions). To suppress the self-signed certificate, add the following command line parameter:
  
  ```
  -Dbamboo.agent.ignoreServerCertName=true
  ```

  Your command line will look something like this:
  
  ```
  ```

  The second option of adding the self-signed certificate to the trusted certificates in your keystore is the more secure option, but is complex to set up. For detailed instructions of how to do this, please refer to the relevant Sun documentation.

Step 3. Configure the Remote Agent's Capabilities

Step 4. (Optional) Rename the Remote Agent

Your new remote agent has been automatically given a default name (e.g. 'Remote Agent on mymachine'). If you wish to rename your new remote agent, please see 2.4 Editing an Agent's Details.
2.3.1 Disabling Remote Agents Support

Disabling remote agent support in Bamboo will disable all remote agents and prevent any users from creating new remote agents. This function will not delete any remote agents that you have already created. To delete a remote agent, see 2.6 Disabling or deleting an Agent.

To disable remote agent support,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. This will display the 'Agents' screen. Click the 'Disable Remote Agent Support' link (see screenshot below).

Screenshot: 'Disabling Remote Agent Support'

If you wish to re-enable remote agent support, please follow the steps below:

To re-enable remote agent support,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. This will display the 'Agents' screen with a message indicating that remote agent support is disabled. Click the 'Enable Remote Agent Support' link (see screenshot below).

Screenshot: 'Re-enabling Remote Agent Support'
2.4 Editing an Agent's Details

To edit an agent's name or description,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all local agents and remote agents in your Bamboo system. Click the 'Edit' link in the 'Operations' column of the agent you wish to edit.
4. The 'Edit Details' page for the agent will be displayed.
5. In the 'Name' field, type the agent's new name.
6. In the 'Description' field, type the new description of the agent.
7. Click the 'Save' button.

To edit an agent's capabilities, see:

- 2.8.1.1 Configuring an Agent-specific Builder Capability
- 2.8.2.1 Configuring an Agent-specific JDK Capability
- 2.8.3.1 Configuring an Agent-specific Custom Capability

Screenshot: 'Agent---Edit Details'

Edit Details - Remote Agent on sapporo.sydney.atlassian.com

Update the details of this agent, then click Update.
2.5 Monitoring Agent Status

You can monitor your agents' status to check that all agents are functioning as expected.

To see which plans are currently being built, look at the 'Current Activity' tab on the Dashboard.

To monitor the status of your agents,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. This will display the 'Agents' screen, showing lists of all Local Agents and all Remote Agents that currently exist in your Bamboo system (see screenshot below). Each agent has one of the following statuses:
   • 'Offline' — not currently running (Note: only applicable to Remote Agents)
   • 'Idle' — available to execute builds
   • 'Building' — currently executing a build
   • 'Cancelling' — currently cancelling a build
   • 'Disabled' — not available to execute builds (see 2.6 Disabling or deleting an Agent)
   • 'Disabled - Building' — currently executing a build, but disabled so cannot execute further builds
   • 'Disabled - Cancelling' — currently cancelling a build, and disabled so cannot execute further builds

Screenshot: 'Agents'

An agent is a service that executes Bamboo builds. You can use this page to view, add and delete agents.

<table>
<thead>
<tr>
<th>Agent</th>
<th>Status</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mr Max</td>
<td>Idle</td>
<td>View</td>
</tr>
</tbody>
</table>

Local Agents

Remote Agents

Remote agents run on computers other than the Bamboo server.

There are currently no remote agents configured on this Bamboo instance.
2.6 Disabling or deleting an Agent

Bamboo allows you to disable or delete an agent, when you do not want an agent to run any further builds.

- Disabling an agent lets you keep the agent in Bamboo, but stops it from running builds.
  - Sometimes you might need to prevent Bamboo from building any plans at all (e.g., while you re-index Bamboo). You could achieve this by disabling all agents. If you do this, all builds will wait in the queue until you re-enable the agents.
- Deleting an agent removes it from Bamboo altogether. If you need to use the agent again in future, you will need to recreate it (see 2.2 Creating a Local Agent and 2.3 Creating a Remote Agent (hidden)).

Note that you can also delete/disable individual plans — this prevents the plan(s) from submitting builds to the queue. See 1.4 Disabling or deleting a Plan.

To disable an agent,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. This will display a list of all agents in your Bamboo system. The 'Status' column indicates which plans are currently enabled/disabled.
4. Locate the relevant agent and click the corresponding 'Disable' link in the 'Operations' column.

To delete an agent,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. This will display a list of all agents in your Bamboo system.
4. Locate the relevant agent and click the corresponding 'Delete' link in the 'Operations' column.

Screenshot: 'Agent---Delete or Disable'
2.7 Viewing an Agent

An agent is a service that runs Bamboo builds. There are two types of agents:

- local agents run on the Bamboo server.
- remote agents run on computers other than the Bamboo server.

(Note: Local agents run in the server's process, i.e. in the same JVM as the server. Each remote agent runs in its own process, i.e. has its own JVM.)

- 2.7.1 Viewing an Agent's Capabilities
- 2.7.2 Viewing the Plans that an Agent can build
- 2.7.3 Determining which Agents can build which Plans
- 2.7.4 Viewing an Agent's System Properties
2.7.1 Viewing an Agent's Capabilities

A capability is a feature of an agent. A capability can be a:

- builder (e.g. Maven)
- JDK
- custom capability (a key-value property which defines a particular characteristic of an agent, e.g. 'operating.system=WindowsXP' or 'fast.builds=true')
- Perforce (location of the P4 client application, if Perforce is being used as the source repository)

Capabilities can be defined specifically for an agent, or they can be shared between either all local agents or all remote agents. Note that the value of an agent-specific capability overrides the value of a shared capability of the same name (if one exists).

How capabilities are used:
A requirement is an agent capability required by a build plan.

Together, capabilities and requirements control which agents can execute builds for particular plans. Each plan can only be built by agents whose capabilities meet the plan's requirements. Matching can be specified as either a regular expression or an exact match. See 1.2.4 Specifying a Plan's Capability Requirements.

To view an agent's capabilities,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system. Click the name of the agent in which you are interested.
4. The 'Agent' page will be displayed (see screenshot below), showing a list of all shared capabilities and agent-specific capabilities (if any exist) that apply to this agent in the 'Capabilities' tab. The capabilities will be grouped under the 'Agent-Specific Capabilities' and 'Shared Capabilities' sections, as follows:
   - 'Builder' — builder capabilities. This sub-section will only display, if you have builder capabilities defined.
   - 'JDK' — JDK capabilities. This sub-section will only display, if you have JDK capabilities defined.
   - 'Custom' — custom capabilities. This sub-section will only display, if you have custom capabilities defined.
   - 'Perforce' — perforce capability. This sub-section will only display, if you have a Perforce capability defined.

To define a new capability, see 2.8 Configuring Capabilities.

Screenshot: 'Agent Capabilities'
Agent-Specific Capabilities

A capability is a feature of an agent. There are 3 types of capabilities: builders, JDKs and custom. No agent specific capabilities currently exist.

Shared Capabilities

The following shared capabilities are inherited by all remote agents. Note that the value of a shared capability is overridden by the value of the same key/label of an agent specific capability.

**Custom**

Custom capabilities are key-value pairs that define particular characteristics of an agent (e.g. operating system=Windows\P, test build=true). For an agent to be able to build a plan, both the key and value must match the plan's requirements.

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>bamboo.functionalTest</td>
<td>true</td>
</tr>
<tr>
<td>local</td>
<td>true</td>
</tr>
</tbody>
</table>

**Builder**

Builder capabilities define the builders which are available to your build plans.

<table>
<thead>
<tr>
<th>Label</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ant (Ant)</td>
<td>/opt/java/jre/aant</td>
</tr>
<tr>
<td>Bash (Command)</td>
<td>/bin/bash</td>
</tr>
<tr>
<td>MAVEN_HOME (Maven)</td>
<td>/opt/java/maven</td>
</tr>
</tbody>
</table>

**JDK**

JDK capabilities define the JDKs which are available to your build plans.

<table>
<thead>
<tr>
<th>Label</th>
<th>Java Home</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5.0_07</td>
<td>/usr/java/jdk1.5.0_07</td>
</tr>
<tr>
<td>JDK</td>
<td>/usr/java/jdk1.5.0_07</td>
</tr>
<tr>
<td>JDK 1.5</td>
<td>/usr/java/jdk1.5.0_07</td>
</tr>
</tbody>
</table>

**Perforce**

The Perforce capability defines the location of the Perforce executable available to your build plans.

<table>
<thead>
<tr>
<th>Perforce</th>
<th>Perforce Executable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The location of the Perforce P4 client application</td>
</tr>
<tr>
<td></td>
<td>/usr/local/bin/p4</td>
</tr>
</tbody>
</table>
2.7.2 Viewing the Plans that an Agent can build

A Bamboo plan (or build plan) is the "recipe" for a build.

A plan defines: what gets built (i.e. the source-code repository); how the build is triggered; which builder to use; which agent capabilities are required for the build; what artifacts the build will produce; what tests to run; who will be notified of the build result; any labels with which the build result or build artifacts will be tagged; and who has permission to view and perform various actions on a plan and its build results.

Every plan belongs to a project.

Determining which plans an Agent can build:
A requirement is an agent capability required by a build plan.

Together, capabilities and requirements control which agents can execute builds for particular plans. Each plan can only be built by agents whose capabilities meet the plan's requirements. Matching can be specified as either a regular expression or an exact match. See 1.2.4 Specifying a Plan's Capability Requirements.

To view the plans that an agent can build,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system. Click the name of the agent in which you are interested.
4. The 'Agent' page will be displayed. Click the 'Executable Plans' tab.
5. The plans that the agent is capable of building will be listed (see screenshot below).

To determine which agents are capable of building which plans, see 2.7.3 Determining which Agents can build which Plans.

Screenshot: 'Agent — View Executable Plans'

Agents > Local Agent 1 (local)
Description of Local Agent 1

Information
Current status: Idle

Capabilities Executable Plans System Properties

Executable Plans
This agent meets the requirements to execute builds for the following plans:
1. Bamboo - Acceptance Tests
2. Bamboo - Distribution
3. Bamboo - JDAQ
4. Bamboo - JDAQ Coverage
5. Bamboo - HEAD Notify
6. Bamboo - LDAP Integration Tests
7. Bamboo - Remote Agent Functional Tests
8. Bamboo - Status Acceptance Tests
9. Bamboo - Stabile Branch
10. Bamboo 2.0 on Win32 - Branch
2.7.3 Determining which Agents can build which Plans

Determining whether an agent can execute builds for a particular plan depends on whether the capabilities of an agent match the requirements specified for a plan. Read more on 1.2.4 Specifying a Plan's Capability Requirements and 2.8 Configuring Capabilities.

The ‘Agents and Plans Matrix’ page displays a matrix of which agents are capable of building which plans.

To view which agents can build which plans,

1. Click the ‘Administration’ link in the top navigation bar.
2. Click the ‘Agent Matrix’ link in the left navigation column.
3. The ‘Agents and Plans Matrix’ page will be displayed (see screenshot below) showing a matrix of all the plans currently set up, including disabled plans. Every shared and remote agent will be listed against each plan with either a tick (agent is capable of building this plan) or a cross (agent is not capable of building this plan).
4. If an agent is not capable of building a particular plan, hover your mouse over the cross requirements are not being met.

To view the plans that a specific agent can build, please also see 2.7.2 Viewing the Plans that an Agent can build.

Screenshot: ‘Agents and Plans Matrix’

<table>
<thead>
<tr>
<th>Agents and Plans Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>BAM-BAMFUNC</td>
</tr>
<tr>
<td>BAM-MANHINDSTRIK</td>
</tr>
<tr>
<td>BAM-MAIN</td>
</tr>
<tr>
<td>BAM-COVERAGE</td>
</tr>
<tr>
<td>BAM-MAINNIGHTLY</td>
</tr>
<tr>
<td>BAM-LDEP</td>
</tr>
<tr>
<td>BAM-REMOTE</td>
</tr>
<tr>
<td>BAM-STAB</td>
</tr>
<tr>
<td>BAM-AFF</td>
</tr>
</tbody>
</table>

1. bamboo functionalTest equal true
2. Maven 2.x exist
3. 1.5.0.07 exist
2.7.4 Viewing an Agent's System Properties

To view the system properties for an agent,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system. Click the name of the agent in which you are interested.
4. The 'Agent' page will be displayed. Click the 'System Properties' tab.
5. The plans that the agent is capable of building will be listed (see screenshot below).

<table>
<thead>
<tr>
<th>Capabilities</th>
<th>Executable Paths</th>
<th>System Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Date:</td>
<td>Monday, 25 Feb 2008</td>
<td></td>
</tr>
<tr>
<td>System Time:</td>
<td>17:42:02</td>
<td></td>
</tr>
<tr>
<td>Uptime:</td>
<td>5 hours, 38 minutes, 1 second</td>
<td></td>
</tr>
<tr>
<td>Username:</td>
<td>bamboo</td>
<td></td>
</tr>
<tr>
<td>User Timezone:</td>
<td>Australia/Sydney</td>
<td></td>
</tr>
<tr>
<td>User Locale:</td>
<td>English (United States)</td>
<td></td>
</tr>
<tr>
<td>System Encoding:</td>
<td>ISO-8859-1</td>
<td></td>
</tr>
<tr>
<td>Operating System:</td>
<td>Linux 2.6.15-1.1833_FC4-temp</td>
<td></td>
</tr>
<tr>
<td>Architecture:</td>
<td>i386</td>
<td></td>
</tr>
<tr>
<td>Available Processors:</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Name:</td>
</tr>
<tr>
<td>IP Address:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Memory Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Memory:</td>
</tr>
<tr>
<td>Free Memory:</td>
</tr>
<tr>
<td>Used Memory:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bamboo Paths</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current running directory:</td>
</tr>
<tr>
<td>Bamboo Home:</td>
</tr>
<tr>
<td>Temporary directory:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>File System Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk space free (for bamboo home):</td>
</tr>
</tbody>
</table>
2.8 Configuring Capabilities

A capability is a feature of an agent. A capability can be a:

- builder (e.g. Maven)
- JDK
- custom capability (a key-value property which defines a particular characteristic of an agent, e.g. 'operating.system=WindowsXP' or 'fast.builds=true')
- Perforce (location of the P4 client application, if Perforce is being used as the source repository)

Capabilities can be defined specifically for an agent, or they can be shared between either all local agents or all remote agents. Note that the value of an agent-specific capability overrides the value of a shared capability of the same name (if one exists).

- To find out what capabilities an agent already has, please see 2.7.1 Viewing an Agent's Capabilities.
- To define a new capability, see:
  - 2.8.1 Configuring a new Builder
  - 2.8.2 Configuring a new JDK
  - 2.8.3 Configuring a new Custom Capability
  - 2.8.4 Configuring a new Perforce Capability
- To edit an existing capability, see:
  - 2.8.5 Editing a Capability
- To rename an existing capability, see:
  - 2.8.6 Renaming a Capability
- To delete a capability, see:
  - 2.8.7 Deleting a Capability
- To view the agents and plans related to a capability, see:
  - 2.8.8 Viewing the Agents & Plans Related to a Capability

Note:

A requirement is an agent capability required by a build plan.

Together, capabilities and requirements control which agents can execute builds for particular plans. Each plan can only be built by agents whose capabilities meet the plan’s requirements. Matching can be specified as either a regular expression or an exact match. See 1.2.4 Specifying a Plan’s Capability Requirements.
2.8.1 Configuring a new Builder

At least one builder was automatically configured when you installed Bamboo¹. You can configure more by defining builder capabilities. Bamboo supports the following types of builders:

- Ant
- Maven
- Maven 2
- NAnt
- devenv.com
- Custom command (e.g. 'make')
- Script

If you need to use a different type of builder, you can create your own 'Builder' plugin (see the Bamboo Plugin Developer's Guide for details).

You can define a new builder capability for:

- a specific local agent --- see 2.8.1.1 Configuring an Agent-specific Builder Capability
- all local agents --- see 2.8.1.2 Configuring a Shared Builder Capability
- a specific remote agent --- see 2.8.1.1 Configuring an Agent-specific Builder Capability
- all remote agents --- see 2.8.1.2 Configuring a Shared Builder Capability

Note that if an agent has its own specific builder capability, the value will override the value of a shared builder capability of the same name (if one exists).

Once you have configured a new builder capability in your Bamboo system, its label (e.g. 'Ant') will appear in the 'Builder' drop-down list on the 'Build Resources' tab when you configure a build plan (see 1.2.3 Specifying a Plan's Builder). The builder that you select when you configure a plan will be used for every build that is executed for that plan. That is, the plan can only be built by agents which have a builder capability whose label is specified in the plan's 'Builder' field.

¹This depends on the system environment variables (e.g. 'ANT_HOME=/opt/java/ant') that were present on the machine on which Bamboo was installed:

- On the Bamboo server, environment variables that were present during installation were saved as local server capabilities in Bamboo.
- On remote agents, environment variables that were present during installation were saved as agent-specific capabilities in Bamboo.
2.8.1.1 Configuring an Agent-specific Builder Capability

Once you have configured a new builder capability in your Bamboo system, its label (e.g. 'Ant') will appear in the 'Builder' drop-down list on the 'Build Resources' tab when you configure a build plan (see 1.2.3 Specifying a Plan’s Builder). The builder that you select when you configure a plan will be used for every build that is executed for that plan. That is, the plan can only be built by agents which have a builder capability whose label is specified in the plan’s 'Builder' field.

An agent-specific capability applies to one agent only. Note that the value of an agent-specific capability will override the value of a shared capability of the same name (if one exists).

To configure a new agent-specific Builder capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click on the name of the agent for which you wish to configure a new Builder capability.
5. The 'Agent Capabilities' screen will be displayed, showing a list of shared capabilities and agent-specific capabilities for your chosen agent. Click the 'Add Capability' link in the top-right of the 'Agent-Specific Capabilities' section.
6. The 'Add Capability' page will be displayed (see screenshot below).
7. Set the 'Capability Type' field to 'Builder'.
8. Select the appropriate 'Type' from the drop-down list.
9. In the 'Label' field, type the name that you want to be displayed in the 'Builders' drop-down list when a plan is configured.
10. In the 'Path' field, type the appropriate path. This will vary depending on the 'Type' you selected in the previous step; relevant instructions will be shown below the 'Type'.
11. Click the 'Add' button. This will verify whether the Builder and Path you have specified are valid. If they are not, re-enter the values and then click the 'Add' button again.

Screenshot: 'Add Capability — Builder'
2.8.1.2 Configuring a Shared Builder Capability

Once you have configured a new builder capability in your Bamboo system, its label (e.g. 'Ant') will appear in the 'Builder' drop-down list on the 'Build Resources' tab when you configure a build plan (see 1.2.3 Specifying a Plan's Builder). The builder that you select when you configure a plan will be used for every build that is executed for that plan. That is, the plan can only be built by agents which have a builder capability whose label is specified in the plan's 'Builder' field.

Shared capabilities are inherited by all applicable agents, that is, (shared) local server capabilities are inherited by all local agents, and shared remote capabilities are inherited by all remote agents. Note, however, that the value of a shared capability will be overridden by the value of an agent-specific capability of the same name (if one exists).

To configure a new local server Builder capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Server Capabilities' link in the left navigation column.
3. The 'Server Capabilities' screen will be displayed, showing a list of all local server capabilities currently defined in your Bamboo system.
4. Locate the 'Add Capability' section below the list (see screenshot below).
5. Select 'Builder' from the 'Capability Type' dropdown.
6. Select the appropriate type of Builder from the 'Type' drop-down list.
7. In the 'Label' field, type the name that wish to appear in the 'Builder' drop-down list when a plan is configured.
8. In the 'Path' field, type the appropriate path. This will vary depending on the 'Type' you selected in the previous step; relevant instructions will be shown below the 'Type'.
9. Click the 'Add' button. This will verify whether the JDK and Path you have specified are valid. If they are not, re-enter the values and then click the 'Add' button again.

To configure a new shared remote Builder capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click the 'Shared Remote Capabilities' link in the top right of the 'Remote Agents' section.
5. The 'Shared Remote Capabilities' screen will be displayed, showing a list of all shared remote capabilities currently defined in your Bamboo system.
6. Locate the 'Add Capability' section below the list (see screenshot below).
7. Select 'Builder' from the 'Capability Type' dropdown.
8. Select the appropriate type of Builder from the 'Type' drop-down list.
9. In the 'Label' field, type the name that wish to appear in the 'Builder' drop-down list when a plan is configured.
10. In the 'Path' field, type the appropriate path. This will vary depending on the 'Type' you selected in the previous step; relevant instructions will be shown below the 'Type'.
11. Click the 'Add' button. This will verify whether the Builder and Path you have specified are valid. If they are not, re-enter the values and then click the 'Add' button again.

Screenshot: 'Add Capability — Builder'
2.8.1.3 Viewing your Builder Capabilities

You can view all the builder capabilities that have been defined in Bamboo on the 'Builders' page. These include local server capabilities, local agent-specific capabilities and remote agent-specific capabilities.

To view the Builder capabilities defined in Bamboo

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Builders' link in the left navigation column.
3. The 'Builders' page will be displayed, showing the list of all the Builder capabilities defined in your Bamboo system.
4. If you want to delete a Builder capability, click the 'Delete' link in the 'Operations' column for the Builder capability you wish to delete.
5. If you want to add the Builder as a local server capability, click the 'Add builder to server capabilities' at the bottom of the page to navigate to the 'Server Capabilities' page.

Screenshot: 'Builders'
2.8.2 Configuring a new JDK

This page last changed on Jan 14, 2008 by rosie@atlassian.com.

At least one JDK was automatically configured when you installed Bamboo¹. You can configure more by defining JDK capabilities.

You can define a new JDK capability for:

• a specific local agent --- see 2.8.2.1 Configuring an Agent-specific JDK Capability
• all local agents --- see 2.8.2.2 Configuring a Shared JDK Capability
• a specific remote agent --- see 2.8.2.1 Configuring an Agent-specific JDK Capability
• all remote agents --- see 2.8.2.2 Configuring a Shared JDK Capability

⚠️ Note that if an agent has its own specific JDK capability, the value will override the value of a shared JDK capability of the same name (if one exists).

Once you have configured a new JDK capability in your Bamboo system, its label (e.g. '1.5') will appear in the 'Build JDK' drop-down list on the 'Build Resources' tab when you configure a build plan (see 1.2.3 Specifying a Plan's Builder). The JDK that you select when you configure a plan will be used for every build that is executed for that plan. That is, the plan can only be built by agents which have a JDK capability whose label is specified in the plan's 'Build JDK' field.

¹This depends on the system environment variables (e.g. 'JAVA_HOME=/opt/java/java_sdk1.5') that were present on the machine on which Bamboo was installed:

• On the Bamboo server, environment variables that were present during installation were saved as shared local capabilities in Bamboo.
• On remote agents, environment variables that were present during installation were saved as agent-specific capabilities in Bamboo.
2.8.2.1 Configuring an Agent-specific JDK Capability

Once you have configured a new JDK capability in your Bamboo system, its label (e.g. '1.5') will appear in the 'Build JDK' drop-down list on the 'Build Resources' tab when you configure a build plan (see 1.2.3 Specifying a Plan's Builder). The JDK that you select when you configure a plan will be used for every build that is executed for that plan. That is, the plan can only be built by agents which have a JDK capability whose label is specified in the plan's 'Build JDK' field.

An agent-specific capability applies to one agent only. Note that the value of an agent-specific capability will override the value of a shared capability of the same name (if one exists).

To configure a new agent-specific JDK capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click on the name of the agent for which you wish to configure a new JDK capability.
5. The 'Agent Capabilities' screen will be displayed, showing a list of shared capabilities and agent-specific capabilities for your chosen agent. Click the 'Add Capability' link in the top-right of the 'Agent-Specific Capabilities' section.
6. The 'Add Capability' page will be displayed (see screenshot below).
7. Set the 'Capability Type' field to 'JDK'.
8. In the 'Label' field, type the name that you wish to appear in the 'Build JDK' drop-down list when a plan is configured.
9. In the 'Java Home' field, type the location of the JDK Home Directory.
10. Click the 'Add' button. This will verify whether the JDK and Path you have specified are valid. If they are not, re-enter the values and then click the 'Add' button again.

Screenshot: 'Add Capability --- JDK'
2.8.2.2 Configuring a Shared JDK Capability

Once you have configured a new JDK capability in your Bamboo system, its label (e.g. '1.5') will appear in the 'Build JDK' drop-down list on the 'Build Resources' tab when you configure a build plan (see 1.2.3 Specifying a Plan's Builder). The JDK that you select when you configure a plan will be used for every build that is executed for that plan. That is, the plan can only be built by agents which have a JDK capability whose label is specified in the plan's 'Build JDK' field.

Shared capabilities are inherited by all applicable agents, that is, (shared) local server capabilities are inherited by all local agents, and shared remote capabilities are inherited by all remote agents. Note, however, that the value of a shared capability will be overridden by the value of an agent-specific capability of the same name (if one exists).

To configure a new local server JDK capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Server Capabilities' link in the left navigation column.
3. The 'Server Capabilities' screen will be displayed, showing a list of all local server capabilities currently defined in your Bamboo system.
4. Locate the 'Add Capability' section below the list (see screenshot below).
5. Select 'JDK' from the 'Capability Type' dropdown.
6. In the 'Label' field, type the name that wish to appear in the 'Builder JDK' drop-down list when a plan is configured.
7. In the 'Path' field, type the location of the JDK Home Directory.
8. Click the 'Add' button. This will verify whether the JDK and Path you have specified are valid. If they are not, re-enter the values and then click the 'Add' button again.

To configure a new shared remote JDK capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click the 'Shared Remote Capabilities' link in the top right of the 'Remote Agents' section.
5. The 'Shared Remote Capabilities' screen will be displayed, showing a list of all shared remote capabilities currently defined in your Bamboo system.
6. Locate the 'Add Capability' section below the list (see screenshot below).
7. Select 'JDK' from the 'Capability Type' dropdown.
8. In the 'Label' field, type the name that wish to appear in the 'Builder JDK' drop-down list when a plan is configured.
9. In the 'Path' field, type the location of the JDK Home Directory.
10. Click the 'Add' button. This will verify whether the JDK and Path you have specified are valid. If they are not, re-enter the values and then click the 'Add' button again.

Screenshot: 'Add Capability --- JDK'
2.8.2.3 Viewing your JDK Capabilities

You can view all the JDK capabilities that have been defined in your Bamboo system on the 'JDKs' page. These include local server capabilities, local agent-specific capabilities and remote agent-specific capabilities.

To view the JDK capabilities defined in Bamboo

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'JDKs' link in the left navigation column.
3. The 'JDKs' page will be displayed, showing the list of all the JDK capabilities defined in your Bamboo system.
4. If you want to delete a JDK capabilities, click the 'Delete' link in the 'Operations' column for the JDK capability you wish to delete.
5. If you want to add the JDK as a local server capability, click the 'Add JDK to shared local capabilities' at the bottom of the page to navigate to the 'Shared Local Capabilities' page.

Screenshot: 'JDKs'

<table>
<thead>
<tr>
<th>JDK</th>
<th>Java Home</th>
<th>Agents</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5.0_07</td>
<td>/System/Library/Frameworks/JavaVM.framework/Versions/1.5.0/Home</td>
<td>All local agents</td>
<td>Delete</td>
</tr>
<tr>
<td>JDK</td>
<td>/System/Library/Frameworks/JavaVM.framework/Versions/1.5.0/Home</td>
<td>xxgoid sydney.atlassian.com</td>
<td>Delete</td>
</tr>
<tr>
<td>JDK 1.5.0_10</td>
<td>/System/Library/Frameworks/JavaVM.framework/Versions/1.5.0/Home</td>
<td>jdk1.5.0_07</td>
<td>Delete</td>
</tr>
<tr>
<td>JDK 1.5.0_14</td>
<td>/System/Library/Frameworks/JavaVM.framework/Versions/1.5.0/Home</td>
<td>jdk1.5.0_14</td>
<td>Delete</td>
</tr>
<tr>
<td>JDK 1.6</td>
<td>/System/Library/Frameworks/JavaVM.framework/Versions/1.6.0/Home</td>
<td>jdk1.6_00_03</td>
<td>Delete</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Add JDK to shared local capabilities
2.8.3 Configuring a new Custom Capability

Custom capabilities can be used to control which build plans will be built by a particular agent. For example, if the builds for a particular plan should only run in a Windows environment, you could create a custom capability 'operating.system=WindowsXP' for the appropriate agent(s), and specify it as a requirement for this plan. (See 1.2.4 Specifying a Plan's Capability Requirements.)

You can configure a new custom capability for:

- a specific local agent --- see 2.8.3.1 Configuring an Agent-specific Custom Capability
- all local agents --- see 2.8.3.2 Configuring a Shared Custom Capability
- a specific remote agent --- see 2.8.3.1 Configuring an Agent-specific Custom Capability
- all remote agents --- see 2.8.3.2 Configuring a Shared Custom Capability

Note that the value of an agent-specific capability overrides the value of a shared capability of the same name (if one exists).
2.8.3.1 Configuring an Agent-specific Custom Capability

Custom capabilities can be used to control which build plans will be built by a particular agent. For example, if the builds for a particular plan should only run in a Windows environment, you could create a custom capability 'operating.system=WindowsXP' for the appropriate agent(s), and specify it as a requirement for this plan. (See 1.2.4 Specifying a Plan's Capability Requirements.)

An agent-specific capability applies to one agent only. Note that the value of an agent-specific capability will override the value of a shared capability of the same name (if one exists).

To configure a new agent-specific custom capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click on the name of the agent for which you wish to configure a new Custom capability.
5. The 'Agent Capabilities' screen will be displayed, showing a list of shared capabilities and agent-specific capabilities for your chosen agent. Click the 'Add Capability' link in the top-right of the 'Agent-Specific Capabilities' section.
6. The 'Add Capability' page will be displayed (see screenshot below).
7. Set the 'Capability Type' field to 'Custom'.
8. In the 'Key' field, type the key for your new custom capability.
9. In the 'Value' field, type the value of your new custom capability.
10. Click the 'Add' button.

Screenshot: 'Add Capability --- Custom'
2.8.3.2 Configuring a Shared Custom Capability

Custom capabilities can be used to control which build plans will be built by a particular agent. For example, if the builds for a particular plan should only run in a Windows environment, you could create a custom capability 'operating.system=WindowsXP' for the appropriate agent(s), and specify it as a requirement for this plan.
(See 1.2.4 Specifying a Plan’s Capability Requirements.)

Shared capabilities are inherited by all applicable agents, that is, (shared) local server capabilities are inherited by all local agents, and shared remote capabilities are inherited by all remote agents. Note, however, that the value of a shared capability will be overridden by the value of an agent-specific capability of the same name (if one exists).

To configure a new local server custom capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Server Capabilities' link in the left navigation column.
3. The 'Server Capabilities' screen will be displayed, showing a list of all local server capabilities currently defined in your Bamboo system.
4. Locate the 'Add Capability' section at the bottom of the screen (see screenshot below).
5. In the 'Capability Type' field, select 'Custom'.
6. In the 'Key' field, type the key for your new custom capability.
7. In the 'Value' field, type the value of your new custom capability.
8. Click the 'Add' button.

To configure a new shared remote custom capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click the 'Shared Remote Capabilities' link (above the 'Remote Agents' list).
5. The 'Shared Remote Capabilities' screen will be displayed, showing a list of all shared remote capabilities currently defined in your Bamboo system.
6. Locate the 'Add Capability' section at the bottom of the screen (see screenshot below).
7. In the 'Capability Type' field, select 'Custom'.
8. In the 'Key' field, type the key for your new custom capability.
9. In the 'Value' field, type the value of your new custom capability.
10. Click the 'Add' button.

Screenshot: 'Add Capability --- Custom'
2.8.4 Configuring a new Perforce Capability

If you wish to build plans on your server and remote agents using a Perforce repository, you need to specify the location of the Perforce P4 client application for your server and each remote agent using Perforce. These locations are set by specifying a mandatory local server Perforce capability for your server and agent-specific remote Perforce capabilities for each of your remote agents using Perforce.

Shared capabilities are inherited by all applicable agents, that is, (shared) local server capabilities are inherited by all local agents, and shared remote capabilities are inherited by all remote agents. Note, however, that the value of a shared capability will be overridden by the value of an agent-specific capability of the same name (if one exists).

An agent-specific capability applies to one agent only. Note that the value of an agent-specific capability will override the value of a shared capability of the same name (if one exists).

To configure a new local server Perforce capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Server Capabilities' link in the left navigation column.
3. The 'Server Capabilities' screen will be displayed, showing a list of all local server capabilities currently defined in your Bamboo system.
4. Locate the 'Add Capability' section at the bottom of the screen (see screenshot below).
5. In the 'Capability Type' field, select 'Perforce'.
6. In the 'Perforce Executable' field, type the location of the P4 client application for your server.
7. Click the 'Add' button.

To configure a new agent-specific remote Perforce capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click on the name of the agent for which you wish to configure a new Custom capability.
5. The 'Agent Capabilities' screen will be displayed, showing a list of shared capabilities and agent-specific capabilities for your chosen agent. Click the 'Add Capability' link in the top-right of the 'Agent-Specific Capabilities' section.
6. The 'Add Capability' page will be displayed (see screenshot below).
7. Set the 'Capability Type' field to 'Perforce'.
8. In the 'Perforce Executable' field, type the location of the P4 client application for your remote agents.
9. Click the 'Add' button.

Screenshot: 'Add Capability — Perforce'
2.8.5 Editing a Capability

Before you begin:
Because each agent can only run builds for plans whose requirements are met by the agent's capabilities (see 1.2.4 Specifying a Plan's Capability Requirements), modifying a capability may mean that some plans can no longer be built.

To edit an agent-specific capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will display, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click on the name of the agent for which you wish to edit an agent-specific capability.
5. The 'Agent Capabilities' screen will be display, showing a list of shared capabilities and agent-specific capabilities for your chosen agent.
6. Click the 'Edit' corresponding to the agent-specific capability you wish to edit.
7. The 'Edit Capability' page will display. Depending on what type of capability you are editing, you will be able to modify the following values:
   - If you are editing a Builder capability, you can modify the 'Path' of the builder (see screenshot 'Edit Builder Capability' below).
   - If you are editing a JDK capability, you can modify the 'Java Home' of the JDK (see screenshot 'Edit JDK Capability' below).
   - If you are editing a Custom capability, you can modify the 'Value' of the capability (see screenshot 'Edit Custom Capability' below).
   - If you are editing a Perforce capability, you can modify the 'Perforce Executable' path (see screenshot 'Edit Perforce Capability' below).

To edit a local server capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Server Capabilities' link in the left navigation column.
3. The 'Server Capabilities' page will display, showing a list of all local server capabilities currently defined in your Bamboo system.
4. Click the 'Edit' link corresponding to the capability you wish to edit.
5. The 'Edit Capability' page will display. Depending on what type of capability you are editing, you will be able to modify the following values:
   - If you are editing a Builder capability, you can modify the 'Path' of the builder (see screenshot 'Edit Builder Capability' below).
   - If you are editing a JDK capability, you can modify the 'Java Home' of the JDK (see screenshot 'Edit JDK Capability' below).
   - If you are editing a Custom capability, you can modify the 'Value' of the capability (see screenshot 'Edit Custom Capability' below).
   - If you are editing a Perforce capability, you can modify the 'Perforce Executable' path (see screenshot 'Edit Perforce Capability' below).

To edit a shared remote capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will display, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click the 'Shared Remote Capabilities' link in the top right of the 'Remote Agents' section.
5. The 'Shared Remote Capabilities' screen will display, showing a list of all shared remote capabilities currently defined in your Bamboo system.
6. Click the 'Edit' link corresponding to the capability you wish to edit.
7. The 'Edit Capability' page will display. Depending on what type of capability you are editing, you will be able to modify the following values:
   - If you are editing a Builder capability, you can modify the 'Path' of the builder (see screenshot 'Edit Builder Capability' below).
   - If you are editing a JDK capability, you can modify the 'Java Home' of the JDK (see screenshot 'Edit JDK Capability' below).
   - If you are editing a Custom capability, you can modify the 'Value' of the capability (see screenshot 'Edit Custom Capability' below).
   - If you are editing a Perforce capability, you can modify the 'Perforce Executable' path (see screenshot 'Edit Perforce Capability' below).

Screenshot: 'Edit Capability — Builder'

**Edit Capability - Maven 2**

You can update the value for the capability on this page

<table>
<thead>
<tr>
<th>Capability Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Capabilities:</td>
</tr>
<tr>
<td>Capability Type:</td>
</tr>
<tr>
<td>Label:</td>
</tr>
<tr>
<td>Path:</td>
</tr>
</tbody>
</table>

[Update] [Cancel]

Screenshot: 'Edit Capability — JDK'

**Edit Capability - JDK 1.5.0_13**

You can update the value for the capability on this page

<table>
<thead>
<tr>
<th>Capability Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Capabilities:</td>
</tr>
<tr>
<td>Capability Type:</td>
</tr>
<tr>
<td>Label:</td>
</tr>
<tr>
<td>Java Home:</td>
</tr>
</tbody>
</table>

[Update] [Cancel]

Screenshot: 'Edit Capability — Custom'

**Edit Capability - OS**

You can update the value for the capability on this page

<table>
<thead>
<tr>
<th>Capability Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Capabilities:</td>
</tr>
<tr>
<td>Capability Type:</td>
</tr>
<tr>
<td>Key:</td>
</tr>
<tr>
<td>Value:</td>
</tr>
</tbody>
</table>

[Update] [Cancel]

Screenshot: 'Edit Capability — Perforce'
# Edit Capability - Perforce Executable

You can update the value for the capability on this page.

## Capability Details

<table>
<thead>
<tr>
<th>Shared Capabilities:</th>
<th>Server Capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capability Type:</td>
<td>Perforce</td>
</tr>
<tr>
<td>Perforce:</td>
<td>Perforce Executable</td>
</tr>
<tr>
<td>Perforce Executable:</td>
<td>/usr/bin/p4</td>
</tr>
</tbody>
</table>

The location of the Perforce P4 client application

[Update] [Cancel]
2.8.6 Renaming a Capability

To rename an agent-specific capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will display, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click on the name of the agent for which you wish to edit an agent-specific capability.
5. The 'Agent Capabilities' screen will be display, showing a list of shared capabilities and agent-specific capabilities for your chosen agent.
6. Click the 'View' corresponding to the agent-specific capability you wish to rename.
7. The 'View Capability' page will display. Click the 'Rename Capability' link.
8. The 'Rename Capability' page will display. Update the 'Key' and click 'Rename Capability'.

To rename a local server capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Server Capabilities' link in the left navigation column.
3. The 'Server Capabilities' page will display, showing a list of all local server capabilities currently defined in your Bamboo system.
4. Click the 'View' corresponding to the agent-specific capability you wish to rename.
5. The 'View Capability' page will display. Click the 'Rename Capability' link.
6. The 'Rename Capability' page will display. Update the 'Key' and click 'Rename Capability'.

To rename a shared remote capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will display, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click the 'Shared Remote Capabilities' link in the top right of the 'Remote Agents' section.
5. The 'Shared Remote Capabilities' screen will display, showing a list of all shared remote capabilities currently defined in your Bamboo system.
6. Click the 'View' corresponding to the agent-specific capability you wish to rename.
7. The 'View Capability' page will display. Click the 'Rename Capability' link.
8. The 'Rename Capability' page will display. Update the 'Key' and click 'Rename Capability'.

Screenshot: 'Rename Capability'
2.8.7 Deleting a Capability

Before you begin:
Because each agent can only run builds for plans whose requirements are met by the agent's capabilities (see 1.2.4 Specifying a Plan's Capability Requirements), deleting a capability may mean that some plans can no longer be built.

To delete an agent-specific capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click on the name of the agent that you wish to delete an agent-specific capability from.
5. The 'Agent Capabilities' screen will be displayed, showing a list of shared capabilities and agent-specific capabilities for your chosen agent.
6. Click the 'Delete' corresponding to the agent-specific capability you wish to delete.

To delete a local server capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Server Capabilities' link in the left navigation column.
3. The 'Server Capabilities' page will be displayed, showing a list of all local server capabilities currently defined in your Bamboo system.
4. Click the 'Delete' link corresponding to the capability you wish to delete.

To delete a shared remote capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will be displayed, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click the 'Shared Remote Capabilities' link in the top right of the 'Remote Agents' section.
5. The 'Shared Remote Capabilities' screen will be displayed, showing a list of all shared remote capabilities currently defined in your Bamboo system.
6. Click the 'Delete' link corresponding to the capability you wish to delete.
2.8.8 Viewing the Agents & Plans Related to a Capability

You can view a capability to determine the following information about it:

- which agents have/inherit the capability.
- which build plans have the capability specified as a requirement.

To view an agent-specific capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will display, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click on the name of the agent for which you wish to view an agent-specific capability.
5. The 'Agent Capabilities' screen will be displayed, showing a list of shared capabilities and agent-specific capabilities for your chosen agent.
6. Click the 'View' link corresponding to the capability you wish to view.
7. The 'View Capability' summary page will display (see example screenshot 'View Capability' below).

To view a local server capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Server Capabilities' link in the left navigation column.
3. The 'Server Capabilities' page will display, showing a list of all local server capabilities currently defined in your Bamboo system.
4. Click the 'View' link corresponding to the capability you wish to view.
5. The 'View Capability' summary page will display (see example screenshot 'View Capability' below).

To view a shared remote capability,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Agents' link in the left navigation column.
3. The 'Agents' page will display, showing lists of all Local Agents and Remote Agents in your Bamboo system.
4. Click the 'Shared Remote Capabilities' link in the top right of the 'Remote Agents' section.
5. The 'Shared Remote Capabilities' screen will display, showing a list of all shared remote capabilities currently defined in your Bamboo system.
6. Click the 'View' link corresponding to the capability you wish to view.
7. The 'View Capability' summary page will display (see example screenshot 'View Capability' below).

Screenshot: 'View Capability'
### Builders > Maven 2

The screen shows the summary of a capability. You can see which plans have a requirement on this capability and which agents have the capability.

#### Agents with capability

<table>
<thead>
<tr>
<th>Agent</th>
<th>Path</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>All local agents</td>
<td>/opt/dev/tools/maven-2.0.5</td>
<td>Edit</td>
</tr>
</tbody>
</table>

#### Plans with requirement

<table>
<thead>
<tr>
<th>Plan</th>
<th>Path</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVN Test - Default</td>
<td>exists</td>
<td></td>
</tr>
</tbody>
</table>
3. Triggering a Build

- 3.1 About Build Triggering
- 3.2 Triggering a Build when Code is Updated
  - 3.2.1 Polling the Repository for Code Changes
  - 3.2.2 Triggering a Build on Code Check-in
- 3.3 Triggering a Build on Schedule
  - 3.3.1 Scheduling a Single Daily Build
  - 3.3.2 Specifying a Cron-based Schedule
- 3.4 Triggering a Build when another Build finishes
- 3.5 Triggering a Build Manually
3.1 About Build Triggering

There are a variety of ways in which a build can be triggered for a plan:

- Code updated — a build can be triggered whenever one or more authors checks-in code.
- Scheduled build — a build can be scheduled to occur at regular intervals.
- Dependency — a build can be triggered whenever a successful build occurs for another plan.
- Manual build — a build can be triggered manually.
- Initial clean build — a build will be triggered when a new plan is created.

The way in which each build was triggered is listed in the 'Reason' column on the Dashboard.

⚠️ Considerations for choosing a Build Strategy

- Code updated:
  Triggering a build when code is updated ensures that a build only occurs when necessary.
  There are two ways to trigger a build when code is updated:
    ° "Pull strategy" — Polling the repository for code changes means that Bamboo will check-out the source-code on a regular basis, and examine it for changes. If Bamboo detects a change, it will trigger a build.
      See 3.2.1 Polling the Repository for Code Changes.
    ° "Push strategy" — Triggering a build on code check-in has the advantage of placing minimal load on your Bamboo server, but requires that your source-code repository is configured to fire an event to the Bamboo server.
      See 3.2.2 Triggering a Build on Code Check-in.

- Scheduled build:
  Triggering a build on schedule can allow a team to structure the day according to a predictable schedule. Note that scheduled builds are run regardless of whether or not any code changes have occurred. There are two ways to schedule a build:
    ° Single Daily Build —
      A single daily build runs at a time of your choice. This is particularly suitable for builds that take a long time to complete.
      See 3.3.1 Scheduling a Single Daily Build.
    ° Cron-Based Scheduling —
      A cron-based schedule allows you to schedule builds according to a flexible cron expression. For example, "0 0/30 9-19 * MON-FRI" would trigger a build every half-an-hour from 9am to 7pm, Monday to Friday. See 3.3.2 Specifying a Cron-based Schedule.

⚠️ Also see 1.4.3 Stopping an Active Build.
3.2 Triggering a Build when Code is Updated

Triggering a build when code is updated ensures that a build only occurs when necessary. There are two ways to trigger a build when code is updated:

- "Pull strategy" — Polling the repository for code changes means that Bamboo will check-out the source-code on a regular basis, and examine it for changes. If Bamboo detects a change, it will trigger a build.

  See 3.2.1 Polling the Repository for Code Changes.

- "Push strategy" — Triggering a build on code check-in has the advantage of placing minimal load on your Bamboo server, but requires that your source-code repository is configured to fire an event to the Bamboo server.

  See 3.2.2 Triggering a Build on Code Check-in.
3.2.1 Polling the Repository for Code Changes

Polling the repository for code changes means that Bamboo will check-out the source-code on a regular basis, and examine it for changes. If Bamboo detects a change, it will trigger a build.

To poll the repository for changes,

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

![Icon](image)

4. The 'Configuration' tab will be displayed. Click the 'Source Repository' sub-tab.
5. In the 'Build Strategy' field, select 'Poll the repository for changes' (see screenshot below).
6. In the 'Polling Frequency' field, specify how often (in seconds) Bamboo should check-out the source-code and examine it for changes.
7. Click the 'Save' button.

Screenshot: 'Plan Configuration--Source Repository-Build Strategy: Poll the repository for changes'

<table>
<thead>
<tr>
<th>Source Repository</th>
<th>Configuration</th>
<th>Build Strategy: Poll the repository for changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repository: CVS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remote: <a href="mailto:rosie@atlassian.com">rosie@atlassian.com</a></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Branch/Path: master</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Poll: 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modifiable:</td>
<td>BUILD</td>
<td></td>
</tr>
<tr>
<td>Version of Module:</td>
<td>HEAD</td>
<td></td>
</tr>
<tr>
<td>Repository URL:</td>
<td><a href="http://bamboo.atlassian.com/CONFLUENCE/ATLASSIAN">http://bamboo.atlassian.com/CONFLUENCE/ATLASSIAN</a></td>
<td></td>
</tr>
<tr>
<td>Repository Module:</td>
<td>BUILD</td>
<td></td>
</tr>
<tr>
<td>Build Strategy:</td>
<td>Poll the repository for changes</td>
<td></td>
</tr>
<tr>
<td>Polling Frequency:</td>
<td>120</td>
<td></td>
</tr>
</tbody>
</table>

RELATED TOPICS

- 3.1 About Build Triggering
- 3.2 Triggering a Build when Code is Updated
  - 3.2.1 Polling the Repository for Code Changes
  - 3.2.2 Triggering a Build on Code Check-In
- 3.3 Triggering a Build on Schedule
  - 3.3.1 Scheduling a Single Daily Build
  - 3.3.2 Specifying a Cron-based Schedule
- 3.4 Triggering a Build when another Build finishes
- 3.5 Triggering a Build Manually

Bamboo Documentation Home
3.2.2 Triggering a Build on Code Check-in

This page last changed on Dec 12, 2007 by alui.

Triggering a build on code check-in has the advantage of placing minimal load on your Bamboo server, but requires that your source-code repository is configured to fire an event to the Bamboo server.

Step 1. Configure your source repository

1. Configure your source-code repository to run post-commit scripts to tell Bamboo whenever a code commit has occurred.
   - For CVS, edit two files in the CVSROOT module: commitinfo and loginfo.
     * For commitinfo add a line like this:
       ```
       ^jira(/|$) /path/to/preCommit.sh
       ```
     where "jira" is your module.
     * For loginfo add a line like this:
       ```
       ^jira(/|$) /path/to/postCommitBuildTrigger.sh %{} http://bambooserver JIRA-MAIN JIRA-BRANCH
       ```
     where JIRA-MAIN and JIRA-BRANCH are the Bamboo plans that you would like to trigger.
   - For Subversion, edit the Subversion repository's hooks/post-commit trigger file with something like:
     ```
     /path/to/postCommitBuildTrigger.sh http://bambooserver JIRA-MAIN JIRA-BRANCH
     ```
   - For Perforce, add the script as a change-commit trigger.
     ```
     triggerName change-commit //myDepot/... "'/usr/local/bin/postCommitBuildTrigger.sh http://bambooserver/ MYPLAN-DEFAULT"
     ```
2. Copy the scripts to your repository. If you are using Bamboo Standalone, the scripts are located in the /scripts folder of your Bamboo Installation Directory. If you are using Bamboo EAR-WAR you can find them in the /repositoryScripts folder. You can also download the scripts by following this link.
3. Depending on which operating system your repository is running on, you may need to edit the scripts. The scripts assume that 'wget' is in '/usr/bin/'; if this isn't the case for your repository (e.g. Solaris 10 has it in '/usr/sfw/bin/'), edit the scripts and change '/usr/bin/' to the appropriate location.
4. Ensure that the user which Bamboo is running as has appropriate file permissions to execute the scripts, i.e. the scripts should be executable by non-root user(s).
5. Enable Bamboo's remote API so that the scripts can use Bamboo's REST-style remote API to access Bamboo's data.

Step 2. Configure Bamboo to trigger a build on code check-in

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:
4. The 'Configuration' tab will be displayed. Click the 'Source Repository' sub-tab.
5. In the 'Build Strategy' field, select 'Repository triggers the build when changes are committed' (see screenshot below).
6. This will display the 'Trigger IP Address' field. If you want Bamboo to receive post-commit notifications from the repository's primary IP address, leave the 'Trigger IP Address' field blank. If you want Bamboo to receive post-commit notifications from a different IP address, type the IP address in the 'Trigger IP Address' field.
7. Click the 'Save' button.

Screenshot: 'Plan Configuration--Source Repository--Build Strategy: Repository triggers the build when changes are committed'
<table>
<thead>
<tr>
<th><strong>Source Repository</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Repository:</strong></td>
</tr>
<tr>
<td><strong>Repository URL:</strong></td>
</tr>
<tr>
<td><strong>Username:</strong></td>
</tr>
<tr>
<td><strong>Authentication Type:</strong></td>
</tr>
<tr>
<td><strong>Include/Exclude Files:</strong></td>
</tr>
<tr>
<td><strong>Web-Repository URL:</strong></td>
</tr>
<tr>
<td><strong>Web-Repository Module:</strong></td>
</tr>
<tr>
<td><strong>Build Strategy:</strong></td>
</tr>
<tr>
<td><strong>Trigger IP Address:</strong></td>
</tr>
</tbody>
</table>

**Note:** The trigger originates from the primary IP address of the repository server. Use this field to override this with a different IP address.
3.3 Triggering a Build on Schedule

Triggering a build on schedule can allow a team to structure the day according to a predictable schedule. Note that scheduled builds are run regardless of whether or not any code changes have occurred. There are two ways to schedule a build:

- Single Daily Build —
  A _single daily build_ runs at a time of your choice. This is particularly suitable for builds that take a long time to complete. \ See 3.3.1 Scheduling a Single Daily Build.

- Cron-Based Scheduling —
  A _cron-based schedule_ allows you to schedule builds according to a flexible [cron](http://www.opensymphony.com/quartz/wikidocs/TutorialLesson6.html) expression. For example, "0 0/30 9-19 ? * MON-FRI" would trigger a build every half-an-hour from 9am to 7pm, Monday to Friday. See 3.3.2 Specifying a Cron-based Schedule.
3.3.1 Scheduling a Single Daily Build

A single daily build runs at a time of your choice. This is particularly suitable for builds that take a long time to complete.

To schedule a single daily build,

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

4. The 'Configuration' tab will be displayed. Click the 'Source Repository' sub-tab.
5. In the 'Build Strategy' field, select 'Single Daily Build' (see screenshot below).
6. In the 'Build Time' field, specify the time of day at which the build should run. Specify the time in "hh:mm" format, using a 24-hour clock.
7. Click the 'Save' button.

Screenshot: 'Plan Configuration--Source Repository-Build Strategy: Single daily build'
3.3.2 Specifying a Cron-based Schedule

A cron-based schedule allows you to schedule builds according to a flexible cron expression. For example, "0 0/30 9-19 ? * MON-FRI" would trigger a build every half-an-hour from 9am to 7pm, Monday to Friday.

To specify a cron-based schedule,

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

   ![icon]

   4. The 'Configuration' tab will be displayed. Click the 'Source Repository' sub-tab.
5. In the 'Build Strategy' field, select 'Cron Based Scheduling' (see screenshot below).
6. In the 'Cron Expression' field, type your cron expression. A cron expression consists of 6 mandatory and one optional field. The fields in sequential order are: seconds, minutes, hours, day-of-month, month, day-of-week and (optional) year. For more information about cron expressions, please see [http://www.opensymphony.com/quartz/wikidocs/TutorialLesson6.html](http://www.opensymphony.com/quartz/wikidocs/TutorialLesson6.html).
7. Click the 'Save' button.

**Screenshot:** 'Plan Configuration--Source Repository-Build Strategy: Cron-based schedule'

![Cron-based Schedule Screenshot]
### 3.4 Triggering a Build when another Build finishes

Sometimes you may want to trigger a build when another plan's build has successfully completed. This ensures that changes to one plan's code do not break a dependent build (known as a 'child' build).

For example, there could be two plans in Bamboo:

1. 'ACME-CORE' — which contains the core code for an application.
2. 'ACME-PLUGIN' — which contains code for a plugin to the application.

In this scenario, ACME-PLUGIN is a child of ACME-CORE. Any changes to the ACME-CORE code should trigger a build of ACME-PLUGIN.

In turn, every time a build completes successfully for ACME-PLUGIN, you might want to then run builds for some additional plans (e.g. 'ACME-FUNCTIONAL-TESTS' and 'ACME-UNIT-TESTS'). In this case, ACME-PLUGIN is a parent of ACME-FUNCTIONAL-TESTS and ACME-UNIT-TESTS, as well as being a child of ACME-CORE.

To trigger a build when another build finishes,

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in which you are interested.
4. Click this icon: 🌐
5. The plan's 'Configuration' tab will be displayed. Click the 'Dependencies' sub-tab.
6. This will display lists of all plans in your Bamboo system (see screenshot below). All of these are possible 'Parent plans' and 'Child plans' for the current plan.
   - In the 'Parent plans' list, select any plans for which a successful build should trigger a build for the current plan.
   - In the 'Child plans' list, select any plans for which a build should be triggered when the current plan's build finishes successfully.
7. Click the 'Save' button.

If you specify that a build should run when another build successfully finishes, you may want to prevent it from running at other times. You can achieve this by specifying 'manual builds only'.

See [3.5 Triggering a Build Manually](#).

**Screenshot: 'Plan Configuration--Dependencies'**
Studio Unit Tests - Navbar Static Plugin: Edit Build Configuration

Dependencies

The two lists below display child and parent plans. Child plans are triggered when the current plan successfully completes building. Inversely, parent plans trigger the current plan. Take care to not create circular dependencies; otherwise your plans may build continuously.

Parent plans:
- Studio Functional Tests - Studio Functional Tests
- Studio Unit Tests - AppNewis Client Tests
- Studio Unit Tests - AppNewis Conference Tests
- Studio Unit Tests - AppNewis Core Tests
- Studio Unit Tests - AppNewis F arguing Tests
- Studio Unit Tests - AppNewis JEE Plugin Tests
- Studio Unit Tests - Conference Theme Tests
- Studio Unit Tests - Studio Federar Tests
- Studio Unit Tests - Studio Navigar Tests
- Studio Unit Tests - User Counter Plugin Tests

Child plans:
- Studio Functional Tests - Studio Functional Tests
- Studio Unit Tests - AppNewis Client Tests
- Studio Unit Tests - AppNewis Conference Tests
- Studio Unit Tests - AppNewis Core Tests
- Studio Unit Tests - AppNewis F arguing Tests
- Studio Unit Tests - AppNewis JEE Plugin Tests
- Studio Unit Tests - Conference Theme Tests
- Studio Unit Tests - Studio Federar Tests
- Studio Unit Tests - Studio Navigar Tests
- Studio Unit Tests - User Counter Plugin Tests

When a parent plan successfully builds, it will trigger the current plan to build.

When the current plan successfully builds, it will trigger the child plans to build.

Save   Cancel
3.5 Triggering a Build Manually

To start a manual build,

1. Click 'Home' to go to the Dashboard.
2. Locate the relevant plan and click the 'Check Out and Build' icon:
   - See also 1.4.3 Stopping an Active Build.

You can specify that a plan should only ever be built manually. This is useful for:

- Broken builds:
  If a build is broken, you may want to temporarily specify 'manual builds only'. This means that a failing build will not be triggered frequently and hence will not take up time and processing power when other builds could be running.

- Dependent builds:
  If you specify that a build should run when another build successfully finishes, you may want to prevent it from running at other times. You can achieve this by specifying 'manual builds only'.

To specify 'manual builds only',

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:
4. The 'Configuration' tab will be displayed. Click the 'Source Repository' sub-tab.
5. In the 'Build Strategy' field, select 'Manual builds only' (see screenshot below).
6. Click the 'Save' button.

Screenshot: 'Plan Configuration--Source Repository-Build Strategy: Manual builds only'
4. Configuring Email and Instant Messaging Notifications

- 4.1 Enabling or disabling Notifications for a Plan
- 4.2 Configuring Bamboo to send SMTP Email
- 4.3 Configuring Bamboo to use Instant Messaging (IM)
  - 4.3.1 Configuring Bamboo to use Google Talk for Instant Messaging
4.1 Enabling or disabling Notifications for a Plan

You can specify which people will receive notifications about build results for a particular plan, and under what circumstances (known as 'Notification Triggers'), i.e.:

<table>
<thead>
<tr>
<th>Notification Trigger</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>'All Completed Builds'</td>
<td>Bamboo will send a notification whenever a build finishes for this plan, regardless of the build result. This trigger is recommended for any plans for which it is critical that people are always informed about the latest build activity. Many organisations start with this trigger, then change it later as they get more confidence in the continuous build process.</td>
</tr>
</tbody>
</table>
| 'Failed Builds And First Successful'  | This trigger is generally suitable for the majority of plans. Bamboo will send a notification whenever:  
  - a build fails for this plan.  
  - the plan is 'fixed' (that is, the plan's latest build is successful and the previous build failed). |
| 'After X Failed Builds'               | This trigger enables you to specify the 'Number Of Failed Builds' after which Bamboo will send a notification. This is a useful way of limiting the number of notifications, if you are concerned about people receiving too many. |

For each plan, you can specify different recipients for each Notification Trigger. Note also that recipients need not be people with Bamboo user accounts.

⚠️ Before you begin
You need to configure Bamboo's SMTP email and/or instant messaging capabilities before Bamboo can send notifications.

To enable notifications for a plan,

ℹ️ If you are creating a new plan, start at step 5.

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

4. The 'Configuration' tab will be displayed. Click the 'Build Notifications' sub-tab.
5. A list of 'Notification Triggers' will be displayed (see screenshot below). Select the 'Notification Trigger' you wish to enable, then specify any or all of the following recipients:
   - 'Roles' — Select from the following:
     - 'Committer' — A committer is the Bamboo user(s) who committed code to a particular build (i.e. someone who committed code after the previous build was checked out by Bamboo).
     - 'Watcher' — A plan’s watchers are the Bamboo users who have marked this plan as one of their favourites.
   - 'Groups' — Type the names of the appropriate Bamboo group(s), separated by commas.
   - 'Users' — Type the usernames of the appropriate Bamboo users, separated by commas; or click the following icon to select from a list of users:

   - 'Email Addresses' — This is useful if you need to send email notifications to people who are not Bamboo users. Type the appropriate email addresses, separated by commas.
   - 'Instant Messaging Addresses' — This is useful if you need to send IM notifications to people who are not Bamboo users. Type the appropriate IM addresses, separated by commas. Note that if you specify a broadcast address (eg. 'project-x@broadcast.chat.mycompany.com'), Bamboo will not know the context of related IM responses.
6. Click the 'Add' button.
7. Repeat steps 5 and 6 until you have added all the Notification Triggers that you wish to enable for this plan.
8. Click the 'Done' button if you are editing an existing plan; or if you are creating a new plan, either click the 'Next' button and go to 1.2.7 Specifying a Plan's Post Actions or click the 'Summarise' button and go to 1.2.9 Completing and Enabling a New Plan (hidden).

Each Bamboo user can choose whether to receive their notifications via email, IM, both or neither.

**Screenshot: 'Build Notifications'**

<table>
<thead>
<tr>
<th>Add Build Notification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notification Condition: All Completed Builds</td>
</tr>
<tr>
<td>When should Bamboo send a notification?</td>
</tr>
<tr>
<td>Rules:</td>
</tr>
<tr>
<td>Committer: Users who have committed to the build trigger. Users who have marked this build as successful</td>
</tr>
<tr>
<td>Users:</td>
</tr>
<tr>
<td>Groups:</td>
</tr>
<tr>
<td>Email Addresses:</td>
</tr>
<tr>
<td>Instant Messaging Addresses:</td>
</tr>
</tbody>
</table>

To disable notifications for a plan,

1. Click 'Home' to go to the Dashboard.
2. Click the 'All Plans' tab.
3. Locate the plan in the list and click this icon:

4. The 'Configuration' tab will be displayed. Click the 'Build Notifications' tab.
5. For each 'Notification Trigger', click the corresponding 'Remove' link in the 'Operations' column.
6. Click the 'Done' button.

