CSV Download from a JSP Application

Applies to:

SAP Enterprise Portal 7.0. For more information, visit the Portal and Collaboration homepage

Summary

CSV download is a commonly required functionality in Web UI projects. Here I have described a simple approach for downloading data to a CSV file from a JSP application.

Author: Sudeep Sura

Company: Larsen and Toubro Infotech Limited

Created on: March 14, 2011

Author Bio

Sudeep Sura is working with Larsen and Toubro Infotech Limited as a SAP NetWeaver Lead. The author has more than four years of experience in diverse aspects of SAP Enterprise Portal, Web Dynpro development, Java/J2EE, and Adobe Document Services.

Table of Contents

Getting Started

CSV is widely used file type for transferring data between systems and also for mass upload of data.

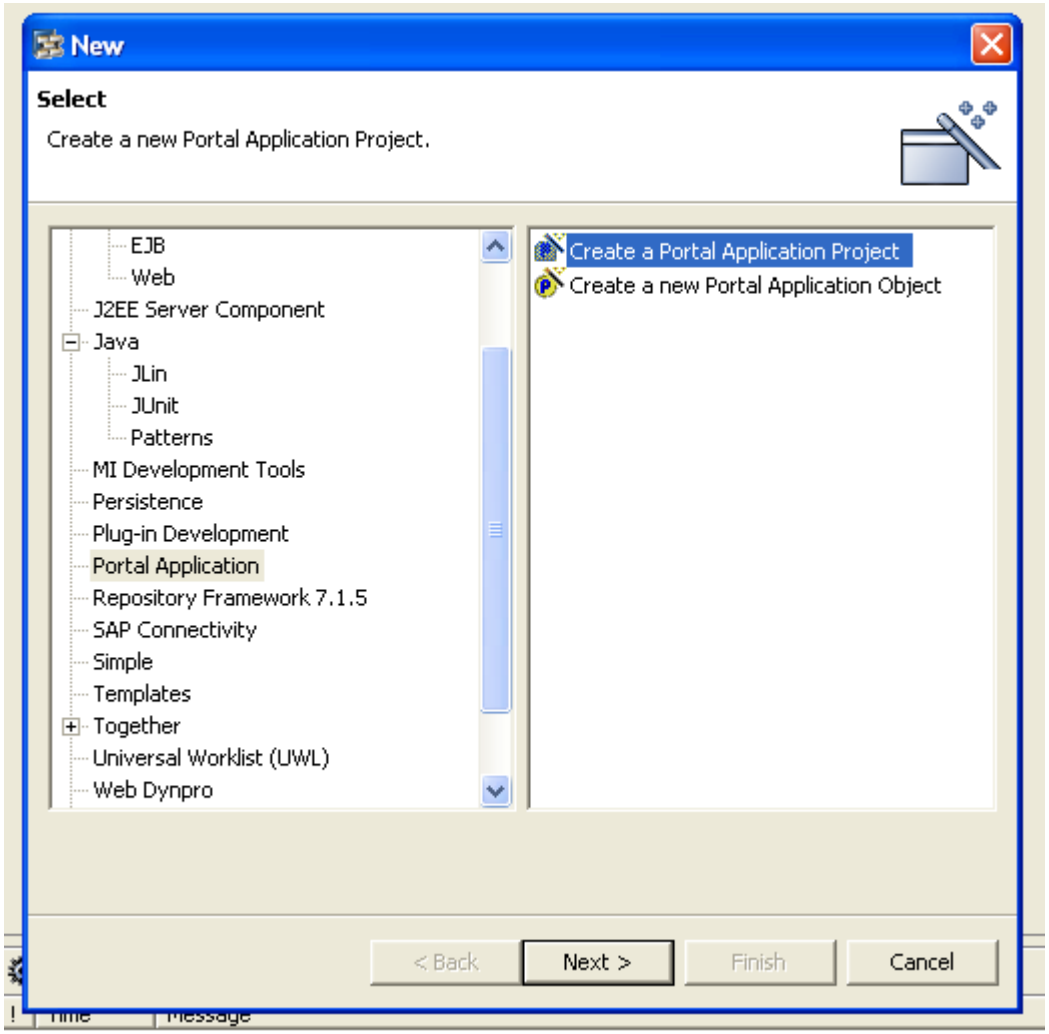
Often, in projects a standard requirement is to provide CSV download. This article provides an easy implementation of CSV download for portal applications.

Scenario

A JSP application provides a hyperlink to download data to CSV file. The file is generated in another JSP file and downloaded using HttpServletResponse.

Creating a Project

Click File > New > Other > Portal Application > Create a Portal Application Project



Click Next.

