Configure a SOAScheduler for a composite in SOA Suite 11g

By Robert Baumgartner, Senior Solution Architect ORACLE

November 2010

Prerequisite

Install the bpel-101-HelloWorld from https://soasamples.samplecode.oracle.com/ into your running Oracle SOA Suite 11g. This is our example SOA composite, which will be triggered by our Scheduler. This scheduler will also work with BPMN processes, if they are exposed as service within their SCA composite.

Implementing SOAScheduler in JDeveloper

In this section, we will create a simple a web service proxy component, a scheduler job class that is called by the SOAScheduler and a java class SOASchedulerServlet that initiate the quartz scheduler.

Creating the Web Service Proxy Component

At first we need the WSDL of the example SOA composite. This can be found at the Oracle Enterprise Manager. You will find the Oracle Enterprise Manager at http://server:port/em, e.g. http://localhost:7001/em. Log in with the administration user, e.g. "weblogic". On the farm overview page click on the newly deployed SOA example composite "*bpel-101-HelloWorld*"

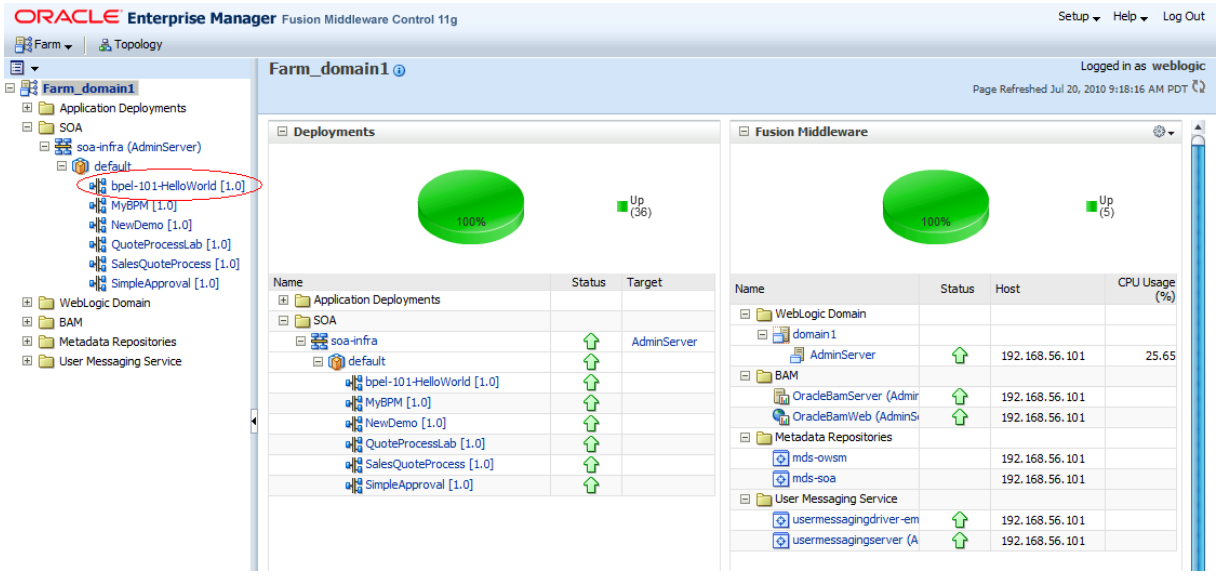

Figure 1

Click on "Show WSDL and endpoint URI".

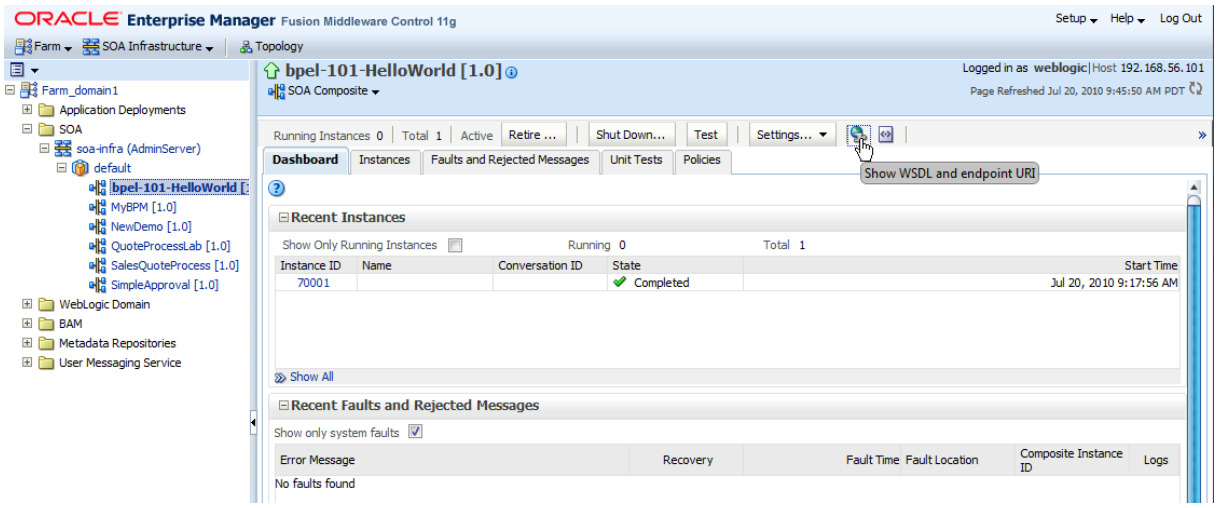

Figure 2

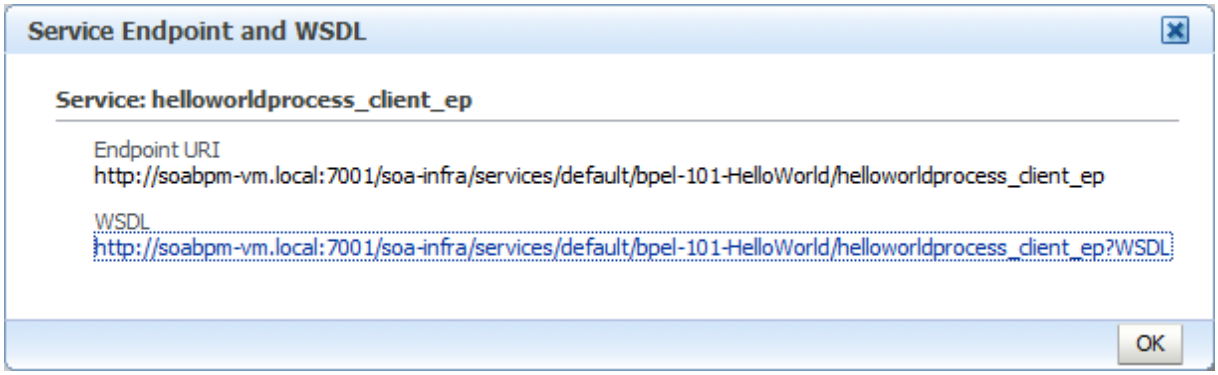

Figure 3

Copy the WSDL location.

To get started, create a new *Application* in JDeveloper by click CTRL-N, name it "*SOAScheduler*", select “Generic Application”...

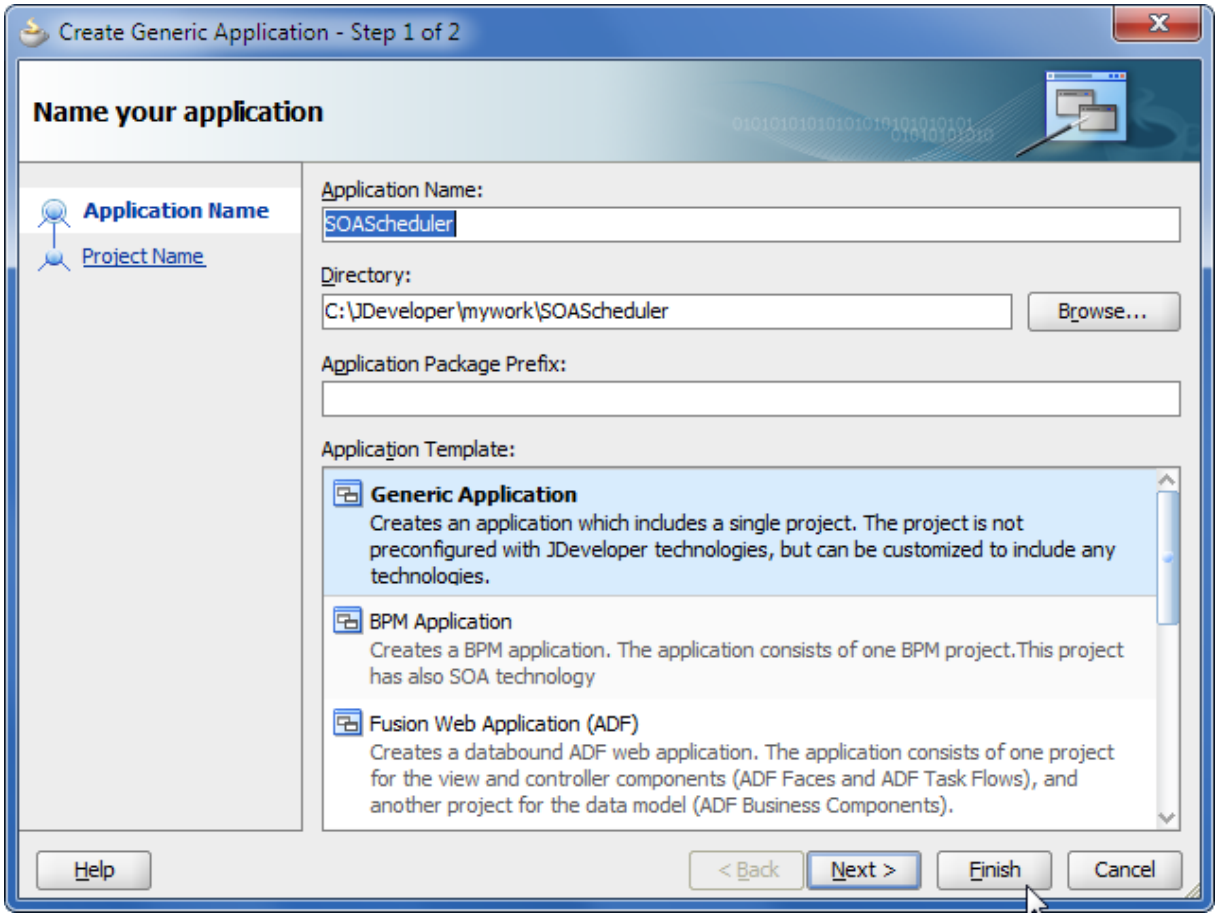

Figure 4

And create a new Project named "*SchedulerProject*", select “Java” and “Web Services” as “Project Technologies”.

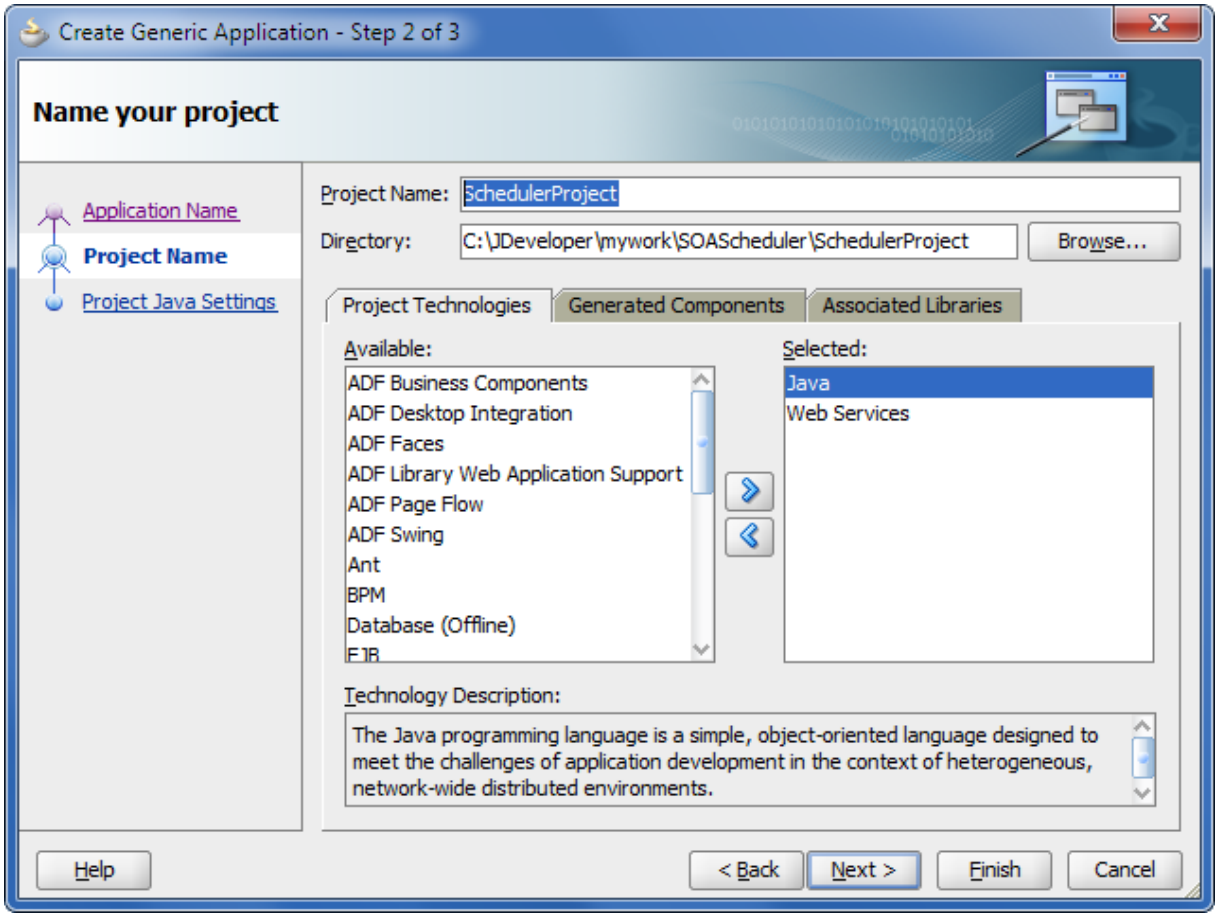

Figure 5

Click "*Finish*".

Create a new Web Service Proxy by click *"File/New" or CRTL-N*.