Bamboo will no longer send notifications about this plan's build results.
4.2 Configuring Bamboo to send SMTP Email

This page last changed on Dec 11, 2007 by alui.

Bamboo can send email notifications about build results. There are two steps to setting this up:

1. Configure Bamboo to send SMTP email (see below).
2. Configure a plan to send SMTP email notifications about build results (see 4.1 Enabling or disabling Notifications for a Plan).

To configure Bamboo to send SMTP email,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Mail Server' link in the left navigation column (under 'Communication').
3. This will display the 'Mail Server Details' page (see screenshot below). Click the 'Edit' button.
4. In the 'Name' field, type a display-name for the email address in step 5 (below).
5. In the 'From Address' field, type the email address from which Bamboo notifications will be sent.
6. In the 'Subject Prefix' field, type the text (if any) with which you would like the email Subject line to begin. E.g. if you type '[Bamboo]', then people will receive emails with Subjects like this:
   - '[Bamboo] TEST build 1,001 has FAILED (77 tests failed, no failures were new) : Change made by jsmith'
   - '[Bamboo] TEST build 1,002 was SUCCESSFUL (with 77 tests) : Change made by jsmith'
7. If you are not using JNDI¹,
   a. In the 'SMTP Server' field, type the name of the email server via which Bamboo notifications will be sent. E.g. 'mail.myserver.com'.
   b. In the 'Username' field, type the login name of the account which Bamboo will use to login to the SMTP server.
   c. In the 'Password' field, type the password for the account specified in step 7 (if any).
   d. Go to step 8.
8. If you are using JNDI¹, type the JNDI name in the 'JNDI Location' field. The JNDI Location will depend on your application server, and on the location of the 'mail' resource within the JNDI tree you specify. E.g. 'java:comp/env/mail/BambooMailServer'.
9. Type a test email address in the 'Test Recipient Address' box.
10. Click the 'Test' button, and verify that a test email is received.
11. Click the 'Save' button.

¹ Note re JNDI: As an alternative to specifying mail details directly in Bamboo, you can configure them in your application server (e.g. in the server.xml file — see 7.1 Locating Important Directories and Files), and then use JNDI to look up a preconfigured mail session. JNDI has the following advantages:

   • Centralised management - mail details are configured in the same place as database details, and may be configured through your application server administration tools.
   • Better security - mail details are not available to Bamboo administrators through the Bamboo interface, and aren’t stored in Bamboo backup files.
   • More SMTP options  - e.g. SSL. If you want to use SMTP over SSL you will need to use JNDI.

Screenshot: 'Email Server Details'
Next step
Now that you have configured Bamboo’s SMTP email capability, you can specify notifications for a plan.
4.3 Configuring Bamboo to use Instant Messaging (IM)

This page last changed on Dec 11, 2007 by alui.

Bamboo can send Instant Messaging (IM) notifications about build results. There are two steps to setting this up:

1. Configure Bamboo to use Instant Messaging (see below).
2. Configure a plan to send IM notifications about build results (see 4.1 Enabling or disabling Notifications for a Plan).

To configure Bamboo to use Instant Messaging,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'IM Server' link in the left navigation column (under 'Communication').
3. This will display the 'Instant Messaging Server Details' page. Click the 'Edit' button.
4. In the 'Host' field, type the address of your IM server (for example, 'chat.atlassian.com').
5. In the 'Port' field, type the TCP port that your organisation uses for IM traffic (or leave this field blank to have Bamboo either perform a DNS lookup or use the default port).
6. In the 'Username' field, type the login name of the IM account from which Bamboo notifications will be sent.
7. In the 'Password' field, type the password for the account specified in step 6.
8. If your IM server uses SSL, select the 'Requires an SSL Connection' check-box.
9. Type a test IM user's address in the 'Test Recipient Address' box.
10. Click the 'Test' button, and verify that a test IM message is received.
11. Click the 'Save' button.

Screenshot: 'Instant Messaging Server Details'

Next step

Now that you have configured Bamboo's IM capability, you can specify notifications for a plan.
4.3.1 Configuring Bamboo to use Google Talk for Instant Messaging

If your Bamboo server has access to the internet, it can use Google Talk to send IM notifications about build results.

To configure Bamboo to use Google Talk for Instant Messaging,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'IM Server' link in the left navigation column (under 'Communication').
3. This will display the 'Instant Messaging Server Details' page. Click the 'Edit' button.
4. In the 'Host' field, type 'gmail.com'.
   - If your IM Server uses an "@googlemail.com" account, type 'googlemail.com' in the 'Host' field.
5. Leave the 'Port' field blank, Bamboo will perform a DNS lookup to figure out which port to use.
6. In the 'Username' field, type the login name of the Google account from which IM notifications will be sent. Only the account name needs to be included e.g. atlassianbamboo NOT atlassianbamboo@gmail.com.
   - Do not select "Requires an SSL connection"
7. In the 'Password' field, type the password for the account specified in step 6.
8. Type a test IM user's address (e.g. atlassianbamboo@gmail.com NOT atlassianbamboo) in the 'Test Recipient Address' box. (Note: use a different user to the one you specified in step 6.)
9. Click the 'Test' button, and verify that the message is successfully received.
10. Click the 'Save' button.

Google Talk does not allow IM messages to be received unless the receiver has approved the sender. Please ensure that the Gmail user specified in step 6 is approved by each Google Talk recipient. That is, ensure that the 'Host' and 'Username' have previously sent messages to each other via Google Talk.

Additional notes about using Google Talk:

- The Google Talk service is hosted at talk.google.com. The default port is 5222. (Note: be aware that your firewall might be blocking traffic to this port.)
- TLS is required.
- The only supported authentication mechanism is SASL PLAIN. For additional information, please see: http://code.google.com/apis/talk/open_communications.html
5. Managing Users and Security

About users and authors

An author is any person who contributes to a build by checking-in code to a repository that is associated with a Bamboo plan. An author need not be a Bamboo user. Depending on your organisation's needs, you can configure Bamboo to grant access to non-users. However, only Bamboo users can:

- view the 'My Bamboo' tab on the Dashboard.
- belong to a group.

About groups

Bamboo groups are used to specify which users will have global permissions and plan permissions. They can also be used to specify which users will receive notifications about a plan's build results. You can create and delete as many groups as you need. You will typically create at least one group per project.

A special group called bamboo-admin is automatically created when you install Bamboo. Members of this group have Bamboo administration rights.

About permissions

A plan permission is the ability to perform a particular operation in relation to a build plan. For each plan, different permissions can be granted to particular groups and/or users.

A global permission is the ability to perform a particular operation in relation to Bamboo as a whole.

More about...

- 5.01 Creating a User
- 5.02 Changing a User's Password or Details
- 5.03 Deleting or deactivating a User
- 5.04 Creating a Group
- 5.05 Deleting a Group
- 5.06 Adding Users to and removing them from Groups
- 5.07 Granting Plan Permissions to Users or Groups
- 5.08 Granting Global Permissions to Users or Groups
- 5.09 Granting Administration Rights to a User
- 5.10 Allowing Anonymous Users to access Bamboo
- 5.11 Enabling or disabling Contact Details Display
- 5.12 Enabling or disabling Public Signup
- 5.13 Working with External User Repositories
  - 5.13.1 Integrating Bamboo with Crowd
  - 5.13.2 Integrating Bamboo with LDAP
    - 5.13.2.1 Configuring the Caching of your LDAP Repository
5.01 Creating a User

A user is someone who can login to Bamboo.
An author is any person who contributes to a build by checking-in code to a repository that is associated with a Bamboo plan. An author need not be a Bamboo user. Depending on your organisation's needs, you can configure Bamboo to grant access to non-users. However, only Bamboo users can:

- view the 'My Bamboo' tab on the Dashboard.
- belong to a group.

To create a Bamboo user,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Users' link in the left navigation column.
3. The 'Manage Users' screen will be displayed. The 'Add User' section (as shown below) will be displayed at the bottom of the 'Manage Users' screen.
4. In the 'Username' field, type the user's login name.
   - Note that the Username cannot be changed after the user is created.
5. In the 'Password' and 'Confirm Password' fields, type the user's password.
   - The user can easily change their password later.
6. In the 'Full Name' field, type the user's display-name.
7. In the 'Email' field, type the user's email address. This address is where the user will receive password notifications.
8. (optional) In the 'Jabber address' field, type the user's Instant Messaging (IM) address. This address is where the user will receive any group-based notifications about build results.
   - If no IM address is specified, Bamboo will not be able to recognise the user's context when interacting via IM.
9. Select at least one group from the 'Groups' list. (To select multiple groups, press the <Ctrl> key.) Once you add a user to a group, the group will be shown in blue; groups to which the user does not belong are shown in white.
10. If the user is a Bamboo author, select 'Add Alias' (instead of 'None') in the 'Source Repository Alias' field. This will display the 'New alias' field. Type the user's login name for their source-code repository.
   - If you don't know the user's login name for their source-code repository, they can specify it themselves later.
11. Click the 'Save' button.

If you have configured SMTP email on your Bamboo server, the new user will automatically receive an email containing their Bamboo login name and password.

Screenshot: 'Add User'

Add User

<table>
<thead>
<tr>
<th>User Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username:</td>
</tr>
<tr>
<td>Password:</td>
</tr>
<tr>
<td>Confirm Password:</td>
</tr>
<tr>
<td>Full Name:</td>
</tr>
<tr>
<td>Email:</td>
</tr>
<tr>
<td>Jabber Address:</td>
</tr>
<tr>
<td>Groups:</td>
</tr>
</tbody>
</table>

Source Repository Alias:
- Add Alias: addalias

New alias: jdoe
5.02 Changing a User's Password or Details

To change a user's password,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Users' link in the left navigation column.
3. The 'Manage Users' screen will be displayed. To locate the user, type part of their 'Username', 'Full Name' or 'Email' and click the 'Enter' button. This will display a list of matching users.
4. Click the 'Edit' link (in the 'Operations' column) that corresponds to the user.
5. The 'User Details' screen will be displayed. Type the new password in the 'Password' and 'Confirm Password' fields.
   ⚠️ If you have configured SMTP email on your Bamboo server, the user will automatically receive an email containing their new password. The user can easily change their password later.
6. Click the 'Save' button.

To change a user's details (e.g. Email or Full Name),

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Users' link in the left navigation column.
3. The 'Manage Users' screen will be displayed. To locate the user, type part of their 'Username', 'Full Name' or 'Email' and click the 'Enter' button. This will display a list of matching users.
4. Click the 'Edit' link (in the 'Operations' column) that corresponds to the user.
5. The 'User Details' screen will be displayed. Enter the new details as described in 5.01 Creating a User.
   ⚠️ If you do not need to change the user's password, simply leave the 'Password' and 'Confirm Password' fields blank.
6. Click the 'Save' button.

Note
Users who have forgotten their passwords can click the 'Forgotten your password?' link on the Bamboo login screen. This will automatically generate a new password and email it to the user (provided the Bamboo server has been configured to send SMTP email). Logged-in users can also change their own password and details, as described in 11. Editing your User Profile in the Bamboo User's Guide:

- 11.1 Changing your Password
- 11.2 Changing your Notification Preferences
- 11.3 Associating your Author Name with your User Profile
Deleting a user removes their Bamboo user account. Deactivating a user revokes their permission to login to Bamboo.

Note that deleting a Bamboo user will not delete their author data — that is, their author statistics and code check-in comments will still exist in Bamboo.

Also note that:

- You cannot delete a user who has created labels or comments about build results. You may want to deactivate them instead.
- You cannot delete the user account with which you are currently logged in to Bamboo.

To delete a Bamboo user,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Users' link in the left navigation column.
3. The 'Manage Users' screen will be displayed. Locate the relevant user in the list, and click the corresponding 'Delete' link in the 'Operations' column.

To deactivate a Bamboo user,

- please see Deactivating a Bamboo user

<table>
<thead>
<tr>
<th>Username</th>
<th>Email</th>
<th>Full Name</th>
<th>Groups</th>
<th>Repository Alias</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>admin</td>
<td><a href="mailto:admin@company.com">admin@company.com</a></td>
<td>admin</td>
<td>bamboo-admin</td>
<td></td>
<td>view</td>
</tr>
<tr>
<td>smith</td>
<td><a href="mailto:smith@company.com">smith@company.com</a></td>
<td>Jill Smith</td>
<td>project</td>
<td>Alice</td>
<td>view delete</td>
</tr>
</tbody>
</table>
5.04 Creating a Group

Bamboo groups are used to specify which users will have global permissions and plan permissions. They can also be used to specify which users will receive notifications about a plan's build results. You can create and delete as many groups as you need. You will typically create at least one group per project.

A special group called bamboo-admin is automatically created when you install Bamboo. Members of this group have Bamboo administration rights. To create a group,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Groups' link in the left navigation column.
3. The 'Manage Groups' screen will be displayed. The 'Create Group' section (as shown below) will be displayed at the bottom of the 'Manage Groups' screen.
4. In the 'Group Name' field, type a name for your new group. Note that the Group Name cannot be changed after the group is created.
5. Select relevant users from the 'Users in Group' list. Hold the <Ctrl> to select multiple users. You can also add or remove users from the group later if required.
6. Click the 'Save' button.

Screenshot: Create Group
5.05 Deleting a Group

Bamboo groups are used to specify which *users* will have *global permissions* and *plan permissions*. They can also be used to specify which users will receive *notifications about a plan's build results*. You can create and delete as many groups as you need. You will typically create at least one group per *project*.

A special group called bamboo-admin is automatically created when you install Bamboo. Members of this group have *Bamboo administration rights*.

The bamboo-admin group cannot be deleted.

To delete a group,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Groups' link in the left navigation column.
3. The 'Manage Groups' screen will be displayed. Locate the relevant group in the list, and click the corresponding 'Delete' link in the 'Operations' column.

<table>
<thead>
<tr>
<th>Group</th>
<th>Number of Users</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>bamboo-admin</td>
<td>20</td>
<td>Edit</td>
</tr>
<tr>
<td>cloud-develop</td>
<td>0</td>
<td>Edit (only)</td>
</tr>
</tbody>
</table>
5.06 Adding Users to and removing them from Groups

Bamboo groups are used to specify which users will have global permissions and plan permissions. They can also be used to specify which users will receive notifications about a plan’s build results. You can create and delete as many groups as you need. You will typically create at least one group per project.

A special group called bamboo-admin is automatically created when you install Bamboo. Members of this group have Bamboo administration rights.

To add users to a group,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Groups' link in the left navigation column.
3. The 'Manage Groups' screen will be displayed. Locate the relevant group in the list, and click the corresponding 'Edit' link in the 'Operations' column.
4. The 'Edit Group Details' screen will be displayed. Users who already belong to the group are shown in blue; users who do not currently belong to the group are shown in white. Press the <Ctrl> key and hold it while you select the user(s) whom you want to add to the group.
5. Click the 'Save' button.

Screenshot: Edit Group Details

To remove users from a group,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Groups' link in the left navigation column.
3. The 'Manage Groups' screen will be displayed. Locate the relevant group in the list, and click the corresponding 'Edit' link in the 'Operations' column.
4. The 'Edit Group Details' screen will be displayed. Users who belong to the group are shown in blue. Press the <Ctrl> key and hold it while you deselect the user(s) whom you want to remove from the group.
5. Click the 'Save' button.

⚠ You cannot remove a user from the bamboo-admin group if they are the only member.
5.07 Granting Plan Permissions to Users or Groups

A plan permission is the ability to perform a particular operation in relation to a build plan. For each plan, different permissions can be granted to particular groups and/or users. The following plan permissions are available:

<table>
<thead>
<tr>
<th>Plan permission</th>
<th>Description</th>
<th>Can be granted to</th>
</tr>
</thead>
</table>
| View            | Permission to:  
|                 | • view this plan's build results  
|                 | • add comments or labels to this plan's build results¹  
|                 | People who don't have the 'View' permission will not know that the plan exists. | - a particular user  
|                 | - a particular group  
|                 | - all logged-in users  
|                 | - anonymous users²  |
| Edit            | Permission to view and edit this plan's configuration, except for the plan's permissions. | - a particular user  
|                 | - a particular group  
|                 | - all logged-in users  
|                 | - anonymous users²  |
| Build           | Permission to:  
|                 | • manually start/stop a build for this plan.  
|                 | • enable/disable this plan from submitting builds to the queue. | - a particular user  
|                 | - a particular group  
|                 | - all logged-in users  
|                 | - anonymous users²  |
| Clone           | Permission to copy this plan when creating a new plan.  
|                 | (Note: only users with the 'Create Plan' global permission can create new plans.) | - a particular user  
|                 | - a particular group  |
| Admin           | Permission to:  
|                 | • edit this plan's permissions.  
|                 | • delete this plan's build results and working files. | - a particular user  
|                 | - a particular group  |

¹ Only logged-in users (not anonymous users) can label or comment on a build result.
² Anonymous users cannot access Bamboo at all unless they have been granted the 'Access' global permission. See 5.10 Allowing Anonymous Users to access Bamboo.

Anyone with the 'Admin' global permission automatically has all plan permissions for every plan.

How plan permissions are granted:

- People who have the 'Admin' global permission can 'bulk edit' permissions for multiple plans at the same time, as described below. Note that this will overwrite any pre-existing plan permissions.
- People who have the 'Admin' plan permission for one or more plans, but do not have the 'Admin' global permission, can only edit one plan at a time, as described in 1.2.8 Specifying a Plan’s Permissions.

The processes for granting and revoking permissions across multiple plans are as follows. Note that, for ongoing ease of management, it is recommended that you grant permissions to groups rather than to individual users.

To grant multiple plan permissions to a user,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Bulk Edit Plan Permissions' link in the left navigation column.
3. The first screen of the 'Bulk Edit Plan Permissions Wizard' will be displayed (see below). Select the plans whose permissions you wish to edit, then click the 'Next' button (at the bottom of the screen).
4. The second screen of the 'Bulk Edit Plan Permissions Wizard' will be displayed (see below).
5. In the 'Grant permission to' list at the bottom of the screen, select 'User'.
6. Type the username into the box, or click the following icon to select from a list of users:
7. Click the 'Add' button.
8. The user will be added to the list of users on the screen. Select the check-box for each permission that you wish to grant to this user.
9. Click the 'Save' button.

To grant multiple plan permissions to a group,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Bulk Edit Plan Permissions' link in the left navigation column.
3. The first screen of the 'Bulk Edit Plan Permissions Wizard' will be displayed (see below). Select the plans whose permissions you wish to edit, then click the 'Next' button (at the bottom of the screen).
4. The second screen of the 'Bulk Edit Plan Permissions Wizard' will be displayed (see below).
5. In the 'Grant permission to' list at the bottom of the screen, select 'Group'.
6. Type the group name into the box.
7. Click the 'Add' button.
8. The group will be added to the list of groups on the screen. Select the check-box for each permission that you wish to grant to this group.
9. Click the 'Save' button.

To grant multiple plan permissions to all Bamboo users,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Bulk Edit Plan Permissions' link in the left navigation column.
3. The first screen of the 'Bulk Edit Plan Permissions Wizard' will be displayed (see below). Select the plans whose permissions you wish to edit, then click the 'Next' button (at the bottom of the screen).
4. The second screen of the 'Bulk Edit Plan Permissions Wizard' will be displayed (see below).
5. Locate 'Logged in users' (under 'Other').
6. Select the check-box for each permission that you wish to grant to all Bamboo users.
7. Click the 'Save' button.

To grant multiple plan permissions to anonymous users,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Bulk Edit Plan Permissions' link in the left navigation column.
3. The first screen of the 'Bulk Edit Plan Permissions Wizard' will be displayed (see below). Select the plans whose permissions you wish to edit, then click the 'Next' button (at the bottom of the screen).
4. The second screen of the 'Bulk Edit Plan Permissions Wizard' will be displayed (see below).
5. Locate 'Anonymous users' (under 'Other').
6. Select the check-box for each permission that you wish to grant to all anonymous users.
7. Click the 'Save' button.

To revoke multiple plan permissions,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Bulk Edit Plan Permissions' link in the left navigation column.
3. The first screen of the 'Bulk Edit Plan Permissions Wizard' will be displayed (see below). Select the plans whose permissions you wish to edit, then click the 'Next' button (at the bottom of the screen).
4. The second screen of the 'Bulk Edit Plan Permissions Wizard' will be displayed (see below).
5. Locate the relevant user/group/all logged-in users/anonymous users.
6. Deselect the check-box for each permission that you wish to revoke from the user/group/all users/anonymous users.

   If you deselect all permissions for a user or group, that user or group will disappear from the screen.

7. Click the 'Save' button.

Screenshot 1: 'Bulk Edit Plan Permissions Wizard — Select Plans'

Bulk Edit Plan Permissions Wizard

Select Plans

All existing permissions defined for the plans you select will be deleted and replaced with the new permissions you specify.

You can apply a consistent set of permissions to multiple plans with this wizard. Simply select the plans you want to edit and then specify the permissions you wish to apply.

Plans:

Confluence (CONF)
- Unit Tests - JDK 1.4 (MAIN)
- Stable RPC Unit Tests (STABRPC)
- Stable Unit Tests (STABUNIT)
- Atlassian User Stable Unit Tests (ATLUSRSTABUNIT)
- Atlassian User Stable Integration Tests (ATLUSRSTABIT)
- Unit Tests - JDK 1.5 (MAINJDK15)
- Distribution (DISTR)
- Unit Tests - JDK 1.6 (MAINJDK16)
- Crowd 1.1 JDK 1.5 (1.1JDK16)

Crowd (CWD)
- Main Build (MAIN)
- Crowd JDK 1.8 (JDK18)
- Crowd JDK 1.4 (JDK14)
- Integration Tests (INT)
- Build Plugin (BUILDPLUGIN)
- Crowd 1.1 JDK 1.4 (1.1JDK14)
- Crowd 1.1-Integration Tests (1.1INT)
- Crowd 1.1-Main Build (1.1MAIN)

Next >>

Screenshot 2: 'Bulk Edit Plan Permissions Wizard — Permissions'
## Bulk Edit Plan Permissions Wizard

### Permissions

You can edit your plan permissions here. Permissions can be granted to specific users or groups.

<table>
<thead>
<tr>
<th>Groups</th>
<th>View</th>
<th>Edit</th>
<th>Build</th>
<th>Clone</th>
<th>Admin</th>
</tr>
</thead>
<tbody>
<tr>
<td>bamboo-admin</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Other</th>
<th>View</th>
<th>Edit</th>
<th>Build</th>
<th>Clone</th>
<th>Admin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logged in Users</td>
<td>☑</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Anonymous users</td>
<td>☑</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Grant permission to [User](#) [Add](#)

[Save] [Cancel]
5.08 Granting Global Permissions to Users or Groups

This page last changed on Dec 12, 2007 by alui.

A global permission is the ability to perform a particular operation in relation to Bamboo as a whole.

You can grant the following global permissions:

<table>
<thead>
<tr>
<th>Global permission</th>
<th>Description</th>
<th>Can be granted to</th>
</tr>
</thead>
<tbody>
<tr>
<td>'Access'</td>
<td>Permission to view the Bamboo system. (Note that the ability to view build plans and build results is subject to individual plan permissions.)</td>
<td>- a particular user</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- all logged-in users</td>
</tr>
<tr>
<td>'Create Plan'</td>
<td>Permission to create new build plans.</td>
<td>- a particular user</td>
</tr>
<tr>
<td>'Admin'</td>
<td>Permission to:</td>
<td>- a particular user</td>
</tr>
<tr>
<td></td>
<td>• access the Bamboo 'Administration' menu.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• delete plans.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The 'Admin' global permission also includes all plan permissions, for every plan.</td>
<td></td>
</tr>
</tbody>
</table>

¹ i.e. people who are not logged in to Bamboo.

The processes for granting and revoking global permissions are described below.

To grant global permissions to a user,

1. Click 'Home' to go to the Dashboard.
2. Click the 'Administration' link in the top navigation bar.
3. Click the 'Global Permissions' link in the left navigation column to display the 'Global Permissions' screen (see screenshot below).
4. In the 'Grant permission to' list at the bottom of the screen, select 'User'.
5. Type the username into the box, or click the following icon to select from a list of users:

   Note that the assignment of permissions to LDAP users and groups in Bamboo is case sensitive. For instance, if the username of the LDAP user is 'Bob', you will need to type in 'Bob' (not 'bob' or 'BOB').
6. Click the 'Add' button.
7. The user will be added to the list of users on the 'Permissions' tab. Select the check-box for each permission that you wish to grant to this user.
8. Click the 'Save' button.

To grant global permissions to a group,

1. Click 'Home' to go to the Dashboard.
2. Click the 'Administration' link in the top navigation bar.
3. Click the 'Global Permissions' link in the left navigation column to display the 'Global Permissions' screen (see screenshot below).
4. In the 'Grant permission to' list at the bottom of the screen, select 'Group'.
5. Type the group name into the box.

   Note that the assignment of permissions to LDAP users and groups in Bamboo is case sensitive. For instance, if the name of the LDAP group is 'Dev', you will need to type in 'Dev' (not 'dev' or 'DEV').
6. Click the 'Add' button.
7. The group will be added to the list of groups on the 'Permissions' tab. Select the check-box for each permission that you wish to grant to this group.
8. Click the 'Save' button.

To grant global permissions to all Bamboo users,

1. Click 'Home' to go to the Dashboard.
2. Click the 'Administration' link in the top navigation bar.
3. Click the 'Global Permissions' link in the left navigation column to display the 'Global Permissions' screen (see screenshot below).
4. Locate 'All logged in users' (under 'Other').
5. Select the check-box for each permission that you wish to grant to all Bamboo users.
6. Click the 'Save' button.

To grant global permissions to anonymous users,

- The 'Access' global permission is the only global permission that can be granted to anonymous users.
1. Click 'Home' to go to the Dashboard.
2. Click the 'Administration' link in the top navigation bar.
3. Click the 'Global Permissions' link in the left navigation column to display the 'Global Permissions' screen (see screenshot below).
4. Locate 'Anonymous users' (under 'Other').
5. Select the 'Access' check-box.
6. Click the 'Save' button.

To revoke global permissions,

1. Click 'Home' to go to the Dashboard.
2. Click the 'Administration' link in the top navigation bar.
3. Click the 'Global Permissions' link in the left navigation column to display the 'Global Permissions' screen (see screenshot below).
4. Locate the relevant user/group/all logged-in users/anonymous users.
5. Deselect the check-box for each permission that you wish to revoke from the user/group/all users/anonymous users.
- If you deselect all permissions for a user or group, that user or group will disappear from the 'Permissions' tab for this plan.
6. Click the 'Save' button.

Screenshot: Global Permissions
Global Permissions

You can edit your global application level permissions here. Permissions can be granted to specific users or groups. Please note these are global application permissions. For plan level permissions, please go to the plan configuration page.

Bamboo Application Permissions

Groups

<table>
<thead>
<tr>
<th>Access</th>
<th>Create</th>
<th>Admin</th>
</tr>
</thead>
<tbody>
<tr>
<td>bamboo-admin</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Other

<table>
<thead>
<tr>
<th>Access</th>
<th>Create</th>
<th>Admin</th>
</tr>
</thead>
<tbody>
<tr>
<td>All logged in users</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Anonymous users</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Grant permission to: User

Add

Save Cancel
5.09 Granting Administration Rights to a User

In Bamboo, there are two types of administrators:

- Global administrators — that is, people with the 'Admin' global permission. These people can access the Bamboo 'Administration' menu. They can also administer every plan.
- Plan administrators — that is, people with the 'Admin' and 'Edit' plan permissions. These people can administer a particular plan.

To grant global administration rights to a user,

- Either grant the 'Admin' global permission to the user explicitly (as described in 5.08 Granting Global Permissions to Users or Groups);
  OR:
  - Add the user to a group which has the 'Admin' global permission (as described in 5.06 Adding Users to and removing them from Groups).

To grant plan administration rights to a user,

- Either grant the 'Admin' and 'Edit' plan permissions to the user explicitly (as described in 5.07 Granting Plan Permissions to Users or Groups);
  OR:
  - Add the user to a group which has the 'Admin' and 'Edit' plan permissions (as described in 5.06 Adding Users to and removing them from Groups).
5.10 Allowing Anonymous Users to access Bamboo

Allowing anonymous users to access your Bamboo system means that people who aren't logged in to Bamboo will be able to perform most of the functions described in the Bamboo User's Guide (e.g. generating reports; viewing plans and build results) — subject to individual plan permissions.

Note that people who aren't logged in to Bamboo do not have a 'My Bamboo' tab on their Dashboard.

To allow anonymous users to access Bamboo,

1. Click 'Home' to go to the Dashboard.
2. Click the 'Administration' link in the top navigation bar.
3. Click the 'Global Permissions' link in the left navigation column to display the 'Global Permissions' screen (see screenshot below).
4. Locate 'Anonymous users' (under 'Other').
5. Select the 'Access' check-box.
6. Click the 'Save' button.

Anonymous users will now be able to access your Bamboo system. However, they will only be able to view plans and build results for plans where the 'Access' plan permission has been granted to 'Anonymous users'.

Screenshot: Global Permissions
5.11 Enabling or disabling Contact Details Display

If you enable contact details display to your Bamboo system, the full contact details for a user, including email address, IM address, and group, will be visible to any visitors of Bamboo. The email addresses of administrators in the 'Contact Administrators' page will also be visible.

To enable contact details display,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Security Settings' link in the left navigation column.
3. Select the 'Enable contact details to be displayed?' check-box.
4. Click the 'Save' button.

To disable contact details display,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Security Settings' link in the left navigation column.
3. Deselect the 'Enable contact details to be displayed?' check-box.
4. Click the 'Save' button.
5.12 Enabling or disabling Public Signup

If you enable signup for your Bamboo system, visitors can create their own Bamboo user accounts.

To enable signup,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Security Settings' link in the left navigation column.
3. Select the 'Enable Signup?' check-box.
4. Click the 'Save' button.
5. Log out of Bamboo and verify that the top navigation bar now contains a 'Signup' link (see screenshot below).

To disable signup,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Security Settings' link in the left navigation column.
3. Deselect the 'Enable Signup?' check-box.
4. Click the 'Save' button.

Screenshot: 'Signup'
5.13 Working with External User Repositories

You can integrate external user repositories with Bamboo:

- 5.13.1 Integrating Bamboo with Crowd
- 5.13.2 Integrating Bamboo with LDAP
  - 5.13.2.1 Configuring the Caching of your LDAP Repository
5.13.1 Integrating Bamboo with Crowd

Atlassian's Crowd identity management system can be integrated with Bamboo. Please see the document Integrating Crowd with Bamboo in the Crowd Administrator's Guide.

External User Management Configuration

If you are connecting Bamboo to an external user management system and do not have rights to update user attributes there, you will need to prevent users from being updated in Bamboo. In this case, you should ensure that the 'Read-only External User Management?' check-box is checked. The table below outlines the correct configuration for Bamboo, depending on your external user management setup:

<table>
<thead>
<tr>
<th>External User Management Setup</th>
<th>'Read-only External User Management?' check-box</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bamboo integrated with — Crowd using the Crowd database (i.e. Internal Directories)</td>
<td>Unchecked</td>
</tr>
<tr>
<td>Bamboo integrated with — Crowd connected to a read-only LDAP</td>
<td>Checked</td>
</tr>
<tr>
<td>Bamboo integrated with — Crowd connected to a read-write LDAP</td>
<td>Unchecked</td>
</tr>
<tr>
<td>Bamboo integrated with — Crowd with authentication-only delegated to LDAP.</td>
<td>Unchecked</td>
</tr>
</tbody>
</table>
5.13.2 Integrating Bamboo with LDAP

Bamboo supports LDAP integration for authentication and authorisation of LDAP users. However, please note that it is not possible to manage/administer LDAP accounts or user groups from Bamboo. Bamboo will continue to use local users and groups, even when LDAP is enabled. If you need to assign LDAP users to particular groups referenced by Bamboo (e.g. for permissions or notifications), the workaround is to assign your LDAP users to local Bamboo groups, and reference these groups rather than LDAP groups.

Also please note that:

- Once LDAP has been enabled, reverting back to local user management is not possible.
- In Bamboo versions prior to 1.2.2 it was not possible to perform XML backups of your Bamboo instance when integrated with LDAP. In Bamboo version 1.2.2, exports are possible, but user data will not be included in the export.
- Bamboo does not currently support multiple LDAP servers. If you need to connect to multiple LDAP servers, please consider using Crowd.

Before you begin

Confirm this information about your LDAP server:

1. Check your server LDAP version. Supported versions are v2 and v3. Supported LDAP servers include OpenLDAP, Microsoft Active Directory, Novell eDirectory, and any server that uses Java JNDI-LDAP mapping.
2. Your LDAP or Active Directory server must support static groups. This means that the user DNs must be stored against a membership attribute inside an LDAP group. An example of a static group is shown below:

   Dn: CN=Sales and Marketing,CN=Users,DC=ad,DC=atlassian,DC=com
   objectClass: top; group;
   cn: Sales and Marketing;
   distinguishedName: CN=Sales and Marketing,CN=Users,DC=ad,DC=atlassian,DC=com;
   name: Sales and Marketing;
   ...
   member: CN=John Smith,CN=Users,DC=ad,DC=atlassian,DC=com
   member: CN=Sally Smith,CN=Users,DC=ad,DC=atlassian,DC=com
   ...

   The membership attribute in this case is member, but this is not required. Note that the full DNs of John Smith and Sally Smith are listed. If the values against member are not full DNs, but are just usernames, then you need to add the flag `<useUnqualifiedUsernameForMembershipComparison>true</useUnqualifiedUsernameForMembershipComparison>` to your LDAP configuration. Open Directory on OS X uses this configuration.
3. You must not have an LDAP group called 'bamboo-admin'.
4. Make sure you don't have duplicate users on LDAP. In such cases, the LDAP users will take precedence over your local Bamboo users.
5. Make sure you don't have duplicate groups on LDAP as this may cause unpredictable behaviour.

Stage 1 - Backup your data

Please backup your data before attempting LDAP integration.

Stage 2 - Configure Connection Details

The LDAP server connection is specified by manually editing the file `atlassian-user.xml`.

1. Edit the file `.../webapp/WEB-INF/classes/atlassian-user.xml` and configure the connection AD or LDAP.
2. Check your configuration against the example connection details shown below.

```
<ldap key="ldapRepository" name="LDAP Repository@hecate.atlassian.com" cache="true">
  <host>hecate.atlassian.com</host>
  <port>389</port>
```
3. Please ensure that the following line is also active in your `atlassian-user.xml` (it should be there by default):

```xml
<hibernate name="Hibernate Repository" key="hibernateRepository" description="Hibernate Repository" />
```

### Stage 3 - Map LDAP Data Tree

1. To configure the mappings in `atlassian-user.xml` for either AD or LDAP, please see:
   - Mapping Active Directory
   - Mapping other LDAP servers
2. Check your configuration against the example connection details shown below.

```xml
<baseUserNamespace>dc=staff,dc=perftest,dc=atlassian,dc=private</baseUserNamespace>
<baseGroupNamespace>dc=groups,dc=perftest,dc=atlassian,dc=private</baseGroupNamespace>
<usernameAttribute>cn</usernameAttribute>
<userSearchFilter> (objectClass=inetorgperson)</userSearchFilter>
<firstnameAttribute>givenname</firstnameAttribute>
<surnameAttribute>sn</surnameAttribute>
<emailAttribute>mail</emailAttribute>
<groupnameAttribute>cn</groupnameAttribute>
<groupSearchFilter> (objectClass=groupOfNames)</groupSearchFilter>
<membershipAttribute>member</membershipAttribute>
```

### Stage 4 - Optional LDAP Settings

The following settings do not appear in the default `atlassian-user.xml` file. Their default values are as follows:

```xml
<poolingOn>true</poolingOn>
<maxSize>0</maxSize>
<initSize>10</initSize>
<prefSize>10</prefSize>
<debugLevel>none</debugLevel>
<securityProtocol>plain ssl</securityProtocol>
<authentication>simple</authentication>
<timeout>0</timeout>
<initialContextFactory>com.sun.jndi.ldap.LdapCtxFactory</initialContextFactory>
<batchSize>100</batchSize>
<timeToLive>0</timeToLive>
```

If you want to override these default values, you can specify any or all of them by adding them onto the end of the `atlassian-user.xml` file. For example, to add your own value for the `<initSize>` setting, you would add an extra line before the `</ldap>` line shown in 'Stage 3' above:
It is important that the connection pool timeout value be set to 0, as this will force Atlassian User (via the JNDI layer) to clean up lingering connections that have lived past one request. For more information about LDAP pools please see http://java.sun.com/products/jndi/tutorial/ldap/connect/config.html.

Stage 5 - External user management

Please ensure that 'Read-only External User Management' is turned ON in Bamboo before assigning LDAP users to Bamboo groups.

To disable Read-only External User Management,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Security Settings' link in the left navigation column.
3. De-select the 'Read-only External User Management' check-box.
4. Click the 'Save' button.

Stage 6 - Assigning LDAP Users to Bamboo Groups

Once Bamboo is started with 'Read-Only External User Management' disabled, you can assign LDAP users to Bamboo groups. Please see 5.06 Adding Users to and removing them from Groups. Note: the 'View Users' and 'View Groups' screens in Bamboo will currently not list all of your LDAP users/groups (please see BAM-1963 for details).

Troubleshooting

To check whether the atlassian-user.xml file is correctly configured, please run the Paddle tool to debug the LDAP configuration in your atlassian-user.xml file. For further reference, please visit the Paddle usage page.
5.13.2.1 Configuring the Caching of your LDAP Repository

This page last changed on Apr 08, 2008 by cbroadfoot.

On this page:
Error formatting macro: toc: java.lang.NullPointerException

Disabling the Caching of Users

By default, caching is activated for your LDAP users. We recommend that you do not disable caching of your LDAP users, as your LDAP repository may be overloaded by the high volume of requests by Bamboo.

To disable the caching of users,

1. Click the 'Administration' link in the top navigation bar.
2. Edit the file ...
3. Set the property cache="false" on your LDAP repository, as shown in the example below:

```xml
<ldap key= "myLdapRepository" name= "LDAP Repository@hecate.atlassian.com" cache= "false" >
  <host>hecate.atlassian.com</host>
  <port>389</port>
</ldap>
```

Enabling the Caching of Users

By default, caching is activated for your LDAP users. If you need to enable caching, follow the instructions below:

To enable the caching of users,

1. Edit the file ...
2. Set the property cache="true" on your LDAP repository as shown in the example below:

```xml
<ldap key= "myLdapRepository" name= "LDAP Repository@hecate.atlassian.com" cache= "true" >
  <host>hecate.atlassian.com</host>
  <port>389</port>
</ldap>
```

Configuring the LDAP Caches

Bamboo uses a number of caches for managing an LDAP repository, each of which can be configured differently. You must enable caching, as described above, before configuring the caches. The caches used by Bamboo are:

Error formatting macro: toc: java.lang.NullPointerException

Each cache can be configured by following the instructions below:

To configure a cache,

1. Edit the file ...
2. Find the cache that you wish to edit. Examples of each of the caches are described in the Configuring Caches for Users and Configuring Caches for User Groups sections below.
3. Modify the cache, as desired. The following properties can be configured for each cache:
• maxElementsInMemory (mandatory) - Sets the maximum number of objects that will be created in memory
• eternal (mandatory) - Sets whether elements are eternal. If eternal, timeouts are ignored and the element is never expired.
• timeToIdleSeconds (optional) - Sets the time to idle for an element before it expires. i.e. The maximum amount of time between accesses before an element expires. This is only used if the element is not eternal. A value of 0 means that an Element can idle for infinity. The default value is 0.
• timeToLiveSeconds (optional) - Sets the time to live for an element before it expires i.e. The maximum time between creation time and when an element expires. This is only used if the element is not eternal. A value of 0 means that an Element can live for infinity. The default value is 0.

If you have caching turned on Bamboo will, by default, set the cache to eternal (elements will never expire), and set the maximum number of elements stored to 500. These can be configured to speed up user retrieval, reduce memory usage or reduce the load on the LDAP repository.

Configuring Caches for Users

⚠ In each of the examples below, replace myLdapRepository with the key of the repository specified in atlassian-user.xml

• LDAPUserManagerReadOnly.*.users

LDAPUserManagerReadOnly.*.users stores the individual users, if you have difficulties with Bamboo picking up new user additions in the LDAP repository you will need to alter the configuration of this cache. In the example below, the users will expire after 5 minutes.

```xml
<cache name= "com.atlassian.user.impl.ldap.LDAPUserManagerReadOnly.myLdapRepository.users"
  maxElementsInMemory="500"
  eternal="false"
  timeToIdleSeconds="300"
  timeToLiveSeconds="300"
/>
```

• LDAPUserManagerReadOnly.*.users_ro

LDAPUserManagerReadOnly.*.users_ro stores whether or not the users are read only. This will have no effect on the functionality of Bamboo, but you may wish to modify this cache for performance and memory tuning purposes.

```xml
<cache name= "com.atlassian.user.impl.ldap.LDAPUserManagerReadOnly.myLdapRepository.users"
  maxElementsInMemory="500"
  eternal="false"
  timeToIdleSeconds="300"
  timeToLiveSeconds="300"
/>
```

• LDAPUserManagerReadOnly.*.repositories

LDAPUserManagerReadOnly.*.repositories stores which repository the user belongs to. Bamboo does not yet support multiple repositories, so modifying this cache will have no effect on functionality. However, you may wish to modify this cache for performance and memory tuning purposes.

```xml
<cache name= "com.atlassian.user.impl.ldap.LDAPUserManagerReadOnly.myLdapRepository.users"
  maxElementsInMemory="500"
  eternal="false"
  timeToIdleSeconds="300"
  timeToLiveSeconds="300"
/>
Configuring Caches for User Groups

⚠️ In each of the examples below, replace myLdapRepository with the key of the repository specified in atlassian-user.xml

- **LDAPUserManagerReadOnly.*.repositories**
  
  LDAPGroupManagerReadOnly.*.groups stores the available groups in LDAP. If you wish Bamboo to pick up changes made to groups, then you will need to configure this cache appropriately.

```xml
<cache name="com.atlassian.user.impl.ldap.LDAPGroupManagerReadOnly.myLdapRepository.groups"
  maxElementsInMemory="500"
  eternal="false"
  timeToIdleSeconds="300"
  timeToLiveSeconds="300"
/>
```

- **LDAPGroupManagerReadOnly.*.groups_hasMembership* and *LDAPGroupManagerReadOnly.*.groups_getGroupsForUser**

  The LDAPGroupManagerReadOnly.*.groups_hasMembership and LDAPGroupManagerReadOnly.*.groups_getGroupsForUser caches store the associations between users and groups. If you wish Bamboo to pick up changes made to group memberships then you will need to configure these caches appropriately.

```xml
<cache
  name="com.atlassian.user.impl.ldap.LDAPGroupManagerReadOnly.myLdapRepository.groups_hasMembership"
  maxElementsInMemory="500"
  eternal="false"
  timeToIdleSeconds="300"
  timeToLiveSeconds="300"
/>
<cache
  name="com.atlassian.user.impl.ldap.LDAPGroupManagerReadOnly.myLdapRepository.groups_getGroupsForUser"
  maxElementsInMemory="500"
  eternal="false"
  timeToIdleSeconds="300"
  timeToLiveSeconds="300"
/>
```

- **LDAPGroupManagerReadOnly.*.repositories**

  LDAPGroupManagerReadOnly.*.repositories stores which repository the group belongs to. Bamboo does not yet support multiple repositories, so modifying this cache will have no effect on functionality. However, you may wish to modify this cache for performance and memory tuning purposes.

```xml
<cache name="com.atlassian.user.impl.ldap.LDAPGroupManagerReadOnly.myLdapRepository.groups"
  maxElementsInMemory="500"
  eternal="false"
  timeToIdleSeconds="300"
  timeToLiveSeconds="300"
/>
```
06. Configuring Plugins

This page last changed on Dec 12, 2007 by alui.

6. Configuring Plugins

- 6.1 About Bamboo Plugins
- 6.2 Enabling the 'Auto-Favourites' Plugin
- 6.3 Installing a new Plugin
6.1 About Bamboo Plugins

A Bamboo plugin is a program that provides a piece of Bamboo functionality. Bamboo comes with some preinstalled plugins, including:

- 'AutoFavourites' plugin
- 'JIRA' plugin
- 'NAnt Builder' plugin

Additional plugins can be created (see the Bamboo Plugin Developer's Guide) or obtained from the library, and installed into your Bamboo system.
6.2 Enabling the 'Auto-Favourites' Plug

This page last changed on Dec 12, 2007 by alui.

If the 'Auto-Favourites' plugin is enabled, then a plan will be automatically added to a user's 'My Bamboo' tab when the user checks-in code to the plan's source-code repository, and the user will become a 'watcher' of that plan (i.e. depending on how the plan's notifications are configured, they may receive notifications about the plan's build results).

To enable the 'Auto-Favourites' plugin,

1. Click the 'Administration' link in the top navigation bar.
2. Click the link 'Configure Auto-Favourites Plugin' in the left-hand column.
3. Tick the box 'Enable Auto-Favourite Plugin'.
4. Click the 'Save' button.

Each plan will now be automatically added to a user's favourites the first time the user checks-in code to the plan's source-code repository. Note that, if the user removes the plan from their favourites, it will not be automatically added again.
6.3 Installing a new Plugin

To install a new plugin,

1. Copy the plugin (i.e. JAR file) into the following directory:
   - If you are using Bamboo Standalone: ../<Bamboo-install>/webapp/WEB-INF/lib/
   - or:
   - If you are using the Bamboo EAR-WAR distribution: ../<Bamboo-Deploy-Location>/WEB-INF/lib/ (e.g. if you are running Bamboo EAR-WAR on Tomcat, copy your plugin to the ../<Tomcat-Install>/webapps/<Bamboo-Deploy-Location>/WEB-INF/lib/ directory).
2. Restart Bamboo.

A number of plugins are available from the library. You can also create your own as described in the Bamboo Plugin Developer's Guide.
07. Managing Data and Backups

7. Managing Data and Backups

- 7.1 Locating Important Directories and Files
- 7.2 Specifying Bamboo's Working Directory
- 7.3 Viewing your Database Connection Details
- 7.4 Moving your Bamboo Data to a different Database
- 7.5 Optimising or Re-indexing Data
- 7.6 Enabling Expiry of Build Results
- 7.7 Specifying a Backup Schedule
- 7.8 Exporting Data for Backup
- 7.9 Importing Data from Backup
7.1 Locating Important Directories and Files

This page last changed on Mar 16, 2008 by alui.

BAMBOO SERVER

When you installed your Bamboo server, you specified two directories:

- **Bamboo installation directory** — This is the directory where the Bamboo application files are installed. (The default location depends on your operating system: Windows, Unix/Linux, Solaris or Mac OS.)
- **Bamboo home directory** — This is the directory where your Bamboo configuration data and build results are stored. (The default location depends on your operating system: Windows, Unix/Linux, Solaris or Mac OS.) This directory can grow quite large when managing large quantities of plans and builds.

The most important contents of these two directories are described below.

**Bamboo home directory**

- **bamboo.cfg.xml** — This is Bamboo's core configuration file. It includes the configuration information for connecting to Bamboo's database.
- **xml-data/** — This directory contains all files relating to source repositories and build results.
  - **xml-data/build-dir/** — This is known as the Working Directory. This is where Bamboo temporarily puts the checked-out files it is building. The location of this directory was specified via the Setup Wizard, can be viewed as described in 8.1 Viewing Bamboo's System Information, and can be changed as described in 7.2 Specifying Bamboo's Working Directory.
  - **xml-data/builds/** — This is known as the Build Directory. This is where Bamboo stores build results and artifacts (note that they will be deleted as described in 7.6 Enabling Expiry of Build Results). The location of this directory was specified via the Setup Wizard, and can be viewed as described in 8.1 Viewing Bamboo's System Information. Its contents can be backed up as per 7.8 Exporting Data for Backup.
    - **xml-data/builds/PLAN_KEY/results** — Contains the build results for all the builds belonging to the 'PLAN_KEY' plan. Each build result is an individual XML file. Do not edit these files or the corresponding information in the database may become corrupt.
    - **xml-data/builds/PLAN_KEY/download-data** — Contains the artifacts and logs for each build belonging to the 'PLAN_KEY' plan.
  - **xml-data/configuration/** — This is known as the Configuration Directory. It contains server-wide configuration information. The location of this directory was specified via the Setup Wizard, and can be viewed as described in 8.1 Viewing Bamboo's System Information. Its contents can be backed up as per 7.8 Exporting Data for Backup.
- **database/** — This directory contains Bamboo's embedded HSQL database. The database contains plan configurations and some build results data.
- **index/** — This directory contains the build results index. Removing or modifying files in this directory may corrupt build history. Rebuilding the search index from Bamboo's global administration screen (see 7.5 Optimising or Re-indexing Data) will completely regenerate the contents of this directory.

**Bamboo installation directory**

- **webapp/WEB-INF/classes/bamboo-init.properties** — This file tells Bamboo where to find the Bamboo home directory. The location of this directory is specified by the Bamboo administrator as described in the Bamboo Installation Guide, and can be viewed as described in 8.1 Viewing Bamboo's System Information.
- **bamboo.sh** — This is the startup file for Bamboo Standalone under Unix/Linux, Solaris and Mac OS.
- **bamboo.bat** — This is the startup file for Bamboo Standalone under Windows.
- **bamboo.pid** — This file, under Linux, contains the Process ID for the running instance of Bamboo.
- **conf/wrapper.conf** — This file provides the means to configure Bamboo on startup, when using the Java Service wrapper under Linux or Windows.
- **scripts/** — This directory contains operational scripts, including scripts for CVS and SVN triggers.
- **wrapper/** — This directory contains the necessary files to start Bamboo using the Java Service wrapper (see the Mac and Linux installation guides).
• logs*/--- This directory contains logs written by the Java Service wrapper. (Note: The Bamboo server logs are written to the root of the installation directory. Build logs are stored in the xml-data/builds/ sub-directories.)
• webapp/ — This directory contains all the Bamboo server application files.
• webapp/WEB-INF/lib/ — This directory is used when deploying Bamboo plugins. It also contains other libraries required by Bamboo.
• webapp/WEB-INF/classes/log4j.properties — This is Bamboo's logging configuration file.
• webapp/WEB-INF/classes/jetty.xml* — This is the configuration file for Jetty, the application server that is bundled with Bamboo Standalone.

* This applies to the Bamboo Standalone distribution. The configuration may differ for the Bamboo EAR-WAR distribution.

**BAMBOO AGENT**

When you installed your remote agents (if any), you specified the following directory:

• Agent home directory — This is the directory where the agent's configuration data is stored. The default name of this directory is bamboo-agent-home. (The default location depends on your operating system: Windows, Unix/Linux, Solaris or Mac OS.) This directory can grow quite large when managing large quantities of plans and builds.

The contents of the agent home directory are:

• bamboo-agent.cfg.xml — This contains configuration information about this remote agent. Most notably, it stores the agent id, which gets generated the first time this agent connects to the Bamboo server.
• xml-data/
  • xml-data/build-dir/ — This is where the agent will check out the files and peform builds (similar to the Bamboo server's xml-data/build-dir/ directory)
7.2 Specifying Bamboo's Working Directory

The working directory is where Bamboo temporarily puts the checked-out files it is building. By default, this directory is located under the `xml-data` directory in the Bamboo home directory.

To change the location of Bamboo's working directory,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'General Configuration' link in the left navigation column.
3. This will display the 'General Configuration' page. In the 'Working Directory' field, type the absolute path of your Bamboo server (for example, "C:/bamboo-home/xml-data/build-dir").
4. Click the 'Save' button.
7.3 Viewing your Database Connection Details

When you installed Bamboo, you would have set up a database connection by following one of these processes:

- **1. Using Bamboo's embedded HSQL database**
- **2. Connecting Bamboo to an external database**
  - 2.1 MySQL 4.1 and 5.0
  - 2.2 Postgres 8+
  - 2.3 Oracle 9i and 10g
  - 2.4 Microsoft SQL Server
  - 2.5 Unsupported databases

Once Bamboo is running, you can view the database configuration details as follows.

To view your database connection details,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Database Configuration' link in the left navigation column.

Screenshot: 'Database Configuration'

<table>
<thead>
<tr>
<th>Database Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Driver Class Name:</strong></td>
</tr>
<tr>
<td><strong>Database URL:</strong></td>
</tr>
<tr>
<td><strong>User Name:</strong></td>
</tr>
<tr>
<td><strong>Hibernate Dialect:</strong></td>
</tr>
</tbody>
</table>
When you installed Bamboo, you would have set up a database connection by following one of these processes:

1. Using Bamboo's embedded HSQL database
2. Connecting Bamboo to an external database
   - 2.1 MySQL 4.1 and 5.0
   - 2.2 Postgres 8+
   - 2.3 Oracle 9i and 10g
   - 2.4 Microsoft SQL Server
   - 2.5 Unsupported databases

You may later wish to use a different database. For example, the embedded HSQL database is suitable for evaluation purposes only — you would typically move to an external database before deploying Bamboo in production.

To move your Bamboo data to a different database,

1. Backup your Bamboo data as described in 7.8 Exporting Data for Backup. Note the filename and path of the exported file for use in Step 8 below.
2. Shut down your old instance of Bamboo.
3. If your old instance of Bamboo was configured to start automatically (e.g. as a Windows service), disable it.
4. Install a new instance of Bamboo as described in the Bamboo Installation Guide. Specify a different Home Directory* and Installation Directory* from the directories used by your old instance of Bamboo. (If you use the same locations, your existing data will be deleted.)
5. Launch your new instance of Bamboo. You will see the Setup Wizard.
6. At Step 1 of the Setup Wizard, ensure that your new Configuration Directory*, Build Data Directory* and Build Working Directory* are in different locations to your old instance of Bamboo.
7. At Step 2 of the Setup Wizard, select your new database and follow the appropriate instructions for your chosen database:
   - 2.1 MySQL 4.1 and 5.0
   - 2.2 Postgres 8+
   - 2.3 Oracle 9i and 10g
   - 2.4 Microsoft SQL Server
   - 2.5 Unsupported databases
8. At Step 3 of the Setup Wizard (see screenshot below), select 'Import existing data' and specify the export file created in Step 1 above.
9. Wait while Bamboo imports your data. (You will not need to complete any more steps of the Setup Wizard.)
10. When the data import has finished, restart your new instance Bamboo.
11. Re-index your Bamboo data as described in 7.5 Optimising or Re-indexing Data.
12. Verify that your build results and system settings look the same as before.

*For information about the contents of these directories, please see 7.1 Locating Important Directories and Files

Screenshot: Setup Wizard — Step 3: 'Starting Data'
Starting Data

Select Starting Data for Bamboo

Would you like to:  
- Create new Bamboo home
- Import existing data

File path:  
Specify the absolute path to the file on the server from which Bamboo is to import.

Continue
7.5 Optimising or Re-indexing Data

This page last changed on Dec 13, 2007 by alui.

About optimising

You may want to optimise your Bamboo build results data if you notice that search-intensive operations (e.g. reporting) are becoming slow.

⚠️ Bamboo will still be accessible while the optimisation process is running.

About re-indexing

You will need to re-index your Bamboo build results data whenever you perform a data import. Re-indexing your data can also be helpful if your reports appear to be out-of-sync with your data.

⚠️ Bamboo will not be accessible while the re-indexing process is running. This may take a few minutes to complete (see System Information for an estimate of how long it will take).

To re-index Bamboo's build results data,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Indexing' link in the left navigation column.
3. The 'Optimise or Re-index Bamboo' screen will appear. Select 'Full re-index' and click the 'Perform' button.

To optimise Bamboo's build results data,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Indexing' link in the left navigation column.
3. The 'Optimise or Re-index Bamboo' screen will appear. Select Optimise current index' and click the 'Perform' button.
7.6 Enabling Expiry of Build Results

By enabling build expiry, you can choose how much build results data will be kept in your Bamboo system, and for how long it will be kept (e.g. for reporting purposes), before being automatically deleted. If you disable build expiry, your build results will never be automatically deleted from Bamboo.

You can enable/disable build expiry for:

- all plans (as described below). This is generally the easiest way to manage your build expiry. Your settings will apply to all plans that do not have individual expiry settings.
- individual plans (see Specifying Expiry for a Plan's Build Results). You would generally only do this if there is a specific reason to keep/delete a particular plan's build results.

Note that you can also build results manually — see 1.4.1 Deleting a Build Result.

⚠️ If you enable build expiry, ensure that you back up your build results data before its expiry date is reached.

To enable expiry for build results data,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Build Expiry' link in the left navigation column.
3. The 'Build Expiry' screen will be displayed. Click the 'Enable' button.
4. The 'Build Expiry Settings' screen will be displayed (see screenshot below).
5. In the 'What should be expired' section, select what type of build results data you want to delete:
   - 'Artifacts' — select this if you want to delete all user-defined artifacts but keep all other build results data.
   - 'Build results' — select this if you want to delete all build results data (including artifacts).
6. Choose one of these three methods for specifying how much data to keep:
   - To keep all build results up to a certain age,
     - In the 'Expiry period' field, specify the number of months/weeks/days for which you want to keep your build results, e.g. specify '24 months' to keep all build results for the last two years.
     - In the 'Minimum builds to keep' field, specify '0'.
   - To keep a certain number of build results per plan,
     - In the 'Expiry period' field, specify '0'.
     - In the 'Minimum builds to keep' field, specify the number of build results you want to keep, e.g. specify '50' to keep the latest 50 build results for each plan.
   - To keep all build results up to a certain age, and a certain number of build results per plan,
     - In the 'Expiry period' field, specify the number of months/weeks/days for which you want to keep your build results data, e.g. specify '24 months' to keep all build results for the last two years.
     - In the 'Minimum builds to keep' field, specify the number of build results you want to keep, e.g. specify '50' to keep the latest 50 build results for each plan. (This means that, even if all of a plan's builds are over two years old, the last 50 build results will not be deleted.)
7. In the 'Labels to keep' field, specify any labels for which you always want to keep labelled builds. (If you want to specify more than one label, use spaces to separate them.) For any label(s) that you specify, all builds that have a matching label will never be deleted, regardless of the method you followed in step 6 above.
   - Note: builds can either be labelled:
     - manually, as described in 8.2 Labelling a Build Result in the Bamboo User's Guide; or
     - automatically, as described in Specifying Labels for a Plan’s Build Results in the Bamboo Administrator's Guide.
8. Click the 'Save' button.

Screenshot: 'Enable Build Expiry'
To disable expiry for build results data,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Build Expiry' link in the left navigation column.
3. The 'Build Expiry' screen will be displayed (see screenshot below). Click the 'Disable' button.

Screenshot: 'Disable Build Expiry'
7.7 Specifying a Backup Schedule

This page last changed on Dec 13, 2007 by alui.

You can configure Bamboo to automatically create a backup each night, rather than doing a manual export every time.

Before you begin:

- Bamboo will be unavailable while the backup process completes. Depending on the number of builds and tests, the export may take a long time to complete and may require large amounts of disk space. Please make sure you have enough disk space before proceeding.
- Bamboo will not export if it detects that plans are currently being built. Please ensure that you do not have plans building (see 1.2 Viewing Bamboo's Current Activity).
- There is a bug in Bamboo 1.2 which causes the dependency associations to not be exported along with the plans. If you are using Bamboo 1.2, please upgrade to Bamboo 1.2.1 for a fix.

To specify a backup schedule,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Scheduled Backups' link in the left navigation column.
3. The 'Scheduled Backups' screen will be displayed. Click the 'Edit' button.
4. In the 'Backup path' field, specify the directory where you want to store your backups. Each backup will be stored as a single file.
   - Note that backups may require large amounts of disk space. Please make sure you have enough disk space before proceeding.
5. In the 'Backup file prefix' field, specify the first part of the filename for all your backup files.
6. In the 'Backup file date pattern' field, specify the date/time format for identifying your individual backup files. This will be appended to the 'Backup file prefix' to form the complete filename for your backup files, e.g.:
   - if your 'Backup file prefix' is 'bamboo_backup_' and your 'Backup file date pattern' is 'yyyy_MM_dd', then your backup file on 31st July 2007 would be named 'bamboo_backup_2007_07_31'.
7. In the 'Backup Frequency' field, specify how often (in days) you want Bamboo to create a backup, e.g.:
   - specify '1' to create a backup every day.
   - specify '7' to create a backup every week.
8. In the 'Backup Time' field, specify the time of day at which you want Bamboo to create a backup. Use 24-hour format, e.g. to create a backup at 8.30 PM you would specify '20:30'.
   - Note that Bamboo will be unavailable for the duration of the backup, so choose a time of day or night when usage is low.
9. Select the 'Backup Artifacts' check-box if you want to backup your build artifacts.
10. Click the 'Save' button.
   - Your first backup will run when your server's clock matches the time specified in the 'Backup Time' field.

To disable a scheduled backup,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Scheduled Backups' link in the left navigation column.
3. The 'Scheduled Backups' screen will be displayed, showing your current backup schedule. Click the 'Edit' button.
4. Select the 'Disable scheduled backups' check-box.
   - Your schedule details will be retained, but no automatic backups will be performed.
5. Click the 'Save' button.

Screenshot: 'Scheduled Backups'
Scheduled Backups

Configure Bamboo to generate a backup at specific time intervals.

Scheduled Backup Details

- Please choose the schedule for your backups carefully. Bamboo will be unavailable for the duration of the backup process.

- **Backup path:**
  - bamboo/bamboo-home/backups

- **Backup file prefix:**
  - bamboo_backup

- **Backup file date pattern:**
  - yyyyMMdd

- **Backup Frequency:**
  - Every 7 days

- **Backup Time (HH:mm):**
  - 01:00

- **Backup Artifacts:**
  - Should generate backup with artifacts

[Save] [Cancel]
7.8 Exporting Data for Backup

Before you begin:

- Bamboo will be unavailable while the export process completes. Depending on the number of builds and tests, the export may take a long time to complete and may require large amounts of disk space. Please make sure you have enough disk space before proceeding.
- Bamboo will not export if it detects that plans are currently being built. Please ensure that you do not have plans building (see 1.2 Viewing Bamboo's Current Activity).
- There is a bug in Bamboo 1.2 which causes the dependency associations to not be exported along with the plans. If you are using Bamboo 1.2, please upgrade to Bamboo 1.2.1 for a fix.
- Bamboo uses the third party TrueZip library to create zip archives. TrueZIP currently implements the ZIP32 specification only. This limits the maximum ZIP file length to 4GB. Unfortunately, Bamboo exports will fail if the resulting ZIP file is over 4GB. As a workaround, please export Bamboo without artifacts.
- Exporting Bamboo doesn't work with Bamboo versions prior to 1.2.2, if LDAP is turned on. If you want to export Bamboo data, please turn off LDAP. Please see 5.13.2 Integrating Bamboo with LDAP for further details.

To export data for backup,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Export' link in the left navigation column.
3. Type the absolute 'File Path' to which Bamboo is to export data. For example, 'c:/Documents and Settings/<me>/bamboo/bamboohome/manual_backups/export.zip'.
4. Select the 'Export Artifacts' check-box if you want to backup your build artifacts.
5. Click the 'Export' button.

Screenshot: Export
7.9 Importing Data from Backup

Before you begin

- The import process will DELETE this instance and restore data from a previous export of Bamboo. This includes login data, hence you will need an administration login that is contained in the Bamboo data to be imported.
- Bamboo will be unavailable until the import process is complete, which may take some time.

To import data from backup,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Import link in the left navigation column.
3. Type the absolute 'File Path' from which Bamboo is to import data. For example, "/opt/bamboo/bamboohome/export.zip".
4. Select the 'Backup Data' check-box (HIGHLY RECOMMENDED).
5. Specify the absolute 'File path of backup' to which Bamboo should backup data (note that this must be different from the 'File Path' above). For example, "/opt/bamboo/bamboohome/backup.zip".
6. Click the 'Import' button.
7. After the import is complete,
   - check the paths of your builders and JDK.
   - index your data.
08. Configuring System Settings

This page last changed on Feb 05, 2007 by rosie@atlassian.com.

8. Configuring System Settings

• 8.1 Viewing Bamboo's System Information
• 8.2 Updating your Bamboo License Details
• 8.3 Specifying Bamboo's Title
• 8.4 Specifying Bamboo's URL
• 8.5 Enabling GZIP Compression
• 8.6 Enabling Bamboo's Remote API
• 8.7 Enabling JIRA integration
8.1 Viewing Bamboo's System Information

When you installed Bamboo, you provided information about how the system should be configured.

To view your Bamboo system information,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'System Info' link in the left navigation column.

This page contains useful information for you to send to Atlassian when requesting support.

Screenshot: 'System Information'

For information about the 'Bamboo Paths' please see 7.1 Locating Important Directories and Files.
8.2 Updating your Bamboo License Details

When you upgrade or renew your Bamboo license, you will receive a new license key. You will need to update your Bamboo server with the new license key.

To update your Bamboo license key,

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Log into Bamboo as a user with admin access.</td>
</tr>
<tr>
<td>2.</td>
<td>Click the 'Administration' link in the top navigation bar.</td>
</tr>
<tr>
<td>3.</td>
<td>Click the 'License Details' link in the left navigation column.</td>
</tr>
<tr>
<td>4.</td>
<td>This will display your existing Bamboo license details, and an empty box called 'License Key'. Paste your new license into this box.</td>
</tr>
<tr>
<td>5.</td>
<td>Click the 'Save New License' button.</td>
</tr>
</tbody>
</table>

Licensing Questions?
Please see the [Licensing FAQ](mailto:).
8.3 Specifying Bamboo's Title

Bamboo's name is the displayed title of this installation of Bamboo. It will appear throughout Bamboo (e.g. on the Dashboard), and in the window-title of your users' browsers.

To specify Bamboo's title,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'General Configuration' link in the left navigation column.
3. This will display the 'General Configuration' page. In the 'Name field, type the display title for your Bamboo server (e.g. "MyCompany's Bamboo").
4. Click the 'Save' button.
8.4 Specifying Bamboo's URL

This page last changed on Dec 13, 2007 by alui.

This is the base URL of this installation of Bamboo. All links created (for emails etc) will be prefixed by this URL.

To specify Bamboo's URL,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'General Configuration' link in the left navigation column.
3. This will display the 'General Configuration' page. In the 'Base URL' field, type the URL address of your Bamboo server (for example, "http://keg:8080/bamboo").
4. Click the 'Save' button.
8.5 Enabling GZIP Compression

You can enable GZIP compression in order to reduce the size of Bamboo's web pages. This is useful if Bamboo is being run over slow networks. There is a slight performance penalty, and note that GZIP may not work for languages other than English.

To enable GZIP Compression,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'General Configuration' link in the left navigation column.
3. Select the 'Apply gzip compression to reduce the size of Bamboo's web pages?' check-box.
4. Click the 'Save' button.
8.6 Enabling Bamboo's Remote API

You can access Bamboo's data from an external program by using Bamboo's REST-style remote API. The remote API is disabled by default. Bamboo will return an error if people try to use the remote API when it is disabled.

To enable the remote API,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'General Configuration' link in the left navigation column.
3. Select the 'Accept remote API calls?' check-box.
4. Click the 'Save' button.
5. Bamboo will now accept remote calls. You do not have to restart the Bamboo server.

⚠️ Looking for Remote API documentation?
See Bamboo API
8.7 Enabling JIRA integration

This page last changed on Dec 13, 2007 by alui.

When Bamboo's JIRA integration plugin is enabled, Bamboo can provide greater visibility of the issue tracking cycle by automatically linking JIRA issues (in commit messages) to Bamboo builds.

- In JIRA 3.7 and later, Bamboo will display issue numbers and issue details, and create links to each JIRA issue.
- In JIRA 3.6 and below, Bamboo will display issue numbers and create links to each JIRA issue.

Note that this will require your Bamboo server to login to your JIRA server, regardless of whether your JIRA server is running in 'public' or 'private' mode.

⚠️ Before you begin:
Ensure that the Remote API option is enabled on your JIRA server.

To enable the JIRA integration plugin,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'JIRA Server' link in the left navigation column.
3. In the 'Host URL' field, type the URL address of your JIRA server (e.g. 'http://jira.atlassian.com').
4. In the 'Username' field, type the name of the JIRA account which your Bamboo server will use to login to your JIRA server. This JIRA account does not require JIRA administration permission.
5. In the 'Password' field, type the corresponding password for the JIRA account you specified in step 4.
6. (Only perform steps 6 and 7 if you are running JIRA 3.7 or later) In the 'Test' section, type a JIRA issue key in the 'Issue Key' field (e.g. 'BAM-738').
7. Click the 'Test' button. This should display the following message: 'Successfully retrieved JIRA issue from remote server'. If not, check that you can login to your JIRA server using the JIRA account and password you specified in steps 4 and 5.
8. When the test is successful, click the 'Save' button.

Two-way integration is available

JIRA's Bamboo plugin allows JIRA users to view the relevant Bamboo builds for a JIRA issue, from within JIRA.
09. Bamboo Security

This page last changed on Feb 03, 2008 by alui.

Application Security Overview

As a distributed application, Bamboo's application-level security is important. This document contains links to version-specific security advisories for the Bamboo application.

This document is intended to provide information to system administrators about the security of the Bamboo application. It does not address Bamboo's internal security model – user management and permissions – except as it relates to the overall application security.

Vulnerabilities, Advisories and Patches.

If you find a security bug in Bamboo

Open an issue on http://jira.atlassian.com in the Bamboo project.

- Set the priority of the bug to "Blocker"
- Provide as much information on reproducing the bug as possible
- Set the security level of the bug to "Developer and Reporters only"

All communication about the vulnerability should be performed through JIRA, so we can keep track of the issue and get a patch out as soon as possible.

Bamboo Security Advisories

When a security issue in Bamboo is discovered and resolved, we will inform customers through the following mechanisms:

- A security advisory will be posted on this page
- A copy of the advisory will be sent to the bamboo-users and bamboo-announce mailing-lists (subscribe here). These lists are mirrored on our forums
- If the person who reported the issue wants to publish an advisory through some other agency (for example, CERT), we'll assist in the production of that advisory, and link to it from our own.

Latest security advisory:

- Bamboo Security Advisory 2008-02-08 (Bamboo 2.0 Beta)

Our Patch Policy

When a security issue is discovered, we will endeavour to:

- issue a new, fixed Bamboo version as soon as possible
- issue a patch to the current stable version of Bamboo
- issue patches for older versions of Bamboo if feasible

Patches will generally be attached to the relevant JIRA issue.

Past Security Advisories

- Bamboo Security Advisory 2008-02-08 (Bamboo 2.0 Beta) — The Bamboo 2.0 Beta does not include the security features that will be present in the final released product. Please note the following security implications when enabling Bamboo's remote agent functionality:
- Securing your Remote Agents
Bamboo Security Advisory 2008-02-08 (Bamboo 2.0 Beta)

In this advisory:

Error formatting macro: toc: java.lang.NullPointerException

Bamboo 2.0 Beta Security Considerations

Risk Assessment

The Bamboo 2.0 Beta does not include the security features that will be present in the final released product. Please note the following security implications when enabling Bamboo's remote agent functionality:

- No encryption of data passed between server and agent — this includes data such as:
  - login credentials for version control repositories
  - build logs
  - build artifacts
- No authentication of the agent or server — this could result in unauthorised actions being taken on your system, such as:
  - Unauthorised parties installing new remote agents — version control repository login credentials could be stolen.
  - Unauthorised parties masquerading as a Bamboo server — the unauthorised server could pass malicious code to the agent to run.

We strongly recommend that you do not enable remote agent installation on any Bamboo instance accessible from a public or untrusted network. Creating remote agents is disabled by default. These are limitations of the beta release only and will be addressed before the final released product.

Vulnerability

An unauthorised party could steal sensitive data passing between the Bamboo server and agents or run malicious code on your agents, as described in the 'Risk Assessment' section.

Fix

These are limitations of the beta release only and will be addressed before the final released product.
Securing your Remote Agents

This page last changed on Apr 08, 2008 by alui.

We strongly recommend that you do not enable remote agent installation without securing them on any Bamboo instance accessible from a public or untrusted network. Creating remote agents is disabled by default. If you choose to enable your remote agents without securing them, please read this Security Advisory to understand the security implications.

You can secure your remote agents by configuring them to use SSL (Secure Sockets Layer). This protocol provides a secure mechanism for communication between your Bamboo server and remote agents. The information below describes how to configure your remote agents to use SSL.

⚠ Please note that you cannot set up client certificates in Bamboo due to limitations with ActiveMQ. Thus, while encryption works both ways between the server and client in this recommended configuration, authentication is only one-way. That is, the clients (i.e. agents) can authenticate the server, but the server will not be able to authenticate the clients (i.e. agents).

On this page:

Error formatting macro: toc: java.lang.NullPointerException

Step 1. Create keys, stores and certificates

The first step in configuring your remote agents to use SSL is to create the required keys, stores and certificates. These artefacts are created using a keytool, as described below:

SSL relies on keys being set up on your server and clients (i.e. agents). To securely store these keys, keystores (databases of keys) need to be created. A certificate is then created by the server (and optionally on the clients, but not for this configuration) to allow publication of the server's key. To establish that the client "trusts" the server, this server certificate is then imported into a truststore (key database file that contains the public keys for a specific server) created on the client.

To create the required keys, stores and certificates for your server and agents,

1. Using a keytool, create a certificate for your server by entering the following command:

```shell
keytool -genkey -alias server -keyalg RSA -keystore server.ks
```

2. The server’s certificate will be created. Export the certificate, so it can be shared with clients, by entering the following command:

```shell
keytool -export -alias server -keystore server.ks -file server_cert
```

3. Each client should now be able to access the server’s certificate. Create a keystore for each client, by entering the following command:

```shell
keytool -genkey -alias client -keyalg RSA -keystore client.ks
```

4. Create a truststore for each client and import the server's certificate, by entering the command below. This establishes that the client "trusts" the server:

```shell
keytool -import -alias server -keystore client.ts -file server_cert
```
Step 2. Tell your Bamboo server and agents where to find the stores

The second step in configuring your agents to use SSL is to instruct your Bamboo server and agents to use the keystores and truststores that you have just created.

**To tell your server where to find the keystore,**

1. Add the system properties 'javax.net.ssl.keyStore=/path/to/broker.ks' and 'javax.net.ssl.keyStorePassword=password' to your VM, by carrying out any of the following three steps:
   - (Standalone or Non-Standalone instances of Bamboo) Set the SSL_OPTS environment variable to hold the 'javax.net.ssl.keyStore=/path/to/broker.ks' and 'javax.net.ssl.keyStorePassword=password' properties.
     e.g.
     ```
     export SSL_OPTS = -Djavax.net.ssl.keyStore=/path/to/broker.ks -Djavax.net.ssl.keyStorePassword=password
     ```
   - (Standalone instances of Bamboo only) Add 'javax.net.ssl.keyStore=/path/to/broker.ks' and 'javax.net.ssl.keyStorePassword=password' as additional properties to the wrapper.conf file.
     e.g.
     ```
     wrapper.java.additional.4=-Djavax.net.ssl.keyStore=/path/to/broker.ks
     wrapper.java.additional.5=-Djavax.net.ssl.keyStorePassword=password
     ```
   - (Non-Standalone instances of Bamboo only) Make the 'javax.net.ssl.keyStore=/path/to/broker.ks' and 'javax.net.ssl.keyStorePassword=password' properties visible to the VM, as per the instructions for your webserver.

**To tell your agents where to find the keystore and truststore,**

For each agent,

1. Tell your agent where to find the keystore and the trust store, by executing the following command to run the agent,

   ```
   java -jar bamboo-agent-2.0-SNAPSHOT.jar <agentserverURL>
   ```

   including the following command line parameters,

   ```
   -Djavax.net.ssl.keyStore=/path/to/client.ks
   -Djavax.net.ssl.keyStorePassword=password
   -Djavax.net.ssl.trustStore=/path/to/client.ts
   ```

   where <agentserverURL> is the URL of the agent's server, e.g.

   ```
   http://192.168.3.235:8085/agentServer/
   ```

For example,
Step 3. Configure your Bamboo server to use SSL

Once the server and agents know where to find the keystores and truststores, the final step is to instruct your Bamboo server to start using SSL so that agents will be able to authenticate the server.

To configure your Bamboo server to use SSL,

If you are setting up Bamboo for the first time,

1. Launch the Bamboo Setup Wizard and change the protocol of the 'Broker URL' to 'SSL'.
   i.e. ssl://host:port/

Or, if you are configuring an existing installation of Bamboo,

1. Shut down your Bamboo server and agents.
2. Change the protocol of your 'Broker URL' in the bamboo.cfg.xml file to 'SSL'. Note, do not change the address of this URL.
   e.g. <property name="bamboo.jms.broker.uri">ssl://myhost:myport?
        wireFormat.maxInactivityDuration=0</property>
3. Start up the Bamboo server.
4. Start up the Bamboo agents. If your agents do not start up, please check that you have set up your certificates correctly.
Appendix A. Embedding Bamboo into Other Applications

- Javascript Widgets
  - All Plans & My Favourite Plans
  - Latest Builds
  - Latest Status of a Plan
  - My Latest Changes
  - Plan Summary Graphs
Javascript Widgets

Bamboo has a number of widgets which can be used by external applications:

- All Plans & My Favourite Plans
- Latest Builds
- Latest Status of a Plan
- My Latest Changes
- Plan Summary Graphs
All Plans & My Favourite Plans

These widgets retrieve a summary of plans for a particular Bamboo instance. The summary is shown as a list and depicts the current status, the last completed builds and the reason for the last build. You can either show all the plans or just those that are in your favourites list (username and password required).

To use this widget

1. Include the style sheet in your html document

```html
<link rel="stylesheet" type="text/css" href="<bamboo-base-url>/styles/bamboo-widget.css" />
```

2. Place the following script tag in your html

   • For all plans

```html
<script type="text/javascript"
src="<bamboo-base-url>/js/jsBuildSummaryAll.action" >
</script>
```

   • For your favourite plans

```html
<script type="text/javascript"
src="<bamboo_base_url>/js/jsBuildSummaryFavourites.action?os_username=<your-user-name>&os_password=<your_password>"
</script>
```

3. Replace `<bamboo-base-url>` with the base url for your bamboo instance.
4. Replace `<your-user-name>` and `<your-password>` with the appropriate values.
5. Style! - The style sheet provided just gives some basic style definitions. You can override these definitions to customise the widgets to suit your needs.

Example

Live example from [http://opensource.bamboo.atlassian.com/](http://opensource.bamboo.atlassian.com/)

```html
<link rel="stylesheet" type="text/css" href="http://opensource.bamboo.atlassian.com/styles/bamboo-widget.css" />

<script type="text/javascript" src="http://opensource.bamboo.atlassian.com/js/jsBuildSummaryAll.action"></script>
```
Latest Builds

This page last changed on May 14, 2007 by rosie@atlassian.com.

This widget produces a list of the last 15 completed builds. A summary is provided for each outlining the
build number, reason for the build, date, duration and test results.

To use this widget

1. Include the style sheet in your html document

```html
<link rel="stylesheet" type="text/css" href="<bamboo-base-url>/styles/bamboo-widget.css"/>
```

2. Place the following script tag in your html

```html
<script type="text/javascript" src="<bamboo-base-url>/js/showRecentlyCompleted.action"></script>
```

3. Replace `<bamboo-base-url>` with the base url for your bamboo instance.

4. Style! - The style sheet provided just gives some basic style definitions. You can override these
definitions to customise the widgets to suit your needs.

Example

Live example from http://opensource.bamboo.atlassian.com/

```html
<link rel="stylesheet" type="text/css" href="http://opensource.bamboo.atlassian.com/styles/bamboo-widget.css">
<script type="text/javascript" src="http://opensource.bamboo.atlassian.com/js/showRecentlyCompleted.action"></script>
```

RELATED TOPICS

- Javascript Widgets
  - All Plans & My Favourite Plans
  - Latest Builds
  - Latest Status of a Plan
  - My Latest Changes
  - Plan Summary Graphs

Bamboo Documentation Home
Latest Status of a Plan

This page last changed on Dec 13, 2007 by alui.

This widget allows you to view the current status of a particular plan.

To use this widget

1. Include the style sheet in your html document

   ```html
   <link rel="stylesheet" type="text/css" href="<bamboo-base-url>/styles/bamboo-status.css" />
   ```

2. Place the following script tag in your html

   ```html
   <script type="text/javascript"
   src="<bamboo-base-url>/js/showLatestBuildStatus.action?buildKey=<plan-key>">
   </script>
   ```

3. Replace `<bamboo-base-url>` with the base url for your bamboo instance.

4. Replace `<plan-key>` with the key of the plan you want to summarise.
   eg. TEST-DEF

5. Style - The style sheet provided just gives some basic style definitions. You can override these definitions to customise the widgets to suit your needs.

Example

Live example from [http://opensource.bamboo.atlassian.com/](http://opensource.bamboo.atlassian.com/)

```html
<link rel="stylesheet" type="text/css" href="http://opensource.bamboo.atlassian.com/styles/bamboo-status.css" />

<script type="text/javascript" src="http://opensource.bamboo.atlassian.com/js/showLatestBuildStatus.action?buildKey=STRUTS-MAIN"></script>
```
My Latest Changes

This page last changed on Dec 13, 2007 by alui.

This widget allows you to view a list of your 10 most recent changes. It provides details of the changes you made (including the commit comments and links to related JIRA issues) as well as details of the build the change was included in (success or failure, how long ago and test results).

To use this widget

1. Include the style sheet in your html document

```html
<link rel="stylesheet" type="text/css" href="<bamboo-base-url>/styles/bamboo-widget.css" />
```

2. Place the following script tag in your html

```html
<script type="text/javascript" src="<bamboo-base-url>/js/myChanges.action?os_username=<your-user-name>&os_password=<your-password>"></script>
```

3. Replace `<bamboo-base-url>` with the base url for your bamboo instance.

4. Replace `<your-use-name>` and `<your-password>` with the appropriate values.

5. Style!! - The style sheet provided just gives some basic style definitions. You can override these definitions to customise the widgets to suit your needs.

Example

```html
<link rel="stylesheet" type="text/css" href="http://localhost:8085/styles/bamboo-widget.css" />
<script type="text/javascript" src="http://localhost:8085/js/myChanges.action?os_username=admin&os_password=admin"></script>
```

<table>
<thead>
<tr>
<th>Build</th>
<th>When</th>
<th>Comments</th>
<th>Test Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUCCESS-DEF-88</td>
<td>6 minutes ago</td>
<td>20th time lucky!!</td>
<td>Passed 2 passed</td>
</tr>
<tr>
<td>SUCCESS-DEF-83</td>
<td>14 minutes ago</td>
<td>Trying to stop build from failing (TST-110)</td>
<td>Passed 4 out of 6 failed!</td>
</tr>
<tr>
<td>SUCCESS-DEF-82</td>
<td>17 minutes ago</td>
<td>Altered test files (TST-110)</td>
<td>Passed 6 out of 6 failed!</td>
</tr>
<tr>
<td>SUCCESS-DEF-81</td>
<td>32 minutes ago</td>
<td>Added failing test to original test suite</td>
<td>Passed 4 out of 6 failed!</td>
</tr>
<tr>
<td>SUCCESS-DEF-75</td>
<td>1 hour ago</td>
<td>*** empty log message ***</td>
<td>Passed 4 out of 6 failed!</td>
</tr>
<tr>
<td>SUCCESS-DEF-74</td>
<td>3 days ago</td>
<td>Brynlie's change is related to TST-1a</td>
<td>Passed 6 out of 6 failed!</td>
</tr>
<tr>
<td>SUCCESS-DEF-73</td>
<td>3 days ago</td>
<td>Trying to break build</td>
<td>Passed 4 out of 6 failed!</td>
</tr>
<tr>
<td>AMP-DEF-10</td>
<td>1 week ago</td>
<td>*** empty log message ***</td>
<td>Passed 3 passed</td>
</tr>
<tr>
<td>TEST-DEF-60</td>
<td>1 week ago</td>
<td>*** empty log message ***</td>
<td>Passed 2 out of 4 failed!</td>
</tr>
<tr>
<td>SUCCESS-DEF-14</td>
<td>1 week ago</td>
<td>*** empty log message ***</td>
<td>Passed 2 passed</td>
</tr>
</tbody>
</table>
Plan Summary Graphs

These widgets allow you to view either of the two summary graphs displayed on the plan summary page. The two graphs are:

- Build Duration & Number of Failures per Build
- Successful Builds & Average Duration Per Time Period

To use this widget

1. Place the following script tag in your html

   - For Build Duration & Number of Failures per Build

     ```html
     <script type="text/javascript"
     src="<bamboo-base-url>/js/viewCombinedByBuildNumberChart.action?buildKey=<Plan-Key>&filterController.selectedFilterKey=<filter-key>">
     </script>
     ```

   - For Successful Builds & Average Duration Per Time Period

     ```html
     <script type="text/javascript"
     src="<bamboo-base-url>/js/jsViewCombinedByTimePeriodChart.action?buildKey=<Plan-Key>&filterController.selectedFilterKey=<filter-key>">
     </script>
     ```

2. Replace `<plan-key>` with the key of the plan you want to summarise.
   eg. TEST-DEF

3. Replace `<filter-key>` with one of the following options:
   * LAST_25_BUILDS
   * LAST_7_DAYS
   * LAST_30_DAYS
   * LAST_90_DAYS
   * ALL_BUILDS

Example

Live example from http://opensource.bamboo.atlassian.com/
The Bamboo Development Hub is for people who are looking to develop their own plugins. For pre-built plugins please visit the [library](#).

- **Bamboo API**
- **Bamboo Developer FAQ**
- **Bamboo Plugin Developer's Guide**
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- **Bamboo Plugin Tutorial**
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- **Changes for Bamboo 2.0**
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- **Setting up Bamboo Development Environment in IDEA**

See also: [6.3 Installing a new Plugin](#)
Bamboo API

This page last changed on May 14, 2007 by rosie@atlassian.com.

The API documentation is installed on your Bamboo server and can be found at: http://your-bamboo-host/api/index.action (replace 'your-bamboo-host' with your Bamboo server's name).

If you have installed Bamboo locally, you can view the API documentation at: http://localhost:8085/api/index.action.

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer's Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Bamboo Plugin Developer's Guide

Bamboo Plugins Overview

A Bamboo plugin is a single JAR containing code, a plugin descriptor (XML) and usually some Freemaker template files to render HTML.

The plugin descriptor is the only mandatory part of the plugin. It must be called atlassian-plugin.xml and be located in the root of your JAR file.

Each plugin consists of one or more plugin modules. These are of different types (for example a report, or a post-build action) and each has an individual XML element describing it. Each module is described below together with the XML element required for it.

Here is a sample of the descriptor with highlighted elements:

```xml
<!-- the plugin key must be unique, think of it as the 'package' of the plugin -->
<atlassian-plugin key= "com.atlassian.plugin.sample"  name= "Sample Plugin">
  <!-- a short block describing the plugin itself -->
  <plugin-info>
    <description>This is a brief textual description of the plugin</description>
    <!-- the version of the plugin -->
    <version>1.1</version>
    <!-- the versions of the application this plugin is for -->
    <application-version min="3.0" max="3.0"/>
    <!-- details of the plugin vendor -->
    <vendor name= "Atlassian Software Systems Pty Ltd"  url= "http://www.atlassian.com"/>
  </plugin-info>
  . . . 1 or more plugin modules . . .
</atlassian-plugin>
```

Each plugin has a plugin key which is unique among all plugins (eg "com.atlassian.plugin.sample"). Semantically this equates to the package of a Java class. Each module within the plugin also has a module key which is unique within the plugin (eg "myreport"). Semantically this equates to the class name of a Java class.

The plugin key + module key are combined to make the complete key of the plugin module (combining the examples above, the complete key would be "com.atlassian.plugin.sample:myreport"). Note: a : is used to separate the plugin key from the module key.

Setting up a Bamboo Plugin Project

Refer to the Getting Started guide.

Deploy a Bamboo Plugin

Installing plugins in Bamboo is easy.

Once you have downloaded or created your plugin jar, follow these steps:

1. Shut down Bamboo
2. copy '$MY_COOL_PLUGIN.jar' into '.../webapp/WEB-INF/lib/'
3. Start up Bamboo. Your plugin should be automatically installed and activated.
4. Enjoy!

Bamboo Plugin Module Types

The following types of plugin modules are supported by Bamboo

<table>
<thead>
<tr>
<th>Module Type</th>
<th>Since version</th>
<th>Documentation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>builder</td>
<td>1.0</td>
<td>Builder Plugin Module</td>
<td>Add new builders to Bamboo</td>
</tr>
</tbody>
</table>

Document generated by Confluence on Apr 14, 2008 01:44
xwork 1.0  XWork Plugin Module  XWork actions and views bundled with the plugin. This enables building generic user interfaces. 

report 1.0  Report Module  Defines a report of build telemetry data. 

preBuildAction 1.1  Pre Build Action Module  Prepends a custom synchronous process to the build. Occurs before the builder has run. 

buildProcessor 1.0  Build Processor Module  Append a custom synchronous process to the build. Occurs after the builder has run. 

buildCompleteAction 1.0  Build Complete Action Module  Add a custom asynchronous action after the build process has completed. 

postBuildIndexWriter 1.0  Post Build Index Writer Module  Writes custom build data into the index for report generation. 

indexReader 1.0  Index Reader Module  Reads out custom index information written by the postBuildIndexWriter from the index. 

web-item 1.0  Web Item Module  Add new links to the Bamboo interface 

web-section 1.0  Web Section Module  Add a new section to the Bamboo interface 

web-resource 1.0  [Web Resource Module]  Add a new resource to the Bamboo application (e.g. javascript) 

notificationCondition 1.1  Notification Condition Module  Add new notification condition 

repository 1.1  Source Repository Module  Add a custom source repository 

Built-in Bamboo system plugins

A number of functions and areas within Bamboo are shipped as built in plugins. These can also be useful for plugin developers who want to know more about how to create their own plugins, as they showcase the functionality that can be built.

The system plugins are referenced from the following files (located in /WEB-INF/classes:

- system-builder-plugin.xml - the built in builders, including Ant, Maven, and Maven 2.
- system-clover-plugin.xml - the built in Clover analytics.
- system-jira-plugin.xml - the built tab view of JIRA issues in a build.
- system-labelling-plugin.xml - the built in automatic build labeller.
- system-reports-plugin.xml - the built reports of builds grouped by time periods under the Reports tab.
- system-repository-plugin.xml - definition for the CVS, SVN and Perforce repositories shipped with Bamboo.
- system-webUI-plugin.xml - the built in menu items under the Administration tab as well as the tab menu items on the View Plan page and the View Build Results page.
- system-notifications-plugin.xml - the build in notification conditions, including all builds, failed builds, after X failed builds.

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer’s Guide
  ° Build Complete Action Module
  ° Builder Plugin Module
- Build Processor Module
- Getting Started
- Index Reader Module
- Notification Condition Module
- Post Build Index Writer Module
- Pre Build Action Module
- Report Module
- Repository Extensions
- Source Repository Module
- Web Item Module
- Web Section Module
- XWork Plugin Module

- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller

- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins

- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Build Complete Action Module

Description

Like the Build Processor Module, this allows you to specify a custom action to take place. However, the difference is that this will run after the full build result has been registered. The build is deemed to have completed before the BuildCompleteAction is fired. Hence, build complete actions cannot impact the state of the build.

Interface

Build Complete Action modules must implement the com.atlassian.bamboo.build.CustomBuildCompleteAction interface.

Sample Module Descriptor Element

```xml
<buildCompleteAction key= "autoLabeller" name= "Build Automatic Labeller"
   class= "com.atlassian.bamboo.labels.AutoLabelBuildCompleteAction">
   <resource type= "freemarker" name= "edit" location= "templates/plugins/buildCompleteAction/autoLabellingEdit.ftl"/>
   <resource type= "freemarker" name= "view" location= "templates/plugins/buildCompleteAction/autoLabellingView.ftl"/>
   <description> An automatic labelling plugin. </description>
</buildCompleteAction>
```

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer's Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Builder Plugin Module

Description

A plugin module which defines a builder in Bamboo, such as Maven, Maven2, or Ant.

Interface

Builder modules must implement the com.atlassian.bamboo.builder.Builder interface.

Sample Module Descriptor Element

```xml
<builder key="mvn2" name="Maven 2.x Builder"
         class="com.atlassian.bamboo.builder.Maven2Builder">
  <description>A Maven 2.x Builder</description>
  <resource type="freemarker" name="edit" location="templates/plugins/builder/
           mavenBuilderEdit.ftl"/>
  <resource type="freemarker" name="view" location="templates/plugins/builder/
           mavenBuilderView.ftl"/>
</builder>
```

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer's Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Build Processor Module

This page last changed on Jun 19, 2007 by ahempel.

Description

The BuildProcessor module allows you to define a custom process that runs during the build. This will occur immediately after the builder has completed execution and the test results have been captured. The BuildProcessor forms part of build execution run and the result of the execution is only registered after your custom BuildProcessor has completed. This means that your plugin has the capability to affect the final BuildState of your build (i.e. success/failure).

Interface

Build Processor modules must implement the com.atlassian.bamboo.build.CustomBuildProcessor interface.

Sample Module Descriptor Element

```xml
<buildProcessor key="cloverResultCollector" name="Clover Results Collector"
class="com.atlassian.bamboo.builder.coverage.CloverBuildProcessor">
   <skipIfFailed>true</skipIfFailed>
   <resource type="freemarker" name="edit" location="templates/plugins/buildProcessor/cloverProcessorEdit.ftl"/>
   <resource type="freemarker" name="view" location="templates/plugins/buildProcessor/cloverProcessorView.ftl"/>
   <description>A Clover report parser and data collector</description>
</buildProcessor>
```

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer's Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Getting Started

This page last changed on Aug 21, 2007 by asridhar.

Sun JAVA Libraries

Due to licensing restrictions, we are not allowed to re-distribute native Sun JAVA libraries through our maven2 public repositories. Please visit our Confluence page on coping with Sun JAVA Libraries for further details.

Requirements

- You will need to have Maven 2 installed. You can download Maven 2 [here](#).
- A copy of Bamboo 1.0, either built from source or a binary distribution, so that you can test your plugin when you develop.
- We strongly recommend that you build with the Bamboo Plugin Development Kit, available from our public SVN Repository.

```
```

- These instructions assume your IDE is IDEA. You will need to ensure your dependencies are set up correctly if you use any other IDE.

Setting up the project

Inside the Bamboo Development kit, you will need to change the `pom.xml` file to correctly setup your project. Within this file, you will need to change the following xml elements:

- `<groupId>` - this is the group identifier for your plugin. It is typically something similar to a Java package name.
- `<artifactId>` - this defines the file name of your plugin JAR file.
- `<version>` - this defines the version of your plugin.
- `<name>` - this defines the name of your plugin.
- `<scm>` - this defines your source repository URL

Once this is done, you can run the command `mvn idea:idea` which will download your dependencies (including the bamboo libraries) and build an IDEA project file `$MY_PLUGIN_NAME.ipr`. To begin development, simply launch the IDEA project file created.

Once IDEA is up, you will also need to modify the file `/src/main/resources/atlassian-plugin.xml` to give your plugin a name and a plugin key. You should also fill in your plugin meta-data.

That's it, you should now be ready to start coding your Bamboo plugin.

RELATED TOPICS

- [Bamboo API](#)
- [Bamboo Developer FAQ](#)
- **Bamboo Plugin Developer's Guide**
  - [Build Complete Action Module](#)
  - [Builder Plugin Module](#)
  - [Build Processor Module](#)
  - [Getting Started](#)
  - [Index Reader Module](#)
  - [Notification Condition Module](#)
  - [Post Build Index Writer Module](#)
  - [Pre Build Action Module](#)
  - [Report Module](#)
  - [Repository Extensions](#)
  - [Source Repository Module](#)
  - [Web Item Module](#)
  - [Web Section Module](#)
  - [XWork Plugin Module](#)
- **Bamboo Plugin Tutorial**
  - [Tutorial 1 - Getting Started with a Simple Post Build Labeller](#)
Tutorial 2 - Configurable Regex Labeller

- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins

- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Index Reader Module

This page last changed on Jun 19, 2007 by ahempel.

Description

Written in conjunction with Post Build Index Writer Module, the IndexReader will translate the fields in the index and re-insert the information into a BuildResultSummary object, which has a specially designated customBuildData map for this purpose.

Interface

Index Reader modules must implement the com.atlassian.bamboo.index.CustomIndexReader interface.

Sample Module Descriptor Element

```
<indexReader key=“cloverIndexReader” name=“Reads Clover result values from index”
class=“com.atlassian.bamboo.builder.coverage.CloverIndexReader”>
  <description>Reads the clover result from an index document and populates into build result summary</description>
</indexReader>
```

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer’s Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Notification Condition Module

Description

A plugin module which allows you to define your own notification condition

You can use this plugin to

- Set conditions as to whether or not to send a notification
- Generate customised notification content

Interface

Notification Condition modules must implement the

Sample Module Descriptor Element

```xml
<negation-condition key="buildCompleted.XFailedBuilds" name="After X Failed Builds Completed"
class="com.atlassian.bamboo.notification.conditions.AfterXFailedBuildsCondition">
  <description>Send Notification After X Failed Builds</description>
  <resource type="freemarker" name="edit" location="templates/plugins/notifications/
afterXFailedEdit.ftl"/>
  <resource type="freemarker" name="view" location="templates/plugins/notifications/
afterXFailedView.ftl"/>
</negation-condition>
```

Other Information regarding the NotificationCondition class

Several of the interface methods for this class accept an Event object as a parameter. Current functionality only allows this to be a BuildCompletedEvent. This event will contain the Build object, BuildResults object and BuildResultsSummary object for you to use.

RELATED TOPICS

- **Bamboo API**
- **Bamboo Developer FAQ**
- **Bamboo Plugin Developer's Guide**
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module

- **Bamboo Plugin Tutorial**
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller

- **Changes for Bamboo 2.0**
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins

- **Setting up Bamboo Development Environment in IDEA**
Post Build Index Writer Module

This page last changed on Jun 19, 2007 by ahempe1.

Description

The PostBuildIndexWriter allows you to write your custom data for a build into the index, which allows for future retrieval in your custom Report Module. The PostBuildIndexWriter will be invoked in three places in Bamboo: when a build completes and it indexes, operations which requires a re-index of a particular build (result), and when you run the re-index all action under the Administration tab.

The PostBuildIndexWriter should always be written in conjunction with a Index Reader Module which will be able to retrieve the data in the index.

Interface

Post Build Index Writer modules must implement the com.atlassian.bamboo.index.CustomPostBuildIndexWriter interface.

Sample Module Descriptor Element

```
<postBuildIndexWriter key="cloverIndexWriter" name="Write Clover Result to Index"
class="com.atlassian.bamboo.builder.coverage.CloverPostBuildIndexWriter">
  <description>Writes the clover result in a build results to an index document</description>
</postBuildIndexWriter>
```

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer's Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Pre Build Action Module

This page last changed on Jun 19, 2007 by ahempel.

Description

The PreBuildAction module allows you to define a custom process that runs before your build begins. This will occur immediately before the builder begins execution. The PreBuildAction will have access to the BuildResults object which contains the information for the build.

Interface

Pre Build Action modules must implement the com.atlassian.bamboo.build.CustomPreBuildAction interface.

Sample Module Descriptor Element

```xml
<preBuildAction key= "vcsVersion" name= "VCS Version Collector"
class= "com.atlassian.bamboo.vcsversion.VCSVersionReader">
  <description> A custom action that reads the identifier of a source repository version and stores it into the custom data map of a build. </description>
  <resource type= "freemarker" name= "edit" location= "templates/plugins/preBuildAction/vcsVersionReaderEdit.ftl" />
  <resource type= "freemarker" name= "view" location= "templates/plugins/preBuildAction/vcsVersionReaderView.ftl" />
</preBuildAction>
```

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer's Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Report Module

This page last changed on May 14, 2007 by rosie@atlassian.com.

Description

This defines a report module. A report module will appear under the Reports tab.

A report typically consists of two objects:

- A ReportCollector object implementing the `com.atlassian.bamboo.reports.collector.ReportCollector` interface. This takes in a list of builds and generates a DataSet.
- A ReportLineChart object extending the `com.atlassian.bamboo.reports.charts.BambooReportLineChart` class. This chart will be responsible for rendering the dataset results generated by the ReportCollector. Charts in Bamboo are generated via jFreeChart.

Sample Module Descriptor Element

```
<report key="ratioOfSuccess" name="Percentage of Successful Builds" class="com.atlassian.bamboo.reports.collector.RatioOfSuccessCollector">
  <description>Comparing success percentages gives you an idea of how stable a build is compared to one another. 100% means your build is always rock solid. 0% means something is seriously wrong.</description>
  <chartClass>com.atlassian.bamboo.reports.charts.BuildSummarySuccessRatioLineChart</chartClass>
</report>
```

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer’s Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeler
  - Tutorial 2 - Configurable Regex Labeler
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Repository Extensions

You can extend the standard repository functionality by implementing any of the optional interfaces described below.

On this page:

Error formatting macro: toc: java.lang.NullPointerException

**com.atlassian.bamboo.v2.build.repository.RepositoryEventAware**

Description

The *com.atlassian.bamboo.v2.build.repository.RepositoryEventAware* interface allows you to instruct the repository to perform a custom action before and/or after the checkout/update occurs.

Methods

```java
void preRetrieveSourceCode(@NotNull BuildContext buildContext);
```

This will run before the `retrieveSourceCode` method is called. It allows you to do custom actions to prepare for the checkout, such as some customised cleanup of the source directory.

```java
void postRetrieveSourceCode(@NotNull BuildContext buildContext);
```

This will run after the `retrieveSourceCode` method is called. The extension point allows you to easily set custom data, do any post checkout collection of data.
Source Repository Module

This page last changed on May 14, 2007 by rosie@atlassian.com.

Description

A plugin module which defines a repository in Bamboo, such as CVS, Subversion, or Perforce.

Sample Module Descriptor Element

```
<repository key="svn" name="SVN Repository"
  class="com.atlassian.bamboo.repository.svn.SvnRepository">
  <description>A Subversion Repository</description>
  <resource type="freemarker" name="edit" location="templates/plugins/repository/svnRepositoryEdit.ftl"/>
  <resource type="freemarker" name="view" location="templates/plugins/repository/svnRepositoryView.ftl"/>
</repository>
```

Notes

Plugins of this type must implement the `com.atlassian.bamboo.repository.Repository` interface. For (comparative) simplicity, you should use the class `AbstractRepository` as a starting point and extend from that.

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer's Guide
  * Build Complete Action Module
  * Builder Plugin Module
  * Build Processor Module
  * Getting Started
  * Index Reader Module
  * Notification Condition Module
  * Post Build Index Writer Module
  * Pre Build Action Module
  * Report Module
  * Repository Extensions
  * Source Repository Module
  * Web Item Module
  * Web Section Module
  * XWork Plugin Module
- Bamboo Plugin Tutorial
  * Tutorial 1 - Getting Started with a Simple Post Build Labeller
  * Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  * Build Process for 2.0
  * Post-Build Processing in Bamboo 2.0
  * Repository plugin changes in 2.0
  * Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Web Item Module

This page last changed on May 14, 2007 by rosie@atlassian.com.

Description

The WebItem allows you to define a link in the Bamboo system. (Usually in some form of menu).

Currently, you can use the web-item to add links to three locations:

- The Administration Menu
- The Plan Sub Menu (tabs on the View Plan page)
- The Results Sub Menu (tabs on the View Build Results page)

Sample Module Descriptor Element

```xml
<web-item key= "pipelineConfig"  name= "Build Queues"  section= "system.admin/builds"  weight= "20" >
  <label key= "webitems.system.admin.build.queues" />
  <link> /admin/configurePipeline!default.action </link>
  <condition class= "com.atlassian.bamboo.plugins.web.conditions.AdminPermissionCondition" />
</web-item>
```

Module Components

- **key** - this is the unique identifier of the web-item, it is also used by Bamboo to give the link an id.
- **name** - in the plan sub menu and results sub menu this is used to determine if the current link (tab) is active
- **section** - the section is made of of the parent section's location followed buy the name of the parent section. In Bamboo this is used to retrieve the appropriate web-items for the menu. (see [Web Section Module](#))
- **weight** - this is used to determine the order of the items on the page
- **label** - this will be displayed on the screen and can be plain text or a property key
- **link** - the link is the url the link will point to. It can be absolute or relative to Bamboo's context path
- **condition** - by implementing the com.atlassian.plugin.web.Condition class you can add rules to determine whether the link will be displayed or not.

Both the link and the id can make use of parameters passed to the page. For example:

```xml
<link>/build/viewBuildFiles.action?buildKey=${buildKey}</link>
```

where `${buildKey}` is the parameter name.

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer's Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
• Post-Build Processing in Bamboo 2.0
• Repository plugin changes in 2.0
• Updates to the build processing plugins
• Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
**Web Section Module**

This page last changed on May 14, 2007 by rosie@atlassian.com.

**Description**

The WebSection module is used to provide a section or grouping of [Web Item Module](#).

Currently, Web Sections are used to group Web Items for the Administration Menu, Plan Sub Menu, Results Sub Menu

**Sample Module Descriptor Element**

```
<web-section key= "builds"  name= "Builds"  location= "system.admin"  weight= "100">
  <label key= "websections.system.admin.build" />
  <icon height= "16"  width= "16">
    <link> /images/icons/icon_spanner.gif </link>
  </icon>
</web-section>
```

**Notes**

The section is only displayed on the Administration Menu but it is required for all locations as Bamboo uses it to place the web items.

Which menu the item gets placed in is determined by the `location` attribute.

The following table indicates the location attribute required for each menu:

<table>
<thead>
<tr>
<th>Menu</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administration Menu</td>
<td>system.admin</td>
</tr>
<tr>
<td>Plan Sub Menu</td>
<td>build.subMenu</td>
</tr>
<tr>
<td>Build Results Sub menu</td>
<td>results.subMenu</td>
</tr>
</tbody>
</table>

**RELATED TOPICS**

- [Bamboo API](#)
- [Bamboo Developer FAQ](#)
- [Bamboo Plugin Developer's Guide](#)
- [Build Complete Action Module](#)
- [Builder Plugin Module](#)
- [Build Processor Module](#)
- [Getting Started](#)
- [Index Reader Module](#)
- [Notification Condition Module](#)
- [Post Build Index Writer Module](#)
- [Pre Build Action Module](#)
- [Report Module](#)
- [Repository Extensions](#)
- [Source Repository Module](#)
- [Web Item Module](#)
- [Web Section Module](#)
- [XWork Plugin Module](#)

- [Bamboo Plugin Tutorial](#)
  - [Tutorial 1 - Getting Started with a Simple Post Build Labeller](#)
  - [Tutorial 2 - Configurable Regex Labeller](#)

- [Changes for Bamboo 2.0](#)
  - [Build Process for 2.0](#)
  - [Post-Build Processing in Bamboo 2.0](#)
  - [Repository plugin changes in 2.0](#)
  - [Updates to the build processing plugins](#)

- [Setting up Bamboo Development Environment in IDEA](#)

[Bamboo Documentation Home](#)
XWork Plugin Module

This page last changed on May 14, 2007 by rosie@atlassian.com.

Description

Each XWork module is deployed as a plugin module of type xwork and contains one or more XWork package elements.

Here is an example `atlassian-plugin.xml` file containing a single XWork module:

The xwork plugin module allows you to define your own xwork package and actions that you can access.

To build the action into the system, you will typically need to add a Web Item Module to link to your action.

Sample Module Descriptor Element

```xml
<xwork key="viewCloverResult"  name= "View Clover Result">
  <package name= "cloverPlugin"  extends= "buildView">
    <action name= "viewCloverResult"  class= "com.atlassian.bamboo.build.ViewBuildResults">
      <result name= "success"  type= "freemarker">/plugins/clover-plugin/viewCloverResult.ftl</result>
      <result name= "error"  type= "freemarker">/error.ftl</result>
    </action>
  </package>
</xwork>
```

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer's Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
- Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Introduction

The purpose of this tutorial is to demonstrate how you can add custom functionality to Bamboo via plugins. The tutorial aims to give you a good starting point for Bamboo plugin development, and how the different Bamboo plugin modules can work together. In this tutorial, we will run through the development of a plugin derived from a real use-case requirement.

Bamboo Labeller Plugin

There are many cases when builds in Bamboo fail because of particular, recurring errors. For example, a functional test in Confluence may periodically fail because of an OutOfMemoryError when things get hectic. It would be useful for developers to keep track of these particular builds, so they can look into it further. For this to happen, Bamboo will need to:

- Parse the error logs after a failed build.
- Look for the text `java.lang.OutOfMemoryError` in the log
- If found, tag the build with a label, say `out_of_memory`

Once that's done, the developer can set up a RSS feed on the `out_of_memory` tag. They will then be able to keep track of the builds which fail with an OutOfMemoryError. We can even extend this concept further, by replacing the search for `java.lang.OutOfMemoryError` with any regular expression, tagging it with a label of choice.

Contents

- Tutorial 1 - Getting Started with a Simple Post Build Labeller
- Tutorial 2 - Configurable Regex Labeller

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer’s Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
### Step 1 - Setting up the project.

The first thing you need to do is to set up your Bamboo Plugin project and source directories. The instructions for how to do this are available [here](#).

In the `atlassian-plugin.xml` located under `/src/main/resources/`, you will need to give the plugin a unique key, as well as some meta information about this plugin. As our plugin simply labels, we have called it "labeller". Below is the `atlassian-plugin.xml` for our labelling plugin:

```xml
<atlassian-plugin key= "com.atlassian.bamboo.plugin.labeller"  name= "Build Labeller">
  <plugin-info>
    <description> Bamboo Labeller </description>
    <version> 1.0 </version>
    <application-version min= "1.0" max= "1.0"/>
    <vendor name= "Atlassian Software Systems Pty Ltd" url= "http://www.atlassian.com"/>
  </plugin-info>
</atlassian-plugin>
```

Now we are ready to move onto writing some code to make our plugin do something.

### Step 2 - Adding the first Build Complete Labeller Module

In this plugin, we want Bamboo to perform a custom action immediately after a build has completed. To do this, we write a Build Complete Action Module. You can see all the available Bamboo module types [here](#).

To start things off, we would like to keep our custom action pretty simple and make sure things work. Our first cut of the `BuildLabeller` will simply label the build as "out_of_memory" if the "OutOfMemoryError" was found in the logs.

```java
public class BuildLabeller implements CustomBuildCompleteAction {
    private static final Logger log = Logger.getLogger(BuildLabeller.class);

    /**
     * Dependency on labelManager. Bamboo's Spring IOC will automatically inject manager
     * into this class via the setter.
     */
    private LabelManager labelManager;

    /**
     * This action will run after a build has completed.
     *
     * The build will be labelled with "out_of_memory" if the "OutOfMemoryError" was detected in the
     * logs.
     * @param build
     * @param buildResults
     */
    public void run(Build build, BuildResults buildResults) {
        List logs = buildResults.getBuildLog();
        for (Iterator iterator = logs.iterator(); iterator.hasNext();)
            { SimpleLogEntry log = (SimpleLogEntry) iterator.next();
```
Our custom module must implement the CustomBuildCompleteAction interface, which defines a run method and a validate method.

The run method is what gets called when a build completes. Our run method in this plugin is fairly simple. It loops through each line of the build logs and searches for the exact string - "OutOfMemoryError". Once found, it stops looping and labels the build.

In the run method, we make use of the services of the LabelManager (a dependency), which is responsible for tagging of a build. Dependencies in plugins are automatically handled by Bamboo Spring container. As long as the plugin has the correct "setter" method, the dependency will be automatically injected.

You may notice that the other method defined by the CustomBuildCompleteAction interface: validate currently doesn't do anything. We will return to this in the next tutorial.

Step 3 - Registering the Build Complete Labeller Module

Once you have written your labeller module, we must now register the plugin module into our plugin descriptor (atlassian-plugin.xml).

```xml
<buildCompleteAction key="labeller" name="Build Labeller"
  class="com.atlassian.bamboo.plugins.labeller.BuildLabeller">
  <description>An automatic labelling plugin.</description>
</buildCompleteAction>
```
Step 4 - Build and Test

That's it. We now need to test our code. To do this, we can build our plugin by returning to the command line in the root directory of your source directory, and run the command: `mvn package`. This created a `bamboo-labeller-plugin-1.1.0.jar`. We can now drop this into Bamboo (/webapp/WEB-INF/lib), and see it in action.

Here is what our plugin produced after we ran a build with a `OutOfMemoryError`:

Next Steps

So we have made our first basic plugin. Right now, it's not very configurable, and runs for every build. In the next tutorial, we will introduce configurability to our Labeller.

Next >

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer's Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Tutorial 2 - Configurable Regex Labeller

In the previous tutorial, we have made our plugin label the build whenever the logs had the words "OutOfMemoryError". This, however, is not very useful for the other builds which don't have this memory problem. Also, it is not very useful to only be able to tag with "out_of_memory". In this tutorial, we will extend on the plugin module so that we can configure when to label, and what to label a build with.

The source code to the plugin used in this tutorial is available on the Atlassian public source repository. You can check out the source code [here](#).

Step 1 - Adding configuration views

To do this, we must first add the views for configuring the labeller. The **BuildCompleteAction** module type comes with the capability to accept Freemarker templates which allows you to edit and view custom configuration in the Build Plan Configuration page, under the Post Action tab.

**Edit Configuration View**

The Freemarker template to edit our Labeller configuration is below (`regexLabellerEdit.ftl`):

```freemarker
[@ui.bambooSection title='Pattern matching labelling.' ]
[@ww.textfield name='custom.bamboo.labeller.regex' label='Regex Pattern'
    description='The regular expression for which to match the log files on.' ]
[@ww.textfield name='custom.bamboo.labeller.label' label='Label(s)'
    description='The label(s) for the build if it matches the specified regex pattern.' ]
[/@ui.bambooSection]
```

Here, we define a section with a title 'Pattern matching labelling.' Inside our configuration section are two text fields, one for the regex expression for matching against the logs, and one for the label(s) that we want to tag a build with if the regex expression matches.

We have named our two text fields `custom.bamboo.labeller.regex` and `custom.bamboo.labeller.label`. These are the keys to your custom configuration property stored in Bamboo.

Please note that these keys must start with "custom." for Bamboo to recognize and store within the plan's configuration. You may also notice that the keys are "namespaced". This is a good idea to prevent a clash of custom configuration properties.

**Display Configuration View**

We also define a Freemarker view for viewing the configuration (read-only). The display configuration view is below (`regexLabellerView.ftl`):

```freemarker
[#if build.buildDefinition.customConfiguration.get('custom.bamboo.labeller.regex')?has_content ]
    [@ui.bambooInfoDisplay titleKey='Pattern Matching Labelling' float=false height='80px']
    [@ww.label label='Regex Pattern' ]
    [@ww.param name='value']${build.buildDefinition.customConfiguration.get('custom.bamboo.labeller.regex')?if_exists}
    [@/ww.param]
    [@ww.label label='Labels' ]
    [@ww.param name='value']${build.buildDefinition.customConfiguration.get('custom.bamboo.labeller.label')?if_exists}
    [@/ww.param]
[/@ui.bambooInfoDisplay]
[/#if]
```

Here we simply build display the configuration by retrieving your custom properties via the same keys we used in the edit view.
Registering the views in the Plugin Descriptor

We need to register these two Freemarker templates as part of our BuildCompleteAction module. We do this by adding `<resource>` tags with the file path of the templates within the module descriptor definition.

```xml
<buildCompleteAction key="labeller" name="Build Labeller
   class="com.atlassian.bamboo.plugins.labeller.BuildLabeller">
   <description>An automatic labelling plugin.</description>
   <resource type="freemarker" name="edit" location="templates/buildCompleteAction/
regexLabellerEdit.ftl"/>
   <resource type="freemarker" name="view" location="templates/buildCompleteAction/
regexLabellerView.ftl"/>
</buildCompleteAction>
```

Once that's done, we can see the templates in action.

Under the edit configuration page:

And under the view configuration page:

Step 2 - Adding validation

Inserting the templates has allowed us to view and edit custom plan configuration properties. However, we should validate the input we provide for the BuildLabeller, to catch invalid labels or regex patterns.

This is where we use the `validate` method within our BuildLabeller class, which we have previously left to return `null` in the first tutorial. Bamboo will run this validate method before trying to save custom configuration properties.

```java
/**
   * This method is used to validate a build configuration for a build plan
   * This is used if the CustomBuildCompleteAction needs to have configuration stored
   * against the build plan.
   *
   * @param buildConfiguration
   * @return
   */
public ErrorCollection validate(BuildConfiguration buildConfiguration) {
    // Check the label values to see if they have any invalid characters
    ErrorCollection errors = new SimpleErrorCollection();
    String labelInput = buildConfiguration.getString("custom.bamboo.labeller.label");
    List labels = LabelParser.split(labelInput);
    for (Iterator iterator = labels.iterator(); iterator.hasNext();)
    {
        String label = (String) iterator.next();
        boolean validLabel = LabelParser.isValidLabelName(label);
        if (!validLabel)
        {
```
The `BuildConfiguration` object passed to the validation method is the in-memory version of the build plan configuration. You can get your custom property by simply calling `getString` on the object, providing the custom property key that you used in the Freemarker templates.

### Step 3 - Applying the configuration

At this stage, we can edit, validate, and view our custom configuration for this plugin module. We now need to modify our original `run` method within the `BuildLabeller` to read the custom configuration properties.

```java
/**
 * This action will run after a build has completed.
 * The build will be tagged with a specified set of labels if the logs matches the specified regex pattern.
 * @param build
 * @param buildResults
 */
public void run(Build build, BuildResults buildResults)
{
    // grab the custom configuration object
    Map customConfiguration = build.getBuildDefinition().getCustomConfiguration();

    if (customConfiguration != null)
    {
        if (customConfiguration.containsKey("custom.bamboo.labeller.label"))
        {
            List logs = buildResults.getBuildLog();

            String pattern = (String) customConfiguration.get("custom.bamboo.labeller.regex");

            Pattern regexPattern = Pattern.compile(pattern);

            // Go through the logs
            for (Iterator iterator = logs.iterator(); iterator.hasNext();)
            {
                SimpleLogEntry log = (SimpleLogEntry) iterator.next();

                Matcher matcher = regexPattern.matcher(log.getLog());

                // Use a matcher to see if the logs contained the specified regex
                if (matcher.find())
                {
```
String labelsInput = (String)
customConfiguration.get("custom.bamboo.labeller.label");

// Our configuration also allows for multiple labels.
List labels = LabelParser.split(labelsInput);

for (Iterator iterator2 = labels.iterator(); iterator2.hasNext();)
{
    String label = (String) iterator2.next();
    
    getLabelManager().addLabel(label, buildResults, null);
}
break;
}
}
}
}
}

So that's it! We have now completed a Bamboo plugin containing one BuildCompleteAction module which will match the output logs against a regular expression, and tag it with a set of label(s).

RELATED TOPICS

- Bamboo API
- Bamboo Developer FAQ
- Bamboo Plugin Developer's Guide
  - Build Complete Action Module
  - Builder Plugin Module
  - Build Processor Module
  - Getting Started
  - Index Reader Module
  - Notification Condition Module
  - Post Build Index Writer Module
  - Pre Build Action Module
  - Report Module
  - Repository Extensions
  - Source Repository Module
  - Web Item Module
  - Web Section Module
  - XWork Plugin Module
- Bamboo Plugin Tutorial
  - Tutorial 1 - Getting Started with a Simple Post Build Labeller
  - Tutorial 2 - Configurable Regex Labeller
- Changes for Bamboo 2.0
  - Build Process for 2.0
  - Post-Build Processing in Bamboo 2.0
  - Repository plugin changes in 2.0
  - Updates to the build processing plugins
- Setting up Bamboo Development Environment in IDEA

Bamboo Documentation Home
Changes for Bamboo 2.0

This page last changed on Mar 26, 2008 by ahempel.
The build process has been refactored in Bamboo 2.0 into a series of BuildTasks, each responsible for a particular part of the build process. Each build task has the ability to store aBuildContext. ABuildContext represents the definition of the part of the build process that the build task is responsible for.

The steps of a build process are described below:

1. Change Detection (performed by the server) — Changes are detected via polling, manual builds, etc. When a new build is required, the change logs are defined for the build and the build number for the new build is determined. The build context is also created.
2. Build Queued (performed by the server) — The server decides which agents can execute the build and queues the build.
3. Agent Picks Up Build (performed by the agent) — The capability context is set. The build is removed from queue. The agent begins looping through the build tasks.
4. Update Source (performed by the agent) — The agent runs the CheckoutUpdateForBuild task. The Repository#retrieveSourceCode is called. If the Repository is RepositoryEventAware the appropriate methods will be called. The agent checks if the repository has changed since the last build and clears the source directory if it has.
5. Prepare for the Build (performed by the agent) — The agent runs the PrepareBuildTask. This begins streaming the logs back to the server. The agent also runs the CustomPreBuildAction plugin point. From this point on, the build will be saved.
6. Executes the Build (performed by the agent) — Timer begins. The agent runs the Builder#executeBuild. After the Builder has been run, all the CustomBuildProcessors are run. Timer is stopped.
7. Results Processing (performed by the server) — The server runs CustomBuildProcessorServer, checks if the build has passed and saves and indexes the build result. The server also fires off the BuildCompletedEvent and PostBuildCompletedEvent events.
8. On BuildCompletedEvent (performed by the server) — Notifications are sent in this phase.
Post-Build Processing in Bamboo 2.0

Due to Bamboo 2.0’s support for distributed builds, the plug-in point for performing post-build processing has been split into two:

**CustomBuildProcessor:** These plug-in modules are executed on the agent. They have access to the build filesystem, but not to any Bamboo managers.

**CustomBuildProcessorServer:** These plug-in modules are executed on the Bamboo server, once the build result has been transmitted from the agent. They have access to Bamboo managers, but they are not guaranteed to have access to the build filesystem.

If you have a plug-in that needs to read data from the build filesystem, and make use of Bamboo managers to retrieve and/or store data, you must use the following approach:

**On the Bamboo agent:**

1. Have a CustomBuildProcessor read the data you require from the build filesystem on the agent;
2. Store the data you require as key-value pairs in the `Map<String, String>` obtained from calling `getCustomData()` of the `BuildResult`;

**Then, on the Bamboo server:**

1. Have a CustomBuildProcessorServer read your key-value pairs from the `BuildResult` on the Bamboo server;
2. Make use of the the Bamboo managers to retrieve and store data as required.
Repository plugin changes in 2.0

The Repository plugin has changed significantly in Bamboo 2.0. Previously, the getChangesSinceLastBuild method was responsible for detecting the changes and updating the repository to the latest code. This has been broken into two separate methods now, as described below:

1. `collectChangesSinceLastBuild` method

```java
BuildChanges collectChangesSinceLastBuild(@NotNull String planKey, @NotNull String lastVcsRevisionKey) throws RepositoryException;
```

The `collectChangesSinceLastBuild` method returns a `BuildChanges` object that encapsulates the commits and changes between the last build and the current source repository (through the `vcsRevisionKey`).

2. `retrieveSourceCode` method

```java
String retrieveSourceCode(@NotNull String planKey, @Nullable String vcsRevisionKey) throws RepositoryException;
```

The `retrieveSourceCode` updates the code to the `vcsRevisionKey` as returned from the previous method. If the `vcsRevisionKey` is null, then the method should check out to latest. The return value is what the source code was actually updated to.
Updates to the build processing plugins

The various pre and post build plugin points have been updated to reflect the new build agent task structure.

- com.atlassian.bamboo.build.CustomPreBuildAction
- com.atlassian.bamboo.build.CustomBuildProcessor
- com.atlassian.bamboo.build.CustomBuildProcessorServer

Essentially the method:

```java
public void run(Build build, BuildResults buildResult);
```

has been replaced with the BuildTask interface.

