

CREATE NEW JAVA RED5 APPLICATION

January 29, 2007 at 10:33 pm · Filed under [Red5](#)

During this tutorial, we will be using Joachim's migration guide:

<http://www.joachim-bauch.de/tutorials/red5/MigrationGuide.txt>

In this tutorial we will be examining how to create applications in the Red5 Standalone Server. The Red5 Standalone Server uses a Spring Framework Managed Jetty Embeddable Http Server and Servlet container. The Spring Framework is a JAVA Framework that provides hooks (configuration metadata is in the form of either XML bean definitions, JAVA properties file or by calling the configurable items of the Spring API from within your POJO—Plain Old JAVA Object) through which you can control the behaviour of various JAVA applications, objects or frameworks (eg Jetty, Acegi, Hibernate). This allows you to leverage a broad array of JAVA technologies from within your POJO without having to face the challenge of coding to interface to all these JAVA objects yourself while still ensuring you adhere the common JAVA Design Patterns. In addition, by making use of the hooks provided by Spring to influence your application's behaviour by editing the configuration metadata (See Dependency Injection), you can easily change the behaviour of your application while leaving your core POJOs intact. Because of IoC, one's application can be developed using a set of simple JavaBeans (or plain-old Java objects—POJOs), with a a lightweight IoC container (Spring) wiring them together and setting the dependencies.

The XML configuration metadata and is used by Spring's IoC to create a fully configured system or application. This configuration metadata tells the Spring container how to "instantiate, configure, and assemble [the objects in your application]".

In addition, you will also learn about the scope of a red5 application, as well as how to create and use server side shared objects.

Before beginning this tutorial, you may download the red5java application explained from here:

[red5java.tar.gz](#)

Extract the contents of this folder into your red5_server/webapps directory.

and the demo flex2 client from here:

[red5_flexclient.tar.gz](#)

If you are having problems downloading from the links above, please try downloading from the link below:

<http://jwamicha.mediamax.com/downloads>

1. Get into the root directory of the red5 server which in my case was red5_server

```
#cd /opt/red5_server/webapps
```

This is the root folder of all your red5 application definitions, from which all red5 applications are started. This is the global scope. If you are running a hosting service this would be your root folder. To create a section for each hosted user you would create a *WEB-INF/user-dir* as shown below;

2. Create your new application as follows (The new application name for this tutorial will be called red5java)

```
#mkdir -p red5java/WEB-INF
```

Pages »

[About](#)

Archives »

[April 2007](#)

[February 2007](#)

[January 2007](#)

[December 2006](#)

Categories »

[Kannel](#)

[Linux TV](#)

[Oracle](#)

[Red5](#)

Search »

Meta »

[Log in](#)

[Entries RSS](#)

[Comments RSS](#)

[WordPress.com](#)

re5java is the name of your application in this example. You will access your application using the URL *rtmp://your-ip/red5java* or *rtmpt://your-ip/red5java* in the case of a proxy-tunneled RTMPT connection). The WEB-INF folder is so that Jetty (the Embeddable HTTP Server and Servlet Container that is used by the Standalone red5 server) knows where to access your application (Servlet) from. This is because your application is actually a JAVA servlet. It is in the WEB-INF file that you will load all the configuration files/beans that will be required to load your application. These files specify all the Spring beans that your application will require as well as bean definitions that will be used by Spring to Dependency Inject your application. These config files include *web.xml*, *red5-web.xml*, *red5-web.properties* and *log4j.properties* as will be shown below.

Under the red5java directory create a folder called src where you will create your own custom application that will be used to communicate with the Flash clients.

```
#mkdir src
```

This would also be a good point at which to create your classes (for the classes) and lib (for the jar file) directories. We will also be doing this in the ant build file below but so that we get the concept, lets create the directories here;

```
#mkdir classes && mkdir lib
```

We now need to create a package for our Application. Lets call it

```
org.red5.webapps.red5java
```

```
#cd src
```

```
#mkdir -p org/red5/webapps/red5java
```

Your sources will go into the org/red5/webapps/red5java directory.