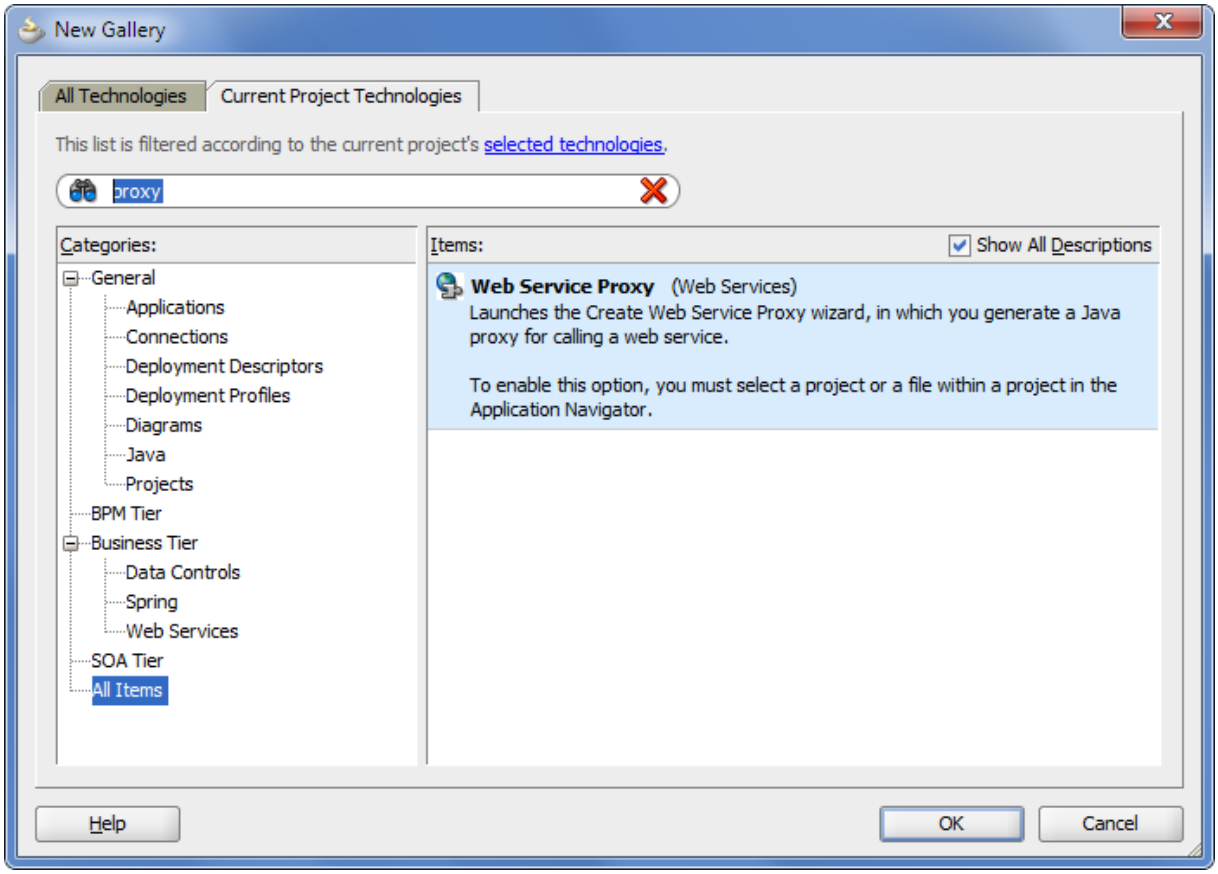

Figure 6

Click "*OK*" and *"Next"*.

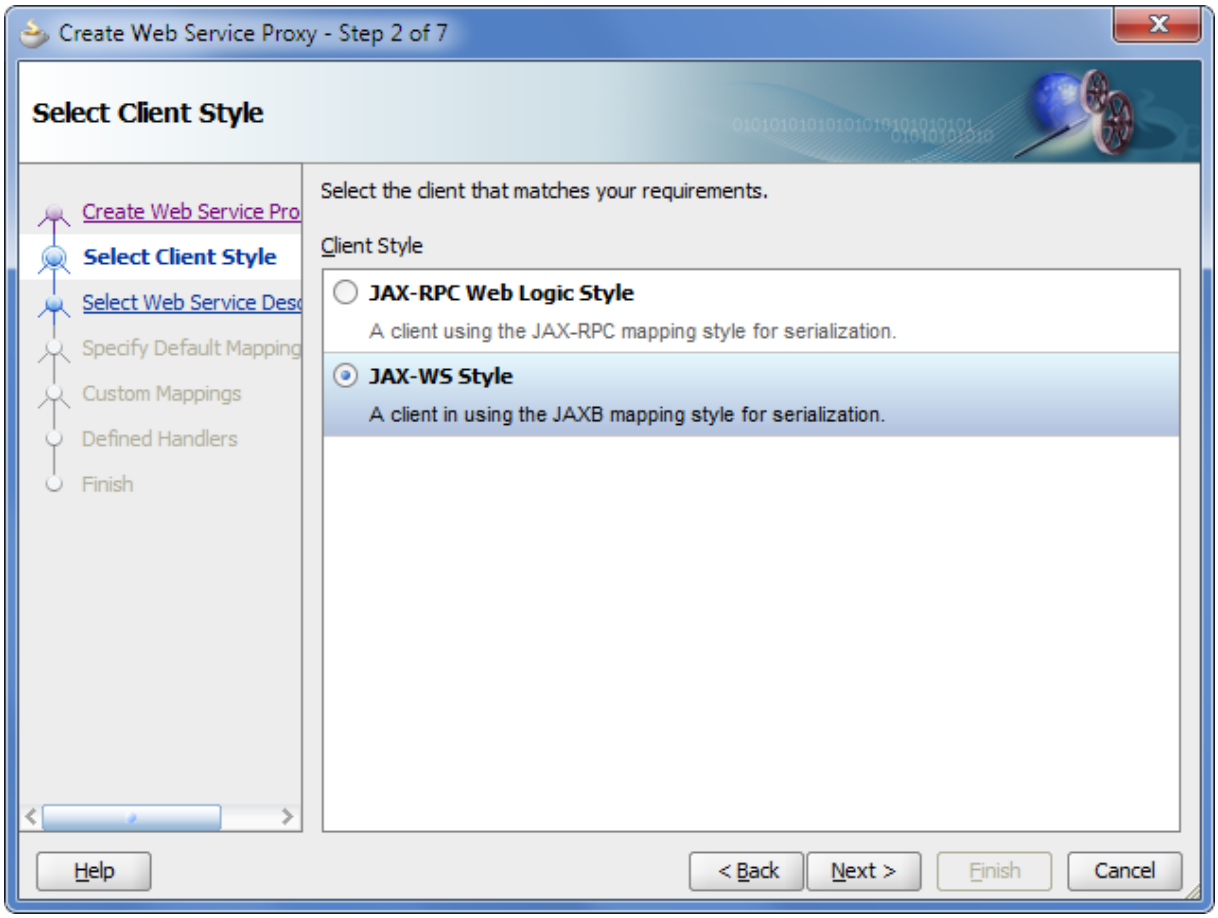

Figure 7

Select "*JAX-WS Style*" and click "*Next*".

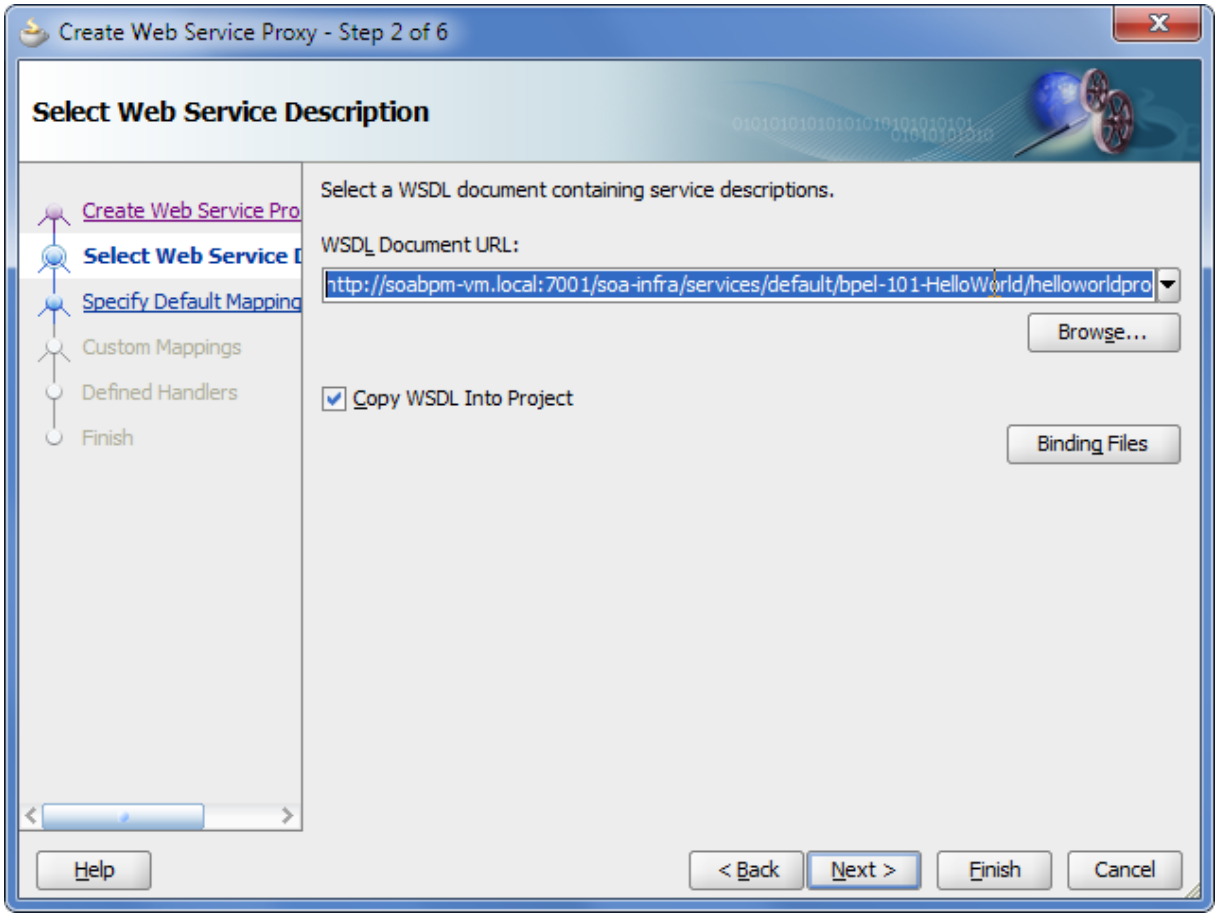

Figure 8

Paste the WSDL location into "*WSDL Document URL*" and click "*Next*".

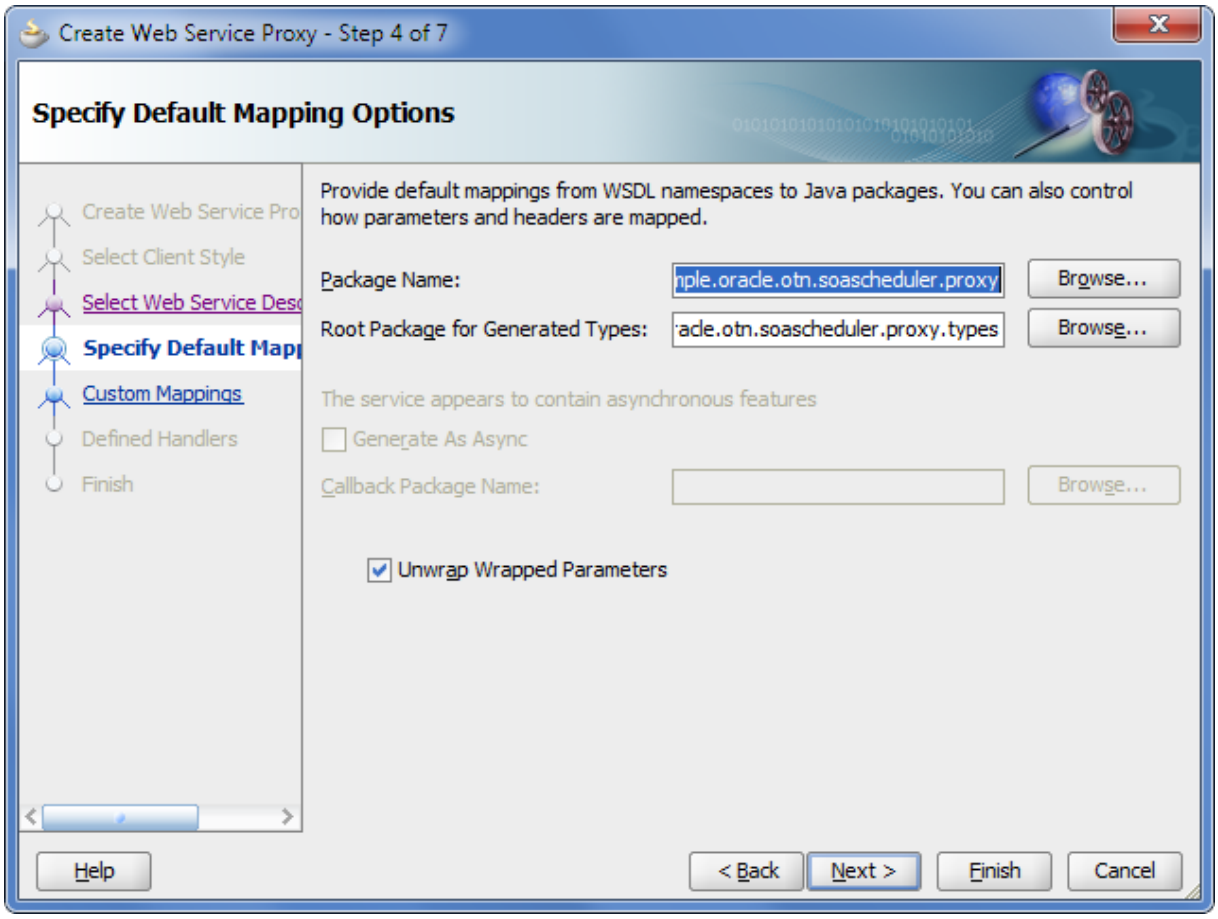

Figure 9

Write "*sample.oracle.otn.soascheduler.proxy*" into "*Package Name*" and "*sample.oracle.otn.soascheduler.proxy.types*" into "Root Package for Generated Types" and click "*Finish*".

In the new HelloWorldProcess_ptClient.java add the Line
System.out.println(helloWorldProcess.process("SOAScheduler"));

```
public static void main(String [] args)
{
  helloworldprocess_client_ep = new Helloworldprocess_client_ep();
  HelloWorldProcess helloWorldProcess = helloworldprocess_client_ep.getHelloWorldProcess_pt();
  // Add your code to call the desired methods.
  System.out.println(helloWorldProcess.process("SOAScheduler"));
}
```

Click on F11 or on the run icon  . The composite will be executed by the new generated service proxy.

In the message window you will see the correct output of SOA example composite.

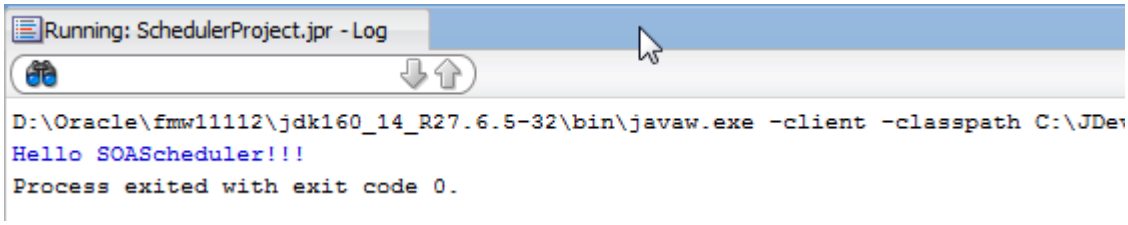

Figure 10

Add Quartz library to the project

Click on "*Application*" → "*Project Properties*", "*Add JAR/Directory…*".

