# **Webscript Training Assignments**

# **Free Marker Demo**

Go to online freemarker tester site : <http://try.freemarker.org/>

**Template:**

hello, ${message}! --- To print a data model variable

-------------------------------

<#assign users = ['Ram', 'Sita']> ------ Variable Assignment

<#list users as user> ----------- Looping an array

${user}

</#list>

-------------------------------------------

<#if x== 10>

You are correct !

<#else>

I don't know what is x ---------------- if – else condition

</#if>

-------------------------------------------

Following are the list of Documents

<#list documents as document> -------- Looping a JSON array

${document.name}

</#list>

**Data Model**

message = Happy coding :-)

x= 11

documents = [{"name": "Sample Doc" , "modifier": "Ram"}, {"name" : " Test Doc" , "modifier" : " Sita"}]

# **Presentation Webscript**

This a sample webscript present while generating alfresco SDK for creating alfresco and share repository extensions

## **Exercise 1**

1. Go to

<**repo project location**>src/main/resources/alfresco/extension/templates/webscripts/alfresco/tutorials for alfresco.war

1. Create / modify following files

helloworld.get.desc.xml

<webscript>

<shortname>Hello World Sample Webscript</shortname>

<description>Hands back a greeting</description>

<url>/sample/helloworld</url>

<authentication>user</authentication>

<format default=*"html"*></format>

<lifecycle>sample</lifecycle>

</webscript>

helloworld.get.js

model["fromJS"] = "Hello from JS!";

model["documents"] = [{"name": "Sample Doc" , "modifier": "Ram"}, {"name" : " Test Doc" , "modifier" : " Sita"}];

helloworld.get.html.ftl

Message: '${fromJS}'

</br></br>

<table border=3 bordercolor=GREEN>

<th>Name</th>

<th>Modifier</th>

<#list documents as document>

<tr>

<td>${document.name}</td>

<td>${document.modifier}</td>

</tr>

</#list>

</table>

1. Deploy webscript by running SDK
2. Register wescript
   1. <http://localhost:8080/alfresco/s/index> and click on Refresh Webscripts
   2. Hit <http://localhost:8080/alfresco/s/sample/helloworld> to see the output

## **Exercise 2**

1. Create the same as above webscript under

src/main/resources/alfresco/web-extension/site-webscripts/tutorials for share.war

1. Deploy webscript by running SDK
2. Register webscript
   1. <http://localhost:8080/share/page/index> and click on Refresh Webscripts
   2. Hit <http://localhost:8080/alfresco/s/sample/helloworld> to see the output

# **Data Webscripts (Repository Wescript) - JS backed**

## **Exercise 1**

Create folder using JavaScript controller

1. Create each file under
2. **<repo projectlocation>\**src\main\resources\alfresco\extension\templates\webscripts\alfresco\com\cts\demo

**create-folder.get.desc.xml**

<webscript>

<shortname>Demo Create Folder Webscript</shortname>

<description>Demo Create folder webscript using javascript controller</description>

<url>/cts/demo/createfolder?folder</url>

<authentication>user</authentication>

<format default=*"html"*></format>

<family>CTS DEMO</family>

</webscript>

**Create-folder.get.js**

**function** **main**() {

**var** folderName = args["folder"];

**if**( folderName == undefined || folderName.length == 0){

status.code = 404;

status.message = "No Folder name found";

status.redirect = **true**;

} **else**{

**var** myfolder = userhome.createFolder("folderName", "cm:folder");

myfolder.save();

model.folderName = folderName;

**return** model;

}

}

main();

**create-folder.get.html.ftl**

<h2> Folder with name ${folderName} created successfully </h2>

**create-folder.get.html.404.ftl ( Response code template)**

<h3> Folder name is not provided </h3>

1. Deploy webscript by running SDK
2. Register wescript
   1. <http://localhost:8080/alfresco/s/index> and click on Refresh Webscripts
   2. Hit [http://localhost:8080/alfresco/s/cts/demo/createfolder?folder=TEST](http://localhost:8080/alfresco/s/cts/demo/createfolder?folder=TEST%20)  to see the output
   3. Hit <http://localhost:8080/alfresco/s/cts/demo/createfolder?folder> to see 404 error message