Enter project name as ‘DataDownloadProj’, select a root folder and click finish.

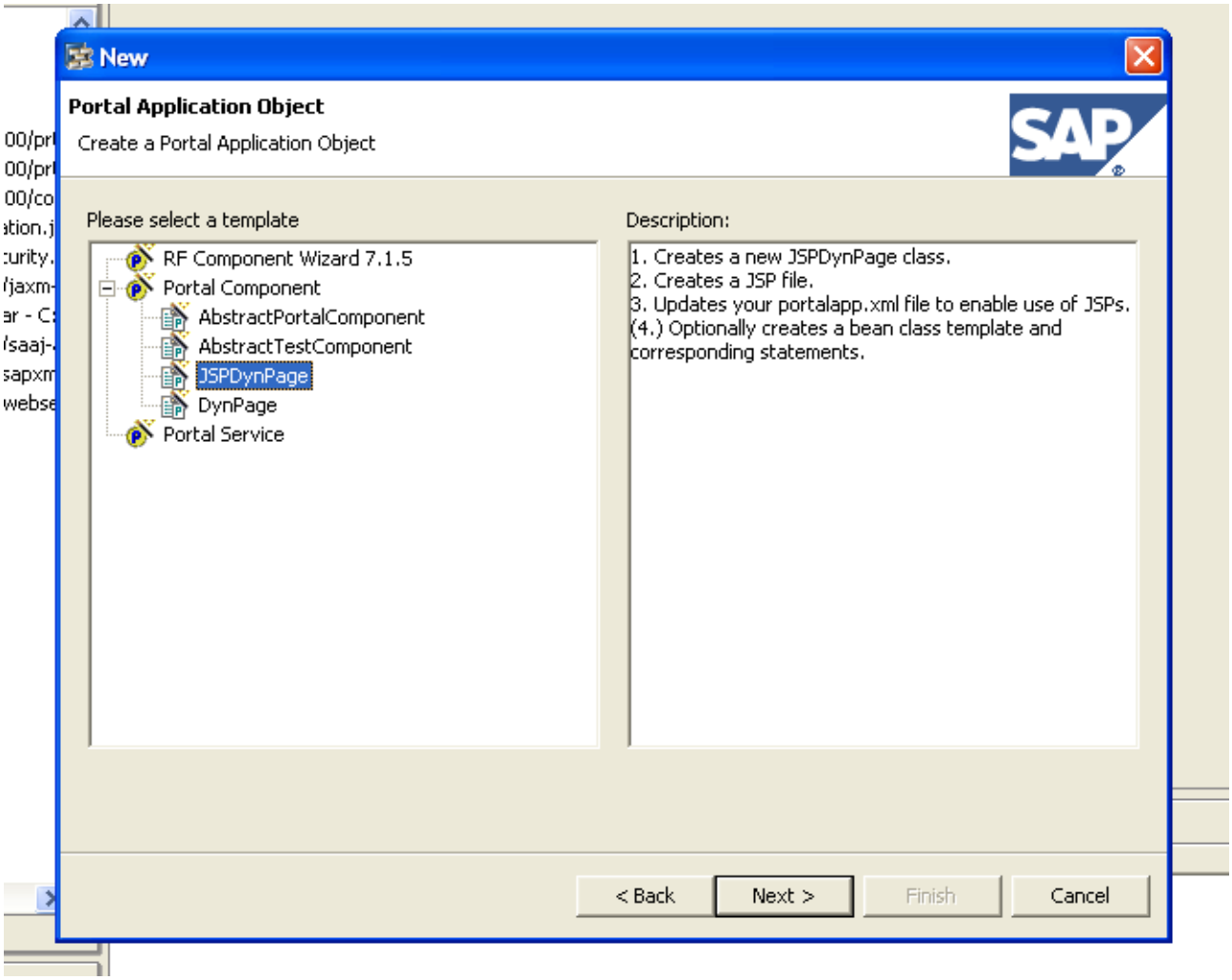
Go to Enterprise Portal Perspective.

