# Percussion Software
# AjaxSwing Content Browser Configuration Guide
# Rhythmyx 7.3 (requires patch 730_20150405)

**What is AjaxSwing?**
AjaxSwing is a commercial product from CreamTec that allows Java Swing Applets and AWT applications to run on a server. The users can interact with these server applications using regular JavaScript technologies without the need for a Java Runtime being installed on the client machine. Further reading on AjaxSwing: http://www.creamtec.com/products/ajaxswing/.

**Why use AjaxSwing?**
Customers have concerns over Java installed on client machines due to recent Java security issues. Although we patch our applet to the most recent and secure Java version, there is a general trend towards removing access to Java in the browser. AjaxSwing allows for a JavaScript based content browser that's built using existing Applet code. This plugin enabled us to provide our customers a quicker and effective solution for replacing Java Applet based content browser, without which would have been massive re-architecture.

**How does AjaxSwing work?**
AjaxSwing wraps an existing applet into its own JVM process. It intercepts responses to and from the base applet and Java Swing UI, converting actions like clicks from the browser into events within the applet. The Java graphical paint requests and responses are then converted into HTML and JavaScript equivalents. Each client accessing the server is connected to a unique instance of the applet on the server. AjaxSwing can handle multiple applet instances for each JVM instance, and can also be configured to automatically start new JVM processes to handle more loads.

**AjaxSwing Performance and Scalability:**
AjaxSwing is as scalable as the current Rhythmyx Applet based content browser and the server you are running it on, and as fast as original implementation. It adds

a thin layer that replaces the Java graphical environment and emulates the events. In many aspects it can make the Rhythmyx content browser load faster because no drawing, buffering and output to the screen occurs in the emulated mode. There are fewer events to process and less memory needs to be allocated and garbage collected. There may be some performance improvements but this mostly depends on the network infrastructure and the bandwidth between the AjaxSwing and Rhythmyx server. There are many factors involved though; the AjaxSwing server is now doing the work that would previously have been on the client's machine, and thus the server must have the resources to handle the required number of users.

Reference:
www.creamtec.com/products/ajaxswing/doc/AjaxSwingScalabilityPerformance.html

**Installing AjaxSwing:**
AjaxSwing runs within its own web container outside of Rhythmyx. AjaxSwing comes with its own embedded Tomcat instance that can be used or can be installed to an existing servlet container like Tomcat. AjaxSwing is packaged in multiple formats which include a cross-platform .zip and a Windows GUI installer. The built-in Apache Tomcat web server gives everything you need to deploy and run your applications. The fastest way to get started is to use the preconfigured embedded Tomcat, even if the target production environment runs on a different servlet container.

**Conventions used in following instructions:**
The following when referenced in the document should be replaced with the path of respective installation folders.

1. RHYTHMYX_HOME: The Rhythmyx installation location.
2. AJAXSWING_HOME: The AjaxSwing installation location.
3. JAVA_HOME: The Java1.6 JDK installation location

**AjaxSwing recommended requirements on installation and running:**

| | |
|---|---|
| **Platform** | **Any platform supported by JDKs listed below. Tested on Windows, Linux, HP UX, Sun Solaris, IBM AIX** |

| | |
|---|---|
| JDK(Java Development Kit) | 1.5, 1.6 or 1.7 from Sun Microsystems |
| Servlet Container | Any application or web server that supports Servlet 2.2 or higher API. |
| Disk space | 25 MB |

The following installation instructions from Percussion were tested against the following platforms with JDK1.6 only:

1. RPM (Centos 6.5)
2. Debian (Ubuntu 12.04.3 LTS)
3. Windows 7 Professional 64 bit

**Step 1: Prerequisites**

1. **Supported Rhythmyx and Patch:** AjaxSwing is fully supported only in Rhythmyx version 7.3 and patch starting 730_20150405.