[*TODO: Verify and test the statement below*] red5java represents your "application scope" under the "global scope" ie webapps. Sub-folders under red5java would represent your "room scope" ie all subfolders under red5java are subsopes of the red5java application scope.

If you are on Windows, you can create your directory structure from within eclipse, so that it will look something like this. After you create the src directory, under WEB-INF, right click src directory then *New>Package* and create a package so that it has the structure shown below (Don't worry, since you are downloading the sources, these folders are already created for you):

```
> |red5_server
- -> |webapps
  - -> |red5java
    - -> |WEB-INF
      - -> |src
        - -> |your
          - -> |package
            - -> |name
          - -> |classes
        - -> |lib
```

The build.xml file that came with the sources can prepare the classes and lib directories for you, simply type:

```
#ant prepare
```

To clean out directories after a build, type:

```
#ant clean
```

You can change the target of your build by editing the *source* and *target* tags. For 1.6, change from 1.5 to 1.6.

Now, for the concept:

If you created a new application (as we will later) eg MYAPPLICATION, your directory structure would look more or less look like this:

```
> |red5_server
- -> |webapps
  - -> |MYAPPLICATION
    - -> |WEB-INF
      - -> |src
        - -> |your
          - -> |package
            - -> |name
          - -> |classes
        - -> |lib
```

3. Let us now study the following files which are contained within the WEB-INF folder: web.xml, red5-web.xml, red5-web.properties, log4j.properties, build.xml. These files are ordinarily dropped in the WEB-INF folder:
red5_server > webapps > red5java > WEB-INF > drop them here!:

a) **web.xml**: This is the first file that your Jetty/Tomcat Servlet engine will read. The parameter `webAppRootKey` defines the application context of your application. This file required by the servlet specification and is the standard deployment descriptor used by your Servlet engine. It will be used to load the other 3 files below. Again consult Joachim's migration guide.

b) **red5-web.xml**: This config file is used to configure and load your scope/context handler, your virtual-host configuration and your very own application. This is file is used by the Spring Framework IoC beans when Dependency Injecting your Servlet so it can be correctly instantiated within the red5 server.

It is also within this file that you configure your Handlers. The Handlers are the methods that are called when a flash client connects to your application ie `rtmp://your-ip/your-application`. "your-application" represents an application scope and so these Handler/Methods are valid within your application scope. Thus, your Handlers will be able to react when a flash client connects or leaves your application scope, you need to implement the interface `org.red5.server.api.IScopeHandler`.

Joachim Bauch's tutorial explains this in good detail:

[HOWTO-NewApplications.txt](#)

The Dependency Injection specified by the bean id `web.handler` is very important or else your application might not start. Please ensure you specify the correct classpath of your application here.

c) **red5-web.properties**: This file is imported into the red5-web.xml file. Use this file for items that often change within your configuration eg the `contextPath` (ie name of your application scope) or `virtualHosts` directives (which might change from one hosted service to another). If you open the file red5-web.xml, you will see that the Spring `PropertyPlaceholderConfigurer` bean is used to load the red5.properties file. The `contextPath` and `virtualHosts` properties are loaded within the red5-web.xml file by using the variables `${webapp.contextPath}` and `${webapp.virtualHosts}` respectively.

d) **log4j.properties**: This is used to configure the logging properties of your application by using the log4j libraries. This file will be used by the [org.apache.commons.logging LogFactory](#) class.