```java
/**
 * Interface defines a basic interface for a task in Bamboo. All {@link #call()} methods return
 * objects.
 */
public interface BuildTask extends Callable<BuildContext> {
    void init(@NotNull BuildContext buildContext);

    /**
     * Execute the build task.
     * Implementations should regularly check if the calling {@link Thread} has been
     * interrupted.
     *
     * @return @throws InterruptedException if the calling {@link Thread} has been interrupted.
     * @throws Exception A general exception that will be handled.
     */
    @NotNull
    BuildContext call() throws InterruptedException, Exception;
}
```

You should be able to access most of the things you need through the `BuildContext` that can be retained from the init method.
Setting up Bamboo Development Environment in IDEA

To get bamboo started, you need to acquire a Commercial License to gain access to Bamboo Source code. Once you have downloaded Bamboo source code, please follow the instructions below to set-up Bamboo in IDEA:

1. Install maven 2.0.7
2. Install JDK 1.5 (Doesn't work with 1.4 ).
3. Download the settings.xml attached to this page, to your <USER_HOME>/m2/ directory
4. In the Bamboo directory run mvn clean install -Dmaven.test.skip=true

   Coping with Sun JAVA libraries
   Due to licensing restrictions, we are not allowed to re-distribute native SUN libraries through our maven2 public repositories.
   If you are developing plugins for Bamboo or building Bamboo from source, you might need
   javax.mail and javax.transaction:jta:jar for Bamboo to build successfully. Please visit our
   confluence page on Working with Sun JAVA libraries for further details.

5. Run mvn idea:clean idea:idea -DdownloadSources=true
6. This should generate atlassian-bamboo.* files in your source directory. Open the project with Idea
7. Right click on atlassian-bamboo module (on the left hand panel under Projects) and click on module
   settings. Click on the dependencies tab and add all other modules except atlassian-bamboo-web-
   server. Thus, making the atlassian-bamboo module to be dependent on all the other modules except
   atlassian-bamboo-web-server
8. Do the same for the atlassian-bamboo-web-server but make it depended only on the atlassian-
   bamboo module.
9. Add the application. Mine looks like..

   Main Class com.atlassian.bamboo.server.Server
   VM Params -Dorg.mortbay.xml.XmlParser.NotValidating=true
              -Dbamboo.home=home1 -server -Xmx512m -
              XX:MaxPermSize=126m
   Program Params 8085 path\bamboo\bamboo-web-app\src\main\webapp /
   Working Dir D:\dev\src\atlassian\bamboo-home
   Class Path of Module atlassian-bamboo-web-server
   That should be it.
Bamboo Installation & Upgrade Guide

This page last changed on Sep 10, 2007 by rosie@atlassian.com.

- **Bamboo Installation Guide**
  - Bamboo EAR-WAR Installation Guide
  - Bamboo Remote Agent Installation Guide
  - Bamboo Standalone Installation Guide (Linux)
  - Bamboo Standalone Installation Guide (Mac)
  - Bamboo Standalone Installation Guide (Windows)
  - Running the Setup Wizard
- **Bamboo Release Notes**
  - Bamboo 1.0 Release Notes
  - Bamboo 1.1 Release Notes
  - Bamboo 1.2 Release Notes
  - Bamboo 2.0 Beta Release Notes
  - Bamboo 2.0 Release Notes
- **Bamboo Upgrade Guide**
Bamboo Installation Guide

This page last changed on Jan 01, 2008 by rosie@atlassian.com.

Requirements

1. JDK/JRE1.5+. For best performance, use Sun JDK 1.5 (Java 5) and above.
   Warning: Bamboo has incompatibilities when not running under Sun JDK. Please make sure you are using Sun JDK to run your Bamboo installation.
2. A database. NOTE: Bamboo ships with a built-in HSQL database, which is fine for evaluation purposes. For production environments it is recommended that you use an enterprise database, as described in 2. Connecting Bamboo to an external database.
3. Only if you are using the Bamboo EAR-WAR distribution: A servlet container that supports Servlet 2.4 specification. Most modern containers should comply to this.

Choose your Bamboo 'Distribution':

Bamboo is available in two 'distributions':

<table>
<thead>
<tr>
<th>Standalone distribution</th>
<th>EAR-WAR distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Pre-packaged with the Jetty application server</td>
<td></td>
</tr>
<tr>
<td>• Requires virtually no setup</td>
<td></td>
</tr>
<tr>
<td>• Recommended for all users</td>
<td>• Deploys into an existing application server</td>
</tr>
<tr>
<td></td>
<td>• Requires manual configuration</td>
</tr>
<tr>
<td></td>
<td>• Suitable only for system administrators</td>
</tr>
</tbody>
</table>

The Standalone distribution is recommended even for organisations with an existing application server environment.

NEXT

• Standalone Installation Guide — Windows
• Standalone Installation Guide — Linux
• Standalone Installation Guide — Mac

or

• EAR-WAR Installation Guide

RELATED TOPICS

Running the Setup Wizard
Upgrade Guide
Release Notes
Bamboo Documentation Home
The Bamboo EAR/WAR edition is intended for deployment into an existing J2EE application server. It is assumed that you already know how to deploy a webapp on the application server of choice. If not, it is recommended to install the Bamboo Standalone edition.

The following instructions are only indicative of the process and examples are based on installing the Bamboo WAR file on Apache Tomcat. Deployment and configuration will differ based on your webserver.

⚠️ Before you begin

Please review the System Requirements.

Step 1. Download and install Bamboo EAR-WAR

1. The Bamboo WAR file is available for download here. Click the 'Show all' link on the downloads page to display the WAR file.
2. Deploy onto your application server. In Tomcat there are two ways you can do this:
   a. Place the WAR file directly into the webapps folder of Tomcat. When Tomcat starts it will perform all the necessary extractions.
   b. Extract the WAR file to your chosen directory in the webapps folder.
      📥 Windows users must avoid Win XP's built in unzip as it doesn't extract all the files. Use a third-party zip extractor like WinZip.
      📥 Solaris users will need to use GNU tar to handle the long filenames.

By default the WAR file will extract to a folder called Bamboo-<version>. Note: The name of the directory in the webapps folder will form the URL required to access Bamboo (eg. Tomcat/webapps/bamboo-1.0/ will become http://host:port/bamboo-1.0/). You may wish to change the directory name for a more concise access URL.

Step 2. Set Bamboo Home

You will need to set your Bamboo Home Directory. You can do this in one of three ways:

1. set the bamboo.home property in the file WEB-INF/classes/bamboo-init.properties to your chosen Bamboo home directory.
2. pass the Bamboo home directory to the application server as a java opt. (eg. -Dbamboo.home=C:/bamboo/bamboo-home).
3. specify an environment variable 'BAMBOO_HOME' which specifies the absolute path to your {BAMBOO_HOME} directory.

Step 3. Set jms.broker.uri

If you are going to use Bamboo remote agents, set the following in the /WEB-INF/classes/bamboo-init.properties file:

```
bamboo.jms.broker.uri=tcp://localhost:54663
```

- replace 'localhost' with the real host name or IP address of your Bamboo server.
- if port number 54663 is already in use, specify a different port number.

Step 4. Set Java OPTs

It is recommended that you set the following Java OPTs on your Application Server.

- server — Ensures that the jvm starts up in server mode. (This will perform various optimisation tasks, which is beneficial for long-running applications.)
- Xmx512m — Sets the maximum memory recommended for Bamboo.
• `-XX:MaxPermSize=256m` — Sets the maximum permgen memory recommended for Bamboo.
• `-Djava.awt.headless=true` — For Unix systems. This allows AWT to run in headless mode and is required if running Bamboo in non-graphical environments. For more details visit the [Sun Developer Network](http://java.sun.com).  

In Tomcat you can set the above Java OPTs as follows

- **Windows:**
  1. Find the `setenv.bat` file.
  2. Assign the desired properties to the `JAVA_OPTS` variable:

    ```
    set JAVA_OPTS=-server -XX:MaxPermSize=256m -Dbamboo.home=/opt/bamboo/bamboohome -Xmx512m -Djava.awt.headless=true $JAVA_OPTS
    ```

- **Linux-based systems:**
  1. Find the `setenv.sh` file
  2. Assign the desired properties to the `JAVA_OPTS` variable:

    ```
    JAVA_OPTS="-server -XX:MaxPermSize=256m -Dbamboo.home=/opt/bamboo/bamboohome -Xmx512m -Djava.awt.headless=true $JAVA_OPTS"
    export JAVA_OPTS
    ```

---

**Step 5. Restart Server**

1. Shut down, and then restart your application server.

**Step 6. Configure Bamboo**

See [Running the Setup Wizard](#).
Before you begin:

- Not sure whether to install a Remote Agent? See About Agents to understand how Remote Agents interact with your Bamboo server.
- Ensure that you have specified the Broker URL, as described in the Bamboo Setup Wizard and the Bamboo 2.0 Upgrade Guide.
- Do you have sufficient Agent licenses? See Bamboo licensing for details.
- Have you enabled the creation of Remote Agents, as described in 2.3.1 Disabling Remote Agents Support.
- Ensure that you have Java Runtime Environment 5.0 or later installed on the agent machine.

To install the Bamboo Remote Agent manually,

Step 1. Download and install the Remote Agent

1. Create a directory on the agent machine (e.g. bamboo-agent-home), to serve as the "Bamboo agent home" for the remote agent.
2. On your Bamboo server, click the 'Administration' link in the top navigation bar.
3. Click the 'Agents' link in the left navigation column.
4. This will display the 'Agents' screen, showing lists of all Local Agents and all Remote Agents that currently exist in your Bamboo system.
5. Click the 'Install Remote Agent' button.
6. The following screen will be displayed:
   - Click the 'DOWNLOAD Remote Agent JAR' button and save the JAR file to the directory you created in step 1.1.
   - Note the command under the heading 'Running a Remote Agent' for use in step 2 below.

Step 2. Launch the Remote Agent

Once installed, you can run the remote agent by executing the command line obtained in the previous step. This command will look something like the following:

```
```

You may wish to configure the remote agent machine to start the Bamboo remote agent automatically when the machine boots. Please consult your operating system documentation for instructions on how to do this.
You can also choose to run the remote agent with different command line parameters, to change where the remote agent stores its data or suppress the self-signed certificate of the server.

- Changing where the remote agent stores its data
  By default, the remote agent will store its data in a directory called `bamboo-agent-home`. If you wish to specify a different directory, add the following command line parameter:

```
--bamboo.home=RemoteAgentHome
```

where `RemoteAgentHome` is the path to the Bamboo agent home directory you created in step 1.1. Your command line will look something like this:

```
```

- Suppressing the self-signed certificate of the server
  If your Bamboo server uses SSL, the following instructions will also appear in the 'Running a Remote Agent' section:

```
Using Remote Agents With a Self-Signed Certificate
If your Bamboo server uses SSL (https) with a self-signed certificate, you will need to do one of the following:
- Add the option `--bamboo.agent.ignoreServerCertName=true` to the remote agent's command line, for example:

```
```

Please be aware that this reduces the security of your configuration, as the identity of your Bamboo server will not be authenticated by the remote agent.
- Use the `keytool` utility to add the self-signed certificate to the trusted certificates in your keystore. This is a more secure option.
```

You will need to carry out either one of the two options listed above. The first option of suppressing the self-signed certificate is simple to execute, but will reduce the security of your configuration (as described in the instructions). To suppress the self-signed certificate, add the following command line parameter:

```
--bamboo.agent.ignoreServerCertName=true
```

Your command line will look something like this:

```
```

The second option of adding the self-signed certificate to the trusted certificates in your keystore is the more secure option, but is complex to set up. For detailed instructions of how to do this, please refer to the relevant Sun documentation.

**Step 3. Configure the Remote Agent's Capabilities**

**Step 4. (Optional) Rename the Remote Agent**

Your new remote agent has been automatically given a default name (e.g. 'Remote Agent on mymachine'). If you wish to rename your new remote agent, please see 2.4 Editing an Agent's Details.
To install Bamboo Standalone on Linux,

**Step 1. Download and installing Bamboo Standalone**

Bamboo Standalone for Linux is available for download [here](#).

**Linux Archive (.tar.gz)**

1. To install Bamboo using the Linux archive version (atlassian-bamboo-x.x-standalone.tar.gz), you need to extract the files to a Bamboo installation directory of your choice. By default, the root directory of the tar file is "Bamboo".

2. You will also need to setup your Bamboo home directory — this is the directory where Bamboo will store its configuration data. To do this, open the file named `bamboo-init.properties` in the `<Bamboo installation directory>/webapp/WEB-INF/classes` directory. In this file, insert the property "bamboo.home", with an absolute path to your Bamboo home directory. Your file should look something like this:

   ```
   bamboo.home=/test/bamboo-home
   ```

   You must use forward-slashes in your directory path. Backslashes are not recognised by Bamboo.

   Alternatively, you can specify an environment variable 'BAMBOO_HOME' which specifies the absolute path to your `{BAMBOO_HOME}` directory. Bamboo will check if an environment variable is defined.

**Step 2. Launch Bamboo Standalone on Linux**

There are two ways you can launch Bamboo on Linux:

1. **Launch via `bamboo.sh` startup script**

   You can start Bamboo with the default `bamboo.sh` file in your installation root directory. The `bamboo.sh` command accepts the following options (e.g. `./bamboo.sh start`):

   - start — this starts Bamboo.
   - stop — this stops Bamboo.
   - restart — this restarts Bamboo
   - status — this provides the current status of Bamboo.

2. **Launch via Java Service Wrapper**

   The wrapper is platform specific, and doesn't work on SunOS.

   Alternatively, you can start Bamboo via a Java Service Wrapper, which provides services such as automatic restarting. To do this, you will need to use the `start-bamboo` command available in the `/wrapper` folder of the Bamboo installation. You will need to fire the command with one of the following options (e.g. `./start-bamboo start`):

   - console — this starts Bamboo in a console. The logs will scroll to standard out.
   - start — this starts Bamboo.
   - stop — this stops Bamboo.
   - restart — this restarts Bamboo
   - status — this provides the current status of Bamboo.
   - dump — stops Bamboo abruptly by killing the process

   Once Bamboo has started, you can access it by going to your web browser and entering the address: [http://localhost:8085/](http://localhost:8085/).
Step 3. Configure Bamboo

See Running the Setup Wizard.
To install Bamboo Standalone on Mac OS,

**Step 1. Download and install Bamboo Standalone**

Bamboo Standalone for Mac OS is available for download [here](#). You can choose an Installer (.dmg) or an Archive (.tgz).

**Mac OS Installer (.dmg)**

1. Launching the Bamboo Mac OS installer (atlassian-bamboo-x.x-standalone.dmg) will mount the Atlassian Bamboo installation volume. Launch the Bamboo Continuous Integration Server Installer.app to begin the installation wizard.
2. The installer requires you to specify two directories:
   - **Bamboo installation directory** — This is the directory where Bamboo's application files will be installed. The default is:
     ```
     /Applications/Bamboo
     ```
   - **Bamboo home directory** — This is the directory where Bamboo will store its configuration data. If the directory you specify doesn't exist, Bamboo will create the directory when it launches. The default is:
     ```
     /Users/<current-user>/Bamboo-home
     ```

   ! You must use forward-slashes in your directory path. Backslashes are not recognised by Bamboo.

**Mac OS Archive (.tgz)**

1. To install Bamboo using the Mac OS archive version (atlassian-bamboo-x.x-standalone.tgz), you need to extract the files to a Bamboo installation directory of your choice. By default, the root directory of your tgz file is "Bamboo".
2. You will also need to setup your Bamboo home directory — this is the directory where Bamboo will store its root configuration data. To do this, open the file named `bamboo-init.properties` in the `<Bamboo installation directory>/webapp/WEB-INF/classes` directory. In this file, insert the property "bamboo.home", with an absolute path to your Bamboo home directory. Your file should look something like this:

   ```
   bamboo.home=/test/bamboo-home
   ```

   Alternatively, you can specify an environment variable 'BAMBOO_HOME' which specifies the absolute path to your `{BAMBOO_HOME}` directory. Bamboo will check if an environment variable is defined.
3. If you are going to use Bamboo remote agents, set the following in the `bamboo-init.properties` file in the `<Bamboo installation directory>/webapp/WEB-INF/classes` directory:

   ```
   bamboo.jms.broker.uri=tcp://localhost:54663
   ```
   - replace 'localhost' with the real host name or IP address of your Bamboo server.
   - if port number 54663 is already in use, specify a different port number.

**Step 2. Launch Bamboo on Mac OS**

There are two ways you can launch Bamboo on Mac OS:
1. Launch via `bamboo.sh` startup script

You can start Bamboo with the default `bamboo.sh` file in your installation root directory. The `bamboo.sh` command accepts the following options (e.g. `.bamboo.sh start`):

- `console` — this starts Bamboo in a console. The logs will scroll to standard out.
- `start` — this starts Bamboo.
- `stop` — this stops Bamboo.
- `status` — this provides the current status of Bamboo.

2. Launch via Java Service Wrapper

Alternatively, you can start Bamboo via a Java Service Wrapper, which provides services such as automatic restarting. To do this, you will need to use the `run-bamboo` command available in the `/wrapper` folder of the Bamboo installation. You will need to fire the command with one of the following options (e.g. `.run-bamboo start`):

- `console` — this starts Bamboo in a console. The logs will scroll to standard out.
- `start` — this starts Bamboo.
- `stop` — this stops Bamboo.
- `status` — this provides the current status of Bamboo.

⚠️ Once Bamboo has started, you can access it by going to your web browser and entering the address: `http://localhost:8085/`.

**Step 3. Configure Bamboo**

See [Running the Setup Wizard](#).
To install Bamboo Standalone on Windows,

**Step 1. Download and install Bamboo Standalone**

Bamboo Standalone for Windows is available for download [here](#). You can choose the Windows Installer (.exe) or the Windows Archive (.zip).

**Windows Installer (.exe)**

1. Launch the Bamboo Windows installer (`atlassian-bamboo-x.x-standalone.exe`) to begin the installation wizard.

2. The installer requires you to specify two directories:
   - **Bamboo installation directory** — This is the directory where Bamboo's application files will be installed. The default is:
     
     \[ C:/Program Files/Bamboo \]

   - **Bamboo home directory** — This is the directory where Bamboo will store its configuration data. If the directory you specify doesn't exist, Bamboo will create the directory when it launches. The default is:
     
     \[ C:/Documents and Settings/\textit{current-user}/Bamboo-home \]

   You must use forward-slashes in your directory path. Backslashes are not recognised by Bamboo.

**Windows Archive (.zip)**

1. To install Bamboo using the Windows archive version (`atlassian-bamboo-x.x-standalone.zip`), you need to extract the files to a Bamboo installation directory of your choice. By default, the root directory in your zip file is named "Bamboo".

2. You will also need to setup your Bamboo home directory — this is the directory where Bamboo will store its root configuration data. To do this, edit the file named `bamboo-init.properties` in the `Bamboo/webapp/WEB-INF/classes` directory. In this file, insert the property "bamboo.home", with an absolute path to your Bamboo home directory. Your file should look something like this:

   ```properties
   bamboo.home=C:/test/bamboo-home
   ```

   Alternatively, you can specify an environment variable 'BAMBOO_HOME' which specifies the absolute path to your {BAMBOO_HOME} directory. Bamboo will check if an environment variable is defined.

3. If you are going to use Bamboo remote agents, set the following in the `bamboo-init.properties` file in the `<Bamboo installation directory>/webapp/WEB-INF/classes` directory:

   ```properties
   bamboo.jms.broker.uri=tcp://localhost:54663
   ```

   - replace 'localhost' with the real host name or IP address of your Bamboo server.
   - if port number 54663 is already in use, specify a different port number.

**Step 2. Launch Bamboo**

Once Bamboo is installed on your machine, you can launch the application either via the Start Menu (if you have used the self installer), or by running the batch files available in the root of the Bamboo installation directory. You can run Bamboo in two modes: either in the console, or as a Windows service (note: if you are running Bamboo as a service, please ensure that the Windows user which the Bamboo service is running as has the necessary privileges, and is logged into Windows).
Bamboo comes with the following batch files:

- BambooConsole.bat — this starts Bamboo in a Windows console.
- InstallAsService.bat — this installs Bamboo as a Windows service. Note that this will not start Bamboo.
- StartBamboo.bat — this starts your installed Bamboo Windows service.
- StopBamboo.bat — this stops your installed Bamboo Windows service
- UninstallService.bat — this un-installs the Bamboo Windows service from your machine. Note that your Bamboo installation still remains.

Once Bamboo has started, you can access it by going to your web browser and entering the default address: [http://localhost:8085/](http://localhost:8085/)

Step 3. Configure Bamboo

See [Running the Setup Wizard](#).
Running the Setup Wizard

This page last changed on Jan 29, 2008 by alui.

Step 1. Installation Settings

When you launch Bamboo for the first time, you will need to provide some configuration information before you can start using it.

Welcome to Atlassian Bamboo!

Welcome to Bamboo Continuous Integration Server. Bamboo needs some information before it is fully installed.

Standard Installation Settings

Licensing

Server ID — A25P-FZ32-RAWT-KVQY
License Key — Please enter your Bamboo license key above - either commercial or evaluation. Please contact sales if you require a license key.

System Paths and Directories

Configuration Directory — /opt/dev/data/bamboo-home-functional-20090115/semi-data/configuration
Build Data Directory — /opt/dev/data/bamboo-home-functional-20090115/semi-data/build
Broker URL — tcp://napporo.sydney.atlassian.com:54663

You may find it simplest to keep the default settings for the above three directories. For more information please see 7.1 Locating Important Directories and Files.

'Broker URL' — This is the URL of the embedded messaging broker that Bamboo sets up to communicate with its remote build agents. This URL will be written to bamboo.cfg.xml as a property, but can also be manually specified as a system property. The system property, if specified, will override the bamboo.cfg.xml property.

tcp://localhost:54663

Replace localhost with the real host name or IP address of your Bamboo server. You should not use localhost as the host name in the Broker URL, as remote agents are provided with the Broker URL on startup and use it to communicate to the server.

If port number 54663 is already in use, specify a different port number.

Step 2. Database

Here, you will choose what type of database Bamboo will use:
Choose a Database Configuration

Choose where Bamboo should store its data

Select Database

- Embedded Database
- External Database

Embedded Database
The embedded database will allow Bamboo to operate without an external database.

External Database
If you wish to store your Bamboo data in an external database, choose it from the list of supported databases. This is recommended for production systems. If your database is not listed in the menu, you may configure an 'Unsupported Database', but be aware that Bamboo may not be fully tested.

- 'Embedded Database' — choose this for quick and easy first-time installation of Bamboo.
  Note that the embedded HSQL database is suitable for evaluation purposes only. You should later move to an external database before deploying Bamboo in production, as described in 7.4 Moving your Bamboo Data to a different Database.
  OR:
  - 'External Database' — if you wish to use an external database, please see 2. Connecting Bamboo to an external database.

Step 3. Starting Data

Here, you will tell Bamboo how to populate the 'home directory' that you setup when you installed Bamboo.

Select Starting Data for Bamboo

Would you like to:
- Create new Bamboo home
- Import existing data

- 'Create new Bamboo home' — choose this if you are performing a normal installation or upgrade.
  OR:
  - 'Import existing data' — only choose this under exceptional circumstances, e.g. if you are connecting Bamboo to a different database, or moving your pre-existing Bamboo installation to a different server.

Step 4. Bamboo Administrator

Here, you will enter the details of the first registered user to the Bamboo system. This user will have global administrative privileges over the entire installation of Bamboo and should not be removed.
Step 5. Server Configuration

The final page of the Setup Wizard allows you to enter some final configuration data for Bamboo.

- 'Name' — for more details please see 8.3 Specifying Bamboo's Title.
- 'Base URL' — for more details please see 8.4 Specifying Bamboo's URL.
- 'Apply gzip compression to reduce the size of Bamboo's web pages?' — for more details please see 8.5 Enabling GZIP Compression.
- 'Accept remote API calls?' — for more details please see 8.6 Enabling Bamboo's Remote API.

Once you have clicked "Complete Installation", the setup process is done and you are now at the Bamboo dashboard.

Next...

1.1 About Projects, Plans and Builds
1.2 Creating a Plan
1. Using Bamboo's embedded HSQL database

For quick and easy installation, use the embedded HSQL database that ships with Bamboo.

⚠️ Note that the embedded HSQL database is suitable for evaluation purposes only. You should move to an external database before deploying Bamboo in production. For details please see 7.4 Moving your Bamboo Data to a different Database.

To use Bamboo's embedded HSQL database,

1. At Step 2 of the Setup Wizard, choose 'Embedded Database'.
2. Click the 'Continue' button.
3. Go to Step 3 of the Setup Wizard.

RELATED TOPICS

• 1. Using Bamboo's embedded HSQL database
• 2. Connecting Bamboo to an external database
  ° 2.1 MySQL 4.1 and 5.0
  ° 2.2 Postgres 8+
  ° 2.3 Oracle 9i and 10g
  ° 2.4 Microsoft SQL Server
  ° 2.5 Unsupported databases

Bamboo Documentation Home
2. Connecting Bamboo to an external database

Bamboo can be connected to an external database. For details and instructions please see:

- 2.1 MySQL 4.1 and 5.0
- 2.2 Postgres 8+
- 2.3 Oracle 9i and 10g
- 2.4 Microsoft SQL Server
- 2.5 Unsupported databases

⚠️ Before you begin
Please note: if you are already using Bamboo with the embedded HSQL database (or any other database), and you want to keep your data, please see 7.4 Moving your Bamboo Data to a different Database.

RELATED TOPICS

- 1. Using Bamboo’s embedded HSQL database
- 2. Connecting Bamboo to an external database
  - 2.1 MySQL 4.1 and 5.0
  - 2.2 Postgres 8+
  - 2.3 Oracle 9i and 10g
  - 2.4 Microsoft SQL Server
  - 2.5 Unsupported databases

Bamboo Documentation Home
2.1 MySQL 4.1 and 5.0

Before you begin
Please ensure that your MySQL database server is set to 'utf8' character encoding, and not 'latin1' character encoding. For details please see:

- MySQL 4.1 documentation: Database Character Set
- MySQL 5 documentation: Database Character Set

First, you need to choose how you will connect to the MySQL database. Please follow the instructions for your chosen method:

JDBC is generally simpler, and is therefore the recommended method.

Connecting via JBDC

To connect Bamboo to a MySQL database, via JDBC,

1. Put the MySQL JDBC driver jar file ([download here](link)) into your application server's classpath:
   - For the Bamboo Standalone distribution, copy the jar file into the `webapp/WEB-INF/lib` directory.
   - For the Bamboo EAR-WAR distribution, the location will depend on which application server you are using.
2. At Step 2 of the Bamboo Setup Wizard, choose 'External Database' and select 'MySQL' from the list.
3. The 'Select Database Connection' screen will appear. Select 'Direct JDBC connection'.
4. The 'Setup JDBC Connection' screen will appear as shown in the screenshot below.
   - 'Driver Class Name' — Type the following: `com.mysql.jdbc.Driver`
   - 'Database URL' — Type the URL where Bamboo will access your database. For syntax, please see the MySQL documentation.
     
     Include the `autoReconnect=true` flag, the `useUnicode=true` flag and the `characterEncoding=utf8` flag, e.g.:
     ```
     jdbc:mysql://localhost/bamboo?autoReconnect=true&useUnicode=true&characterEncoding=utf8
     ```
     
     If the `autoReconnect` flag is not set, the MySQL JDBC driver will eventually time out and Bamboo will no longer be able to communicate with the database.
   - 'User Name' — Type the username that Bamboo will use to access your database.
   - 'Password' — Type the password (if required) that Bamboo will use to access your database.
5. Select the 'Overwrite existing data' checkbox if you wish Bamboo to overwrite any tables that already exist in the database.
6. Go to Step 3 of the Setup Wizard.

Screenshot 1: 'Setup JDBC Connection (MySQL)'
Connecting via a datasource

To connect Bamboo to a MySQL database, via a datasource,

1. Configure a datasource in your application server (consult your application server documentation for details). For the syntax of the JDBC URL to use, please see the MySQL documentation.
   - In the JDBC URL that you configure in your application server, include the `autoReconnect=true` flag, the `useUnicode=true` flag and the `characterEncoding=utf8` flag, e.g.:
     ```
     jdbc:mysql://localhost/bamboo?autoReconnect=true&useUnicode=true&characterEncoding=utf8
     ```
   - If the `autoReconnect` flag is not set, the MySQL JDBC driver will eventually time out and Bamboo will no longer be able to communicate with the database.
2. At Step 2 of the Bamboo Setup Wizard, choose 'External Database' and select 'MySQL' from the list.
3. The 'Select Database Connection' screen will appear. Select 'Connect via a datasource (configured in the application server)'.
4. The 'Setup Datasource Connection' screen will appear as shown in the screenshot below. In the 'JNDI name' field, type the JNDI name of your datasource, as configured in your application server.
   - If `java:comp/env/jdbc/DataSourceName` doesn't work, try `jdbc/DataSourceName` (and vice versa).
5. Select the 'Overwrite existing data' checkbox if you wish Bamboo to overwrite any tables that already exist in the database.
6. Go to Step 3 of the Setup Wizard.

Screenshot 2: 'Setup Datasource Connection'

RELATED TOPICS

- 1. Using Bamboo's embedded HSQL database
- 2. Connecting Bamboo to an external database
  - 2.1 MySQL 4.1 and 5.0
  - 2.2 Postgres 8+
  - 2.3 Oracle 9i and 10g
  - 2.4 Microsoft SQL Server
  - 2.5 Unsupported databases
2.2 Postgres 8+

Before you begin

If connecting to a remote PostgreSQL server (i.e. if your PostgreSQL server is not installed locally on your Bamboo server host system), please ensure that your `data/postgresql.conf` and `data/pg_hba.conf` files are configured to accept remote TCP connections from the Bamboo server's IP address. Refer to the PostgreSQL documentation for the `listen_addresses` value in the `postgresql.conf` file, as well as documentation for the `pg_hba.conf` file, for enabling TCP connections to your PostgreSQL server. (Note that you will need to restart PostgreSQL once any changes to these files have been made.) See:

- PostgreSQL 8.0 documentation
- PostgreSQL 8.1 documentation
- PostgreSQL 8.2 documentation

First, you need to choose how you will connect to the Postgres database. Please follow the instructions for your chosen method:

Error formatting macro: toc: java.lang.NullPointerException

JDBC is generally simpler, and is therefore the recommended method.

Connecting via JBDC

To connect Bamboo to a Postgres database, via JDBC,

1. Put the Postgres JDBC driver `jar` file (download here) into your application server's classpath:
   - For the Bamboo Standalone distribution, copy the `jar` file into the `webapp/WEB-INF/lib` directory.
   - For the Bamboo EAR-WAR distribution, the location will depend on which application server you are using.
2. At Step 2 of the Bamboo Setup Wizard, choose 'External Database' and select 'Postgres' from the list.
3. The 'Select Database Connection' screen will appear. Select 'Direct JDBC connection'.
4. The 'Setup JDBC Connection' screen will appear as shown in the screenshot below.
   - 'Driver Class Name' — Type the following: `org.postgresql.Driver`
   - 'Database URL' — Type the URL where Bamboo will access your database. For syntax, please see the PostgreSQL JDBC driver documentation.
   - 'User Name' — Type the username that Bamboo will use to access your database.
   - 'Password' — Type the password (if required) that Bamboo will use to access your database.
5. Select the 'Overwrite existing data' checkbox if you wish Bamboo to overwrite any tables that already exist in the database.
6. Go to Step 3 of the Setup Wizard.

Screenshot 1: 'Setup JDBC Connection (Postgres)'

