

Troubleshooting WebSphere Commerce Struts configurations

Skill Level: Intermediate

[Mike Callaghan \(mcallagh@ca.ibm.com\)](mailto:mcallagh@ca.ibm.com)
WebSphere Commerce Support Analyst
IBM

08 Jul 2009

Learn how to troubleshoot configuration problems with the Struts framework within WebSphere® Commerce V6.0. This article explains how action-mappings and global-forwards are used during the WebSphere Commerce runtime, and how to interpret the runtime tracing to understand the behavior of Web requests.

Introduction

In WebSphere Commerce 6.0, the runtime engine uses the Struts framework to invoke forwards and redirects. Each WebSphere Commerce Web module has its own Struts configurations, defining base actions and global-forwards that can be difficult to debug runtime problems. This article will help you understand the runtime framework of WebSphere Commerce, now based upon Struts, by interpreting the trace produced when problems arise. The examples will focus on global-forward and action-mapping configurations.

Debugging Struts issues

The best way to understand the behavior when debugging a runtime Struts issue is to enable WebSphere Application Server tracing in the WebSphere Commerce application server. The trace string to enable is:

```
com.ibm.websphere.commerce.WC_SERVER=all:org.apache.struts.*=all
```

The rest of this article will review the trace produced using the above trace string and explain how to understand the configuration problems in the Struts configuration files.

Struts configuration files

Each WebSphere Commerce Web module has its own Struts configuration files defined. The Web modules's `WEB-INF/web.xml` defines the list of Struts configuration files for the module.

For example, the out-of-the-box Stores.war configuration has the following definition:

```
<init-param>
  <param-name>config</param-name>
  <param-value>/WEB-INF/struts-config.xml ,
  /WEB-INF/struts-config-GiftCenter.xml ,
  /WEB-INF/struts-config-migrate.xml ,
  /WEB-INF/struts-config-ext.xml
</param-value>
</init-param>
```

This lists the Struts configuration files that will be merged and read for Web requests coming to the Store's module. Enabling additional features, such as component services, may add additional Struts configuration files as well.

During instance creation, the `struts-config.xml` is populated with bootstrap data, all with "STORE_ID = 0". This is the default for all stores, unless another entry exists for the exact STORE_ID for which the Web request is made.

During store publish, the `struts-config-ext.xml` is populated with store specific implementations for the action-mappings and global-forwards definitions. This is either a new class implementation for a specific command interface, such as modified settings for the https or authenticate properties, or different JSP references for global-forwards.

Custom definitions for customized actions or forwards also need to be placed in the `struts-config-ext.xml`.

Note that the order in which the Struts XML files are listed in the parameter above in `web.xml` identifies their precedence. The files have precedence in reverse order of their listing in the parameter. Here, `struts-config-ext.xml` takes precedence over all other XMLs.

Understanding action-mappings

The action-mappings in the WebSphere Commerce Struts files are used to map to

either a command interface or map to a global-forward, which in turn, maps to a JSP to be displayed for a view.

In the case of mapping to a command interface, these are mappings to WebSphere Commerce controller commands, which are registered in the CMDREG table with a corresponding implementation class. Each store may have a different implementation for the same interface registered.

For example, the default entry for OrderItemAdd action mapping is:

```
<action
  parameter="com.ibm.commerce.orderitems.commands.OrderItemAddCmd"
  path="/OrderItemAdd" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="authenticate" value="0:0"/>
  <set-property property="https" value="0:1"/>
</action>
```

The path (/OrderItemAdd in this case) maps to a command interface (com.ibm.commerce.orderitems.commands.OrderItemAddCmd), which is a registered command interface in the CMDREG table.

Working case: Request for a controller command (action-mapping)

We will explain how to understand the traces in some problematic cases below, but first we will highlight what a working case looks like in the trace. Two examples show requests to the store's Web module: one where the request is for a controller command (action) and later, the other where the request is for a view (action and forward).

Take an example where the requested URL from the browser was:

```
http://localhost/webapp/wcs/stores/servlet/OrderItemAdd?storeId=10001&catalogId=10001
&orderId=.
```

The first thing to look for in the trace is the ECActionServlet recording the URLINFO of the request:

```
[4/8/09 9:27:38:853 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff5 com.ibm.commerce.
struts.ECActionServlet.processRequest URLINFO - Method = GET
ServletPath = /servlet
PathTranslated =
/opt/WebSphere/AppServer/profiles/demo/installedApps/WC_demo_cell/
WC_demo.ear/Stores.war/OrderItemAdd
RequestURI = /webapp/wcs/stores/servlet/OrderItemAdd
PathInfo = /OrderItemAdd
ContextPath=/webapp/wcs/stores
QueryString = storeId=10001&catalogId=10001&orderId=.
```

The *RequestURI* value shows the URI that was requested above. The *QueryString* value shows the parameters passed in via the URL to the request.

Shortly after, you can see the Struts RequestProcessor looking for the entry that matches the *PathInfo*, which is the action in Struts:

```
[4/8/09 9:27:38:855 EDT] 000000a2 RequestProces 1
org.apache.struts.action.RequestProcessor process Processing a 'GET'
for path '/'
OrderItemAdd'
[4/8/09 9:27:38:856 EDT] 000000a2 RequestProces 1
org.apache.struts.action.RequestProcessor processActionCreate
Looking for Action instance for class
com.ibm.commerce.struts.BaseAction
[4/8/09 9:27:38:856 EDT] 000000a2 RequestProces 3
org.apache.struts.action.RequestProcessor processActionCreate
Returning existing Action instance
[4/8/09 9:27:38:856 EDT] 000000a2 WC_SERVER >
63a3dfc9:1207bd65480:-7ff5
com.ibm.commerce.struts.BaseAction.execute Entry
[4/8/09 9:27:38:856 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff5
com.ibm.commerce.struts.BaseAction.execute
ActionConfig[path=/OrderItemAdd,
parameter=com.ibm.commerce.orderitems.commands.OrderItemAddCmd,
scope=session,type=com.ibm.commerce.struts.BaseAction https?0:1
authenticate?0:0
saveToken?null
valdiateToken?null
```

The properties retrieved from the Struts configuration are shown in the last entry above. You can also see that in this case, the path `/OrderItemAdd` maps to `com.ibm.commerce.orderitems.commands.OrderItemAddCmd`, which is a controller command registered in the CMDREG table.