For the settings in this file to take effect, you have to register the logger for your application in this file. (Alternatively though, you could add your log4j definition in the conf/log4j.properties file, together with the logging definitions for the other sample Red5 applications. However, you may just use the log4j.properties file that is within your own application). Add the following entry (modify as appropriate for your own application):

```
In red5_server/webapps/red5java/WEB-INF/log4j.properties or
red5_server/conf/log4j.properties:
log4j.logger.org.red5.webapps.red5java=DEBUG
```

In your class you will retrieve the log by using:

```
protected static Log log = \
LogFactory.getLog(Application.class.getName());
```

You can obtain a template of these files from within the red5 server folder by going to *doc/templates/myapp/WEB-INF*

The combination of the above 4 files is used to instantiate your red5 application. This is the application we will be using to communicate with flash clients for this tutorial.

e) **build.xml**: This is the file that we will use to compile our application before restarting red5. You may use this example build file to compile/build for your own custom applications. Modify the following parameters as appropriate:

In our case it was red5java.jar and so we have:

4. Now going back to Eclipse, click within the Navigator Panel and then click F5 to refresh your Navigator panel. You will now see the folder red5java under webapps. Under red5java is WEB-INF and under WEB-INF is src. The folder src contains the folders that represent your package above ie org.red5.webapps.red5java

[Please note that all Shared Objects below are "Remote Shared Objects" ie they are stored on the Server, not the flash client]

CREATE A PLAYLIST AND STORE IT WITHIN "TRANSIENT" RED5 SERVER SHARED OBJECT USING JAVA AND OTHER SHARED OBJECTS FUNCTIONALITY

In this section we will create a Shared Red5 object. This object represents a Playlist of all available content. Once this playlist has been loaded into the Shared Object, all Flash clients that connect to our red5java application scope will have access to the same playlist. This is because in an ideal environment, we would only need one playlist for all our Flash clients. This playlist could typically be loaded from an XML file (see loadPlaylist method of Application.java). But rather than always re-reading the XML file everytime a client connects, this XML could be loaded only once when the first client connects and then made immediately available to all other clients when they connect. This is why we put the method appStart(IScope app) hook.

[****This is a very important NOTE****: When debugging your application, do not put any code inside the appStart method unless you are absolutely sure it works. This is because an error in the application scope of your application will result in your application never ever even seeing the light of day. Debug in say appConnect method (so you are able to view related debug logs) and only transfer to appStart once you are sure it works.]

1. From your red5java WEB-INF folder, right click and create a new folder called playlist. Inside this folder we drop the following file inside this playlist folder: playlist.xml

Now while still within your red5java WEB-INF folder, navigate to

```
org>red5>server>webapps>red5java>WEB-INF>src>org>red5>webapps>red5java.
```

The following source java files are dropped in: Application.java, DemoService.java, SOEventListener.java, SOHandler.java, ScheduledJob.java. These are the files that we will use to demonstrate creation of transient/non-persisted shared objects, handlers, listeners and a scheduled job. Our package name is org.red5.webapps.red5java

Below I explain some aspects of this example red5 application.

2. The Application.java...

> Application.java is the scope handler of you application. By extending the ApplicationAdapter class, we can determine or code what happens whenever the application is started, whenever the a new client connects or disconnects and whenever an application is stopped.

> loadPlaylist loads playlist items from an xml file. The main scope of your red5 application will be the folder of your red5 application. In our case, the main scope was therefore red5java. Thus on doing the following:

```
Resource playlistXML =  
\ this.appScope.getResource(fileName);
```

we actually get the resource from red5java/playlist/playlist.xml

We retrieve the InputStream to the xml file by using the usual JAVA BufferedReader.

3. The DemoService.java...

There is no reason we could not have put the methods in this file in the Application.java file. However, we all know that it is good programming practise to separate application logic into different classes depending on the pattern that emerges in your application. Anyway, that won't be the case for now, I just want to show you something very interesting...

Methods in DemoService.java are called from the flex2 client using the following call for example:

```
nc.call("demoService.getPlayList", nc_responder);
```

Now for the interesting question, how the hell does by flash client get to know how and where to call the demoservice.getPlayList method from. How does it do this? This is accomplished by wiring your DemoService class in the red5-web.xml file, through defining its bean in this file. If you have more classes, this is exactly the same place you would place them in. Thus, whenever your flash client calls a remote red5 method, the rtmp handler will be able in turn find and call your class due to this bean definition(**confirm/test this statement). You will also notice that the red5-web.xml file contains a bean definition for the Application.java file above; it is defined as the main web handler of your application as described above.