```javascript
// Code snippet
```
Connecting via a datasource

To connect Bamboo to a Postgres database, via a datasource,

1. Configure a datasource in your application server (consult your application server documentation for details). For the syntax of the JDBC URL to use, please see the Postgres JDBC driver documentation.
2. At Step 2 of the Bamboo Setup Wizard, choose 'External Database' and select 'Postgres' from the list.
3. The 'Select Database Connection' screen will appear. Select 'Connect via a datasource (configured in the application server)'.
4. The 'Setup Datasource Connection' screen will appear as shown in the screenshot below. In the 'JNDI name' field, type the JNDI name of your datasource, as configured in your application server. If java:comp/env/jdbc/DataSourceName doesn't work, try jdbc/DataSourceName (and vice versa).
5. Select the 'Overwrite existing data' checkbox if you wish Bamboo to overwrite any tables that already exist in the database.
6. Go to Step 3 of the Setup Wizard.

Screenshot 2: 'Setup Datasource Connection'

RELATED TOPICS

- 1. Using Bamboo's embedded HSQL database
- 2. Connecting Bamboo to an external database
  - 2.1 MySQL 4.1 and 5.0
  - 2.2 Postgres 8+
  - 2.3 Oracle 9i and 10g
  - 2.4 Microsoft SQL Server
  - 2.5 Unsupported databases

Bamboo Documentation Home
2.3 Oracle 9i and 10g

Bamboo provides two ways to connect to an Oracle database — via JDBC or via a datasource. JDBC is generally simpler and is the recommended method.

⚠️ Drivers for Oracle connectivity

We recommend using the following configuration to run Bamboo with Oracle:

- Regardless of what version of the Oracle database you are using, you should use the Oracle 10g JDBC drivers. (Note: Oracle 10g JDBC drivers will not work with Oracle 8.1.6. For further reference, see Oracle FAQ)
- We highly recommend using the thin drivers.

Error formatting macro: toc: java.lang.NullPointerException

Connecting via JDBC

To connect Bamboo to a Oracle database, via JDBC,

1. Put the Oracle JDBC driver jar file into your application server's classpath:
   - For the Bamboo Standalone distribution, copy the jar file into the /webapp/WEB-INF/lib directory.
   - For the Bamboo EAR-WAR distribution, the location will depend on which application server you are using.
2. At Step 2 of the Bamboo Setup Wizard, choose 'External Database' and select 'Oracle' from the list.
3. The 'Select Database Connection' screen will appear. Select 'Direct JDBC connection'.
4. The 'Setup JDBC Connection' screen will appear as shown in the screenshot below.
   - 'Driver Class Name' — Type the following: oracle.jdbc.driver.OracleDriver
   - 'Database URL' — Type the URL where Bamboo will access your database. For syntax, please see the Oracle documentation, e.g. jdbc:oracle:thin:@localhost:1521:SID
   - 'Username' — Type the username that Bamboo will use to access your database.
   - 'Password' — Type the password that Bamboo will use to access your database.
5. Select the 'Overwrite existing data' checkbox if you wish Bamboo to overwrite any tables that already exist in the database.
6. Click 'Continue' to finish specifying your connection settings.

Screenshot 1: 'Setup JDBC Connection (Oracle)'

<table>
<thead>
<tr>
<th>Setup JDBC Connection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Driver Class Name:</td>
</tr>
<tr>
<td>Database URL:</td>
</tr>
<tr>
<td>Username:</td>
</tr>
<tr>
<td>Password:</td>
</tr>
</tbody>
</table>

Connecting via a datasource

To connect Bamboo to a Oracle database, via a datasource,

1. Configure a datasource in your application server (consult your application server documentation for details). For the syntax of the JDBC URL to use, please see the Oracle documentation.
2. At Step 2 of the Bamboo Setup Wizard, choose 'External Database' and select 'Oracle' from the list.

3. The 'Select Database Connection' screen will appear. Select 'Connect via a datasource (configured in the application server)'.

4. The 'Setup Datasource Connection' screen will appear as shown in the screenshot below. In the 'JNDI name' field, type the JNDI name of your datasource, as configured in your application server.
   
   □ If java:comp/env/jdbc/DataSourceName doesn't work, try jdbc/DataSourceName (and vice versa).

5. Select the 'Overwrite existing data' checkbox if you wish Bamboo to overwrite any tables that already exist in the database.

6. Click 'Continue' to finish specifying your connection settings.

Screenshot 2: 'Setup Datasource Connection'

 RELATED TOPICS

- **1. Using Bamboo's embedded HSQL database**
- **2. Connecting Bamboo to an external database**
  - 2.1 MySQL 4.1 and 5.0
  - 2.2 Postgres 8+
  - 2.3 Oracle 9i and 10g
  - 2.4 Microsoft SQL Server
  - 2.5 Unsupported databases

Bamboo Documentation Home
example, WordNet, Google Urban Dictionary, Answers.com, rhymezone.com, Merriam-Webster<}
2.4 Microsoft SQL Server

Before you begin

We strongly recommend, using the open source JTDS JDBC driver version 1.0.3 or above, for MS SQL Server integration.

Bamboo provides two ways to connect to an MS SQL Server database — via JDBC or via a datasource. JDBC is generally simpler and is the recommended method.

Error formatting macro: toc: java.lang.NullPointerException

Connecting via JBDC

To connect Bamboo to a MS SQL Server database, via JDBC,

1. Copy the MS SQL Server JDBC driver jar (download here) file into your application server's classpath:
   • For the Bamboo Standalone distribution, copy the jar file into the webapp/WEB-INF/lib directory.
   • For the Bamboo EAR-WAR distribution, the location will depend on which application server you are using.
2. At Step 2 of the Bamboo Setup Wizard, choose 'External Database' and select 'MSSQL' from the list.
3. The 'Select Database Connection' screen will appear. Select 'Direct JDBC connection'.
4. The 'Setup JDBC Connection' screen will appear as shown in the screenshot below.
   • 'Driver Class Name' — Type the following: net.sourceforge.jtds.jdbc.Driver
   • 'Database URL' — Type the URL where Bamboo will access your database. For syntax, please see this MS SQL Server documentation e.g: jdbc:jtds:sqlserver://localhost:1433/<database>
   • 'Username' — Type the username that Bamboo will use to access your database.
   • 'Password' — Type the password that Bamboo will use to access your database.
5. Select the 'Overwrite existing data' checkbox if you wish Bamboo to overwrite any tables that already exist in the database.
   • Mixed mode Windows Authentication.
     On a typical MS SQL Server installation, 'Windows Authentication' mode is the default security mode. This means that if you try to connect to the database with a database user, MS SQL Server will throw the following error, "Login failed for user 'sa'. Reason: Not associated with a trusted SQL Server connection." To resolve this, you can either,
     • Enable 'Mixed Mode Authentication'
     • Log in with a Windows user account, which has permission to administer the Bamboo database.
6. Click 'Continue' to finish specifying your connection settings.

Screenshot 1: 'Setup JDBC Connection (MS SQL Server)'
Connecting via a datasource

To connect Bamboo to a MS SQL Server, via a datasource,

1. Configure a datasource in your application server (consult your application server documentation for details). For the syntax of the MS SQL Server URL to use, please see the MS SQL Server documentation.
2. At Step 2 of the Bamboo Setup Wizard, choose 'External Database' and select 'MSSQL' from the list.
3. The 'Select Database Connection' screen will appear. Select 'Connect via a datasource (configured in the application server)'.
4. The 'Setup Datasource Connection' screen will appear as shown in the screenshot below. In the 'JNDI name' field, type the JNDI name of your datasource, as configured in your application server.
   - If java:comp/env/jdbc/DataSourceName doesn't work, try jdbc/DataSourceName (and vice versa).
5. Select the 'Overwrite existing data' checkbox if you wish Bamboo to overwrite any tables that already exist in the database.
   - Mixed mode Windows Authentication.
     On a typical MS SQL Server installation, 'Windows Authentication' mode is the default security mode. This means that if you try to connect to the database with a database user, MS SQL Server will throw the following error, "Login failed for user 'sa'. Reason: Not associated with a trusted SQL Server connection." To resolve this, you can either,
       • Enable 'Mixed Mode Authentication'
       • Log in with a Windows user account, which has permission to administer the Bamboo database.

   For further information, please consult this MSDN article.
6. Click 'Continue' to finish specifying your connection settings.

Screenshot 2: 'Setup Datasource Connection'

RELATED TOPICS

• 1. Using Bamboo's embedded HSQL database
• 2. Connecting Bamboo to an external database
   • 2.1 MySQL 4.1 and 5.0
   • 2.2 Postgres 8+
   • 2.3 Oracle 9i and 10g
   • 2.4 Microsoft SQL Server
   • 2.5 Unsupported databases

Bamboo Documentation Home
2.5 Unsupported databases

First, you need to choose how you will connect to your database. Please follow the instructions for your chosen method:

JDBC is generally simpler, and is therefore the recommended method.

Connecting via JBDC

To connect Bamboo to an unsupported database, via JDBC,

1. Put the appropriate JDBC driver jar file into your application server's classpath:
   - For the Bamboo Standalone distribution, copy the jar file into the webapp/WEB-INF/lib directory.
   - For the Bamboo EAR-WAR distribution, the location will depend on which application server you are using.
2. At Step 2 of the Bamboo Setup Wizard, choose 'External Database' and select 'Unsupported Database' from the list.
3. The 'Select Database Connection' screen will appear. Select 'Direct JDBC connection'.
4. The 'Setup JDBC Connection' screen will appear as shown in the screenshot below.
   - 'Driver Class Name' — Type the classname of your JDBC driver (consult your JDBC driver documentation for details).
   - 'Database URL' — Type the URL where Bamboo will access your database (consult your JDBC driver documentation for details).
   - 'User Name' — Type the username that Bamboo will use to access your database.
   - 'Password' — Type the password (if required) that Bamboo will use to access your database.
   - 'Hibernate Dialect' — Type the Hibernate dialect for your particular database:
     
     ![Screenshot 1: 'Setup JDBC Connection (Unsupported Database)'](image)

     NOTE: the databases on this list are not supported by Atlassian. Using these databases is not recommended as there is no guarantee that they will operate correctly with Bamboo. Please consider using a supported database instead.

     | Database          | Dialect                        |
     |-------------------|-------------------------------|
     | DB2              | net.sf.hibernate.dialect.DB2Dialect |
     | DB2 AS/400       | net.sf.hibernate.dialect.DB2400Dialect |
     | DB2 OS390        | net.sf.hibernate.dialect.DB2390Dialect |
     | Oracle 9/10g     | net.sf.hibernate.dialect.Oracle9Dialect |
     | Oracle (other versions) | net.sf.hibernate.dialect.OracleDialect |
     | Sybase           | net.sf.hibernate.dialect.SybaseDialect |
     | Sybase Anywhere  | net.sf.hibernate.dialect.SybaseAnywhereDialect |
     | Microsoft SQL Server | net.sf.hibernate.dialect.SQLServerDialect |
     | SAP DB           | net.sf.hibernate.dialect.SAPDBDialect |
     | Informix         | net.sf.hibernate.dialect.InformixDialect |
     | Ingres           | net.sf.hibernate.dialect.IngresDialect |
     | Progress         | net.sf.hibernate.dialect.ProgressDialect |
     | Mckoi SQL        | net.sf.hibernate.dialect.MckoiDialect |
     | Interbase        | net.sf.hibernate.dialect.InterbaseDialect |
     | Pointbase        | net.sf.hibernate.dialect.PointbaseDialect |
     | FrontBase        | net.sf.hibernate.dialect.FrontbaseDialect |
     | Firebird         | net.sf.hibernate.dialect.FirebirdDialect |
5. Select the 'Overwrite existing data' checkbox if you wish Bamboo to overwrite any tables that already exist in the database.
6. Go to Step 3 of the Setup Wizard.
Connecting via a datasource

To connect Bamboo to an unsupported database, via a datasource,

1. Configure a datasource in your application server (consult your application server documentation for details). For the syntax of the JDBC URL to use, please see your JDBC driver documentation.
2. At Step 2 of the Bamboo Setup Wizard, choose 'External Database' and select 'Unsupported Database' from the list.
3. The 'Select Database Connection' screen will appear. Select 'Connect via a datasource (configured in the application server)'.
4. The 'Setup Datasource Connection' screen will appear as shown in the screenshot below. In the 'JNDI name' field, type the JNDI name of your datasource, as configured in your application server. If java:comp/env/jdbc/DataSourceName doesn't work, try jdbc/DataSourceName (and vice versa).
5. Select the 'Overwrite existing data' checkbox if you wish Bamboo to overwrite any tables that already exist in the database.
6. Go to Step 3 of the Setup Wizard.

Screenshot 2: 'Setup Datasource Connection'

RELATED TOPICS

- 1. Using Bamboo's embedded HSQL database
- 2. Connecting Bamboo to an external database
  - 2.1 MySQL 4.1 and 5.0
  - 2.2 Postgres 8+
  - 2.3 Oracle 9I and 10g
  - 2.4 Microsoft SQL Server
  - 2.5 Unsupported databases

Bamboo Documentation Home
Bamboo Release Notes

This page last changed on Oct 17, 2007 by rosie@atlassian.com.

Latest Version

✔ Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.

Previous Releases

- **Bamboo 2.0 Beta Release Notes**
  - Bamboo 2.0 Beta 9 Release Notes
    - Bamboo 2.0 Beta 9 Upgrade Guide
  - Bamboo 2.0 Beta 8 Release Notes
    - Bamboo 2.0 Beta 8 Upgrade Guide
  - Bamboo 2.0 Beta 6 Release Notes
    - Bamboo 2.0 Beta 6 Upgrade Guide
  - Bamboo 2.0 Beta 5 Release Notes
    - Bamboo 2.0 Beta 5 Upgrade Guide
  - Bamboo 2.0 Beta 4 Release Notes
    - Bamboo 2.0 Beta 4 Upgrade Guide
  - Bamboo 2.0 Beta Upgrade Guide
  - Bamboo 2.0 Beta 3 Release Notes
    - Bamboo 2.0 Beta 3 Upgrade Guide
  - Bamboo 2.0 Beta 2 Release Notes
    - Bamboo 2.0 Beta 2 Upgrade Guide
  - Bamboo 2.0 Beta 1 Release Notes
    - Bamboo 2.0 Beta 1 Upgrade Guide
- **Bamboo 2.0 Release Notes**
  - Bamboo 2.0 Upgrade Guide
- **Bamboo 1.2 Release Notes**
  - Bamboo 1.2.4 Release Notes
    - Bamboo 1.2.4 Upgrade Guide
  - Bamboo 1.2.3 Release Notes
    - Bamboo 1.2.3 Upgrade Guide
  - Bamboo 1.2.2 Release Notes
    - Bamboo 1.2.2 Upgrade Guide
  - Bamboo 1.2.1 Release Notes
    - Bamboo 1.2.1 Upgrade Guide
  - Bamboo 1.2 Plugin Interface Changes
  - Bamboo 1.2 Upgrade Guide
- **Bamboo 1.1 Release Notes**
  - Bamboo 1.1.2 Release Notes
    - Bamboo 1.1.2 Upgrade Guide
  - Bamboo 1.1.1 Release Notes
    - Bamboo 1.1.1 Upgrade Guide
  - Bamboo 1.1 Upgrade Guide
- **Bamboo 1.0 Release Notes**
  - Bamboo 1.0.5 Release Notes
    - Bamboo 1.0.5 Upgrade Guide
  - Bamboo 1.0.4 Release Notes
    - Bamboo 1.0.4 Upgrade Guide
  - Bamboo 1.0.3 Release Notes
    - Bamboo 1.0.3 Upgrade Guide
  - Bamboo 1.0.2 Release Notes
    - Bamboo 1.0.2 Upgrade Guide
  - Bamboo 1.0.1 Release Notes
    - Bamboo 1.0.1 Upgrade Guide
  - Bamboo 1.0 Upgrade Guide
  - Bamboo 1.0-Beta Release Notes
    - Bamboo 1.0-Beta Upgrade Guide
Atlassian is proud to announce the final release of Bamboo 1.0! Bamboo 1.0 is the first official release of Atlassian’s new Continuous Integration and Build Server.

Bamboo is more than just a build server — it is an entire Build Telemetry system designed to provide you with unprecedented insight into your development processes.

To check out Bamboo’s features and see what it can do for you, please visit our Feature Tour.

⚠️ Upgrading from a pre-release version? Please see the Bamboo 1.0 Upgrade Guide.

⚠️ Doing an upgrade? Make sure you re-index Bamboo by going to the Administration section and hitting ‘Re-index’.

Changes since RC2

The final steps to 1.0 since RC2 has been focused on resolving issues. Release 1.0 includes over 30 issues resolved.

In addition, the 1.0 release also sports another revised “All Plans” tab in the dashboard.

Other updates and bug fixes.

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-875</td>
<td>User page no longer show tabs with author information on them</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-857</td>
<td>Document our external Javascript widgets</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-821</td>
<td>Unable to export build configuration - no info on how to repair</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-879</td>
<td></td>
<td></td>
<td>Resolved</td>
</tr>
</tbody>
</table>
Where are my nice build result commit message tool tips?

**BAM-878** NumberFormatException for Test
Resolved

**BAM-876** Move the Clover plugin to opensource as an example of a Bamboo plugin
Resolved

**BAM-873** All Projects table shows the 'little hand' icon over rows that can't be expanded
Resolved

**BAM-867** test mail should contain clickable base url
Resolved

**BAM-860** New more condensed dashboard
Resolved

**BAM-839** Tests Page Has URL Escapes
Resolved

**BAM-837** Allow Properties to be Passed to Ant
Resolved

**BAM-835** Build completed time on summary page is actually build start time
Resolved

**BAM-803** Use minified version of js libs in 1.0 final
Resolved

**BAM-796** Number Format Exception
Resolved

**BAM-788** IM bot should reconnect before sending message if it was disconnected
Resolved

**BAM-783** Build test result tab taking to long to load.
Resolved

**BAM-779** Report graphs are not displaying data: build duration and # of tests
Resolved

**BAM-778** Clicking the previous build button while viewing changes - got stacktrace
Resolved

**BAM-775** Disabled plans should have visual cue on the Summary page.
Resolved

**BAM-722** Administrator's Guide
Resolved

**BAM-721** Bamboo User Guide
Resolved

**BAM-720** Upgrade Guide (generic)
Resolved

**BAM-719** Release Notes & Upgrade Guides: reformat as per JIRA's/Confluence's
Closed

**BAM-718** Installation Guide (WAR)
Resolved

**BAM-717** Installation Guide (Standalone)
Closed

**BAM-706** Added two builds to the queue, canceled the 2nd one, got a hibernate exception
Resolved

**BAM-628** Test reponsibility for a build summary
Resolved

**BAM-610** HTML in test output log is doubly-escaped
Resolved

**BAM-16** Ability to externally embed full build status
Resolved
| BAM-807 | Bamboo passes bad parameter diff ViewVC | 🔄 | 🍰 | Resolved |
| BAM-770 | Bamboo User ID should also be a repository alias | 🔄 | 🍰 | Resolved |
| BAM-690 | Improve validation for CVS :ext | 🔄 | 🍰 | Resolved |
| BAM-499 | Use 307 response code instead of meta-refresh when hitting path of bamboo | 🔄 | 🍰 | Resolved |
| BAM-433 | exe installer home setting issue | 🔄 | 🍰 | Resolved |
| BAM-683 | Allow multiple tabs on dashboard page | 🔄 | 🍰 | Resolved |
Atlassian is proud to announce the release of Bamboo 1.0.1! Bamboo 1.0.1 is largely a bug fix build with over 20 issues resolved, including:

- Support for SVN cached default authentication.
- IE7 Javascript issues.
- Startup Script issues.

New startup procedures for Mac OS X and Linux distributions

The Bamboo startup procedure for Mac OS X and Linux distributions have now changed. Instead of using the Java Service Wrapper by invoking `run-bamboo` (in Mac OS X) or `start-bamboo` in Linux, the default startup script has been replaced by a generic `bamboo.sh` script in the root Bamboo installation folder. Using this script bypasses the Java Service Wrapper.

Usages for bamboo.sh

- **start** - starts Bamboo
- **stop** - stops Bamboo
- **console** - runs Bamboo in the console
- **status** - checks the status of Bamboo.

The Java Service Wrapper is still available, and you can startup Bamboo with it if you so choose. To do this, simply run your startup command in the `/wrapper` folder rather than the installation root folder.

Updates and Issues fixed.

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-943</td>
<td>Sessions need to closed in the finally block</td>
<td>![ ]</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-937</td>
<td>Importing data doesn't guarantee unique ids</td>
<td>![ ]</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-935</td>
<td>SVN Repository doesn't use default authentication</td>
<td>![ ]</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-958</td>
<td>Links to source is broken</td>
<td>![ ]</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-939</td>
<td>Need to convert build level plugins to use web fragments</td>
<td>![ ]</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-908</td>
<td>Standalone Bamboo cannot start in certain Linux environments</td>
<td>![ ]</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-889</td>
<td>Precedence: bulk mail header cuasing notifications to be blocked</td>
<td>![ ]</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-975</td>
<td>Edit Configuraton broken</td>
<td>![ ]</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-965</td>
<td>Plan Summary does not render in IE 7 when logged in as a user</td>
<td>![ ]</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-936</td>
<td>Smack Client does not recognize project/plan keys with numbers</td>
<td>![ ]</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-933</td>
<td>Export and Import doesn't work when moving to a new Bamboo Home path</td>
<td>![ ]</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-925</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Issue</td>
<td>Description</td>
<td>Resolution</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>BAM-917</td>
<td>Viewing User via authors and via profile need to be separate requests as different info is needed in both.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-899</td>
<td>If an initial build has no queues to go into, it may cause repeated clean builds.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-897</td>
<td>Unable to Edit Build Configuration.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-897</td>
<td>Null pointer exception creating a build plan that uses svn+ssh.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-851</td>
<td>Bamboo cannot run on 64-bit linux machines.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-848</td>
<td>Responding via IM to build notifications is unreliable, the comment is not ascribed to me.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-841</td>
<td>IE7 Fails Often, Cancels Page.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-790</td>
<td>IE 7 sometimes can't display build page.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-746</td>
<td>Headless Unix Server.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-692</td>
<td>Manual build strategy gobbles up CVS errors.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-634</td>
<td>Sample plugin: Out of Memory tagging.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-297</td>
<td>&quot;Disabled&quot; status should be noted prominently on build summary.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-90</td>
<td>Lower priority of spawned build processes.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-959</td>
<td>Broken builds have incorrect links when restarting builds.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-918</td>
<td>Builders edit screen not populating existing values.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-916</td>
<td>Subversion Event Handler is not all that Null safe.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-869</td>
<td>Error creating new build plan.</td>
<td>Resolved</td>
<td></td>
</tr>
</tbody>
</table>
Bamboo 1.0.1 Upgrade Guide

This page last changed on Mar 06, 2007 by edwin@atlassian.com.

Upgrading from Bamboo 1.0 to 1.0.1

Please follow the Bamboo Upgrade Guide

⚠️ You will need to reindex your data after the upgrade is complete and Bamboo has started. To do this, go to the indexing page under the Administration section in Bamboo.

Upgrading from Bamboo 1.0 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.0.2 Release Notes

This page last changed on Dec 03, 2007 by rosie@atlassian.com.

✔ Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.

Atlassian is proud to announce the release of Bamboo 1.0.2! Bamboo 1.0.2 is mainly a bug fix release with over 10 issues resolved.

In addition, Bamboo 1.0.2 also sees added support for ssh private key authentication for both Subversion and CVS repositories.

Updates and Issues fixed

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-982</td>
<td>Bamboo fails to start under JDK 1.4</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1098</td>
<td>No page associated with this URI</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1026</td>
<td>Links Are Incorrect When Using 'latest' as the build in the URL</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1023</td>
<td>Internal error when deleting plans with dependencies</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1006</td>
<td>Cannot view logs of latest build</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1002</td>
<td>Perforce commands need better logging</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-994</td>
<td>Internal server error when trying to view TestData history</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-989</td>
<td>Duplicate JARs in classpath</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-988</td>
<td>Error when entered License Key</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-987</td>
<td>FreeMarker template error in plan summary</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-983</td>
<td>Freemarker template error when viewing 'latest' builds</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-962</td>
<td>Provide ability to point to a CVS tag instead of HEAD/Branch</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-960</td>
<td>Builds tend to disable themselves far too frequently</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-852</td>
<td>Internal error after &quot;Specify Source Repository&quot;</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-806</td>
<td>Support for SSH private key authentication (possible using jssh)</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-791</td>
<td>svn+ssh support for private key auth</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-691</td>
<td>CVS Client should use the CVS_RSH environment variable if available</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1031</td>
<td>Clicking on latest build from home screen</td>
<td></td>
<td>Resolved</td>
</tr>
</tbody>
</table>
BAM-980

Clicking on Tests tab in Build Result Summary renders empty screen

Resolved
Bamboo 1.0.2 Upgrade Guide

Upgrading from Bamboo 1.0.1 to 1.0.2

Please follow the Bamboo Upgrade Guide

Upgrading from Bamboo 1.0.1 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Atlassian is proud to announce the release of Bamboo 1.0.3! Bamboo 1.0.3 is mainly a bug fix release with over 10 issues resolved.

In this release, the focus has been on improving SVN integration (detection of SVN Externals) and CVS integration (detection of ampersand modules).

Updates and Issues fixed

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-1005</td>
<td>Setup fails</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1017</td>
<td>Never Can Retrieve Changelogs</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1063</td>
<td>Clover doesn't use the checkbox to determine if it should run.</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1008</td>
<td>Breaks of display in &quot;All plans&quot; Dashboard when svn comments have html inside.</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1000</td>
<td>Standalone Bamboo cannot start on Solaris</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-993</td>
<td>To provide easier configuration between Crowd and Bamboo</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-986</td>
<td>Emails should be more intelligent</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-976</td>
<td>IM message recipients input accumulates square brackets on form load</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-964</td>
<td>&quot;FreeMarker template error!&quot; on &quot;Plan Summary&quot; page in JDK 1.4</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-884</td>
<td>Redirection error when browsing to Bamboo pages</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-882</td>
<td>Add an option to use SVN Externals</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-862</td>
<td>Java error on startup</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-844</td>
<td>Commit changes do not trigger builds due to the use of SVN:externals</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-710</td>
<td>Internal server error: org.springframework.dao.DataIntegrityViolationException when deleting projects</td>
<td></td>
<td>Resolved</td>
</tr>
</tbody>
</table>
Bamboo 1.0.3 Upgrade Guide

This page last changed on Mar 27, 2007 by edwin@atlassian.com.

Upgrading from Bamboo 1.0.2 to 1.0.3

In this version, an upgrade task has been added to upgrade your CVS commit files data to a correct path (which includes module name). This may take a while to run, and it is strongly recommended that you back up your xml-data directory before proceeding. For fuller instructions please follow the Bamboo Upgrade Guide.

Upgrading from Bamboo 1.0.1 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.0.4 Release Notes

This page last changed on Dec 03, 2007 by rosie@atlassian.com.

✔ Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.

Atlassian is proud to announce the release of Bamboo 1.0.4! Bamboo 1.0.4 is mainly a bug fix release with over 10 issues resolved.

In this release, the focus has been on resolving connectivity issues with Subversion and Perforce

Perforce Improvements

There have been a few changes in Bamboo's Perforce integration

- Bamboo will now cache the client root rather than polling the repository continuously to obtain it. This reduces the load on the Perforce server considerably. However, if you change the root in the client definition on Perforce, Bamboo will require a restart to pick up the change.
- Bamboo now uses changelist numbers to detect source code changes rather than a timestamp. This will avoid all sorts of problems that occur when the Bamboo server clock and Perforce server clock are out of sync.
- Bamboo now picks up multi line change descriptions from Perforce.
- Bamboo can now generate web urls for perforce files when using Fisheye.

Updates and Issues fixed

<table>
<thead>
<tr>
<th>Key</th>
<th>Atlassian JIRA (14 issues)</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-1056</td>
<td></td>
<td>Failed to get the build source code: svn: report aborted</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-974</td>
<td></td>
<td>Bamboo penetrated perforce server with repeated requests on plan creation</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-750</td>
<td></td>
<td>Perforce changes are not displayed when a manual build is executed</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1113</td>
<td></td>
<td>Perforce modifications not causing build</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1096</td>
<td></td>
<td>Change the way bamboo detects changes in perforce</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1085</td>
<td></td>
<td>Subversion code refresh failing to pick up new revisions</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1078</td>
<td></td>
<td>BuildChangeDetector continuously polling Perforce repository</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1028</td>
<td></td>
<td>Bamboo throws exception when it polls Subversion repository</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-979</td>
<td></td>
<td>Different time zone on Perforce server does not work</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-890</td>
<td></td>
<td>SVN triggered update failing.</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-747</td>
<td></td>
<td>Perforce repository polling build plan not building</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-871</td>
<td></td>
<td>Manual builds still poll the perforce server</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-829</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
BAM-823

Only first line of change description is displayed for Perforce changes
Web Repository URL is not persisted for Perforce repositories

Resolved
Bamboo 1.0.4 Upgrade Guide

This page last changed on Apr 15, 2007 by rosie@atlassian.com.

Upgrading from Bamboo 1.0.3 to 1.0.4

In this version, an upgrade task has been added to update Perforce plans to use the change list number rather than the timestamp when detecting changes. Please ensure that you have connectivity to the Perforce server before you upgrade.

If Bamboo encounters any errors during the upgrade task it will set the Perforce plan's last change list number to 0. This means that the next time you build that plan there may be some unusual results (e.g. picking up every single change list). Once this build is complete normal behaviour will resume.

It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

Upgrading from Bamboo 1.0.2 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.

Atlassian is proud to announce the release of Bamboo 1.0.5! Bamboo 1.0.5 is mainly a bug fix release related to subversion connectivity issues.

Updates and Issues fixed

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-1139</td>
<td>Locked externals in SVN causes infinite building loop</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1137</td>
<td>Authentication always fails for subversion repository</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1108</td>
<td>Removing last build queue blocks use</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1100</td>
<td>Cannot log into Bamboo</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1060</td>
<td>Bamboo source update problem: &quot;Failed to get the build source code&quot;</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1050</td>
<td>Null pointer when relogging in after session has died</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-911</td>
<td>Cannot authenticate with Svn repository</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1125</td>
<td>Project Creation Fails with Self Signed SSL Certificate for SVN</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1118</td>
<td>FishEye link from Perforce project causes exception</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1022</td>
<td>Login link on comment page broken</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-891</td>
<td>Error when logging in to open source project</td>
<td></td>
<td>Resolved</td>
</tr>
</tbody>
</table>
Upgrading from Bamboo 1.0.4 to 1.0.5

It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

Upgrading from Bamboo 1.0.4 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.0-Beta Release Notes

This page last changed on Dec 03, 2007 by rosie@atlassian.com.

✅ Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.

The Atlassian Bamboo team is proud to announce the release of Bamboo 1.0 beta. This release includes over 40 bug fixes and improvements.

⚠️ Upgrading? Please see the Bamboo 1.0-Beta Upgrade Guide.

New in Release 1.0 - Beta

Anonymous access and sign on control.

In this release, you can now control whether your Bamboo is a public or private instance via the anonymous access and sign on options. Anonymous access allows users not signed in to view read only sections of Bamboo. Sign on allows users to create their own account for login. Disable these options to fully protect your Bamboo instance.

Auto favourite feature

Bamboo gets smarter with an auto-favourite marking feature. It'll mark those builds you commit against as your favourites.

Longest time to fix tests

Get a view of which tests in your builds are taking the longest the fix.

Other updates and bug fixes

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-697</td>
<td>CVS connection fails if password has @ in it</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-652</td>
<td>Checkboxes don't work properly when removing dependant builds</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-644</td>
<td>Script builder fails for windows</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-616</td>
<td>Capture code changes for dependent and scheduled builds</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-566</td>
<td>The back button on create plan wizard clears previously selected values</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-162</td>
<td>Passwords in plaintext</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-714</td>
<td>Support for SVN File Protocol</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-693</td>
<td>If project only has one plan, the project summary should redirect to the Plan Summary Page</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-685</td>
<td>Commit comments lose line breaks</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-684</td>
<td>Can't add builder of type Ant</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-676</td>
<td>Unable to re-index due to locked file</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-674</td>
<td>Build fails to start</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-661</td>
<td>Security and Login Improvements</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-656</td>
<td>No way to 'complete' setting up a project as 'save' hidden by javascript</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-647</td>
<td>Split webapp WAR module into a JAR and a WAR module</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-637</td>
<td>Reports build table not sorted</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-635</td>
<td>Auto favourite functionality</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-632</td>
<td>Breadcrumbs should have build numbers</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-631</td>
<td>Reports on top ten longest time to fix for tests</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-630</td>
<td>Test summary page still using the old style</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-625</td>
<td>Redirect after a plan is created</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-624</td>
<td>IE caches ajax response for comments and labels</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-622</td>
<td>Last screen of create build broken on Safari</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-609</td>
<td>error displaying build queue admin page too many files open error</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-608</td>
<td>Invalid path to clover throws error</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-606</td>
<td>Adding a comment from the Summary page doesn't work in IE</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-595</td>
<td>Need option to disable signups</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-572</td>
<td>Testing for mail and IM servers should be more visible</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-570</td>
<td>Auto report grouping for Tests doesn't seem to work</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-563</td>
<td>Validation for report not being selected</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-543</td>
<td>Minor tweaks of the Admin pages</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-542</td>
<td>Plugin Points for Web Fragments</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-490</td>
<td>Ability to run a Bamboo in 'private mode'</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-91</td>
<td>Static files are not cached, increases size of downloads</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-668</td>
<td>JIRA tabs shows up regardless</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-659</td>
<td>Edit configuration needs formatting fixes</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-612</td>
<td>Label grammar</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-611</td>
<td>Allow two character plan keys</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-603</td>
<td>Accessing /api/index.action throws a freemarker error</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>Issue</td>
<td>Description</td>
<td>Resolution</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>BAM-602</td>
<td>Minor issues with the build status wizard section on the editBuildConfiguration.action page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-658</td>
<td>Project names not ordered in dropdown to create project</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-657</td>
<td>Typo and user interface improvement</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-579</td>
<td>Make top right options clearer</td>
<td>Resolved</td>
<td></td>
</tr>
</tbody>
</table>
Bamboo 1.0-Beta Upgrade Guide

This page last changed on Jan 17, 2007 by rosie@atlassian.com.

Upgrading from Bamboo 0.9 to 1.0-Beta

Please follow the Bamboo Upgrade Guide.

Upgrading from Bamboo 0.8 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.0 Upgrade Guide

This page last changed on Feb 20, 2007 by edwin@atlassian.com.

Upgrading from Bamboo 1.0-RC2 to 1.0

Please follow the Bamboo Upgrade Guide

⚠️ You will need to reindex your data after the upgrade is complete and Bamboo has started. To do this, go to the indexing page under the Administration section in Bamboo.

Upgrading from Bamboo 1.0-RC1 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.1 Release Notes

This page last changed on Dec 03, 2007 by rosie@atlassian.com.

✔ Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.

The Atlassian Bamboo team is proud to announce the release of Bamboo 1.1! This release contains a whole host of new features targetted to make your build plans even more powerful and flexible.

⚠ Want to see Bamboo 1.1 in action? Check out our live opensource instance.

Advanced Notifications

In this release, we have extended Bamboo notifications framework to provide more flexibility, allowing you to select the how, who and when of notifications.

Notification Rules

Rather than having static fields for emails/IM recipients, Bamboo now allows you to define your own notifications for your build plans as a set of rules, giving you greater granularity in controlling exactly which recipient gets notified and when.

<table>
<thead>
<tr>
<th>Notification Trigger</th>
<th>Notification Recipients</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed Builds And First Successful</td>
<td>Users: test, admin</td>
<td>Edit, Remove</td>
</tr>
<tr>
<td>Notify After X Failed Builds</td>
<td>Roles: Writhe</td>
<td>Edit, Remove</td>
</tr>
<tr>
<td>All Completed Builds</td>
<td>Roles: Committer</td>
<td>Edit, Remove</td>
</tr>
</tbody>
</table>

Notification Triggers

In release 1.1, we introduce notification triggers, defining exactly when you would like a notification to be sent by Bamboo. By default, you can select a notification to be sent on "all builds completion", "after X failed builds" or "failed builds and first successful build". Want more customised triggers? You can now write your own as a notification condition plugin.

Add Build Notification

Notification Preferences

Different users prefer to get notified in different ways. Bamboo now lets you control that, via the new user notification preferences.

<table>
<thead>
<tr>
<th>Set Your Notification Preference</th>
</tr>
</thead>
<tbody>
<tr>
<td>How would you like bamboo to send you notifications:</td>
</tr>
<tr>
<td>✅ Do not send notifications</td>
</tr>
<tr>
<td>✅ Send instant message</td>
</tr>
<tr>
<td>✅ Send text message</td>
</tr>
<tr>
<td>✅ Send email and instant message</td>
</tr>
</tbody>
</table>

Please ensure that you have set the appropriate address for the notification type.

Dynamic recipients

Only want to receive a notification when you have committed against the build? Want to opt-in to receive notifications on the build plan that you are keeping an eye on? Bamboo 1.1 introduces two new dynamic recipient roles: committers (those users who have committed to the plan triggering the particular build to
execute) and watchers (those users who have marked the build plan as their favourite), which allow you to do just that!

Roles:

- **Committer** - Users who have committed to the build
- **Watcher** - Users who have marked this build as favourite

Select the various roles you wish to receive notifications.

Build Metadata

Every build process is different, and each build will have its own information that you may want to keep track of and use on top of the information that Bamboo stores about your build. This is particularly the case if you run custom plugins in your build process.

Pass them to your build

One way to use your build metadata is to pass it along to your builder as a property or target. To do this, you simply specify your variables in your target (or goal) field in your builder configuration. During build execution, the variables will be substituted with the actual values from your build metadata.

Global Variables

Bamboo 1.1 also allows you the option to specify variables globally. When a build begins, the global variables will be populated to the build's metadata. This is a handy option for you to control many plans in one go.

Global Variables

You can use this page to view, add and delete global variables. Global variables are available on every build run in Bamboo.

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>bambooVersion</td>
<td>1.1</td>
<td>Delete</td>
</tr>
</tbody>
</table>

View your metadata

Use the "Metadata" tab to keep track of all of your build's metadata.

File Trigger Inclusions/Exclusions

In this release, we also introduce the file trigger inclusion/exclusion filter. Instead of listening and picking up all changes from a repository, you can now use regex patterns to define those files which you do (or don't) want to trigger builds.
More pluggability

In release 1.1, we have added more plugin points to make Bamboo even more extensible than before. On top of the notification condition plugin point, we have also added pre-build action plugins, as well as repository plugins.

- Repository Plugins Not using SVN, CVS, or Perforce? You can now write a plugin to integrate with your very own source control.
- Pre-build Plugins Similar to the post-build action plugin, the pre-build action plugin will allow you to perform any custom task you may wish. The only difference is, of course, that it occurs before the build execution begins.

Improved Maven 2 error log parsing

Bamboo now intelligently parses the Maven 2 error log for possible errors in the build errors log, giving you a better view of what really went wrong in your build summary.

Error Summary

The build generated some errors. See the full build log for details.

LDAP and external user management support

In release 1.1, we have improved our user management capability to support externally sourced users and groups, including LDAP, and Crowd

Performance of Dashboard

With this release, we have also made significant performance improvements to the dashboard, which should see its load times reduce dramatically.

Other updates and bug fixes

On top of these features, we have also made a whole host of bug fixes, with over x bugs fixed since release 1.0.5.

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-940</td>
<td>Make Repository pluggable</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1191</td>
<td>IM notifications should allow group to be un-selected</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1189</td>
<td>Add global variables to be used in plans</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1183</td>
<td>Tool tip in the dashboard for the plans</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>Ticket Number</td>
<td>Description</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>BAM-1163</td>
<td>Dashboard very slow</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1155</td>
<td>Error cancelling a user preferences dialog</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1134</td>
<td>&quot;operation not permitted&quot; clicking Completed Builds tab</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1115</td>
<td>Artifact URL fails to escape invalid characters</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1090</td>
<td>Allow pluggable pre-build actions</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1088</td>
<td>Hide User Info From Non-Admins</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1081</td>
<td>Triggering a build only on certain commits</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1080</td>
<td>Lock obtain timed out error</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1073</td>
<td>No way to pass proxy information down to Ant</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1059</td>
<td>Implementation of new style of Notifications</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1058</td>
<td>Error viewing Build: Expression failingSinceBuild.buildResultsSummary is undefined</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1053</td>
<td>ANT_HOME is used for ant</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1032</td>
<td>NullPointerException accessing RSS feed</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1029</td>
<td>LDAP integration</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1020</td>
<td>Inconsistent Test Results between builds</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1016</td>
<td>Allow include/exclude patterns for triggering build</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1011</td>
<td>bamboo.home is not a valid environment variable</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1010</td>
<td>Error summary should parse logs in Maven 2 builds</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-996</td>
<td>FreeMarker template error in Completed Build Results screen</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-952</td>
<td>Bamboo uses older m2 version even when maven 2.0.5 builder configured for the plan</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-950</td>
<td>Out of memory error while processing Clover 2.0a4 results</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-942</td>
<td>Upgrade jfreechart version due to concurrency bug</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-938</td>
<td>Editing LDAP users in Bamboo</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-934</td>
<td>OutOfMemory while checking a build result</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-856</td>
<td>Add a simple batch / shell script to run the standalone</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-847</td>
<td>Remote API for manual checkout and build, Important Files and Directories</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-819</td>
<td>'Important Files and Directories'</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>Issue</td>
<td>Description</td>
<td>Resolution</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------------------------------------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>BAM-787</td>
<td>Add screenshots (when Clover data is available)</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-767</td>
<td>We able to add custom project specific Build Telemetry data</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-729</td>
<td>Make terminolody consistent (&quot;build&quot; vs. &quot;plan&quot;)</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-641</td>
<td>Ability to ignore file patterns for updates</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-633</td>
<td>Add Recent Activity tab for the dashboard</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-574</td>
<td>The interface used to provide a view of artifacts has issues dealing with subdirectories</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-561</td>
<td>Configure dependencies in the opposite direction</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-517</td>
<td>Email Notifications</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-493</td>
<td>Better handling of large number of unit tests</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-146</td>
<td>Build time graph should use minutes in scale</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-57</td>
<td>Improve configuration of notifications</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1173</td>
<td>Add an option to use a non Ajax dashboard &quot;hideDashboard&quot;</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1124</td>
<td>Crowd 1.0.6 + Bamboo 1.0.4</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NoClassDefFoundError; com/atlassian/user/security/authentication/InvalidPasswordException</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-915</td>
<td>Time values in Report data table should be more readable</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-792</td>
<td>Pass in bamboo parameters such as project and plan key to builders</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-604</td>
<td>API access to the SCM repository's build identifier for a given build</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-681</td>
<td>Checkboxes for build queues</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-680</td>
<td>Picker for e-mail notifications</td>
<td>Resolved</td>
<td></td>
</tr>
</tbody>
</table>

If you want to check out a live Bamboo instance, take a look at our opensource instance.
Bamboo 1.1.1 Release Notes

This page last changed on Dec 03, 2007 by rosie@atlassian.com.

✔ Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.

Atlassian is proud to announce the release of Bamboo 1.1.1! Bamboo 1.1.1 is mainly a bug fix release.

Major fixes include:

- LDAP - Many problems with LDAP integration have been overcome
- IMPORT/EXPORT - Several import fixes were implemented
- CVS - CVS change detection has been improved

Updates and Issues fixed

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-1244</td>
<td>Error after I have upgraded ldap in Bamboo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1233</td>
<td>A failed build is shown as a successful build</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1226</td>
<td>Building a project within Bamboo with JDK 1.4 sometimes fails when it does build fine outside of Bamboo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1215</td>
<td>Import fails if the import file is too large</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1206</td>
<td>Unable to remove build plan dependency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1151</td>
<td>Create plan input lost on timeout</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1857</td>
<td>Document BAM-1224</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1234</td>
<td>Ant build log appears incorrect through Bamboo which causes build to be successful rather than failed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1227</td>
<td>Bamboo throws freemarker exception when LDAP accounts don’t have full names for users</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1224</td>
<td>Improve the REST api in Bamboo to allow getting details from Bamboo at the project level instead of the plan/build level.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1216</td>
<td>Logging in as an ldap user causes a db exception</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1200</td>
<td>Unable to change how Bamboo sends notifications to user</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1192</td>
<td>&quot;build.dependency.select.none&quot; shown when there are no dependencies</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1154</td>
<td>Changes Made While Build is In Queue Don’t Get Change Logs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-1120</td>
<td>Arguments field in bash script/script builders are mandatory</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>BAM-1117</strong></td>
<td>Ant builder checks last 5 lines for BUILD SUCCESSFUL</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td><strong>BAM-1066</strong></td>
<td>Bamboo sometimes doesn't know who did CVS commit</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td><strong>BAM-825</strong></td>
<td>OutOfMemoryException thrown when importing large zip-files</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td><strong>BAM-505</strong></td>
<td>Bamboo may pickup wrong java installation in sub-processes</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td><strong>BAM-1195</strong></td>
<td>Wording changes for notifications</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td><strong>BAM-1165</strong></td>
<td>Text on 'Completed Builds' page doesn't match plan build strategy</td>
<td>Resolved</td>
<td></td>
</tr>
</tbody>
</table>
Bamboo 1.1.1 Upgrade Guide

This page last changed on May 16, 2007 by bmccoy.

Upgrading from Bamboo 1.1 to 1.1.1

Please follow the Bamboo Upgrade Guide

Upgrading from Bamboo 1.0.5 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.1.2 Release Notes

This page last changed on Dec 03, 2007 by rosie@atlassian.com.

✔ Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide. Atlassian is proud to announce the release of Bamboo 1.1.2! Bamboo 1.1.2 is mainly a bug fix release.

Major fixes include:
• Export - Windows Export caused some problems, these are now fixed
• Subversion - We have ungraded to the latest SVNKit to incorporate many of their bug fixes
• Fisheye Integration - The Fisheye links for perforce have been fixed
• Character Encoding - Bamboo now lets you use all Unicode characters
• LDAP - More LDAP fixes!

Updates and Issues fixed

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-1289</td>
<td>Export not working in Bamboo 1.1.1</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1256</td>
<td>Upgrade SVNKit to the latest version</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1254</td>
<td>Upgrade bundled svnkit version to get bugfixes</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1296</td>
<td>PATH variable gets clobbered in JDK 1.4</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1279</td>
<td>User and group browser very slow in LDAP environment</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1262</td>
<td>Browse user and group pages occasionally throws exception (LDAP)</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1260</td>
<td>Cron triggers that specify multiples (entries with commas) get interpreted as arrays of Strings</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1238</td>
<td>Improve the CVS trigger doco</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1232</td>
<td>Update docs after packaging the scripts in the WAR version</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1231</td>
<td>Include the scripts for triggering CVS and SVN builds in the WAR version</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1229</td>
<td>Dependency builds incorrectly reported as initial or manual build in email notifications</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1094</td>
<td>start-bamboo wrapper uses false uname syntax</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1046</td>
<td>Valid cron expression not really valid</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-971</td>
<td>JIRA issue parser should look for numbers in project key as well. Encoding (of project name)</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-745</td>
<td></td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1265</td>
<td>When adding builders - adding a label with space</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>Issue</td>
<td>Description</td>
<td>Resolution</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>BAM-1138</td>
<td>File Version Number problems with Perforce</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1065</td>
<td>Bamboo reports twice as many unit test</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-518</td>
<td>Unable to copy build artifacts</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1235</td>
<td>Backslashes in usernames not displayed correctly on tooltip</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-973</td>
<td>Users with non standard characters in their names show up corrupted.</td>
<td>Resolved</td>
<td></td>
</tr>
</tbody>
</table>

in the beginning of fails to build
Bamboo 1.1.2 Upgrade Guide

This page last changed on May 31, 2007 by bmccoy.

Upgrading from Bamboo 1.1.1 to 1.1.2

Please follow the Bamboo Upgrade Guide

Upgrading from Bamboo 1.1 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.1 Upgrade Guide

This page last changed on May 07, 2007 by edwin@atlassian.com.

Upgrading from Bamboo 1.0.5 to 1.1

It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

Please note that the upgrade process may take a while to complete.

Upgrading from Bamboo 1.0.5 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.2 Release Notes

This page last changed on Feb 21, 2008 by asridhar.

✅ Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.

Atlassian Software Systems is proud to announce the release of Bamboo 1.2. This release contains:

- Permissions (global and plan-based)
- External database support
- Perforce triggering support
- Scheduled backups
- New Bundled NAnt plugin
- Lots of minor features and bug fixes

Bamboo 1.2 can be downloaded here, and is of course free to all customers who purchased their Bamboo licence or maintenance after July 9, 2006.

When upgrading, please refer to the Bamboo 1.2 Upgrade Guide.

⚠️ Want to see Bamboo 1.2 in action? Check out our live opensource instance.

Permissions (global and plan-based)

Different organisations, and different projects, have different security requirements. Some information can be made public, while sensitive information may need to be confined to a particular group of people.

Bamboo 1.2 gives you the ability to set security on individual build plans, as well as on your entire Bamboo system:

- **Plan permissions** allow your chosen users to perform a particular operation in relation to a particular build plan (e.g. view its build results).
- **Global permissions** allow your chosen users to perform a particular operation in relation to Bamboo as a whole.

External database support

Bamboo ships with a built-in HSQL database, which is well suited to evaluation purposes. When deploying Bamboo in production, however, you will probably prefer to connect Bamboo to an enterprise database of your choice.

Bamboo 1.2 now includes support for MySQL and Postgres. If you need to use a different database, generic instructions for connecting Bamboo to an unsupported database are also provided.
Perforce triggering support

We are pleased to announce that Bamboo builds can now be triggered by Perforce repositories (previously only Subversion and CVS repositories were supported).

Scheduled backups

You can now schedule your Bamboo data exports to occur automatically at a convenient time:

New Bundled NAnt plugin

Want to build your .Net projects on Bamboo? Now you can, with the NAnt plugin, which comes bundled by default with Bamboo 1.2.

On the topic of plugins, have you checked out Bamboo Extensions space, home to a whole host of cool Bamboo plugins?

Other updates and bug fixes

On top of these features, Bamboo 1.2 also includes a host of minor new features, improvements and bug fixes:
<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-1380</td>
<td>Exporting from an instance with large ZIP artifacts may fail</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1298</td>
<td>Bamboo doesn't URL encode &quot;&quot;</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1683</td>
<td>Fix for IndexOutOfBoundsException exceptions in the SVNDeltaReader</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1426</td>
<td>Ability to customize the installation name for a Bamboo instance</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1422</td>
<td>Option to Export/Backup without Artifacts</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1417</td>
<td>Ability to remove artifacts for a build result</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1415</td>
<td>Perforce Build Trigger Scripts</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1406</td>
<td>Perforce change logs not picked up due to update returning too many results.</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1399</td>
<td>Log Output Download Option</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1395</td>
<td>New Jabber command for getting more change logs information</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1384</td>
<td>build expiry should be available on a per project basis</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1374</td>
<td>Delete recovery mechanism in Bamboo</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1373</td>
<td>Bamboo throws nullpointer exceptions while deleting build</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1366</td>
<td>NullPointerException after clicking &quot;1. Plan Details&quot; tab</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1348</td>
<td>Bamboo shows &quot;null build&quot; in RSS header (for project builds)</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1346</td>
<td>Single quotes in logs are prefixed with a back slash \</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1345</td>
<td>Subscribing to a RSS feed from a plan page leads to a 404 page not found exception</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1340</td>
<td>Deadlock issue while view currently running build</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1337</td>
<td>REST API invalid user id error not handled correctly</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1336</td>
<td>REST API documentation not accurate</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1328</td>
<td>Look at Clover XML Even If Build Fails</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1322</td>
<td>Bamboo error when browsing authors</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1282</td>
<td>Bamboo goes through password reminder even if no mail server configured</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>Issue Key</td>
<td>Description</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------------------------------------------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>BAM-1255</td>
<td>Duplicate Email Notifications</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1253</td>
<td>Allow for expiry of just the artifacts</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1250</td>
<td>Null Pointer error in GetReturnURL</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1152</td>
<td>Bamboo should accept repository triggers for Perforce builds</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1130</td>
<td>access control to specific plan (based on groups)</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1077</td>
<td>Change terminology/function of Perforce Source Repository page</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1042</td>
<td>Modify bamboo.sh to start the JVM in 'server' mode</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1030</td>
<td>Ability to download particular build log file</td>
<td>Closed</td>
<td></td>
</tr>
<tr>
<td>BAM-957</td>
<td>Ability to configure a database</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-906</td>
<td>Automate backups with a task scheduler</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-859</td>
<td>Downloadable logs</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-840</td>
<td>JUnit XML Improperly Parsed</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-752</td>
<td>Perforce Configuration should allow passwords</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-701</td>
<td>Native support for NAnt builder</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-525</td>
<td>Restrict access to statistics</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-249</td>
<td>Configurable Permissions for Dashboard</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-248</td>
<td>Permissions for User and Groups</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-58</td>
<td>Bootstrapping process</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1382</td>
<td>Build Labeller plugin fails to validate regex pattern</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1309</td>
<td>Edit build notification screen has no title</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1302</td>
<td>Perforce depot access will fail if the depot contains no workspace mapping</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1259</td>
<td>deleting a build plan caused NPE</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-1068</td>
<td>New mime types for bamboo artifact downloads</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-798</td>
<td>Sub Menu Tabs move from side to side in IE customise site title</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-354</td>
<td>General Configuration, Build Expiry --&gt; change to view and edit screens</td>
<td>Resolved</td>
<td></td>
</tr>
</tbody>
</table>

^Top
Bamboo 1.2.1 Release Notes

This page last changed on Dec 03, 2007 by rosie@atlassian.com.

✅ Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.

Atlassian is proud to announce the release of Bamboo 1.2.1! Bamboo 1.2.1 is mainly a bug fix release.

⚠️ It is strongly recommended that you upgrade to Bamboo 1.2.1! It contains a fix to a critical security exploit in the system.

Major fixes include:

- Security exploit in Webwork 2.2.
- JDK 1.4 support
- Import & Export of build plan dependencies
- Upgrading from 1.0.x to 1.2.

Updates and Issues fixed

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-1477</td>
<td>Webwork 2.2.2 security exploit</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1475</td>
<td>Plan dependencies not exported or imported (not sure which)</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1471</td>
<td>Bamboo 1.2 JAVA (Unsupported major.minor version 49.0) error.</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1460</td>
<td>Bamboo POM’s need to be updated for the next point release</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1455</td>
<td>Slow performance on test results page.</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1453</td>
<td>Importing from a different server from builds pre: 406 causes inconsistent build directories</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1452</td>
<td>Upgrades from 1.0.x directly to 1.2 fails</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1359</td>
<td>Integrate enhanced regex labeller plugin</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-978</td>
<td>Bamboo Plugin Dev Kit pom.xml has a snapshot dependency.</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1470</td>
<td>The Exclude email header &quot;Precedence: bulk&quot; option is not exported and imported.</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1437</td>
<td>Line breaks are lost in user comments</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-977</td>
<td>Plugin Dev Kit has Mac OSX specific files in it.</td>
<td></td>
<td>Resolved</td>
</tr>
</tbody>
</table>
Bamboo 1.2.1 Upgrade Guide

This page last changed on Jul 18, 2007 by edwin@atlassian.com.

Upgrading from Bamboo 1.2 to 1.2.1

Please follow the Bamboo Upgrade Guide

Upgrading from Bamboo 1.1 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.2.2 Release Notes

This page last changed on Dec 03, 2007 by rosie@atlassian.com.

Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.

Atlassian is proud to announce the release of Bamboo 1.2.2!

Major features include:

- **Bulk editing of plan permissions.**
- Administrators can now change users' passwords.
- Improved caching on the dashboard, for better performance.

Major fixes include:

- Import and export when integrated with LDAP or Crowd.
- More import and export fixes.
- Users can now IM with Crowd integrated.

Updates and Issues fixed

<table>
<thead>
<tr>
<th>Atlassian JIRA (23 issues)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>BAM-1494</td>
</tr>
<tr>
<td>BAM-144</td>
</tr>
<tr>
<td>BAM-1086</td>
</tr>
<tr>
<td>BAM-1482</td>
</tr>
<tr>
<td>BAM-1547</td>
</tr>
</tbody>
</table>

Document generated by Confluence on Apr 14, 2008 01:44  Page 279
create plans

BAM-1252
Users Can't Export Bamboo with LDAP turned on

Brydie mayank McCoy
[Atlassian]

FIXED Resolved May 17, Aug 13, 2007

BAM-1284
Trying to edit the build properties to include the bamboo_buildVersion parameter

Brydie John McCoy Reynolds
[Atlassian]

FIXED Resolved May 24, Aug 08, 2007

BAM-1562
"a" duplicated in the Indexing admin page

Brydie James McCoy Odeen
[Atlassian]

FIXED Resolved Aug 06, Aug 07, 2007

BAM-1566
Bamboo does not export users notification preference

Brydie McCoy McCoy
[Atlassian][Atlassian]

FIXED Resolved Aug 07, Aug 07, 2007

BAM-1550
Exports (Scheduled backups) do not work when Crowd is integrated with Bamboo.

Brydie Ajay McCoy Sridhar
[Atlassian][Atlassian]

FIXED Resolved Aug 02, Aug 07, 2007

BAM-1188
Can't send IM when Bamboo is integrated with Crowd

Mark Ajay Chaimungkalanont
[Atlassian][Atlassian]

FIXED Resolved May 06, Aug 06, 2007

BAM-1522
Bamboo leaves build queue's in a disabled state, if exports (scheduled backups) fail

Adrian Ajay Hempel Sridhar
[Atlassian][Atlassian]

FIXED Resolved Jul 31, Aug 01, 2007

BAM-1530
Poor documentation for link on the

Brydie McCoy Sridhar
[Atlassian][Atlassian]

FIXED Resolved Jul 31, Aug 01, 2007
<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
<th>Resolution</th>
<th>Assigned To</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-1436</td>
<td>It's possible to add duplicate labels for a build result you then can't remove them</td>
<td>FIXED</td>
<td>Adrian Mark Hempel Chaimungkalanont [Atlassian]</td>
<td>Jul 05, 2007, Jul 27, 2007</td>
</tr>
</tbody>
</table>
Bamboo 1.2.2 Upgrade Guide

This page last changed on Aug 16, 2007 by rosie@atlassian.com.

Upgrading from Bamboo 1.2 (or 1.2.1) to 1.2.2

Please follow the Bamboo Upgrade Guide, plus:

⚠️ Bamboo on Tomcat 5
If you are running Bamboo on Tomcat 5, please follow the instructions on this page.

Upgrading from Bamboo 1.1 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.2.3 Release Notes

This page last changed on Dec 12, 2007 by rosie@atlassian.com.

✅ Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.

12 September 2007

Atlassian is proud to announce the release of Bamboo 1.2.3. This point release includes more than 20 minor fixes and improvements. Most notably, for greater flexibility when configuring a build plan, **variables** can now be used in a number of different places.

Bamboo 1.2.3 can be downloaded [here](#). When upgrading, please refer to the Bamboo 1.2.3 Upgrade Guide.

**Updates and issues fixed**

<table>
<thead>
<tr>
<th>Key</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Pr</th>
<th>Status</th>
<th>Res</th>
<th>Created</th>
<th>Updated</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-1644</td>
<td>Implement rules to determine whether an artifact is to be downloaded or viewed in the browser</td>
<td>Adrian Hempel [Atlassian]</td>
<td></td>
<td>FIXED</td>
<td>Resolved</td>
<td>Aug 28, 2007</td>
<td>Oct 21, 2007</td>
<td></td>
</tr>
<tr>
<td>BAM-1654</td>
<td>Upgrade Serpan dependency from 0.7.17 to 0.7.23</td>
<td>Edwin Wong [Atlassian]</td>
<td></td>
<td>FIXED</td>
<td>Resolved</td>
<td>Aug 29, 2007</td>
<td>Sep 10, 2007</td>
<td></td>
</tr>
<tr>
<td>BAM-1309</td>
<td>Ability to specify (global) System Variables to be global in Bamboo</td>
<td>Brydie McCoy [Atlassian]</td>
<td></td>
<td>FIXED</td>
<td>Resolved</td>
<td>May 31, 2007</td>
<td>Sep 04, 2007</td>
<td></td>
</tr>
</tbody>
</table>

[Atlassian JIRA](#) (24 issues)
**BAM-1442**

User comments should have the standard link formatting applied to it.


**BAM-1443**

Error Log needs more context e.g. build no, timestamp.

Resolved Sep 10, 2006 - Aug 15, 2007

**BAM-1713**

Bamboo.developer.atlassian.com forgot my name and email.

Resolved Sep 20, 2007 - Sep 20, 2007

**BAM-1660**

Duplicate Path variable(s) added by Bamboo when executing Build on windows.

Resolved Aug 31, 2007 - Sep 11, 2007

**BAM-1661**

User without Global Admin Permission can not create a new plan.

Resolved Aug 31, 2007 - Sep 10, 2007

**BAM-1680**

SSO with Crowd causes NoSuchMethodError.

Resolved Sep 07, 2007 - Sep 10, 2007

**BAM-1634**

Bamboo will not export under JDK 1.4 using JBoss 4.0.5.

Resolved Aug 26, 2007 - Sep 03, 2007

**BAM-1668**

Bamboo can't parse test results, where method name is "Test".

Resolved Sep 03, 2007 - Sep 03, 2007
BAM-1489 Typo on "Post Actions" tab

BAM-402 Can't display artifacts with spaces in the filename

BAM-163 Bamboo fails to export when no plans are configured

BAM-162 AccessLogFilter is configured but does not show free in bamboo.log

BAM-1610 When configuring Perforce scm, with Triggered build, The "Trigger IP address" field does not get saved.

BAM-1563 Incorrect format for Build Time when configuring a plan

BAM-159 Bamboo logs show a harmless LicenseException during Setup Wizard

BAM-1559 Build results list sorted by build start date
time, displays completed time
Bamboo 1.2.3 Upgrade Guide

Upgrading from Bamboo 1.2.x to 1.2.3

Please follow the Bamboo Upgrade Guide, plus:

⚠️ Bamboo on Tomcat 5
If you are running Bamboo on Tomcat 5, please follow the instructions on this page.

Upgrading from Bamboo 1.1.x or earlier

In addition to the above, please read the Bamboo 1.2 Upgrade Guide and the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.2.4 Release Notes

Atlassian is proud to announce the release of Bamboo 1.2.4. This point release includes more than 20 minor fixes and improvements. Bamboo 1.2.4 can be downloaded here. When upgrading, please refer to the Bamboo 1.2.4 Upgrade Guide.

Updates and issues fixed

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Pr</th>
<th>Status</th>
<th>Created</th>
<th>Updated</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-151</td>
<td>Have a nicer Error page, when trying to see a non existing build report</td>
<td>Brydie McCoy</td>
<td>Samuel Le Atlassian[Berrigaud [Atlassian]</td>
<td></td>
<td>FIXED</td>
<td>Jul 25, 2007</td>
<td>Sep 17, 2007</td>
<td></td>
</tr>
<tr>
<td>BAM-204</td>
<td>Bamboo displays Freemarker exception</td>
<td>Unassigned</td>
<td>Adrian Hempel Atlassian</td>
<td></td>
<td>OBSOLETE</td>
<td>Dec 23, 2007</td>
<td>Jan 09, 2008</td>
<td></td>
</tr>
</tbody>
</table>
when displaying test result for which no duration was reported

**BAM-1799** Build expiry is set to expire from last built date - as opposed to current working date.

Brydie McCoy, Ajay Sridhar

Resolved

FIXED

Oct 08, 2007 Jan 09, 2008

**BAM-1742** Freemarker error on BDAC

Unassigned, Jonathan Nolen

Resolved

FIXED


**BAM-1812** Clarify exactly what needs to be backed up when upgrading Bamboo

Rosie Jameson, Max Bowsher

Resolved

FIXED


**BAM-1744** Indexing error when deleting builds.

Brydie McCoy, Ajay Sridhar

Resolved

FIXED


**BAM-1808** When you disable the build expiry it doesn't actually disable

Brydie McCoy, Brydie McCoy

Resolved

FIXED


**BAM-1666** Build expiry does not occur if global configuration is turned off

Brydie McCoy, Mark Chaimungkalanont

Resolved

FIXED

Sep 02, 2007 Oct 10, 2007

**BAM-1794** Bamboo gives the impression that Build Expiry will run at the end of the build

Unassigned, Brydie McCoy

Resolved

FIXED

BAM-174 Viewing plan without "Anonymous User" view rights results in "Could not find the null'th build for build sf - Default"

Mark Chaimungkalanont [Atlassian]"
Error Log does not respect permission rights

BAM-1693 Build artifacts won't open correctly in Firefox
Brydie Arjan McCoy Schaal [Atlassian]

BAM-1687 Users without Admin rights can't change number of shown builds in ViewBuildResults page
Edwin Arjan Wong Schaal [Atlassian]

BAM-1692 When editing notification rule, warning message about IM server missing appears even when one configured
Brydie Edwin McCoy Wong [Atlassian][Atlassian]

BAM-1698 Duplicated EhCache jar in Bamboo causes caching issue with Crowd
Edwin Edwin Wong Wong [Atlassian][Atlassian]

Fixed
Resolved
Sep 12, 2007
Sep 19, 2007

Fixed
Resolved
Sep 10, 2007
Sep 17, 2007

Fixed
Resolved
Sep 12, 2007
Sep 16, 2007

Fixed
Resolved
Sep 13, 2007
Sep 16, 2007
Bamboo 1.2.4 Upgrade Guide

Upgrading from Bamboo 1.2.x to 1.2.4

Please follow the Bamboo Upgrade Guide, plus:

⚠️ Bamboo on Tomcat 5
If you are running Bamboo on Tomcat 5, please follow the instructions on this page.

Upgrading from Bamboo 1.1.x or earlier

In addition to the above, please read the Bamboo 1.2 Upgrade Guide and the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.
Bamboo 1.2 Plugin Interface Changes

This page last changed on Jul 16, 2007 by mark@atlassian.com.

Below are details of plugin interface changes with Bamboo 1.2

Notification Condition

The method getTextEmail has changed from

```java
public void getTextEmail(Event event, Email email);
```

to

```java
public Email getTextEmail(Event event, Email email);
```

It now requires you to return the email object with the content populated (body, subject mimeType etc)
Bamboo 1.2 Upgrade Guide

This page last changed on Jan 15, 2008 by asridhar.

Upgrading from Bamboo 1.1.2 to 1.2

It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

⚠️ If you are using plugins, please make sure that your plugins are compile against 1.2 before upgrading.

⚠️ Crowd on Bamboo 1.2
If you are using Bamboo with Crowd, please make sure that you upgrade to Crowd 1.1.2 before upgrading Bamboo.

Bamboo on Tomcat 5
If you are running Bamboo on Tomcat 5, please follow the instructions on this page.

Please note that the upgrade process may take a while to complete.

Upgrading from Bamboo 1.1.1 and earlier

In addition to the above, please read the Upgrade Guide for every version you are skipping during the upgrade. The complete list of Upgrade Guides is available here.

⚠️ If you're upgrading from Bamboo 1.0.x to Bamboo 1.2, please upgrade to 1.1.2 first. There is an issue with the upgrade process from the 1.0.x series that we're currently looking into.
Bamboo 2.0 Beta Release Notes

This page last changed on Apr 02, 2008 by alui.

This page contains a live summary of all Bamboo release notes for the Bamboo 2.0 Beta. Click through to each of the individual release notes to view the complete list of issues and fixes associated with each release.

Please read the release notes up to and including the version that you are interested in. You may also wish to read the Bamboo 2.0 Beta upgrade guides.

Bamboo 2.0 Beta Release Notes

- Bamboo 2.0 Beta 1
- Bamboo 2.0 Beta 2
- Bamboo 2.0 Beta 3
- Bamboo 2.0 Beta 4
- Bamboo 2.0 Beta 5
- Bamboo 2.0 Beta 6
- Bamboo 2.0 Beta 8
- Bamboo 2.0 Beta 9

Bamboo 2.0 Beta 1

Bamboo 2.0 will be launched in early 2008 and will introduce a number of new features, including the ability to run distributed builds, flexible build agent management and memory usage improvements.

Because Bamboo 2.0 will introduce major architectural changes, the Bamboo 2.0 Beta program is being provided to enable you to preview the upcoming features and perform preliminary testing.

Please note that this release is a beta and should not be used on production systems.

<table>
<thead>
<tr>
<th>Upgrading to Bamboo 2.0 Beta</th>
</tr>
</thead>
</table>

Bamboo 2.0 Beta can be downloaded from the Bamboo Download Centre. Before upgrading, please refer to the Bamboo 2.0 Beta Upgrade Guide. You must upgrade to Bamboo 1.2.x before upgrading to 2.0.

What's New in Bamboo 2.0?

- **Distributed Builds** — This release introduces the ability to run distributed builds. You will find this particularly useful if you need to run your builds in different geographic locations, or on different platforms. Simply install the new Bamboo Agent on your additional build servers, and your main Bamboo 2.0 server will be able to manage them. We have also provided a number of plugin points in case you need to control your distributed builds programmatically.

- **Flexible Build Agent Management** — Bamboo 2.0 also gives you much more flexibility in managing your builds. Build queues are no longer required, with the introduction of requirements and capabilities. You can direct builds to be run on a particular agent, by specifying build plan requirements to match the builder, JDK and custom capabilities that you have set up for the agent. Read more about agents and capabilities here.

- **Memory Usage Improvements** — The underlying engine for Bamboo has been revamped to decrease memory usage. You will notice a distinct improvement in the performance of your builds, especially if you have very large logs.

Read more about Bamboo 2.0 Beta 1...
Bamboo 2.0 Beta 2

This point release includes more than 10 minor fixes and improvements. Bamboo 2.0 Beta 2 can be downloaded [here](#). Before upgrading, please read the Bamboo 2.0 Beta 2 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

Read more about Bamboo 2.0 Beta 2...

Bamboo 2.0 Beta 3

Bamboo 2.0 Beta 3 introduces a number of new features, including Perforce support and the ability to connect to Oracle and MS SQL as external databases. A number of significant fixes have been included as well, improving the overall quality and experience of the beta.

<table>
<thead>
<tr>
<th>Upgrading to Bamboo 2.0 Beta</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bamboo 2.0 Beta can be downloaded from the Bamboo Download Centre. Before upgrading, please read the Bamboo 2.0 Beta 3 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta 2, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.</td>
</tr>
</tbody>
</table>

What's New in Bamboo 2.0 Beta 3?

- **Perforce Support** — Bamboo brings back out of the box support for Perforce source repositories. A number of changes have been made to Perforce configuration to ensure that it works correctly with distributed builds. Read more about configuring Perforce.

- **Oracle and MS SQL Server Support** — By popular request, Bamboo's supported databases now include Oracle and MS SQL Server. Read more about connecting Bamboo to Oracle and MS SQL Server.

- **Major Bug Fixes** — Bamboo now works with PostgreSQL and MySQL correctly. Please see the relevant Postgresql and MySQL JIRA issues for details about the fixes.

Read more about Bamboo 2.0 Beta 3...

Bamboo 2.0 Beta 4

This point release includes more than 15 minor fixes and improvements. Bamboo 2.0 Beta 4 can be downloaded [here](#). Before upgrading, please read the Bamboo 2.0 Beta 4 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

Major Bug Fixes

- Previously, Bamboo determined which agents could build a plan when the plan was queued, but would then incorrectly ignore any subsequent changes (including disabling the agent). Bamboo will now correctly update where plans can be built, even if changes are made after the plan has been queued.
For the Developers

• The new 'RepositoryEventAware' interface allows you to implement custom actions before and/or after retrieving source code from your repository. Read more about extending the standard repository functionality.

Read more about Bamboo 2.0 Beta 4...

Bamboo 2.0 Beta 5

This point release includes more than 10 minor fixes and improvements. Bamboo 2.0 Beta 5 can be downloaded here.

Before upgrading, please read the Bamboo 2.0 Beta 5 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

JDK and Builders page now included in Bamboo 2.0 Beta

• The JDK and Builder pages which were temporarily removed from Bamboo for the 2.0 Beta have now been restored.

Read more about Bamboo 2.0 Beta 5...

Bamboo 2.0 Beta 6

This point release includes more than 5 minor fixes and improvements. Bamboo 2.0 Beta 6 can be downloaded here.

Before upgrading, please read the Bamboo 2.0 Beta 6 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

Secured Remote Agents are now Supported

• Remote agents can now be secured with the appropriate SSL configuration. Read more about Securing your Remote Agents.

We strongly recommend that you do not enable remote agent installation on any Bamboo instance accessible from a public or untrusted network without securing your remote agents. If you choose to enable your remote agents without securing them, please read this security advisory to understand the security implications.

Changes to Client Workspace Configuration for Perforce

• If you use Perforce, you can now choose whether you want Bamboo to manage your client workspace (i.e. set the client root) or manage it yourself. Read more about Perforce configuration.

'Shared Local Capabilities' are now called 'Local Server Capabilities'

• All references to 'Shared Local Capabilities' (or equivalent terminology) have been changed to 'Local Server Capabilities' in the Bamboo user interface.

Read more about Bamboo 2.0 Beta 6...

Bamboo 2.0 Beta 8

This point release includes more than 20 minor fixes and improvements. Bamboo 2.0 Beta 8 can be downloaded here.

Before upgrading, please read the Bamboo 2.0 Beta 8 Upgrade Guide (Please note that there is no 2.0 Beta 7). If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.
Quiet Period Functionality Supported for Subversion & Perforce

By popular request, Quiet Period parameters can now be specified for Subversion and Perforce when configuring a source repository for a build plan. You can choose to set how long Bamboo should wait after a commit before triggering a build, and the number of times it retries before initiating a build. Read more about configuring Subversion and Perforce source repositories.

'Force Clean Builds' Supported

Also by popular request, you can now force Bamboo to run 'Clean Builds' in a build plan. That is, the source directory is removed and then checked out from the repository prior to each build. Read more about this function in Specifying a Plan's Source Repository.

Bamboo 2.0 Beta 9

This point release includes more than 10 minor fixes and improvements. Bamboo 2.0 Beta 9 can be downloaded here.

Before upgrading, please read the Bamboo 2.0 Beta 9 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

Edit and Rename Capabilities

You can now edit capabilities in Bamboo, as listed below:

- If you are editing a Builder capability, you can modify the 'Path' of the builder.
- If you are editing a JDK capability, you can modify the 'Java Home' of the JDK.
- If you are editing a Custom capability, you can modify the 'Value' of the capability.
- If you are editing a Perforce capability, you can modify the 'Perforce Executable' path.

You can also rename a capability. This is reflected in any plans that the capability is specified as a requirement for.

Read more about editing capabilities and renaming capabilities.

View Agents and Plans related to a Capability

A 'View Capability' screen is now available in Bamboo 2.0. This screen lists the agents that have/inherit a particular capability, as well as which plans have the capability specified as a requirement.

Read more about viewing capabilities.

Read more about Bamboo 2.0 Beta 9...
Atlassian Software Systems presents Bamboo 2.0 Beta

Bamboo 2.0 will be launched in early 2008 and will introduce a number of new features, including the ability to run distributed builds, flexible build agent management and memory usage improvements.

Because Bamboo 2.0 will introduce major architectural changes, the Bamboo 2.0 Beta program is being provided to enable you to preview the upcoming features and perform preliminary testing.

Please note that this release is a beta and should not be used on production systems.

### Upgrading to Bamboo 2.0 Beta

Bamboo 2.0 Beta can be downloaded from the Bamboo Download Centre. Before upgrading, please refer to the Bamboo 2.0 Beta Upgrade Guide. You must upgrade to Bamboo 1.2.x before upgrading to 2.0.

### What's New in Bamboo 2.0?

1. **Distributed Builds** — This release introduces the ability to run distributed builds. You will find this particularly useful if you need to run your builds in different geographic locations, or on different platforms. Simply install the new Bamboo Agent on your additional build servers, and your main Bamboo 2.0 server will be able to manage them. We have also provided a number of plugin points in case you need to control your distributed builds programmatically.

2. **Flexible Build Agent Management** — Bamboo 2.0 also gives you much more flexibility in managing your builds. Build queues are no longer required, with the introduction of requirements and capabilities. You can direct builds to be run on a particular agent, by specifying build plan requirements to match the builder, JDK and custom capabilities that you have set up for the agent. Read more about agents and capabilities here.

3. **Memory Usage Improvements** — The underlying engine for Bamboo has been revamped to decrease memory usage. You will notice a distinct improvement in the performance of your builds, especially if you have very large logs.

### Security Considerations

⚠️ Important security information for the Bamboo 2.0 Beta has been published. Please refer to the security advisory for details.

### Known Issues

The following issues are applicable at the time of the Bamboo 2.0 Beta 1 release. Please refer to the aggregated Bamboo 2.0 Beta Release Notes to review the complete list of issues and fixes for each beta release.

- Perforce is not supported in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release).
- Bamboo currently does not work with MySQL. See BAM-2260 for further details.
- Plans currently cannot be edited, if Bamboo is integrated with a PostgreSQL database. See BAM-2208 for further details.
- Secured remote agents are not supported in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release). Please see the Bamboo security advisory for further details.
• JDK and Builders pages have not been included in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release).

• Please note that builds are currently allocated to agents during queuing time, not execution time. This may occasionally mean that a build is executed by an agent that you have disabled, which is slightly different from the functionality described in 2.5 Monitoring Agent Status. For example:
  1. Agent A is enabled and is currently executing a build for Plan X.
  2. Plan Y submits a build to the queue, and the queue assigns the build to Agent A.
  3. You disable Agent A.
  4. Agent A completes Plan X’s build.
  5. Although agent A is disabled, it will still run Plan Y’s build, because it was determined as executable when plan Y was queued.

Updates and Fixes in this Release

Please help us with the final 2.0 release by reporting any bugs and issues you find, in the Bamboo project at jira.atlassian.com.

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-2462</td>
<td>Add link to the Status Summary Screen</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2460</td>
<td>Caching error may cause wrong build result to be cached</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2448</td>
<td>File Added In Attic</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2447</td>
<td>CVS Plans Don't Work For Branches</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2435</td>
<td>When deleting a requirement within the error screen, creates null pointer</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2431</td>
<td>Unable to save buildresultsummary if a code commit doesn't have comment</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2430</td>
<td>Dashboard quirk while removing plan from favourites list on the My Bamboo tab of dashboard.</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2429</td>
<td>500 on /admin/configurePipeline!default.action</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2428</td>
<td>Allow configuration of SVNKit to spool / not spool</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2425</td>
<td>Errors in IE7 while viewing the Edit Build configuration page</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2422</td>
<td>Error while importing Bamboo buildresults from bdac due to missing authors from commits</td>
<td></td>
<td>Closed</td>
</tr>
<tr>
<td>BAM-2407</td>
<td>Remote Agent running on Sun JDK 1.6 exits after build is queued for it</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2405</td>
<td>Add extension builder interfaces</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2404</td>
<td>Clicking on the previous button of the Plan creation wizard doesn't</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>Ticket</td>
<td>Issue Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2401</td>
<td>Remote agent restart can leave builds in a building state while the agent reports it is idle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2391</td>
<td>SVN externals now prohibitively slow</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2390</td>
<td>Allow capabilities to be used in variable replacements</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2388</td>
<td>Javascript messagebox shown when ajax connection to bamboo fails is very annoying</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2383</td>
<td>NPE trying to access <a href="https://bamboo.developer.atlassian.com/browse/FISH-STABLE/test">https://bamboo.developer.atlassian.com/browse/FISH-STABLE/test</a></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2382</td>
<td>Perforce Web RepositoryUrls are not implemented in 2.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2381</td>
<td>Failing since build for a test reports the wrong result</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2372</td>
<td>Repository web URL still create links even though none is specified</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2368</td>
<td>Queue Reconstruction intermittently fails when queue size is large</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2366</td>
<td>Perforce Validation does not work if there is no global permissions set</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2360</td>
<td>Manual Build No Longer Shows Changes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2359</td>
<td>Plan Not Building</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2358</td>
<td>Can't Find JUnit Results</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2355</td>
<td>Bamboo 2.0 Upgrade Task 608 Fails</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2353</td>
<td>CVS Repository last update time should be from the files updated</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2348</td>
<td>Build result navigator misaligned</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2346</td>
<td>Failure to clear working directory on repository change if the agent has never built that plan before</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2330</td>
<td>Can't add new notification rules for plans</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2329</td>
<td>Bamboo sends IM messages with an empty &quot;to&quot; field</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2327</td>
<td>Dashboard Refresh problems</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2325</td>
<td>ScheduleBackupConfiguration exception if backup is configured</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2322</td>
<td>New Perforce Library puts password into command</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Issue</td>
<td>Description</td>
<td>Resolution</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>BAM-2321</td>
<td>Perforce failing to connect on agent</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2317</td>
<td>Quirks with logging on Bamboo agent.</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2314</td>
<td>Allow further hooks for the repository</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2312</td>
<td>Cannot Setup Perforce Repository: &quot;This user is not available on this port (Perforce server)&quot;</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2309</td>
<td>NullPointerException in AgentHeartBeat on startup</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2308</td>
<td>Improve memory footprint of Bamboo</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2299</td>
<td>Provide the ability to add a comment to a build result via Bamboo's REST API</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2294</td>
<td>Setup wizard fails at first step if IP address cannot be determined</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2293</td>
<td>On repository change clearing build directory fails</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2291</td>
<td>Can't add custom build requirement</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2290</td>
<td>Incorrect instructions for running remote agent JAR</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2288</td>
<td>Make details of the commits contained in a build visible via the Bamboo REST API</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2284</td>
<td>Upgrading from 1.2.4 will update the JDK keys to the wrong value</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2283</td>
<td>Upgrade task to change &quot;Queue&quot; to &quot;Agent&quot; in the 1.2.4 queue names</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2282</td>
<td>Ability to run automatic detection of environment variables as JDKs &amp; Builders</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2279</td>
<td>Improve link to download WAR edition of Bamboo</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2278</td>
<td>Build status doesn't update correctly on dashboard until a refresh is done</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2276</td>
<td>Better logging when remote agents disabled</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2270</td>
<td>Link incorrectly escaped in Builder JDK error Message</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2267</td>
<td>Editing capabilities and requirements</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2264</td>
<td>Extend Bamboo REST API to provide access to details of test cases</td>
<td>Resolved</td>
<td></td>
</tr>
<tr>
<td>BAM-2262</td>
<td>Improve Bamboo REST API error response to provide information that</td>
<td>Resolved</td>
<td></td>
</tr>
</tbody>
</table>
can be presented to an end user

<table>
<thead>
<tr>
<th>Issue</th>
<th>Description</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-2260</td>
<td>Mysql integration with Bamboo 2.0 Beta release 1 doesn't work</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2250</td>
<td>Agent bootstrap doesn't handle paths without trailing /</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2248</td>
<td>Unable to delete a build</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2246</td>
<td>Improve logging of why a build passes or fails</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2245</td>
<td>Forgotten password reports wrong error when user doesn't exist</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2241</td>
<td>Build Configuration page for script builder, doesn't show the values in the Arguments field in the UI</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2240</td>
<td>Number Format Exception when adding a label to a build</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2238</td>
<td>pre-authenticated RSS feed</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2236</td>
<td>Ensure that java.io.tmp exists when server / agent starts up</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2235</td>
<td>Live logs should show full username where possible</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2231</td>
<td>Agents matrix now display which requirements are missing</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2230</td>
<td>Autonaming of agents only goes up to (2)</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2229</td>
<td>More agent meta data</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2226</td>
<td>Build expiry fails when excluding labels</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2225</td>
<td>Bamboo sent empty notification e-mail</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2224</td>
<td>Regression: Failing tests no longer displayed in build failure summary</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2223</td>
<td>Build requirements are duplicated</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2222</td>
<td>Clicking on the &quot;stop build&quot; icon (red square) next to a build in the &quot;Current Activity&quot; tab throws exception</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2208</td>
<td>Cannot edit maven 2 plan</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2207</td>
<td>Unable to start Bamboo successfully if JRE cannot determine IP address</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2206</td>
<td>Alt text appearing instead of icons on All Plans tab</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2205</td>
<td>Cannot add comment to failed build</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2202</td>
<td>Agent shutdown if the server is shutdown</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2201</td>
<td>Bamboo use a unexistent SQL function with postgreSQL</td>
<td>Resolved</td>
</tr>
<tr>
<td>Issue</td>
<td>Description</td>
<td>Resolution</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>BAM-2199</td>
<td>BEAC Upgrade issues</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2197</td>
<td>UpgradeTask610BuilderPlan requires failing if we can't parse JDK</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2196</td>
<td>UpgradeTask608BuildLogStorageUpdate will fail a plan doesn't have a results folder</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2195</td>
<td>Clicking on favourite link navigates to malformed page</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2194</td>
<td>Clicking edit for an agent, then cancel, should return you to the page you were just on</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2193</td>
<td>Bamboo has problems if you change the SVN Source</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2176</td>
<td>Put in a null check to UpgradeTask608BuildLogStorageUpdate, to ensure the buildresultxml file being parsed is not null!</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2166</td>
<td>Importing Bambood data between different locales produces java.text.ParseException: Unparseable date: due to different date/time formats.</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2152</td>
<td>Indexer is not thread safe</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2150</td>
<td>Clicking &quot;Clear error from log&quot; on a plan's page navigates to the Dashboard</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2147</td>
<td>Reshuffle the admin menu to have Plan &amp; Build Resources subheading</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2146</td>
<td>Readd Builder and JDK page</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2140</td>
<td>A capability screen that allows bulk editing of capabilities and requirements</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2132</td>
<td>Two attempts to start a manual build of a plan in quick session locks the plan, and does not result in a build</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2129</td>
<td>getLastBuildResultSummaries is returning first, rather than last, BuildResultSummaries</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2122</td>
<td>Some exceptions during a build don't cause build to abort</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2118</td>
<td>CustomBuildInfo is not transferred from remote agent to server</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2117</td>
<td>VCSVersionReader returns an illegal null on Remote Agent</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2116</td>
<td>If Remote Agent dies, subsequent attempt to start it fails</td>
<td>Resolved</td>
</tr>
</tbody>
</table>
BAM-2113  Clover plug-in may use out-of-date Clover data  Resolved
BAM-2104  Confusing to upgrade an expired Bamboo 1.2 license, to Bamboo 2.0! Resolved
BAM-2103  Bamboo 1.2.4 import into Bamboo 2.0, throws exception due to UpgradeTask 612 Resolved
BAM-2102  Queued builds with no eligible builder are not built even if requirements are edited to make a builder eligible Resolved
BAM-2100  Bamboo should indicate queued builds that cannot be built on any active agent Resolved
BAM-2096  JDKs autodetected on Bamboo 2.0 agents don't match builds requiring autodetected JDKs imported from Bamboo 1.2.4 Resolved
BAM-2095  Detected JDKs may not be persisted Resolved
BAM-2077  Build Result Not Saved Resolved
BAM-2073  Nullpointer exception when trying to save build configuration Resolved
BAM-2063  Reimplement include / exclude files Resolved
BAM-2057  Installers should configure JMS broker URI Resolved
BAM-2032  Perforce configuration throws nullpointer exception Resolved
BAM-2029  Bamboo upgrade task fails if the build Results XML file is corrupted Resolved
BAM-2017  BuildResultsImpl.cloneAsBuildResults() does not copy the data included in the customBuildData variable Resolved
BAM-2007  Bamboo doesn't update available builders after each restart. Resolved
BAM-1996  Anonymous user is able to download artifacts even if Anonymous mode is disabled both at global level and plan level Resolved
BAM-1976  Build artifacts should display the size of the file (if single file) Resolved
BAM-1968  Make the install4j path a property in the standalone distribution project Resolved
BAM-1967  the main distribution POM doesn't use the correct parent POM Resolved
BAM-1966  The title of the Configure Groups page is incorrect Resolved
<table>
<thead>
<tr>
<th>Issue Key</th>
<th>Description</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-1964</td>
<td>Use the enforcer plugin to fail fast if a developer uses a jdk below 1.5.</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1944</td>
<td>Users should not be able to view plans if Global Anonymous access is disabled.</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1932</td>
<td>Fix up files tab</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1897</td>
<td>Change log entries that contain URLs appear fine on screen but the actual HREF is mangled.</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1882</td>
<td>Code coverage setup says optional, but is required.</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1847</td>
<td>Bamboo does not export ServerID</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1821</td>
<td>Bamboo should not die with freemarker error due to undefined values.</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1813</td>
<td>The project level RSS feeds are ordered wrong.</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1807</td>
<td>Subversion/source control queue</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1797</td>
<td>Bamboo runs out of Permgen space, when saving build result - due to massive build log output</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1752</td>
<td>Bamboo does not handle the failure to delete source code very well.</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1709</td>
<td>FileNotFoundException when accessing 'Index Operations' screen in a new empty instance.</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1696</td>
<td>Increase the VARCHAR(4000) in 'CUSTOM_INFO_DATA' column of 'BUILDRESULTSUMMARY_CUSTOMDATA' table from 4000</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1688</td>
<td>When updating group membership in crowd changes don't get reflected in Bamboo for a while</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1682</td>
<td>Prevent OutOfMemory errors while viewing the logs page</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1658</td>
<td>BuildNumber Stamper should be sidegraded to a CustomPreBuildAction</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1652</td>
<td>ChangeLogEntry has invalid myDate field</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1628</td>
<td>NullPointerException when adding many artifacts</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1618</td>
<td>Feedback as an admin message upon catastrophic IO exceptions</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1604</td>
<td>SVN Externals, do not work when pointed to a specific revision number</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1586</td>
<td>Bamboo doesn't free up SVN connections</td>
<td>Resolved</td>
</tr>
<tr>
<td>Issue ID</td>
<td>Description</td>
<td>Status</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>BAM-1585</td>
<td>Successful test results are apparently unordered</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1572</td>
<td>Add also the user who triggered the manual build to the Build Results page and Recently Completed Builds</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1518</td>
<td>Build logs should be separated from the activity logs</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1504</td>
<td>Ability to import data during setup without restarting</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1466</td>
<td>Bamboo should support MS SQL Database</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1465</td>
<td>Bamboo should support Oracle Database</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1416</td>
<td>Allow parallel VCS checkouts</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1401</td>
<td>Allow multiple build plans to be updating from SCM at the same time</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1365</td>
<td>Assigning of build to queues is non-optimal</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1356</td>
<td>Collect build changes for full checkout/clean builds</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1314</td>
<td>An option to delete local working copy of the repository before building</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1303</td>
<td>Ability to see who executed a manual build</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1297</td>
<td>Keep track of what/who triggered a build</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1295</td>
<td>Show a list of all successful tests run for a build</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1294</td>
<td>Ability to find a specific test on the successful tests screen</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1288</td>
<td>CVS changelogs are not picked up when the repository is on different timezone</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1257</td>
<td>Pause in between builds</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1178</td>
<td>Commit quiet period</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1110</td>
<td>Ability to add plan to build queue as you are creating said plan</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1082</td>
<td>Perforce source code directory (client root) is currently cached</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1069</td>
<td>Bamboo keeps building and won't stop</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1041</td>
<td>Build Expiry Settings - mark some build as non expirable</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1034</td>
<td>Allow a full checkout to be performed for each change</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1012</td>
<td>Output Shows Directories With Undisplalyable Character</td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-991</td>
<td>Need a BuildRequired queue</td>
<td></td>
</tr>
<tr>
<td>BAM-990</td>
<td>BuildChangeDetector should only run on a Quartz schedule</td>
<td></td>
</tr>
<tr>
<td>BAM-981</td>
<td>Checkins during a build don’t seem to trigger another build after the current one finished</td>
<td></td>
</tr>
<tr>
<td>BAM-926</td>
<td>An option to delay building after checkout detected for SVN</td>
<td></td>
</tr>
<tr>
<td>BAM-924</td>
<td>Perforce client error forces full checkout/build</td>
<td></td>
</tr>
<tr>
<td>BAM-760</td>
<td>Errors on dashboard should show which build they were raised in</td>
<td></td>
</tr>
<tr>
<td>BAM-705</td>
<td>Log the user who started the build</td>
<td></td>
</tr>
<tr>
<td>BAM-662</td>
<td>Having independent build queues underutilizes resources</td>
<td></td>
</tr>
<tr>
<td>BAM-498</td>
<td>Building hangs if one build does not get response from source repository.</td>
<td></td>
</tr>
<tr>
<td>BAM-479</td>
<td>Starting builds on multiple servers</td>
<td></td>
</tr>
<tr>
<td>BAM-462</td>
<td>Some errors in XML results parsing not reported to the UI</td>
<td></td>
</tr>
<tr>
<td>BAM-293</td>
<td>Ability to do remote/distributed builds</td>
<td></td>
</tr>
<tr>
<td>BAM-229</td>
<td>Link dependent builds to the build which launched them</td>
<td></td>
</tr>
<tr>
<td>BAM-195</td>
<td>Create new BuildReason object which can pass along more information for dependency builds.</td>
<td></td>
</tr>
<tr>
<td>BAM-21</td>
<td>Investigate making change detector to run asynchronously or in a queue</td>
<td></td>
</tr>
</tbody>
</table>
Bamboo 2.0 Beta 1 Upgrade Guide

This page last changed on Feb 26, 2008 by alui.

Upgrading to Bamboo 2.0 Beta 1

⚠️ If you are using a version of Bamboo prior to version 1.2, you will need to upgrade Bamboo to version 1.2 before you can upgrade to the 2.0 Beta. Note that the upgrade process from version 1.0.x is different from the upgrade process from version 1.1.x. Please follow the appropriate instructions below:

Upgrading from Bamboo 1.1.x
You will need to:
1. Upgrade to Bamboo 1.2 — please see the Bamboo 1.2 Upgrade Guide.
2. Then upgrade to the desired version of the Bamboo 2.0 Beta, as per the instructions below.

Upgrading from Bamboo 1.0.x
You will need to:
1. Upgrade to 1.1.2 first — please see the Bamboo 1.1.2 Upgrade Guide. (This step is necessary as there is an issue with the upgrade process from the 1.0.x series that we're currently looking into.)
2. Then upgrade to Bamboo 1.2 — please see the Bamboo 1.2 Upgrade Guide.
3. Then upgrade to the desired version of the Bamboo 2.0 Beta, as per the instructions below.

⚠️ It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide. Additionally, please note the following:

1. Adding a Broker URL property.

Bamboo uses a messaging broker to communicate with its remote build agents. To ensure this works properly, a URL must be specified. This URL is where Bamboo will set up its embedded broker. Remote agents will also be provided with this URL on startup.

To specify the broker URL, please add a `bamboo.jms.broker.url` property in your `bamboo.cfg.xml` file, located inside the Bamboo home directory. For example:

```xml
<property name="bamboo.jms.broker.uri">tcp://HOSTNAME:54663</property>
```

where HOSTNAME is the canonical name of your Bamboo server.

Please note, as remote agents use this URL to communicate to the server, you should take care not to specify localhost as the host name in the broker URL.

If no broker URL is found in `bamboo.cfg.xml`, Bamboo will default the broker URL to `tcp://HOSTNAME:54663` in the `bamboo.cfg.xml` file, as seen in the example above. Bamboo will also append the parameter `wireFormat.maxInactivityDuration=0` by default to any broker URL coming from `bamboo.cfg.xml`.

2. Changes to Server Configuration

JDK support

Bamboo 2.0 requires JDK 1.5 (i.e. JDK 1.4 is no longer supported). Please note that this does not affect the actual builds: it is only the Bamboo server itself that must be running JDK 1.5.

Database changes

The release of 2.0 will include some changes to column names in the database as follows:

- In the `BUILD_DEFINITION` table, the column `XML_DATA` will be changed to `XML_DEFINITION_DATA`
• In the BUILDRESULTSUMMARY_CUSTOMDATA table, the column CUSTOM_INFO_DATA will be changed to CUSTOM_INFO_VALUE

These fields have also had types changed to CLOB to increase their maximum lengths.

Plugins

If you are using external or custom plugins, please make sure that your plugins compile against Bamboo 2.0 before upgrading.

⚠️ We've made significant changes to the internals of the application for Bamboo 2.0. If you've installed an external plugin for 1.2.4, it's likely that it will be broken. Please take care when upgrading.

3. Changes to Build Queues and Build Plans

Bamboo 2.0 introduces the concepts of agents and capabilities. To preserve the functionality of your existing plans, JDKs, Builders and Build Queues, the following will automatically happen during the upgrade:

Conversion of Build Queues to Agents

Prior to Bamboo 2.0, you could have multiple build queues. In Bamboo 2.0, there is now only one build queue, but multiple agents (see diagram).

As part of the upgrade process,

• Each of your build queues will be converted to a local agent.
• If, prior to the upgrade, the build queue accepted builds from all plans, the agent will be given the following capability (and every plan will be given an equivalent requirement):
  ° Key: bamboo.1.2.queue
  ° Value: ALLOW_ANY_BUILDS
• Or if, prior to the upgrade, the build queue only accepted builds from specific plans, the agent will be given the following capability (and the relevant plans will be given an equivalent requirement):
  ° Key: bamboo.1.2.queue
  ° Value: <name of old queue>

If you wish to change this after the upgrade, please see 02. Configuring Agents and Capabilities and 1.2.4 Specifying a Plan’s Capability Requirements.

Conversion of Builders to Capabilities

Prior to Bamboo 2.0, your builders (e.g. Maven) were defined globally. In Bamboo 2.0, builders are now defined as agent capabilities and specified as plan requirements.

As part of the upgrade process,

• Each of your builders will be converted to a shared local capability (that is, it will apply to every local agent).
• Every plan will continue to have the same builder that it had before the upgrade.

If you wish to change this after the upgrade, please see 2.8 Configuring Capabilities and 1.2.4 Specifying a Plan’s Capability Requirements.

Conversion of JDKs to Capabilities

Prior to Bamboo 2.0, your JDKs (e.g. JDK 1.5) were defined globally. In Bamboo 2.0, JDKs are now defined as agent capabilities and specified as plan requirements.

As part of the upgrade process,

• Each of your JDKs will be converted to shared local capabilities (that is, it will apply to every local agent).
• Upon conversion, the labels of each of your JDKs will upgraded to the Bamboo 2.0 JDK label format, (i.e. ‘JDK 9.9.9_99’).
• Upon conversion, two more generic versions of the labels will be created for each JDK, (i.e. 'JDK 9.9' and 'JDK').
• Every plan will have its requirements upgraded, to keep the association with the same JDK that it had before the upgrade.

If you wish to change this after the upgrade, please see 2.8 Configuring Capabilities and 1.2.4 Specifying a Plan's Capability Requirements.
Atlassian is proud to announce the release of Bamboo 2.0 Beta 2. This point release includes more than 10 minor fixes and improvements. Bamboo 2.0 Beta 2 can be downloaded here.

Before upgrading, please read the Bamboo 2.0 Beta 2 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

Known Issues

The following issues are applicable at the time of the Bamboo 2.0 Beta 2 release. Please refer to the aggregated Bamboo 2.0 Beta Release Notes to review the complete list of issues and fixes for each beta release.

- Perforce is not supported in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release).
- Bamboo currently does not work with MySQL. See BAM-2260 for further details.
- Plans currently cannot be edited, if Bamboo is integrated with a PostgreSQL database. See BAM-2208 for further details.
- Secured remote agents are not supported in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release). Please see the Bamboo security advisory for further details.
- JDK and Builders pages have not been included in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release).
- Please note that builds are currently allocated to agents during queuing time, not execution time. This may occasionally mean that a build is executed by an agent that you have disabled, which is slightly different from the functionality described in 2.5 Monitoring Agent Status. For example:
  1. Agent A is enabled and is currently executing a build for Plan X.
  2. Plan Y submits a build to the queue, and the queue assigns the build to Agent A.
  3. You disable Agent A.
  4. Agent A completes Plan X's build.
  5. Although agent A is disabled, it will still run Plan Y's build, because it was determined as executable when plan Y was queued.

Updates and issues fixed

Please help us with the final 2.0 release by reporting any bugs and issues you find, in the Bamboo project at jira.atlassian.com.

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Assigned To</th>
<th>Reporter</th>
<th>Pr</th>
<th>Status</th>
<th>Res</th>
<th>Created</th>
<th>Updated</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-2236</td>
<td>Ensure java.io.tmp exists when server / agent starts up</td>
<td>Unassigned</td>
<td>Mark Chaimungkalanont [Atlassian]</td>
<td></td>
<td>FIXED</td>
<td>Resolved</td>
<td>Feb 12, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
</tbody>
</table>
BAM-223 | Agents matrix now display which requirements are missing
Mark Chaimungkalanont [Atlassian]
Resolved FIXED Feb 12, 2008 Apr 01, 2008

BAM-222 | Regression: Failing tests no longer displayed in build failure summary
Mark Chaimungkalanont [Atlassian]
Resolved FIXED Feb 11, 2008 Apr 01, 2008

BAM-222 | Clicking on the "stop build" icon (red square) next to a build in the "Current Activity" tab throws exception
Mark Chaimungkalanont [Atlassian]
Resolved FIXED Feb 11, 2008 Apr 01, 2008

BAM-220 | Cannot edit maven 2 plan
Mark Chaimungkalanont [Atlassian]
Resolved FIXED Feb 10, 2008 Apr 01, 2008

BAM-220 | Unable to start Bamboo successfully if JRE cannot determine IP address
Adrian Hempel Hempel [Atlassian]
Resolved FIXED Feb 10, 2008 Apr 01, 2008

BAM-220 | Alt text appearing instead of icons on All Plans tab
Mark Chaimungkalanont [Atlassian]
Resolved FIXED Feb 10, 2008 Apr 01, 2008

BAM-220 | Cannot add comment to failed build
Mark Chaimungkalanont [Atlassian]
Resolved FIXED Feb 10, 2008 Apr 01, 2008

BAM-220 | Bamboo use a unexistant SQL function withpostgresql
Mark Benjamin ELMAN [Atlassian]
Resolved FIXED Feb 08, 2008 Apr 01, 2008
BAM-2199
Upgrade issues
Mark Chaimungkalanont
Resolved
Feb 07, 2008
Apr 01, 2008

BAM-1658
BuildNumberStamper should be sidegraded to a CustomPreBuildAction
Brydie McCoy
Resolved
Aug 30, 2007
Apr 01, 2008
Upgrading to Bamboo 2.0 Beta 2

These instructions outline how to upgrade Bamboo from version 2.0 Beta 1 to 2.0 Beta 2. If you are upgrading from a version prior to 2.0 Beta 1, please also refer to the aggregated upgrade guides for details on the previous beta releases.

⚠️ It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

No additional upgrade tasks are required to upgrade from Bamboo 2.0 Beta 1 to 2.0 Beta 2.
Bamboo 2.0 Beta 3 Release Notes

This page last changed on Mar 03, 2008 by alui.

27 February, 2008

Atlassian Software Systems presents Bamboo 2.0 Beta 3

Bamboo 2.0 Beta 3 introduces a number of new features, including Perforce support and the ability to connect to Oracle and MS SQL as external databases. A number of significant fixes have been included as well, improving the overall quality and experience of the beta.

Upgrading to Bamboo 2.0 Beta

Bamboo 2.0 Beta can be downloaded from the Bamboo Download Centre. Before upgrading, please read the Bamboo 2.0 Beta 3 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta 2, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

What's New in Bamboo 2.0 Beta 3?

- **Perforce Support** — Bamboo brings back out of the box support for Perforce source repositories. A number of changes have been made to Perforce configuration to ensure that it works correctly with distributed builds. Read more about configuring Perforce.

- **Oracle and MS SQL Server Support** — By popular request, Bamboo's supported databases now include Oracle and MS SQL Server. Read more about connecting Bamboo to Oracle and MS SQL Server.

- **Major Bug Fixes** — Bamboo now works with PostgreSQL and MySQL correctly. Please see the relevant Postgresql and MySQL JIRA issues for details about the fixes.

Known Issues

The following issues are applicable at the time of the Bamboo 2.0 Beta 3 release. Please refer to each of the previous beta release notes to review the complete list of issues and fixes for each beta release.

- Perforce is not supported in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release). FIXED!
- Bamboo currently does not work with MySQL. See BAM-2260 for further details. FIXED!
- Plans currently cannot be edited, if Bamboo is integrated with a Postgresql database. See BAM-2208 for further details. FIXED!
- Hibernate Errors in logs - this is a known issue, caused due to our pre-hibernate upgrade tasks to prepare Bamboo database for Oracle and MS SQL Server compatibility - For further details, see this Knowledge base article.
- Secured remote agents are not supported in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release). Please see the Bamboo security advisory for further details.
- JDK and Builders pages have not been included in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release).
- Please note that builds are currently allocated to agents during queuing time, not execution time. This may occasionally mean that a build is executed by an agent that you have disabled, which is slightly different from the functionality described in 2.5 Monitoring Agent Status. For example:
  1. Agent A is enabled and is currently executing a build for Plan X.
  2. Plan Y submits a build to the queue, and the queue assigns the build to Agent A.
  3. You disable Agent A.
  4. Agent A completes Plan X's build.
5. Although agent A is disabled, it will still run Plan Y's build, because it was determined as executable when plan Y was queued.

Updates and Fixes in this Release

Please help us with the final 2.0 release by reporting any bugs and issues you find, in the Bamboo project at jira.atlassian.com.

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Pr</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-2290</td>
<td>Incorrect instructions for running remote agent JAR</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2284</td>
<td>Upgrading from 1.2.4 will update the JDK keys to the wrong value</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2282</td>
<td>Ability to run automatic detection of environment variables as JDKs &amp; Builders</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2276</td>
<td>Better logging when remote agents disabled</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2264</td>
<td>Extend Bamboo REST API to provide access to details of test cases</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2262</td>
<td>Improve Bamboo REST API error response to provide information that can be presented to an end user</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2260</td>
<td>Mysql integration with Bamboo 2.0 Beta release 1 doesn't work</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2245</td>
<td>Forgotten password reports wrong error when user doesn't exist</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2229</td>
<td>More agent meta data</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-2112</td>
<td>Crowd with delegated LDAP auth = update documentation for Bamboo-Crowd integration</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1807</td>
<td>Subversion/source control queue</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1504</td>
<td>Ability to import data during setup without restarting</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1466</td>
<td>Bamboo should support MS SQL Database</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1465</td>
<td>Bamboo should support Oracle Database</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1365</td>
<td>Assigning of build to queues is non-optimal</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1110</td>
<td>Ability to add plan to build queue as you are creating said plan</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-1082</td>
<td>Perforce source code directory (client root) is currently cached</td>
<td></td>
<td>Resolved</td>
</tr>
<tr>
<td>BAM-924</td>
<td>Perforce client error forces full checkout/build</td>
<td></td>
<td>Resolved</td>
</tr>
</tbody>
</table>
Upgrading to Bamboo 2.0 Beta 3

These instructions outline how to upgrade Bamboo from version 2.0 Beta 2 to 2.0 Beta 3. If you are upgrading from a version prior to 2.0 Beta 2, please refer to the aggregated upgrade guides for details on the previous beta releases.

It is strongly recommended that you back up your xml-data directory before proceeding. You are also strongly recommended to back up your database due to schema changes in this release. For full instructions please follow the Bamboo Upgrade Guide. Additionally, please note the following:

1. Changes to Repositories

Bamboo 2.0 introduces the concepts of agents and capabilities. To preserve the functionality of your existing Repositories, the following will automatically happen during the upgrade:

Conversion of Perforce P4 Client Application Location to a Capability

With the introduction of remote agents in Bamboo 2.0, the location of the Perforce P4 client application now needs to be specified as a capability. To create build plans using Perforce as repository, a shared local capability must be created for the P4 client application location. In addition, agent-specific remote capabilities must be created for each remote agent using Perforce.

As part of the upgrade process,

- A shared local Perforce capability will be created for the Perforce P4 client application location. The upgrade task reads this information from the system's environment variables. If the Perforce P4 client application location has not been specified as an environment variable, the shared local capability will need to be set up manually.

The upgrade task will not create agent-specific Perforce capabilities for any remote agents. These capabilities will need to be set up manually.

Please see 2.8.4 Configuring a new Perforce Capability for further details on creating Perforce capabilities.
4 March 2008

Atlassian is proud to announce the release of Bamboo 2.0 Beta 4. This point release includes more than 15 minor fixes and improvements. Bamboo 2.0 Beta 4 can be downloaded here.

Before upgrading, please read the Bamboo 2.0 Beta 4 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

Major Bug Fixes

- Previously, Bamboo determined which agents could build a plan when the plan was queued, but would then incorrectly ignore any subsequent changes (including disabling the agent). Bamboo will now correct update where plans can be built, even if changes are made after the plan has been queued.

For the Developers

- The new 'RepositoryEventAware' interface allows you to implement custom actions before and/or after retrieving source code from your repository. Read more about extending the standard repository functionality.

Known Issues

The following issues are applicable at the time of the Bamboo 2.0 Beta 4 release. Please refer to each of the previous beta release notes to review the complete list of issues and fixes for each beta release.

- Please note that builds are currently allocated to agents during queuing time, not execution time. This may occasionally mean that a build is executed by an agent that you have disabled, which is slightly different from the functionality described in 2.5 Monitoring Agent Status. FIXED!
- Hibernate Errors in logs - this is a known issue, caused due to our pre-hibernate upgrade tasks to prepare Bamboo database for Oracle and MS SQL Server compatibility - For further details, see this Knowledge base article.
- Secured remote agents are not supported in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release). Please see the Bamboo security advisory for further details.
- JDK and Builders pages have not been included in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release).

Updates and issues fixed

Please help us with the final 2.0 release by reporting any bugs and issues you find, in the Bamboo project at jira.atlassian.com.

### Atlassian JIRA (16 issues)

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Assigned</th>
<th>Reporter</th>
<th>Pr</th>
<th>Status</th>
<th>Res</th>
<th>Created</th>
<th>Updated</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-232</td>
<td>Perforce failing to connect on agent</td>
<td>Brydie McCoy</td>
<td>Brydie McCoy [Atlassian\Atlassian]</td>
<td></td>
<td>FIXED</td>
<td>Resolved</td>
<td>Mar 02, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
<tr>
<td>BAM-231</td>
<td>Allow further hooks for the repository</td>
<td>Mark Chaimungkalanont</td>
<td>Mark Chaimungkalanont [Atlassian\Atlassian]</td>
<td></td>
<td>FIXED</td>
<td>Resolved</td>
<td>Feb 28, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
<tr>
<td>BAM-230</td>
<td>Improve memory footprint of Bamboo</td>
<td>Mark Chaimungkalanont</td>
<td>Mark Chaimungkalanont [Atlassian\Atlassian]</td>
<td></td>
<td>FIXED</td>
<td>Resolved</td>
<td>Feb 28, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
</tbody>
</table>
**BAM-2299**
Provide the ability to add a comment to a build result via Bamboo's REST API

**BAM-2299**
Adrian Hempel [Atlassian]  
*Fixed*  
Resolved  
Feb 27, 2008  
Apr 01, 2008

**BAM-2299**
Set up wizard fails at first step if IP address cannot be determined

**BAM-2299**
Brydie McCoy [Atlassian]  
*Fixed*  
Resolved  
Feb 26, 2008  
Apr 01, 2008

**BAM-2299**
On repository change clearing build directory fails

**BAM-2299**
Adrian Hempel [Atlassian]  
*Fixed*  
Resolved  
Feb 26, 2008  
Apr 01, 2008

**BAM-2299**
Can't add custom build requirement

**BAM-2299**
Adrian Hempel [Atlassian]  
*Fixed*  
Resolved  
Feb 25, 2008  
Apr 02, 2008

**BAM-2299**
Make details of the commits contained in a build visible via the Bamboo REST API

**BAM-2299**
Adrian Hempel [Atlassian]  
*Fixed*  
Resolved  
Feb 25, 2008  
Apr 01, 2008

**BAM-2299**
Unable to delete a build

**BAM-2299**
Mark Chaimungkalanont [Atlassian]  
*Fixed*  
Resolved  
Feb 15, 2008  
Apr 01, 2008

**BAM-2299**
Number Format Exception when adding a label to a build

**BAM-2299**
Brydie McCoy Nick Pellow [Atlassian]  
*Fixed*  
Resolved  
Feb 13, 2008  
Apr 01, 2008

**BAM-2299**
Pre-authenticated RSS feed

**BAM-2299**
Ajay Sridhar [Atlassian]  
*Fixed*  
Resolved  
Feb 13, 2008  
Apr 01, 2008

**BAM-2299**
Bamboo 2.0 beta 1 WAR file not working in Tomcat

**BAM-2299**
Ajay Matty [Atlassian]  
*Not a Bug*  
Resolved  
Feb 08, 2008  
Apr 01, 2008

**BAM-2299**
Bamboo has problems if you change

**BAM-2299**
Brydie McCoy [Atlassian]  
*Fixed*  
Resolved  
Feb 07, 2008  
Apr 01, 2008
**BAM-2102**: Queued builds with no eligible builder are not built even if requirements are edited to make a builder eligible.


**BAM-1996**: Anonymous user is able to download artifacts even if Anonymous mode is disabled both at global level and plan level.


**BAM-1944**: Users should not be able to view plans if Global Anonymous access is disabled.

Upgrading to Bamboo 2.0 Beta 4

These instructions outline how to upgrade Bamboo from version 2.0 Beta 3 to 2.0 Beta 4. If you are upgrading from a version prior to 2.0 Beta 3, please also refer to the aggregated upgrade guides for details on the previous beta releases.

⚠️ It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

No additional upgrade tasks are required to upgrade from Bamboo 2.0 Beta 3 to 2.0 Beta 4.
Bamboo 2.0 Beta 5 Release Notes

This page last changed on Mar 11, 2008 by alui.

12 March 2008
Atlassian is proud to announce the release of Bamboo 2.0 Beta 5. This point release includes more than 10 minor fixes and improvements. Bamboo 2.0 Beta 5 can be downloaded here.

Before upgrading, please read the Bamboo 2.0 Beta 5 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

JDK and Builders page now included in Bamboo 2.0 Beta

- The JDK and Builder pages which were temporarily removed from Bamboo for the 2.0 Beta have now been restored.

Known Issues

The following issues are applicable at the time of the Bamboo 2.0 Beta 5 release. Please refer to each of the previous beta release notes to review the complete list of issues and fixes for each beta release.

- JDK and Builders pages have not been included in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release). FIXED!
- Hibernate Errors in logs - this is a known issue, caused due to our pre-hibernate upgrade tasks to prepare Bamboo database for Oracle and MS SQL Server compatibility - For further details, see this Knowledge base article.
- Secured remote agents are not supported in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release). Please see the Bamboo security advisory for further details.

Updates and issues fixed

Please help us with the final 2.0 release by reporting any bugs and issues you find, in the Bamboo project at jira.atlassian.com.

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Pr</th>
<th>Status</th>
<th>Res</th>
<th>Created</th>
<th>Updated</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-2346</td>
<td>Failure to clear working directory on repository change if the agent has never built that plan before.</td>
<td>Brydie McCoy</td>
<td>Brydie McCoy</td>
<td></td>
<td></td>
<td>FIXED</td>
<td>Mar 06, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Resolved</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAM-2326</td>
<td>Bamboo sends IM messages with an empty &quot;to&quot; field.</td>
<td>Ajay Sridhar</td>
<td>Ajay Sridhar</td>
<td></td>
<td>FIXED</td>
<td>Resolved</td>
<td>Mar 04, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Resolved</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Atlassian JIRA (15 issues)
Figure out why functional tests stop logging

Bam-232 Schedule backup configuration exception if backup is configured

Bam-232 New Perforce Library puts password into command line which is not all that secure

Bam-231 Cannot Setup Perforce Repository: "This user is not available on this port (Perforce server)"

Bam-227 Build status doesn't update correctly on dashboard until a refresh is done

Bam-223 Autonaming of agents only goes up to (2)

Bam-219 Clicking on favourite link navigates to malformed page

Bam-219 Clicking edit for an agent, then cancel, should return
BAM-214 Readd Builder and JDK page
Mark Chaimungkalanont [Atlassian\Atlassian]
Mark Chaimungkalanont [Atlassian\Atlassian]
Resolved
FIXED
Jan 22, 2008
Apr 01, 2008

BAM-211 VCSVersionReader returns an illegal null on Remote Agent
Brydie McCoy Hempel [Atlassian\Atlassian]
Mark Chaimungkalanont [Atlassian\Atlassian]
Resolved
FIXED
Jan 22, 2008
Apr 01, 2008

BAM-210 Confusing to upgrade an expired Bamboo 1.2 license to Bamboo 2.0!
Brydie McCoy Sridhar [Atlassian\Atlassian]
Ajay Sridhar [Atlassian\Atlassian]
Resolved
FIXED
Jan 16, 2008
Apr 01, 2008

BAM-206 Reimplement include / exclude files
Edwin Wong Chaimungkalanont [Atlassian\Atlassian]
Mark Chaimungkalanont [Atlassian\Atlassian]
Resolved
FIXED
Jan 06, 2008
Apr 01, 2008
Upgrading to Bamboo 2.0 Beta 5

These instructions outline how to upgrade Bamboo from version 2.0 Beta 4 to 2.0 Beta 5. If you are upgrading from a version prior to 2.0 Beta 4, please also refer to the aggregated upgrade guides for details on the previous beta releases.

It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

No additional upgrade tasks are required to upgrade from Bamboo 2.0 Beta 4 to 2.0 Beta 5.
18 March 2008

Atlassian is proud to announce the release of Bamboo 2.0 Beta 6. This point release includes more than 5 minor fixes and improvements. Bamboo 2.0 Beta 6 can be downloaded here.

Before upgrading, please read the Bamboo 2.0 Beta 6 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

Secured Remote Agents are now Supported

- Remote agents can now be secured with the appropriate SSL configuration. Read more about Securing your Remote Agents.
  
  We strongly recommend that you do not enable remote agent installation on any Bamboo instance accessible from a public or untrusted network without securing your remote agents. If you choose to enable your remote agents without securing them, please read this security advisory to understand the security implications.

Changes to Client Workspace Configuration for Perforce

- If you use Perforce, you can now choose whether you want Bamboo to manage your client workspace (i.e. set the client root) or manage it yourself. Read more about Perforce configuration.

'Shared Local Capabilities' are now called 'Local Server Capabilities'

- All references to 'Shared Local Capabilities' (or equivalent terminology) have been changed to 'Local Server Capabilities' in the Bamboo user interface.

Known Issues

The following issues are applicable at the time of the Bamboo 2.0 Beta 6 release. Please refer to each of the previous beta release notes to review the complete list of issues and fixes for each beta release.

- Secured remote agents are not supported in the Bamboo 2.0 Beta (but will be supported in the official Bamboo 2.0 release). Please see the Bamboo security advisory for further details. RESOLVED!
- Hibernate Errors in logs - this is a known issue, caused due to our pre-hibernate upgrade tasks to prepare Bamboo database for Oracle and MS SQL Server compatibility - For further details, see this Knowledge base article.

Updates and issues fixed

Please help us with the final 2.0 release by reporting any bugs and issues you find, in the Bamboo project at jira.atlassian.com.

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Pr</th>
<th>Status</th>
<th>Res</th>
<th>Created</th>
<th>Updated</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-2386</td>
<td>Perforce not connecting when running Bamboo as a service</td>
<td>Brydie McCoy</td>
<td>Brydie McCoy</td>
<td>✔️</td>
<td>FIXED</td>
<td>Resolved</td>
<td>Mar 18, 2008</td>
<td>Mar 19, 2008</td>
<td></td>
</tr>
<tr>
<td>BAM-2368</td>
<td>Queue Reconstruction intermittently fails</td>
<td>Edwin Wong</td>
<td>Edwin Wong</td>
<td>✔️</td>
<td>FIXED</td>
<td>Resolved</td>
<td>Mar 12, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
</tbody>
</table>
when queue size is large

BAM-236
Perforce Validation does not work if there is no global permissions set

BAM-236
Resolve perforce source directory issue

BAM-235
Bamboo 2.0 Upgrade Task 608 Fails

BAM-235
CVS Repository last update time should be from the files updated

BAM-234
Build result navigator misaligned

BAM-225
Agent bootstrap doesn't handle paths without trailing /

BAM-222
Build expiry fails when excluding labels

BAM-193
Fix up files tab

BAM-175
Bamboo does not handle the failure to delete source code very well
Upgrading to Bamboo 2.0 Beta 6

These instructions outline how to upgrade Bamboo from version 2.0 Beta 5 to 2.0 Beta 6. If you are upgrading from a version prior to 2.0 Beta 5, please also refer to the aggregated upgrade guides for details on the previous beta releases.

⚠️ It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

1. Changes to Perforce Workspace Management

If you use Perforce, you can now choose whether you want Bamboo to manage your workspace or whether you want to manage it yourself. Prior to this release, Bamboo would automatically manage your workspace (i.e. changed the client root). Hence, if you want to manage your workspace in this release, you will need to reset your client roots.
Bamboo 2.0 Beta 8 Release Notes

This page last changed on Mar 27, 2008 by alui.

27 March 2008

Atlassian is proud to announce the release of Bamboo 2.0 Beta 8. This point release includes more than 20 minor fixes and improvements. Bamboo 2.0 Beta 8 can be downloaded here.

Before upgrading, please read the Bamboo 2.0 Beta 8 Upgrade Guide (Please note that there is no 2.0 Beta 7). If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

Quiet Period Functionality Supported for Subversion & Perforce

By popular request, Quiet Period parameters can now be specified for Subversion and Perforce when configuring a source repository for a build plan. You can choose to set how long Bamboo should wait after a commit before triggering a build, and the number of times it retries before initiating a build. Read more about configuring Subversion and Perforce source repositories.

'Force Clean Builds' Supported

Also by popular request, you can now force Bamboo to run 'Clean Builds' in a build plan. That is, the source directory is removed and then checked out from the repository prior to each build. Read more about this function in Specifying a Plan's Source Repository.

Known Issues

The following issues are applicable at the time of the Bamboo 2.0 Beta 8 release. Please refer to each of the previous beta release notes to review the complete list of issues and fixes for each beta release.

- Hibernate Errors in logs - this is a known issue, caused due to our pre-hibernate upgrade tasks to prepare Bamboo database for Oracle and MS SQL Server compatibility - For further details, see this Knowledge base article.

Updates and issues fixed

Please help us with the final 2.0 release by reporting any bugs and issues you find, in the Bamboo project at jira.atlassian.com.

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Pr</th>
<th>Status</th>
<th>Res</th>
<th>Created</th>
<th>Updated</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-239</td>
<td>SVN exteranls now prohibitively slow</td>
<td>Mark</td>
<td>Chaimungkalanont</td>
<td></td>
<td></td>
<td>Resolved</td>
<td>Mar 19, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
<tr>
<td>BAM-2384</td>
<td>Test setting up database with JNDI</td>
<td>Brydie</td>
<td>McCoy</td>
<td></td>
<td></td>
<td>Resolved</td>
<td>Mar 18, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
<tr>
<td>BAM-2383</td>
<td>NPE trying to access <a href="https://bamboo.developer.atlassian.com/browse/FISH-STABLE/test">https://bamboo.developer.atlassian.com/browse/FISH-STABLE/test</a></td>
<td>Mark</td>
<td>Chaimungkalanont</td>
<td></td>
<td></td>
<td>Resolved</td>
<td>Mar 17, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
<tr>
<td>BAM-2382</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Resolved</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Perforce Web Repository URLs are not implemented in 2.0

BAM-2379 Artifacts copy problems with local/remote agents

BAM-2377 Repository web URL still create links even though none is specified

BAM-2362 Can't edit existing perforce repository plan to a new P4 workspace

BAM-2358 Can't Find JUnit Results

BAM-2309 NullPointerException in AgentHeartBeat on startup

BAM-2283 Upgrade task to change "Queue" to "Agent" in the 1.2.4 queue names

BAM-2270 Link incorrectly escaped in Builder JDK error Message

BAM-2257 Variable substitution doesn't work in Bamboo 2.0

BAM-2202 Agent shutdown if the
server is shutdown

**BAM-2152**

Indexer is not thread safe

Mark Edwin Chaimungkalanont [Atlassian/Atlassian]

**BAM-2142**

Reshuffle the admin menu to have Plan & Build Resources subheading

Mark Mark Chaimungkalanont [Atlassian/Atlassian]

**BAM-2099**

Detected JDKs may not be persisted

Adrian Adrian Hempel Hempel [Atlassian/Atlassian]

**BAM-2000**

Bamboo doesn't update available builders after each restart.

Mark Ajay Chaimungkalanont [Atlassian/Atlassian]

**BAM-1314**

An option to delete local working copy of the repository before building

Mark Eric Chaimungkalanont [Atlassian]

**BAM-1178**

Commit quiet period

Mark Bob Swift Chaimungkalanont [Atlassian]

**BAM-1034**

Allow a full checkout to be performed for each change

Mark Mark Chaimungkalanont [Atlassian/Atlassian]

**BAM-926**

An option to delay building after checkout detected for SVN

Mark Mark Chaimungkalanont [Atlassian/Atlassian]
Upgrading to Bamboo 2.0 Beta 8

These instructions outline how to upgrade Bamboo from version 2.0 Beta 6 to 2.0 Beta 8 (Please note that there is no 2.0 Beta 7). If you are upgrading from a version prior to 2.0 Beta 6, please also refer to the aggregated upgrade guides for details on the previous beta releases.

⚠️ It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

No additional upgrade tasks are required to upgrade from Bamboo 2.0 Beta 6 to 2.0 Beta 8.
Bamboo 2.0 Beta 9 Release Notes

This page last changed on Apr 02, 2008 by alui.

3 April 2008
Atlassian is proud to announce the release of Bamboo 2.0 Beta 9. This point release includes more than 10 minor fixes and improvements. Bamboo 2.0 Beta 9 can be downloaded here.

Before upgrading, please read the Bamboo 2.0 Beta 9 Upgrade Guide. If you are upgrading from a version of Bamboo prior to the Bamboo 2.0 Beta, please read all of the Bamboo 2.0 Beta Release Notes and Bamboo 2.0 Beta Upgrade Guides before upgrading.

Edit and Rename Capabilities

You can now edit capabilities in Bamboo, as listed below:

- If you are editing a Builder capability, you can modify the 'Path' of the builder.
- If you are editing a JDK capability, you can modify the 'Java Home' of the JDK.
- If you are editing a Custom capability, you can modify the 'Value' of the capability.
- If you are editing a Perforce capability, you can modify the 'Perforce Executable' path.

You can also rename a capability. This is reflected in any plans that the capability is specified as a requirement for.

Read more about editing capabilities and renaming capabilities.

View Agents and Plans related to a Capability

A 'View Capability' screen is now available in Bamboo 2.0. This screen lists the agents that have/inherit a particular capability, as well as which plans have the capability specified as a requirement.

Read more about viewing capabilities.

Known Issues

The following issues are applicable at the time of the Bamboo 2.0 Beta 9 release. Please refer to each of the previous beta release notes to review the complete list of issues and fixes for each beta release.

- Hibernate Errors in logs - this is a known issue, caused due to our pre-hibernate upgrade tasks to prepare Bamboo database for Oracle and MS SQL Server compatibility - For further details, see this Knowledge base article.

Updates and issues fixed

Please help us with the final 2.0 release by reporting any bugs and issues you find, in the Bamboo project at jira.atlassian.com.

<table>
<thead>
<tr>
<th>Key</th>
<th>Summary</th>
<th>Assignee</th>
<th>Reporter</th>
<th>Pr</th>
<th>Status</th>
<th>Res</th>
<th>Created</th>
<th>Updated</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-2428</td>
<td>Allow configuration of SVNKit to spool / not spool</td>
<td>Mark Chaimungkalanont</td>
<td>[Atlassian]</td>
<td></td>
<td>FIXED</td>
<td>Resolved</td>
<td>Apr 01, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
<tr>
<td>BAM-2407</td>
<td>Remote Agent running on Sun JDK 1.6 exits after build is</td>
<td>Adrian Hempel</td>
<td>[Atlassian]</td>
<td></td>
<td>FIXED</td>
<td>Resolved</td>
<td>Mar 27, 2008</td>
<td>Apr 01, 2008</td>
<td></td>
</tr>
</tbody>
</table>
BAM-2405
Add extension builder interfaces

BAM-2408
Remote agent restart can leave builds in a building state while the agent reports it is idle

BAM-2389
Add more sample plugins to Bamboo Development Kit

BAM-2388
Failing since build for a test reports the wrong result

BAM-2360
Manual Build No Longer Shows Changes

BAM-2359
Plan Not Building

BAM-2315
LDAP user cache is not refreshed when principals are added to AD/LDAP

BAM-2260
Editing capabilities and requirements

BAM-2241
Build Configuration page for script builder, doesn't show the values
in the Arguments field in the UI.

**BAM-2140**
Mark Chaimungkalanont
A capability screen that allows bulk editing of capabilities and requirements
Mark Chaimungkalanont
[Atlassian] [Atlassian]

Resolved
FIXED
Jan 22, 2008
Apr 01, 2008

**BAM-2122**
Some exceptions during a build don't cause build to abort
Adrian Hempel
Adrian Hempel
[Atlassian] [Atlassian]

Resolved
FIXED
Jan 22, 2008
Apr 01, 2008

**BAM-1862**
Move the XFire dependency for Crowd from 1.2.1 to 1.2.6
Brydie McCoy
Justin Koke
Brydie McCoy
[Atlassian] [Atlassian]

Resolved
FIXED
Oct 24, 2007
Apr 01, 2008

**BAM-1813**
The project level RSS feeds are ordered wrong.
Brydie McCoy
Ajay Sridhar
Brydie McCoy
[Atlassian] [Atlassian]

Resolved
FIXED
Oct 11, 2007
Apr 01, 2008
Upgrading to Bamboo 2.0 Beta 9

These instructions outline how to upgrade Bamboo from version 2.0 Beta 8 to 2.0 Beta 9. If you are upgrading from a version prior to 2.0 Beta 8, please also refer to the aggregated upgrade guides for details on the previous beta releases.

⚠️ It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

No additional upgrade tasks are required to upgrade from Bamboo 2.0 Beta 8 to 2.0 Beta 9.
Bamboo 2.0 Beta Upgrade Guide

This page last changed on Apr 02, 2008 by alui.

This page contains a live aggregate of all version-specific upgrade guides for the Bamboo 2.0 Beta.

How to read the Bamboo 2.0 Beta Upgrade Guides

- Read all Bamboo 2.0 Beta Upgrade Guides up to the version that you are upgrading to.
- Read the relevant release notes.
- Read the Bamboo Upgrade Guide for general upgrade instructions.

Bamboo 2.0 Beta Upgrade Guides

- Upgrading to Bamboo 2.0 Beta 1
- Upgrading to Bamboo 2.0 Beta 2
- Upgrading to Bamboo 2.0 Beta 3
- Upgrading to Bamboo 2.0 Beta 4
- Upgrading to Bamboo 2.0 Beta 5
- Upgrading to Bamboo 2.0 Beta 6
- Upgrading to Bamboo 2.0 Beta 8
- Upgrading to Bamboo 2.0 Beta 9

Upgrading to Bamboo 2.0 Beta 1

⚠️ If you are using a version of Bamboo prior to version 1.2, you will need to upgrade Bamboo to version 1.2 before you can upgrade to the 2.0 Beta. Note that the upgrade process from version 1.0.x is different from the upgrade process from version 1.1.x. Please follow the appropriate instructions below:

Upgrading from Bamboo 1.1.x

You will need to:

1. Upgrade to Bamboo 1.2 — please see the Bamboo 1.2 Upgrade Guide.
2. Then upgrade to the desired version of the Bamboo 2.0 Beta, as per the instructions below.

Upgrading from Bamboo 1.0.x

You will need to:

1. Upgrade to 1.1.2 first — please see the Bamboo 1.1.2 Upgrade Guide. (This step is necessary as there is an issue with the upgrade process from the 1.0.x series that we're currently looking into.)
2. Then upgrade to Bamboo 1.2 — please see the Bamboo 1.2 Upgrade Guide.
3. Then upgrade to the desired version of the Bamboo 2.0 Beta, as per the instructions below.

⚠️ It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide. Additionally, please note the following:

1. Adding a Broker URL property.

Bamboo uses a messaging broker to communicate with its remote build agents. To ensure this works properly, a URL must be specified. This URL is where Bamboo will set up its embedded broker. Remote agents will also be provided with this URL on startup.

To specify the broker URL, please add a bamboo.jms.broker.url property in your bamboo.cfg.xml file, located inside the Bamboo home directory. For example:

```xml
<property name="bamboo.jms.broker.uri">tcp://HOSTNAME:54663</property>
```

where HOSTNAME is the canonical name of your Bamboo server.
Please note, as remote agents use this URL to communicate to the server, you should take care not to specify localhost as the host name in the broker URL.

If no broker URL is found in bamboo.cfg.xml, Bamboo will default the broker URL to tcp://HOSTNAME:54663 in the bamboo.cfg.xml file, as seen in the example above. Bamboo will also append the parameter wireFormat.maxInactivityDuration=0 by default to any broker URL coming from bamboo.cfg.xml.

2. Changes to Server Configuration

JDK support

Bamboo 2.0 requires JDK 1.5 (i.e. JDK 1.4 is no longer supported). Please note that this does not affect the actual builds: it is only the Bamboo server itself that must be running JDK 1.5.

Database changes

The release of 2.0 will include some changes to column names in the database as follows:

- In the BUILD_DEFINITION table, the column XML_DATA will be changed to XML_DEFINITION_DATA
- In the BUILDRESULTSUMMARY_CUSTOMDATA table, the column CUSTOM_INFO_DATA will be changed to CUSTOM_INFO_VALUE

These fields have also had types changed to CLOB to increase their maximum lengths.

Plugins

If you are using external or custom plugins, please make sure that your plugins compile against Bamboo 2.0 before upgrading.

⚠️ We've made significant changes to the internals of the application for Bamboo 2.0. If you've installed an external plugin for 1.2.4, it's likely that it will be broken. Please take care when upgrading.

3. Changes to Build Queues and Build Plans

Bamboo 2.0 introduces the concepts of agents and capabilities. To preserve the functionality of your existing plans, JDKs, Builders and Build Queues, the following will automatically happen during the upgrade:

Conversion of Build Queues to Agents

Prior to Bamboo 2.0, you could have multiple build queues. In Bamboo 2.0, there is now only one build queue, but multiple agents (see diagram).

As part of the upgrade process,

- Each of your build queues will be converted to a local agent.
- If, prior to the upgrade, the build queue accepted builds from all plans, the agent will be given the following capability (and every plan will be given an equivalent requirement):
  - Key: bamboo.1.2.queue
  - Value: ALLOW_ANY_BUILDS
- Or if, prior to the upgrade, the build queue only accepted builds from specific plans, the agent will be given the following capability (and the relevant plans will be given an equivalent requirement):
  - Key: bamboo.1.2.queue
  - Value: <name of old queue>

If you wish to change this after the upgrade, please see 02. Configuring Agents and Capabilities and 1.2.4 Specifying a Plan’s Capability Requirements.

Conversion of Builders to Capabilities

Prior to Bamboo 2.0, your builders (e.g. Maven) were defined globally. In Bamboo 2.0, builders are now defined as agent capabilities and specified as plan requirements.
As part of the upgrade process,

- Each of your builders will be converted to a shared local capability (that is, it will apply to every local agent).
- Every plan will continue to have the same builder that it had before the upgrade.

If you wish to change this after the upgrade, please see 2.8 Configuring Capabilities and 1.2.4 Specifying a Plan's Capability Requirements.

Conversion of JDKs to Capabilities

Prior to Bamboo 2.0, your JDKs (e.g. JDK 1.5) were defined globally. In Bamboo 2.0, JDKs are now defined as agent capabilities and specified as plan requirements.

As part of the upgrade process,

- Each of your JDKs will be converted to shared local capabilities (that is, it will apply to every local agent).
- Upon conversion, the labels of each of your JDKs will upgraded to the Bamboo 2.0 JDK label format, (i.e. 'JDK 9.9.9_99').
- Upon conversion, two more generic versions of the labels will be created for each JDK, (i.e. 'JDK 9.9' and 'JDK').
- Every plan will have its requirements upgraded, to keep the association with the same JDK that it had before the upgrade.

If you wish to change this after the upgrade, please see 2.8 Configuring Capabilities and 1.2.4 Specifying a Plan's Capability Requirements.

Upgrading to Bamboo 2.0 Beta 2

💡 It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

No additional upgrade tasks are required to upgrade from Bamboo 2.0 Beta 1 to 2.0 Beta 2.

Upgrading to Bamboo 2.0 Beta 3

💡 It is strongly recommended that you back up your xml-data directory before proceeding. You are also strongly recommended to back up your database due to schema changes in this release. For full instructions please follow the Bamboo Upgrade Guide. Additionally, please note the following:

1. Changes to Repositories

Bamboo 2.0 introduces the concepts of agents and capabilities. To preserve the functionality of your existing Repositories, the following will automatically happen during the upgrade:

Conversion of Perforce P4 Client Application Location to a Capability

With the introduction of remote agents in Bamboo 2.0, the location of the Perforce P4 client application now needs to be specified as a capability. To create build plans using Perforce as repository, a shared local capability must be created for the P4 client application location. In addition, agent-specific remote capabilities must be created for each remote agent using Perforce.

As part of the upgrade process,

- A shared local Perforce capability will be created for the Perforce P4 client application location. The upgrade task reads this information from the system's environment variables. If the Perforce P4 client application location has not been specified as an environment variable, the shared local capability will need to be set up manually.

The upgrade task will not create agent-specific Perforce capabilities for any remote agents. These capabilities will need to be set up manually.

Please see 2.8.4 Configuring a new Perforce Capability for further details on creating Perforce capabilities.
Upgrading to Bamboo 2.0 Beta 4

It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

No additional upgrade tasks are required to upgrade from Bamboo 2.0 Beta 3 to 2.0 Beta 4.

Upgrading to Bamboo 2.0 Beta 5

It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

No additional upgrade tasks are required to upgrade from Bamboo 2.0 Beta 4 to 2.0 Beta 5.

Upgrading to Bamboo 2.0 Beta 6

It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

1. Changes to Perforce Workspace Management

If you use Perforce, you can now choose whether you want Bamboo to manage your workspace or whether you want to manage it yourself. Prior to this release, Bamboo would automatically manage your workspace (i.e. changed the client root). Hence, if you want to manage your workspace in this release, you will need to reset your client roots.

Upgrading to Bamboo 2.0 Beta 8

It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

No additional upgrade tasks are required to upgrade from Bamboo 2.0 Beta 6 to 2.0 Beta 8.

Upgrading to Bamboo 2.0 Beta 9

It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

No additional upgrade tasks are required to upgrade from Bamboo 2.0 Beta 8 to 2.0 Beta 9.
Atlassian Software Systems presents Bamboo 2.0

Upgrading to Bamboo 2.0 is free for all customers with active Bamboo software maintenance as at 14th April 2008. This release introduces the ability to run distributed builds. You will find this particularly useful if you need to run your builds in different geographic locations, or on different platforms. Simply install the new Bamboo Agent on your additional build servers, and your main Bamboo 2.0 server will be able to manage them. We have also provided a number of plugin points in case you need to control your distributed builds programmatically.

Highlights of Bamboo 2.0:

Distributed builds

In response to the most popular feature on your wish-list, Bamboo 2.0 introduces agents — that is, services which execute builds. These can either run on the Bamboo server (‘local agents’) or on other machines (‘remote agents’), which is particularly useful if you need to run your builds in different geographic locations, or on different platforms.

A single build queue manages the distribution of builds to appropriate agents, using capability matching (see below).

The activity of all agents can be seen on the Bamboo dashboard:

Thank you for your feedback:

- over 70 new features and improvements implemented
- over 240 votes fulfilled

Your votes and issues help us keep improving our products, and are much appreciated.
Capability matching

To enable you to control exactly which agents may execute builds for particular plans, Bamboo 2.0 introduces capability matching:

- A capability is a feature of an agent. A capability can be a:
  - builder (e.g. Maven)
  - JDK
  - custom capability (a key-value property which defines a particular characteristic of an agent, e.g. 'operating.system=WindowsXP' or 'fast.builds=true')
  - Perforce (location of the P4 client application, if Perforce is being used as the source repository)

Capabilities can be defined specifically for an agent, or they can be shared between either all local agents or all remote agents. Note that the value of an agent-specific capability overrides the value of a shared capability of the same name (if one exists). See 2.8 Configuring Capabilities.

- A requirement is an agent capability required by a build plan.

Together, capabilities and requirements control which agents can execute builds for particular plans. Each plan can only be built by agents whose capabilities meet the plan's requirements. Matching can be specified as either a regular expression or an exact match. See 1.2.4 Specifying a Plan's Capability Requirements.

For more details please see these diagrams.
Note that for ease of conversion, the Bamboo 2.0 upgrade process will automatically create appropriate agent capabilities and assign appropriate requirements to all your pre-existing build plans (see the Bamboo 2.0 Upgrade Guide).

Memory usage improvements

The underlying engine for Bamboo has been revamped to decrease memory usage. You will notice a distinct improvement in the performance of your builds, especially if you have very large logs.

Parallel VCS updates and checkouts

No more waiting! Plans can now perform checkouts and updates to your version control system in parallel, rather than serially. Hence, the time taken to run plans will be improved.

Ability to force a 'clean build'

You can now instruct Bamboo to delete the old working files and perform a new checkout of the entire source code directory, before commencing a build.

See the documentation for more details.

Quiet Period functionality supported for Subversion & Perforce

By popular request, Quiet Period parameters can now be specified for Subversion and Perforce when configuring a source repository for a build plan. You can choose to set how long Bamboo should wait after a commit before triggering a build, and the number of times it retries before initiating a build. Read more about configuring Subversion and Perforce source repositories.

Bamboo Plugin for Confluence

Atlassian brings collaboration to the next level with the introduction of the Bamboo plugin for Confluence. Here's some of the build information that your wiki users will be able to have at their fingertips:

- the most recent status of any given build plan.
- the current status of all builds in a project.
- the recent build history of a plan.
• the recent build history of a user across all projects.
• the recent build history of all plans in a project.
• Bamboo charts, including duration of builds, build failures, numbers of test, percentage of test failures and more!
Read more about the Bamboo Plugin for Confluence.

Support for Oracle and MS SQL Server databases

By popular request, Bamboo's supported databases now include Oracle and MS SQL Server.

Status Summary screens

See the status of your builds at a glance! Set up a build status monitor for your development team and display Bamboo's new status summary screens. These screens show the status of your builds in a color-coded and easy to view format.

Plus over 160 other fixes and improvements

See them here.
Upgrading from Bamboo version 1.1.x or earlier to 2.0

If you are using a version of Bamboo prior to version 1.2, you will need to upgrade Bamboo to version 1.2 before you can upgrade to version 2.0. Note that the upgrade process from version 1.0.x is different from the upgrade process from version 1.1.x. Please follow the appropriate instructions below:

Upgrading from Bamboo 1.1.x

You will need to:

1. Upgrade to Bamboo 1.2 — please see the Bamboo 1.2 Upgrade Guide.
2. Then upgrade to Bamboo 2.0, as per the 'Upgrading from Bamboo 1.2.x to 2.0' instructions below.

Upgrading from Bamboo 1.0.x

You will need to:

1. Upgrade to 1.1.2 first — please see the Bamboo 1.1.2 Upgrade Guide. (This step is necessary as there is an issue with the upgrade process from the 1.0.x series that we're currently looking into.)
2. Then upgrade to Bamboo 1.2 — please see the Bamboo 1.2 Upgrade Guide.
3. Then upgrade to Bamboo 2.0, as per the 'Upgrading from Bamboo 1.2.x to 2.0' instructions below.

It is strongly recommended that you back up your xml-data directory before proceeding. For full instructions please follow the Bamboo Upgrade Guide.

Upgrading from Bamboo 1.2.x to 2.0

1. Adding a Broker URL property.

Bamboo uses a messaging broker to communicate with its remote build agents. To ensure this works properly, a URL must be specified. This URL is where Bamboo will set up its embedded broker. Remote agents will also be provided with this URL on startup.

To specify the broker URL, please add a `bamboo.jms.broker.url` property in your `bamboo.cfg.xml` file, located inside the Bamboo home directory. For example:

```xml
<property name="bamboo.jms.broker.url">tcp://HOSTNAME:54663</property>
```

where HOSTNAME is the canonical name of your Bamboo server.

Please note, as remote agents use this URL to communicate to the server, you should take care not to specify `localhost` as the host name in the broker URL.

If no broker URL is found in `bamboo.cfg.xml`, Bamboo will default the broker URL to `tcp://HOSTNAME:54663` in the `bamboo.cfg.xml` file, as seen in the example above. Bamboo will also append the parameter `wireFormat.maxInactivityDuration=0` by default to any broker URL coming from `bamboo.cfg.xml`. 
2. Changes to Server Configuration

JDK support

Bamboo 2.0 requires JDK 1.5 (i.e. JDK 1.4 is no longer supported). Please note that this does not affect the actual builds: it is only the Bamboo server itself that must be running JDK 1.5.

Database changes

The release of 2.0 will include some changes to column names in the database as follows:

• In the BUILD_DEFINITION table, the column XML_DATA will be changed to XML_DEFINITION_DATA
• In the BUILDRESULTSUMMARY_CUSTOMDATA table, the column CUSTOM_INFO_DATA will be changed to CUSTOM_INFO_VALUE

These fields have also had types changed to CLOB to increase their maximum lengths.

Plugins

If you are using external or custom plugins, please make sure that your plugins compile against Bamboo 2.0 before upgrading.

We've made significant changes to the internals of the application for Bamboo 2.0. If you've installed an external plugin for 1.2.4, it's likely that it will be broken. Please take care when upgrading.

3. Changes to Build Queues and Build Plans

Bamboo 2.0 introduces the concepts of agents and capabilities. To preserve the functionality of your existing plans, JDks, Builders and Build Queues, the following will automatically happen during the upgrade:

Conversion of Build Queues to Agents

Prior to Bamboo 2.0, you could have multiple build queues. In Bamboo 2.0, there is now only one build queue, but multiple agents (see diagram).

As part of the upgrade process,

• Each of your build queues will be converted to a local agent.
• If, prior to the upgrade, the build queue accepted builds from all plans, the agent will be given the following capability (and every plan will be given an equivalent requirement):
  ° Key: bamboo.1.2.queue
  ° Value: ALLOW_ANY_BUILDS
• Or if, prior to the upgrade, the build queue only accepted builds from specific plans, the agent will be given the following capability (and the relevant plans will be given an equivalent requirement):
  ° Key: bamboo.1.2.queue
  ° Value: <name of old queue>

If you wish to change this after the upgrade, please see 02. Configuring Agents and Capabilities and 1.2.4 Specifying a Plan’s Capability Requirements.

Conversion of Builders to Capabilities

Prior to Bamboo 2.0, your builders (e.g. Maven) were defined globally. In Bamboo 2.0, builders are now defined as agent capabilities and specified as plan requirements.

As part of the upgrade process,

• Each of your builders will be converted to a local server capability (that is, it will apply to every local agent).
• Every plan will continue to have the same builder that it had before the upgrade.

If you wish to change this after the upgrade, please see 2.8 Configuring Capabilities and 1.2.4 Specifying a Plan’s Capability Requirements.
Conversion of JDKs to Capabilities

Prior to Bamboo 2.0, your JDKs (e.g. JDK 1.5) were defined globally. In Bamboo 2.0, JDKs are now defined as agent capabilities and specified as plan requirements.

As part of the upgrade process,

- Each of your JDKs will be converted to local server capabilities (that is, it will apply to every local agent).
- Upon conversion, the labels of each of your JDKs will upgraded to the Bamboo 2.0 JDK label format, (i.e. 'JDK 9.9.9_99').
- Upon conversion, two more generic versions of the labels will be created for each JDK, (i.e. 'JDK 9.9' and 'JDK').
- Every plan will have its requirements upgraded, to keep the association with the same JDK that it had before the upgrade.

If you wish to change this after the upgrade, please see 2.8 Configuring Capabilities and 1.2.4 Specifying a Plan's Capability Requirements.

4. Changes to Repositories

Bamboo 2.0 introduces the concepts of agents and capabilities. To preserve the functionality of your existing Repositories, the following will automatically happen during the upgrade:

Conversion of Perforce P4 Client Application Location to a Capability

With the introduction of remote agents in Bamboo 2.0, the location of the Perforce P4 client application now needs to be specified as a capability. To create build plans using Perforce as repository, a local server capability must be created for the P4 client application location. In addition, agent-specific remote capabilities must be created for each remote agent using Perforce.

As part of the upgrade process,

- A local server Perforce capability will be created for the Perforce P4 client application location. The upgrade task reads this information from the system's environment variables. If the Perforce P4 client application location has not been specified as an environment variable, the local server capability will need to be set up manually.

The upgrade task will not create agent-specific Perforce capabilities for any remote agents. These capabilities will need to be set up manually.

Please see 2.8.4 Configuring a new Perforce Capability for further details on creating Perforce capabilities.

Upgrading from Bamboo 2.0 Beta to 2.0

If you are already using the latest Bamboo 2.0 Beta, no additional upgrade tasks are required. Your Beta license key will continue to function until it expires. We encourage you to consider purchasing a license, if you wish to continue using Bamboo 2.0.
Bamboo Upgrade Guide

This page last changed on Feb 12, 2008 by asridhar.

⚠ Before you begin
Please read the Release Notes and Upgrade Guides for the version you are upgrading to.

Step 1. Identify your Bamboo directories

Go to the ‘System Info’ page in the ‘Administration’ menu of your Bamboo instance and note the location of the Bamboo Home, Build Data Path and Configuration Path directories:

Step 2. Shut down Bamboo

You need to shut down Bamboo before backing up or performing the upgrade.

Step 3. Back up Bamboo

Back up the three directories you identified in Step 1 — your Bamboo Home, Build Data Path and Configuration Path directories (for more information about these directories please see Important Directories and Files).

Note that it is not necessary to back up your Working Directory (located inside your Bamboo Home directory).

Step 4. Re-install Bamboo

⚠ Please note:

- When specifying the \{BAMBOO_HOME\} directory, use the same Bamboo Home directory as in your old installation. That is, specify the same directory and path as the Bamboo Home directory shown in ‘Step 1’ (above).
- Make sure that your \{BAMBOO_INSTALL\} directory is either a new directory, or else delete your old \{BAMBOO_INSTALL\} directory before you begin, as legacy files may cause problems.
- The \{BAMBOO_HOME\} directory must be different from the \{BAMBOO_INSTALL\} directory. This will ensure that your data is not lost when upgrading or re-installing Bamboo.

Follow steps 1 and 2 of the installation instructions for your operating system:

- Bamboo Standalone Installation Guide — Windows
- Bamboo Standalone Installation Guide — Linux
- Bamboo Standalone Installation Guide — Mac
- Bamboo EAR-WAR Installation Guide

Step 5. Start Bamboo

Once you have installed Bamboo and set the ‘bamboo.home’ property (as described in the Installation Guides), start Bamboo. The upgrade process will be performed when Bamboo starts up. You will not see the Setup Wizard.

Monitor the atlassian-bamboo.log to ensure that the upgrade process has completed successfully.
Step 6. Re-index Bamboo (if indicated in release notes)

Bamboo maintains an index of its build results. This allow Bamboo to display aggregate build results information across builds. You may need to perform a re-index of Bamboo if the upgrade process requires it. This step may or may not be required (depending on the upgrade versions). Also note that you only need to do this if you have existing data in Bamboo.

To re-index, go to 'Administration', then 'Indexing', and click the 'Reindex' button.

⚠️ Depending on the number of builds and tests you may have, the indexing process may take a significant amount of time. During this period, Bamboo will not be available. Also, it is advisable to ensure that all you have disabled all build queues (or all agents, if you are upgrading from Bamboo 2.0 or later), and that no builds are in progress when you start the re-indexing process. If you have a large instance, it is recommended that you reindex overnight.

Step 7. Update plugins

If you are using any plugins other than the ones that ship with Bamboo, check that each one is compatible with the new version of Bamboo. Upgrade any plugins that are out-of-date, and disable any plugins that are incompatible with your new version of Bamboo.

Step 8. Re-configure external user repositories (if applicable)

- LDAP integration — If you had previously integrated Bamboo with LDAP/AD, copy your old `<Bamboo-install>/webapps/WEB-INF/classes/atlassian-user.xml` file to the new Bamboo installation.
- Crowd integration — If you had previously integrated Bamboo with Crowd, you will need to re-enable Crowd integration. For details please see integrating Crowd with Bamboo.

Troubleshooting

If you have any problems during upgrade, please raise a support request at https://support.atlassian.com/ and attach your `atlassian-bamboo.log` so we can help you find out what's gone wrong.
Bamboo Knowledge Base

<table>
<thead>
<tr>
<th>Answers to commonly raised questions about configuring and using Bamboo:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Installation FAQ</strong></td>
</tr>
<tr>
<td>• Adding MIME types to Bamboo Standalone</td>
</tr>
<tr>
<td>• Bamboo 1.2 on Tomcat 5</td>
</tr>
<tr>
<td>• Can I have Bamboo.home pointing to a resource on a network share?</td>
</tr>
<tr>
<td>• Changing Bamboo's port from the default 8085</td>
</tr>
<tr>
<td>• Changing the Root Context Path</td>
</tr>
<tr>
<td>• Configuring Bamboo on start-up</td>
</tr>
<tr>
<td>• Getting Bamboo Standalone to use the jetty.xml file</td>
</tr>
<tr>
<td>• Hardware sizing considerations</td>
</tr>
<tr>
<td>• Installation notes for Bamboo on JBoss 4.x</td>
</tr>
<tr>
<td>• Moving Bamboo between machines.</td>
</tr>
<tr>
<td>• Running Bamboo over HTTPS</td>
</tr>
<tr>
<td>• Setting up JNDI on Jetty</td>
</tr>
<tr>
<td>• Supported databases</td>
</tr>
<tr>
<td><strong>Usage FAQ</strong></td>
</tr>
<tr>
<td>• Backing up Bamboo instances over 4GB</td>
</tr>
<tr>
<td>• Binding Bamboo to one IP address</td>
</tr>
<tr>
<td>• Can Bamboo build and test non-Java projects?</td>
</tr>
<tr>
<td>• Can multiple plans share a common 3rd-party directory? — For example, you might have three repository directories, say, A, B, and C, where A is specific to one project, and B and C are common across many projects.</td>
</tr>
<tr>
<td>• Connecting to Subversion repositories</td>
</tr>
<tr>
<td>• Deactivating a Bamboo user</td>
</tr>
<tr>
<td>• Fixing failing Bamboo builds, with OutOfMemory errors</td>
</tr>
<tr>
<td>• Fixing OutOfMemory Errors in Bamboo</td>
</tr>
<tr>
<td>• Hibernate errors in logs after upgrading to Bamboo 2.0</td>
</tr>
<tr>
<td>• JUnit parsing in Bamboo</td>
</tr>
<tr>
<td>• Logging in Bamboo</td>
</tr>
<tr>
<td>• Performing a thread dump.</td>
</tr>
<tr>
<td>• Problems running Bamboo under Sun JDK 1.4</td>
</tr>
<tr>
<td>• Removing Coverage plug-in data from the Bamboo database</td>
</tr>
<tr>
<td>• Restoring passwords to recover admin users</td>
</tr>
<tr>
<td>• Testing LDAP or Active Directory connectivity with Paddle</td>
</tr>
<tr>
<td>• Troubleshooting an SVN connection in Bamboo</td>
</tr>
<tr>
<td>• Using Bamboo with Clover</td>
</tr>
<tr>
<td>• Working with Sun JAVA libraries</td>
</tr>
</tbody>
</table>

⚠️ Do you have a question, or need help with Bamboo? Please create a support request.

You may also like to check out the forums:

• Bamboo Announcements
• Bamboo General Forum
• Bamboo Developers Forum
Installation FAQ

This page last changed on Jun 07, 2007 by rosie@atlassian.com.

- Adding MIME types to Bamboo Standalone
- Bamboo 1.2 on Tomcat 5
- Can I have Bamboo.home pointing to a resource on a network share?
- Changing Bamboo's port from the default 8085
- Changing the Root Context Path
- Configuring Bamboo on start-up
- Getting Bamboo Standalone to use the jetty.xml file
- Hardware sizing considerations
- Installation notes for Bamboo on JBoss 4.x
- Moving Bamboo between machines
- Running Bamboo over HTTPS
- Setting up JNDI on Jetty
- Supported databases
Adding MIME types to Bamboo Standalone

Bamboo Standalone ships with the Jetty application server (see Bamboo 'distributions'). To add additional MIME content types, edit the `mimetypes.xml` file in `../<Bamboo-install>/webapps/WEB-INF/mimetypes.xml` and insert your new mime-mapping.

For instance to add an additional text/plain mime type with txt2 extension, insert the following:

```
<mime-mapping>
  <extension>txt2</extension>
  <mime-type>text/plain</mime-type>
</mime-mapping>
```

Restart Bamboo, for the changes to take effect.
If you are upgrading to Bamboo 1.2.2 on Tomcat 5, the endorsed XML libraries (Specifically, 'xml-apis.jar' and 'xercesImpl.jar') in "/<Tomcat_install>/common/endorsed/" get loaded before Bamboo starts-up, which results in the exception below, as Bamboo uses the wrong XML libraries.

