

Upgrade and Migration Guide

You can upgrade to this version of Clearspace with these guidelines for each distribution.

Important Note Make sure to [backup your jiveHome directory](#) (page 4) and [backup your database](#) (page 4) before doing an upgrade.

Sections in this Topic

[Upgrading the Standalone Distribution](#) (page 1)

[Upgrading the WAR Installation](#) (page 2)

[Upgrading the Source Build](#) (page 2)

[Migrating from One DBMS to Another](#) (page 3)

[Back Up Your jiveHome Directory](#) (page 4)

[Back Up Your Database](#) (page 4)

[Remove Plugins Before Upgrading](#) (page 4)

[Clearspace Upgrade Philosophy](#) (page 4)

[Known Issues](#) (page 5)

Upgrading the Standalone Distribution

1. Verify you've backed up your [jiveHome directory](#) (page 4) and [database](#) (page 4) .
2. Backup your CLEARSPACE_HOME/server/lib directory (or CLEARSPACE_HOME/server/shared/lib on versions prior to version 2) or, alternatively, after unzipping your new Clearspace download, you will need to copy the appropriate JDBC driver into the server/lib directory. In either case the JDBC driver needs to be in this directory before restarting the standalone instance.
3. Unpack the new distribution of Clearspace.
4. Rename the jiveHome/ directory that comes with this distribution to something other than jiveHome, you won't be using this directory (for example, `mv jiveHome/ jiveHomeUnused/`).
5. Create a symbolic link inside the unpacked distribution to your permanent jiveHome:
 - **UNIX/Linux:** Create a symbolic link to your permanent jiveHome directory in the Clearspace directory called jiveHome (for example, `ln -s /opt/jiveHome jiveHome`)
 - **Windows:** You can also create a symbolic link to your jiveHome directory if you have Junction installed (<http://www.microsoft.com/technet/sysinternals/utilities/Junction.mspx>), or copy your permanent jiveHome into the Clearspace directory.
6. You will need to copy your database JDBC driver back into the server/lib directory before restarting the standalone Clearspace instance
7. Issue the command `start-clearspace.sh` or `start-clearspace.bat`

Your new version of Clearspace should be up and running using your previously configured database and customized settings.

Upgrading the WAR Installation

1. Verify you've backed up your [jiveHome directory](#) (page 4) and [database](#) (page 4) . (If you're using MySQL, be sure to see the [note under known issues](#) (page 5) .)
2. Delete or move your previous WAR (and exploded WAR files) from your application server's web application directory (usually webapps/).
3. Unpack the new distribution of Clearspace.
4. You will need to point your Clearspace instance at your permanent jiveHome directory (which you have already backed up), this can be done in a few different ways:
 - **Option 1: Set with a system property (advanced)**. Set the location of the jiveHome directory manually by passing in a Java system property to your appserver. Set a property with the name of "jiveHome". Most app servers allow you to pass in an environment variable in the startup script. That might look like this:

```
java -DjiveHome=/usr/foo/jiveHome -cp . com.myappserver.Server
```

- **Option 2: Use the EditWAR tool**. The EditWAR tool is a small application which will modify the packaged web application (clearspace.war) to point to the jiveHome directory. The clearspace.war file is also in the base directory of this distribution. To invoke the EditWAR tool, open a command prompt in the base directory of this distribution and execute this command:

```
java -jar EditWAR.jar clearspace.war
```

The tool will then lead you through the process of updating the WAR.

- **Option 3: Set a JNDI value (advanced)**. Set the location of the jiveHome directory via JNDI. In your app server, you can set a JNDI value of java:comp/env/jiveHome with a String value of the path to your jiveHome directory.
5. Copy the new clearspace.war into application server's web application directory, or redeploy the WAR using your application server's administration console.
 6. Your new version of Clearspace should be up and running using your previously configured database and customized settings.

Upgrading the Source Build

1. Verify you've backed up your [jiveHome directory](#) (page 4) , [database](#) (page 4) , and Clearspace_HOME/custom directory if you have added any of you own custom source code.
2. Delete or move your previous WAR (and exploded WAR files) from your application server's web application directory (usually webapps/).
3. Unpack the new distribution of Clearspace.
4. Rebuild the Clearspace source as detailed in the Clearspace Source Build.
5. You will need to point your Clearspace instance at your permanent jiveHome directory (the one you backed up), this can be done in a few different ways:
 - **Option 1: Set with a system property (advanced)**. Set the location of the jiveHome directory manually by passing in a Java system property to your appserver. Set a property with the name of

"jiveHome". Most app servers allow you to pass in an environment variable in the startup script. That might look like this:

```
java -DjiveHome=/usr/foo/jiveHome -cp . com.myappserver.Server
```

- **Option 2: Use the EditWAR tool.** The EditWAR tool is a small application which will modify the packaged web application (clearspace.war) to point to the jiveHome directory. The clearspace.war file is also in the base directory of this distribution. To invoke the EditWAR tool, open a command prompt in the base directory of this distribution and execute this command:

```
java -jar EditWAR.jar clearspace.war
```

The tool will then lead you through the process of updating the WAR.