2. **Additional libraries in Linux and Ubuntu:** Install 32bit libraries if doesn't exists
   a. **sudo yum install libXtst-***
   b. **sudo yum install glibc.i686**(use **apt-get** in Ubuntu)

3. **Setup Java JDK 1.6**: AjaxSwing depends on specific version of JDK. We require JDK1.6 for version 4.2.1 of AjaxSwing.
   **a. For Windows server:**
      i. Download and install JDK 64 bit in server machine: jdk-6u45-windows-x64.exe

   **b. For Centos server:**
      i. Download jdk-6u45-linux-x64-rpm.bin from Oracle link: jdk-6u45-linux-x64-rpm.bin
      ii. Create directory "java" under "/usr" if one doesn't exists: sudo mkdir /usr/java

iii. Move “jdk-6u45-linux-x64-rpm.bin” to “/usr/java” folder:
sudo mv jdk-6u45-linux-x64.bin /usr/java
iv. Change execute permission for jdk-6u45-linux-x64-rpm.bin file if doesn’t exists: chmod +x jdk-6u45-linux-x64-rpm.bin
v. Install JDK: sudo ./jdk-6u45-linux-x64-rpm.bin
vi. Verify that JDK is installed under folder /usr/java/jdk1.6.0_45

c. **For Ubuntu server:**
i. Download jdk-6u45-linux-x64-rpm.bin from Oracle link:
jdk-6u45-linux-x64.bin
ii. Create directory “java” under “/usr” if one doesn’t exists: sudo mkdir /usr/java
iii. Move “jdk-6u45-linux-x64.bin” to “/usr/java” folder:
sudo mv jdk-6u45-linux-x64.bin /usr/java
iv. Change execute permission for jdk-6u45-linux-x64.bin file if doesn’t exists: chmod +x jdk-6u45-linux-x64.bin
v. Install JDK: sudo ./ jdk-6u45-linux-x64.bin
vi. Verify that JDK is installed under folder /usr/java/jdk1.6.0_45

4. **AjaxSwing licensing**:
AjaxSwing trial instance does not require AjaxSwing licensing but it’s restricted to only 5 users at any given time. Details for commercial licensing can be found here: http://www.creamtec.com/products/ajaxswing/prices.html

## Step 2: Ensure a graphics environment is available

### XWindows Environment or Xvfb (X virtual frame buffer)

AjaxSwing requires a graphical environment on the server to access fonts and to draw elements. Some Linux servers may not have XWindows environment installed by default. If XWindows environment cannot be installed, then we must use a virtual frame buffer to draw to and make sure that the required fonts are installed. A small package available on most environments called Xvfb which can provide this required environment without the overhead and security implications of a full desktop install.

The following commands can be used to install a dummy screen for AjaxSwing to draw if a full XWindows installation does not exist. If needed, use “sudo”, or run as “root” user. The script and configurations will allow Xvfbd to be automatically started on server startup and will run on display number 55. This display number is chosen to not conflict with any existing displays on the server, so this process could be used even if a display is already available.

i. **For Centos/RHEL**:

1. **sudo yum install Xvfb xorg-x11-fonts* chkconfig**
2. **cp Xvfbd from /sys_resources/ajaxswing/xvfbd to /etc/init.d/**
3. **sudo chmod 755 /etc/init.d/Xvfbd**
4. **sudo /sbin/chkconfig --add Xvfbd**
5. **sudo /sbin/chkconfig --level 2345 Xvfbd on**
6. **sudo /etc/init.d/Xvfbd start**

ii. **For Ubuntu:**

1. **cd /etc/init.d**
2. **cp /sys_resources/ajaxswing/xvfbd/Xvfbd to /etc/init.d/**
3. **sudo chmod 755 /etc/init.d/Xvfbd**
4. **sudo apt-get update**
5. **sudo apt-get install xvfb**
6. **sudo apt-get install xfonts***
7. **sudo update-rc.d Xvfbd defaults**
8. **sudo update-rc.d Xvfbd start 20 3 4 5**
9. **sudo /etc/init.d/Xvfbd start**

To verify whether the Xvfb service is running in CentOS or Ubuntu, use “grep” command as **: ps -ef |grep Xvfb**

The result should similar to as shown below:

```
root   4609   1 0 18:06 pts/2  00:00:00 /usr/bin/Xvfb :55 -screen 0
1600x1200x24+32
```

iii. **For Windows**

**Windows is provided with a graphics environment, therefore no additional installation required.**

**Step 3 – Install AjaxSwing:**

1. **Download latest AjaxSwing package**: Download latest AjaxSwing package from CreamTec Downloads. The installation instructions listed in this document is tested using AjaxSwing version 4.2.1. Download links for AjaxSwingv4.2.1 for **All platforms:** ajaxswing421.zip

2. **Install AjaxSwing**: Extract the zip file in any folder (AJAXSWING_HOME). Example /home/ajaxswing in Linux or c:/ajaxswing in windows machine.

