Upgrading Rhythmyx 5.x to Rhythmyx 6.5.2

The Upgrade Process

This document describes the system requirements for upgrading to Rhythmyx Version 6.5.2, outlines the new features introduced in the upgrade, and describes the tasks required to returns your server to stability after the upgrade.

Note that you are not required to use all new features after the upgrade. Once your return your server to stability, you can generally continue to operate your server as before. You can then convert to using the new features at your leisure. For details about implemeting the new features introduced in Rhythmyx Version 6.5.2, see the document *Converting to Rhythmxy Version 6.5.2 Features*.

The recommended upgrade process includes the following tasks:

1. Pre-upgrade maintenance
   a) Clean up your implementation
      - Remove log files. (Log files are stored in `<Rhythmyxroot>/AppServer/WebApps/logs`.)
      - Remove trace files. (Trace files are stored in `<RhythmyxRoot\<application directory>~<appname>.trace`)
      - Remove unused applications. (including FastForward applications that were never used)
      - Remove unused Variant registrations associated with applications that were removed.
      - Clean up orphaned Variant registrations.

   b) Back up your Rhythmyx tree and Repository database.
2. Run the upgrade wizard.
3. Stabilize your server.

Hardware Requirements for Upgrade to Rhythmyx 6.0 and Later

Recommended minimum system requirements for all operating systems have increased:

	Development Server	**Production Server**
Processor	1 GHz	2GHz
RAM	512 MB	1 GB
Disk Space	750 MB	1 GB (plus additional space for application data)

Rhythmyx Version 6.0 is certified on the following operating systems:

- Windows
  - 2000 (Professional, Advanced Server, Datacenter Server)
  - XP
  - 2003 Server and Advanced Server
- Solaris
  - 9
  - 10
- Linux
  - Red Hat Linux AS 3.0
  - Red Hat Linux AS 4.0

The following RDBM systems are supported

- Microsoft SQL Server
  - 2000
  - 2005
- Oracle
  - 9i
  - 10g
  - (Oracle Version 8 is no longer supported)
- IBM DB2 UDB
  - 8.2

The minimum recommended version of the Java Runtime Engine browser plugin is version 1.5.0_08.

New Features Introduced in Rhythmyx Version 6.0 and Their Effect on Upgrades

Beginning in Rhythmyx Version 6.0, the Rhythmyx server was ported to run as a servlet within the JBoss Web application server. The Rhythmyx Publisher, WebDAV servlet, and Web Services servlet also run in the JBoss container. Rhythmyx no longer requires a separate Web application server for these servlets, so they no longer run in a separate Tomcat Web application server. The Rhythmyx server and all associated servlets now start and stop together when the JBoss container starts or stops. Existing WebDAV configurations will not be migrated to the servlet in JBoss; you must manually configure the new WebDAV servlet to match previous configurations. Note that the WebDAV servlet can no longer be deployed remotely to other containers.

During the upgrade, the existing AppServer directory will be preserved and renamed to AppServer.bak. Existing servlets will be preserved so you can migrate customizations to the new servlets installed with JBoss.

In Rhythmyx 5.7 and earlier, standard practice when adding custom extension classes was to store the class and .jar files in the directory `<Rhythmyxroot>/lib` or `<Rhythmyxroot>/libextras`. This practice is not valid in Rhythmyx Version 6.0 because the JBoss server has its own classloader and it does not reference .cp files. Any custom extensions files should be moved to the following directories:

- Class files should be moved to the directory rxapp.war/WEB-INF/classes.
- Java archive files (.jar files) should be moved to the directory rxapp.war/WEB-INF/lib.

Note that any .cp files that exist in the installation directory will be moved to the directory <Rhythmyxroot>/upgrade/rx_cp_files during the upgrade process.

The JBoss server uses JNDI datasources to define database connections. This functionality supersedes the database connection definitions previously defined in Rhythmyx. The installation wizard collects data used to define the default JNDI datasource used to connect to the Rhythmyx Repository. A Datasources tab has been added to the Rhythmyx server administrator to provide the ability to define and maintain datasources. All dialogs that previously defined database connectivity data have been refactored to specify a datasource.