Here are the bean definitions we are talking about in the red5-web.xml file:

```
bean id="web.handler" class="org.red5.webapps.red5java.Application"
```

and

```
bean id="demoService.service"  
class="org.red5.webapps.red5java.DemoService"
```

The new service you define in red5-web.xml should end with service, so that red5 will be able to invoke it.

4. You may call `getPlaylist` method that makes use of a "Transient Shared Object" from the flex2 application which you can download from here:

[red5_flexclient.tar.gz](#)

Open the Flash client from the following URL. Put the following URL: `rtmp://ip-of-your-red5-server/red5java`.

Click the "Connect" button. On connecting to the red5 application from the flash client, the following happens:

>the xml playlist is loaded into a transient shared object, once only during the start of the application. To view the contents of this shared object, select `getPlaylist` method and click on the Execute button. You will see the contents of the `playlistSO SharedObject` we loaded in the red5 server printed on the Text Area. However, this method will only work with the latest trunk of red5.

>a counter of all the persons who have connected to your application scope is printed by clicking on the Execute button after selecting the `getCounter` method (This method should work with the latest release of red5). Follow this sequence several times to see the counter changing:

Connect>select getCounter >click Execute>click Disconnect>click Connect>select getCounter>click Execute ... etc...

>a transient shared object with a listener is created in the red5 `initTSOwithListener()` method. The listener should be triggered whenever a change occurs to the shared object. To demonstrate this, select the `sendMessage(trigger Red5 TSO update)` method and click execute. If you go to the red5_server console, you will see the default message "This is to modify TSO" printed, and `onSharedObjectUpdate` for the `SOEventListener` triggered. You may change the message that is sent to red5 by entering your own text, inside the textbox labelled `Message(message to send to red5)`. This textbox can also be used to change the message used to update the persistent shared object with listener below. At this point it is very important that I mention that make sure that you put your `onSharedObjectUpdate` functionality in the

```
onSharedObjectUpdate \  
(ISharedObjectBase so, String key, Object value)
```

and not

`onSharedObjectUpdate` functionality in the

```
onSharedObjectUpdate \  
(ISharedObject so, String key, Object value)
```

or else the `onSharedObjectUpdate` Event will never be triggered. Please remember this as it is very easy to confuse these 2 interface methods.

> a persistent shared object with a listener is created with the `initPSOwithListener` method, and just like the transient shared object above, the listener should be triggered everytime there is a change with the persistent shared object. Again, if you experience problems with this method, delete the `red5java/persistence/SharedObjects` folder.

> Calling a remote shared object method using `remote_sharedobject.send("handler", "args")` should cause your red5 shared object handler to be called. Shared object handlers can either be defined in red5-web.xml or by defining a Handler class and registering it using the `sharedobject registerServiceHandler` method. Please consult [Joachim's guide](#) for a good explanation of this.

>If on the flex client you listen for change events on shared objects, then red5 will enable this synchronisation for you, so that whenever a sharedobject value changes, the flex client will be informed. Please refer the red5_flexclient.tar.gz sources for examples of how to do this.

SHARED OBJECT LISTENERS NOTES

These implement the *ISharedListener* interface.

1. You can register several listener classes (which in our case was SOEventListener), for the same shared object.
2. Several SharedObjects can use the same listener. Simply identify the shared object attributes whose values are changing by ensuring names (keys to the shared object attributes/values) are unique. React differently if the name returned is different.