3. **Copy AjaxSwing related files from Rhythmyx server to AjaxSwing folder:**

| **#** | **From RHYTHMYX_HOME /sys_resources/ajaxswing** | **To AJAXSWING_HOME** | **File** |
|---|---|---|---|
| **1** | /lib | /lib | plugin.jar |
| **2** | /wcapps/lib | /wcapps/lib (Note: this step need to be repeated every time applet is refreshed in a Rhythmyx patch) | ajax-swing-rxcx.jar |
| **3** | /conf | /conf | CMSystem.properties |

| | | | |
|---|---|---|---|
| **4** | /conf/templates/default-page | /conf/templates/default-page | page-header.html |
| **5** | /tomcat/webapps/ajaxswing/scripts | /tomcat/webapps/ajaxswing/scripts(please create this folder if doesn’t exists) | custom.js |

**Step 4 – AjaxSwing Configuration:**

1. Update “applet.codeBase” AJAXSWING_HOME/conf/CMSystem.properties file to point to Rhythmyx server and port.

   **applet.codeBase=http\://server\:port/Rhythmyx**

2. Unix/Linux specific configurations:

   **a.** Set **JAVA_HOME** and **PATH** environment variables:
   - **export JAVA_HOME=/usr/java/jdk1.6.0_45**
   - **export PATH=${PATH}:${JAVA_HOME}/bin**

   **b.** Update **JAVA_HOME**, **AJAXSWING_HOME** and **DISPLAY** values in AJAXSWING_HOME/bin/**setEnv.sh**

   For **JAVA_HOME** and **AJAXSWING_HOME:**

   By default **JAVA_HOME** and **AJAXSWING_HOME** configurations are commented out in setEnv.sh file. Please uncomment JAVA_HOME and AJAXSWING_HOME to update with appropriate JAVA_HOME and AJAXSWING_HOME installation location.

   For **DISPLAY:**

If using Xvfb, update DISPLAY value to 55.0. If not using Xvfb or already have a windows environment the DISPLAY value need not be updated.

Original configurations from setEnv.sh file:

```
# export JAVA_HOME=/usr/jdk1.3
# export AJAXSWING_HOME=/usr/AjaxSwing4.2.1
DISPLAY=:0.0
export DISPLAY
```

Sample updated configuration:

```
export JAVA_HOME=/usr/java/jdk1.6.0_45
export AJAXSWING_HOME=/home/ ajaxswing/AjaxSwing4.2.1
DISPLAY=:55.0
export DISPLAY
```

**c.** Add **JAVA_HOME** value in AJAXSWING_HOME/bin/**clientAgent.sh** file:
Add "**export JAVA_HOME=<JAVA_HOME>**" next to "#! /bin/sh" statement.

Example configuration:

```
#! /bin/sh
export JAVA_HOME=/usr/java/jdk1.6.0_45
```

**d.** Update following properties in AJAXSWING_HOME/conf/**default.properties** file as shown below.
   i. **router.inProcess=false**
   ii. **router.updateInterval=60**
   iii. **router.clientsPerJVM=1**
   iv. **agent.useAjaxSwingFontsConfigFile=false**
   v. **agent.usePlatformFontConfiguration=true**

**Note: Whenever these configurations are updated, the "setupForXvfb.sh" script from following section need to run without that would throw exception as listed in "Known Issues" section of this document.**

**e.** Run following commands/scripts to setup AjaxSwing, JAVA and file permissions.
   i. **cd AJAXSWING_HOME/bin/setup**
   ii. **chmod +x setup.sh**
   iii. **./setup.sh**
   iv. **./enableJdk16.sh**
   v. **./setupForXvfb.sh**

**f.** **Unzip** AJAXSWING_HOME/tomcat/webapps/ajaxswing.war file
Please run following command under AJAXSWING_HOME/tomcat/webapps/ajaxswing.war folder
   i. **unzip -d ajaxswing ajaxswing.war (**choose 'n' for the prompt for whether to overwrite ajaxswing/scripts/custom.js file**)**
   ii. **rm ajaxswing.war**

**g.** **Update "ajaxswing.home" in:**
AJAXSWING_HOME/tomcat/webapps/ajaxswing.war/WEB-INF/**web.xml**
Remove the comment from <init-param> node and configure the ajaxswing.home value to the AJAXSWING_HOME location.