You can see the lookup to CMDREG here:

```
[4/8/09 9:27:38:860 EDT] 000000a2 WC_SERVER
> 63a3dfc9:1207bd65480:-7ff5
com.ibm.commerce.webcontroller.WebControllerHelper.executeCommand
(ViewCommandContext,Map,String) Entry
[4/8/09 9:27:38:860 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff5
com.ibm.commerce.webcontroller.WebControllerHelper.executeCommand
(ViewCommandContext,Map,String) invokeCommand
com.ibm.commerce.orderitems.commands.OrderItemAddCmd
[4/8/09 9:27:38:860 EDT] 000000a2 WC_SERVER >
63a3dfc9:1207bd65480:-7ff5
com.ibm.commerce.component.WebAdapterComponentImpl.executeCommand(CommandContext,
Map,
String) Entry
[4/8/09 9:27:38:860 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff5
com.ibm.commerce.component.WebAdapterComponentImpl.executeCommand(CommandContext,
Map,
String)
interfaceName=com.ibm.commerce.orderitems.commands.OrderItemAddCmd
[4/8/09 9:27:38:860 EDT] 000000a2 WC_SERVER >
63a3dfc9:1207bd65480:-7ff5
CommandFactory.locateCommandEntry Entry
[4/8/09 9:27:38:860 EDT] 000000a2 WC_SERVER 3
```

```
63a3dfc9:1207bd65480:-7ff5
CommandRegistry.find storePath length=1
[4/8/09 9:27:38:860 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff5
CommandRegistry.find found entry with key
com.ibm.commerce.orderitems.commands.OrderItemAddCmd&0
[4/8/09 9:27:38:860 EDT] 000000a2 WC_SERVER <
63a3dfc9:1207bd65480:-7ff5
CommandFactory.locateCommandEntry Exit
```

This line shows that it found a definition in CMDREG for STORE_ID of 0:

```
CommandRegistry.find found entry with key
com.ibm.commerce.orderitems.commands.OrderItemAddCmd&0
```

The above request mapped this action-mapping in the Stores.war/WEB-INF/struts-config.xml:

```
<action
  parameter="com.ibm.commerce.orderitems.commands.OrderItemAddCmd"
  path="/OrderItemAdd" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="authenticate" value="0:0"/>
  <set-property property="https" value="0:1"/>
</action>
```

Problem case: Missing action definition

Now that you have seen a working case, you can focus on the first problematic case that may arise. This deals with a missing action definition.

The requested URL is:

```
https://localhost/webapp/wcs/stores/servlet/MyOrderItemAdd?catalogId=10001&storeId=10001
&langId=-1
```

When invoked, you get the generic error page in the browser. Notice that the "My" is prepended to the command name. This error can happen if a new controller command was created, extending the base implementation.

To debug this, look at the trace again. As before, the RequestURI appears normal, but note the key piece where `PathInfo = /MyOrderItemAdd`. `/MyOrderItemAdd` is what you will look up in Struts.

Whereas in the working case, you can see the RequestProcessor finding an action configuration that matches the PathInfo, but this time you see:

```
[4/8/09 11:00:19:079 EDT] 000000a2 RequestProces 1
org.apache.struts.action.RequestProcessor process Processing a 'GET'
```

```

for path '/'
MyOrderItemAdd'
[4/8/09 11:00:19:082 EDT] 000000a2 RequestProces 1
org.apache.struts.action.RequestProcessor processActionCreate
Looking for Action
instance
for class com.ibm.commerce.struts.BaseAction
[4/8/09 11:00:19:082 EDT] 000000a2 RequestProces 3
org.apache.struts.action.RequestProcessor processActionCreate
Returning existing
Action instance
[4/8/09 11:00:19:082 EDT] 000000a2 WC_SERVER >
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.BaseAction.execute Entry
[4/8/09 11:00:19:082 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.BaseAction.execute ActionConfig[path=/
JspView,scope=session,type=com.ibm.commerce.struts.BaseAction
https?null
authenticate?null
saveToken?null
valdiateToken?null

```

Since no action name matches `/MyOrderItemAdd` in the Struts configuration files, it defaults to `/JspView`, which has the runtime framework thinking this must be a JSP being invoked directly, rather than a view. To invoke a JSP directly, it does not need to be registered in Struts.

When the Base Action invokes the action definition, `/JspView` is found rather than `/MyOrderItemAdd`, so it fails:

```

[4/8/09 11:00:19:084 EDT] 000000a2 WC_SERVER >
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.BaseAction.executeView Entry
[4/8/09 11:00:19:084 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.BaseAction.executeView actionName=JspView
[4/8/09 11:00:19:084 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.BaseAction.executeView unknown view from
path info:
MyOrderItemAdd
[4/8/09 11:00:19:085 EDT] 000000a2 CommerceSrvr A
com.ibm.commerce.struts.BaseAction
executeView CMN0203E: Command not found: "MyOrderItemAdd ".
[4/8/09 11:00:19:087 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.BaseAction.executeView exception
com.ibm.commerce.exception.ECApplicationException: Command not
found:
"MyOrderItemAdd".
[4/8/09 11:00:19:087 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.webcontroller.WebControllerHelper.rollbackRequest
exception
com.ibm.commerce.exception.ECApplicationException: Command not
found:
"MyOrderItemAdd".
[4/8/09 11:00:19:087 EDT] 000000a2 WC_SERVER >
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.webcontroller.WebControllerHelper.rollbackService
Entry

```

```
[4/8/09 11:00:19:088 EDT] 000000a2 WC_SERVER <
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.webcontroller.WebControllerHelper.rollbackService
Exit
```

Here you get an exception and a rollback. Also note the "Command not found: MyOrderItemAdd" message from ECAApplicationException.

Viewing the source of the HTML on the generic error page, you see in the browser the same message:

```
Exception Type: 0
Message Key: _ERR_CMD_CMD_NOT_FOUND
Message: CMN3101E The system is unavailable due to "CMN0203E".
System Message: Command not found: "MyOrderItemAdd".
Originating Command:
Corrective Action:
```

"Command not found" errors will result if the requested path (MyOrderItemAdd) is not any of these:

- Registered in Struts as an action
- Registered in CMDREG table as a command
- A JSP that can be invoked directly (for example, MyOrderItemAdd.jsp)

To correct this problem, add an action definition for the command in the Stores.war/WEB-INF/struts-config-ext.xml. An example is:

```
<action
  parameter="com.mycompany.orders.MyOrderItemAddCmd"
  path="/MyOrderItemAdd" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="authenticate" value="0:0"/>
  <set-property property="https" value="0:1"/>
</action>
```

After adding this entry, and either restarting the server or refreshing the Struts registry, your custom implementation is now invoked.