Creating a Portal Application Object - JSPDynPage

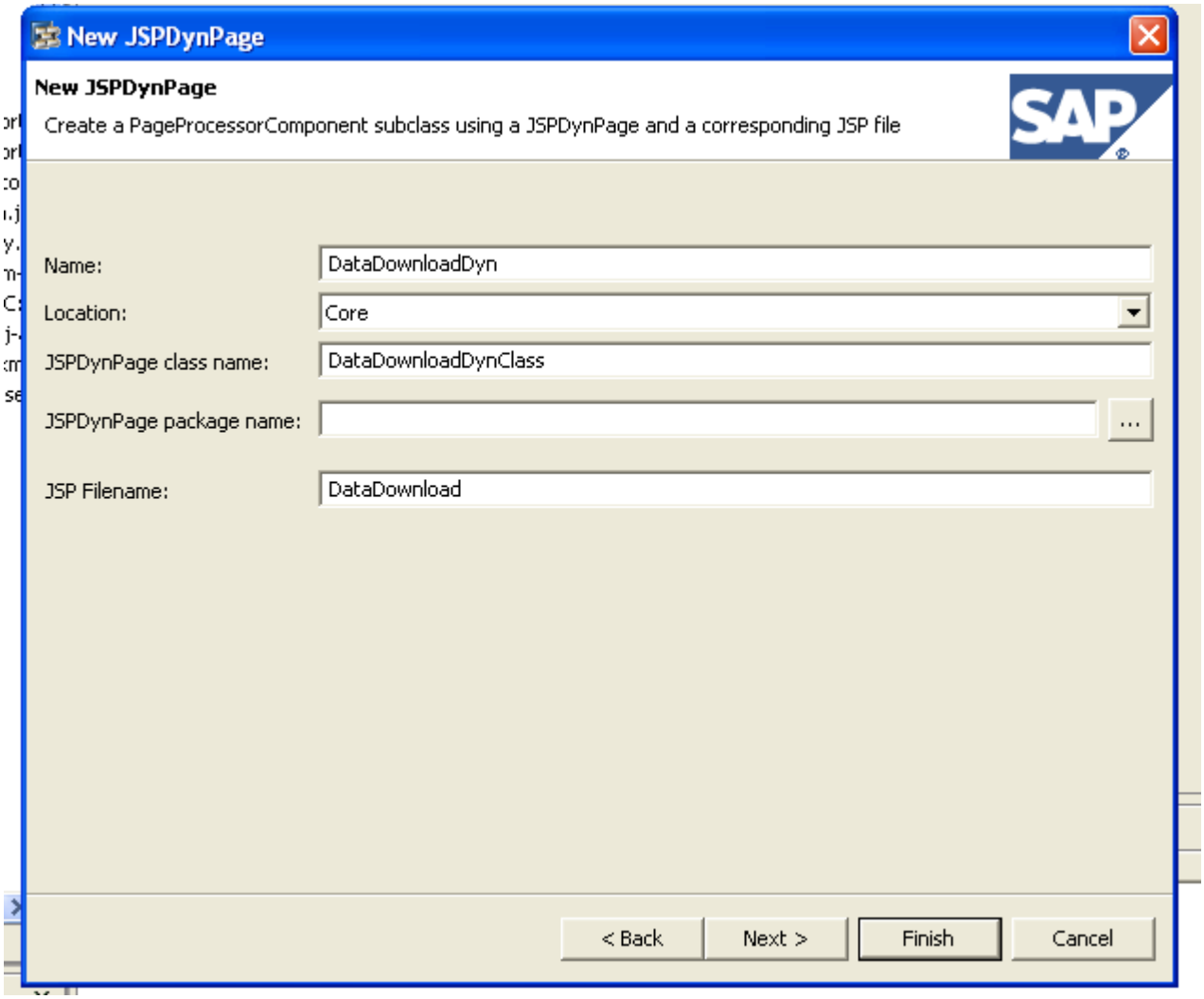
Next create a Portal Application Object

Click File > New > Other > Portal Application > Create a Portal Application Project

Click Next and select JSPDynPage



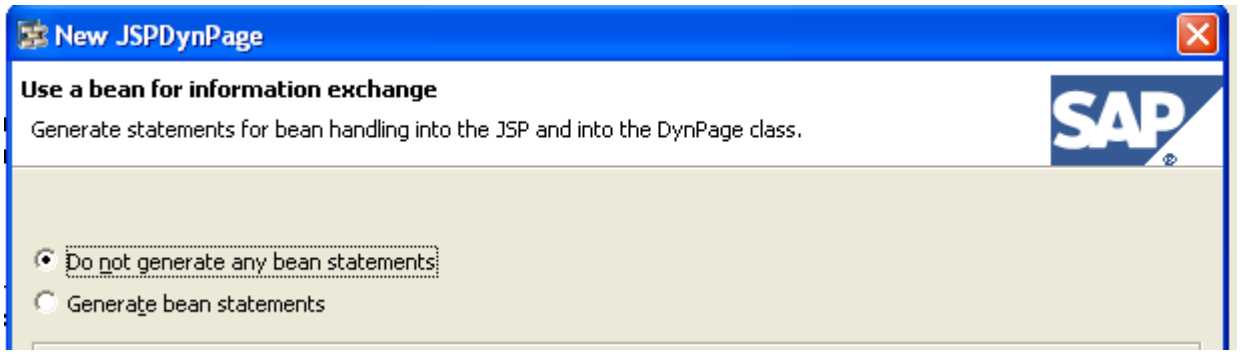
Click Next and Enter the details as shown in the screenshot below.