Original configuration:
**<!--**

```
<init-param>
 <param-name>ajaxswing.home</param-name>
 <param-value>c:/AjaxSwing4.2.1</param-value>
</init-param>
-->
```

Example configuration:

```
<init-param>
        <param-name>ajaxswing.home</param-name>
        <param-value>/home/ajaxswing/AjaxSwing4.2.1</param-
value>
</init-param>
```

3. **Windows specific configurations:**
   a. Create a **JAVA_HOME** environment variable with the Java JDK1.6 installation path using “Control Panel / System / Advanced system settings / Environment Variables”
   b. Update the following properties in AJAXSWING_HOME/conf/**default.properties** file as shown below.
      i. **router.inProcess=false**
      ii. **router.updateInterval=60**
      iii. **router.clientsPerJVM=1**

   c. Update **JAVA_HOME** and **AJAXSWING_HOME** values in AJAXSWING_HOME/bin/**setEnv.bat** file

      Remove the comment to JAVA_HOME and AJAXSWING_HOME statements as shown below to update the JAVA and AjaxSwing installation location

      Original statements:

```
rem set JAVA_HOME=C:\Java\jdk1.5
rem set AJAXSWING_HOME=C:\Java\AjaxSwing
```

      Sample updated statements:

**set JAVA_HOME=C:\Java\jdk1.6.0_45**
**set AJAXSWING_HOME= C:\ajaxswing\AjaxSwing4.2.1**

**d.** Run the following batch files:
   i. **AJAXSWING_HOME\bin\setEnv.bat**
   ii. **AJAXSWING_HOME\bin\installAjaxSwingService.bat**
   iii. **AJAXSWING_HOME\bin\setup\enableJDK16.bat**

**Note: After executing the batch files, please confirm that the AjaxSwing Server starts from Windows services. Please stop the server after confirming that the service starts successfully.**

**e. Extract** AJAXSWING_HOME/tomcat/webapps/ajaxswing.war file
   i. Extract AJAXSWING_HOME/tomcat/webapps/ajaxswing.war to AJAXSWING_HOME/tomcat/webapps/ajaxswing folder (Make sure that ajaxswing/scripts/custom.js file is not overwritten - choose “no” to replace this file when extracting the war file)

**f.** Update ” ajaxswing.home” in AJAXSWING_HOME\tomcat\webapps\ajaxswing.war\WEB-INF\**web.xml**

Remove comment from <init-param> node and configure ajaxswing.home value to AJAXSWING_HOME location.

Original configuration:
```
<!--
 <init-param>
  <param-name>ajaxswing.home</param-name>
  <param-value>c:/AjaxSwing4.2.1</param-value>
 </init-param>
 →
```

Sample configuration:
```
<init-param>
```

```
        <param-name>ajaxswing.home</param-name>
        <param-value> C:/ajaxswing/AjaxSwing4.2.1</param-
    value>
    </init-param>
```

**Step 5: Start and test AjaxSwing service:**

**To start:**

- **In Linux:**
  Run AJAXSWING_HOME/bin/startServer.sh

- **In Windows:**
  Start AjaxSwingServer service in Windows services.

  Note: The embedded AjaxSwing default tomcat port is 8040 whereas the default tomcat port is 8080. Please make sure that this port is properly mapped if the AjaxSwing server is behind a firewall or proxy.

**To test:**

Use either of these URLs to test AjaxSwing:

a. AjaxSwing demo application:
   - http://server:ajaxswingport/ajaxswing/apps/SwingSet2

b. Rhythmyx AjaxSwing application:
   - http://server:ajaxswingport/ajaxswing/apps/CMSystem

   Expected Test Result: A gray box.

**Step 6: Enable AjaxSwing in Rhythmyx**

1. Update server.properties file:
   To enable AjaxSwing in Rhythmyx Server, add the following properties to

the Rhythmyx/rxconfig/Server/**server.properties** file.

i. **ajaxSwingEnabled=true**
ii. **ajaxSwingHost=<ajaxswingserver>**
iii. **ajaxSwingPort=8040**

**Note: The configured host and port need to be publicly accessible.**

2. Database update:
   Currently the AjaxSwing integration through patch doesn't support database table update which is required for persisting specific user's preference for the type of UI (Applet or AjaxSwing). To accomplish this, we need to manually run a SQL script against the Rhythmyx database to add a new CATEGORY "**sys_ui**" in the "**PSX_PERSISTEDPROPERTYMETA**" table. Please run the following SQL statement against Rhythmyx database:

```
INSERT INTO "PSX_PERSISTEDPROPERTYMETA"
(PROPERTYID, USERNAME, PROPERTYNAME,
CATEGORY, PROPERTYSAVETYPE, ENABLEDSTATE,
OVERRIDABLE) VALUES ('4', '**psxsystem', 'sys_ui',
'sys_session', '1', '1', '1')
```

   This "sys_ui" category allows storing a specific user's UI preferences upon loading the Content Explorer.