# Java Backed Webscript – Declarative Webscript

### Use Case : Add user to DemoUsers group

It is assumed that there are already a few users present in alfresco repository and there is a group created with name “**DemoUsers”**

1. Create Java Controller :
2. Java Class : AddUsersToGroupWebscript.java
3. Package : [com](eclipse-javadoc:%E2%98%82=demo-platform-jar/src%5C/main%5C/java%3Ccom).[cts](eclipse-javadoc:%E2%98%82=demo-platform-jar/src%5C/main%5C/java%3Ccom.cts).[demo](eclipse-javadoc:%E2%98%82=demo-platform-jar/src%5C/main%5C/java%3Ccom.cts.demo)

**package** com.cts.demo;

**import** java.util.ArrayList;

**import** java.util.HashMap;

**import** java.util.List;

**import** java.util.Map;

**import** java.util.concurrent.ConcurrentHashMap;

**import** org.alfresco.query.PagingRequest;

**import** org.alfresco.query.PagingResults;

**import** org.alfresco.repo.security.authentication.AuthenticationUtil;

**import** org.alfresco.service.ServiceRegistry;

**import** org.alfresco.service.cmr.repository.DuplicateChildNodeNameException;

**import** org.alfresco.service.cmr.security.PersonService.PersonInfo;

**import** org.alfresco.service.namespace.QName;

**import** org.alfresco.util.Pair;

**import** org.apache.commons.logging.Log;

**import** org.apache.commons.logging.LogFactory;

**import** org.springframework.extensions.webscripts.Cache;

**import** org.springframework.extensions.webscripts.DeclarativeWebScript;

**import** org.springframework.extensions.webscripts.Status;

**import** org.springframework.extensions.webscripts.WebScriptRequest;

/\*\*

\* The Class AddUsersToGroupWebscript.

\*/

**public** **class** AddUsersToGroupWebscript **extends** DeclarativeWebScript {

/\*\* The Constant LOGGER. \*/

**private** **static** **final** Log ***LOGGER*** = LogFactory.*getLog*(AddUsersToGroupWebscript.**class**);

/\*\* The service registry. \*/

**private** ServiceRegistry serviceRegistry;

/\*\*

\* Gets the service registry.

\*

\* **@return** the service registry

\*/

**public** ServiceRegistry getServiceRegistry() {

**return** serviceRegistry;

}

/\*\*

\* Sets the service registry.

\*

\* **@param** serviceRegistry

\* the new service registry

\*/

**public** **void** setServiceRegistry(**final** ServiceRegistry serviceRegistry) {

**this**.serviceRegistry = serviceRegistry;

}

/\*

\* (non-Javadoc)

\*

\* @see org.springframework.extensions.webscripts.DeclarativeWebScript#

\* executeImpl(org.springframework.extensions.webscripts.WebScriptRequest,

\* org.springframework.extensions.webscripts.Status,

\* org.springframework.extensions.webscripts.Cache)

\*/

**public** Map<String, Object> executeImpl(**final** WebScriptRequest req, **final** Status status, **final** Cache cache) {

***LOGGER***.debug("Inside AddUsersToGroupWebscript ....");

**final** Map<String, Object> model = **new** ConcurrentHashMap<String, Object>();

String group = req.getParameter("group");

String[] groupList = group.split(",");

Map<String, List<String>> updatedUserList = **new** HashMap<String, List<String>>();

**final** List<String> allUsers = **new** ArrayList<String>();

AuthenticationUtil.*setRunAsUserSystem*();

PagingResults<PersonInfo> users = serviceRegistry.getPersonService().getPeople("\*", **new** ArrayList<QName>(),

**new** ArrayList<Pair<QName, Boolean>>(),

**new** PagingRequest(serviceRegistry.getPersonService().countPeople()));

***LOGGER***.info("The number of users in the system : " + serviceRegistry.getPersonService().countPeople());

**do** {

List<PersonInfo> personInfos = users.getPage();

**for** (PersonInfo personInfo : personInfos) {

String userName = personInfo.getUserName();

allUsers.add(userName);

}

} **while** (users.hasMoreItems());