Click Next.

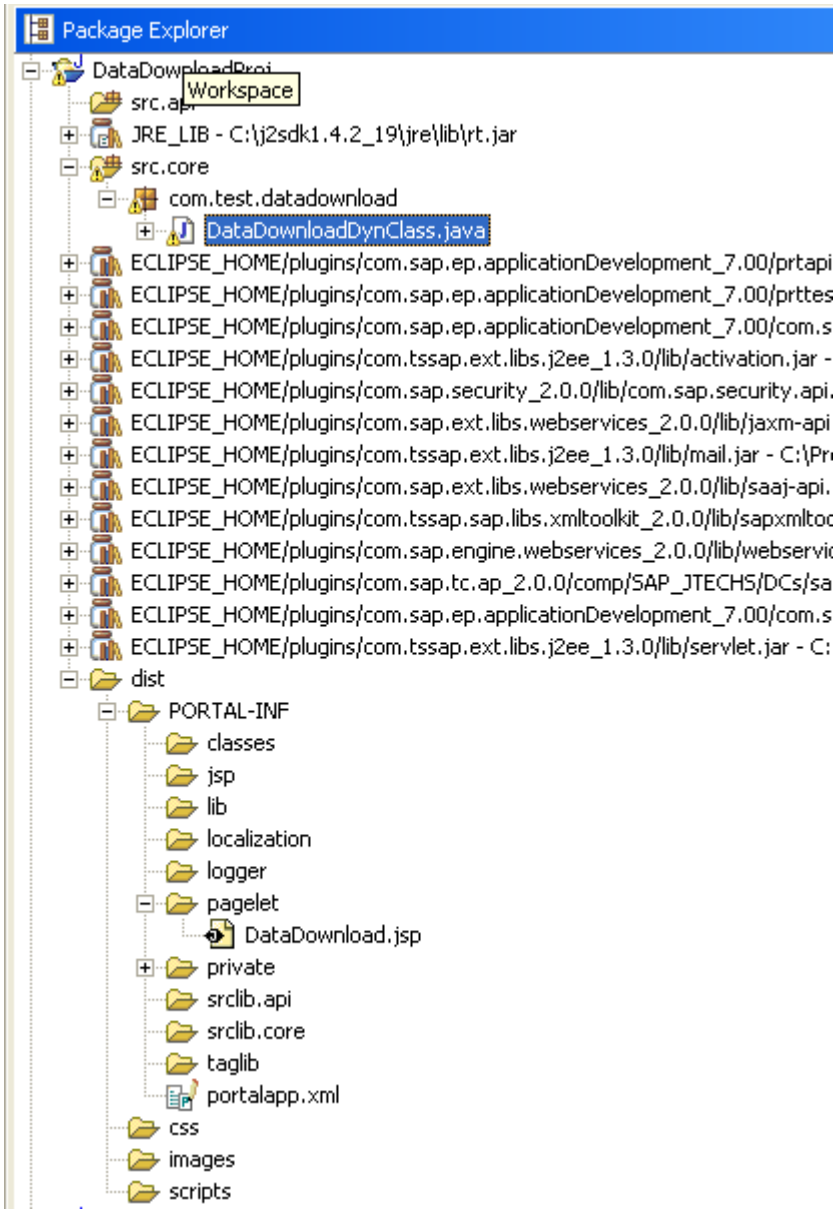
We will not be requiring Bean class to hold data for our example.

So select the 'Do not generate any bean statements' option and click Next.



Now we have created a portal application project with a JSPDynPage and a java file.