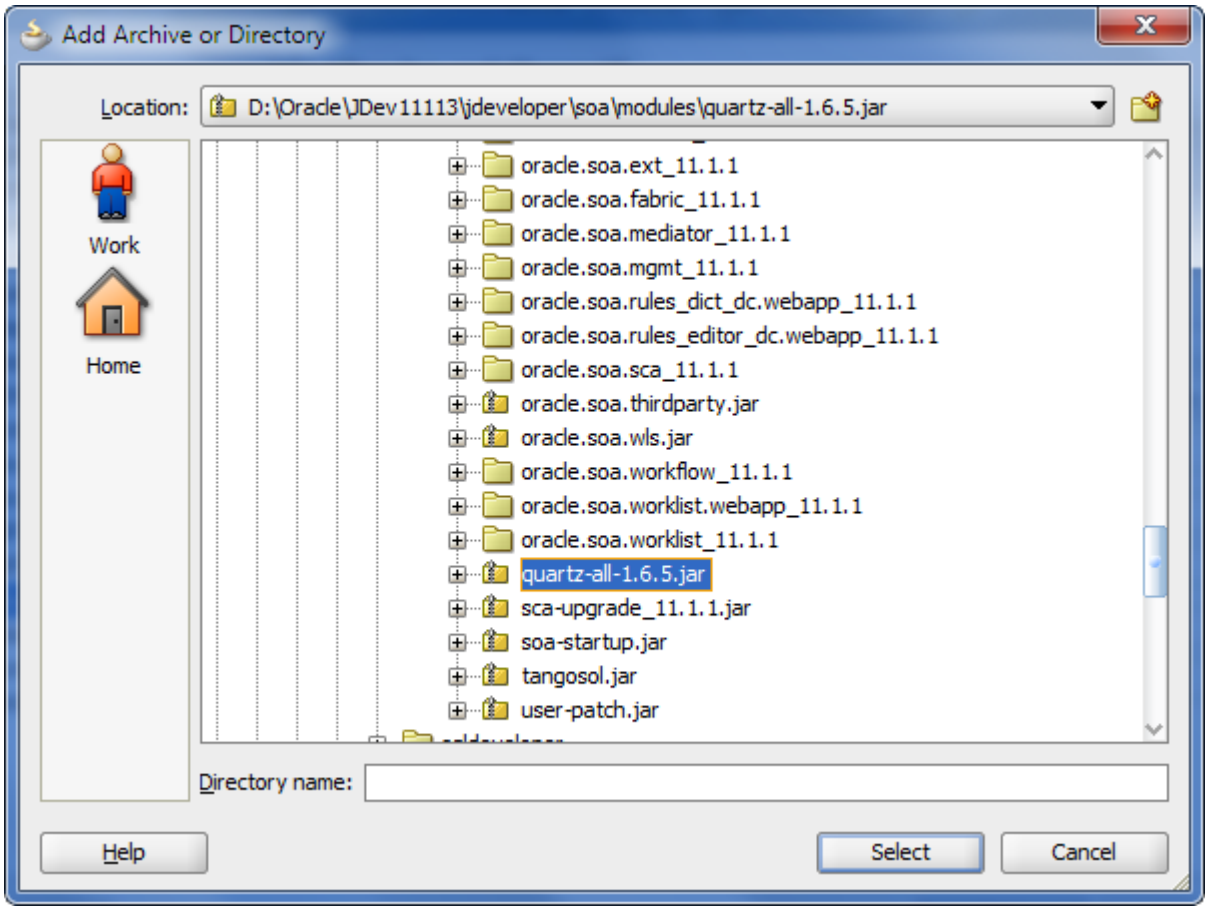

Figure 11

Select in your JDeveloper home *"…\jdeveloper\soa\modules\quartz-all-1.6.5.jar*". Click "*Select*".

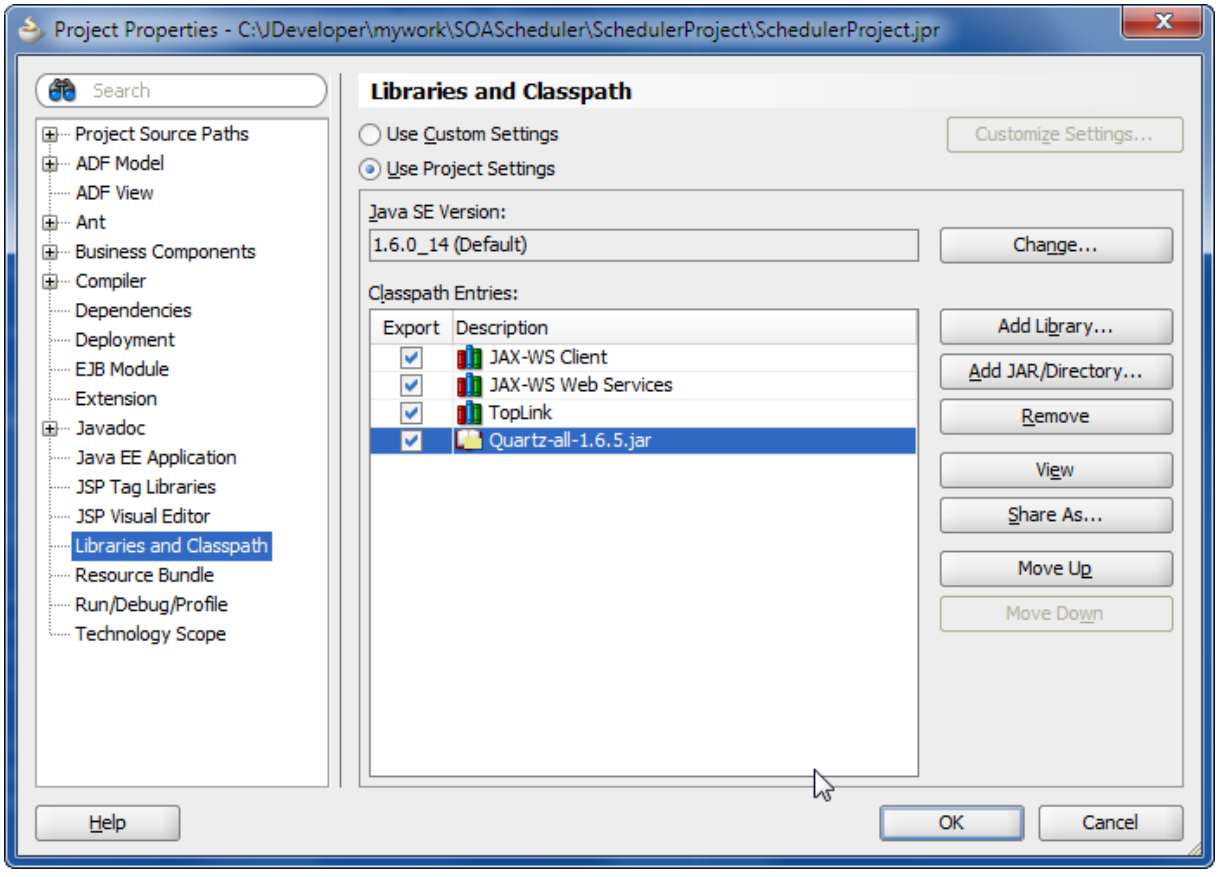

Figure 12

Click "*OK*".

Creating the Scheduler Job Component

Create a new "*Java Class"* in JDeveloper by click *"File/New" or CRTL-N*...

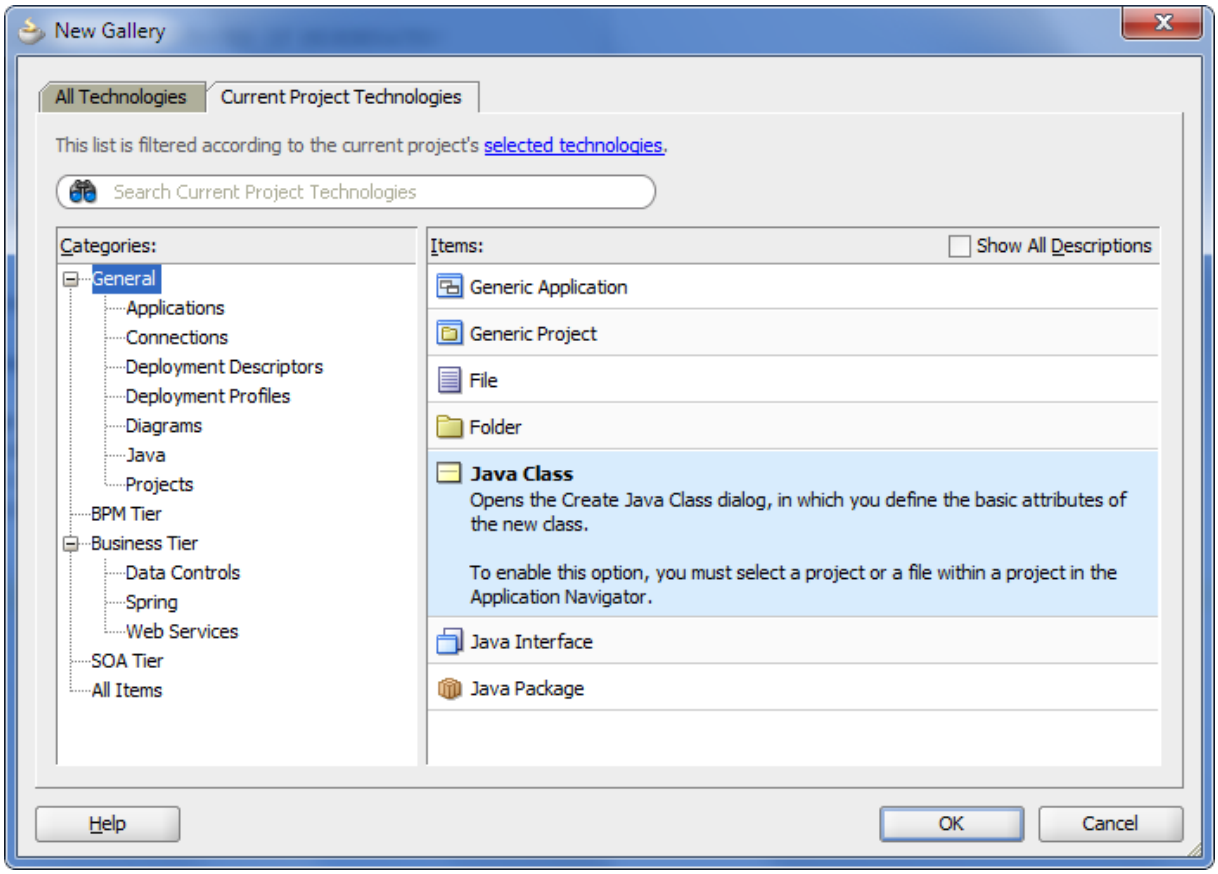

Figure 13

Click *"OK"*.

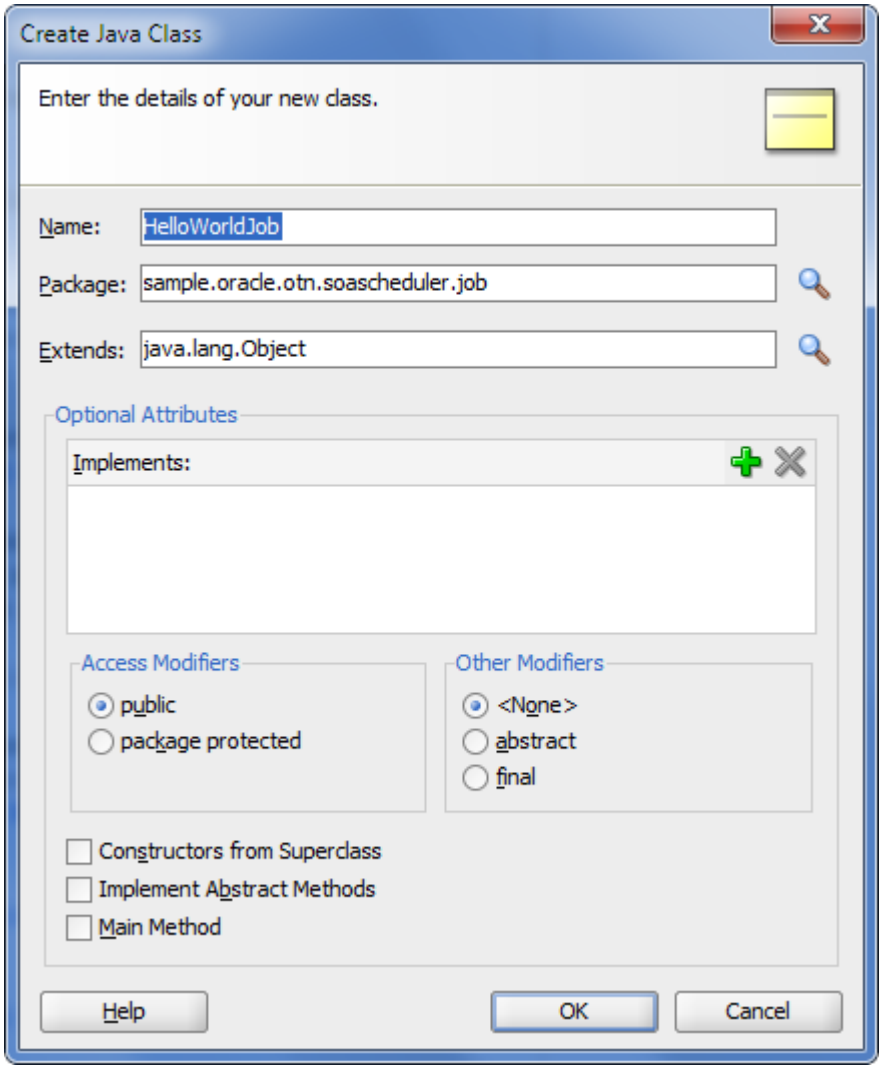

Figure 14

Set the Name to "*HelloWorldJob*" and Package to "*sample.oracle.otn.soascheduler.job*" and click "*OK*".

Replace the generated source with the following lines.

```
package sample.oracle.otn.soascheduler.job;

import java.text.DateFormat;
import java.text.SimpleDateFormat;

import java.util.Date;

import sample.oracle.otn.soascheduler.proxy.Helloworldprocess_client_ep;
import sample.oracle.otn.soascheduler.proxy.HelloWorldProcess;

import javax.xml.ws.WebServiceRef;

import org.quartz.Job;
import org.quartz.JobExecutionContext;

public class HelloWorldJob implements Job{
   @WebServiceRef
   private static Helloworldprocess_client_ep helloworldprocess_client;

   public HelloWorldJob() {
      helloworldprocess_client = new Helloworldprocess_client_ep();
```

```
    }

    public void execute(JobExecutionContext jobExecutionContext) {
      DateFormat df = new SimpleDateFormat("yyyy/MM/dd HH:mm:ss");
      Date date = new Date();
      System.out.println("HelloWorldJob started");
      try {
        helloworldprocess_client = new Helloworldprocess_client_ep();
        HelloWorldProcess helloWorldProcess = helloworldprocess_client.getHelloWorldProcess_pt();
        // Add your code to call the desired methods.
        System.out.println("HelloWorld Response: " + helloWorldProcess.process("SOAScheduler@" +
df.format(date)));

      } catch (Exception e) {
        System.out.println("HelloWorld Process failed: " + e.toString());
          e.printStackTrace();
      }
    }
}
```

Creating the SOASchedulerServlet java class

Create a new *Java Class* in JDeveloper by click *"File/New" or CRTL-N*...