When using datasources, application-level database credentials are not valid. The installation wizard scans the installation for application-level database credentials; if any are found, the installation terminates. All application-level database credentials must be removed before launching the upgrade process. Best practice is to define a system-level database connection corresponding to each application-level connection. The installation process will create a datasource for each system database connection detected.

If you use database publishing, the JNDI datasources specified in Tomcat are superseded by datasources defined in JBoss. These datasources are not migrated during the upgrade. You will have to redefine them manually in JBoss. You should also convert your database publishing Variants to Templates as soon as possible. (Variants for publishing files can be converted later.) For details about converting Variants to Templates, see Converting _to_Version_6_5_Features.pdf. For details about writing Templates for database publishing, see "Database Publishing in Rhythmyx" in the *Rhythmyx Implementation Guide*.

Several methods of the class PSDatabasePool, which is used to obtain database connections, cannot be supported under JBoss and will be removed during the upgrade. The class as a whole is being deprecated and extensions should instead use methods of the class PSConnectionHelper.

In Rhythmyx 5.7 and earlier, the working directory was the Rhythmyx root directory. In Rhythmyx 6.0 and later, the working directory changes to `<Rhythmyxroot>/AppServer/bin`. Any custom implementations that rely on the Rhythmyx root directory as the working directory will continue to work, but this practice was deprecated in Rhythmyx Version 6.0 and will not be supported in the future. Any customizations that rely on the Rhythmyx root directory as the working directory should be re-implemented to use the new working directory as soon as possible.

If you use SSL, and your keystore file is stored within the Rhythmyx installation directory, the keystore file will be moved to the following location:

```
<Rhythmyxroot>/AppServer/server/rx/conf
```

Note that any certificates imported into the Java cacerts file must be re-imported into the cacerts file of the new JRE installed with the Rhythmyx server.

Design Object Naming Policy Change

In Rhythmyx Version 5.7 and earlier, spaces were valid in the names of Rhythmyx design objects, as were the following characters:

Name	Character
Ampersand	**&**
Apostrophe	`
At sign	**@**
Backslash	\
Caret	^
Colon	**:**
Curly braces	**{ }**
Equals sign	=
Forward slash	/
Greater than sign [or left angle bracket]	<
Less than sign [or right angle bracket]	>
Percent sign	**%**
Period or dot	**.**
Pipe	|
Pound sign (also known as number sign, hash, or octothorp)	**#**
Question mark	**?**
Semicolon	**;**
Square brackets	**[ ]**

Tilde	~

In Rhythmyx Version 6.0, these characters are no longer valid in the names of design objects. During the upgrade, these characters are replaced with underscores in the names of all design objects. (NOTE: In Rhythmyx Version 6.0, each design object also has a separate Label, which can include the restricted characters.)

After the upgrade, review your extension code and other customizations to ensure that references to design objects conform to the new naming requirements. Also review applications, especially Auomated Index applications, for references to design objects and update those references to conform to the new naming policy.

New Feature of Rhythmyx Version 6.5

Rhythmyx Version 6.5 introduces one new feature, a new interface for Active Assembly with enhanced usability. This new interface is only available for Active Assembly based on Velocity Templates. XSL Variants must be converted to Velocity Templates to make this interface available. It is automatically available when the standard Rhythmyx Velocity macros are used. If you define custom Velocity macros, you must incorporate special macros into your custom macros to make the new interface functionality available. For details about these macros, see “Velocity in Rhythmyx” in the *Rhythmyx Technical Reference Manual*.

Upgrade to sys_EditLive Control

In Rhythmyx Version 6.5.2, the Ephox EditLive rich text editor is upgraded to EditLive Version 6.3. In Content Editors, users will notice that fields using the sys_EditLive control are not available until they click in the field.

The sys_EditLiveDynamic control is deprecated in this release. Any field using the sys_EditLiveDynamic control should be modified to use the sys_EditLive control.