SHARED OBJECT HANDLERS NOTES

1. You may call a remote shared object from the red5 server by using SharedObject.getRemote syntax (please see the red5_flexclient sources attached).
2. You may call a red5 method (without ever having to register the class in red5-web.xml), by using so.registerServiceHandler syntax, where you can register a method handler for a shared object (You may still do the same using the red5-web.xml file though - see [Joachim's migration guide](#)). You then call this method using the remote_so.send syntax (refer to the red5_flexclient.tar.gz for how to do so)

SCHEDULE FOR THE PLAYLIST TO BE RELOADED EVERY 1 HOUR USING JAVA

You might want the playlist Transient shared object above to be reloaded every 10 minutes with updated content from a changing playlist.xml file. The Application.java and ScheduledJob.java files shows you how to create a regularly scheduled job.

BUILD YOUR RED5 APPLICATION

Whenever you make a change to your application, you need to rebuild it and restart red5. Using the red5java build file provided, here's an example that allows you to automate and speed up this process using ant:

1. First create red5.jar by running from within the root/base of the red5 source folder. You only need to do this once (ie just after downloading red5), and not all the time.

```
#ant jar
```

2. Now go to the red5java directory within your webapps directory and do:

```
#ant build
```

To adapt the build.xml file for yourself, simply change the following build.xml items depending on your system:

- >target.jar (put the name of your own jar file)
- >source= (put 1.5 or 1.6 depending on the java jvm version you are running on your machine)
- >target= (put 1.5 or 1.6 depending on the java jvm version you are running on your machine)

3. If your application builds successfully, restart red5 using ant server, ./red5.sh or red5.bat and then try to connect to your red5 application using the sample flex2 application you have modified.

CONCLUSION

The code written here is by no means clean and is not designed using the best principles. Most of it was written while half-asleep ;). It is only meant as a guide so

you are able to see Red5 in action by example based on Joachim's Migration Guide. In addition, this code has been little tested and therefore comes with no warranty whatsoever.

SPRING REFERENCES

1. <http://www.onjava.com/pub/a/onjava/2005/05/11/spring.html>
2. <http://www.springframework.org/docs/reference/beans.html>

FLEX2 REFERENCES

1. <http://flash-communications.net/technotes/fms2/flex2FMS/index.html>
2. http://livedocs.macromedia.com/flex/15/asdocs_en/
3. <http://www.amfphp.org/docs/datatypes.html>
4. <http://coenraets.org/testdrive/flex4j/index.htm>
5. http://www.adobe.com/devnet/flashmediaserver/articles/rmi_fms2_02.html
6. <http://livedocs.macromedia.com/flex/201/langref/flash/net/SharedObject.html#send>

LOG4J REFERENCE

1. <http://jakarta.apache.org/commons/logging/commons-logging-1.0.4/docs/apidocs/org/apache/commons/logging/package-summary.html>
2. <http://gef.tigris.org/LoggerConfiguration.html>
3. http://www.spikesource.com/docs/cs_1.4-linux/doc/commons-logging/commons-logging_quickstartguide.html

RED5 FUTURE DIRECTORY STRUCTURE

<http://jira.red5.org/confluence/display/appserver/3rd+proposal+or+Searching+the+>

 [Permalink](#)

55 Comments »



Bobby Berberyan said,

February 7, 2007 @ 11:25 pm

Hi there, this is very impressive and I can't wait to integrate Red 5 into my site. I've looked at many sites and tutorials, but can't figure out one thing. I want to be able to add character validation function when users log into the chatroom. Can you prescribe methods of achieving this? Thanks

-Bobby



jwamicha said,

February 8, 2007 @ 11:30 pm

Hi Bobby, Thanks.

I would probably add the character validation methods on the flash client side rather than server side. The server side just acts as a passage way for the chat messages.



piha said,

February 11, 2007 @ 1:07 am

all links to 196.202.192.126 are broken.



jwamicha said,

February 11, 2007 @ 9:23 am

We had a temporary outage yesterday. It should be back on now though.