This working case looks similar to the working case previously shown.

Action-mapping properties

Note the default settings for *authenticate* and *https*. The 0:0 for authenticate property means that "STORE_ID = 0" will have an https value of 0. STORE_ID 0 is the default STORE_ID that is used if no other STORE_ID entry matches for the Web

request.

Authenticate property

The authenticate value of 0:0 means that the request does not require the user to be authenticated against WebSphere Commerce. This implies that a generic, guest, or registered user can invoke the action. If the value read 0:1, then this indicates the request requires an authenticated user. If the user submitting the request is already authenticated, such as a registered user, then the request will continue. If the user is not authenticated, then it redirects the user to LogonForm, where they must authenticate before the initial request continues.

Problem case: User is being redirected to the LogonForm when requesting a view

This is a common case where a guest (or generic) user makes a request, while you expect them to see the view they specified, it is instead being redirected to LogonForm. Here it asks the user to authenticate before the previous request is fulfilled.

Take again the example of:

```
https://localhost/webapp/wcs/stores/servlet/OrderItemAdd?catalogId=10001&storeId=10001&langId=-1
```

However this time, you will use a new configuration, which requires authentication for store 10001:

```
<action
  parameter="com.ibm.commerce.orderitems.commands.OrderItemAddCmd"
  path="/OrderItemAdd" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="authenticate" value="0:0,10001:1"/>
  <set-property property="https" value="0:1,10001:1"/>
</action>
```

In this example, you have made the request as a guest user to add an item to the cart using OrderItemAdd. Yet when you submit the request, the browser is redirected to LogonForm, such as:

```
https://localhost/webapp/wcs/stores/servlet/LogonForm?storeId=10001&catalogId=10001&someencryptedparameters
```

The RequestURI appears normal again. Here, the first interesting piece is the Base Action listing the properties of the action that the RequestProcessor has identified:

```
[4/8/09 11:22:44:501 EDT] 000000a1 RequestProces 1
```

```

org.apache.struts.action.RequestProcessor process Processing a 'GET'
for path '/'
OrderItemAdd'
[4/8/09 11:22:44:504 EDT] 000000a1 RequestProces 1
org.apache.struts.action.RequestProcessor processActionCreate
Looking for Action
instance for class com.ibm.commerce.struts.BaseAction
[4/8/09 11:22:44:504 EDT] 000000a1 RequestProces 3
org.apache.struts.action.RequestProcessor processActionCreate
Returning existing
Action instance
[4/8/09 11:22:44:504 EDT] 000000a1 WC_SERVER >
63a3dfc9:1207bd65480:-7ff1
com.ibm.commerce.struts.BaseAction.execute Entry
[4/8/09 11:22:44:504 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff1
com.ibm.commerce.struts.BaseAction.execute ActionConfig[path=/
OrderItemAdd,parameter=com.ibm.commerce.orderitems.commands.OrderItemAddCmd,
scope=session, type=com.ibm.commerce.struts.BaseAction
https?0:1,10001:1
authenticate?0:0,10001:1
saveToken?null
valdiateToken?null

```

Notice `authenticate` is set to 1 (meaning, requires authentication) for store 10001.

Since the action requires authentication, you can see the check performed to see if the current user is an authenticated (logged in) user:

```

[4/8/09 11:22:44:513 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff1
com.ibm.commerce.browseradapter.HttpBrowserAdapter.preInvokeCommand(ViewCommandContext,
HttpRequestAttributes) check for authenticated caller
[4/8/09 11:22:44:519 EDT] 000000a1 WC_SERVER >
63a3dfc9:1207bd65480:-7ff1
com.ibm.commerce.browseradapter.HttpBrowserAdapter.isCallerAuthenticated
Entry
[4/8/09 11:22:44:541 EDT] 000000a1 WC_SERVER <
63a3dfc9:1207bd65480:-7ff1
com.ibm.commerce.browseradapter.HttpBrowserAdapter.isCallerAuthenticated
Exit
authenticated? false

```

Here, the user is identified as a non-authenticated user ("authenticated? false").

From this point on, it will build a redirect to the LogonForm, and then remember the original URL requested (OrderItemAdd) so it can redirect there after a successful logon.

```

[4/8/09 11:22:44:542 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff1
com.ibm.commerce.browseradapter.HttpBrowserAdapter.generateLogonRedirectResponse
Properties
redirectURL=
OrderItemAdd?storeId=10001&orderId=.&URL=SetPendingOrder%3FURL%
3DOrderCalculate%3FURL%3DOrderItemDisplay%26updatePrices%3D1%26calculationUsageId%3D-1%
26orderId%
3D.&catEntryId=10001&errorViewName=ProductDisplayErrorView&catalogId=10001
&quantity=1

```

```
[4/8/09 11:22:44:542 EDT] 000000a1 WC_SERVER <
63a3dfc9:1207bd65480:-7ff1
com.ibm.commerce.browseradapter.HttpBrowserAdapter.generateLogonRedirectResponse
Properties
Exit
respProperties=WcUseHttps=true
storeId=10001
orderId=.
URL=OrderItemAdd?storeId=10001&orderId=.
redirecturl=LogonForm
```

This however, is not the desired behavior. In this case, since you want the guest user to add to their shopping cart and change the Struts definition for the action to say that the command does not require authentication (10001:0 for the authenticate property).

```
<action
  parameter="com.ibm.commerce.orderitems.commands.OrderItemAddCmd"
  path="/OrderItemAdd" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="authenticate" value="0:0,10001:0"/>
  <set-property property="https" value="0:1,10001:1"/>
</action>
```

Credaccept property

When the authenticate property is set to a value of 1, the request requires an authenticated user, which means partial credentials. Partial credentials are provided when persistent sessions are enabled for a store and the user has selected "Remember Me". This creates a persistent cookie keeping their user ID. Upon return to the store in a new browser session, the detection of the persistent cookie allows WebSphere Commerce to partially authenticate the remembered user. If the authenticate property is set to 1, it requires authentication, but you can add an additional property called "credaccept". This indicates whether or not partial credentials are accepted. A partially authenticated user is a "Remembered" user.

Setting the value to "P" means partial credentials are accepted. Any other value (or the absence of the property means that partial credentials are not accepted. The user must be fully authenticated (or logged in within the browser session) to access a view or command.