```
javax.xml.transform.TransformerFactoryConfigurationError: Provider org.apache.xalan.processor.TransformerFactoryImpl not found
    at javax.xml.transform.TransformerFactory.newInstance(Unknown Source)
    at com.opensymphony.xwork.util.DomHelper$DOMBuilder.<clinit>(DomHelper.java:168)
    at com.opensymphony.xwork.util.DomHelper.parse(DomHelper.java:116)
    at com.opensymphony.xwork.config.providers.XmlConfigurationProvider.loadConfigurationFile(XmlConfigurationProvider.java:91)
    at com.opensymphony.xwork.config.providers.XmlConfigurationProvider.init(XmlConfigurationProvider.java:55)
    at com.opensymphony.xwork.DefaultActionProxy.<init>(DefaultActionProxy.java:60)
    at com.opensymphony.webwork.dispatcher.DispatcherUtils.serviceAction(DispatcherUtils.java:216)
    at com.opensymphony.webwork.dispatcher.FilterDispatcher.doFilter(FilterDispatcher.java:202)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:186)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:157)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:186)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:157)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:186)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:157)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:186)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:157)
    at com.atlassian.bamboo.filter.BambooAcegiProxyFilter.doFilter(BambooAcegiProxyFilter.java:21)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:186)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:157)
    at com.atlassian.spring.filter.FlushingSpringSessionInViewFilter.doFilterInternal(FlushingSpringSessionInViewFilter.java:26)
    at org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:77)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:186)
```

To fix this issue:

1. Stop your Tomcat5 application server.
2. Remove the 'xml-apis.jar' and 'xercesImpl.jar' from "./<Tomcat_install>/common/endorsed/".
3. Start Tomcat5, and upgrade to 1.2.2.

The error above seems to be persistent with Bamboo 1.2.2, and doesn't affect any other versions,

Should you have problems upgrading, please raise a support request at https://support.atlassian.com/
Can I have Bamboo.home pointing to a resource on a network share?

Yes, it is possible to point Bamboo.home to a directory on your Windows network share. However, you need to specify the absolute path to the network share (e.g. `\\Network_computer\path_to_director`).

⚠️ Note:

Please ensure that the user which Bamboo is running as has sufficient privileges to access the network resource.
Changing Bamboo's port from the default 8085

To configure Bamboo to start on a port other than 8085, please see Configuring Bamboo on start-up.

Note: If you are running Bamboo on Linux, you need to start Bamboo on a privileged port (0-1024). You also need to run the Bamboo process as root.
Changing the Root Context Path

When running Bamboo behind a proxy, you might need to change the Root Context Path i.e. the host URL referenced while accessing Bamboo (e.g. http://localhost:8085/bamboo).

To change the context path from '/' to '/Your_Context_Path':

- If you are using the bamboo.sh script to start Bamboo:
  Change the following line in your bamboo.sh script:

  ```
  RUN_CMD="java -server -Xms256m -Xmx512m -XX:MaxPermSize=256m -Djava.awt.headless=true -classpath $CLASSPATH -Dorg.mortbay.xml.XmlParser.NotValidating=true -Djetty.port=8085 com.atlassian.bamboo.server.Server 8085 ./webapp /
  
  to:
  
  RUN_CMD="java -server -Xms256m -Xmx512m -XX:MaxPermSize=256m -Djava.awt.headless=true -classpath $CLASSPATH -Dorg.mortbay.xml.XmlParser.NotValidating=true -Djetty.port=8085 com.atlassian.bamboo.server.Server 8085 ./webapp /Your_Context_Path"
  ```

- Or, if you are using the wrapper to start Bamboo:
  The wrapper reads the configuration information from the wrapper.conf file in the ../<Bamboo-Install>/conf/ folder. Find the following line:

  ```
  wrapper.app.parameter.4=/
  ```

  Replace it with the following line:

  ```
  wrapper.app.parameter.4=/Your_Context_Path
  ```
Configuring Bamboo's start-up parameters under Linux

Bamboo on Linux/Unix can be started by either executing the bamboo.sh script or using the wrapper. Either way, the Bamboo server can be customised at start-up.

Modifying the bamboo.sh script.

The bamboo.sh script takes four parameters: start|stop|restart|status.

To customise these parameters at startup, edit the $RUN_CMD variable:

```
# This is how the Bamboo server will be started
# RUN_CMD="java -Xms256m -Xmx512m -Djava.awt.headless=true -classpath $CLASSPATH -Dorg.mortbay.xml.XmlParser.NotValidating=true -Djetty.port=8085 com.atlassian.bamboo.server.Server 8085 ./webapp /
```

- java -Xms256m -Xmx512m specifies the minimum and maximum Java Heap size.
- -classpath $CLASSPATH sets the class path at startup.
- -Djetty.port=8085 specifies the port number for the Jetty server.
- com.atlassian.bamboo.server.Server 8085 ./webapp / is the main class that will be executed followed by the context path.

In some cases it might be useful to increase the PermGen space. To do this, add the following parameter to the RUN_CMD variable: "XX:MaxPermSize=512m". This will set the PermGen space to 512mb next time Bamboo is run.

Modifying the wrapper.

The wrapper reads the configuration from wrapper.conf found in ../<BAMBOO_INSTALL>/conf. (The properties are documented inside the file.)

Configuring Bamboo's start-up parameters under Windows

Bamboo can be started in Windows with the startup.bat file (from the command line) or as a Windows Service. Both use the wrapper to start Bamboo. As in Linux (see above), the wrapper reads the configuration from wrapper.conf. Please edit the ...(wrapper)/wrapper.conf file (situated in the root of your Bamboo_Installation directory) as required.
Getting Bamboo Standalone to use the jetty.xml file

By default Bamboo doesn't use the jetty.xml file to configure itself. If you need to modify the jetty.xml for advanced configuration (such as JNDI or https), you will also need to tell Bamboo to use it.

The method for doing this depends on whether you are using the bamboo.sh startup script or the Java Service Wrapper.

Step 1 - Instructing Bamboo to use jetty.xml

If you are using the **bamboo.sh** script to start Bamboo:

The standard Bamboo startup script can be customised to use the jetty.xml file by modifying the following section in your bamboo.sh script (this section specifies how the Bamboo server will start):

```java
RUN_CMD="java -Xms256m -Xmx512m -Djava.awt.headless=true -classpath $CLASSPATH -Dorg.mortbay.xml.XmlParser.NotValidating=true -Djetty.port=8085 com.atlassian.bamboo.server.Server 8085 ./webapp /

Now, modify this startup script to read the jetty.xml file from webapp/WEB-INF/classes/jetty.xml by changing the RUN_CMD argument as follows:

```java
RUN_CMD="java -Xms256m -Xmx512m -Djava.awt.headless=true -classpath $CLASSPATH -Dorg.mortbay.xml.XmlParser.NotValidating=true com.atlassian.bamboo.server.Server webapp/WEB-INF/classes/jetty.xml
```

If you are using the Java Service Wrapper to start Bamboo:

When starting up Bamboo with the Java Service Wrapper, you'll need to modify the wrapper.conf file in the conf directory.

- You will need to replace the argument which specifies your port number "wrapper.app.parameter.2=8085" with "wrapper.app.parameter.2=../webapp/WEB-INF/classes/jetty.xml".
- You will need to comment out the other arguments: "wrapper.app.parameter.3=../webapp" and "wrapper.app.parameter.4=/"

This will make Bamboo start up using your jetty.xml configuration file instead of the default three arguments (port, web app directory, context path).

Step 2 - Setting root context web application in jetty.xml

Edit the webapp root context in your jetty.xml file situated in `<Bamboo-install>/webapp/WEB-INF/classes/jetty.xml`. From

```xml
<Call name="addWebApplication">
    <Arg>/bamboo</Arg>
    <Arg>
        <SystemProperty name="bamboo.webapp" default="/bamboo-web-app/src/main/webapp"/>
    </Arg>
</Call>
</Configure>
```

To

```xml
<Call name="addWebApplication">
    <Arg>/bamboo</Arg>
    <Arg>
        <SystemProperty name="bamboo.webapp" default="full/path/to/bamboo/install/directory/webapp"/>
    </Arg>
</Call>
```
Hardware sizing considerations

This page last changed on Nov 01, 2007 by rosie@atlassian.com.

}\Also see the Bamboo System Requirements.

For Bamboo, the minimum hardware requirements depend on the size and complexity of your plans. Considerations include:

1. Do your builds will have functional tests as part of the plans?
2. Are your plans executed simultaneously? If so, how many plans will be running at any given time?
3. What are the requirements for your running builds, e.g. do they need large amounts of memory/disk/swap space?
4. How many users will be using Bamboo at any given time? Like any web application, the system resource needed is proportional to the load experienced by the server.
Installation notes for Bamboo on JBoss 4.x

This page is for people who are deploying the Bamboo EAR/WAR edition on the JBoss 4.x application server.

⚠️ For full installation instructions please see the Bamboo EAR-WAR Installation Guide.

File extraction notes

To deploy Bamboo EAR-WAR onto your JBoss application server, copy the Bamboo WAR file to ../<JBoss-install>/server/default/deploy/atlassian-bamboo-1.1.2.war.

By default the WAR file will extract to atlassian-bamboo-<version>. The name of the directory in the webapps folder will form the URL required to access Bamboo, e.g. <JBoss-install>/server/default/deploy/atlassian-bamboo-1.1.2.war will become http://host:port/atlassian-bamboo-1.1.2/

How to set Java OPTs on JBoss 4.x

- Windows:
  1. Find the run.bat file.
  2. Edit JAVA_OPTS to set the desired properties variable:

```
FROM
if exist "%JBOSS_HOME%\bin\native" set JAVA_OPTS=%JAVA_OPTS% -Djava.library.path=%JBOSS_HOME% \bin\native

TO
if exist "%JBOSS_HOME%\bin\native" set JAVA_OPTS=%JAVA_OPTS% -Djava.library.path=%JBOSS_HOME%\bin\native -server -XX:MaxPermSize=256m -Dbamboo.home=/opt/bamboo/bamboohome -Xmx512m -Djava.awt.headless=true -Dbamboo.home=Your_Path_To_Bamboo
```

- Linux-based systems:
  1. Find the run.sh file
  2. Edit JAVA_OPTS to set the desired properties variable:

```
FROM
# Setup JBoss sepecific properties
JAVA_OPTS="-Dprogram.name=$PROGNAME $JAVA_OPTS"

TO
# Setup JBoss specific properties
JAVA_OPTS="-Dprogram.name=$PROGNAME -server -XX:MaxPermSize=256m -Dbamboo.home=/opt/bamboo/bamboohome -Xmx512m -Djava.awt.headless=true -Dbamboo.home=Your_Path_To_Bamboo $JAVA_OPTS"
```

⚠️ For further reference
Please visit the JBoss Wiki page on setting JavaOpt
Moving Bamboo between machines.

To migrate Bamboo between machines,

1. Back up your current Bamboo instance as per our Bamboo backup guide.
2. Login to My Atlassian portal and retrieve your Bamboo license.
3. Download a fresh distribution of Bamboo from the Bamboo download center to your new machine.
   
   **Tip** If you are using the WAR version, click the 'Show all' link on the Bamboo download center page to display the download link.

4. Unpack your Bamboo distribution on your new machine, (if you are using the WAR version, you need to deploy it to your application server) - See our installation guide.
5. Start your new Bamboo installation and run the Setup Wizard.
6. In step (4) of the Setup Wizard, import your old Bamboo instance from step (1).
Running Bamboo over HTTPS

This page last changed on Jan 23, 2008 by alui.

This document is a guide to configuring Bamboo Standalone with basic HTTPS authentication. For further reference please visit the Jetty page on [configuring SSL with Jetty](http://www.eclipse.org/jetty/doc/).

1. Generating a certificate with the JDK keytool

The simplest way to generate keys and certificates is to use the keytool application that comes with the JDK, as it generates keys and certificates directly into the keystore.

The following command will generate a key pair and certificate directly into a keystore:

```
keytool -keystore keystore -alias jetty -genkey -keyalg RSA
```

This command will prompt for information about the certificate and for passwords to protect both the keystore and the keys within it. The only mandatory response is to provide the fully qualified host name of the server at the "first and last name" prompt.

Now, we need to configure an SSL listener.

2. Configuring Jetty

Using the Sun JVM, add the SunJsseListener as a HttpListeners, In the ../<Bamboo_Application_Directory>/webapp/WEB-INF/classes/jetty.xml file add the following lines:

```
<Call name="addListener">
  <Arg>
    <New class="org.mortbay.http.SunJsseListener">
      <Set name="Port">8443</Set>
      <Set name="Keystore"><SystemProperty name="jetty.home" default="."/>/keystore</Set>
      <Set name="Password">password</Set>
      <Set name="KeyPassword">password</Set>
    </New>
  </Arg>
</Call>
```

This will make Bamboo accessible in port 8443 on [https://localhost:8443/](http://https://localhost:8443/)

The keystore file in this example is given relative to the Bamboo Application Directory.

Please ensure that jcert.jar, jnet.jar and jsse.jar are on your classpath.

3. Getting Bamboo to use the jetty.xml file

Follow [this Knowledge Base article](http://issues.redbeams.com/), to instruct Bamboo to use the jetty.xml file configured in step 2.
Setting up JNDI on Jetty

The Bamboo start up script can be customised to setup JNDI resources
Find the following section in your Bamboo.sh script. (This section specifies how the Bamboo server will start)

```
#       This is how the Bamboo server will be started
#
RUN_CMD="java -Xms256m -Xmx512m -Djava.awt.headless=true -classpath $CLASSPATH -Dorg.mortbay.xml.XmlParser.NotValidating=true -Djetty.port=8085 com.atlassian.bamboo.server.Server 8085 ./webapp /
```

Now, modify this startup script to read the jetty.xml file from webapp/WEB-INF/classes/jetty.xml, which can be used to specify JNDI resources. Modify the RUN_CMD argument as below

```
RUN_CMD="java -Xms256m -Xmx512m -Djava.awt.headless=true -classpath $CLASSPATH -Dorg.mortbay.xml.XmlParser.NotValidating=true webapp/WEB-INF/classes/jetty.xml
```

You will also need to change the jetty.xml file under webapp/WEB-INF/classes by change the context path from /bamboo to /. Example of this is below:

```
<!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
<!-- Add root context web applications. -->
<!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
<Call name= "addWebApplication">
  <Arg></Arg>
  <Arg>
    <SystemProperty name= "bamboo.webapp" default= "webapp"/>
  </Arg>
</Call>
```

To set up the JNDI mail session, you will also need to uncomment and modify the section of this jetty.xml which looks like this:

```
Supported databases

Bamboo is bundled with a pre-configured HSQL database for evaluation purposes only. We recommend migrating to a external database, if Bamboo is to be used in production. For a list of supported databases, please see 2. Connecting Bamboo to an external database.
Usage FAQ

This page last changed on Jun 07, 2007 by rosie@atlassian.com.

- Backing up Bamboo instances over 4GB
- Binding Bamboo to one IP address
- Can Bamboo build and test non-Java projects?
- Can multiple plans share a common 3rd-party directory?
- Connecting to Subversion repositories
- Deactivating a Bamboo user
-Fixing failing Bamboo builds, with OutOfMemory errors
- Fixing OutOfMemory Errors in Bamboo
- Hibernate errors in logs after upgrading to Bamboo 2.0
- JUnit parsing in Bamboo
- Logging in Bamboo
- Performing a thread dump.
- Problems running Bamboo under Sun JDK 1.4
- Removing Coverage plug-in data from the Bamboo database
- Restoring passwords to recover admin users
- Testing LDAP or Active Directory connectivity with Paddle
- Troubleshooting an SVN connection in Bamboo
- Using Bamboo with Clover
- Working with Sun JAVA libraries
Backing up Bamboo instances over 4GB

Due to limitations of the original ZIP file format, and the TrueZIP library used to generate ZIP files, it is not possible to export a Bamboo instance when the resulting ZIP file, or the original size of any of its components, is larger than 4GB. Instead, you will need to backup Bamboo manually. We strongly recommend performing regular backups.

To backup Bamboo manually:

1. Shut down Bamboo.
2. Copy the contents of your Bamboo-Home directory.
3. If you are using an external database, use the database's native backup tool to backup your database (please consult your database documentation for further instructions). Alternatively, perform an SQL dump of your database.

To restore your Bamboo instance to a previous state:

1. Edit the ..Bamboo-Install-Directory/webapps/WEB-INF/bamboo.init.properties file to point to your backed-up Bamboo-Home directory.
2. If you are using an external database, restore your database using the database's native backup tool.
Binding Bamboo to one IP address

These instructions apply to Bamboo Standalone, which ships with the Jetty application server.

If you have installed Bamboo on a machine with multiple interfaces, and need to bind Bamboo to a single IP address, follow these instructions.

Step 1 — Instruct Bamboo to read its configuration from the jetty.xml file

By default Bamboo doesn't use the jetty.xml file to configure itself. You will need to tell Bamboo to use it.

Step 2 — Edit the jetty.xml file

Your jetty.xml file is located in <Bamboo_Install_directory>/webapp/WEB-INF/classes/jetty.xml. Find the following section:

```xml
<Call name="addListener">
  <Arg>
    <New class="org.mortbay.http.SocketListener">
      <Set name="Port">
        <SystemProperty name="jetty.port" default="8085"/>
      </Set>
      <Set name="Host">127.0.0.1</Set>
    </New>
  </Arg>
</Call>
```

Change the last line as follows:

```xml
<Call name="addListener">
  <Arg>
    <New class="org.mortbay.http.SocketListener">
      <Set name="Port">
        <SystemProperty name="jetty.port" default="8085"/>
      </Set>
      <Set name="Host">YOUR_HOST_URL</Set>
    </New>
  </Arg>
</Call>
```

Please note: YOUR_HOST_URL should be the same as the Bamboo base URL configured in Bamboo.

Step 3 — Restart Bamboo
Can Bamboo build and test non-Java projects?

Bamboo can be ported to be used on any architecture and can build projects in virtually any language/script (Java, C++, ruby, perl, VB.net, bash, make and C# to name a few of many projects currently built with Bamboo).

Bamboo can execute any script/build that has a return code after the build process is completed. Ideally, you would configure a build tool (such as Maven or Ant) to build your code. Bamboo will then call on the build tool to build your project (depending on how your build process is configured).

Regarding tests, Bamboo uses JUnit tests to integrate test results with Java and is capable of reading test results from any testing framework that outputs to a Junit XML report.
Can multiple plans share a common 3rd-party directory?

For example, you might have three repository directories, say, A, B, and C, where A is specific to one project, and B and C are common across many projects. At this stage, Bamboo doesn't support having multiple checkout directories per build plan. However, you can work around this by setting these three directories up as separate Bamboo build plans, with B and C both being dependant on A (see 3.4 Triggering a Build when another Build finishes). This ensures that B and C will both build if you check-in source changes against A.

To make this work, you will also need to specify as an argument to your build scripts for B and C the location of A, which will be something like this: 
../A/

Using a set up like this, your library module (A) should only be checked out once across the Bamboo instance.
Connecting to Subversion repositories

Note:
Bamboo 1.0.4 does not authenticate with Subversion. Please upgrade Bamboo to 1.0.5 or above (we recommend upgrading to the latest version of Bamboo).

Cannot validate Subversion repository during plan configuration

The most common repository error is:

```
This is not a valid Subversion Repository: svn: Authentication required for '<https://svn.atlassian.com:443&gt; Atlassian Subversion Repository'
```

In all likelihood, the authentication failed due to invalid login credentials. Please double-check your username and password and try again.

If you are still having trouble connecting to repository,

1. Try logging in outside of Bamboo, from the command line or from any other repository client. If this step fails, then it's likely that your repository is not configured correctly.
2. Please the consult our [JIRA issue tracker](http://support.atlassian.com) for known Bamboo repository issues.
3. If you still can't connect to Subversion, please [click here](http://support.atlassian.com) to raise a support request. We will get back to you ASAP.
Deactivating a Bamboo user

To deactivate a Bamboo user account (rather than deleting it), change the password so that the user cannot login. To do this,

1. Click the 'Administration' link in the top navigation bar.
2. Click the 'Users' link in the left navigation column.
3. The 'Manage Users' screen will be displayed. Locate the relevant user in the list, and click the corresponding 'Edit' link in the 'Operations' column
4. The 'User Details' screen will be displayed. Type a new password in the 'Password' and 'Confirm Password' fields.
   - If you have configured SMTP email on your Bamboo server, the user will automatically receive an email containing their new password.
5. To get around the email problem, in the 'Email' field, enter an invalid email address. E.g. foobar@foobaremailaddress.foobar
6. Delete the users, Jabber Address so that he does not receive Jabber notifications on build events.
7. Click on the Save button.
Fixing failing Bamboo builds, with OutOfMemory errors

If your Maven/Ant builds are failing with OutOfMemory errors, this means there isn't enough memory assigned to your build. To fix this error, you need to increase the memory allocated to your build(s). To do this, we need to edit the plans build configuration and increase the memory allocated for your build.

If you are using Maven, to build your project.

Edit your plan's build configuration, and add the following variable to the 'System Environment Variables' field in your plan's build configuration page.

```
MAVEN_OPTS=-Xmx512m
```

If you are using Maven, to build your project.

Edit your plan's build configuration, and add the following variable to the 'System Environment Variables' field in your plan's build configuration page.

```
ANT_OPTS=-Xmx512m
```

Note: You may wish to change the memory allocation from 512mb to fit your plan's requirements.
Fixing OutOfMemory Errors in Bamboo

I am getting 'Out of Memory' errors. How can I allocate more memory to Bamboo?

Since the default memory setting usually is 256MB in Bamboo, you might have to adjust the settings to run a bigger Bamboo instance with sufficient memory.

On Linux:

- In the unpacked Bamboo Standalone directory, edit the file `bamboo.sh`
- Edit the line beginning with `RUN_CMD=` substituting new values for `-Xms` (starting memory) and `-Xmx` (maximum memory)
- Leave the rest of the options in that line unchanged.

Here is an example of a minimal setting for a large system (maximum heap size is set to 768 megabytes):

```
RUN_CMD="java -server -Xms512m -Xmx768m -XX:MaxPermSize=256m -Djava.awt.headless=true \
-classpath $CLASSPATH -Dorg.mortbay.xml.XmlParser.NotValidating=true -Djetty.port=8085 \
com.atlassian.bamboo.server.Server 8085 ./webapp /
"
```

On Windows:

Bamboo uses a wrapper to start, either as a service or in a console.

- The wrapper reads the configuration from the `wrapper.conf` file, which is found in `BAMBOO_INSTALL/conf` (The parameters are documented inside the file.)
- Edit the `-Xms` value to increase your Bamboo memory.

Other Notes

Allocating too much memory to your JVM Heap can also cause OutOfMemory Errors.

The error `java.lang.OutOfMemoryError: unable to create new native thread` occurs when the operating system is unable to create new threads. This is due to the JVM Heap taking up the available RAM. Big heaps take away from the space that can be allocated for the stack of a new thread. For Linux the maximum heap size of the JVM cannot be greater than 2GB. If you only have 2GB RAM in your server, it is not recommended to set the Max size of the JVM that high. The size of the stack per thread can also contribute to this problem. The stack size can reduce the number of threads that can be created.

To fix this problem, you should reduce the size of your JVM Heap and also the size of the stack per thread. The stack size can be changed with the following (example) parameter:

```
-Xss512k
```

Please refer to the following guide as a reference for JVM tuning: [http://goobsoft.homeip.net/Wiki.jsp?page=JavaDebianTuning](http://goobsoft.homeip.net/Wiki.jsp?page=JavaDebianTuning).

Permanent Generation Size

If you get the error message: `java.lang.OutOfMemoryError: PermGen space` this means that you have exceeded Java's fixed 64Mb block for loading class files. You will need to add the argument `-XX:MaxPermSize` and increase the memory.

- JDK 1.4 does not provide information as to why the OutOfMemory error occurred.
- JDK 1.5 and above are recommended as they provide a description of the error as in the above example.
Hibernate errors in logs after upgrading to Bamboo 2.0

If you are upgrading to Bamboo 2.0 from Bamboo 1.2.4 by pointing to your Bamboo-Home, you may see the following errors in your logs.

```
2008-02-21 09:13:39,891 ERROR [main] [SchemaUpdate] Constraint already exists in statement
  \{alter table USER_COMMIT add constraint FKF8936C2BA958B29F foreign key (BUILDRESULTSUMMARY_ID) references BUILDRESULTSUMMARY\}
2008-02-21 09:13:39,892 ERROR [main] [SchemaUpdate] Unsuccessful: alter table USER_COMMIT add constraint FKF8936C2BFEDC684F foreign key (AUTHOR_ID) references AUTHOR
2008-02-21 09:13:39,892 ERROR [main] [SchemaUpdate] Constraint already exists in statement
  \{alter table USER_COMMIT add constraint FKF8936C2BFEDC684F foreign key (AUTHOR_ID) references AUTHOR\}
2008-02-21 09:13:39,894 ERROR [main] [SchemaUpdate] Constraint already exists in statement
  \{alter table USER_COMMENT add constraint FK19DA09CBA958B29F foreign key (BUILDRESULTSUMMARY_ID) references BUILDRESULTSUMMARY\}
```

In Bamboo 2.0 we introduced compatibility with Oracle and MS SQL Server, this meant we the up had to make a few changes to the Bamboo Database schema and as a side-affect of these changes - you might notice hibernate warnings above in your atlassian-bamboo logs while Bamboo starts up. These warnings will not prevent your instance of Bamboo from working correctly, but will display every time Bamboo is started.

If you do wish to remove these warning messages, follow the steps below:

1. **Perform an export** from your current Bamboo 2.0 instance.
2. Re-install Bamboo 2.0 on a fresh instance.
3. Import the old exported data into your new Bamboo 2.0 instance.

This will remove the Hibernate Error messages while Bamboo starts up.
JUnit parsing in Bamboo

Bamboo can parse any test output that conforms to standard JUnit XML format. The implementation of this is pretty simple — Bamboo looks for specific tags in the JUnit XML output. Here is a simple JUnit output that is successfully parsed by Bamboo.

```xml
<?xml version= "1.0" encoding= "UTF-8" ?>
<testsuite errors= "0" tests= "3" time= "0.391" failures= "1"
   name="com.atlassian.bamboo.repository.perforce.PerforceSyncCommandTest">
   <properties>
     <property value= "Java(TM) 2 Runtime Environment, Standard Edition" name= "java.runtime.name"/>
     <property value= "UnicodeBig" name= "sun.io.unicode.encoding"/>
     ............
   </properties>
   < testcase time= "0.001" name= "testGeneratesCorrectP4CommandLine" />
   < testcase time= "0" name= "testGettersReturnExpectedStuff" />
   < testcase time= "0.164" name= "testUsingPerforceWhenNoFilesHaveChanged">
     < failure type= "junit.framework.AssertionFailedError" message= "Should not have any errors. [Perforce client error:, Connect to server failed; check $P4PORT., TCP connect to keg failed., keg: host unknown.]">
       expected:&amp;gt;0; but was:&amp;gt;4
     </ failure>
   </ testcase>
   < test>
     at com.atlassian.bamboo.repository.perforce.PerforceSyncCommandTest.testUsingPerforceWhenNoFilesHaveChanged(PerforceSyncCommandTest.java:60)
     at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
     at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
     at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
     at java.lang.reflect.Method.invoke(Method.java:585)
     at junit.framework.TestCase.runTest(TestCase.java:154)
     at junit.framework.TestCase.runBare(TestCase.java:127)
     at junit.framework.TestResult$1.protect(TestResult.java:109)
     at junit.framework.TestCase.run(TestCase.java:118)
     at junit.framework.TestSuite.runTest(TestSuite.java:208)
     at junit.framework.TestSuite.run(TestSuite.java:203)
     at sun.reflect.GeneratedMethodAccessor17.invoke(Unknown Source)
     at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
     at java.lang.reflect.Method.invoke(Method.java:585)
     at org.apache.maven.surefire.battery.JUnitBattery.executeJUnit(JUnitBattery.java:242)
     at org.apache.maven.surefire.battery.JUnitBattery.executeJUnit(JUnitBattery.java:216)
     at org.apache.maven.surefire.Surefire.executeBattery(Surefire.java:213)
     at org.apache.maven.surefire.Surefire.run(Surefire.java:163)
     at org.apache.maven.surefire.Surefire.run(Surefire.java:87)
     at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
     at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
     at java.lang.reflect.Method.invoke(Method.java:585)
     at org.apache.maven.surefire.booter.SurefireBooster.runTestsInProcess(SurefireBooster.java:313)
     at org.apache.maven.surefire.booter.SurefireBooster.run(SurefireBooster.java:221)
     at org.apache.maven.test.SurefirePlugin.execute(SurefirePlugin.java:371)
     at org.apache.maven.plugin.DefaultPluginManager.executeMojo(DefaultPluginManager.java:412)
```
Click here for the AntXmlResultParser.java file which contains the Bamboo code for parsing JUnit XML output.
Logging in Bamboo

Overview

There are two distinct logs generated by Bamboo:

- **build logs** — The build logs are generated each time a plan is executed. All information specific to the build is stored in these logs. The build logs are located in the `xml-data/builds/` sub-directories. The build logs can be downloaded as an artifact from the [Viewing a Build's Artifacts] page in Bamboo.

- **atlassian-bamboo logs** — Bamboo records all server activity in these logs. The atlassian-bamboo logs can be found either in the root `<Bamboo-Install>` directory or the directory you started Bamboo from. If you are running Bamboo as a service on windows, the logs are generated in the `<Bamboo-Install>/log` folder.

⚠️ **Logging on the Bamboo Agent**

All agent activity is recorded in `atlassian-bamboo-agent.log` file, stored on the agent machine.

Managing the atlassian-bamboo logs

Controlling the level of logging

Bamboo uses the [log4j library](https://log4j.org) for logging during runtime. The logging levels can be changed by editing the `<Bamboo-Install>/webapp/WEB-INF/classes/log4j.properties` file. There are four logging levels available: 'INFO', 'DEBUG', 'ERROR' and 'FATAL'.

The `rootLogger` property controls the verbosity of logs being generated at the top level. By default, the root level logging is set to 'INFO'. To change the root level logging, follow the steps below:

1. Find the following lines in `<Bamboo-Install>/webapp/WEB-INF/classes/log4j.properties` file:

   ```
   # Change the following line to configure the bamboo logging levels (one of INFO, DEBUG, ERROR, FATAL)
   #
   log4j.rootLogger=INFO, console, filelog
   ```

2. Update the value of `log4j.rootLogger` to the desired logging level.
3. Save changes to the file.
4. Restart Bamboo.

Selecting where the atlassian-bamboo logs will be stored

By default, the atlassian-bamboo logs can either be found either in the root `<Bamboo-Install>` directory or the directory you started Bamboo from. In case of a Tomcat webapp deployment, the logs are piped out to `catalina.out` file.

To change the directory that the atlassian-bamboo logs are generated to, you must set the environment variable for the target location of the logs, as seen below:

```log4j.appender.fileLog.file=/my/path/to/atlassian-bamboo.log```

See also [7.1 Locating Important Directories and Files](#).
Performing a thread dump.

This page last changed on Mar 06, 2008 by asridhar.

If Bamboo stops responding, or is performing poorly, you should create a thread dump to help Atlassian determine the cause of the problem.

This will show the state of each thread in the JVM, including a stack trace and information about what locks that thread is holding and waiting for.

Windows Users

To take a thread dump from Windows:

2. Click Run for any security warnings
3. Select Process -> Thread Dump
4. Under Process Id, select the '...' button.
5. From the drop-down list, select the Bamboo process. Users running Bamboo Standalone, select the 'Java (Jetty) ...' option. Users running Bamboo WAR should select their application server process.
6. Ensure that the "Thread dump" and "Keep Remote Thread Running" is selected.
7. Click OK to capture the thread dump.
8. Save the output to a file, eg 'threaddump.log'
9. If you were asked by Atlassian technical support to create the thread dump, attach the logfile to the support ticket.

   Alternatively, if you are not running Bamboo as a service, click on the console and press <CTRL>+BREAK

Linux (and Solaris and other Unices) Users

Find the process id of the JVM and issue the command:

```
kill -3 <pid>
```

**Note:** This will not kill your server (so long as you included the "-3" option, no space in between). The thread dump will be printed to Bamboo's standard output (catalina.out).

Thread Dump Tools

- Samurai
- Thread Dump Analyzer TDA
Problems running Bamboo under Sun JDK 1.4

This page last changed on Nov 01, 2007 by asridhar.

Backing up Bamboo

With Bamboo versions 1.2.1 and 1.2.2, exporting Bamboo when running under Sun JDK version 1.4 will fail due to incompatible with JAVA versions.

⚠️ Exporting Bamboo

This issue was fixed in Bamboo 1.2.3.

Bamboo version 1.2 will not start under Sun JDK 1.4

It is recommended that users upgrade to Bamboo 1.2.1 or above as a fix for this issue.
Removing Coverage plug-in data from the Bamboo database

The third-party Coverage plug-in for Bamboo stores very large amounts of data in the Bamboo database. There are two consequences of this:

1. Using the Coverage plug-in with an embedded Bamboo database may result in poor performance, OutOfMemoryErrors, and/or Bamboo start-up failures; and
2. After installing the Coverage plug-in, you may encounter difficulties with Bamboo's import, export and backup features, such as OutOfMemoryErrors and corrupted export and backup files.
3. We have also had reports of indexing issues, caused due to the coverage plugin

Precautionary Measures

To mitigate the risk of these problems, Atlassian makes the following recommendations to users of the Coverage plug-in:

1. Atlassian strongly recommends that you migrate to a supported external database before installing this plug-in and
2. Once you have installed the Coverage plug-in, Atlassian strongly recommends that you regularly backup your bamboo-home and external database using external tools, as the plug-in may interfere with the reliability of Bamboo’s built-in backup feature. We intend to address the underlying issue in a future release of Bamboo.

Recovery Procedure

If an instance of Bamboo is configured with an embedded database and the Coverage plug-in is failing for the reasons described above, this can be rectified by removing the Coverage plug-in's data from the database, using the following procedure.

On Linux, Mac OS X and other Unix-like platforms:

1. Shut down Bamboo.
2. Execute the following commands in a shell, substituting bamboo-home with the path to your Bamboo home directory:

```bash
    cd bamboo-home/database
    grep "^INSERT INTO BUILDRESULTSUMMARY_CUSTOMDATA .*'coverage\." defaultdb.script | gzip > coverage.sql.gz
    mv defaultdb.script defaultdb.script.backup_with_coverage
    gzip defaultdb.script.backup_with_coverage
    gunzip defaultdb.script.backup_with_coverage.gz | grep -v "^INSERT INTO BUILDRESULTSUMMARY_CUSTOMDATA .*'coverage\." > defaultdb.script
```

On Microsoft Windows:

For assistance, please raise a Bamboo support request.

If you are using an external database, login to your database using a command line client and execute the following command:

```sql
    DELETE FROM buildresultsummary_customdata where custom_info_key like '%coverage%'
```
Restoring passwords to recover admin users

This page last changed on Nov 01, 2007 by rosie@atlassian.com.

Use this document if you are unable to login as administrator or have forgotten your password and do not have Mail Server configured, to manually replace administrator passwords.

Follow the instructions for either the Embedded Database or External Database. If you have not configured a database, use the Embedded instructions.

Embedded Database Instructions

Stage One - Identify Administrator

This guide assumes that the first user added was an administrator. If this is not the case, search for the admin username and find their user id number, then modify their password hash instead.

1. Shutdown Bamboo
2. In your Bamboo home directory, open \database\defaultdb.script file in a text editor
3. Search for the text:

   INSERT INTO USERS VALUES(1
   To find the administrator login entry:
   INSERT INTO USERS VALUES(1,'USERNAME','PASSWORD_HASH')
   Where the 1 is the user id number, and USERNAME and PASSWORD_HASH are actual values. As an example, my table entry for user admin with password admin looks like this;

   INSERT INTO USERS
   VALUES(1,'admin','x61Ey612Kl2gpFL56FT9weDnpSo4AV8j8+qx2AuThdRyY036xxzTTrw10Wq3+4qQyB+XURFPx1ONxp3Y3pB37A==','admin@admin.com','2007-08-14 11:26:18.504000000','admin')

1. This step makes admin the administrator's password. Bamboo does not store passwords in plain text in the database, but uses hashes computed from the original password. The hash for the characters admin is below:

   x61Ey612Kl2gpFL56FT9weDnpSo4AV8j8+qx2AuThdRyY036xxzTTrw10Wq3+4qQyB+XURFPx1ONxp3Y3pB37A==

Paste the admin password hash between the '' characters of their existing PASSWORD_HASH. The new administrator login entry should look like:

   INSERT INTO USERS
   VALUES(1,'USERNAME','x61Ey612Kl2gpFL56FT9weDnpSo4AV8j8+qx2AuThdRyY036xxzTTrw10Wq3+4qQyB+XURFPx1ONxp3Y3pB37A==','EMAIL','DATE_TIME','FULL_NAME')

   Where USERNAME is the administrator username.
2. Save the file
3. Start up Bamboo
4. Login with the administrator username and password admin

External Database Instructions

Stage One - Identify User

The first user added is always an admin. To restore your password you simply need to update the password hash in the USERS table with the admin hash
Connect to your database using a database admin tool such as DBVisualiser. Please download a database admin tool now if you do not have one installed already. Once installed, connect to your database and retrieve the list of administrator usernames with:

```
select * from USERS where ID=1
```

This command should list all users who belong to Bamboo-Admin user group.

**Stage Two - Replace Administrator Password**

Bamboo does not store passwords in plain text in the database, but uses hashes computed from the original password. You instead cut and a paste a hash, rather than the plain password, over the existing password. Below is the hash for the password `admin`

```
x61Ey612Kl2gpFL56FT9weDnpSo4AV8j8+qx2AuThdRy036xxzTTrw10Wq3+4qQyB+XURFWx1ONxp3Y3pB37A==
```

To change the password to `admin` for a given username:

1. Shutdown Bamboo
2. Connect to your database. run this SQL on your database:
   ```
   select * from USERS where NAME='admin'
   ```
   If you are using LDAP integration for user managment (not only authentication) then your admin user will be in a different table. The SQL to run is:
   ```
   update USERS set PASSWORD =
   'x61Ey612Kl2gpFL56FT9weDnpSo4AV8j8+qx2AuThdRy036xxzTTrw10Wq3+4qQyB
   +XURFWx1ONxp3Y3pB37A==' where NAME = 'USER_NAME_FROM_STAGE.ONE'
   ```
3. Start Bamboo
4. Login with your username and your password is now `admin`
**Testing LDAP or Active Directory connectivity with Paddle**

This page last changed on Nov 01, 2007 by rosie@atlassian.com.

### Introduction

Paddle is a tool that will test the LDAP or Active Directory settings in your atlassian-user.xml.

### Instructions for use

You do not need to have Bamboo running to run this tool. The steps are:

1. Download into a directory where you have permissions to create files.
2. Copy your atlassian-user.xml into that directory - this is found in your <Bamboo-Install>/webapp/WEB-INF/classes/ directory.
3. Run java -jar paddle-x.x.jar (where x.x is the version of Paddle you downloaded).

### Parameters

Paddle currently supports the following parameters:

<table>
<thead>
<tr>
<th>Name</th>
<th>Example</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>debug</td>
<td>java -jar paddle-x.x.jar debug</td>
<td>Prints DEBUG messages to the console as well as paddle.log.</td>
</tr>
<tr>
<td>limit</td>
<td>java -jar paddle-x.x.jar limit=100</td>
<td>Sets the limit on the number of results returned by user and group queries. Defaults to 10.</td>
</tr>
</tbody>
</table>

### Sample output

This is an example of a successful run:

```
LDAP Support Tool version 1.1

Connection to LDAP/Active Directory Server at ldap://192.168.0.86:389 SUCCESSFUL.

TEST 1: Search and list 10 users

User: CN=Administrator
Member of:
    (1) CN=Schema Admins
    (2) CN=Enterprise Admins
    (3) CN=Domain Admins
    (4) CN=Group Policy Creator Owners

User: CN=Guest
Does not belong to any LDAP groups.

User: CN=SUPPORT_388945a0
Member of:
    (1) CN=HelpServicesGroup

User: CN=IUSR_MALTSHOVEL
Does not belong to any LDAP groups.

User: CN=IWAM_MALTSHOVEL
Member of:
    (1) CN=IIS_WPG
```
User: CN=ASPNET
  Does not belong to any LDAP groups.

User: CN=krbtgt
  Does not belong to any LDAP groups.

User: CN=John\, Smith
  Member of:
    (1) CN=Domain Users
    (2) CN=Sales and Marketing

User: CN=Matt Ryall
  Member of:
    (1) CN=Enterprise Admins
    (2) CN=Domain Admins

User: CN=Justin Koke
  Member of:
    (1) CN=Domain Controllers
    (2) CN=Enterprise Admins

Found more than 10 results.

-----------------------------------------------------------------
TEST 2: Search and list 10 groups
-----------------------------------------------------------------

Group: CN=HelpServicesGroup
  Members:
    (1) CN=SUPPORT_388945a0,CN=Users,DC=ad,DC=atlassian,DC=com

Group: CN=TelnetClients
  No members in this group.

Group: CN=IIS_WPG
  Members:
    (1) CN=S-1-5-20,CN=ForeignSecurityPrincipals,DC=ad,DC=atlassian,DC=com
    (2) CN=S-1-5-6,CN=ForeignSecurityPrincipals,DC=ad,DC=atlassian,DC=com
    (3) CN=S-1-5-18,CN=ForeignSecurityPrincipals,DC=ad,DC=atlassian,DC=com
    (4) CN=IWAM_MALTSHOVEL,CN=Users,DC=ad,DC=atlassian,DC=com

Group: CN=SQLServer2005SQLBrowserUser$MALTSHOVEL
  Members:
    (1) CN=S-1-5-18,CN=ForeignSecurityPrincipals,DC=ad,DC=atlassian,DC=com

Group: CN=SQLServer2005MSSQLServerADHelperUser$MALTSHOVEL
  Members:
    (1) CN=S-1-5-20,CN=ForeignSecurityPrincipals,DC=ad,DC=atlassian,DC=com

Group: CN=SQLServer2005SQLAgentUser$MALTSHOVEL$MSSQLSERVER
  Members:
    (1) CN=S-1-5-18,CN=ForeignSecurityPrincipals,DC=ad,DC=atlassian,DC=com

Group: CN=SQLServer2005MSSQLUser$MALTSHOVEL$MSSQLSERVER
  Members:
    (1) CN=S-1-5-18,CN=ForeignSecurityPrincipals,DC=ad,DC=atlassian,DC=com

Group: CN=SQLServer2005MSFTEmUser$MALTSHOVEL$MSSQLSERVER
  Members:
    (1) CN=S-1-5-18,CN=ForeignSecurityPrincipals,DC=ad,DC=atlassian,DC=com

Group: CN=SQLServer2005MSOLAUser$MALTSHOVEL$MSSQLSERVER
  Members:
    (1) CN=S-1-5-18,CN=ForeignSecurityPrincipals,DC=ad,DC=atlassian,DC=com
Group: CN=SQLServer2005NotificationServicesUser$MALTSHOVEL
      No members in this group.

Found more than 10 results.
To troubleshoot your SVN connection, it is useful to enable logging. Modify the system property while launching the Java VM with:

```java
$java -Djava.util.logging.config.file=path/to/logging.properties.disabled
```

To do this:
1) Download the `logging.properties.disabled` file attached to this document.

The `logging.properties.disabled` file contains a system property `svnkit.level` used to control the log level:

- FINE (default) — the level for non-detailed logging
- FINER — the level for more detailed logging
- FINEST --- the level for full logging

2) Configure Bamboo to start with the `-Djava.util.logging.config.file=*path/to/*logging.properties.disabled` command, where "path/to/" refers to the absolute path to the `logging.properties.disabled` file from step (1).

The method of doing this depends on which operating system you are using:

- If you are running Bamboo under Linux:
  Modify the bamboo .sh script to pass in the logging parameter. To do this, find the following section:

  ```
  RUN_CMD="java -Xms256m -Xmx512m -XX:MaxPermSize=256m -Djava.awt.headless=true -classpath $CLASSPATH -Dorg.mortbay.xml.XmlParser.NotValidating=true -Djetty.port=8085 com.atlassian.bamboo.server.Server 8085 ./webapp /
  
  Change the RUN_CMD variable to:
  
  ```
  ```
  RUN_CMD="java -Xms256m -Xmx512m -XX:MaxPermSize=256m -Djava.awt.headless=true -classpath $CLASSPATH -Dorg.mortbay.xml.XmlParser.NotValidating=true -Djava.util.logging.config.file=*path/to/*logging.properties.disabled -Djetty.port=8085 com.atlassian.bamboo.server.Server 8085 ./webapp /
  
  ```

- If you are running Bamboo under Windows:
  Modify the `<Bamboo_install_Home>/conf/wrapper.conf` file to add the logging parameter. To do this, find the following section:

  ```
  wrapper.java.classpath.1=../lib/*.jar
  wrapper.java.classpath.2=../lib
  wrapper.java.classpath.3=../webapp/WEB-INF/classes
  wrapper.java.classpath.4=../webapp/WEB-INF/lib/*.jar
  wrapper.java.library.path.1=../lib
  wrapper.java.additional.1=-Dorg.mortbay.xml.XmlParser.NotValidating=true
  wrapper.java.additional.2=-XX:MaxPermSize=256m
  wrapper.java.additional.3=-Djava.awt.headless=true
  
  Add the following line to the end of the section:
  
  ```
  ```
  wrapper.java.additional.4=-Djava.util.logging.config.file=*path/to/*logging.properties.disabled
  ```

3) Save your changes, and restart Bamboo for your changes to take effect.
4) You will find the SVNKit log file in USER_HOME/svnkit.0.log (where USER_HOME is the home directory of the user running Bamboo or logged in e.g: /home/user/svnkit.0.log).
Using Bamboo with Clover

This page last changed on Dec 12, 2007 by rosie@atlassian.com.

Getting started

To use Clover with Bamboo, you need to:

1. Either:
   • call the Clover goal in your plan configuration (see 1.2.3 Specifying a Plan's Builder);
   or:
   • add the maven-clover-plugin report to the reports section in your POM.
2. Ensure that there are tests present in your build plan that generate test results in JUnit test report format.
3. Ensure that your build creates a Clover report (that is, a clover.xml file). Bamboo will use this Clover report as source.
4. Set up Bamboo to read the Clover report (clover.xml file) generated by Clover. To do this:
   a. Ensure the 'Clover output will be produced' check-box is ticked in your plan's build configuration page.
   b. Instruct Bamboo on the location of your 'Clover XML Directory' — where Bamboo will look for the XML report output file from Clover. Please specify file path relative to your plan's root directory (e.g. /home/bamboouser.bamboo-home/xml-data/build-dir/MY_PLAN/), i.e. please do not specify an absolute path.

For further details, please see 1.2.3 Specifying a Plan's Builder.

Q&A:

Q: I have managed to get Clover statistics displayed in numerical form for each build, but the graphs do not show a history of these statistics?
A: The history of Clover is displayed over time periods (e.g. a day, a week, a month), and the minimum data point is per day. The Clover coverage will not display data that is less than a day old.

Q: Will the Bamboo/Clover integration run on failed builds?
A: Before Bamboo version 1.2.1, Bamboo would only report Clover coverage for successful builds. As of Bamboo 1.2.1, Bamboo will report Clover coverage regardless of the build outcome.
Working with Sun JAVA libraries

Due to licensing restrictions, we are not allowed to re-distribute native SUN libraries through our maven2 public repositories.

If you are developing plugins for Bamboo or building Bamboo from source, you might need javax.mail and javax.transaction:jta:jar for Bamboo to build successfully. The relevant POMs for this look something like this:

```
<dependency>
  <groupId>javax.mail</groupId>
  <artifactId>mail</artifactId>
  <version>1.3.2</version>
  <scope>compile</scope>
</dependency>
<dependency>
  <groupId>jta</groupId>
  <artifactId>jta</artifactId>
  <version>1.0.1</version>
  <scope>compile</scope>
</dependency>
```

Before building, please install the Sun JAR's into your local Maven2 repositories by following the instructions below.

To install the **javax.mail** JAR into your local Maven2 repository:

1. Download the **javax.mail** Jar from Sun's website.
2. Install it on your local machine by entering the following command in a terminal:

   ```
   mvn install:install-file -DgroupId=javax.mail -DartifactId=mail -Dversion=1.3.3 -
   Dpackaging=jar -Dfile=YOUR/PATH/TO/FILE
   ```

To install **javax.transaction:jta:jar** JAR into your local Maven2 repository:

1. Download the **javax.transaction:jta:jar** Jar from Sun's website.
2. Install it on your local machine by entering the following command in a terminal:

   ```
   mvn install:install-file -DgroupId=javax.transaction -DartifactId=jta -Dversion=1.0.1B -
   Dpackaging=jar -Dfile=path/to/file
   ```
Bamboo User's Guide

This page last changed on Aug 09, 2007 by rosie@atlassian.com.

About

Bamboo is a continuous integration (CI) server. Bamboo assists software development teams by providing:

• automated building and testing of software source-code status.
• updates on successful/failed builds.
• reporting tools for statistical analysis.

The Bamboo User's Guide provides information about using Bamboo. If you need information about installing Bamboo or configuring builds, please visit Bamboo Documentation Home.

If you have a question about using Bamboo that hasn't been answered here, please let us know.

Download

You can download the Bamboo documentation in PDF, HTML or XML formats.

Search the User's Guide

Getting Help

Support | Feature Request | Forums | Knowledge Base

Table of Contents

01. Getting Started

• 1.1 Using the Bamboo Dashboard
• 1.2 Viewing Bamboo's Current Activity
• 1.3 Viewing your Latest Build Results
• 1.4 Working with Favourites
  • 1.4.1 Adding a Plan to your Favourites
  • 1.4.2 Removing a Plan from your Favourites
• 1.5 Displaying a Build Monitor
• 1.6 Viewing Bamboo's Agents

02. Working with Projects and Plans

• 2.1 About Projects and Plans
• 2.2 Viewing a Plan's Details
• 2.3 Viewing a Plan's Activity Log

03. Working with Build Results

• 3.1 About Builds and Build Results
• 3.2 Viewing a Build Result
• 3.3 Viewing the Code Changes that triggered a Build
• 3.4 Viewing a Build's Artifacts
• 3.5 Viewing a Build Log
• 3.6 Viewing the Metadata for a Build Result
• 3.7 Viewing the Clover Code-Coverage for a Build Result
• 3.8 Viewing the JIRA Issues for a Build Result

04. Working with Tests

• 4.1 Viewing Test Results for a Build
• 4.2 Viewing a Test's History
• 4.3 Viewing Test Statistics for a Plan

05. Reporting on Plan Trends

• 5.1 Viewing Build Statistics for a Plan
• 5.2 Generating Reports across multiple Plans
  ° 'Build Activity per Plan' Report
  ° 'Build Duration per Plan' Report
  ° 'Clover Code Coverage per Plan' Report
  ° 'Clover Lines of Code per Plan' Report
  ° 'Number of Build Failures per Plan' Report
  ° 'Number of Tests per Plan' Report
  ° 'Percentage of Successful Builds per Plan' Report
  ° 'Time to Fix per Plan' Report

• 5.2 Generating Reports across multiple Plans
  ° 'Build Activity per Plan' Report
  ° 'Build Duration per Plan' Report
  ° 'Clover Code Coverage per Plan' Report
  ° 'Clover Lines of Code per Plan' Report
  ° 'Number of Build Failures per Plan' Report
  ° 'Number of Tests per Plan' Report
  ° 'Percentage of Successful Builds per Plan' Report
  ° 'Time to Fix per Plan' Report

06. Reporting on Author Trends

• 6.1 Viewing Build Statistics for all Authors
• 6.2 Viewing Build Results for an Author
• 6.3 Generating Reports on selected Authors
  ° 'Build Activity per Author' Report
  ° 'Number of Build Failures per Author' Report
  ° 'Number of Builds Broken per Author' Report
  ° 'Number of Builds Fixed per Author' Report
  ° 'Percentage of Successful Builds per Author' Report

07. Working with Comments

• 7.1 About Comments
• 7.2 Commenting about a Build Result
• 7.3 Viewing Comments about a Build Result
• 7.4 Viewing Code Check-in Comments

08. Working with Labels

• 8.1 About Labels
• 8.2 Labelling a Build Result
• 8.3 Removing a Label from a Build Result
• 8.4 Viewing Labelled Build Results
• 8.5 Viewing Popular Labels

09. Subscribing to RSS Feeds

• 9.1 Subscribing to an RSS Feed for All Build Results for All Plans
• 9.2 Subscribing to an RSS Feed for Failed Builds for All Plans
• 9.3 Subscribing to an RSS Feed for All Build Results for a Particular Plan
• 9.4 Subscribing to an RSS Feed for Failed Builds for a Particular Plan
• 9.5 Subscribing to an RSS Feed for Labelled Build Results

10. Working with Instant Messenger (IM) Notifications

• 10.1 About Instant Messenger (IM) Notifications
• 10.2 Labelling a Build Result via IM
• 10.3 Commenting about a Build Result via IM

11. Editing your User Profile

• 11.1 Changing your Password
• 11.2 Changing your Notification Preferences
• 11.3 Associating your Author Name with your User Profile
01. Getting Started

1. Getting Started

- 1.1 Using the Bamboo Dashboard
- 1.2 Viewing Bamboo's Current Activity
- 1.3 Viewing your Latest Build Results
- 1.4 Working with Favourites
  - 1.4.1 Adding a Plan to your Favourites
  - 1.4.2 Removing a Plan from your Favourites
- 1.5 Displaying a Build Monitor
- 1.6 Viewing Bamboo's Agents
1.1 Using the Bamboo Dashboard

This page last changed on Jan 06, 2008 by rosie@atlassian.com.

The Dashboard is your Bamboo 'home' page. The Dashboard contains three tabs:

- 'All Plans' — a list of build plans and each plan's latest build result.
- 'Current Activity' — Bamboo's agents and build queue, showing which plans Bamboo is currently building and which plans are waiting to be built.
- 'My Bamboo'¹ — a convenient summary of information that is relevant to you:
  ° plans which you have nominated as your favourites.
  ° your latest build results (i.e. builds that were triggered by your latest code changes).
  ° a summary of your build statistics².

Screenshot: Bamboo Dashboard-'All Plans' tab

Each blue name (e.g. 'Crowd') represents a project. The name(s) to the right of each project name are the plan(s) (e.g. 'Crowd 1.1 - Build Plugin') belonging to that project, and the build number (e.g. 'CWD-BUILDPLUGIN-11') represents the latest build result for the plan, while the icon indicates the plan's current status:

- ![This plan's latest build was successful.](image)
- ![This plan's latest build failed.](image)
- ![Bamboo is currently checking-out the source-code for this plan, in preparation for starting a build.](image)
- ![Bamboo is currently executing a build for this plan.](image)
- ![This plan has been disabled.](image)

You can:

- click the plan name (e.g. 'Crowd 1.1 - Build Plugin') to view the plan details.
- click the build number (e.g. 'CWD-BUILDPLUGIN-11') to view the build result.
- click the author's name to view the author's details (the author is the person who triggered the build by checking-in code).

Handy Hint
You can return to the Dashboard from anywhere in Bamboo by clicking the 'Home' link in the top navigation bar.