Thijs said,

February 11, 2007 @ 9:21 pm

Great tutorial Joseph!! I added it on the Red5 wiki page:

http://osflash.org/red5#red5_help_and_information



jwamicha said,

February 11, 2007 @ 11:51 pm

Wow! Thanks alot Thijs!



joshua noble said,

February 12, 2007 @ 8:01 am

This is a really exciting, thorough, and really informative article. Thanks so much for taking the time to write it.



Sunil Gupta said,

February 27, 2007 @ 11:51 am

This is really very impressive and complete tutorial. I have developed a very large video conference application with features like chat, user list, poll, whiteboard using Red5 and it's working great.



goood.guy said,

March 6, 2007 @ 12:35 pm

Is there a way to use the FLVPlayback component with red5 to stream flv-videos?



Alan D said,

March 7, 2007 @ 4:53 pm

I followed the instructions but still no luck, can anyone help?

It compiles ok in Eclipse but it doesn't create a ".jar" file. Also, when I restart Red5 and try to test using "red5_tester.swf" I get the following message:

description = No scope "red5java" on this server.



jwamicha said,

March 7, 2007 @ 6:02 pm

Hi good.guy,

Yes it is possible to do so. Please check out the following post:
<http://www.mail-archive.com/red5@osflash.org/msg04567.html>



jwamicha said,

March 7, 2007 @ 6:05 pm

Hi Alan,

The reason the jar file isn't created is because it is not the default target of the ant build file. You need to run off the shell/command prompt:

```
>ant jar
```

For the jar file to be created.

Have you tried downloading the flexclient code that comes with this tutorial?

It will connect to the red5java application scope ok.



Dragos said,

March 19, 2007 @ 2:53 pm

Hi jwamicha, I can't download the files, the server doesn't respond to ping, please could you be so kind and post them somewhere red5java.tar.gz and red5_flexclient.tar.gz

Thanks, Dragos



jwamicha said,

March 19, 2007 @ 6:23 pm

Hi Dragos,

We have been having problems with our ISP since Friday. I have put in some new links. Please let me know if you are still unable to reach the server.



Omar said,

March 27, 2007 @ 8:53 pm

Hi everyone, I've developed some apps but I still with the same problem.

How can I call a method located at my client-side application (at the .fla or .swf file) from my java application?

thanks



jwamicha said,

March 27, 2007 @ 11:20 pm

Hi Omar,

You need to use something like:

```
import org.red5.server.api.service.IServiceCapableConnection;
```

```
IServiceCapableConnection service = (IServiceCapableConnection) conn;
```

```
service.invoke("setId", new Object[] { conn.getClient().getId() }, this);
```

Where setId would be a public function on your flash client. Object[] array will contain all the parameters of your flash function. If I remember well, a good example is in fitcDemo's Application.java



Omar said,

March 27, 2007 @ 11:59 pm

Ok, I'll try, thanks



Omar said,

March 29, 2007 @ 12:51 am

Hi, It's me again... I have done this:

-----[Server side]-----

```
public void ChangeThumbnail(int direction)
{
Object[] parameters = {1};

IServiceCapableConnection service = (IServiceCapableConnection)
Red5.getConnectionLocal();

service.invoke("change", parameters,this);
}
```

```
public void resultReceived(IPendingServiceCall ipsc)
{
System.out.println("Resultado:" + ipsc.toString());
}
```

-----[Client side (.fla file)]-----

```
conexion = new NetConnection();
conexion.connect("rtmp://localhost/app");
conexion.onStatus=ConnStatus;
```

...

```
function ConnStatus(obj)
```

```
{
for (e in obj)
{
trace("Connection - ["+e+" : "+obj[e]+"");
}
}
```

-----[Client-side output]-----

```
Connection - [code : NetConnection.Connect.Success]
Connection - [description : Connection succeeded.]
Connection - [level : status]
Conexion - [application : null]
```



Omar said,

March 29, 2007 @ 12:52 am

Conexion - [application : null]



Omar said,

March 29, 2007 @ 12:59 am

Connection - [application : null] — I think that this is the problem, isn't it?