Problem case: User is being redirected to LogonForm

This example is based on the previous issue. Again, a guest user makes the request to OrderItemAdd:

```
https://localhost/webapp/wcs/stores/servlet/OrderItemAdd?catalogId=10001&storeId=10001
&langId=-1
```

As per the original configuration, the OrderItemAdd action requires authentication:

```
<action
  parameter="com.ibm.commerce.orderitems.commands.OrderItemAddCmd"
  path="/OrderItemAdd" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="authenticate" value="0:0,10001:1"/>
  <set-property property="https" value="0:1,10001:1"/>
</action>
```

However, in this case, persistent sessions are enabled and the user has selected "Remember Me" when they previously logged into the site. When they return to the site at a later time, "Remember Me" is working fine, and they can see their information. However, when the OrderItemAdd is clicked, they are still redirected to LogonForm.

Again, this is not the desired behavior. In this case, the "remembered" user, who is partially authenticated, is expected to add an item to their cart. The only additional key line from the trace (based on the trace of the previous example) is:

```
[4/8/09 11:22:44:543 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff1
com.ibm.commerce.browseradapter.HttpBrowserAdapter.preInvokeCommand
(ViewCommandContext, HttpRequestAttributes) partial credentials are
not
used .... (this could have been done because of the migrate guest
user)
```

This indicates that partial credentials are not accepted for this action. Based on what you just learned however, to have a partially authenticated user qualify as an "authenticated" user in terms of the runtime framework, you need to allow for partial credentials. You do this by adding a new "credAccept" property into the action definition and set it to a value of "P" (meaning "partial credentials are accepted"):

```
<action
  parameter="com.ibm.commerce.orderitems.commands.OrderItemAddCmd"
  path="/OrderItemAdd" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="authenticate" value="0:0,10001:0"/>
  <set-property property="https" value="0:1,10001:1"/>
  <set-property property="credAccept" value="10001:P"/>
</action>
```

After making the change, the authenticated user appears for this action, even though it is only partially authenticated:

```
[4/8/09 11:47:06:388 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7fee
com.ibm.commerce.
browseradapter.HttpBrowserAdapter.preInvokeCommand
(ViewCommandContext, HttpRequestAttributes)
check for authenticated caller
[4/8/09 11:47:06:388 EDT] 000000a2 WC_SERVER >
63a3dfc9:1207bd65480:-7fee
```

```

com.ibm.commerce.browseradapter.HttpBrowserAdapter.isCallerAuthenticated
Entry
[4/8/09 11:47:06:388 EDT] 000000a2 WC_SERVER <
63a3dfc9:1207bd65480:-7fee
com.ibm.commerce.
browseradapter.HttpBrowserAdapter.isCallerAuthenticated Exit
authenticated? true

```

Whereas before the trace said partial credentials are not going to be allowed for this action, you now see that the trace looks like this:

```

[4/8/09 11:47:06:389 EDT] 000000a2 WC_SERVER >
63a3dfc9:1207bd65480:-7fee
com.ibm.commerce.browseradapter.HttpBrowserAdapter.isAccessAllowedWithPartial
Credentials (CommandContext aCommandContext) Entry OrderItemAdd
[4/8/09 11:47:06:389 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7fee
com.ibm.commerce.browseradapter.HttpBrowserAdapter.isAccessAllowedWithPartial
Credentials (CommandContext aCommandContext) credentials accepted
flag = P
[4/8/09 11:47:06:389 EDT] 000000a2 WC_SERVER <
63a3dfc9:1207bd65480:-7fee
com.ibm.commerce.browseradapter.HttpBrowserAdapter.isAccessAllowedWithPartial
Credentials (CommandContext aCommandContext) Exit true
[4/8/09 11:47:06:389 EDT] 000000a2 WC_SERVER 3
63a3dfc9:1207bd65480:-7fee
com.ibm.commerce.browseradapter.HttpBrowserAdapter.preInvokeCommand
(ViewCommandContext, HttpRequestAttributes) partial authenticated
user can
access the resource OrderItemAdd
[4/8/09 11:47:06:389 EDT] 000000a2 WC_SERVER <
63a3dfc9:1207bd65480:-7fee
com.ibm.commerce.browseradapter.HttpBrowserAdapter.preInvokeCommand
(ViewCommandContext, HttpRequestAttributes) Exit

```

This shows credAccept was set to "P" and that access is allowed for a user with partial credentials (that is, they are a "remembered" user).

Note: In some cases, not all views that require authentication are secure enough to allow a non-logged in user to display. In this case, leave out the credAccept property and continue to redirect a remembered user to LogonForm prior to invoking the command.

HTTPS property

An https property of 0 means that if a Web request was received over http, it will stay in http. If a Web request was received over https, it stays in https.

Conversely, if the value was 0:1, then this means if the request was received over http, it forces it to be https. If a request was already over https, it stays as https.

In the case of the global-forward mapping, this corresponds to a WebSphere Commerce view, which in turn, maps to a JSP file to be rendered for the view.

Problem case: User is redirected to HTTPS for a request

Again, this example is based on the previous two cases. A user makes the request to OrderItemAdd. However, in this case, they submit the request over http:

```
http://localhost/webapp/wcs/stores/servlet/OrderItemAdd?catalogId=10001
&storeId=10001&langId=-1
```

As per the original configuration, the OrderItemAdd action requires https:

```
<action
  parameter="com.ibm.commerce.orderitems.commands.OrderItemAddCmd"
  path="/OrderItemAdd" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="authenticate" value="0:0,10001:1"/>
  <set-property property="https" value="0:1,10001:1"/>
</action>
```

In this case, although you are not getting an error, and the request is still fulfilled, you are seeing that the original http request is ultimately sent over https (SSL protocol) for a secure connection.

The requestURI comes in as expected in the trace:

```
[4/8/09 12:45:28:550 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fed
com.ibm.commerce.struts.ECActionServlet.processRequest URLINFO -
Method = GET
ServletPath = /servlet
PathTranslated =
/opt/WebSphere/AppServer/profiles/demo/installedApps/
WC_demo_cell/WC_demo.ear/
Stores.war/OrderItemAdd
RequestURI = /webapp/wcs/stores/servlet/OrderItemAdd
PathInfo = /OrderItemAdd
..
Scheme = http
```

In this case, it is important to note that the Scheme at the end is http.