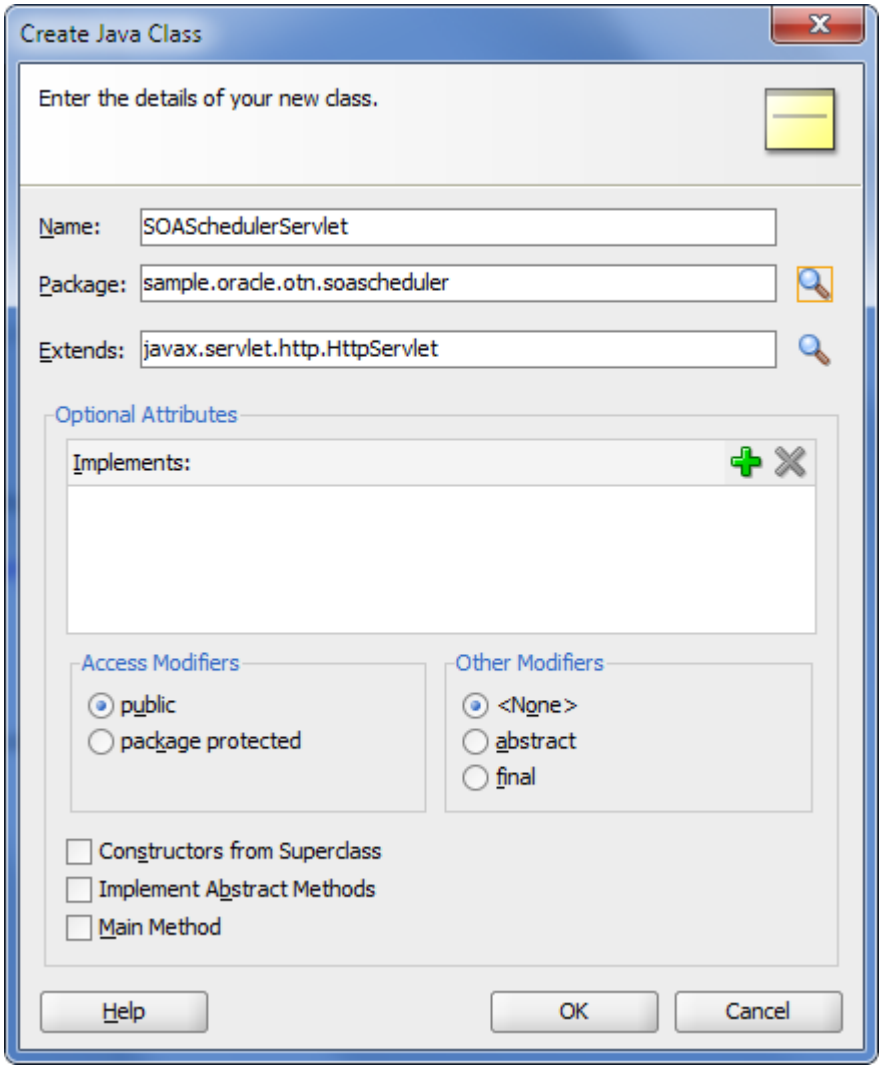

Figure 15

Set the Name to "*SOASchedulerServlet*", Package to "*sample.oracle.otn.soascheduler*", Extends "javax.servlet.http.HttpServlet" and click on "*OK*".

Replace the generated source with the following lines.

```
package sample.oracle.otn.soascheduler;

import sample.oracle.otn.soascheduler.job.HelloWorldJob;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.Date;
import java.util.Calendar;

import java.util.GregorianCalendar;

import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.quartz.CronTrigger;
import org.quartz.JobDetail;
import org.quartz.Scheduler;
import org.quartz.SimpleTrigger;
import org.quartz.Trigger;
import org.quartz.impl.StdSchedulerFactory;
import org.quartz.impl.calendar.AnnualCalendar;

public class SOASchedulerServlet extends HttpServlet {
   StdSchedulerFactory schedFact;
   Scheduler sched;

   public void init(ServletConfig config) throws ServletException {
      super.init(config);
      try {
         schedFact = new StdSchedulerFactory("soa_quartz.properties");
         sched = schedFact.getScheduler();
         System.out.println(this.getClass().getName() + " started");

/*
         // Add the holiday calendar to the schedule
         AnnualCalendar holidays = new AnnualCalendar();

         // fourth of July (July 4)
         Calendar fourthOfJuly = new GregorianCalendar(2011, 7, 4);
         holidays.setDayExcluded(fourthOfJuly, true);
         // halloween (Oct 31)
         Calendar halloween = new GregorianCalendar(2011, 9, 31);
         holidays.setDayExcluded(halloween, true);
         // christmas (Dec 25)
         Calendar christmas = new GregorianCalendar(2011, 11, 25);
         holidays.setDayExcluded(christmas, true);

         // tell the schedule about our holiday calendar
         sched.addCalendar("holidays", holidays, false, false);
*/
         sched.start();
```

```
        JobDetail jd = new JobDetail(JOB_NAME, GROUP_NAME, HelloWorldJob.class);

        CronTrigger cronTrigger = new CronTrigger(TRIGGER_NAME, GROUP_NAME);

        String cronExpr = null;
        // Get the cron Expression as an Init parameter
        cronExpr = getInitParameter("cronExpr");
        System.out.println(this.getClass().getName() + " Cron Expression for " + JOB_NAME + ":" + cronExpr);
        cronTrigger.setCronExpression(cronExpr);
        System.out.println(this.getClass().getName() + " Scheduling Job " + JOB_NAME);
        sched.scheduleJob(jd, cronTrigger);
        System.out.println(this.getClass().getName() + " Job " + JOB_NAME + " scheduled.");

     } catch (Exception e) {
        System.out.println(this.getClass().getName() + e.getLocalizedMessage());
        e.printStackTrace();
     }
  }
public void destroy() {
   try {
      if (sched != null) {
         sched.unscheduleJob(TRIGGER_NAME, JOB_NAME);
         sched.shutdown();
      }
   } catch (Exception e) {
      System.out.println(this.getClass().getName() + " failed to shutdown: " + e.toString());
      e.printStackTrace();
   }
   System.out.println(this.getClass().getName() + " stopped");
}

public void doGet(HttpServletRequest request,
            HttpServletResponse response) throws ServletException,
                                        IOException {
   PrintWriter ajax = new PrintWriter(response.getOutputStream());
   // logger.warning("get");

   String action = request.getParameter("action");
   if ("single".equals(action)) {
      if (sched != null) {
         try {
            Trigger trigger =
               new SimpleTrigger("SOASingleTrigger", GROUP_NAME, new Date());
            trigger.setJobName(JOB_NAME);
            trigger.setJobGroup(GROUP_NAME);

            // Schedule the trigger
            sched.scheduleJob(trigger);
         } catch (Exception e) {
            System.out.println(this.getClass().getName() + e.getLocalizedMessage());
            e.printStackTrace();
         }
      }
   } else if ("start".equals(action)) {
      if (sched != null) {
         try {
            JobDetail jd = new JobDetail(JOB_NAME, GROUP_NAME, HelloWorldJob.class);

            CronTrigger cronTrigger = new CronTrigger(TRIGGER_NAME, GROUP_NAME);
```

```
            // Get the cron Expression as an Init parameter
            String cronExpr = getInitParameter("cronExpr");
            System.out.println(this.getClass().getName() + " Cron Expression for " + JOB_NAME + ":" + cronExpr);
            cronTrigger.setCronExpression(cronExpr);
            System.out.println(this.getClass().getName() + " Scheduling Job " + JOB_NAME);
            sched.scheduleJob(jd, cronTrigger);
            System.out.println(this.getClass().getName() + " Job " + JOB_NAME + " scheduled.");
         } catch (Exception e) {
            System.out.println(this.getClass().getName() + e.getLocalizedMessage());
            e.printStackTrace();
         }
      }
   } else if ("stop".equals(action)) {
      if (sched != null) {
         try {
            sched.unscheduleJob(TRIGGER_NAME, GROUP_NAME);
            System.out.println(this.getClass().getName() + " stopped");
         } catch (Exception e) {
            System.out.println(this.getClass().getName() + " failed to shutdown: " + e.toString());
            e.printStackTrace();
         }
      }
   }

   ajax.println("<html>");
   ajax.println("  <head>");
   ajax.println("    <title>SOAScheduler - Web Interface</title>");
   ajax.println("    <link rel=\"stylesheet\" type=\"text/css\" href=\"css/mystyle.css\"></link>");
   ajax.println("  </head>");
   ajax.println("  <body onload='startAjaxPeriodicalUpdater()'>");
   ajax.println("    <h2>");
   ajax.println("      SOAScheduler @");
   ajax.println("      <span class=\"server\">" + System.getProperty("weblogic.Name") + "</span>");
   ajax.println("    </h2>");

   ajax.println("<table id=\"events_table\" class=\"events_table\" width=\"100%\">");
   ajax.println("<tbody>");

   String[] jobGroups;
   String[] jobsInGroup;
   String[] triggerGroups;
   Trigger[] jobTriggers;
   String[] calendersList;
   AnnualCalendar calen;

   CronTrigger cronTrigger;
   int i, j, k;

   try {

      jobGroups = sched.getJobGroupNames();
      triggerGroups = sched.getTriggerGroupNames();
      calendersList = sched.getCalendarNames();
      for (i= 0; i < calendersList.length; i++) {
         calen = (AnnualCalendar)sched.getCalendar(calendersList[i]);
         //System.out.println("Calendar: " + calendersList[i]);
         ajax.printf("Calendar: " + calendersList[i]);
      }
```

```
        for (i = 0; i < jobGroups.length; i++) {
          // System.out.println("Group: " + jobGroups[i] + " contains the following jobs");
          jobsInGroup = sched.getJobNames(jobGroups[i]);

          for (j = 0; j < jobsInGroup.length; j++) {
            // System.out.println("- " + jobsInGroup[j]);
            jobTriggers = sched.getTriggersOfJob(jobsInGroup[j], jobGroups[i]);

            for (k = 0; k < jobTriggers.length; k++) {
              // System.out.println("- " + triggersInGroup[j]);
              if ("org.quartz.CronTrigger".equals(jobTriggers[k].getClass().getName())) {
                cronTrigger = (CronTrigger)jobTriggers[k];
                ajax.printf("<tr class=\"%s\"><td align=\"left\">Trigger: %s</td><td>Next: %s</td><td>Last:
%s</td><td>Cron: %s</td></tr>",
                        "events", jobTriggers[k].getName(),
                        jobTriggers[k].getNextFireTime(),
                        jobTriggers[k].getPreviousFireTime(),
                        cronTrigger.getCronExpression());

              } else {
                ajax.printf("<tr class=\"%s\"><td align=\"left\">Trigger: %s</td><td>Next: %s</td></tr>",
                        "events", jobTriggers[k].getName(),
                        jobTriggers[k].getNextFireTime());

              }
            }
          }
        }
      } catch (Exception e) {
        System.out.println("SOASchdulerServlet failed: " + e.toString());
        e.printStackTrace();
      }

      ajax.println("</tbody>");
      ajax.println("</table>");
      ajax.flush();
      }

    static final String GROUP_NAME = "SOAGroup";
    static final String TRIGGER_NAME = "SOATrigger";
    static final String JOB_NAME = "SOAJob";
    static final String TARGET_PAGE = "index.jsp";

}
```

Create a quartz property file

Create a new *File (General)* in JDeveloper by click *"File/New" or CRTL-N*...

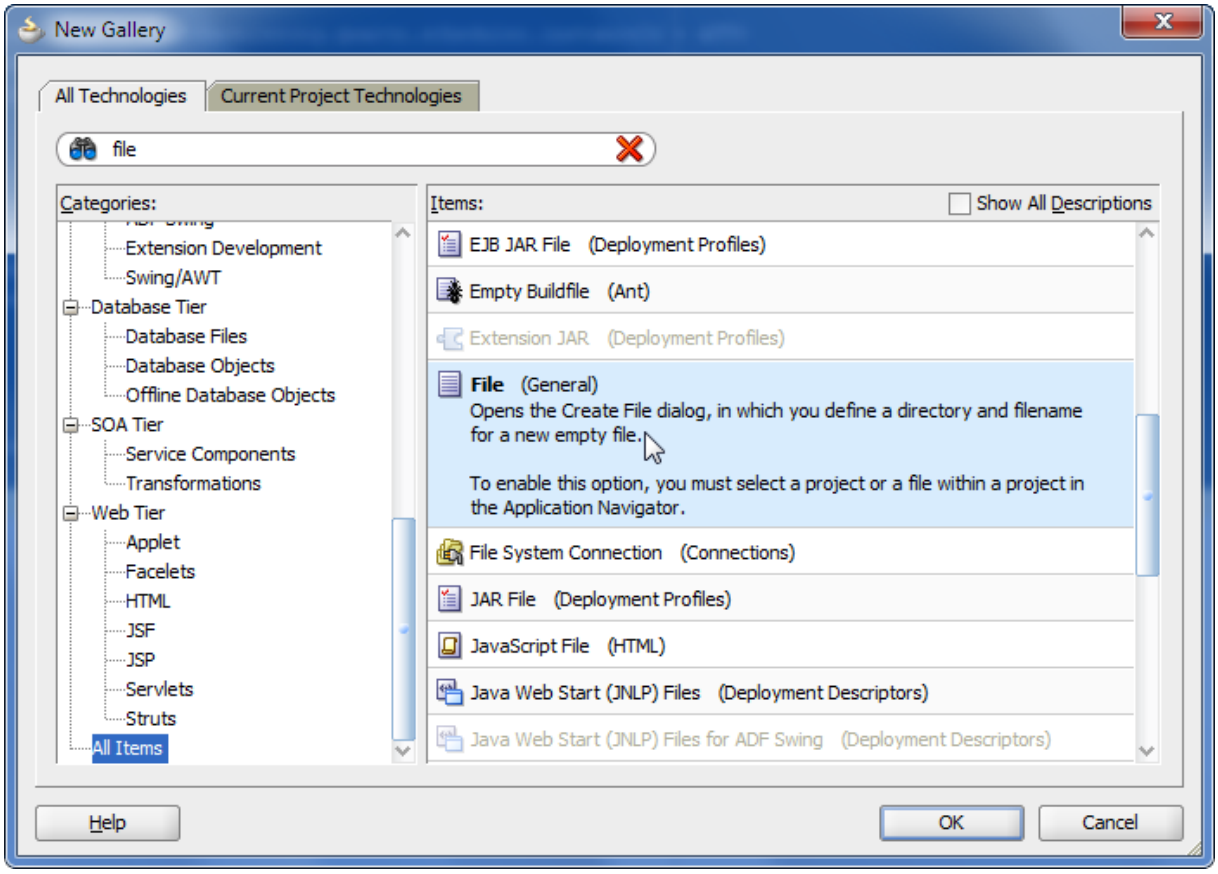

Figure 16

Select *"File (General)"*.

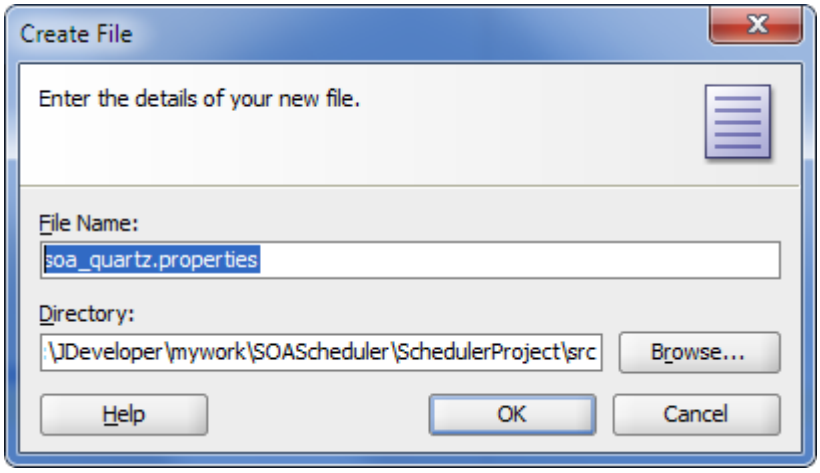

Figure 17

Set File Name to "soa_quartz.properties" and Directory to the src directory of your project. Click *"OK"*.