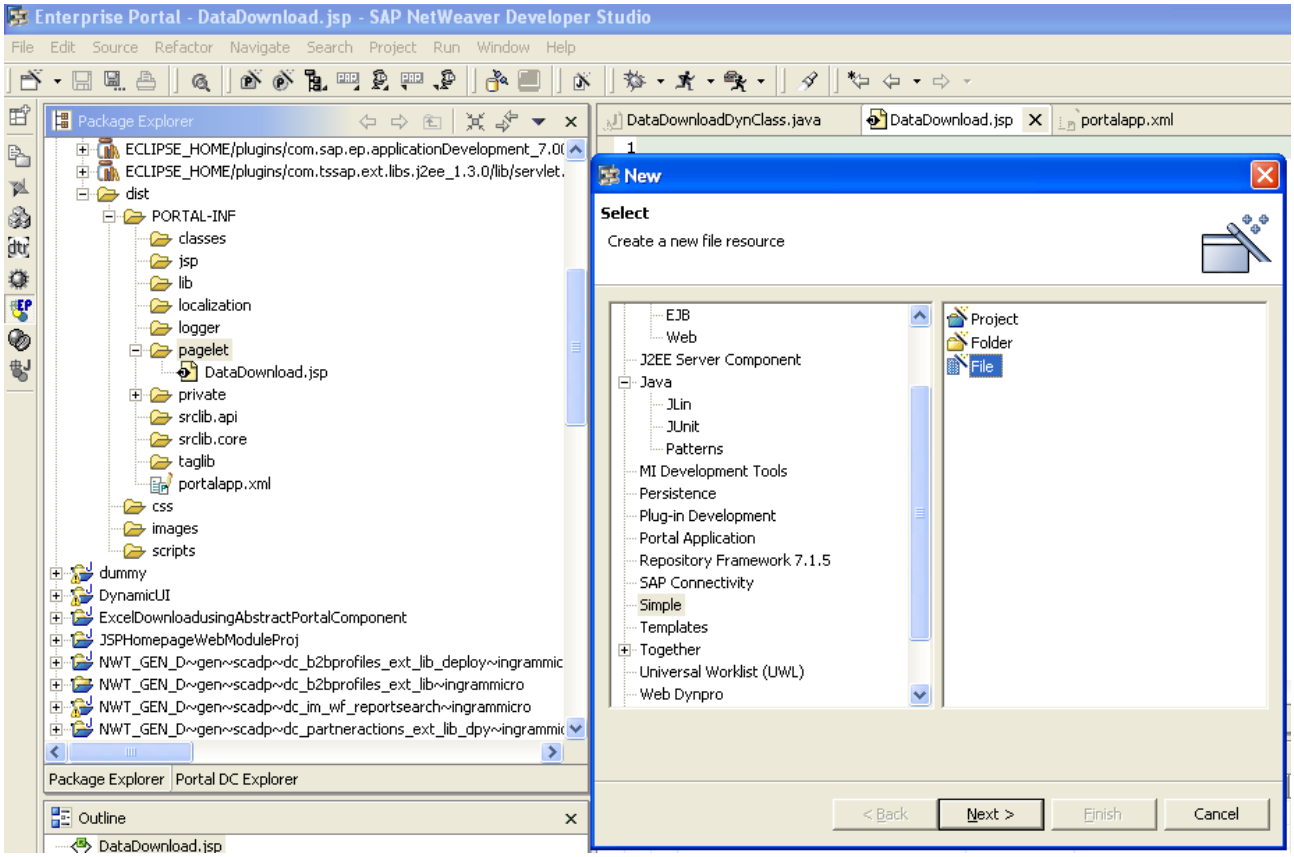
This is how the project structure will look like.



Creating a JSP File

Now we will create a jsp file from which data is to be downloaded to csv.

Right click on the pagelet folder and Select New > Other > Simple > File



Click Next

Enter File name as CSVgenerator.jsp and Click Finish

Open the newly created file and go to Source tab.

Here we use the java.io.FileWriter class to enter data into a csv file.

Next create a byte array to hold the data

Using the HttpServletResponse, the byte array is provided t ouser for download.

```
<%@ page import = "java.io.FileWriter" %>
<%@ page import = "java.io.BufferedWriter" %>
<%@ page import="java.io.IOException"%>
<%@ page import="java.io.FileInputStream"%>
<%@ page import="java.io.InputStream"%>
<%
      String lengthstr = "";
      try
      {
            String filename = "test.csv";
FileWriter writer = new FileWriter(filename,false);
            writer.flush();
            writer.close();
            writer = new FileWriter(filename,true);
         writer.write("Employee ID");
         writer.write(',');
            writer.write("Name");
            writer.write(',');
            writer.write("Age");
         writer.write(',');
         writer.write("Specialization");
            writer.write("\r\n");
            writer.write("873214");
         writer.write(',');
         writer.write("Keith Nickelberg");
         writer.write(',');
         writer.write("25");
            writer.write(',');
            writer.write("SAP FI");
            writer.write("\r\n");
         writer.write("875814");
         writer.write(',');
            writer.write("Adam Perry");
            writer.write(',');
            writer.write("29");
         writer.write(',');
         writer.write("SAP OTC");
            writer.write("\r\n");
         writer.write("875853");
         writer.write(',');
         writer.write("Tina Ray");
         writer.write(',');
         writer.write("27");
         writer.write(',');
         writer.write("SAP P2P");
         writer.write("\r\n");
         writer.write("870822");
         writer.write(',');
         writer.write("Aster McMillan");
         writer.write(',');
         writer.write("31");
         writer.write(',');
         writer.write("SAP MM");
         writer.write("\r\n");
            writer.flush();
            writer.close();
```

```
            InputStream is = new FileInputStream(filename);
            File file = new File(filename);

            // Get the size of the file
            long length = file.length();
            // You cannot create an array using a long type.
            // It needs to be an int type.
            // Before converting to an int type, check
            // to ensure that file is not larger than Integer.MAX_VALUE.
            if (length > Integer.MAX_VALUE) {
                  // File is too large
                  System.out.println("File size is too large");
            }
            // Create the byte array to hold the data
            byte[] bytes = new byte[(int)length];
            // Read in the bytes
            int offset = 0;
            int numRead = 0;
            while (offset < bytes.length
                     && (numRead=is.read(bytes, offset, bytes.length-offset)) >= 0)
{
                  offset += numRead;
            }

// Ensure all the bytes have been read in
            if (offset < bytes.length) {
                  throw new IOException("Could not completely read file");
            }
            // Close the input stream and return bytes
            is.close();

            HttpServletResponse res = componentRequest.getServletResponse(true);
            res.setContentType("application/text");
            res.addHeader("Content-Disposition","attachment; filename="+filename );
            res.getOutputStream().write(bytes);
      }
      catch(Exception e)
      {
         System.out.println(e.getCause());
      }
%>
```