You can see it build a redirect for the same PathInfo (OrderItemAdd), but over https instead:

```
[4/8/09 12:45:28:567 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fed
ServletHelper.buildHttpRedirectUrl3
inUrl=/webapp/wcs/stores/servlet/OrderItemAdd
encoding=UTF-8
[4/8/09 12:45:28:568 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fed
ServletHelper.buildHttpRedirectUrl3 query string before
encryption=WcUseHttps=true&orderId=.&URL=SetPendingOrder
%3FURL%3DOrderCalculate%3FURL%
```

```
3DOrderItemDisplay%26updatePrices%3D1%26calculationUsageId%3D-1
%26orderId%3D.&catEntryId=10001&errorViewName=
ProductDisplayErrorView&quantity=1
```

The redirect is sent back to the browser, and a new request is then seen immediately after, but now with https:

```
[4/8/09 12:45:28:601 EDT] 000000a1 WC_SERVER 3
com.ibm.commerce.webcontroller.RuntimeServletFilter doFilterAction
URLINFO -
Method = GET
ServletPath = /servlet
PathTranslated =
/opt/WebSphere/AppServer/profiles/demo/installedApps/
WC_demo_cell/WC_demo.ear/
Stores.war/OrderItemAdd
RequestURI = /webapp/wcs/stores/servlet/OrderItemAdd
PathInfo = /OrderItemAdd
....
Scheme = https
```

In some cases, you do not want the additional overhead of submitting a secured request over https/SSL, so you want to not force the redirect. In this case, simply update the https property to be 0 for that storeId:

```
<action
  parameter="com.ibm.commerce.orderitems.commands.OrderItemAddCmd"
  path="/OrderItemAdd"
  type="com.ibm.commerce.struts.BaseAction">
  <set-property property="authenticate" value="0:0,10001:1"/>
  <set-property property="https" value="0:1,10001:0"/>
</action>
```

Understanding global-forwards

The global-forwards in the WebSphere Commerce Struts files are used to forward to a view, which is the rendering of a JSP. All global-forwards have a corresponding action-mapping. The action-mapping corresponds to what the request URI would be from the browser, and in the case of a view, it maps to a global-forward definition identifying which JSP should be rendered for that view.

For example, LogonForm is defined first by an action-mapping:

```
<action path="/LogonForm" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="https" value="0:1"/>
</action>
```

This /LogonForm corresponds to what was entered in the browser, such as:

```
http://localhost/webapp/wcs/stores/servlet/LogonForm?storeId=1
```

Since this is a view, the LogonForm action maps to a forward as well:

```
<forward className="com.ibm.commerce.struts.ECActionForward"
  name="LogonForm" path="/LoginForm.jsp">
  <set-property property="resourceClassName"
    value="com.ibm.commerce.command.HttpForwardViewCommandImpl" />
  <set-property property="properties"
    value="generic=true&storeId=no" />
</forward>
```

Notice the name of the forward "LogonForm" matches the path of the action "/LoginForm".

Working case: Request for a view (action-mapping and global-forward)

When a request for a view is made, you get similar behavior to when a controller command is requested, and you have another RequestURI logged. Take an example of LogonForm being requested:

```
https://localhost/webapp/wcs/stores/servlet/LogonForm?catalogId=10001&storeId=10001
&langId=-1
```

This appears as:

```
[4/8/09 10:34:58:936 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.ECActionServlet.processRequest URLINFO -
Method = GET
ServletPath = /servlet
PathTranslated =
/opt/WebSphere/AppServer/profiles/demo/installedApps/
WC_demo_cell/WC_demo.ear/
Stores.war/LogonForm
RequestURI = /webapp/wcs/stores/servlet/LogonForm
PathInfo = /LogonForm
ContextPath=/webapp/wcs/stores
QueryString = catalogId=10001&storeId=10001&langId=-1
```

Again, you can see the Struts request processor looking up the action "/LogonForm" in the configuration:

```
[4/8/09 10:34:58:936 EDT] 000000a1 RequestUtils 1
org.apache.struts.util.
RequestUtilsgetModuleName Get module name for path /servlet
[4/8/09 10:34:58:937 EDT] 000000a1 RequestUtils 1
org.apache.struts.util.
RequestUtilsgetModuleName Module name found: default
[4/8/09 10:34:58:937 EDT] 000000a1 RequestProces 1
org.apache.struts.action.RequestProcessor process Processing a 'GET'
```

```

    for path '/LogonForm'
[4/8/09 10:34:58:937 EDT] 000000a1 RequestProces 1
org.apache.struts.action.RequestProcessor processActionCreate
Looking
    for Action instance for class com.ibm.commerce.struts.BaseAction
[4/8/09 10:34:58:937 EDT] 000000a1 RequestProces 3
org.apache.struts.action.RequestProcessor processActionCreate
Returning existing Action instance
[4/8/09 10:34:58:937 EDT] 000000a1 WC_SERVER  >
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.BaseAction.execute Entry
[4/8/09 10:34:58:937 EDT] 000000a1 WC_SERVER  3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.BaseAction.execute ActionConfig[path=/
LogonForm,scope=session,type=com.ibm.commerce.struts.BaseAction
https?0:1,10001:1
authenticate?null
saveToken?null
valdiateToken?null

```

The request above mapped to the action-mapping from the `Stores.war/WEB-INF/struts-config-ext.xml`:

```

<action path="/LogonForm" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="https" value="10001:1"/>
  <set-property property="credentialsAccepted" value="0:P"/>
</action>

```

There are two https properties in the trace: 0:1 and 10001:1. The 0:1 comes because you merge the Struts configuration files together, as well as the properties for the same action definitions.

In the `struts-config.xml`, you also had:

```

<action path="/LogonForm" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="https" value="0:1"/>
</action>

```

This action-mapping then corresponds to the global-forward from `Stores.war/WEB-INF/struts-config-ext.xml`, which is specific to store 10001:

```

<forward className="com.ibm.commerce.struts.ECActionForward"
name="LogonForm/10001"
path="/UserArea/AccountSection/LogonSubsection/LogonSetup.jsp"/>

```

Note that this entry takes precedence over the bootstrap entry in `struts-config.xml` due to the ordering of the XML files listed in the referenced `web.xml` above:

```

<forward className="com.ibm.commerce.struts.ECActionForward"
name="LogonForm" path="/LoginForm.jsp">

```

```

    <set-property property="resourceClassName"
value="com.ibm.commerce.command.HttpForwardViewCommandImpl" />
    <set-property property="properties" value="generic=true
&amp;storeId=no" />
</forward>

