Use events and subscriptions

In this section, you learn how to use events and subscriptions to extend Content Platform Engine functions.

About server extensions

You can extend Content Platform Engine (CPE) functions in the following ways with your own server-based action handlers: Events and Subscriptions, Change Preprocessors, Custom Sweeps, Document Lifecycle Policies, and Automatic Document Classification

The above methods are implemented in either Java or JavaScript. Java interfaces are provided with the product and you create your action handlers by implementing them. The application developer for your company provides the code for the action handlers. As the solution builder, you create required Content Platform Engine objects that use the code to initiate the required actions.

In this course, you will learn about Events and Subscriptions.

What are events and subscriptions?

A subscription is a device for starting a user-implemented, server-side component that extends the core functionality of the Content Platform Engine (CPE).

Events provide a mechanism for initiating actions that are invoked when objects are created, modified, and deleted.

A subscription has the following three elements:

- One or more trigger events

  A specified action on an object in an object store.

  For example, the following actions can be performed on a document: Add document (create), check-in, check-out, update a metadata value, or delete

- Subscription target object

  It is a CPE object upon which the events can be triggered. Examples of objects include documents and folders.

- Event action or workflow (or both)

  It identifies an event action handler that runs when an event is triggered on a target object.

For example, you can have a subscription that notifies you by email (event action) when documents of a particular class (target object) are created (triggered event).

Event actions

You can configure the Content Platform Engine (CPE) to run user-defined code in response to system or custom events. This user-defined code is called an event action handler, which you register with CPE as an event action. By using a subscription, you associate an event action with one or more events and objects.

The code is in Java (or JavaScript) that a developer implements by using the Content Engine API EventActionHandler interface in the Content Engine API.

A handler can be implemented in the following ways:

- A class file that is in the Java virtual machine (JVM) class path
- A class or JAR file that is contained in a code module

Code modules

A code module is a CPE object that contains one or more Java action handlers and any supporting libraries. You can create a code module in Administration Console for Content Platform Engine (ACCE).

Code modules are automatically available when the CPE is deployed to multiple application server instances, or when you move your content metadata from one system to another.

If you modify the code for a Java event action handler that is contained within a code module, you must update any event action that references the code module.

Types of subscriptions

- Event Subscription: runs user-defined code
- Workflow Subscription: launches an IBM FileNet P8 workflow

Define subscription filter

You can create a filter to restrict the application scope of a subscription.

For example, you can filter out creation events that are triggered by check-out events. A creation event occurs when a user adds a document or checks out a document (a new reservation object is created). If you want to do something only when adding a document, you must filter out the creation events that are caused by a check-out by adding the following filter:

```
MajorVersionNumber=1 and
MinorVersionNumber=0)OR(MajorVersionNumber=0 and
MinorVersionNumber=1)
```

The filter in the preceding example applies to the new document object (the source object) that is passed into the event handler. As a new document, it has a version number of 1.0 or 0.1.

Workflow subscription

The workflow subscription starts the workflow event action, which in turn launches a workflow. The subscription specifies a workflow in addition to specifying the trigger event, target object, and event action.

The workflow definition must exist in the object store and must be transferred. A workflow subscription applies to a specific version of a workflow definition. If the workflow definition is updated, then the workflow subscription must be updated as well.

Subscriptions run mode

Event subscriptions can be run synchronously or asynchronously.

- In a synchronous subscription, the operations of the object and the event actions are completed as a single transaction. Failure in either results in rollback of both operations.
- In an asynchronous subscription, the operations of the object and the event actions are completed as separate transactions. Object operation can succeed independently of the event action operations.

Disabling subscriptions

You can disable a subscription without deleting it. For example, you can disable it for testing and while you fix a problem. After you change the event action, re-enable the subscription.

- Deleting a subscription is permanent, but disabling the subscription is temporary.
- For disabled subscriptions, the Enabled column displays the value False.

Activity: Create a subscription with an event action

In this activity, you create a code module with a Java class, an event action, and a subscription for the Order document subclass. The Java class (available as a JAR file) is already created for the student system. You associate the event action with the subscription and test it by creating an Order document. Document creation triggers the subscription and the code is executed which creates an entry in a log file.

In this activity, you will accomplish the following:

- Create an Event Action.
- Create a Subscription.
- Test the Subscription and Event Action.
- Examine the EventLog.txt file.

Create an Event Action.

In this task, you will create an event action and specify the code module. The Java class (available as a JAR file) is already created for the student system.