¹ only if you have logged in to Bamboo.
² only if your Bamboo User Profile has been associated with your Author Name.
1.2 Viewing Bamboo's Current Activity

Sometimes you may want to see which plans are currently being built, and which plans (if any) are waiting in the build queue.

The Bamboo build queue controls the sequence of builds. When a plan submits a build to the build queue, the build will wait in the build queue until a suitable agent is available to run the build.

To view Bamboo's current activity,

1. Click the 'Home' link in the top navigation bar. This will display the Dashboard.
2. Click the 'Current Activity' tab. This will display Bamboo's Agents and Build Queue, as well as a list of Recently Completed Builds.

The above screenshot shows a Bamboo system that has three Agents: 'Local Agent 1', 'Local Agent 2' and 'Remote Agent on sapporo.sydney.atlassian.com'. 'Local Agent 1' is currently building a plan called 'Studio Functional Tests - Studio Functional Tests'. 'Local Agent 2' is currently building a plan called 'Bamboo - Acceptance Tests'. 'Remote Agent on sapporo.sydney.atlassian.com' is currently not building any plans.

You can:

- click a plan name (e.g. 'Studio Functional Tests') to view the plan details.
- click the icon to view the plan's continuous scrolling activity log.

Additionally, in the 'Recently Completed Builds' section, you can:

- click a build number (e.g. 'SUT-SNF-40) to view the build result.
- click a 'Reason' (e.g. 'Updated by...') to view the code changes that triggered the build.
1.3 Viewing your Latest Build Results

To view your latest build results,

1. Click the 'Home' link in the top navigation bar. This will display the Dashboard.
2. Click the 'My Bamboo' tab.
3. Your 10 latest build results (that is, builds that were triggered when you checked-in code) are listed in the 'My Latest Changes' section.

**Handy Hint**
Click any build number (e.g. 'BAM-MAIN-1846') to view the build result.

**Screenshot: 'My Bamboo--My Latest Changes'**

<table>
<thead>
<tr>
<th>Build</th>
<th>When</th>
<th>Comments</th>
<th>Test Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAM-MAIN-1846</td>
<td>2 days ago</td>
<td>Moved the powered by to the website</td>
<td>![1308 passed]</td>
</tr>
<tr>
<td>BAM-MAIN-1844</td>
<td>3 days ago</td>
<td>Improved release-plug-in prepare release:atlassian.bamboo_1_6_13</td>
<td>![1308 passed]</td>
</tr>
<tr>
<td>BAM-MAIN-1842</td>
<td>2 days ago</td>
<td>Updated so that by default JS toggle thing works</td>
<td>![1200 passed]</td>
</tr>
<tr>
<td>BamAfsha-1842</td>
<td>3 days ago</td>
<td>Improved release-plug-in prepare for next development iteration</td>
<td>![1308 passed]</td>
</tr>
<tr>
<td>BAM-MAIN-1840</td>
<td>3 days ago</td>
<td>IE doesn't quite seem to like clearer &lt;ion&gt; tags using &lt;div&gt; instead</td>
<td>![1308 passed]</td>
</tr>
<tr>
<td>BAM-MAIN-1839</td>
<td>3 days ago</td>
<td>Fixed footer for IE, some grammatical errors and the default favicon</td>
<td>![1308 passed]</td>
</tr>
<tr>
<td>BAM-MAIN-1838</td>
<td>3 days ago</td>
<td>Updated the step in the showBuildResults to redirect correctly</td>
<td>![1308 passed]</td>
</tr>
<tr>
<td>BAM-MAIN-1836</td>
<td>3 days ago</td>
<td>Updated the install rename and the installer profile</td>
<td>![1308 passed]</td>
</tr>
<tr>
<td>BAM-MAIN-1834</td>
<td>2 days ago</td>
<td>Added an offline argument for the release builds</td>
<td>![1200 passed]</td>
</tr>
<tr>
<td>BamAfsha-1833</td>
<td>3 days ago</td>
<td>Restored the ability to restore the hide / show thing through javascript</td>
<td>![1308 passed]</td>
</tr>
</tbody>
</table>

⚠️ If your Bamboo User Profile has not yet been associated with your Author Name, there will be no 'My Latest Changes' section.
1.4 Working with Favourites

Whereas the 'All Plans' tab on the Bamboo Dashboard lists every plan that exists in your Bamboo system, the 'My Bamboo' tab lists just your chosen favourites — that is, the plans you work with the most. You can easily add and remove plans from your favourites.

When you add a plan to your favourites, you become a 'watcher' of the plan. This means that you may receive notifications about the build results for some or all of your favourite plans, depending on how your administrator has configured each plan's notifications. You can choose whether you would like to receive your notifications by email and/or Instant Messaging (IM).

To view your favourite plans,

1. Click the 'Home' link in the top navigation bar. This will display the Dashboard.
2. Click the 'My Bamboo' tab.
3. Your favourite plans are listed in the 'My Favourite Plans' section. A yellow star is shown next to each.

Screenshot: 'My Bamboo--My Favourite Plans'
1.4.1 Adding a Plan to your Favourites

To add a plan to your favourites,

1. Click the 'Home' link in the top navigation bar. This will display the Dashboard.
2. Click the 'All Plans' tab.
3. This will display a list of all plans in your Bamboo system. (Note: Plans that have already been added to your favourites are indicated by a yellow star icon. Plans that have not been added to your favourites are indicated by a grey star icon.)
4. Locate the plan and click the grey star icon:
5. Click the 'My Bamboo' tab.
6. Verify that the plan is now listed in the 'My Favourite Plans' section.

You are now a 'watcher' of the plan. This means that you may receive notifications about the build results for this plan, depending on how your administrator has configured the plan's notifications. You can choose whether you would like to receive your notifications by email and/or Instant Messaging (IM).

Handy Hint
If your administrator has enabled 'Auto-Favourites', each plan will be automatically added to your favourites the first time you check-in code for that plan.
1.4.2 Removing a Plan from your Favourites

To remove a plan from your favourites,

1. Click the 'Home' link in the top navigation bar. This will display the Dashboard.
2. Click the 'All Plans' tab.
3. This will display a list of all plans in your Bamboo system. (Note: Plans that have been added to your favourites are indicated by a yellow star icon. Plans that have not been added to your favourites are indicated by a grey star icon.)
4. Locate the plan and click the yellow star icon:
   🌟
5. Click the 'My Bamboo' tab.
6. Verify that the plan is not listed in the 'My Favourite Plans' section.
1.5 Displaying a Build Monitor

Sometimes a development team can benefit from setting up a monitor to display Bamboo's latest build results, e.g.:

You can choose to display Bamboo's latest results for your favourite plans only, or for all plans that you have permission to see.

To display Bamboo's latest build results, for all plans,

1. Log into Bamboo. (Note: if your Bamboo administrator has allowed anonymous access, this step is optional.)
2. Type the following URL into your browser, but replace 'bambooserver' with the real name of your Bamboo server:

   http://bambooserver:8080/bamboo/telemetry.action

To display Bamboo's latest build results, for your favourite plans only,

1. Log into Bamboo. (Note: only logged-in users can have favourites.)
2. Type the following URL into your browser, but replace 'bambooserver' with the real name of your Bamboo server:


Hint
If you are going to display the build monitor permanently, you may want to ask your Bamboo administrator to create a user who has only a limited set of permissions.
1.6 Viewing Bamboo’s Agents

An agent is a service that runs Bamboo builds. There are two types of agents:

- local agents run on the Bamboo server.
- remote agents run on computers other than the Bamboo server.

(Note: Local agents run in the server’s process, i.e. in the same JVM as the server. Each remote agent runs in its own process, i.e. has its own JVM.)

To view only agents which are currently active, see 1.2 Viewing Bamboo’s Current Activity

To view all of Bamboo’s agents,

1. Click the ‘Home’ link in the top navigation bar. This will display the Dashboard.
2. Navigate to the
3. Click the ‘Agents’ link in the 'Agents' panel of the 'Current Activity' tab.
4. A list of all agents in your Bamboo system will display (see screenshot below).

To view a specific agent,

1. Click the ‘Home’ link in the top navigation bar. This will display the Dashboard.
2. Click the name of the agent you wish to view, in the 'Agents' panel of the 'Current Activity' tab.
3. The details of the selected agent will display (see screenshot below).
4. Click on the 'Executable Plans' tab to view the plans that this agent is capable of building (see screenshot below).

   Screenshot: View Agent - Executable Plans

5. Click on the 'Capabilities' tab to view the capabilities of this agent.

   Screenshot: View Agent - capabilities
02. Working with Projects and Plans

This page last changed on Feb 01, 2007 by rosie@atlassian.com.

2. Working with Projects and Plans

- 2.1 About Projects and Plans
- 2.2 Viewing a Plan's Details
- 2.3 Viewing a Plan's Activity Log
2.1 About Projects and Plans

A Bamboo plan (or build plan) is the "recipe" for a build.

A plan defines: what gets built (i.e. the source-code repository); how the build is triggered; which builder to use; which agent capabilities are required for the build; what artifacts the build will produce; what tests to run; who will be notified of the build result; any labels with which the build result or build artifacts will be tagged; and who has permission to view and perform various actions on a plan and its build results.

Every plan belongs to a project.

A project enables easy identification of plans that are logically related to each other, which is useful for instance when generating reports across multiple plans. Each project has a Name (e.g. "CRM System") and a Key (e.g. "CRM"). The Project Key is prefixed to the relevant Plan Keys, e.g. the "CRM" project could have plans "CRM-TRUNK" and "CRM-BRANCH".

Every Bamboo plan is listed on the Dashboard, from where you can:

- Click on a Plan Name to view the plan details
- Click on a Build Number to view the plan's latest build result

Projects and plans can only be configured by Bamboo administrators. Please see the Bamboo Administrator's Guide for details.
2.2 Viewing a Plan’s Details

To view a plan’s details,

- From the Dashboard, locate and click a Plan Name from the list;
- OR:
  - From within a build result, click the Plan Name at the top left of the screen.

The Plan Summary will be displayed as follows:

**Screenshot: 'Plan Summary'**

In the above screenshot:

The green box indicates that this plan's latest build was successful. Note that a red box in this position would indicate that the plan's latest build failed, while a blue box would indicate that a build is currently in progress.

- Click the build number (i.e. 'BUCKET-MAIN-71') to view the build result.
• Click the 'Updated by' link to view the code changes that triggered the latest build result.

• _Build duration_ is the total time taken to execute a [build plan](#) --- that is, the time taken to compile the code and run all of the plan's tests.

The large '%' box indicates the success rate of this plan's recent builds. This percentage is calculated on the last 25 builds, or as per your selection via the blue down-arrow:

Click the blue down-arrow to choose how you would like the percentage and graphs on this screen to be calculated. Choose from the following:

• this plan's last 25 builds.
• this plan's builds in the last 7 days.
• this plan's builds in the last 30 days.
• this plan's builds in the last 90 days.
• all of this plan's builds. The percentage and graphs on this screen will all be recalculated automatically when you choose a different option.

The 'Summary' tab provides a quick snapshot of the current status of the plan. For more details:

• Click the 'Activity' tab to view the plan's current activity.
• Click the 'Completed Builds' tab to view a list of build results for this plan's recent builds (i.e. the last 25 builds, or as per your selection via the blue down-arrow).
• Click the 'Tests' tab to view a summary of the test results for this plan's recent builds (i.e. the last 25 builds, or as per your selection via the blue down-arrow).
• Click the 'Files' tab to view a list of all the files currently contained in this plan's source-code repository. (Note: The 'Files' tab will only be visible if the plan's latest build ran on a local agent, since the files will not be available if the build ran on a remote agent.)

ℹ️ Plans can only be configured by an administrator. For details please see the Bamboo Administrator's Guide.
2.3 Viewing a Plan's Activity Log

Every plan has an activity log. An activity log is a temporary display of the latest output from the plan's most recent build log.

To view a plan's activity log,

1. From the Dashboard, locate and click a plan name from the list; OR:
   From within a build result, click the plan name at the top left of the screen.
2. Click the 'Activity Log' tab.

The plan's current activity will be displayed. For example, the following screenshot shows a plan for which a build is currently in progress:

![Screenshot: Plan Activity]

You can also monitor a plan's build activity over time by using the 'Build Activity per Plan' report. Build activity is the number of builds that occur in a given period of time.
03. Working with Build Results

3. Working with Build Results

- 3.1 About Builds and Build Results
- 3.2 Viewing a Build Result
- 3.3 Viewing the Code Changes that triggered a Build
- 3.4 Viewing a Build's Artifacts
- 3.5 Viewing a Build Log
- 3.6 Viewing the Metadata for a Build Result
- 3.7 Viewing the Clover Code-Coverage for a Build Result
- 3.8 Viewing the JIRA Issues for a Build Result
3.1 About Builds and Build Results

A build is one execution of a plan.

Every build has a Build Number, which is appended to the relevant Plan Key to form the Build Key. For example, if a plan with the key "CRM-BRANCH" is executed for the seventeenth time, the build key will be "CRM-BRANCH-17".

Every completed build has a build result:

- 'Successful' — the code compiled, with or without errors, and all tests completed successfully.
- 'Failed' — either the code did not compile, or at least one test failed.

Additionally,

- if the build result is 'Failed', and the previous build result was 'Successful', the build is said to be 'Broken'.
- if the build result is 'Successful', and the previous build result was 'Failed', the build is said to be 'Fixed'.

The latest build result for every plan is listed on the Dashboard. Bamboo can also send notifications and generate RSS feeds about build results.
3.2 Viewing a Build Result

To view a plan's most recent build result:

1. Go to the Dashboard.
2. Locate the plan in the list, then click the Build Number.

To view all build results for a plan:

1. Go to the relevant plan.
2. Click the 'Completed Builds' tab to see a summary list of build results.
   ❖ To view the details for a particular build result, click the Build Number in the list.

A build result looks like this:

Screenshot: 'Build Result Summary'

In the above screenshot:

- You can click the plan name ('Main Build' in this example) to see the plan details for this build:

  | Project | Bonnie | Plan | Main Build | Build | 21 |

- You can use the build results navigator to scroll through other build results for this plan:

  ❖ Summary: 55

- The 'Summary' tab shows a snapshot of the build result. To see more detail:
  ❖ Click the 'Tests' tab to view the build's test results.
  ❖ Click the 'Changes' tab to view the code changes that triggered this build (if applicable).
  ❖ Click the 'Artifacts' tab to view any artifacts relating to this build.
  ❖ Click the 'Logs' tab to view a complete build log.
  ❖ Click the 'Comments' tab to view a trail of comments regarding this build result. You can also click the following icon to add a comment:

  ❖ Click the 'Metadata' tab to view any metadata that relates to this build result.

- Depending on how your Bamboo administrator has configured the system, the following additional tabs may be available:
  ❖ Click the 'Clover' tab to view the Clover code-coverage that relate to this build result (if applicable).
  ❖ Click the 'JIRA' tab to view any JIRA issues that relate to this build result (if applicable).
3.3 Viewing the Code Changes that triggered a Build

This page last changed on Mar 16, 2008 by asridhar.

If a build was triggered by a code change, the code changes will be shown in the build result.

To view the code changes that triggered a particular build result,

1. Go to the build result.
2. Click the 'Changes' tab.
3. A list of updated files will be shown. Click the filename to view the changes; or, click the version number to view the entire file; or, click the 'diffs' links to view the differences between the current and previous version of each file.

Screenshot: 'Code Changes'

Showing code changes

- Links to individual source-code files will only be available if your Bamboo administrator has specified a 'Web Repository URL' in the build's plan. For details please see the Bamboo Administrator's Guide.
- If there are no code changes since last build (in case of a manual build) there will not be any changes displayed on screen.

A note about build triggering

There are a variety of ways in which a build can be triggered for a plan:

- Code updated — a build can be triggered whenever one or more authors checks-in code.
- Scheduled build — a build can be scheduled to occur at regular intervals.
- Dependency — a build can be triggered whenever a successful build occurs for another plan.
- Manual build — a build can be triggered manually.
- Initial clean build — a build will be triggered when a new plan is created.

The way in which each build was triggered is listed in the 'Reason' column on the Dashboard.
3.4 Viewing a Build's Artifacts

An artifact is something created by a build. There are two types of artifacts:

- User-defined artifacts (e.g. JAR files) are specified in the build’s plan by a Bamboo administrator.
- Auto-generated artifacts are created automatically by Bamboo.

To view a build's artifacts,

1. Go to the build result.
2. Click the 'Artifacts' tab.

Screenshot: 'Build Artifacts'
3.5 Viewing a Build Log

Every build has a build log. A build log is a permanent record of all the output generated by compiling the plan’s source-code and executing the tests.

To view a build log,

1. Go to the build result.
2. Click the 'Logs' tab.

Screenshot: 'Build Log'
3.6 Viewing the Metadata for a Build Result

This page last changed on Dec 13, 2007 by alui.

If your source-code repository provides metadata for your build results, Bamboo will display it.

To view the metadata for a build result:

1. Go to the plan.
2. Click the 'Completed Builds' tab, then click the Build Number in the list.
3. This will display the Build Result Summary. Click the 'Metadata' tab.

### Screenshot: Metadata for a Build Result

![Metadata for a Build Result](image)

This build has the following metadata. These are property key-value pairs describing the build. You can specify your own metadata in the build process via plugins.

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>buildKey</td>
<td>CORE-MAIN</td>
</tr>
<tr>
<td>buildNumber</td>
<td>55</td>
</tr>
<tr>
<td>custom.exe.config</td>
<td>201.75</td>
</tr>
</tbody>
</table>
3.7 Viewing the Clover Code-Coverage for a Build Result

If your organisation uses the Atlassian Clover code-coverage tool, Bamboo can record code-coverage details (i.e. the percentage of code covered by tests) for each build result.

⚠️ This is only available if the build’s plan specifies a Clover directory (for details please refer to the Bamboo Administrator’s Guide). Also note that the Clover analysis will only be recorded for successful builds.

To view Clover code-coverage for a build result:

1. Go to the plan.
2. Click the 'Completed Builds' tab, then click the Build Number in the list.
3. This will display the Build Result Summary. Click the 'Clover' tab.

Screenshot: Clover Code-Coverage for a Build Result

<table>
<thead>
<tr>
<th>Code Coverage</th>
<th>Build Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Build Coverage:</strong> 99.9% (OK)</td>
<td>LOC: 3873</td>
</tr>
<tr>
<td><strong>Statement Coverage:</strong> 22.2%</td>
<td>NLLOC: 2047</td>
</tr>
<tr>
<td><strong>Method Coverage:</strong> 36.4%</td>
<td>Files: 46</td>
</tr>
<tr>
<td><strong>Conditionale Coverage:</strong> 32.8%</td>
<td>Methods: 204</td>
</tr>
</tbody>
</table>

Bamboo also provides data on code-coverage trends for a plan over a period of time. For details see:

- 'Clover Code Coverage per Plan' Report.
3.8 Viewing the JIRA Issues for a Build Result

If your organisation uses the JIRA issue-tracker, Bamboo can automatically create links to any issues mentioned in code check-in comments, as well as displaying a summary in the build result (see below). This provides an easy way to jump to relevant issue(s) to see details about what the code is intended to achieve.

Links to JIRA issues are indicated by a small green arrow to the right of the issue key.

⚠️ This will only be available if Bamboo-JIRA communication has been enabled by your Bamboo administrator.

To view the JIRA issues for a build result:

1. Go to the plan.
2. Click the 'Completed Builds' tab, then click the Build Number in the list.
3. This will display the Build Result Summary. Click the 'JIRA' tab.

⚠️ Note that, when typing your check-in comment, the JIRA issue key(s) must be in upper case (e.g. "ABC-123", not "abc-123").

Screenshot: JIRA Issues for a Build Result

To view all build results related to a JIRA issue:

1. Click the 'Show related builds' link in the above screenshot.
04. Working with Tests

This page last changed on Feb 01, 2007 by rosie@atlassian.com.

4. Working with Tests

- 4.1 Viewing Test Results for a Build
- 4.2 Viewing a Test's History
- 4.3 Viewing Test Statistics for a Plan
4.1 Viewing Test Results for a Build

Bamboo provides a convenient summary of all the tests that were run when a particular build was executed — as well as full details of any errors. This is useful when you are investigating what caused a build to fail.

To view the tests for a particular build:

1. Go to the build result.
2. Click the 'Tests' tab.
3. Click the 'Failed Tests' / 'Successful Tests' links (see screenshot below) to view a list of which tests failed or were successful.

To see a particular test's results for other builds, click the test name.

Screenshot: Test Results for a Build

For more meaningful display of test names within Bamboo, the word 'test' is stripped out of test case name names if it occurs at the beginning, and capitals and underscores are treated as word separators.
4.2 Viewing a Test's History

This page last changed on Dec 13, 2007 by alui.

A test's history shows you:

- The occasions when the test has failed. This can be useful when investigating what code changes were related to a failed test (see below).
- The test's average duration (running time), and whether the duration is increasing or decreasing across builds.

To view a test's history,

1. Go to a plan or a build result.
2. Click the 'Tests' tab.
3. Click the name of the test in which you are interested.
4. (Skip this step if you are looking at a plan.) The test's latest result will be displayed. Click the link 'View test case across builds'.

To view the code changes that relate to a failed test,
1. Under 'Recent Failures', click the relevant build result ('47' in the above screenshot).
2. This will display the build result. Click the 'Changes' tab to display the code changes.
4.3 Viewing Test Statistics for a Plan

Bamboo provides a summary of test results across all of a plan's builds. This helps you to:

- Troubleshoot by identifying which tests fail most frequently, and which tests take longest to fix.
- Manage your build duration by identifying the plan's slowest running tests.
- Ensure quality by monitoring the number of tests over time: are your test cases growing with your code base?

To view the test statistics for all of a plan's builds:

1. Go to the plan.
2. Click the 'Tests' tab.
3. The plan's 'Top 10 Most Failing Tests' sub-tab will be displayed. Click the other three sub-tabs to view the plan's 'Top 10 Longest to Fix Tests', 'Top 10 Longest Running Tests', 'Number of Tests' (see screenshots below).

Screenshot 1: Top 10 Most Failing Tests

<table>
<thead>
<tr>
<th>Test</th>
<th>Times Failed</th>
<th>Most Recent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freq</td>
<td>2</td>
<td>10/11/08</td>
</tr>
<tr>
<td>Image canonical case on mapped</td>
<td>1</td>
<td>04/04/08</td>
</tr>
<tr>
<td>Link 3:17</td>
<td>1</td>
<td>04/04/08</td>
</tr>
<tr>
<td>European example 91</td>
<td>1</td>
<td>04/04/08</td>
</tr>
<tr>
<td>Kit member template case off 11/09</td>
<td>1</td>
<td>04/04/08</td>
</tr>
<tr>
<td>Kit member template case on 21/09</td>
<td>1</td>
<td>04/04/08</td>
</tr>
<tr>
<td>Link 1:19</td>
<td>1</td>
<td>04/04/08</td>
</tr>
<tr>
<td>Kit member template case off 11/09</td>
<td>1</td>
<td>04/04/08</td>
</tr>
<tr>
<td>Kit member template case on 21/09</td>
<td>1</td>
<td>04/04/08</td>
</tr>
</tbody>
</table>

To view a test's history, click the test name.

Screenshot 2: Top 10 Longest to Fix Tests

<table>
<thead>
<tr>
<th>Test</th>
<th>Average Time to Fix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freq</td>
<td>1 hour 2 minutes</td>
</tr>
<tr>
<td>Image canonical case on mapped</td>
<td>19 minutes</td>
</tr>
<tr>
<td>Link 3:17</td>
<td>19 minutes</td>
</tr>
<tr>
<td>European example 91</td>
<td>19 minutes</td>
</tr>
<tr>
<td>Kit member template case off 11/09</td>
<td>19 minutes</td>
</tr>
<tr>
<td>Kit member template case on 21/09</td>
<td>19 minutes</td>
</tr>
<tr>
<td>Link 1:19</td>
<td>19 minutes</td>
</tr>
<tr>
<td>Kit member template case off 11/09</td>
<td>19 minutes</td>
</tr>
<tr>
<td>Kit member template case on 21/09</td>
<td>19 minutes</td>
</tr>
<tr>
<td>Link 1:19</td>
<td>19 minutes</td>
</tr>
</tbody>
</table>

Screenshot 3: Top 10 Longest Running Tests
To compare the number of tests over time across multiple plans, view the ‘Number of Tests per Plan’ Report.
05. Reporting on Plan Trends

This page last changed on Feb 04, 2007 by rosie@atlassian.com.

5. Reporting on Trends

• **5.1 Viewing Build Statistics for a Plan**
• **5.2 Generating Reports across multiple Plans**
  ° 'Build Activity per Plan' Report
  ° 'Build Duration per Plan' Report
  ° 'Clover Code Coverage per Plan' Report
  ° 'Clover Lines of Code per Plan' Report
  ° 'Number of Build Failures per Plan' Report
  ° 'Number of Tests per Plan' Report
  ° 'Percentage of Successful Builds per Plan' Report
  ° 'Time to Fix per Plan' Report
5.1 Viewing Build Statistics for a Plan

To view a plan’s build statistics,

- From the Dashboard, locate and click a Plan Name from the list;
- OR:
- From within a build result, click the Plan Name at the top left of the screen.

The Plan Summary will be displayed as follows:

**Screenshot : 'Plan Summary'**

- Click the build number (i.e. 'BUCKET-MAIN-71') to view the build result.
- Click the 'Updated by' link to view the code changes that triggered the latest build result.
• Build duration is the total time taken to execute a build plan — that is, the time taken to compile the code and run all of the plan's tests. The large '%' box indicates the success rate of this plan's recent builds. This percentage is calculated on the last 25 builds, or as per your selection via the blue down-arrow:

Click the blue down-arrow to choose how you would like the percentage and graphs on this screen to be calculated. Choose from the following:

• this plan's last 25 builds.
• this plan's builds in the last 7 days.
• this plan's builds in the last 30 days.
• this plan's builds in the last 90 days.
• all of this plan's builds. The percentage and graphs on this screen will all be recalculated automatically when you choose a different option.

The 'Summary' tab provides a quick snapshot of the current status of the plan. For more details:

• Click the 'Activity' tab to view the plan's current activity.
• Click the 'Completed Builds' tab to view a list of build results for this plan's recent builds (i.e. the last 25 builds, or as per your selection via the blue down-arrow).
• Click the 'Tests' tab to view a summary of the test results for this plan's recent builds (i.e. the last 25 builds, or as per your selection via the blue down-arrow).
• Click the 'Files' tab to view a list of all the files currently contained in this plan's source-code repository. (Note: The 'Files' tab will only be visible if the plan's latest build ran on a local agent, since the files will not be available if the build ran on a remote agent.)
5.2 Generating Reports across multiple Plans

Bamboo provides a report generator that enables you to compare build statistics across one or more plans, using a variety of different metrics.

To report on build statistics per plan,

1. Click the 'Reports' link in the top navigation bar. This will display the 'Report Parameters' screen as shown below.
2. 'Report' — choose from the available reports. Available reports include:
   - 'Build Activity per Plan' Report
   - 'Build Duration per Plan' Report
   - 'Clover Code Coverage per Plan' Report
   - 'Clover Lines of Code per Plan' Report
   - 'Number of Build Failures per Plan' Report
   - 'Number of Tests per Plan' Report
   - 'Percentage of Successful Builds per Plan' Report
   - 'Time to Fix per Plan' Report

   Additionally, your Bamboo administrator may configure custom reports by using plugins. For details please see the Bamboo Administrator's Guide.
3. 'Build plans' — choose the plan(s) on which you want to report. You can use the <Ctrl> key to select multiple plans.
   - Project names are shown in italics, e.g. 'Geronimo SVN'. Plan names are shown in non-italics, e.g. 'Main Build'.
4. 'Group By' — choose whether your report's horizontal axis should show days, months or weeks. You can also specify 'Auto', which varies by report, but will generally default to 'week'.
5. 'Date Filter' — choose from:
   - 'All builds'
   - 'Last 7 days'
   - 'Last 30 days'
   - 'Last 90 days'
   - 'Select Range' — choosing this option will display two boxes in which you will need to specify the 'from' and 'to' dates (dd/MM/yyyy).
6. Click the 'Submit' button to generate your report.

Screenshot: 'Report Parameters--Build Plans'
'Build Activity per Plan' Report

Build activity is the number of builds that occur in a given period of time. You can choose the plan(s) and time period on which you want to report.

Sample Report: 'Build Activity per Plan'

[Graph showing build activity over time]
'Build Duration per Plan' Report

Build duration is the total time taken to execute a build plan — that is, the time taken to compile the code and run all of the plan's tests.

You can choose the plan(s) and time period on which you want to report.

Sample Report: 'Build Duration per Plan'

![Build Duration Chart](image-url)
'Clover Code Coverage per Plan' Report

This page last changed on Dec 13, 2007 by alui.

This report will only be available if your administrator has specified 'Clover output will be produced' in the plan's configuration. For details please see the Bamboo Administrator's Guide.

You can choose the plan(s) and time period on which you want to report.

Sample Report: 'Clover Code Coverage per Plan'

Clover Code Coverage

Comparing code coverage gives you an idea of how well the code base is tested. 100% coverage means that all code elements have been covered by your tests.
This report will only be available if your administrator has specified 'Clover output will be produced' in the plan's configuration. For details please see the Bamboo Administrator's Guide.

You can choose the plan(s) and time period on which you want to report.

Sample Report: 'Clover Lines of Code per Plan'

<table>
<thead>
<tr>
<th>Clover Lines of Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provides an indication of the size of the code base for the build.</td>
</tr>
</tbody>
</table>
'Number of Build Failures per Plan' Report

You can choose the plan(s) and time period on which you want to report.

Sample Report: 'Number of Build Failures per Plan'

![Graph showing number of build failures per plan over time.](image)

How many build failures are being tracked? A high value indicates a relatively unstable build that needs to be broken often.
'Number of Tests per Plan' Report

This page last changed on Dec 13, 2007 by alui.

You can choose the plan(s) and time period on which you want to report.

Sample Report: 'Number of Tests per Plan'

How many tests does your build have? This provides a rough indication of the level of testing over time for the build.
'Percentage of Successful Builds per Plan' Report

You can choose the plan(s) and time period on which you want to report.

Sample Report: 'Percentage of Successful Builds per Plan'

<table>
<thead>
<tr>
<th>Percentage of Successful Builds</th>
</tr>
</thead>
</table>

Comparing success percentages gives you an idea of how stable a build is compared to one another. 100% means your build is always perfect. 0% means something is always wrong.

![Chart showing percentage of successful builds over time.](image)

- Chart: Visual representation of successful build percentages over different time periods.
- Data Table: Detailed numerical data behind the chart.
- Graph: Detailed graph showing successful build percentages over time.
You can choose the plan(s) and time period on which you want to report.

**Sample Report: 'Time to Fix per Plan'**
06. Reporting on Author Trends

This page last changed on May 14, 2007 by rosie@atlassian.com.

6. Reporting on Author Trends

An author is any person who contributes to a build by checking-in code to a repository that is associated with a Bamboo plan. An author need not be a Bamboo user.

- **6.1 Viewing Build Statistics for all Authors**
- **6.2 Viewing Build Results for an Author**
- **6.3 Generating Reports on selected Authors**
  - 'Build Activity per Author' Report
  - 'Number of Build Failures per Author' Report
  - 'Number of Builds Broken per Author' Report
  - 'Number of Builds Fixed per Author' Report
  - 'Percentage of Successful Builds per Author' Report
6.1 Viewing Build Statistics for all Authors

This page last changed on Dec 13, 2007 by alui.

An author is any person who contributes to a build by checking-in code to a repository that is associated with a Bamboo plan. An author need not be a Bamboo user.

To view a summary of all authors' statistics,

1. Click the 'Authors' link in the top navigation bar.
2. This will display the following screen, where you can click any column-header to sort in ascending order (or click twice to sort in descending order).

Screenshot: 'Authors Summary-sorted by 'Fixed' (descending)'

<table>
<thead>
<tr>
<th>Name</th>
<th>Triggered</th>
<th>Failed</th>
<th>% Failed</th>
<th>Broken</th>
<th>Fixed</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jane Brown</td>
<td>962</td>
<td>275</td>
<td>28%</td>
<td>124</td>
<td>136</td>
<td>5</td>
</tr>
<tr>
<td>Max Smith</td>
<td>654</td>
<td>90</td>
<td>21%</td>
<td>47</td>
<td>43</td>
<td>3</td>
</tr>
<tr>
<td>Tom Brown</td>
<td>401</td>
<td>80</td>
<td>22%</td>
<td>30</td>
<td>40</td>
<td>10</td>
</tr>
<tr>
<td>Sally Johnson</td>
<td>422</td>
<td>102</td>
<td>24%</td>
<td>41</td>
<td>22</td>
<td>6</td>
</tr>
</tbody>
</table>

Handy Hint
You can click any author's name to see their recent build results.
6.2 Viewing Build Results for an Author

An author is any person who contributes to a build by checking-in code to a repository that is associated with a Bamboo plan. An author need not be a Bamboo user.

To view an author's build results,

1. Click the 'Authors' link in the top navigation bar.
2. This will display the 'Authors Summary' screen. Click the relevant author's name.
3. This will display the author's 'User Details' (email address, etc) — see screenshot below. Click through the following tabs to view recent build results:
   - 'Builds Summary' — a statistical summary of all the author's builds.
   - 'Last 10 Builds' — a list of the last 10 builds that were triggered by this author.
   - 'Last 10 Broken' — a list of the last 10 builds that were triggered by this author, where the build failed and the previous build for the same plan was successful.
   - 'Last 10 Fixed' — a list of the last 10 builds that were triggered by this author, where the build was successful and the previous build for the same plan failed.

If your Bamboo User Profile has not yet been associated with your Author Name, there will be no 'User Details' tab.

Screenshot 1: 'User Details' tab

<table>
<thead>
<tr>
<th>User Details</th>
<th>Build Summary</th>
<th>Last 10 Builds</th>
<th>Last 10 Broken</th>
<th>Last 10 Fixed</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Personal Details</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Username:</td>
<td>rot</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Full Name:</td>
<td>rot</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Email:</td>
<td><a href="mailto:rot@att.com">rot@att.com</a></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Auditor Address:</td>
<td><a href="mailto:rot@att.com">rot@att.com</a></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Org unit:</td>
<td>rot</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source Repository Aliases:</td>
<td>rot</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Screenshot 2: 'Builds Summary' tab

<table>
<thead>
<tr>
<th>User Details</th>
<th>Build Summary</th>
<th>Last 10 Builds</th>
<th>Last 10 Broken</th>
<th>Last 10 Fixed</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Builds</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Builds triggered by rot:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Builds triggered by user:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failed build:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Successful build:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Builds Summary</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Builds triggered by rot:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Builds triggered by user:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failed build:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Successful build:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Screenshot 3: 'Last 10 Builds' tab
## Screenshot 4: 'Last 10 Broken' tab

<table>
<thead>
<tr>
<th>Build</th>
<th>When</th>
<th>Comments</th>
<th>Test Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>B658499</td>
<td>4 days ago</td>
<td>Moved horror movie poster from the right location (GNAM-7260)</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>4 days ago</td>
<td>Fixed up stupid pointer to the saturation on edit added player (B658499)</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>2 days ago</td>
<td>Added script diff for database.</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>2 days ago</td>
<td>Fixed up wording of some comments, not needed</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>1 day ago</td>
<td>Added pull request... Might be worth checking out?</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>1 day ago</td>
<td>Updated the instructions for the image</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>6 days ago</td>
<td>Added some code for latest build template and the help page</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>6 days ago</td>
<td>Fixed up some fake data</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>6 days ago</td>
<td>Removed refused to track some fake template and the help text</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>6 days ago</td>
<td>Fix issues with <a href="https://github.com/issue_1234">source</a></td>
<td>[passed]</td>
</tr>
</tbody>
</table>

## Screenshot 5: 'Last 10 Fixed' tab

<table>
<thead>
<tr>
<th>Build</th>
<th>When</th>
<th>Comments</th>
<th>Test Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>B658499</td>
<td>2 days ago</td>
<td>Fixed some fake data in the fake build template and the fake text</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>2 days ago</td>
<td>You've gotta add the fake</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>2 days ago</td>
<td>Fixed some bugs in the fake build template and the fake text</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>2 days ago</td>
<td>Fixed some bugs in the fake build template and the fake text</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>2 days ago</td>
<td>Fixed some bugs in the fake build template and the fake text</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>1 day ago</td>
<td>Fixed some bugs in the fake build template and the fake text</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>1 day ago</td>
<td>Fixed some bugs in the fake build template and the fake text</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>1 day ago</td>
<td>Fixed some bugs in the fake build template and the fake text</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>1 day ago</td>
<td>Fixed some bugs in the fake build template and the fake text</td>
<td>[passed]</td>
</tr>
<tr>
<td>B658499</td>
<td>1 day ago</td>
<td>Fixed some bugs in the fake build template and the fake text</td>
<td>[passed]</td>
</tr>
</tbody>
</table>

Document generated by Confluence on Apr 14, 2008 01:44
6.3 Generating Reports on selected Authors

An author is any person who contributes to a build by checking-in code to a repository that is associated with a Bamboo plan. An author need not be a Bamboo user.

To generate a report on selected authors,

1. Click the 'Authors' link in the top navigation bar.
2. Click the 'Statistics' tab. This will display the 'Report Parameters' screen as shown below.
3. 'Report' — choose from the available reports. Available reports include:
   - 'Build Activity per Author' Report
   - 'Number of Build Failures per Author' Report
   - 'Number of Builds Broken per Author' Report
   - 'Number of Builds Fixed per Author' Report
   - 'Percentage of Successful Builds per Author' Report

   Additionally, your Bamboo administrator may configure custom reports by using plugins. For details please see the Bamboo Administrator's Guide.
4. 'Authors' — choose the author(s) on whom you want to report. You can use the <Ctrl> key to select multiple author.
5. 'Group By' — choose whether your report's horizontal axis should show days, months or weeks. You can also specify 'Auto', which varies by report, but will generally default to 'month'.
6. Click the 'Submit' button to generate your report.

Screenshot: 'Report Parameters--Authors'
'Build Activity per Author' Report

Build activity is the number of builds that occur in a given period of time. You can select the author(s) on whom you want to report.

Sample Report: 'Build Activity per Author'

![Build Activity Chart]

- Chart Title
- X-axis: Dates
- Y-axis: Number of Builds
- Authors: martha, thibogg

Document generated by Confluence on Apr 14, 2008 01:44
'Number of Build Failures per Author' Report

You can select the author(s) on whom you want to report.

Sample Report: 'Number of Build Failures per Author'
'Number of Builds Broken per Author' Report

You can select the author(s) on whom you want to report.

Sample Report: 'Number of Builds Broken per Author'
You can select the author(s) on whom you want to report.

Sample Report: 'Number of Builds Fixed per Author'
'Percentage of Successful Builds per Author' Report

This page last changed on Dec 13, 2007 by alui.

You can select the author(s) on whom you want to report.

Sample Report: 'Percentage of Successful Builds per Author'

![Chart showing percentage of successful builds per author over time]
07. Working with Comments

This page last changed on Feb 04, 2007 by rosie@atlassian.com.

7. Working with Comments

- 7.1 About Comments
- 7.2 Commenting about a Build Result
- 7.3 Viewing Comments about a Build Result
- 7.4 Viewing Code Check-in Comments
7.1 About Comments

Comments are a useful way to record and share information about builds. There are two types of comments in Bamboo:

- Comments you make when you commit code — these comments are automatically copied into Bamboo from your source-code repository. See Viewing Code Check-in Comments.
- Comments you make about a build result — these are comments that you make ad-hoc about a particular build result. See Commenting about a Build Result and Viewing Comments about a Build Result.
7.2 Commenting about a Build Result

Bamboo allows you to record comments about a build result. This is a convenient way to record relevant information for future reference, and to collaborate with colleagues.

To comment on a build result,

1. From within the 'Build Result' screen, click the 'Comments' tab. A list of existing comments about this build result will be displayed.
2. Type your comment into the 'Add Comment' box, then click the 'Save' button.

Screenshot: Build Result - Comments

You must login to Bamboo before you can comment on a build result.
7.3 Viewing Comments about a Build Result

To view comments about a particular build result,

1. From within the 'Build Result' screen, click the 'Comments' tab. A list of all comments about this build result will be displayed, including author and timestamp:

   Build Result BUCKET-MAIN-89

     Summary | Texts | Changes | Log | Comments | JIRA

     Comments
     - Smita Jamison (Jan 13, 2007 9:30 AM)
       This is another comment.
     - Matt Reay (Jan 11, 2007 12:27 PM)
       This is a reply to Smita's comment.
     - Edwina Wong (Jan 12, 2007 4:02 PM)
       This is a comment.

To view comments about all build results for a particular plan,

1. From the Dashboard, click the plan you are interested in.
2. Click the plan's 'Completed Builds' tab.
3. This will display a list of the plan's build results. The icon indicates that there are one or more comments about a particular build result. Hold your mouse over the comment(s), e.g.:
7.4 Viewing Code Check-in Comments

If a build was triggered by a code change, the commit comment (or check-in comment) will be shown in the build result.

To view the code check-in comments for a particular build result,

1. Go to the build result.
2. The build's commit comment will be shown to the right of the screen, under the heading 'Code Changes'.

A note about build triggering
There are a variety of ways in which a build can be triggered for a plan:

- Code updated — a build can be triggered whenever one or more authors checks-in code.
- Scheduled build — a build can be scheduled to occur at regular intervals.
- Dependency — a build can be triggered whenever a successful build occurs for another plan.
- Manual build — a build can be triggered manually.
- Initial clean build — a build will be triggered when a new plan is created.

The way in which each build was triggered is listed in the 'Reason' column on the Dashboard.
08. Working with Labels

8. Working with Labels

- 8.1 About Labels
- 8.2 Labelling a Build Result
- 8.3 Removing a Label from a Build Result
- 8.4 Viewing Labelled Build Results
- 8.5 Viewing Popular Labels
8.1 About Labels

A label is a convenient way to tag and group build results that are logically related to each other. Labels can also be used to define RSS feeds and to control build expiry.

Labels can be applied to build results automatically, by specifying the label(s) in a build plan (note that only Bamboo administrators can do this). Labels can also be applied ad hoc to build results by Bamboo users.
8.2 Labelling a Build Result

With Bamboo, you can label your build results in whatever way works best for your team. Labels are not restricted to a particular plan, so you can apply the same label to build results from different plans.

For example, it might not be practical for your QA team to review every build, and you need to know which builds they have reviewed. By using labels such as "qa_passed" and "qa_failed", Bamboo allows them to simply indicate which builds have passed and failed QA.

To label a build result,

1. Go to the build result.
2. Locate the 'Labels' link at the top of the screen:
   - Labels: NONE
   - Add
   (Hint: To view a list of existing labels, click the 'Labels' link.)
3. Click the 'Add' link. This will display the following:

   ![](image)

   The 'Add' link will only be visible if you have logged in to Bamboo.

4. Type the relevant label (or multiple labels, separated by commas). Note that the label will be saved in lowercase characters.
5. Click the 'Done' button.

You can also label a build result via Instant Messaging (IM).
8.3 Removing a Label from a Build Result

To remove a label from a build result,

1. Go to the build result.
2. Locate the 'Labels' link at the top of the screen.
3. Click the 'Edit' link. This will display the following:

   ![Label Editor](image)

   *Label as per build*

   Enter labels to add to this build

   [Done] [Cancel]

4. Click the small red 'x' at the right of the label you want to remove.
5. Click the 'Done' button.

You must login to Bamboo before you can remove a label from a build result.
8.4 Viewing Labelled Build Results

To view all build results which have a particular label,

1. Go to any build result.
2. Click the 'Labels' link at the top of the screen (above the 'Summary' tab).
3. Click the link 'See also labels in all projects'.
4. This will display a list of all labels that are used in Bamboo. Click the label of interest.
5. This will display a list of all build results which have that label.
8.5 Viewing Popular Labels

This page last changed on Dec 13, 2007 by alui.

When labelling a build result, it can be useful to see which labels are most popular, that is, most frequently used by your colleagues.

To view the most popular labels,

1. Go to any build result (not necessarily a labelled one).
2. Click the 'Labels' link at the top of the screen (above the 'Summary' tab).
3. Click the link 'See also labels in all projects'.
4. This will display a list all labels that are used in Bamboo. The most popular labels are indicated by the largest text.

Screenshot: 'Labels'

Handy Hint
You can click any label to see a list of all build results which have that label.
09. Subscribing to RSS Feeds

This page last changed on Feb 04, 2007 by rosie@atlassian.com.

9. Subscribing to RSS Feeds

- 9.1 Subscribing to an RSS Feed for All Build Results for All Plans
- 9.2 Subscribing to an RSS Feed for Failed Builds for All Plans
- 9.3 Subscribing to an RSS Feed for All Build Results for a Particular Plan
- 9.4 Subscribing to an RSS Feed for Failed Builds for a Particular Plan
- 9.5 Subscribing to an RSS Feed for Labelled Build Results
9.1 Subscribing to an RSS Feed for All Build Results for All Plans

To subscribe to an RSS feed for all build results for all plans,

1. Go to the Dashboard's 'All' tab.
2. Locate the RSS icon at the bottom of the screen:
3. Right-click the 'all builds' link and copy its URL.
4. Paste the URL into your RSS reader.
9.2 Subscribing to an RSS Feed for Failed Builds for All Plans

To subscribe to an RSS feed for failed builds for all plans,

1. Go to the Dashboard’s 'All' tab.
2. Locate the RSS icon at the bottom of the screen:
   ![RSS Feed Icon](rss_feed_icon.png)
3. Right-click the 'failed builds' link and copy its URL.
4. Paste the URL into your RSS reader.
9.3 Subscribing to an RSS Feed for All Build Results for a Particular Plan

To subscribe to an RSS feed for all build results for a particular plan,

1. Go to the plan.
2. Locate the RSS icon at the bottom of the screen:
3. Right-click the 'all builds' link and copy its URL.
4. Paste the URL into your RSS reader.
9.4 Subscribing to an RSS Feed for Failed Builds for a Particular Plan

To subscribe to an RSS feed for failed builds for a particular plan,

1. Go to the **plan**.
2. Locate the RSS icon at the bottom of the screen:
   ![RSS icon](rss_icon.png)
3. Right-click the 'failed builds' link and copy its URL.
4. Paste the URL into your RSS reader.
9.5 Subscribing to an RSS Feed for Labelled Build Results

To subscribe to an RSS feed for all build results with a particular label,

1. Go to the Dashboard.
2. Click any build result (not necessarily a labelled one).
3. Click the 'Labels' link at the top of the screen (above the 'Summary' tab).
4. This will display a list of any labels that are used in the build's plan. Click the link 'See also labels in all projects'.
5. This will display a list of all labels that are used in Bamboo. Click the label of interest.
6. This will display a list of build results which have been labelled with your chosen label. Locate the RSS icon at the bottom of the screen:
7. Right-click the 'Feed for builds labelled' link and copy its URL.
8. Paste the URL into your RSS reader.

What is a label?
10. Working with Instant Messenger (IM) Notifications

10. Working with Instant Messenger (IM) notifications

- 10.1 About Instant Messenger (IM) Notifications
- 10.2 Labelling a Build Result via IM
- 10.3 Commenting about a Build Result via IM
10.1 About Instant Messenger (IM) Notifications

Bamboo can send you notifications about build results for a particular plan(s). Each plan's recipients are specified by a Bamboo administrator, but you can choose whether you would like to receive your Bamboo notifications via email and/or instant messenger (IM). See 11.2 Changing your Notification Preferences.

As well as receiving IM notifications, you can interact with Bamboo via IM. By responding to an IM notification, you can:

- Label a build result via IM
- Comment about a build result via IM
10.2 Labelling a Build Result via IM

You can respond to a Bamboo IM notification message with commands to label or comment on a build result.

To label a build result via Instant Messaging (IM),

In your Instant Messenger, type your comment in the following format:

```
label [build key] <labels>
```

Screenshot: Interacting with Bamboo via IM

What is a label?
10.3 Commenting about a Build Result via IM

You can respond to a Bamboo IM notification message with commands to label or comment on a build result.

To comment on a build result via Instant Messaging (IM),

In your Instant Messenger, type your comment in the following format:

```
comment [build key] <comment message>
```

Screenshot: Interacting with Bamboo via IM
11. Editing your User Profile

11.1 Changing your Password
11.2 Changing your Notification Preferences
11.3 Associating your Author Name with your User Profile
11.1 Changing your Password

To change your Bamboo password,

1. Click the 'Profile' link in the top right corner of the screen. This will display the 'User Profile' screen.
2. Click the 'Change Password' link.
3. Type your old and new passwords.
4. Click the 'Change Password' button.
11.2 Changing your Notification Preferences

Bamboo can send you notifications about **build results**. You can choose whether to receive your notifications via email or [Instant Messaging (IM)](http://example.com), or both. You can also choose not to receive notifications at all.

To change your notification preferences,

1. Click the 'Profile' link in the top right corner of the screen. This will display the 'User Profile' screen.
2. Click the 'Edit Profile' link. The 'Edit User Profile' screen will be displayed, as shown below.
3. Under 'Set Your Notification Preference', choose how you would like to receive your notifications about build results:
   - 'Do not send notifications'
   - 'Send instant message'*
   - 'Send email'
   - 'Send email and instant message'*
4. Click the 'Save' button.

* If you select this option, you need to specify your IM address in the 'Jabber Address' field.

---

**Screenshot: User Profile**

**Edit User Profile**

<table>
<thead>
<tr>
<th>Username:</th>
<th>jsmith</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Name:</td>
<td>Joe Smith</td>
</tr>
<tr>
<td>Email:</td>
<td><a href="mailto:jsmith@newcompany.com">jsmith@newcompany.com</a></td>
</tr>
<tr>
<td>Jabber Address:</td>
<td><a href="mailto:jsmith@chat.mycompany.com">jsmith@chat.mycompany.com</a></td>
</tr>
</tbody>
</table>

**Set Your Notification Preference**

- [ ] Do not send notifications
- [ ] Send instant message
- [ ] Send email
- [ ] Send email and instant message

*Please ensure that you have set the appropriate address for the notification type.*

---

⚠️ Only Bamboo administrators can **enable notifications** for a plan. For details please see the [Bamboo Administrator's Guide](http://example.com).
11.3 Associating your Author Name with your User Profile

An author is any person who contributes to a build by checking-in code to a repository that is associated with a Bamboo plan. An author need not be a Bamboo user. Your Author Name is your login name for the source-code repository.

⚠️ If your Bamboo User Profile has not yet been associated with your Author Name, then:
- your 'My Bamboo' screen will not contain any data about your recent builds.
- your 'Author' information will not include a 'User Details' tab.

To associate your Author Name with your User Profile,

1. Click the 'Profile' link in the top right corner of the screen. This will display the 'User Profile' screen.
2. Click the 'Edit Profile' link.
3. In the 'Source Repository Alias' field, select your Author Name from the list. If your Author Name does not appear in the list, select 'Add Alias' (the second item in the list) then type your Author Name in the 'New Alias' field. Note that your Author Name (Alias) need not be identical to your User Name.
4. Click the 'Save' button.

Screenshot: Bamboo User Profile
Glossary

activity log
agent
agent-specific capability
artifact
author
build
build activity
build duration
builder
build log
build plan
build queue
build result
build telemetry
capability
child
committer
custom capability
favourites
global permission
label
local agent
permission
plan
plan permission
project
queue
reason
remote agent
requirement
shared capability
triggering
watcher
activity log

This page last changed on Jul 04, 2007 by rosie@atlassian.com.

Every plan has an activity log. An activity log is a temporary display of the latest output from the plan's most recent build log.
An agent is a service that runs Bamboo builds. There are two types of agents:

- local agents run on the Bamboo server.
- remote agents run on computers other than the Bamboo server.

(Note: Local agents run in the server's process, i.e. in the same JVM as the server. Each remote agent runs in its own process, i.e. has its own JVM.)

Each agent has a defined set of capabilities. An agent can only run builds for plans whose requirements are met by the agent's capabilities.

See 02. Configuring Agents and Capabilities and 1.2.4 Specifying a Plan's Capability Requirements.
An agent-specific capability applies to one agent only. Note that the value of an agent-specific capability will override the value of a shared capability of the same name (if one exists).

See 2.1 About Agents and Capabilities and 2.8 Configuring Capabilities.
artifact

This page last changed on Jul 05, 2007 by rosie@atlassian.com.

An artifact is something created by a build. There are two types of artifacts:

- User-defined artifacts (e.g. JAR files) are specified in the build's plan by a Bamboo administrator.
- Auto-generated artifacts are created automatically by Bamboo.
An author is any person who contributes to a build by checking-in code to a repository that is associated with a Bamboo plan. An author need not be a Bamboo user.

See 06. Reporting on Author Trends.
A build is one execution of a plan.

Every build has a Build Number, which is appended to the relevant Plan Key to form the Build Key. For example, if a plan with the key "CRM-BRANCH" is executed for the seventeenth time, the build key will be "CRM-BRANCH-17".
Build activity

Build activity is the number of builds that occur in a given period of time.
**build duration**

Build duration is the total time taken to execute a build plan — that is, the time taken to compile the code and run all of the plan's tests. Variations in a plan's build duration can be monitored over time.
A builder is a software compiler program external to Bamboo. Bamboo supports multiple builders. Once a builder is defined in the Bamboo system, it can then be specified in build plans by a Bamboo administrator.

See 2.8.1 Configuring a new Builder.
build log

This page last changed on Jan 29, 2007 by rosie@atlassian.com.

Every build has a build log. A build log is a permanent record of all the output generated by compiling the plan's source-code and executing the tests.
build plan

This page last changed on Feb 04, 2007 by rosie@atlassian.com.

See plan.
build queue

The Bamboo build queue controls the sequence of builds. When a plan submits a build to the build queue, the build will wait in the build queue until a suitable agent is available to run the build. The build queue is displayed on the Dashboard.
build result

This page last changed on Jan 28, 2007 by rosie@atlassian.com.

Every completed build has a build result:

- 'Successful' — the code compiled, with or without errors, and all tests completed successfully.
- 'Failed' — either the code did not compile, or at least one test failed.

Additionally,

- if the build result is 'Failed', and the previous build result was 'Successful', the build is said to be 'Broken'.
- if the build result is 'Successful', and the previous build result was 'Failed', the build is said to be 'Fixed'.


Build telemetry

This page last changed on Feb 04, 2007 by rosie@atlassian.com.

Build telemetry is the insight provided by Bamboo’s dynamic reports, charts and collation of build metrics. Build telemetry helps identify trends across build plans and across authors — not just focusing on the results of a single build.
A capability is a feature of an agent. A capability can be a:

- builder (e.g. Maven)
- JDK
- custom capability (a key-value property which defines a particular characteristic of an agent, e.g. 'operating.system=WindowsXP' or 'fast.builds=true')
- Perforce (location of the P4 client application, if Perforce is being used as the source repository)

Capabilities can be defined specifically for an agent, or they can be shared between either all local agents or all remote agents. Note that the value of an agent-specific capability overrides the value of a shared capability of the same name (if one exists).
See 2.8 Configuring Capabilities.
A child is a plan which gets triggered when another plan completes a build. See 3.4 Triggering a Build when another Build finishes.
A committer is the Bamboo user(s) who committed code to a particular build (i.e. someone who committed code after the previous build was checked out by Bamboo). Administrators can configure a plan’s notifications to be sent to the build's committer(s).
Custom capabilities can be used to control which build plans will be built by a particular agent. For example, if the builds for a particular plan should only run in a Windows environment, you could create a custom capability 'operating.system=WindowsXP' for the appropriate agent(s), and specify it as a requirement for this plan.

- To create a new custom capability in your Bamboo system, see 2.8.3 Configuring a new Custom Capability.
- To specify a build plan's requirement for a custom capability, see 1.2.4 Specifying a Plan's Capability Requirements.
favourites

This page last changed on Dec 11, 2007 by alui.

Each Bamboo user can nominate their favourite plans — that is, the plans they work with the most.

Each user's favourites are displayed on the 'My' page of the Dashboard. Bamboo administrators can also configure each plan to send build result notifications to users who have nominated the plan as one of their favourites (these users are known as the plan's watchers).
global permission

A global permission is the ability to perform a particular operation in relation to Bamboo as a whole. See 5.08 Granting Global Permissions to Users or Groups. See also plan permission.
A label is a convenient way to tag and group build results that are logically related to each other. Labels can also be used to define RSS feeds and to control build expiry.

Labels can be applied to build results automatically, by specifying the label(s) in a build plan (note that only Bamboo administrators can do this). Labels can also be applied ad hoc to build results by Bamboo users.
A parent is a plan which triggers another plan to build whenever it completes a build. See 3.4 Triggering a Build when another Build finishes.
permission

This page last changed on Jun 28, 2007 by rosie@atlassian.com.

See plan permission and global permission.
A Bamboo plan (or build plan) is the "recipe" for a build.

A plan defines: what gets built (i.e. the source-code repository); how the build is triggered; which builder to use; which agent capabilities are required for the build; what artifacts the build will produce; what tests to run; who will be notified of the build result; any labels with which the build result or build artifacts will be tagged; and who has permission to view and perform various actions on a plan and its build results.

Every plan belongs to a project. Each plan has a Plan Key, which is prefixed by the relevant Project Key. E.g. the "CRM" project could have plans "CRM-TRUNK" and "CRM-BRANCH". Projects and plans can only be configured by Bamboo administrators (see 1.2 Creating a Plan).
A plan permission is the ability to perform a particular operation in relation to a build plan. For each plan, different permissions can be granted to particular groups and/or users.

See 5.07 Granting Plan Permissions to Users or Groups.

See also global permission.
A project is a collection of plans.

A project enables easy identification of plans that are logically related to each other, which is useful for instance when generating reports across multiple plans. Each project has a Name (e.g. "CRM System") and a Key (e.g. "CRM"). The Project Key is prefixed to the relevant Plan Keys, e.g. the "CRM" project could have plans "CRM-TRUNK" and "CRM-BRANCH".
queue

This page last changed on Dec 12, 2007 by rosie@atlassian.com.

See build queue.
A build's reason is the way in which the build was triggered.

There are a variety of ways in which a build can be triggered for a plan:

- Code updated — a build can be triggered whenever one or more authors checks-in code.
- Scheduled build — a build can be scheduled to occur at regular intervals.
- Dependency — a build can be triggered whenever a successful build occurs for another plan.
- Manual build — a build can be triggered manually.
- Initial clean build — a build will be triggered when a new plan is created.

The way in which each build was triggered is listed in the 'Reason' column on the Dashboard.

Note that build triggering can only be configured by a Bamboo administrator. For more information please see 3.1 About Build Triggering.
remote agent

This page last changed on Dec 10, 2007 by rosie@atlassian.com.

See agent.
A requirement is an agent capability required by a build plan.

Together, capabilities and requirements control which agents can execute builds for particular plans. Each plan can only be built by agents whose capabilities meet the plan's requirements. Matching can be specified as either a regular expression or an exact match. See 1.2.4 Specifying a Plan's Capability Requirements.
Shared capabilities are inherited by all applicable agents, that is, (shared) local server capabilities are inherited by all local agents, and shared remote capabilities are inherited by all remote agents. Note, however, that the value of a shared capability will be overridden by the value of an agent-specific capability of the same name (if one exists).
See 2.1 About Agents and Capabilities and 2.8 Configuring Capabilities.
triggering

There are a variety of ways in which a build can be triggered for a plan:

- Code updated — a build can be triggered whenever one or more authors checks-in code.
- Scheduled build — a build can be scheduled to occur at regular intervals.
- Dependency — a build can be triggered whenever a successful build occurs for another plan.
- Manual build — a build can be triggered manually.
- Initial clean build — a build will be triggered when a new plan is created.

The way in which each build was triggered is listed in the 'Reason' column on the Dashboard.

Note that build triggering can only be configured by a Bamboo administrator. For more information please see 3.1 About Build Triggering.
A plan's watchers are the Bamboo users who have marked this plan as one of their favourites. Administrators can configure a plan's notifications to be sent to the plan's watchers.
Bamboo 1.2 Bookmarks

This page is a container for all the bookmarks in this space. Do not delete or move it or you will lose all your bookmarks.

Bookmarks in Bamboo 1.2 | Links for Bamboo 1.2

The 15 most recent bookmarks in Bamboo 2.0

There are no bookmarks to display.
How are builds distributed to agents?

Plan ACME submits build number 123 to the Build Queue.

The Build Queue forwards build no. ACME-123 to the next available agent whose capabilities meet Plan ACME's requirements.
How do capabilities affect the distribution of builds to agents?

Plan ACME

Capability requirements
Builder = Maven
JDK = 1.5
Custom: fast build = true

Build no. ACME-123

Plan ACME submits build number 123 to the Build Queue

Build Queue

The Build Queue forwards build no. ACME-123 to the next available agent whose capabilities meet Plan ACME’s requirements. (Note: agent-specific capabilities override shared capabilities.)

BAMBOO SERVER
Local Agent 1
Local Agent 2

Remote Agent 1

Remote Agent 2

Remote Agent 3

Agent-specific capabilities
JDK = 1.5
Custom: fast build = true

Shared local capabilities
Builder = Ant
JDK = 1.3

Shared remote capabilities
Builder = Maven
JDK = 1.4
Bamboo 2.0 has now been released. Read the full Bamboo 2.0 Release Notes and Upgrade Guide.
TreeNavigation

This page last changed on Aug 09, 2007 by rosie@atlassian.com.

Index