**for** (String user : allUsers) {

List<String> addedGroups = **new** ArrayList<String>();

**for** (String groupName : groupList) {

**try** {

serviceRegistry.getAuthorityService().addAuthority("GROUP\_" + groupName, user);

addedGroups.add(groupName);

} **catch** (DuplicateChildNodeNameException dcExp) {

***LOGGER***.error("Authority already exists for : " +user + " with Group Name : "+ groupName);

}

updatedUserList.put(user, addedGroups);

}

}

model.put("Result", updatedUserList);

**return** model;

}

}

1. Descriptor file :

/src/main/resources/alfresco/extension/templates/webscripts/alfresco/com/cts/demo/adduserstogroup/adduserstogroup.get.desc.xml

<webscript>

<shortname>Adding Users to Groups</shortname>

<description>Adding Users to Groups</description>

<family>CTS DEMO</family>

<url>/cts/adduserstogroup?group</url> <!-- Comma separated groups can be given -->

<authentication>admin</authentication>

</webscript>

1. Freemarker : /src/main/resources/alfresco/extension/templates/webscripts/alfresco/com/cts/demo/adduserstogroup/adduserstogroup.get.html.ftl

<head>

<style>

table, th, td {

border: 1px solid black;

}

</style>

</head>

<body>

<#assign userNames = Result?keys>

<table style="width:100%">

<tr>

<th>User Name</th>

<th>Group Name</th>

</tr>

<#list userNames as userName>

<tr>

<td>${userName}</td>

<td>${Result[userName]?join(", ")}</td>

</tr>

</#list>

</table>

</body>

1. Bean configuration for DI

/src/main/resources/alfresco/module/demo-platform-jar/context/webscript-context.xml

<!--Webscript to add all users to groups provided -->

<**bean** **id**="webscript.alfresco.com.cts.demo.adduserstogroup.adduserstogroup.get" **class**="com.cts.demo.AddUsersToGroupWebscript" **parent**="webscript">

<**property** **name**="serviceRegistry">

<**ref** **bean**="ServiceRegistry" />

</**property**>

</**bean**>

1. Run SDK for deploying the webscript
2. Register webscript and invoke the url

[http://localhost:8080/alfresco/service/cts/adduserstogroup?group= DemoUsers](http://localhost:8080/alfresco/service/cts/adduserstogroup?group=%20DemoUsers)

# Java Backed Webscript – Abstract Webscript

### Print JSON response in browser webpage

1. Java Controller : \src\main\java\com\cts\demo\JSONResponseWebScript.java

**package com.cts.demo;**

**import org.json.JSONException;**

**import org.json.JSONObject;**

**import org.springframework.extensions.webscripts.AbstractWebScript;**

**import org.springframework.extensions.webscripts.WebScriptException;**

**import org.springframework.extensions.webscripts.WebScriptRequest;**

**import org.springframework.extensions.webscripts.WebScriptResponse;**

**import java.io.IOException;**

**public class JSONResponseWebScript extends AbstractWebScript {**

**@Override**

**public void execute(WebScriptRequest req, WebScriptResponse res)**

**throws IOException {**

**try {**

**JSONObject obj = new JSONObject();**

**obj.put("name", "Alfresco");**

**String jsonString = obj.toString();**

**res.getWriter().write(jsonString);**

**} catch (JSONException e) {**

**throw new WebScriptException("Unable to serialize JSON");**

**}**

**}**

**}**

1. Descriptor : /src/main/resources/alfresco/extension/templates/webscripts/alfresco/com/cts/demo/showjson/show-json-output.get.desc.xml

<webscript>

<shortname>Show Json output</shortname>

<description>Show Json output Using AbstractWebscript</description>

<family>CTS DEMO</family>

<url>/cts/showJson</url>

<authentication>admin</authentication>

</webscript>

1. Bean configuration : \src\main\resources\alfresco\module\demo-platform-jar\context\webscript-context.xml

<!-- Show Json response using ABstractWebscript-->

<**bean** **id**="webscript.alfresco.com.cts.demo.showjson.show-json-output.get"

**class**="com.cts.demo.JSONResponseWebScript"

**parent**="webscript">

</**bean**>

1. Restart server and register webscript and invoke url