The sys_EditLiveDynamic control should not be used in new Content Editor fields (although it is still listed as an available control for fields in the Workbench).

Before Launching the Upgrade Installation

Before launching the upgrade installation:

- Review applications for local backend credentials, which will cause the installation to abort. Remove any local backend credentials and create a corresponding system-level credential.
- Determine how far back you want to preserve publication logs (publogs). During the installation, publication logs will be purged. You can determine the oldest date for which to preserve the logs. Any logs older than the date you specify will be permanently purged from the system and cannot be recovered.

Setting Up an Upgrade Implementation Environment

Best practice is to implement the upgrade in a separate development environment from your production environment. Make all changes in this environment and ensure before implementing them in your production environment.

To set up an upgrade implementation environment:

4. Use a copy of your production server tree and a dump of your production database.
5. Set up a Web server to match your production environment
6. Prune dead code before upgrading this environment:
    - Delete any unused Content Types and Variants
    - Deactivate any unused support applications. Confirm that the system works as expected without these applications, then delete them.
    - Delete any application that does not start.
7. Test publishing and ensure that the system works as expected BEFORE beginning to implement the conversion.
    - All Content Editors work as designed (test on both new and existing Content Items).
    - All Variants preview correctly and all Snippets are assembled into Pages correctly.
    - Publishing runs are completed without errors, all expected content is published to the expected locations.
8. You can now proceed to upgrade this system. Upgrade the system, and document any changes required to restore system stability. You can then apply the same changes to your production system.

Restoring System Stability

Once you complete the upgrade installation, perform the following procedures to restore system stability:

- During the upgrade, restricted characters in the names of the design objects are replaced with underscores. Review extension code and other customizations for design object names and change any restricted characters to underscores. For details about restricted characters, see “Design Object Naming Policy Change” on page 4.
- During the upgrade, existing server registrations in Multi-server Manager are purged. Re-register you servers with Multi-server Manager.
- During the upgrade, a change is made to the configuration of the full-text search engine. This modification requires that content be re-indexed after the upgrade is complete. To re-index content, submit the command `search index type [id]` where *id* is the Content Type ID of the Content Type you want to index. If you do not specify a value for `[id]`, all Content Types will be indexed. Note that on a system with a large volume of content, indexing can take several hours. Consider this fact when scheduling upgrade installations.
- Any extensions that use methods of the class PSDatabasePool should be rewritten to use methods of PSConnectionHelper instead. The following methods of PSDatabasePool are removed on upgrade:
  - public String getDefaultServer()
  - public Connection getConnection(String driver, String server, String database, long waitMS) throws SQLException
  - public Connection getConnection(String driver, String server, String database, String uid, String password, long waitMS) throws SQLException
  - public static String prepareCredentials(String uid, String password)

  The remaining methods will continue to work as before:
  - public Connection getConnection() throws SQLException
  - public String getDefaultDriver()
  - public String getDefaultDatabase()
  - public String getDefaultSchema()
  - public void releaseConnection(Connection conn)

  The class PSDatabasePool is deprecated, however, and its methods should be replaced with methods from the class PSConnectionHelper:
  - public static java.sql.Connection getDbConnection(IPSConnectionInfo info)
  - public static PSConnectionDetail getConnectionDetail(IPSConnectionInfo info)
- If you use the local Publisher installed with Rhythmyx, check the Publisher configuration and confirm that the value of the Port parameter was updated to the Rhythmyx port (9992 by default) instead of the Rhythmyx application server port (9980 by default). The Publisher now runs in the same container as the Rhythmyx server and uses the same port.
- If you use Database Publishing, reconfigure the JNDI datasources specifying the output database in JBoss.
- If you have customized log4j configurations, you will need to redefine those configurations in the file AppServer/server/rx/conf/log4j.xml.