3. Restart Rhythmyx
4. Test Rhythmyx Applet Content Browser: http://server:port/Rhythmyx
5. Test Rhythmyx AjaxSwing Content Browser:
   http://server:port/Rhythmyx?sys_ui=ajaxswing

   **Note:**
   - Can use sys_ui=applet in Rhythmyx url for Applet based content browser.
     Eg: http://server:port/Rhythmyx?sys_ui=applet
   - Both Applet and AjaxSwing can be used in parallel as long as

"ajaxSwingEnabled" parameter in server.properties is set to "true". Setting this property to "false" would disable the AjaxSwing based content browser.

- Please note that the user must manually change their default UI by accessing the Rhythmyx URL with sys_ui parameter. This selection is stored in the user's profile on the server so the option will persist between logins.

**Step 7: Optional configuration**

- **Logging in AjaxSwing:**

  AjaxSwing logs are created under AJAXSWING_HOME/logs directory. A separate log file is created for each instance of the content explorer. The logging level for AjaxSwing can be adjusted between 0 and 10. Note however that setting logging level to a higher level will create a lot of logs which will use a significant amount of disk space as well as causing performance issues.

  To change the logging level, update com.creamtec.core.TraceMgr.level in: AJAXSWING_HOME/conf/TraceMgr.properties file. Default level is 2.

  Example update: com.creamtec.core.TraceMgr.level = 6

- **Rhythmyx Content Editor:**

  The Applet based Content Editor - EditLive can be replaced with JavaScript based Content Editor – TinyMCE. The instructions for converting existing EditLive content editor to TinyMCE is available on the Percussion help site : Implementing TinyMCE Control in Rhythmyx

**Known Issues**

1. User drag and drop to copy or move items in Content Explorer may not work as expected. The issue is that, the UI wouldn't paint as if the item is

being dragged to move or copy any selected item(s).

2. Impact Analysis menu and Administrator tool in workflow tab load using Applet even with AjaxSwing enabled. These features are not yet converted to AjaxSwing. Future updates will address this. The user without a JRE installed should use the client end Developer Tools to access server administrator.
3. Dialogs (workflow related, Help, About Rhythmyx etc) are now handled by AjaxSwing and as such cannot be dragged out of the scope of the current Content Browser window. The Content Browser window may need to be resized large enough in order to see the full dialog.
4. The domain name configured as "**applet.codeBase**" in AJAXSWING_HOME/conf/CMSystem.properties file must match the URL used to access the Rhythmyx server. Mismatch between domain name and IP when accessing the Rhythmyx UI would fail due to cross domain security issue. Example, if a domain name is used in "**applet.codeBase**" configuration but IP address is used to load preview of an item, would fail.
5. Whenever configuration in "AJAXSWING_HOME/conf/**default.properties**" is updated, the "setupForXvfb.sh" script from section (**Step 4 – AjaxSwing Configuration**) needs to run. If it is not run, a trace exception would result (as listed below) and the AjaxSwing GUI would not render as expected.

   **Sample exception trace from AjaxSwing server log:**

```
java.lang.Error: Could not instantiate Graphics Environment:
sun.awt.X11GraphicsEnvironment)
2014/06/23 13:56:31:610 :
[com.creamtec.ajaxswing.core.ClientAgentFactory$1] Exception while
warming up client agent for application cmsystem
java.rmi.ServerError: Error occurred in server thread; nested exception is:
java.lang.Error: Could not instantiate Graphics Environment:
sun.awt.X11GraphicsEnvironment
    at sun.rmi.server.UnicastServerRef.dispatch(UnicastServerRef.java:331)
    at sun.rmi.transport.Transport$1.run(Transport.java:159)
    at java.security.AccessController.doPrivileged(Native Method)
    at sun.rmi.transport.Transport.serviceCall(Transport.java:155)
```