- **Option 3: Set a JNDI value (advanced).** Set the location of the jiveHome directory via JNDI. In your app server, you can set a JNDI value of java:comp/env/jiveHome with a String value of the path to your jiveHome directory.
6. Redeploy the new clearspace.war into application server's web application directory, or redeploy the WAR using your application server's administration console.
 7. Your new version of Clearspace should be up and running using your previously configured database and customized settings.

Migrating from One DBMS to Another

You can use the Clearspace admin console to migrate Clearspace data from one database management system to another. Clearspace supports the following DBMSs as either source or destination systems (supported versions are those that are supported for installation; see the Clearspace Installation Guide for more details):

- MySQL
- PostgreSQL

In the admin console, go to **System > Database Migration > Database Migration**. The database migration tool will migrate your Clearspace instance from one database server to another. Before beginning the migration, you must:

- Ensure that the **Clearspace instance is not needed** during migration. It will be unavailable while this tool is running.
- Ensure that the **destination database has no tables or data**. This tool will create tables and delete data in the destination database.
- Ensure that **you haven't customized the source database** you're migrating from. This tool might not work at all if your source database doesn't conform to the default schema. For customizations made by Jive's professional services team, please contact your service representative before migrating. For other issues, please contact Jive's support team.
- Copy the **correct driver for the new database** to the application classpath (such as the Clearspace WEB-INF/lib directory).
- **Be prepared to restart Clearspace** after migrating.

Note: If you begin migration but don't finish, it's possible to put Clearspace into a in-between state. In this state, you'll receive an "in-progress" message even though you're no longer migrating. If this

happens, you can fix it in the admin console. In the console, go to System > Management > System Properties, scroll to locate the `skin.default.maintenace.enabled` property, then click its delete button. After you do this you should be able to refresh your browser to view the Clearspace UI.

Back Up Your jiveHome Directory

The jiveHome directory is the place where Clearspace stores a number of things about your environment. The database connection information is stored there as well as logs, cached attachments, your license file, and the evaluation database files (if used).

You should back this up before upgrading and you should get in the practice of backing up this directory on a regular basis. The easiest way to back up the directory is to use a ZIP or TAR application or script to package the directory.

Back Up Your Database

You should back up your database on a regular basis and before upgrades. For now, the best way to manage database backups is to follow the recommendations of your DBA or the recommendations of your database software. There are a number of tools built in to various databases. Here are a few example:

- MySQL — Use the "mysqldump" tool.
- Postgres — Use the "pg_dump" tool.

There are many tools for each database; try to pick one that suits your environment.

Remove Plugins Before Upgrading

Before starting your upgrade be sure to remove any plugins you've installed. For those plugins that aren't compatible with the version you're upgrading to, you'll need to separately upgrade your plugin code (or get upgraded versions of the plugins from their developer), then install the upgraded versions after you've upgraded Clearspace.

Clearspace Upgrade Philosophy

Upgrading web applications doesn't have to be hard. We've made every effort for this to be a seamless process. If you have [feedback](#) or suggestions for this process, we'd love to hear it.

In general, upgrading between revision releases will be easy (for example, 2.5.1 to 2.5.2). Upgrades between minor releases (for example, 2.0 to 2.5) might require a bit more work, but we have an upgrade wizard in the application to assist.

Known Issues

Upgrades Using MySQL Will Convert MyISAM Tables to InnoDB

In order to support transactions in version 2, MyISAM MySQL database tables will be converted to InnoDB during upgrade. The upgrade process will handle this conversion; there is no action required on your part.

Clear the Tomcat work Directory After Upgrading

After you finish the upgrade process, be sure to clear the clearspace subdirectory of Tomcat work directory before starting the application and viewing the admin console. You'll likely find this subdirectory at `<tomcat_root>/work/.../clearspace`. If you don't clear the directory (which contains cached application components), you might see errors when you try to view application pages.

Default Code Markup is Now Plain Instead of Java

Prior to version 2, using the `{code}` macro without a property would result in Java-style syntax coloring. As of version 2.5, text that was enclosed in `{code}` tags will be rendered merely as plain monospace text, rather than with Java syntax coloring.

Version 2.1 replaces the plain text editor, which supported wiki markup, with a more full-featured rich text editor. Here's how to change code markup back to Java syntax coloring beginning with version 2.5:

1. Begin editing the content in the new editor.
2. Right-click the code you want to change, then click **Format > Java**.

Clear Moderation Queues Before Upgrading

The moderation-related settings made in versions prior to 2.5.0 aren't respected after upgrade to a 2.5.x version. To avoid problems, you should clear all moderation queues in the earlier version *before* upgrading (such as by approving or rejecting items in the queue).

Upgrading from 2.0.x to 2.5.x Might Truncate Content of Formatted Text Widgets

The upgrade process converts contents of formatted text widgets to HTML from wiki markup (which isn't used in 2.5.x). This increases the number of characters in the widget, sometimes beyond its 3500-character limit. In cases where the limit is exceeded, the widget's text is truncated to meet the limit. Before upgrading, you can address this by shortening the widget's contents. After upgrade, you'll find the text of truncated widgets in the log file, where you can use it to update the upgraded widgets.