- If you use WebDAV, existing configurations will not be migrated to the new WebDAV file. These configurations must be manually migrated to the new format and location. For details, see *Implementing WebDAV*.
- If you use SSL, re-import your certificates into the cacerts file of the new server JRE.
- If you have customized the web.xml of the Publisher, you must reapply these customizations to the new Publisher servlet.
- In Rhythmyx Version 6.0 and later, the name of the Rhythmyx Service is changed to "Percussion Rhythmyx Server". This name is not changed during upgrades. However, if an installation of an earlier version Rhythmyx is deleted, and the existing Windows service is not deleted or disabled, followed by installation of Rhythmyx Version 6.0 to the same location as the earlier version, both services will attempt to start. The two services will interfere with one another and neither will function properly. To avoid this problem, delete or disable the service for the earlier version of Rhythmyx.
- If your implementation includes a Display Format that displays thumbnail previews of you images, you will need to copy the contents of the file `rxconfig/Server/sys_AddThumbnailURL.properties` into the file `rxconfig/Server/addThumbnailURL.properties`.
- During upgrade, Display Mappings are created for several system fields, but no Display Mapping is created for the field sys_pubdate. Consequently, this field does not appear in the list of available system fields in the Rhythmyx Workbench. To include this field in the list, add the following Display Mapping to ContentEditorSystemDef.xml:

```
<PSXDisplayMapping>
  <FieldRef>sys_pubdate</FieldRef>
    <PSXUISet>
      <Label>
        <PSXDisplayText>Pub Date:</PSXDisplayText>
      </Label>
      <PSXControlRef id="0"
         name="sys_CalendarSimple"/>
    </PSXUISet>
</PSXDisplayMapping>
```

System Operations after Upgrade

After upgrade, the following operational changes should be noted:

- The JBoss server terminal window is not interactive. Commands to server cannot be issued directly in the terminal window. Instead, a JSP page is provided:

  ```
  http://localhost:9992/Rhythmyx/admin/console.jsp
  ```

  Where

  localhost is the name or IP address of the host machine of the Rhythmyx server; and

  9992 is the Rhythmyx server port.

  Access to this page requires authentication. Most server commands can also be issued from the Server Administrator. (EXCEPTION: The `quit` command cannot be issued from the Server Administrator. The `quit` command can only be issued from the administration JSP.) The command `dump databasepools` is no longer available. A new command, `dump datasources`, is available. For a complete list of the server commands available, see the Server Administrator Help.
- The JBoss server shuts down immediately on a fatal error. No interactive pause is available.
- The Rhythmyx Server Administrator is not backwards compatible. The Rhythmyx Server Administrator installed with Rhythmyx Version 6.0 and later cannot connect to a Rhythmyx Version 5.7 server or a server of any earlier version. The Rhythmyx Server Administrator installed with Rhythmyx Version 5.7 or earlier cannot connect to a Rhythmyx server of Version 6.0 or later.
- Use of the Rhythmyx root directory as the working directory is deprecated. Custom implementations that require the working directory to be the Rhythmyx root directory should be re-implemented to use the directory <Rhythmyxroot>/AppServer/bin as the working directory as quickly as possible. In a future release, Rhythmyx will no longer support the use of the Rhythmyx root directory as the working directory.
- The Rhythmyx WebDAV servlet can no longer be deployed remotely or in other servlet containers, such as Tomcat, BEA WebLogic, or IBM Websphere.
- After upgrading, the console log (<Rhythmyxroot>/ console.log) should be reviewed to ensure that all Content Editors have started correctly.
- System Keywords cannot be deployed properly and should not be included in Multi-Server Manager deployment archives. Modifications to system Keywords should be manually recreated on the target server.
- The FastForward descriptors in this release do not include a modification adding the Site Folder Content Authorization Type to the target server. This Authorization Type must be added to the target server manually when deploying implementations based on FastForward. Note that Site Folder Publishing in FastForward will not work properly without this Keyword included on the server.
- In this release, several new Java interfaces were added for extensions specific to Content Editors (such as Field Validations, Visibility Rules, Input and Output Transformers). For example, in Rhythmyx Version 5.7 and earlier, a Field Validation was a UDF. In Rhythmyx Version 6.5, a Field Validation is treated specifically as a Field Validation extension. This functionality was added to simplify the lists of available extensions in the Content Editor dialogs of the Rhythmyx Workbench.