——[Server-side output]——

Service: null Method: change Num Params: 10: 1

And there's no change, I have to change a thumbnail on a swf app but I can't do that, excuse me for my ignorance.



chr said,

March 30, 2007 @ 11:20 am

How I can to obtain a files of the Tutorial. Server <http://www.korandu.com/> is don't response.



chr said,

March 30, 2007 @ 11:28 am

Tanx, it works!



David said,

April 21, 2007 @ 2:38 am

I have no familiarity with Flex. Have done the Red5 setup and then ran swf in flexclient but nothing happened. I assume one needs a copy of Flex to get this to happen. And if so what needs to be changed on the client side to make this work?



David said,

April 21, 2007 @ 11:42 am

So I decided to take matters into my own hands. I downloaded the Flex2 SDK, lobbed the Flex client into the Flex2 samples directory and after checking the names of all directories were correct I ran 'ant' to build the Flex client.

I got the build error:

BUILD FAILED

E:\Flex2\samples\red5_flexclient\build.xml:29: Execute failed:

java.io.IOException: CreateProcess: E:\Flex2\bin\mxmxc -default-frame-rate=30 -incremental=true -default-background-color=0x869CA7 -default-size 900 650 E:\Flex2\samples\red5_flexclient/src/main.mxml -o=E:\Flex2\samples\red5_flexclient/bin/main.swf error=193

So undaunted I ran the mxmxc compiler command directly from the command line and it worked - the client was successfully rebuilt! Would just like to know why that ant build is failing.



sven schaeztl said,

April 28, 2007 @ 1:29 pm

Sorry, but the Server is offline again?

Could you put these two sample files elsewhere for download?

Thanks a lot!
Sven



jwamicha said,

April 30, 2007 @ 4:50 pm

Hey guys I'm really sorry! I'm looking for a permanent hosting solution for this as the server in our office just will not do! Should be solved by Wednesday!



jwamicha said,

May 9, 2007 @ 3:05 pm

Finally a mirror for guys who have been having problems downloading!



omar said,

May 15, 2007 @ 2:04 am

hi

I'm finishing my red5 app, my app handles video and I've mounted it on a server in my company but when I play video it plays for a while (about 5 seconds), then it stops for a long time, then plays a while and stops, does everyone know why?



David Engelmaier said,

June 14, 2007 @ 5:57 pm

Hello Jwamicha, thanks a lot for this tutorial as well as the provided codes, together with Joachim's migration and first application tutorials it gave me an idea of how the Red5 works together with AS 3, it's a fun to read and a clear way of getting your fingers wet. Thanks again!



Ilias said,

July 8, 2007 @ 5:58 pm

interesting



Kris said,

July 9, 2007 @ 7:02 pm

Cool.



Constantinos said,

July 10, 2007 @ 12:01 am

interesting



omar said,

July 10, 2007 @ 12:07 am

hi, i've finally made my full app with red5 and flash even when I don't know much about flash...

well, before, I have a question, someone knows how where developed the .fla apps that come with red5???? I've tried it but I haven't found code on the apps, just images...



Vaggelis said,

July 10, 2007 @ 1:15 pm

Nice!



Chrysostomos said,

July 10, 2007 @ 2:51 pm

Cool...



Angelo said,

July 10, 2007 @ 8:19 pm

Nice...



Stefanos said,

July 11, 2007 @ 8:14 am

Nice!



Sreenivas said,

July 12, 2007 @ 3:04 pm

Hi jwamicha,

just two days before i came to about Red5 flash server.

Will Red5 work with java/jsp files/tomcat sever or will it only work for Spring framework/jetty server/tomcat server.