Modify the Default JSP

Open the DataDownload.jsp

You will find auto generated code in the jsp file.

We will modify the code to provide a hyperlink and a javascript function.

On click of the hyperlink, this function is triggered.

The function calls DataDownloadDynClass.java and passes a parameter to it:

param=csv

```
<hbj:content id="myContext" >
  <hbj:page title="PageTitle">
   <hbj:form id="myFormId" >

<html>
<head>

<script type="text/javascript">
function toCSV() {
         var wo_url = "DataDownloadProj.DataDownloadDynClass?param=csv";
         window.open(wo_url,"wblank","toolbar=no, location=no, directories=no,
status=no, menubar=no, scrollbars=yes, resizable=yes, copyhistory=yes, width=680,
height=520 , left=20 , top=40", "replace = true");
}
</script>

</head>

<body>
<a href="javascript:toCSV()">Please click on the link here to download data to
CSV</a>
</body>

</html>

   </hbj:form>
  </hbj:page>
</hbj:content>
```

Modify the java file

Open the DataDownloadDynClass.java file

In the doProcessBeforeOutput() method, use the IPortalComponentRequest to get the url parameter.

If the parameter is csv, redirect to CSVgenerator.jsp else open the default jsp - DataDownload.jsp

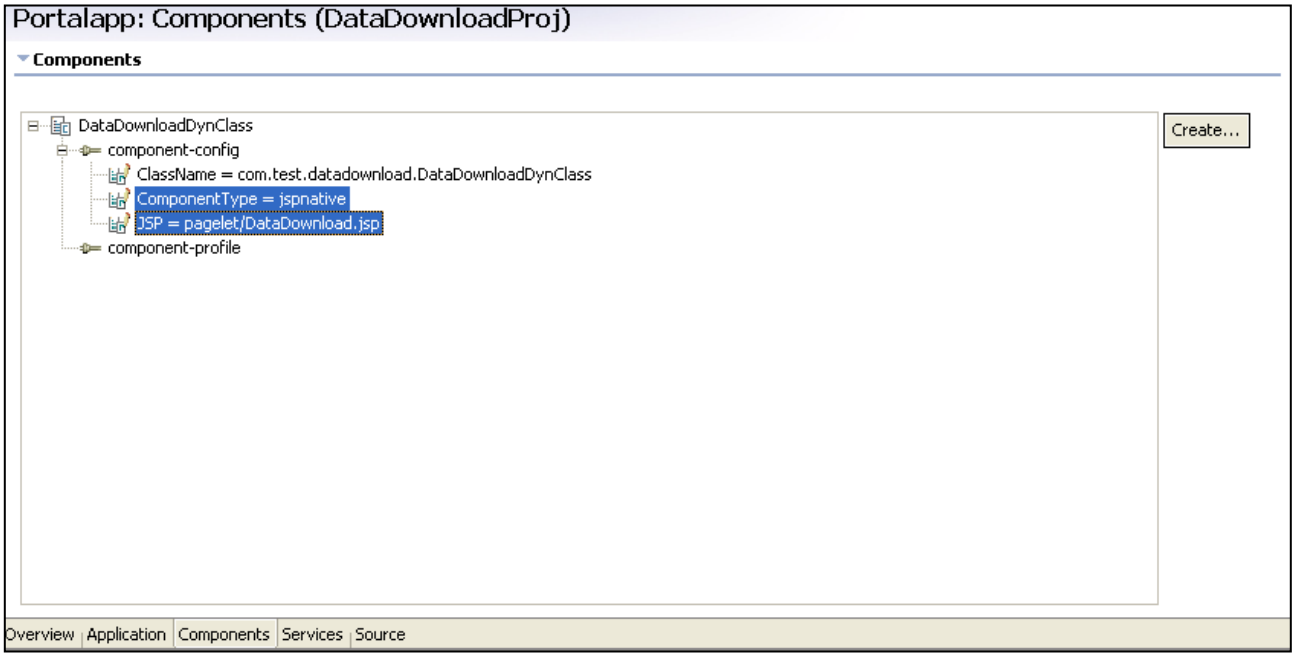
```
        IPortalComponentRequest request =
    (IPortalComponentRequest)this.getRequest();
        String param = request.getParameter("param");

        if(param != null && param.equalsIgnoreCase("csv")) {
              this.setJspName("CSVgenerator.jsp");
        } else {
            this.setJspName("DataDownload.jsp");
```