Insert the following lines into the property file.

```
#
# Configure Main Scheduler Properties
#
org.quartz.scheduler.instanceName = SOASchedulerorg.quartz.scheduler.instanceId = AUTO
org.quartz.scheduler.rmi.export = false
org.quartz.scheduler.rmi.proxy = false
#
# Configure ThreadPool
#
```

```
org.quartz.threadPool.class = org.quartz.simpl.SimpleThreadPool
org.quartz.threadPool.threadCount =  5
org.quartz.threadPool.threadPriority = 4
#
# Configure JobStore
#
org.quartz.jobStore.misfireThreshold = 5000
org.quartz.jobStore.class = org.quartz.simpl.RAMJobStore
```

Create a J2EE Deployment Descriptor (web.xml)

Create a new *web.xml* in JDeveloper by click *"File/New" or CRTL-N*...

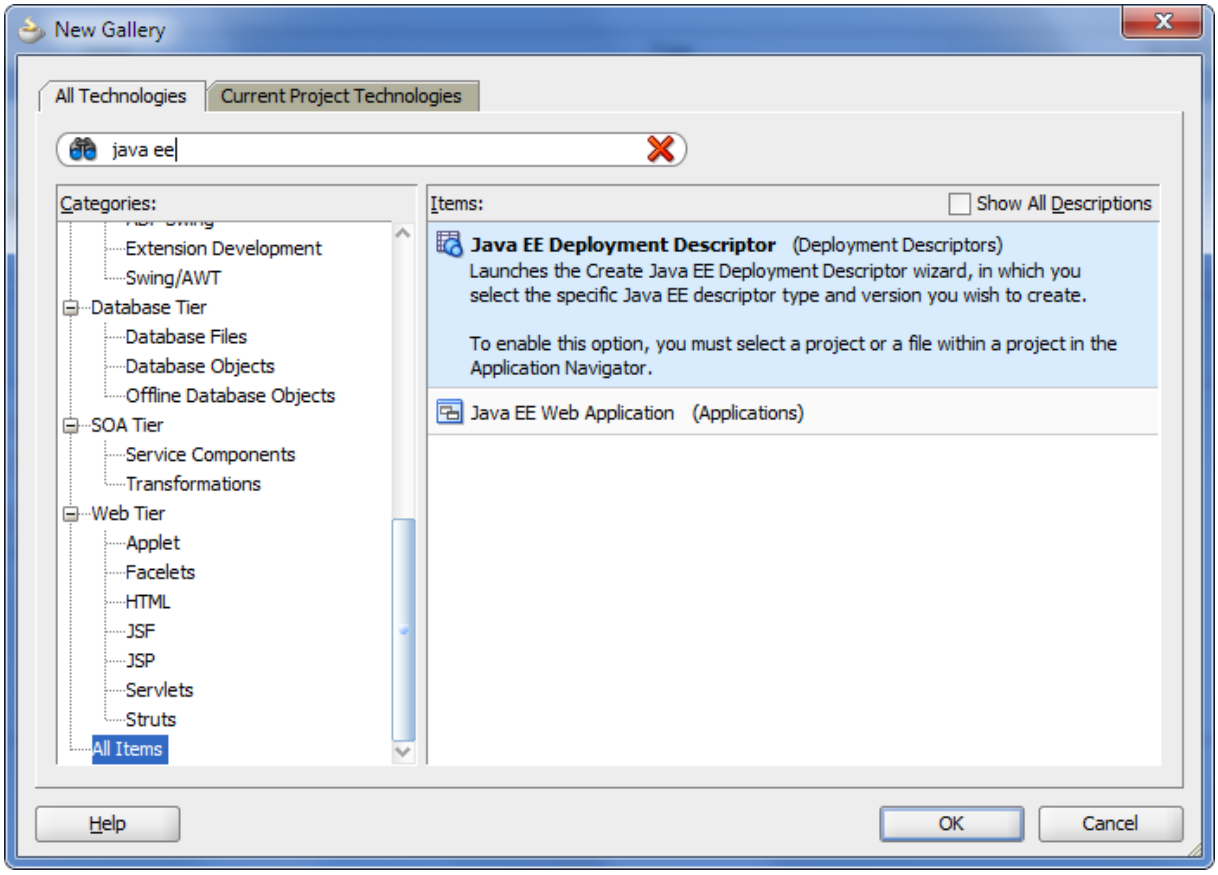

Figure 18

Select "*Java EE Deployment Descriptor*". Click "*OK*".

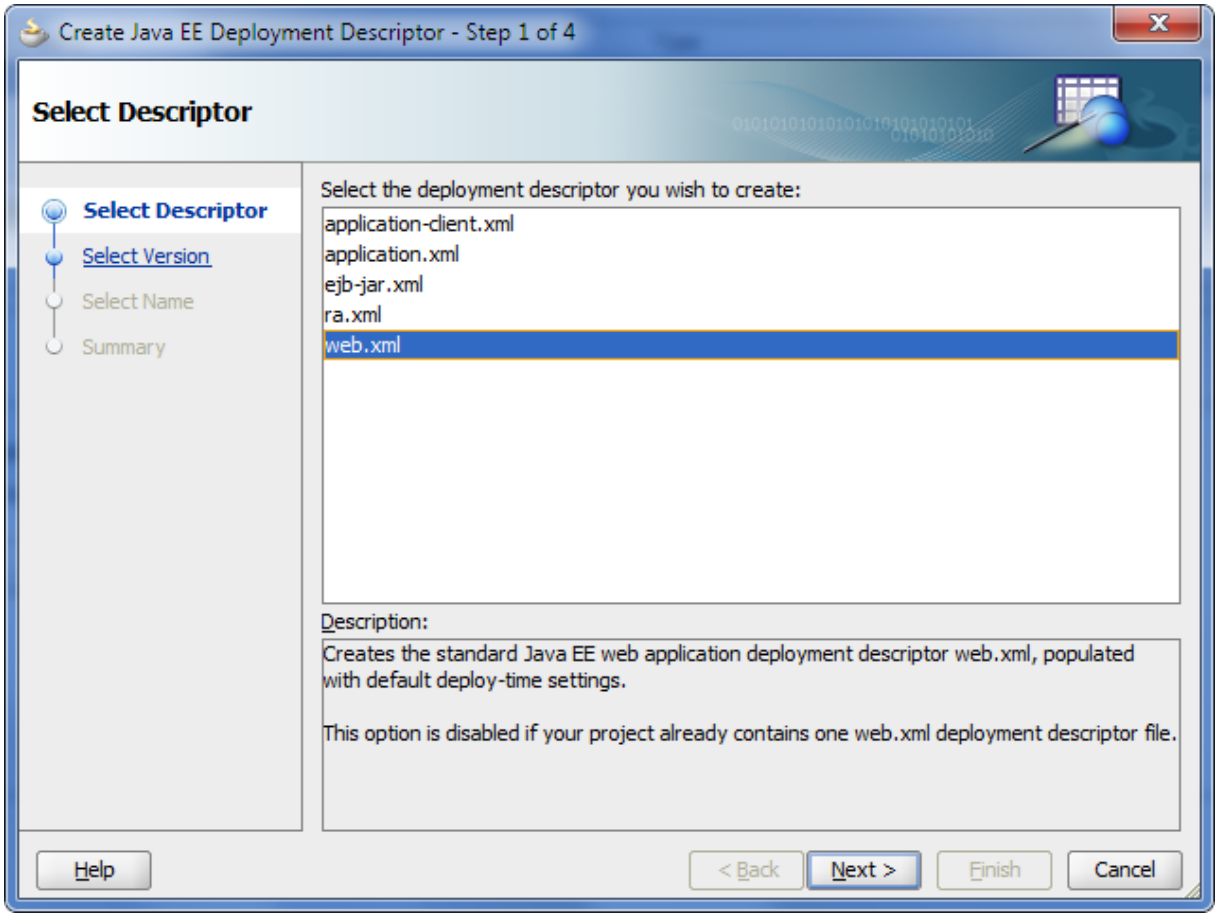

Figure 19

Select "*web.xml*". Click "*Next*".

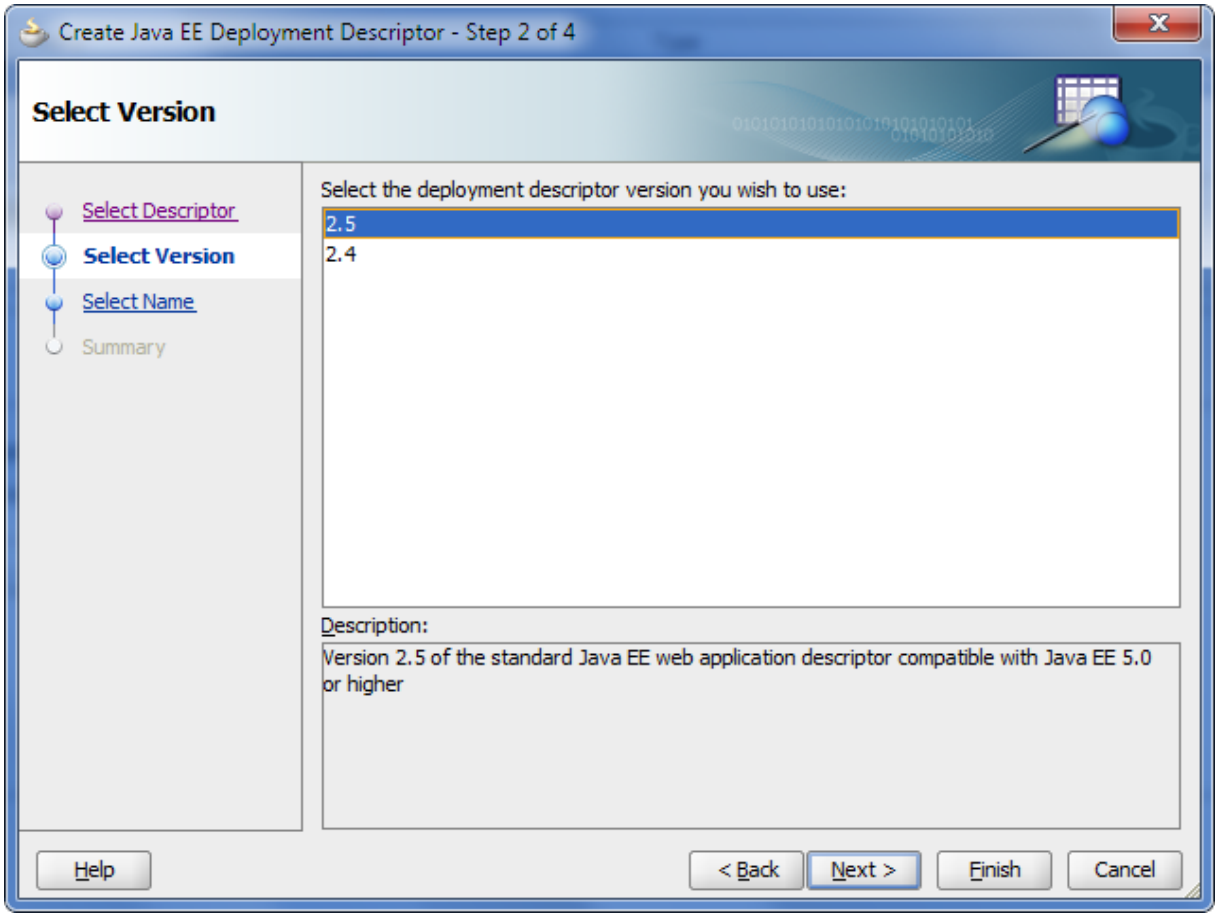

Figure 20

Select "*2.5*". Click "*Finish*".

Select "*Servlets*" and click on the green *"+" (Create Servlet)*.
Set Name to "*SOASchedulerServlet*", Servlet Class to "*sample.oracle.otn.soascheduler.SOASchedulerServlet*". Set *"Load Servlet on"* to *"Application Start"*.
Select "*Servlet Mappings*" and click on the green *"+" (Create "Servlet Mappings")*. Add *"/soaschedulerservlet*" as URL Pattern.
Select "Initialization Parameters" and click on the green *"+" (Create "Servlet Initialization parameters")*. Add "*cronExpr*" as Name and "*0 0,5,10,15,20,25,30,35,40,45,50,55 * * * ?*" as Value.

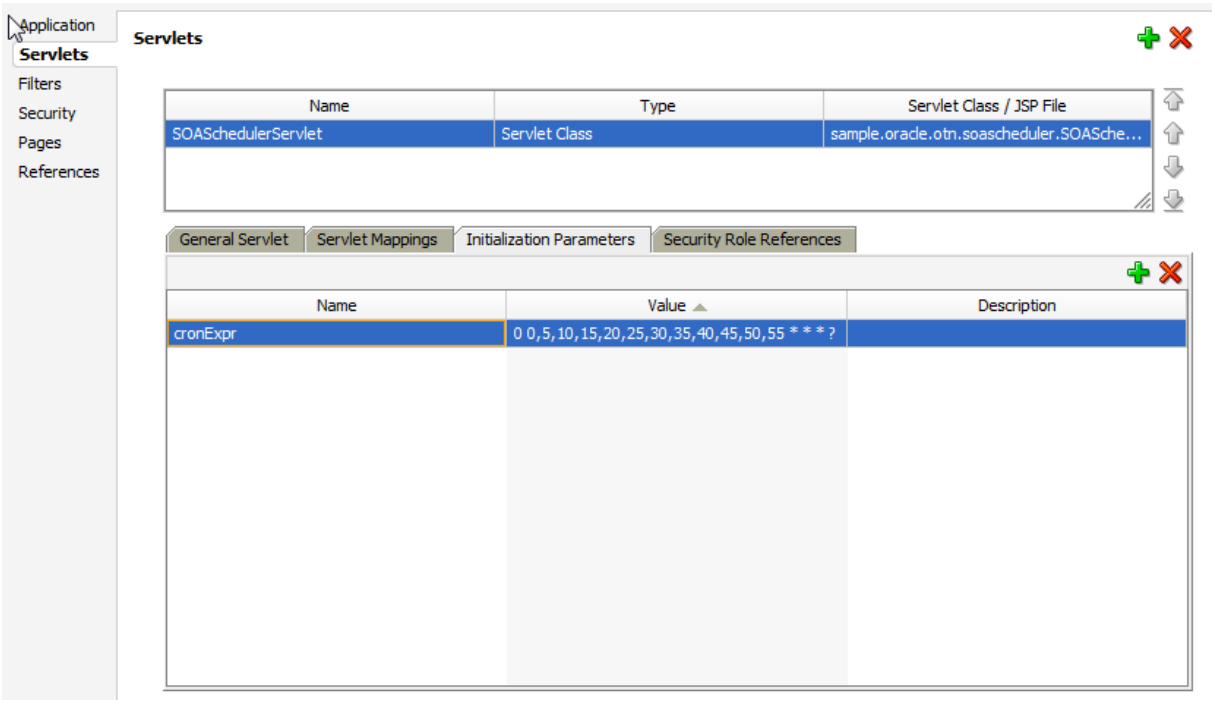

Figure 21

Select "Pages" and click on the green *"+" (Create "Welcome File")*. Add "/soaschedulerservlet".