```

This time, since the action maps to a global-forward rather than a controller command, you see it do the lookup of the forward as well. Note it is looking based on the view name LogonForm, the storeId 10001, and device -1. The device -1 corresponds to a standard browser. If this was a messaging request, you see -3.

```

[4/8/09 10:34:58:966 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.StrutsActionHelper.locateView
(String,ViewCommandContext,ActionMapping,TypedProperty,ServletContext)
view name = LogonForm
[4/8/09 10:34:58:966 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.StrutsActionHelper.locateView
(String,ViewCommandContext,ActionMapping,TypedProperty,ServletContext)
storeId = 10001
[4/8/09 10:34:58:966 EDT] 000000a1 WC_SERVER >
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.StrutsActionHelper.findForwardConfig
(ViewCommandContext,String,ServletContext) Entry
[4/8/09 10:34:58:966 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.StrutsActionHelper.findForwardConfig
(ViewCommandContext,String,ServletContext) context.device= -1
[4/8/09 10:34:58:966 EDT] 000000a1 WC_SERVER <
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.StrutsActionHelper.findForwardConfig
(ViewCommandContext,String,ServletContext) Exit

```

Here it found the forward definition and parses the properties tied to it:

```

[4/8/09 10:34:58:967 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
StrutsViewExecUnit.locateView generic setting from ECForwardConfig
?true
[4/8/09 10:34:58:967 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
StrutsViewExecUnit.locateView located forward config LogonForm/10001
[4/8/09 10:34:58:967 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
StrutsViewExecUnit.locateView include store dir
[4/8/09 10:34:58:967 EDT] 000000a1 WC_SERVER >
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.StrutsActionHelper.getForwardInstance
(ActionForward,ViewCommandContext,ActionMapping,TypedProperty) Entry
[4/8/09 10:34:58:967 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.ECDefaultActionForwardFactory.getForwardInstance
normal case, no implClassName defined
[4/8/09 10:34:58:967 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
DeviceFormatManager.getDeviceFormatAdapter Get adapter from request
attribute
saveAdapterInstance
[4/8/09 10:34:58:967 EDT] 000000a1 WC_SERVER 3

```

```
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.ECActionForwardInstance.ECActionForwardInstance
isContextRelative=false userStoreDir=true
[4/8/09 10:34:58:967 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.ECActionForwardInstance.ECActionForwardInstance
original path=
/UserArea/AccountSection/LogonSubsection/LogonSetup.jsp
device path= null redirect= false
```

Since you did not specify "storeDir=false" in the forward definition for this store, the JSP Store directory will be retrieved and prepended to the path of the JSP from the forward config:

```
[4/8/09 10:34:58:968 EDT] 000000a1 WC_SERVER <
63a3dfc9:1207bd65480:-7ff4
ECActionForward.getInstance Exit storeId=10001
[4/8/09 10:34:58:968 EDT] 000000a1 WC_SERVER >
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.registry.ViewRegistry.determineJspStoreDir(Integer,Integer)
EntryviewStoreId: 10001 storeId: 10001
[4/8/09 10:34:58:968 EDT] 000000a1 WC_SERVER <
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.registry.ViewRegistry.determineJspStoreDir(Integer,Integer)
ExitConsumerDirect
[4/8/09 10:34:58:968 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.ECActionForwardInstance.resolveStorePrefix
store
prefix=ConsumerDirect
[4/8/09 10:34:58:968 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.ECActionForwardInstance.getDocumentPathName
base
path is a jsp
UserArea/AccountSection/LogonSubsection/LogonSetup.jsp
[4/8/09 10:34:58:968 EDT] 000000a1 WC_SERVER 3
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.ECActionForwardInstance.getDocumentPathName
device
modified based doc name is UserArea/AccountSection/LogonSubsection
/LogonSetup.jsp
[4/8/09 10:34:58:969 EDT] 000000a1 WC_SERVER <
63a3dfc9:1207bd65480:-7ff4
com.ibm.commerce.struts.ECActionForwardInstance.getDocumentPathName
Exit
docname is
ConsumerDirect/UserArea/AccountSection/LogonSubsection/LogonSetup.jsp
```

From there, the runtime framework continues on to invoke the JSP for this view correctly.

After seeing the working case of when a global-forward is invoked, you can focus on the first problem that may arise for forwards.

Problem case: Missing global-forward definition

Take the example of the LogonPage view again. The previous section showed what

happens if the action definition was missing from Struts. In the case of a view, you also need a corresponding global-forward definition. In this case, take the example of the LogonPage request:

```
https://localhost/webapp/wcs/stores/servlet/LogonPage?catalogId=10001&storeId=10001
&langId=-1
```

You have already defined the action in this case:

```
<action path="/LogonPage" type="com.ibm.commerce.struts.BaseAction">
  <set-property property="https" value="0:0"/>
</action>
```

When requesting the URL above, you end up at the generic error page.

Looking at the trace to debug, you see the request comes in as normal again:

```
[4/8/09 13:11:21:098 EDT] 00000062 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.ECActionServlet.processRequest URLINFO -
Method = GET
ServletPath = /servlet
PathTranslated =
/opt/WebSphere/AppServer/profiles/demo/installedApps/
WC_demo_cell/WC_demo.ear/
Stores.war/LogonPage
RequestURI = /webapp/wcs/stores/servlet/LogonPage
PathInfo = /LogonPage
ContextPath=/webapp/wcs/stores
```

It appears that the action definition has been found:

```
[4/8/09 13:11:21:100 EDT] 00000062 RequestProces 1
org.apache.struts.action.RequestProcessor process Processing a 'GET'
for path
'/LogonPage'
[4/8/09 13:11:21:101 EDT] 00000062 RequestProces 1
org.apache.struts.action.RequestProcessor processActionCreate
Looking for Action instance
for class com.ibm.commerce.struts.BaseAction
[4/8/09 13:11:21:101 EDT] 00000062 RequestProces 3
org.apache.struts.action.RequestProcessor processActionCreate
Returning existing Action
instance
[4/8/09 13:11:21:102 EDT] 00000062 WC_SERVER >
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.BaseAction.execute Entry
[4/8/09 13:11:21:102 EDT] 00000062 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.BaseAction.execute
ActionConfig[path=/LogonPage,
scope=session,type=com.ibm.commerce.struts.BaseAction
https?0:0
authenticate?null
saveToken?null
valdiateToken?null
```