Changing the portalapp.xml Properties

Next open the portalapp.xml.

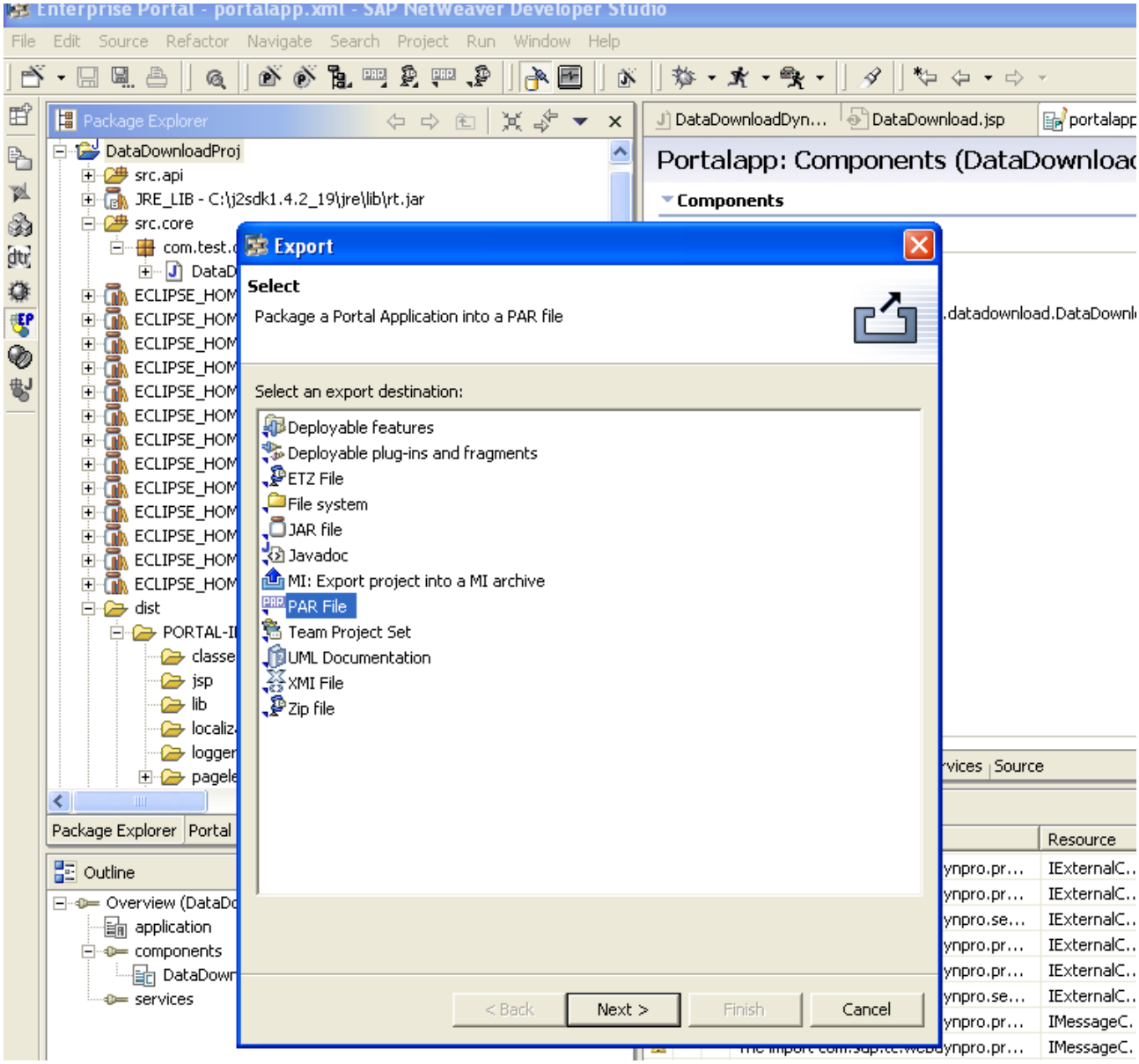
Go to Components tab.