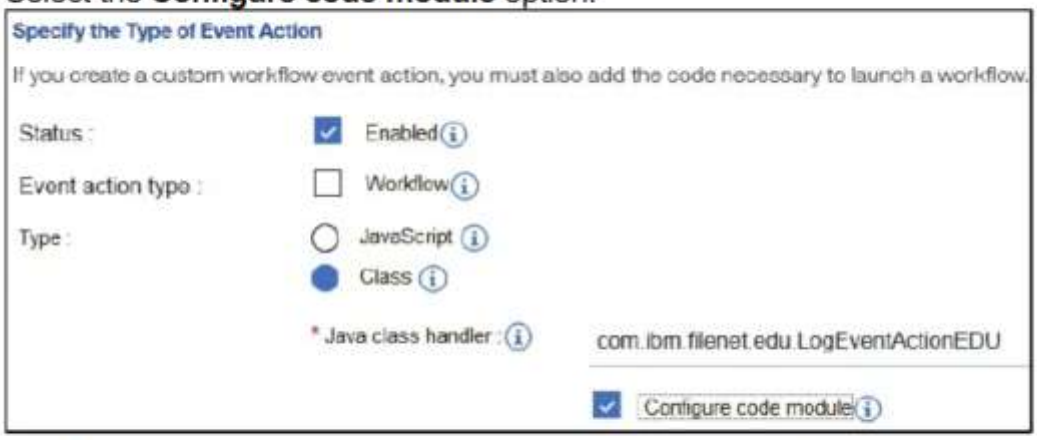
- Ensure that the IBM FileNet P8 Platform components are started.

  If you have not started them earlier, start the components by using the earlier activity: *Prepare your system - Start IBM FileNet P8 Platform.*

- In the **Mozilla Firefox** browser, click the **ACCE** bookmark or type the following URL: **http://vclassbase:9080/acce**
- Type **p8admin** for the **User name** field, **FileNet1** for the **Password** field, and then click **Log In**.
- On the left pane of the **EDU_P8** tab, expand the **Object Stores** folder and click the **Sales** object store.
- From the **Sales** tab, expand the **Sales** > **Events, Actions, Processes** node on the left pane and then click **Event Actions**.
- From the **Event Actions** tab on the right pane, click **New**.
- On the **New Event Action** tab, type **Log Event Action** in the **Display name** field and then click **Next**.
- Make sure that the **Enabled** option is selected for the **Status** field and the **Class** option for the **Type** field.
- For the **Java class handler** field, type the following text: **com.ibm.filenet.edu.LogEventActionEDU**

  Type the Java class name exactly as shown because it is case-sensitive.

- Select the **Configure code module** option.



- Click **Next** and then click **Browse** on the **Specify the Code Module** page.
- On the **File Upload** window, navigate to the **C:\Training\F2810G** folder, select **EDULog.jar** and then click **Open**.
- Back on the **New Event Action** tab, for the **Code module title** field, type **Log Event Action**.

- Click **Next**, review the entries that you made on the **Summary** page, and then click **Finish**.
- On the **Success** page, click **Close**.
- On the **Event Actions** tab, click **Refresh**, verify that the event action that you created is listed, and then close the tab.

Create a Subscription.

In this task, you will create a subscription and specify subsctiption behavior.

- On the **Sales** tab, click **Refresh**.
- On the left pane, expand **Sales** > **Data Design** > **Classes** > **Document**, right-click **Order**, and then click **New Subscription** from the list.
- From the **New Subscription** tab on the right, type **Log Subscription** in the **Display name** field, verify the **Description** field, and then click **Next**.
- For the **Scope** field, leave the default option of **Applies to all objects of this class** and then click **Next**.
- For the **Triggers** field, select **Creation Event** from the **Event Name** list and then click **Next**.
- Select **Log Event Action** from the list.

- Click **Next** and on the **Specify Additional Options** page, select the **Enable this subscription** and **Include subclasses** options.
- For the **Filter expression** field, type the following text:

  (MajorVersionNumber=1 and MinorVersionNumber=0) OR (MajorVersionNumber=0 and MinorVersionNumber=1)

  The filter expression ensures that the triggering event occurs only when the document is first added to the repository.

  Because checking in a document can also trigger creation event. You want the event action to get triggered only when you add a document and not when you check in a document.

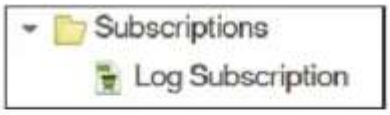
- Click **Next** and on the **Specify Additional Options** page, select the **Enable this subscription** and **Include subclasses** options.
- For the **Filter expression** field, type the following text:

  (MajorVersionNumber=1 and MinorVersionNumber=0) OR (MajorVersionNumber=0 and MinorVersionNumber=1)

  The filter expression ensures that the triggering event occurs only when the document is first added to the repository.

  Because checking in a document can also trigger creation event. You want the event action to get triggered only when you add a document and not when you check in a document.