For backwards-compatability, a preference has been added to the Rhythmyx Workbench, Show legacy interfaces for Content Type extensions. When this preference is checked, the Rhythmyx Workbench lists both the Content Type extensions and UDFs in the Content Editor dialogs. The preference is checked by default when upgrading from Rhythmyx Version 5.7 or earlier.

Extension code should be updated to use the correct Java interfaces:

Content Type Extension	**Interface**
Field Input Transformer	com.percussion.extension.IPSFieldInputTransformer
Field Output Transformer	com.percussion.extension.IPSFieldOutputTransformer
Field Read Only Rule	com.percussion.extension.IPSFieldEditabilityRule
Field Validations	com.percussion.extension.IPSFieldValidator
Field Visibilty Rules	com.percussion.extension.IPSFieldVisibilityRule
Item Input Transformer	com.percussion.extension.IPSItemInputTransformer
Item Ouptut Transformer	om.percussion.extension.IPSItemOutputTransformer
Item Validation	com.percussion.extension.IPSItemValidator

After updating the extension code, you should uncheck the Show legacy interfaces for Content Type extensions preference.

Additional Upgrade Information for Customers Upgrading from Rhythmyx Version 5.5 and Rhythmyx Version 5.6

In Rhythmyx Version 5.7, a new rich text DHTML editor was introduced, Ephox EditLive! for Java. This control supersedes the Ektron eWebEditPro rich text DHTML editor.

Customer intervention is required to use the EditLive! control. You do not have to replace eWebEditPro with EditLive! You can continue to use eWebEditPro if you prefer. Note, that support for e-WebEditPro was terminated in November 2006 and use of this control is now deprecated.

NOTE: You cannot use both the EditLive! control and the eWebEditPro control in the same Content Type. The two controls are incompatible, and you must use one or the other consistently within any Content Type. You can, however, use the EditLive! control in one Content Type and the eWebEditPro control in a different Content Type.

To change the rich text DHTML editor for a field from the eWebEditPro control to the EditLive! control:

1. In the Rhythmyx Workbench, open the Content Type in which you want to change the rich text DHTML editor.

   Rhythmyx displays the Content Type Editor.
2. In the row of the field you want to change, double-click in the Control column and from the drop list choose *sys_EditLive*.
3. Rhythmyx automatically sets common parameters to the same values used for the sys_eWebEditPro control that was used for the field.

4 If you want to use a customized configuration file, or modify other parameters:

c) Click the browse button next to the Control field.

Rhythmyx displays the Display Control Properties for <field> dialog.

d) Enter the parameters and associated values you want to assign to the control.

e) Click [**OK**] to save your edits.

5 On the Field Properties dialog, click [**OK**] to save the modifications you made to the field.

The changes take effect immediately (except for currently open browser sessions, which must be closed and restarted).

Additional Upgrade Information for Customers Upgrading from Rhythmyx Version 5.0x

The following features introduced in Rhythmyx Version 5.5 require customer intervention to gain full benefit:

- Full-text search engine
- WebDAV
- Directory Services
- Accessibility enhancements
- Revised HTML Search pages

See also the discussion of the EditLive! rich text DHTML editor control in the section "Additional Upgrade Information for Customers Upgrading from Rhythmyx Version 5.5 and Rhythmyx Version 5.6" above.

Full-text Search Engine

The Convera RetrievalWare full-text search engine is an optional feature you can install to provide a robust, full-text alternative to the light-weight database search engine available with Rhythmyx. If you choose to install the full-text search engine, you will need to index your Content before searches will return results. Indexing can take several hours. Consider this fact when scheduling your upgrade installation.

To create search indices, enter the command:

```
search index type id
```

Where *id* is the Content Type ID of the Content Type you want to index. If you do not provide a value (in other words if you only enter the command `search index type`) all Content Types in your system will be indexed.