Delete the ComponentType and JSP references from component-config.



Export the par file

Export the par file to the server by right-clicking on the project name and selecting Export > PAR File

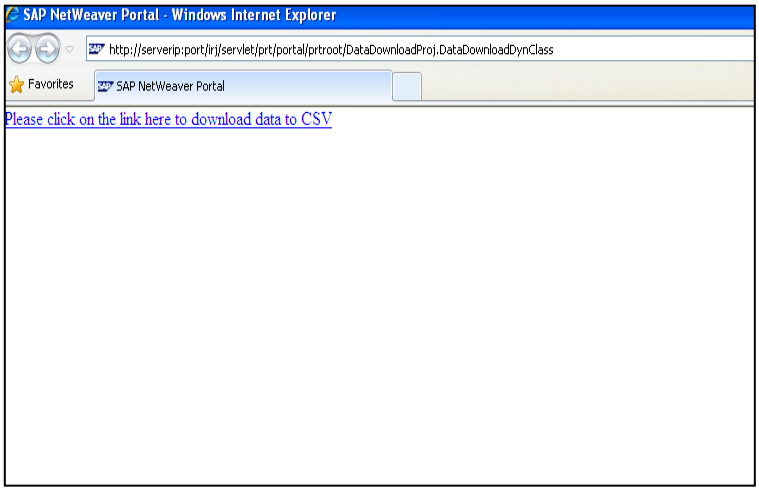


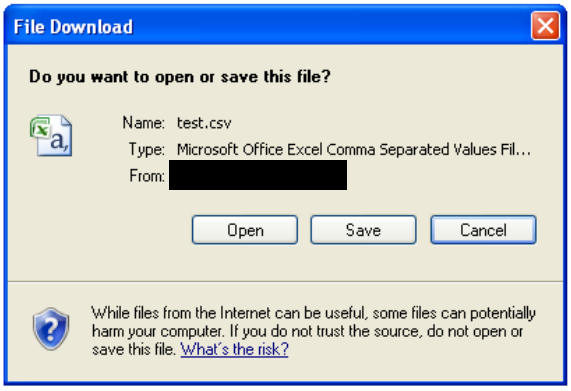
Run the Application

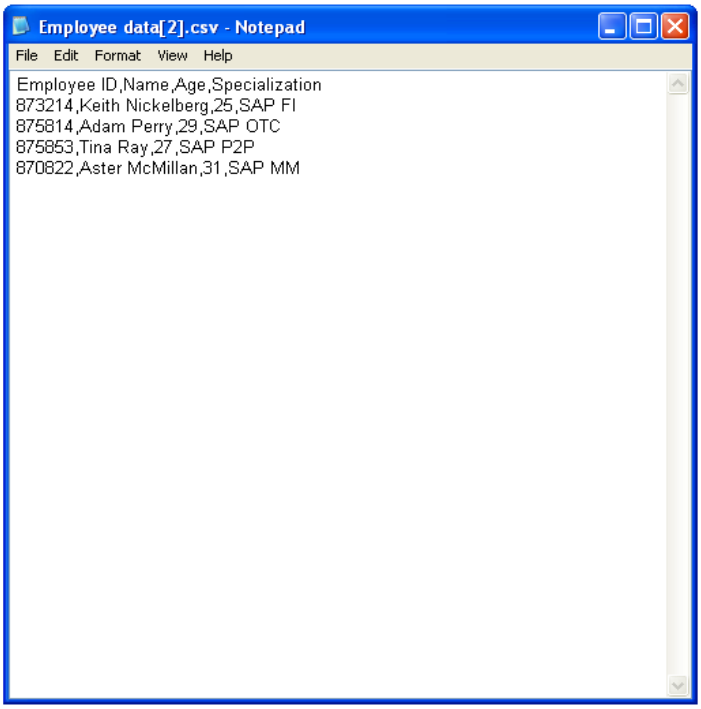
Run the application by clicking ‘Run’ on portalapp.xml

Output

Below is the output:







Related Content

Spreadsheet integration - which approach for which SAP technology

www.help.sap.com

www.sdn.sap.com

For more information, visit the Portal and Collaboration homepage

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.