- Click **Next**, verify the summary of details, and then click **Finish**.
- On **Success** page, click **Close**.
- From the **Sales** tab, and click **Refresh**.
- On the left pane, expand **Sales** > **Events, Actions, Processes** > **Subscriptions** and then verify that **Log Subscription** is listed.



Test the Subscription and Event Action.

The Java code contains instructions to write an entry into a log file each time a document of the class that is associated with this subscription is created. In this task, you will create a folder and a document and test the subscription.

- From the **Sales** tab, expand **Sales** > **Browse** > **Root Folder** node on the left pane.
- Right-click the **Root Folder** and then click **New Folder**.
- From the **New Folder** tab on the right pane, type **Test Events Folder** for the **Folder name** field.

  Leave the default value (Folder) for the Class field.
- Click **Next,** leave the default values for all other fields, and then click **Next** again.
- On the **Summary** page, click **Finish** and then click **Close** on the **Success** page.
- On the **Sales** tab, click **Refresh** to refresh the object store.
- On the **Sales** tab, expand the **Sales** > **Browse** > **Root Folder** node on the left pane, right-click **Test Events Folder**, and then click **New Document**.

- From the **New Document** tab on the right pane, type **Log Test** as the **Document title**.

- Select **Order** from the list for the **Class** field.

  This step is very important since you configured the Order document class for subscription.

- Clear the **With Content** checkbox and complete the wizard by clicking **Next** several times.

  Leave the default values for all the fields.

- On the **Summary** page, click **Finish** and then click **Close** on the **Success** page.
- From the **Sales** tab, expand **Sales** > **Browse** > **Root Folder** node on the left pane and click **Test Events Folder**.
- From the **Test Events Folder** tab, click **Refresh**.

  Verify that the new document (Log Test) is listed.

- Log out of the administration console and then close the browser.

Examine the EventLog.txt file.

In this task, you will verify the log file for the entry that is created by the event action.

- In **Windows Explorer**, open the following folder: **C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01**.
- Open the **EventLog.txt** file in **Notepad++** and verify that the file contains current date, time and an entry for the **Order** document that you created.

  The code for the Log Action adds a text line to the EventLog.txt file each time that the event action executes.

  When you test it in the admin console (ACCE), it shows the class:

```
EventLog.txt
1  A new document is created on: Tue Feb 26 07:25:37 EST 2019
2  Document class name= Order
3  Document id= {70C42969-0000-C01F-8D63-ABD488621596}
4
```

  If you test in the IBM Content Navigator desktop, document class is shown as GUID:

```
EventLog.txt
1  A new document is created on: Tue Feb 26 07:12:41 EST 2019
2  Document class name= {1567CF74-E5D6-40AC-A603-446CB28AD269}
3  Document id= {A0B82969-0000-C119-B267-FCF06A80FC43}
4
```

- Close the **EventLog.txt** file.

  Optional step: You can add another document by using the steps in the previous task and check the EventLog.txt file for more entries.

```
EventLog.txt
1 A new document is created on: Tue May 14 19:12:25 EDT 2019
2 Document class name= Order
3 Document id= {509EB86A-0000-C91D-8A25-F4AB07068D5F}
4 A new document is created on: Tue May 14 19:18:20 EDT 2019
5 Document class name= Order
6 Document id= {B0A3B86A-0000-C810-B95C-9F9084C993D8}
```

  Optional task: You can add a document in the IBM Content Navigator desktop by using the following steps and check the EventLog.txt file for the log entry.

- In the **Mozilla Firefox** browser, click the **Sample Desktop** bookmark or enter the following URL: **http://vclassbase:9081/navigator**
- Type **p8admin** for the **User name** field, **FileNet1** for the **Password** field, and then click **Log In**.
- On the **Browse** page, click the down arrow next to **LoanProcess** on the upper right and select **Sales** from the list.
- Double-click **Test Events Folder** to open the folder and then click **Add Document** from the toolbar.
- On **Add Document** page, for the **What do you want to save?** field, select **Information about the document** from the list.
- Under the **Properties** section, for the **Class** field, select **Order** from the list and then click **OK**.
- Type **Log Test 5** for the **Document Title** field.

  Title could be any text. Leave the default values (or no values) for all other fields.
- Click **Add** in the lower right corner of the page.
- Back on the **Browse** page, verify that the new document is listed under your folder.
- Log out of ICN **Sample Desktop** and then close the browser.
- In **Windows Explorer**, open the **EventLog.txt** file (in **Notepad++**) from the **C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01** folder.

  Verify that the file contains an additional entry and the document class value is shown as GUID.

Activity: Update the event action with new code module

In an example scenario, your management wants to include the user who creates the document in the event log every time a document is added. Your developer provides the new JAR file that contains the updated code.

In this activity, you will modify the code module to use the new JAR file. You will also update the Event Action that references the code module and test it.

