TABLE OF CONTENTS

Tutorial: Building a EDA/CEP “HelloWorld” Application................................................ 3
  IDE HOMEWORK EXERCISE 1. ........................................................................ 9
  IDE HOMEWORK EXERCISE 2. ....................................................................... 15
Tutorial: Building a EDA/CEP “HelloWorld” Application

The media kit installers for Oracle CEP 11gR1 include the Eclipse IDE tooling feature set. Your task is to create an EDA/CEP HelloWorld sample application using this Tooling.

Open the Eclipse (Europa) IDE and start the Oracle CEP server using the green arrow (indicated below).

Once, the Oracle CEP server has successfully started the messages will be shown in the console window.

Create the Oracle CEP EDA HelloWorld Application Project and artifacts by using the “FILE/NEW” menu options as shown below.
The next few simple steps require you to provide a project name (a sample name is shown below) and OSGi bundle properties. The defaults provided can be used.
The final step is to use one of the currently available EDA application templates, in this case, the HELLOWORLD.

After the “HelloWorld” application has been successfully created, you will see the Event Processing Network visualized on the IDE canvass.
Your resulting project should be similar to show below; before we run the HelloWorld Application, let’s explore some of the major artifacts of the generated project.

The Event Processing Network (EPN) is dynamically visualized by the IDE, but can you locate the actual project spring assembly file that “wires” together the EPN?

See below

☐ Locate the Spring Assembly File Artifact

Next, it is very important to review and manipulate the Oracle CEP Processor Continuous Query Language (CQL) queries. In this exercise, we will locate the sample file that has been dynamically created, by the IDE, when the application was built.

☐ Locate the CQL File Artifact
Deploy (publish) your application to the integrated Oracle CEP server.

Once the “HelloWorld” application has been published you will see the generated messages appearing in the console window.
Edit your “HelloWorld” Application to produce its dynamically created messages every 1 minute (1000 ms).

Edit the HelloWorldAdapter.java file and make the change as indicated below.

Remember to SAVE your updates to the application. This can be achieved by using the right mouse button and selecting “save” from the contextual menu.
If you make the change correctly then the application will republish and show the new messages displayed just each minute.

IDE HOMEWORK EXERCISE 1.

The Eclipse IDE “does all the heavy lifting” for you dynamically creating and updating this Spring Assembly file, but take some time to study the various elements and practice later extending the EPN with another processor, and watch the changes made to this file.

Unpublish your “HelloWorld” Application and dynamically create a new processor “listening” to the Channel (Stream) of events generated by the adapter.
Firstly, you will need to save a copy of the first processors CQL rules. We will be covering CQL, in a lot more detail later in this course, but for now follow the steps below.

Now using the dynamic creation capabilities, use the “right” mouse button anywhere on the IDE canvass and create the new processor.
Use the default processor name and file name.

Connect your new EPN as show below by selecting the relevant stream end-point and dragging the mouse pointer to connect to the processor. Then select the processor end-point and drag a line to the output channel stream node.
You can use the EPN layout buttons to optimize the display on the canvass.

Now using the example below, PASTE the rules section (that you copied previously) in the correct location in this new processor file.
Save your changes.
Now it is time to test your new Application EPN solution. Publish the application as shown below.

Once the application has been published (redeployed) you will now see that the console window shows 2 messages for each one generated by the Adapter.

To conclude your tutorial work, unpublish (undeploy) again your application, and shutdown the SERVER.
IDE HOMEWORK EXERCISE 2.

The Eclipse IDE creates a sample CQL file when the application is created. As you learned during this tutorial, your ability to understand the CQL syntax and clauses will be key in Enterprise implementations. Later, study the provided CQL and understand its usage.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<nl:config xmlns:nl="http://www.bea.com/ns/wlevs/config/application"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <processor>
    <name>helloworldProcessor</name>
    <rules>
      <query id="helloworldRule">
        <![CDATA[ select * from helloworldInputChannel [Now] ]]>
      </query>
    </rules>
  </processor>
</nl:config>
```