I have a requirement where i have use video conferencing in jsp pages of my web application which is running on tomcat server.

please tell how to handle the above problem.give an example

Thanks



jwamicha said,

July 20, 2007 @ 11:08 am

Hello Sreenivas,

Sorry for taking so long to get back to you. I haven't tried this yet but probably go to your red5 server directory and do an #ant war

Grub the created war and drop it inside the tomcat webapps directory. Will try this out this weekend and report on the exact steps.



jwamicha said,

July 29, 2007 @ 8:29 am

For tomcat everything is exactly as above but the command is:

#ant webwar



jwamicha said,

July 29, 2007 @ 8:38 am

Grab the red5.war from the following path:

red5_server/dist/red5.war



snowman said,

August 23, 2007 @ 2:48 pm

Hey,

thank you for this cool tutorial, I like it, but, the links don't work 😞 I can't download red5java.tar.gz. The first link never even loads and the second one "has a small hiccup" ... :S



Ambrosios said,

September 2, 2007 @ 10:47 am

Interesting...



Kris said,

September 12, 2007 @ 11:00 am

Cool...



Titos said,

September 14, 2007 @ 10:10 pm

Cool...



Rafa said,

September 15, 2007 @ 8:22 pm

Hi,

Thank you for this great tutorial, pretty valuable for a flex / red5 beginner. Everything works as expected, but i'm having troubles calling the remote shared object handler method from Flex,

- When first connected, calling TSOHandler.sendMessage(..) from Flex gets registered in red5 log, but tsoOnSync event handler never gets called in Flex.

- If I disconnect from the server and reconnect, subsequent calls don't even get logged in Red5.

Everything goes fine by using NetConnection.call instead. Any clue about this?

Is there any performance penalty for using the NetConnection call method instead of the SharedObject method?

Thanks in advance



Rafa said,

September 16, 2007 @ 8:07 pm

Hi,

Seems the second issue has been solved in Red5 trunk. The room scope was not being stopped when the last client exited the room although my shared object was being properly freed. Since i was creating this shared object at roomStart handler which was never fired again -since from server's point of view the room actually was running-, i was getting a null reference to the SO. Now it's (apparently) fixed.

First issue's still driving me mad.

Thank you.



Yawei said,

September 20, 2007 @ 4:56 pm

This is a great article.

Can anybody email me the sample codes? I can't download them.

Thanks a lot.



Yawei said,

September 20, 2007 @ 4:56 pm

my email zhiyawei@hotmail.com





Manoj sharma said,

October 3, 2007 @ 12:41 pm

this is the very big problem



Mandar Khankhoje said,

November 20, 2007 @ 12:19 pm

Thanks. for a very good application, and a great article.



Mandar Khankhoje said,

November 20, 2007 @ 12:36 pm

Hey Jwamicha,

Thanks. for a very good application, and a great article.

Can u tell me will jsp, php, js files/servlet will work with red5 server.



Raihan Kibria said,

November 21, 2007 @ 7:44 pm

Excellent article. Thanks a lot.



Ed said,

December 5, 2007 @ 8:03 pm

Hey jwamicha, I think you accidentally left something out that you meant to include (and it's the very thing that I am looking for).

In this part:

—

e) build.xml: This is the file that we will use to compile our application before restarting red5. You may use this example build file to compile/build for your own custom applications. Modify the following parameters as appropriate:

In our case it was red5java.jar and so we have:

—

There is nothing after the two colons. (And then it just goes on to the next topic). I even checked the underlying html and there are just empty code tags (as if you were going to paste it in there). That code is what I need. 😊
Can you please fill in these two empty spaces? Thanks so much.



Cheap Web Hosting and Domains said,

February 6, 2008 @ 7:43 pm

For diifernt hosting When I made an application it shares it for all the sites. I want it to process the requests as different for the connections from

different sites...

[RSS feed for comments on this post](#) · [TrackBack URI](#)

Leave a Comment

Name (required)

E-mail (required)

URI

Submit Comment

[Blog at WordPress.com.](#) · [Design by Beccary](#) · [XHTML](#) · [CSS](#)