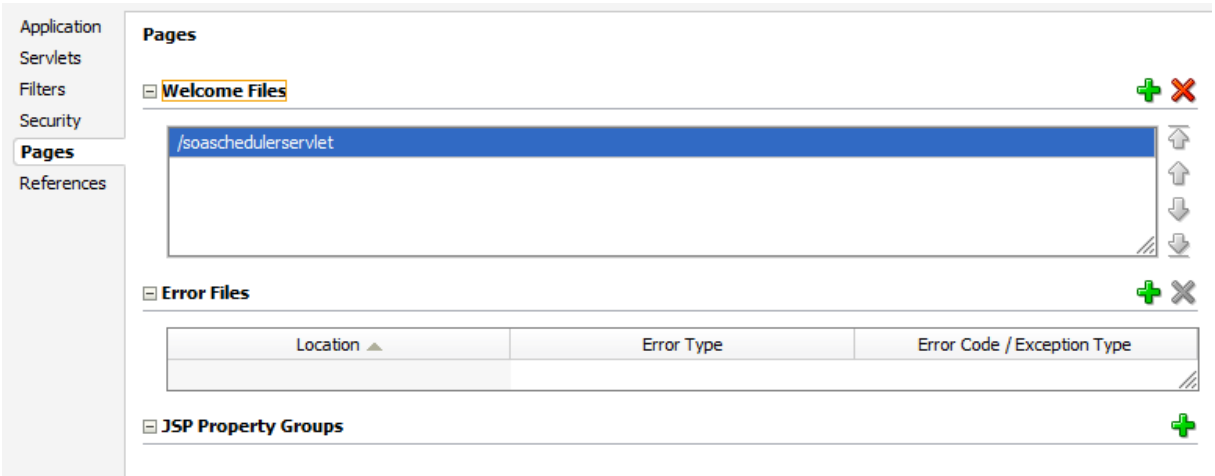

Figure 22

The Source should look like this…

```
<?xml version = '1.0' encoding = 'windows-1252'?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
```

```
      version="2.5" xmlns="http://java.sun.com/xml/ns/javaee">
  <servlet>
    <servlet-name>SOASchedulerServlet</servlet-name>
    <servlet-class>sample.oracle.otn.soascheduler.SOASchedulerServlet</servlet-class>
    <init-param>
      <param-name>cronExpr</param-name>
      <param-value>0 0,5,10,15,20,25,30,35,40,45,50,55 * * * ?</param-value>
    </init-param>
    <load-on-startup>0</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>SOASchedulerServlet</servlet-name>
    <url-pattern>/soaschedulerservlet</url-pattern>
  </servlet-mapping>
  <welcome-file-list>
    <welcome-file>/soaschedulerservlet</welcome-file>
  </welcome-file-list>
</web-app>
```

Click *"Save All"*.

Attention: The name "*cronExpr*" is requested in the SOASchedulerServlet.java, and the value means, that the SOA composite bpel-101-helloworld is started every five minutes every day. For other values you will find more examples later.

Create WAR File (Deployment Profile)

Create a new "*WAR File*" in JDeveloper by click *"File/New" or CRTL-N*...

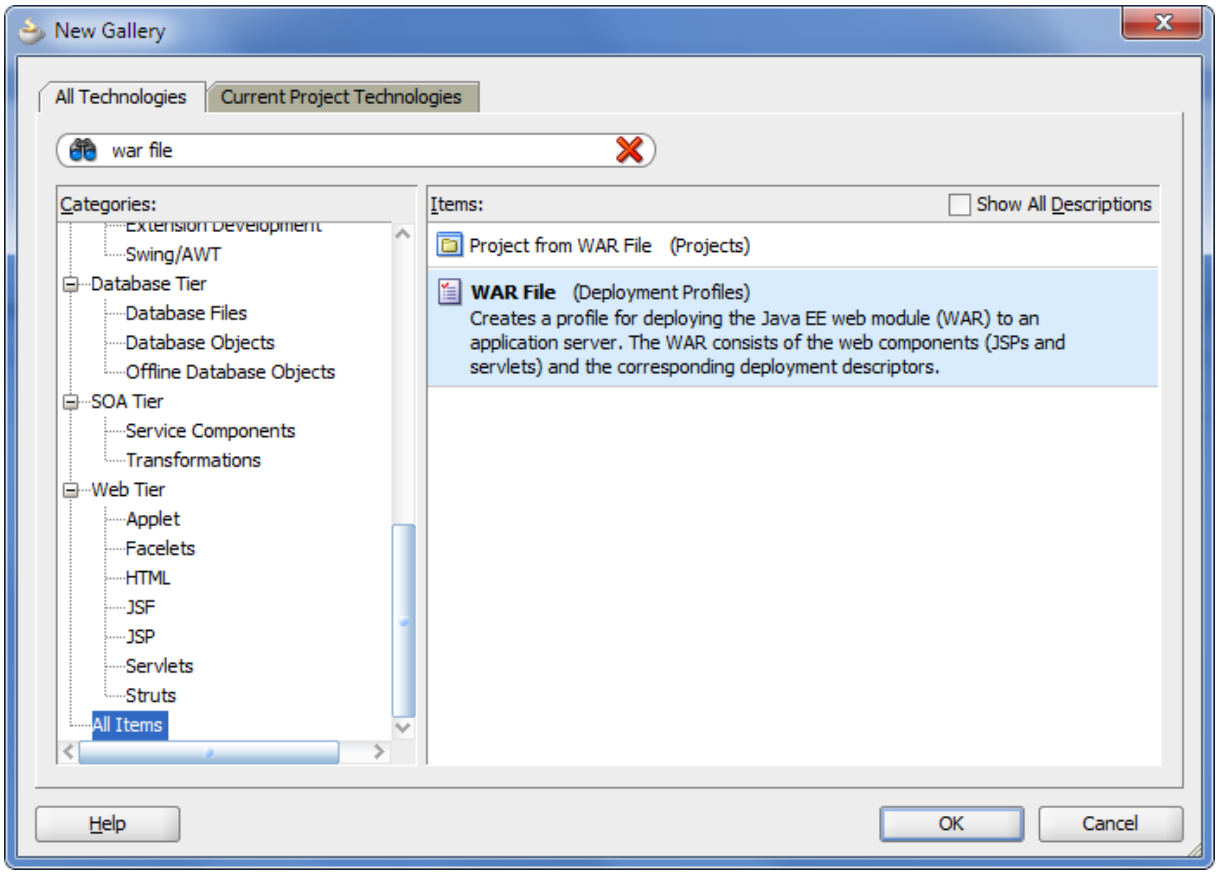

Figure 23

Click *"OK"*.

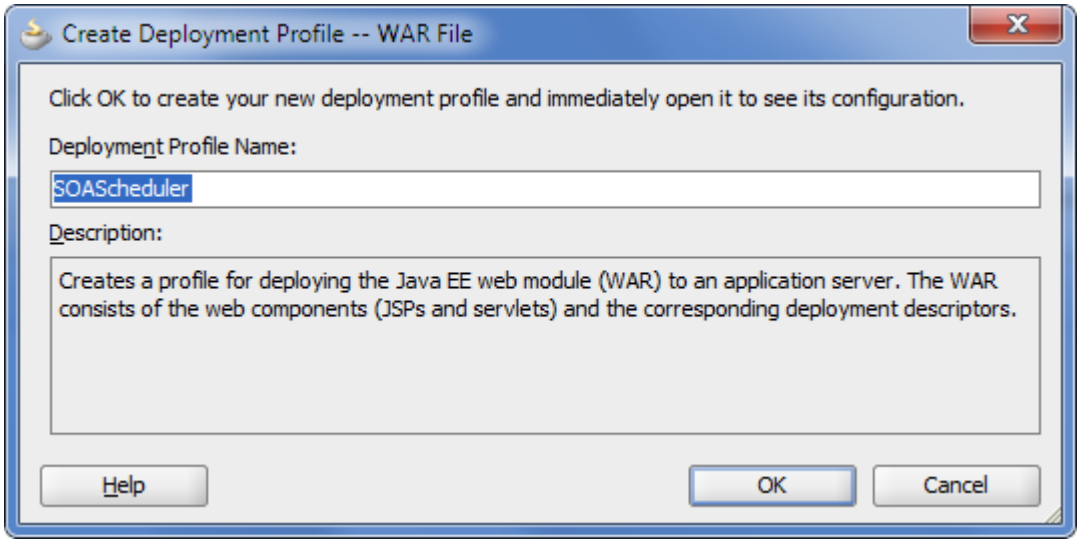

Figure 24

Set the Profile Name to "SOAScheduler". Click *"OK"*.

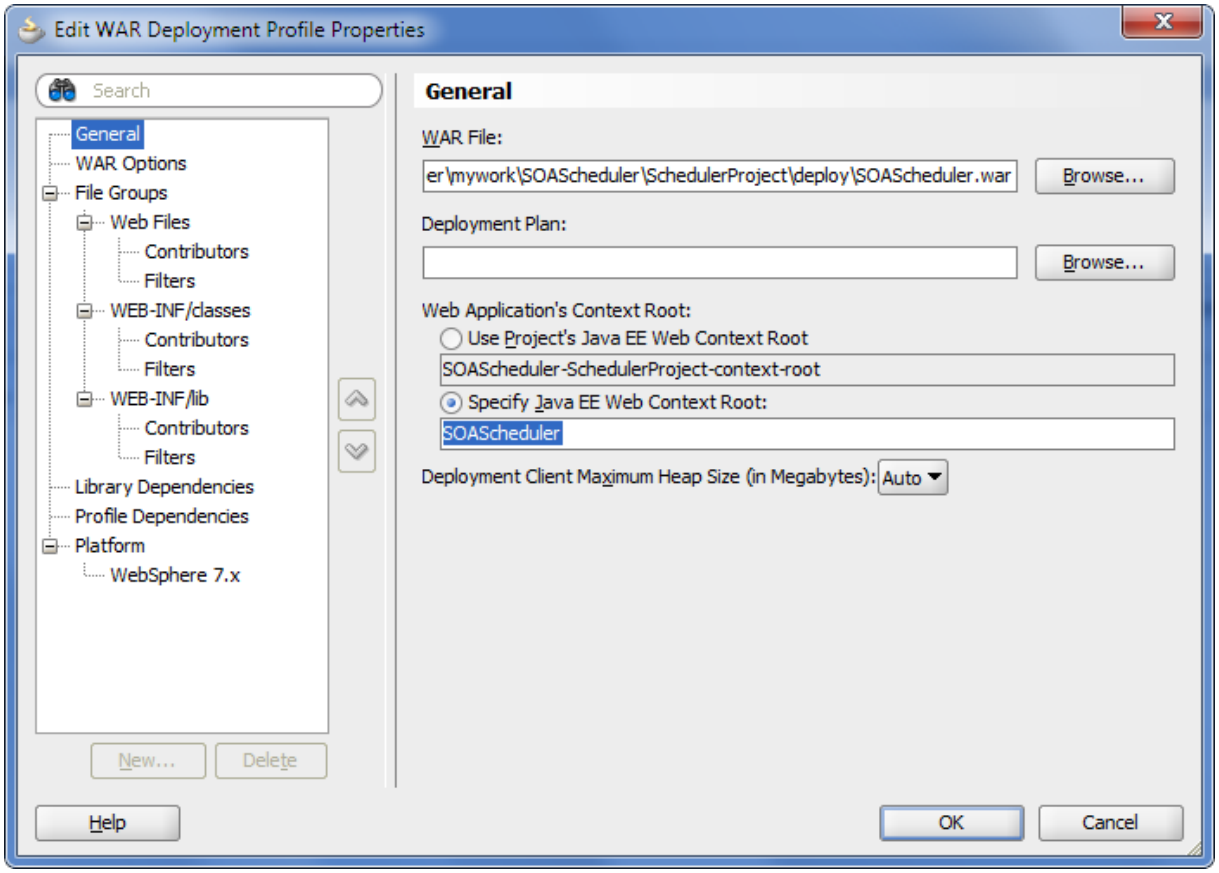

Figure 25

On the "General" tab set the "Web Context Root" to "SOAScheduler".

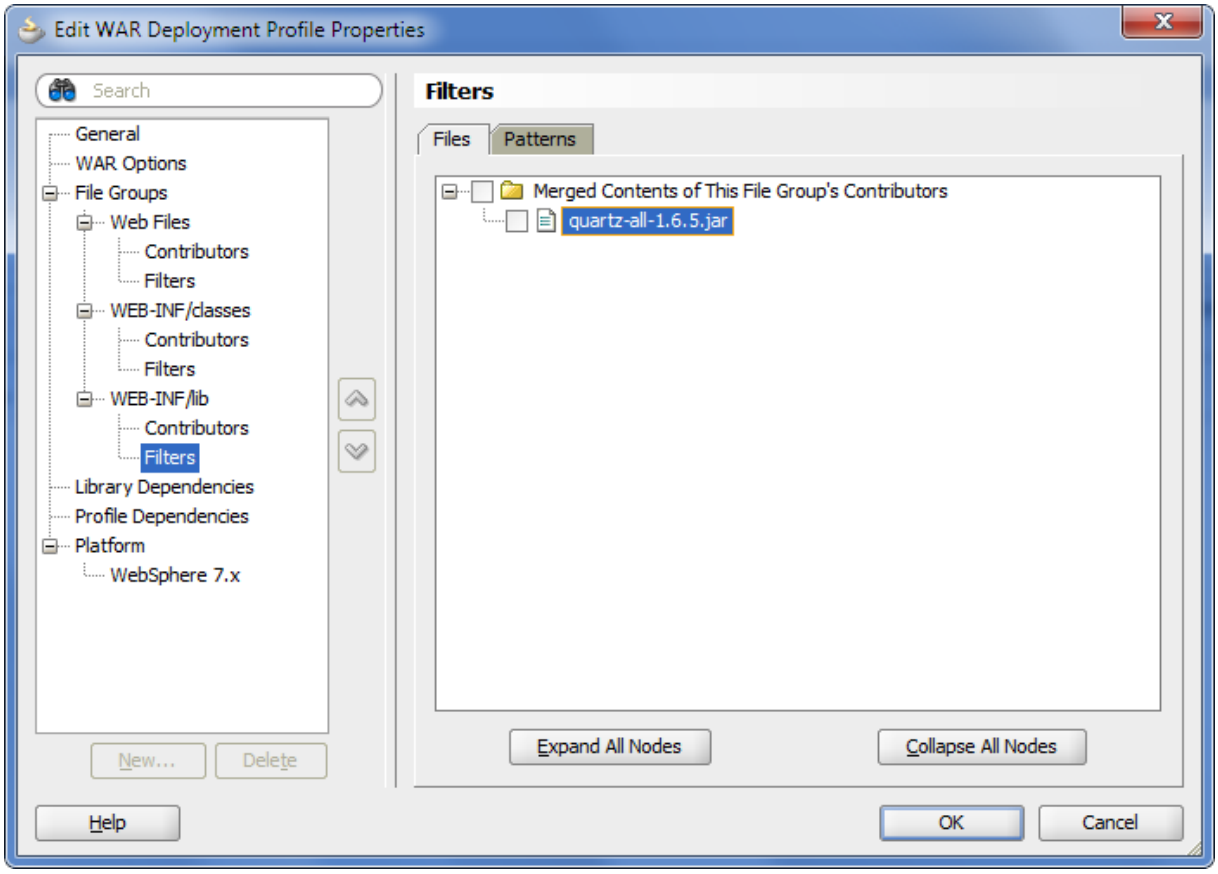

Figure 26

On the Filters subtub of WEB-INF/lib deselect “quartz-all-1.6.5.jar”, because it is already available on the WebLogic Server 10.3.1/2/3. Click *“OK”*.