In this activity, you will accomplish the following:

- Update the code module.
- Update the Event Action.
- Test the updated code module

Update the code module.

In this task, you will check out the existing code module and then check in a new version of code module that contains the updated code.

- In the **Mozilla Firefox** browser, click the **ACCE** bookmark or type the following URL: **http://vclassbase:9080/acce**
- Type **p8admin** for the **User name** field, **FileNet1** for the **Password** field, and then click **Log In**.
- On the left pane of the **EDU_P8** tab, expand the **Object Stores** folder and click the **Sales** object store.
- From the **Sales** tab, expand the **Sales** > **Browse** > **Root Folder** node on the left pane and then click **Code Modules**.
- From the **CodeModules** tab on the right pane, click the **Log Event Action** link.
- From the **Log Event Action** tab, click the **Action** button, and then select **Checkin, checkout, cancel** > **Exclusive Check Out**.
- On the **Exclusive Checkout** page, select the **EDULog.jar** option and then click **Checkout**.
- From the **Log Event Action** tab, click the **Action** button again, and then select **Checkin, checkout, cancel** > **Checkin**.
- On the **Check In** window, click **Add**.
- On the **Add Content Element** page, click **Browse**.

- Navigate to **C:\Training\F2810G**, select **EDULogv2.jar,** and then click **Open**.

  This jar file contains the updated code.
- Back on the **Add Content Element** page, click **Add Content**.
- Back on the **Check In** page, scroll down and then click **Check In Major Version**.
- From the **Log Event Action** tab, click **Refresh** and then verify that the new version (**Version: 2.0**) is created.

  Both the tab and the Major version number field shows the version value.
- Leave the **Log Event Action** tab open.

Update the Event Action.

In this task, you will update the event action to associate it with the new version of code module. You will use the object reference for the new code module version.

The Log Event Action tab is already open in the administration console.

- From the **Log Event Action** (code module) tab, open the **Versions** subtab and then select **Log Event Action** for the **version 2**.

  Select the checkbox on the first column.
- From the toolbar, click the **Action** button, and then select **Copy Object Reference**.
- On the left pane, expand **Sales** > **Events, Actions, Processes** > **Event Action** and click **Log Event Action**.
- From the **Log Event Action** tab, click the **Properties** subtab.

  Make sure the tab for event action is selected, because the code module tab also has the same name (but has the version number).
- Click the **Property Name** column header to order the properties in alphabetical order and then scroll down to the **Code Module** property.
- Click the down arrow to the right of the **Code Module** field, then click **Paste Object**.

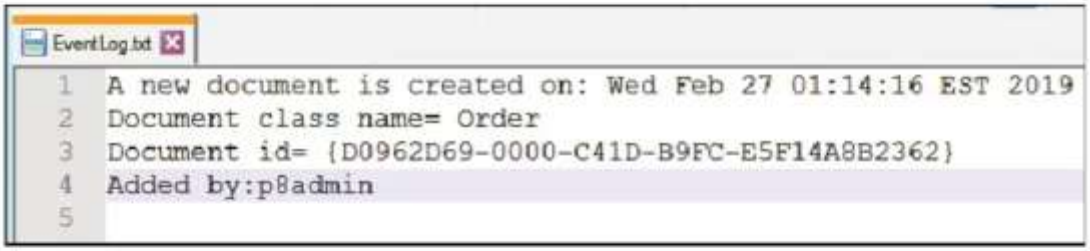
- Click **Save** to save your change to the event action and then click **Close**.
- Close the **Log Event Action** tab.

Test the updated code module.

In this task, you test the updated code module, by adding a document and verifying the log file that is created. You are in Administration Console for Content Platform Engine and the Sales object store is already open.

- In the **Sales** tab, expand **Sales** > **Browse** > **Root Folder** on the left pane, right-click **Test Events Folder** node, and then click **New Document**.
- Enter **Log Update** as the **Document title** and then select **Order** from the list for the **Class** field.
- Clear the **With Content** checkbox and then complete the wizard by clicking **Next** several times.

  Leave the default values for all the fields.
- In the **Summary** page, click **Finish** and then click **Close** to close the **New Document** tab.
- In the **Sales** tab, expand **Sales** > **Browse** > **Root Folder** on the left pane, click **Test Events Folder** node.
- From the **Test Events Folder** tab on the right, click **Refresh** and then verify that the new document (**Log Update**) is listed.
- In **Windows Explorer**, navigate to the **C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01** folder.
- Open the **EventLog.txt** file in **Notepad++** and verify that the file contains current date, time, and an entry for the **Order** document that you created and name of the user.



  The updated code module generates an event log entry that also includes the name of user that added the document. On the student system, you might have more entries depending on the number documents added in the two activities.
- Close the **EventLog.txt** file, log out of the administration console, and then close the browser.