Any existing saved searches created using the light-weight database search engine will continue to exist until you delete them manually. Once these saved searches are deleted, you will not be able to recreate them.

The existing saved searches will continue to use the original, light-weight search functionality, but no full-text search functionality will be available with them. To take advantage of the full-text search functionality, you must create new saved searches after upgrading to the full-text search engine.

WebDAV

Rhythmyx Version 5.5 introduced support for WebDAV to link Rhythmyx Folders to working directories in your network. Using WebDAV, Content Contributors can automatically manage image- and file-type Content Items simply by working with them in their system directories. The system directories are linked to Rhythmyx Folders by WebDAV servlets.

WebDAV requires some manual implementation to link the system directories to the Rhythmyx Folders. For details, see *Implementing WebDAV in Rhythmyx*.

WebImageFX

Rhythmyx Version 5.5 incorporated the WebImageFX graphics editor from Ektron to provide basic image editing functionality. The WebImageFX editor, like the eWebEditPro DHTML text editor, is available as a Content Editor Control that can be added to Image Content Types. This control must be added to a Content Editor to make it available to Content Contributors.

Directory Services

Directory services provides functionality that allows you to recover user attributes from a directory server, such as an LDAP server or a Microsoft ActiveDirectory server. You can use these attributes as part of user authentication, or use them as data for Rhythmyx processing.

To use Directory Services, you must define directory services configurations in the Rhythmyx Server Administrator. For details about implementing Directory Services, see “Implementing a Directory Services Security Provider” in the *Rhythmyx Implementation Guide* or “Maintaining Directory Services Configurations” Server Administrator Help.

Accessibility Enhancements

Rhythmyx Version 5.5 introduced a number of enhancements to improve accessibility for users with disabilities. Most of these enhancements require no additional implementation. An enhancement has been added to the Content Editor field properties, however, that allows you to add an accelerator key to each field in a Content Editor. The topic “Implementing the ‘shared’ Field Set” in the *Rhythmyx Implementation Guide* describes how to specify the accelerator (or mnemonic). See also “Field Properties Editor” in the Rhythmyx Workbench Help.

Revised HTML Search Pages

New Search pages are available for the searches launched from Active Assembly, or when inserting inline links. These searches are rendered using HTML rather than using the search applet. The new search pages have been refactored to enhance their accessibility to users requiring assistive devices.

When you upgrade, the inline search page is automatically upgraded to use the new search page, but the Active Assembly search page is not. If you want to use the new pages, you must update the Active Assembly search component to use the correct Rhythmyx application. To update the Active Assembly search component:

1. In Content Explorer, choose the System tab.
2. In the Left Navigation, under the Components option, click the By Name link.
3. Click on the rcsearch link. (You may have to scroll through a few pages of component listings to reach this link.)
4. Change the value in the **URL** field to *../sys_searchSupport/getQuery.html*.

5. Click the [Save] button.
6. In the Workbench, restart the sys_rcSupport application.

If you want to return to the original Active Assembly search page, use the same procedure, but in step 4., change the value in the URL field to *../sys_relatedSearch/relatedsearch.html*.

Upgrade Checklist

- Before upgrade
  - Review applications for local backend credentials and remove any found.
  - Determine how far back to preserve publication logs.
- After upgrade
  - Review extensions for design object references that include spaces and updated with underscores.
  - Re-register servers in Multi-server Manager
  - Update extensions that use methods of PSDatabasePool to use PSConnectionHelper methods instead.
  - Update publisher configurations to use the Rhythmyx port rather than the application server port.
  - If using database publishing, reconfigure JDNI datasources.
  - Redefine log4j configurations.
  - Migrate WebDAV configurations.
  - Re-import SSL certificates.
  - Re-apply customizations for Publisher web.xml.
  - Add Display Mapping for sys_PubDate system field.
  - Convert custom implementations that use the Rhythmyx root directory as the working to directory so they use <Rhythmyxroot>/AppServer/bin.
  - Check console.log to ensure all Content Editors have started correctly.