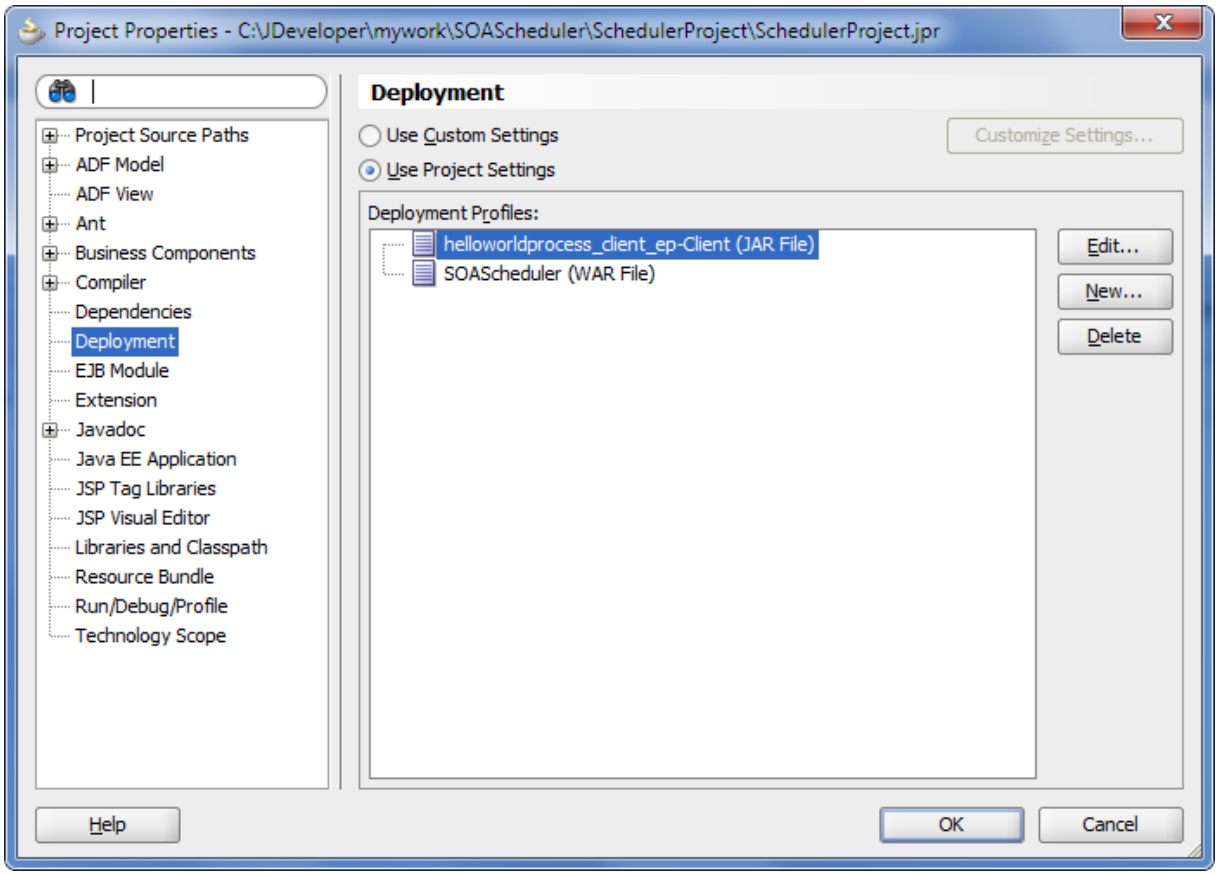

Figure 27

Click *“OK”*.

Deploy your application

Right click on your Project, select "*Deploy*" and click on "*SOAScheduler*".

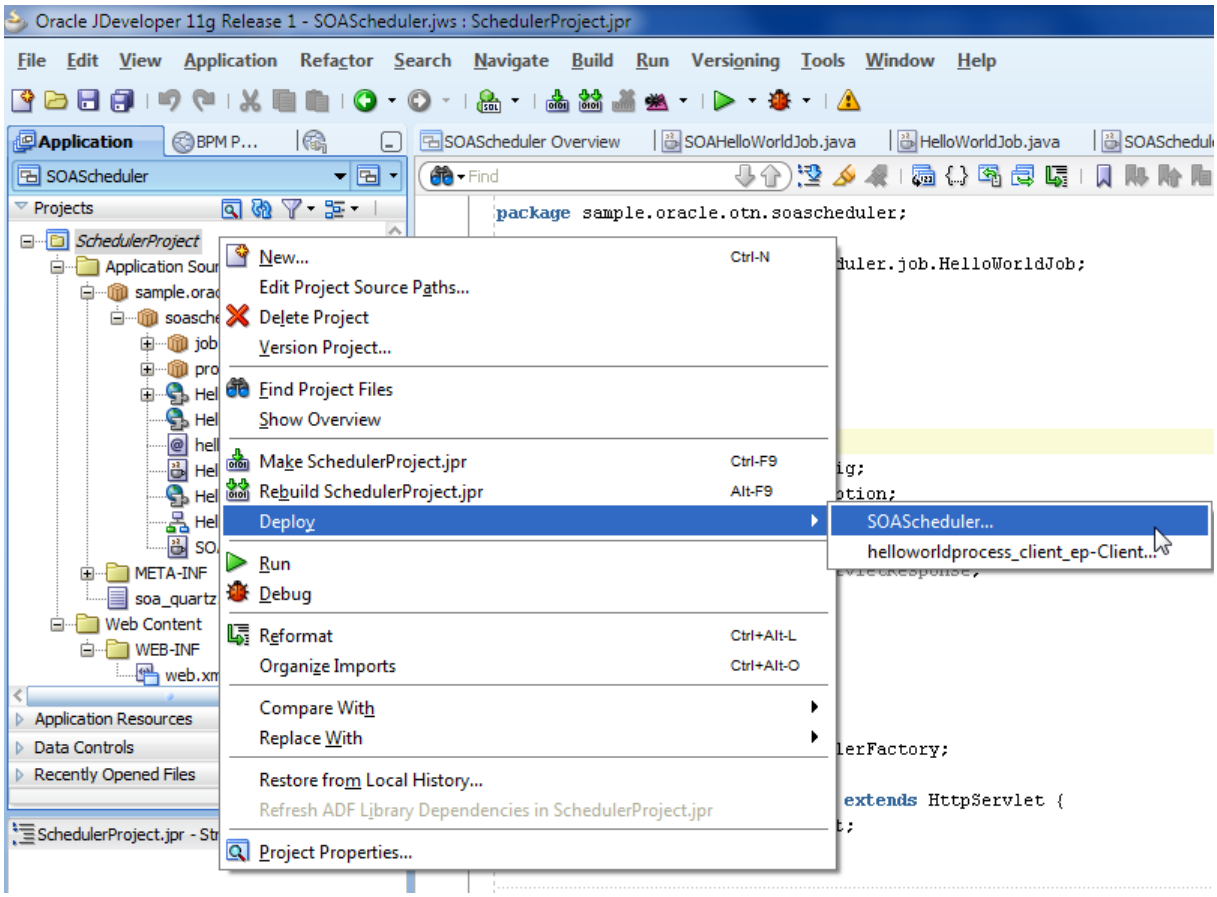

Figure 28

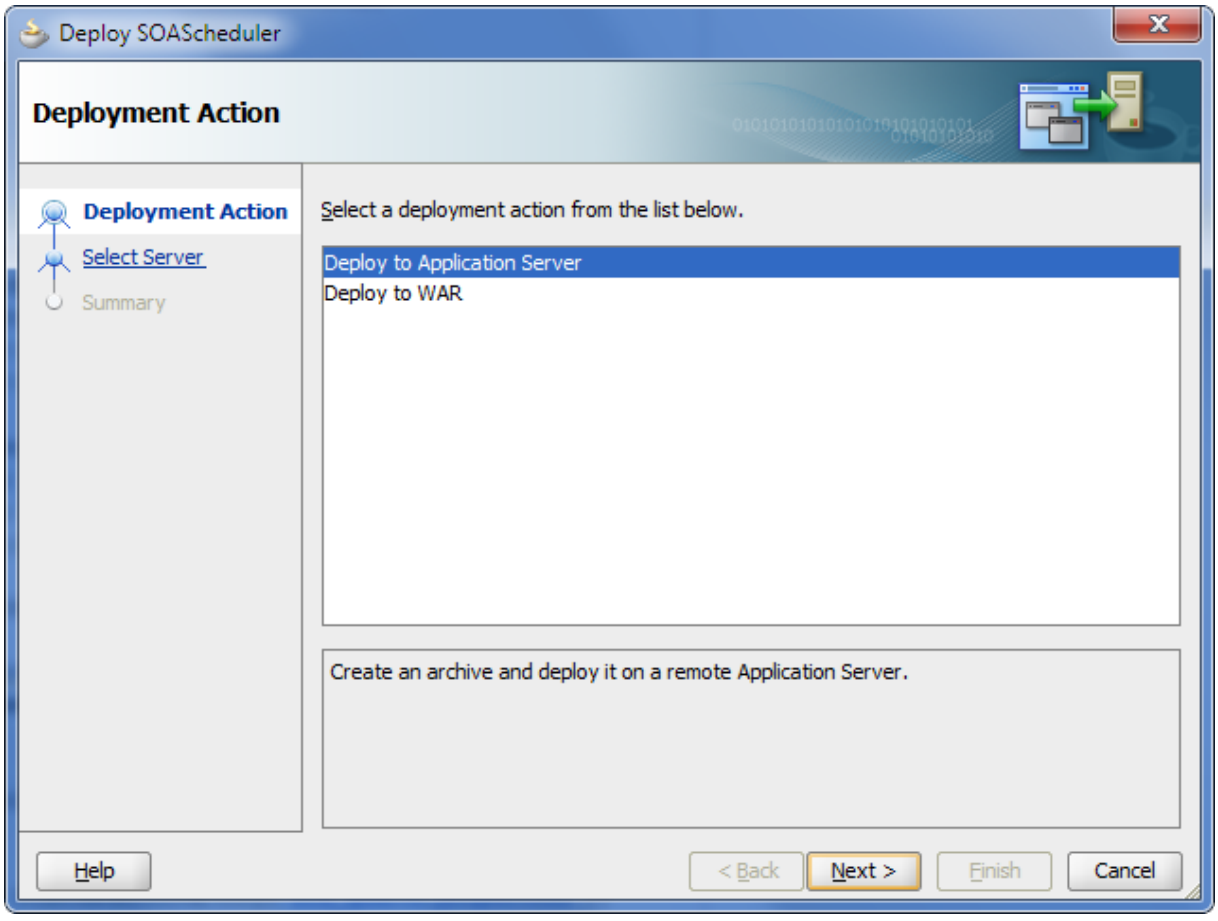

Figure 29

Select "*Deploy to Application Server*". Click "*Next*".

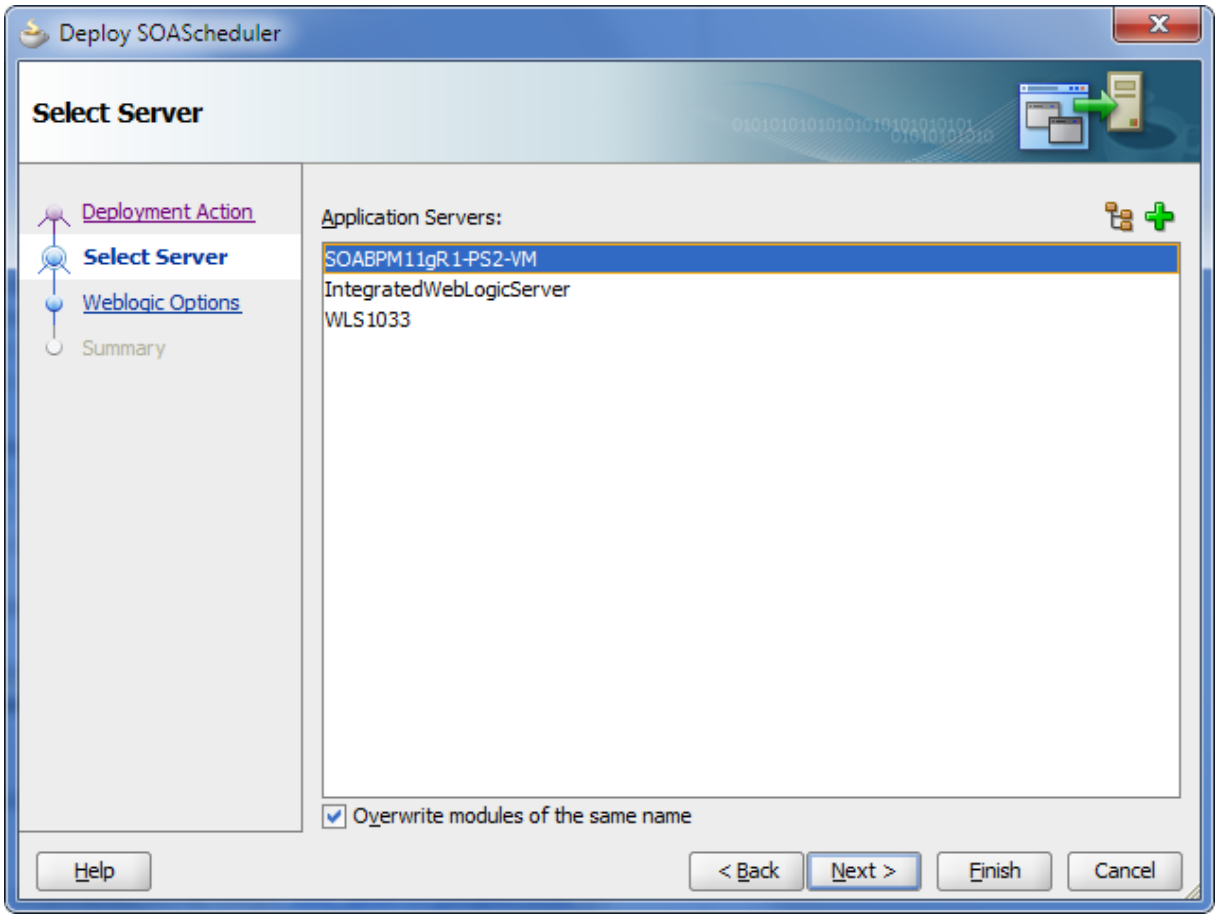

Figure 30

Select your Application Server, where the Oracle SOA Suite is installed. If you have not your application server already configured. Add a new connection with the green *"+"*. Click "*Next*".