However, when you attempt to look up the global-forward, you cannot find it:

```
[4/8/09 13:11:21:172 EDT] 00000062 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findForwardConfig
(ViewCommandContext,String,ServletContext) context device=-1
defaultDevice= -1
[4/8/09 13:11:21:172 EDT] 00000062 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findForwardConfig
(ViewCommandContext,String,ServletContext)
servletContext.getAttribute
(ECAttributes.ATTR_EC_DEFAULT_DEVICE_TYPE)=-1
[4/8/09 13:11:21:172 EDT] 00000062 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findForwardConfig
(ViewCommandContext,String,ServletContext) call findForwardConfig
with:
baseName,storeId,device,fallbackDevice=LogonPage,10001,null,-1
[4/8/09 13:11:21:172 EDT] 00000062 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findForwardConfig
(ActionMapping,String,Integer,Integer,Integer) storeId= 10001
[4/8/09 13:11:21:172 EDT] 00000062 WC_SERVER >
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findForwardConfigForDevice
Entry
[4/8/09 13:11:21:172 EDT] 00000062 WC_SERVER >
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findStoreSpecificForwardConfig
Entry
[4/8/09 13:11:21:172 EDT] 00000062 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findStoreSpecificForwardConfig
look
for view LogonPage/10001
[4/8/09 13:11:21:173 EDT] 00000062 WC_SERVER <
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findStoreSpecificForwardConfig
Exit
cannot find view LogonPage/10001
[4/8/09 13:11:21:173 EDT] 00000062 WC_SERVER >
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findStoreSpecificForwardConfig
Entry
[4/8/09 13:11:21:173 EDT] 00000062 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findStoreSpecificForwardConfig
look for view LogonPage
[4/8/09 13:11:21:173 EDT] 00000062 WC_SERVER <
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findStoreSpecificForwardConfig
Exit
cannot find view LogonPage
[4/8/09 13:11:21:173 EDT] 00000062 WC_SERVER <
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.findForwardConfigForDevice
Exit
cannot find forward config for device null
[4/8/09 13:11:21:173 EDT] 00000062 WC_SERVER <
63a3dfc9:1207bd65480:-7fe9
com.ibm.commerce.struts.StrutsActionHelper.locateView
(String,ViewCommandContext,ActionMapping,TypedProperty,ServletContext)
Exit
```

```

cannot find config for view LogonPage
[4/8/09 13:11:21:174 EDT] 00000062 WC_SERVER <
63a3dfc9:1207bd65480:-7fe9
StrutsViewExecUnit.locateView Exit
cannot located forward config
[4/8/09 13:11:21:174 EDT] 00000062 CommerceSrvr A
StrutsViewExecUnit
StrutsViewExecUnit
CMN0203E: Command not found: "LogonPage".

```

This results in a "Command not found" `ECApplicationException` shortly thereafter. Viewing the HTML source on the generic error page, you get a similar message at the bottom:

```

Exception Type: 0
Message Key: _ERR_CMD_CMD_NOT_FOUND
Message: CMN3101E The system is unavailable due to "CMN0203E".
System Message: Command not found: "LogonPage".
Originating Command:
Corrective Action:

```

This occurs because you have failed to define the `LogonPage` as a forward in Struts, even though you are trying to implement a view, rather than a controller command URL. To resolve the problem, add a definition for the forward for `LogonPage` into the Struts configuration to match the action definition:

```

<forward className="com.ibm.commerce.struts.ECActionForward"
  name="LogonPage" path="/LoginForm.jsp">
  <set-property property="resourceClassName"
    value="com.ibm.commerce.command.HttpForwardViewCommandImpl" />
  <set-property property="properties"
    value="generic=true&storeDir=no" />
</forward>

```

You are just using the example of `LoginForm.jsp` as the JSP to be rendered here.

Global-forward properties

Many of the properties for global-forward definitions are the same as action definitions, such as `https`. However, there are a few other relevant properties specific to forwards only.

Path

The path specified in the form is the path relative to the Web module root where the JSP resides. For shared definitions, you can include the `storeDir=yes` value in the `properties` property to have the runtime framework check for the JSP path relative to the store directory (as defined in `STORE.DIRECTORY` table) for the `STORE_ID`

of the request, rather than relative to the root of the Web module.

Problem case: Incorrect JSP path defined

Take the above example a step further. Let's say you defined the action already and have added the following to correct the missing forward entry:

```
<forward className="com.ibm.commerce.struts.ECActionForward"
  name="LogonPage" path="/LoginPage.jsp">
  <set-property property="resourceClassName"
value="com.ibm.commerce.
  command.HttpForwardViewCommandImpl"/>
  <set-property property="properties" value="generic=true"/>
</forward>
```

The JSP is placed at `Stores.war/LoginPage.jsp` on filesystem. However, when you invoke the view, you get a "404 error" in the browser saying:

```
JSPG0036E: Failed to find resource /ConsumerDirect/LoginPage.jsp
```

Looking in the trace, you can see that the forward is found and a message saying "include store dir":

```
[4/8/09 13:21:41:806 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe7
StrutsViewExecUnit.locateView located forward config LogonPage
[4/8/09 13:21:41:806 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe7
StrutsViewExecUnit.locateView include store dir
[4/8/09 13:21:41:806 EDT] 00000061 WC_SERVER >
63a3dfc9:1207bd65480:-7fe7
com.ibm.commerce.struts.StrutsActionHelper.getForwardInstance
(ActionForward,ViewCommandContext,ActionMapping,TypedProperty) Entry
[4/8/09 13:21:41:806 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe7
com.ibm.commerce.struts.ECDefaultActionForwardFactory.getForwardInstance
normal case, no implClassName defined
[4/8/09 13:21:41:806 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe7
DeviceFormatManager.getDeviceFormatAdapter Get adapter from request
attribute
saveAdapterInstance
[4/8/09 13:21:41:806 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe7
com.ibm.commerce.struts.ECActionForwardInstance.ECActionForwardInstance
isContextRelative=false userStoreDir=true
[4/8/09 13:21:41:806 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe7
com.ibm.commerce.struts.ECActionForwardInstance.ECActionForwardInstance
original path=/LoginPage.jsp device path= null redirect= false
```