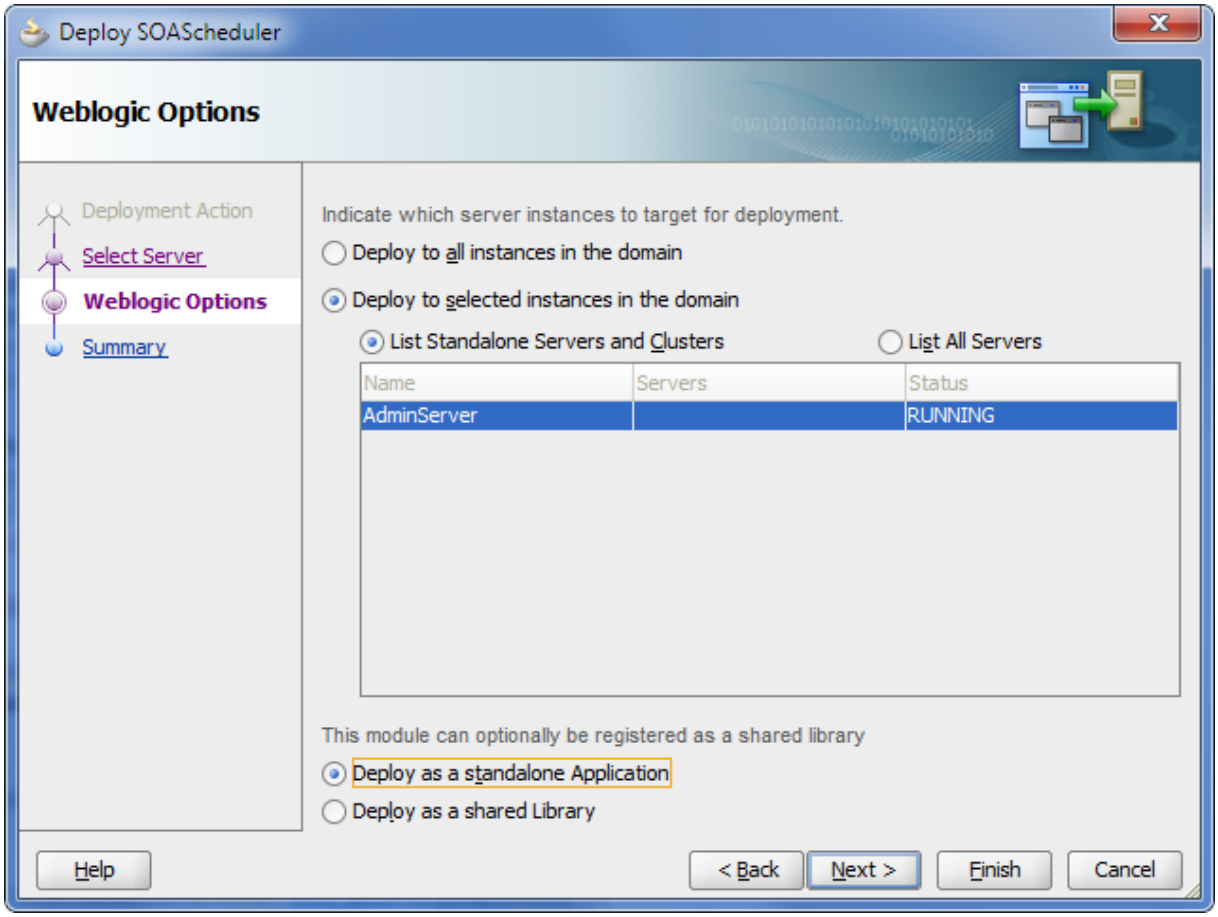

Figure 31

Select "*AdminServer*". Click "*Finish*".

If deployment successful, the message window shows following:

```
Deployment - Log
[01:21:14 PM] ----  Deployment started.  ----
[01:21:14 PM] Target platform is  (Weblogic 10.3).
[01:21:17 PM] Retrieving existing application information
[01:21:19 PM] Running dependency analysis...
[01:21:19 PM] Building...
[01:21:20 PM] Deploying profile...
[01:21:21 PM] Wrote Web Application Module to C:\JDeveloper\mywork\SOAScheduler\SchedulerProject\deploy\SOAScheduler.war
[01:21:21 PM] Deploying Application...
[01:21:25 PM] [Deployer:149192]Operation 'deploy' on application 'SOAScheduler' is in progress on 'AdminServer'
[01:21:26 PM] [Deployer:149194]Operation 'deploy' on application 'SOAScheduler' has succeeded on 'AdminServer'
[01:21:26 PM] Application Deployed Successfully.
[01:21:26 PM] The following URL context root(s) were defined and can be used as a starting point to test your application:
[01:21:26 PM] http://192.168.56.101:7001/SOAScheduler
[01:21:26 PM] Elapsed time for deployment:  12 seconds
[01:21:26 PM] ----  Deployment finished.  ----
```

Figure 32

Test your SOAScheduler

When the application SOAScheduler is deployed, the SOA composite bpel-101-helloworld is executed every 5 minutes. You will find this in the Oracle Enterprise Manager at http://server:port/em, e.g. http://localhost:7001/em

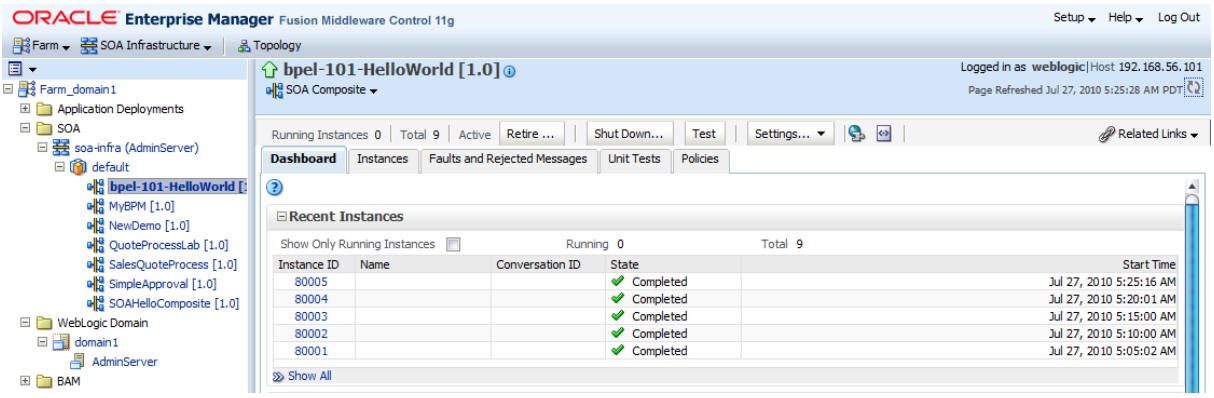

Figure 33

By click on one “Instance ID” you will get the details of the SOA compsite.

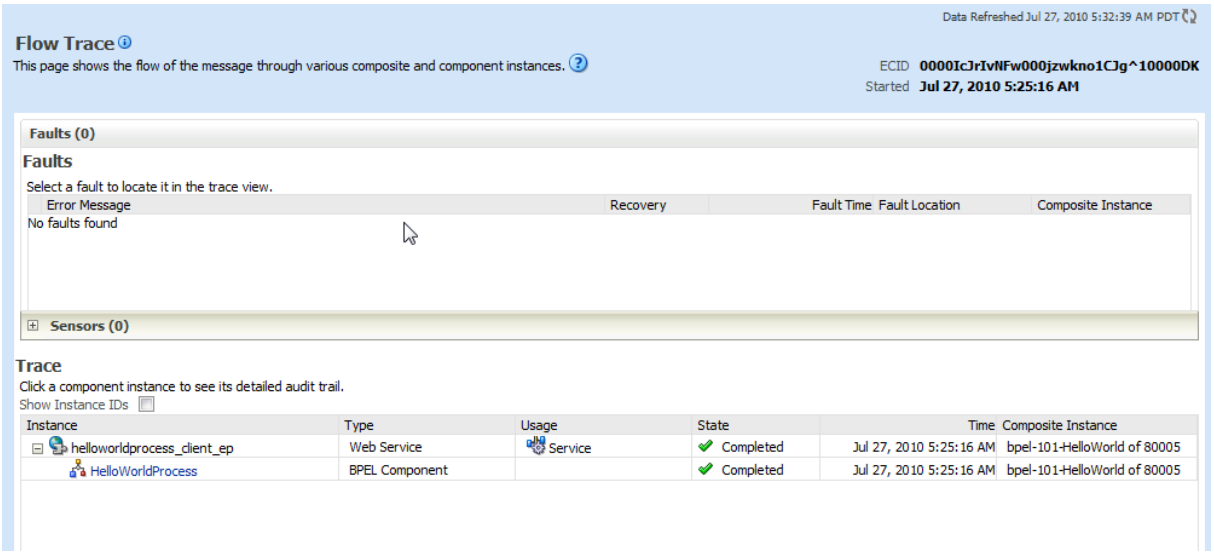

Figure 34

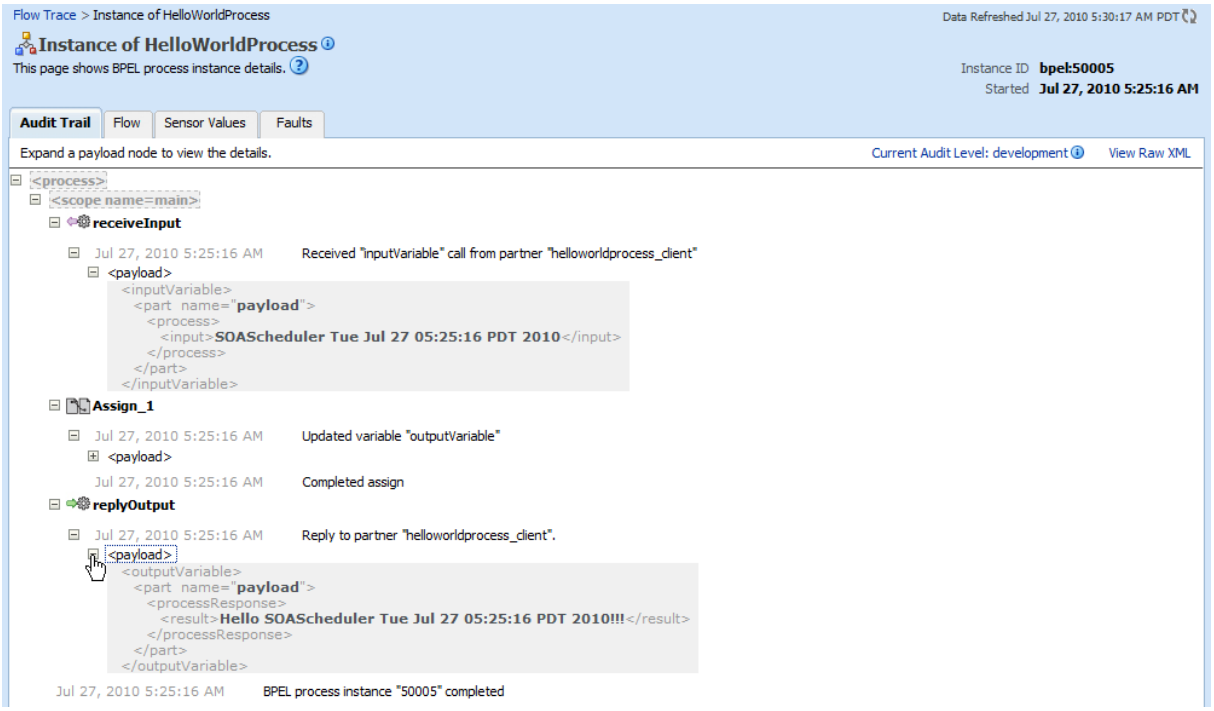

Figure 35

You will find also some output in your server log file.

```
sample.oracle.otn.soascheduler.SOASchedulerServlet stopped
sample.oracle.otn.soascheduler.SOASchedulerServlet started
sample.oracle.otn.soascheduler.SOASchedulerServlet Cron Expression for SOAJob:0 0,5,10,15,20,25,30,35,40,45,50,55 * * * ?
sample.oracle.otn.soascheduler.SOASchedulerServlet Scheduling Job SOAJob
sample.oracle.otn.soascheduler.SOASchedulerServlet Job SOAJob scheduled.
HelloWorldJob started
HelloWorld Response: Hello SOAScheduler@2010/11/08 16:25:00!!!
```

Advanced Configuration

Calling the SOAScheduler Servlet and Starting/Stopping the SOAScheduler

You can go to http://server:port/SOAScheduler/soaschedulerservlet and will see the next and the last run of the trigger, e.g. http://localhost:7001/SOAScheduler/soaschedulerservlet .
This servlet has a parameter with the name action. Valid values are “start”, “stop” and “single”.

start	Start the SOAScheduler, if it has been previously stopped
Stop	Stop the SOAScheduler
single	Start the job immediately. Works only if the scheduler is running.

e.g. http://localhost:7001/SOAScheduler/soaschedulerservlet ?action=stop

Business Calendar

In the SOASchedulerServlet is an example how to add holidays to the calendar. If you want to use this feature you have to uncomment this coding and to add the correct date for your local holidays.

```
/*
        // Add the holiday calendar to the schedule
        AnnualCalendar holidays = new AnnualCalendar();

        // fourth of July (July 4)
        Calendar fourthOfJuly = new GregorianCalendar(2011, 7, 4);
        holidays.setDayExcluded(fourthOfJuly, true);
        // halloween (Oct 31)
        Calendar halloween = new GregorianCalendar(2011, 9, 31);
        holidays.setDayExcluded(halloween, true);
        // christmas (Dec 25)
        Calendar christmas = new GregorianCalendar(2011, 11, 25);
        holidays.setDayExcluded(christmas, true);

        // tell the schedule about our holiday calendar
        sched.addCalendar("holidays", holidays, false, false);
*/
```

SOAScheduler in a Cluster

You can use this SOAScheuler in a cluster. Quartz – the software used by the SOAScheduler - ships with well-proven clustering capabilities that offer scaling and high availability features. You can read about these features in the Quartz Configuration Reference. You will need to setup a JDBC-JobStore. See http://www.quartz-scheduler.org/docs/configuration/ConfigJDBCJobStoreClustering.html

Examples for cron expressions

A cron expression is a string comprised of 6 or 7 fields separated by white space.

Field Name	Mandatory	Allowed Values	Allowed Special Characters
Seconds	YES	0-59	, - * /
Minutes	YES	0-59	, - * /
Hours	YES	0-23	, - * / L W
Day of month	YES	1-31	, - * ? / L W
Month	YES	1-12 or JAN-DEC	, - * /
Day of week	YES	1-7 or SUN-SAT	, - * ? / L #
Year	NO	empty, 1970-2099	, - * /

Examples for cron:

Expression	Meaning
0 0 12 * * ?	Fire at 12pm (noon) every day
0 15 10 ? * *	Fire at 10:15am every day
0 15 10 * * ?	Fire at 10:15am every day
0 15 10 * * ? *	Fire at 10:15am every day
0 15 10 * * ? 2005	Fire at 10:15am every day during the year 2005
0 * 14 * * ?	Fire every minute starting at 2pm and ending at 2:59pm, every day
0 0/5 14 * * ?	Fire every 5 minutes starting at 2pm and ending at 2:55pm, every day
0 0/5 14,18 * * ?	Fire every 5 minutes starting at 2pm and ending at 2:55pm, AND fire every 5 minutes starting at 6pm and ending at 6:55pm, every day
0 0-5 14 * * ?	Fire every minute starting at 2pm and ending at 2:05pm, every day
0 10,44 14 ? 3 WED	Fire at 2:10pm and at 2:44pm every Wednesday in the month of March
0 15 10 ? * MON-FRI	Fire at 10:15am every Monday, Tuesday, Wednesday, Thursday and Friday
0 15 10 15 * ?	Fire at 10:15am on the 15th day of every month
0 15 10 L * ?	Fire at 10:15am on the last day of every month
0 15 10 ? * 6L	Fire at 10:15am on the last Friday of every month
0 15 10 ? * 6L	Fire at 10:15am on the last Friday of every month
0 15 10 ? * 6L 2002-2005	Fire at 10:15am on every last Friday of every month during the years 2002 to 2005
0 15 10 ? * 6#3	Fire at 10:15am on the third Friday of every month
0 0 12 1/5 * ?	Fire at 12pm (noon) every 5 days every month, starting on the first day of the month
0 11 11 11 11 ?	Fire every November 11th at 11:11am

See http://www.quartz-scheduler.org/docs/tutorials/crontrigger.html

Tested with Oracle SOA Suite 11g(11.1.1.3).