Since it is including the `storeDir`, it looks up the `STORE.DIRECTORY` for store 10001 and determines it is `ConsumerDirect`. It then prepends this directory to the path to the JSP.

```
[4/8/09 13:21:41:807 EDT] 00000061 WC_SERVER >
63a3dfc9:1207bd65480:-7fe7
com.ibm.commerce.registry.ViewRegistry.determineJspStoreDir(Integer,Integer)
  EntryviewStoreId: 0 storeId: 10001
[4/8/09 13:21:41:807 EDT] 00000061 WC_SERVER <
63a3dfc9:1207bd65480:-7fe7
com.ibm.commerce.registry.ViewRegistry.determineJspStoreDir(Integer,Integer)
Exit ConsumerDirect
[4/8/09 13:21:41:807 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe7
com.ibm.commerce.struts.ECActionForwardInstance.resolveStorePrefix
store
prefix=ConsumerDirect
[4/8/09 13:21:41:807 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe7
com.ibm.commerce.struts.ECActionForwardInstance.getDocumentPathName
base path
is a jsp LoginPage.jsp
[4/8/09 13:21:41:807 EDT] 00000061 WC_SERVER 3
63a3dfc9:1207bd65480:-7fe7
com.ibm.commerce.struts.ECActionForwardInstance.getDocumentPathName
device
modified based doc name is LoginPage.jsp
[4/8/09 13:21:41:807 EDT] 00000061 WC_SERVER <
63a3dfc9:1207bd65480:-7fe7
com.ibm.commerce.struts.ECActionForwardInstance.getDocumentPathName
Exit
docname is ConsumerDirect/LoginPage.jsp
```

This JSP, of course, does not exist. It was placed right under the `Stores.war` root, as indicated by the path `"/LoginPage.jsp"`. The problem is that you did not specify `"storeDir=no"` in the properties so it assumes that this is a store-specific implementation and looks for the JSP in the store directory.

To correct this issue, set `"storeDir=no"` in the properties so that it does not prepend the store directory to the specified path to the JSP.

```
<forward className="com.ibm.commerce.struts.ECActionForward"
  name="LogonPage" path="/LoginPage.jsp">
  <set-property property="resourceClassName"
  value="com.ibm.commerce.command.HttpForwardViewCommandImpl"/>
  <set-property property="properties" value="generic=true&
  storeDir=no"/>
</forward>
```

ResourceClassName

The *resourceClassName* property is used for access control purposes only. It is the resource that will be used during an access control check when the forward (or view) is attempting to be invoked by a user.

Properties

The *properties* property is similar to the old way WebSphere Commerce implemented views using the VIEWREG table. This maps to the PROPERTIES field

in that case, and is now used to define additional custom properties for the view itself. The `storeDir` (mentioned above) is one such property that you can specify. Another common one is `generic=true` or `generic=false`. If set to true, then a generic user (`user_id = -1002`) can access this page. If `generic` is set to false, then if the view is invoked by a generic user, it will convert them to a newly generated guest user instead. Commonly, you want `generic` to be false if the page contains user-specific information, which will not be common or shared across all non-registered users.

InterfaceName and ImplClassName

Some forwards have additional properties as well, if a different class other than `ECActionForward` is to be used to invoke the HTTP forward. For example, you can define messaging views as:

```
<forward className="com.ibm.commerce.struts.ECActionForward"
  name="MerchantOrderNotifyView/0/-3"
  path="/MerchantOrderNotification.jsp">
  <set-property property="resourceClassName"
    value="com.ibm.commerce.messaging.
      viewcommands.MessagingViewCommandImpl" />
  <set-property property="properties" value="storeDir=no"/>
  <set-property property="interfaceName"
    value="com.ibm.commerce.messaging.
      viewcommands.MessagingViewCommand"/>
  <set-property property="implClassName"
    value="com.ibm.commerce.messaging.
      viewcommands.MessagingViewCommandImpl" />
  <set-property property="direct" value="true"/>
</forward>
```

This defines a different implementation (`ImplClassName`), which implements a different interface (`InterfaceName`). `InterfaceName` will be invoked when the forward is to be executed. Custom view commands are defined in a similar way.

Special considerations

There are a few key points to note when dealing with Struts within the WebSphere Commerce runtime framework that affect the behavior of requests.

Merging Struts configuration files

Each WebSphere Commerce Web module (`Stores.war`, `CommerceAccelerator.war`, and so on) has its own Struts configuration files within the `WEB-INF` directory. These are all merged during server startup to create one repository. The order the module is loaded during startup indicates which one will take priority. It is important to remember that the out-of-the-box scheduler and messaging views are stored in the `IntializationServlet.war`'s Struts configuration files. You may have merging conflicts if the same actions (with different properties) exist in multiple Web modules.

Extended-sites

The configurations for actions and forwards for the Extended-Sites model work the same general way as the ConsumerDirect (B2C) or AdvancedB2BDirect (B2B) models, with one important difference.

For the B2C and B2B direct models, actions and forwards will be looked up for the STORE_ID matching the parameter of the Web request. If no entry exists, it defaults to the default "STORE_ID = 0" and performs another lookup.

In the case of Extended-Sites, you also need to consider the StorePath, which is defined by the relationships between the extended site customer facing stores and the asset stores. These are defined in STOREREL. The relevant relationships for Struts are:

```
NAME, STRELTYP_ID
com.ibm.commerce.URL, -10
com.ibm.commerce.view, -11
```

The lookups for actions and forwards first check the Extended-site STORE_ID (as passed in from the Web request). If no entry exists it then looks up based on the related STORE_ID's from STOREREL (for example, asset stores), and then reverts to the default STORE_ID = 0, performing another lookup.

Web services

WebSphere Commerce uses `struts-wc-webservices.xml` for Web services definition. Definitions in the `struts-config.xml` or `struts-config-ext.xml` will not be registered when a Web service is executed.

Conclusion

You have now seen different aspects of the Struts configuration and how it pertains to the WebSphere Commerce runtime framework. The action mappings correspond to WebSphere Commerce controller commands, and the global-forwards correspond to a WebSphere Commerce view. An incorrect configuration of each creates different problems that can be difficult to debug. Reviewing application server tracing can help you understand the behavior that is being seen, and how it relates back to the different Struts configuration XML files in the WebSphere Commerce Web modules.

Resources

Learn

- To learn more about the Struts framework, see [Webcast replay: Introduction to the Struts framework with WebSphere Commerce](#).
- Read more about the [WebSphere Commerce Struts](#) framework in the WebSphere Commerce Information Center.
- Refer to the [Apache Struts](#) documentation for more details about Struts.
- If troubleshooting a Struts problem for WebSphere Commerce, refer to [MustGather: WebSphere Commerce Struts framework problems](#).

Discuss

- Participate in the [WebSphere Commerce discussion forum](#).

About the author

Mike Callaghan

Mike Callaghan is a Staff Software Engineer at the IBM Toronto Lab. He has been part of the WebSphere Commerce Support team since 2005, specializing in runtime components. He was also part of the DB2 Development Infrastructure team, automating the build process with UNIX and Perl tooling.