External Library Deployment and Usage in SAP NetWeaver Developer Studio (NWDS)

Applies to:

SAP NetWeaver, SAP NetWeaver Developer Studio 7.0.16, SAP Java, External Libraries For more information, visit the Java homepage.

Summary

This article will guide you through step by step process of using an external library in existing web application. An external library Development Component will be first created and then deployed to application server.

Author: Saurabh Agarwal

Company: Steria, India

Created on: 27th February 2010

Author Bio

Saurabh Agarwal is working as a Consulting Engineer with Steria India Ltd. He has been involved in Java based portal development (Netweaver Developer Studio), and also has knowledge of SAP Enterprise portal (Administration), Java WebDynpros. He can also contribute and modify applications in ABAP and ABAP WebDynpros.

Table of Contents

Introduction

Often there are scenarios where inside your web application you need to reference some JAR file. Like, you need to give download functionality in xlf format for your java report. For this purpose you wish to use Apache POI Jar files.

This document will guide you through a step by step process for using an external library inside your existing web project. First, you will learn how to deploy this JAR file in application server and then how to reference it from within your existing web development component.

I assume that you already have your web development component and its corresponding deployment component (Enterprise Application Development Component) already created in your workspace.

I have named them:

- WebApplicationDC_WA (Web Development Component)
- WebApplicationDC_EA (Enterprise Application Development Component)

In below step by step process we will create an “External Library” Development Component which will contain our JAR file. We will name it as

- ExternalLibraryContainerDC

Once we have a JAR inside above container (ExternalLibraryContainerDC), we need to deploy it on application server. For this purpose we require a deployable unit for this library container DC of type

“J2EE server component -> Library”. We will name it as

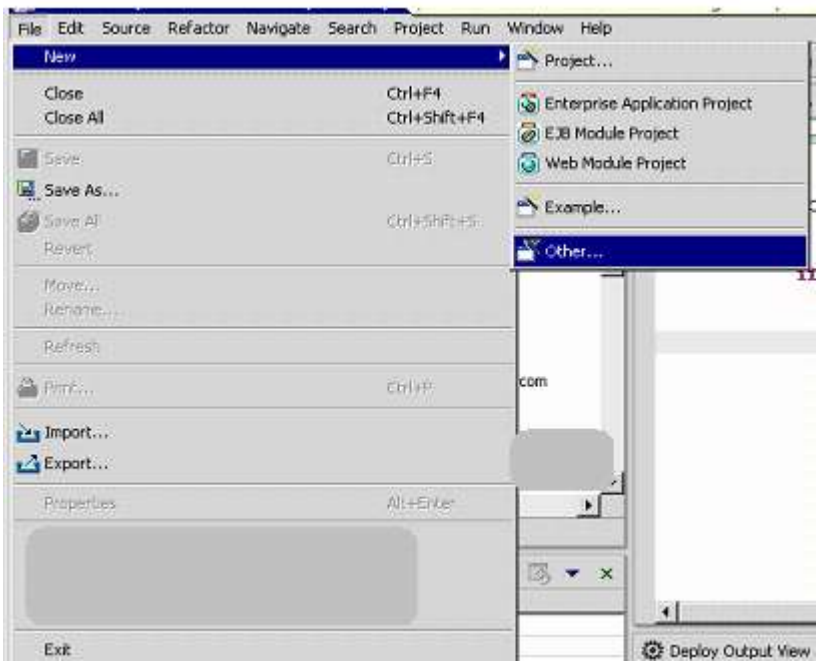
- ExternalLibrary_DeployableUnit

Add JAR to “External Library” type Development Component

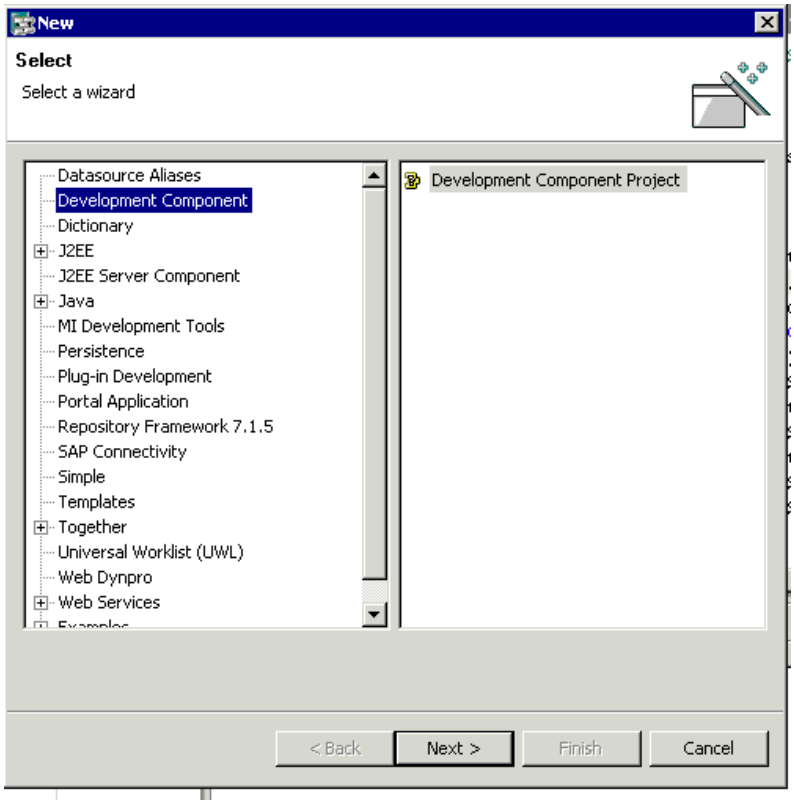
This component will act as actual container for our JAR file.

Step – Create Development Component “ExternalLibraryContainerDC”

Click on File -> New -> Other

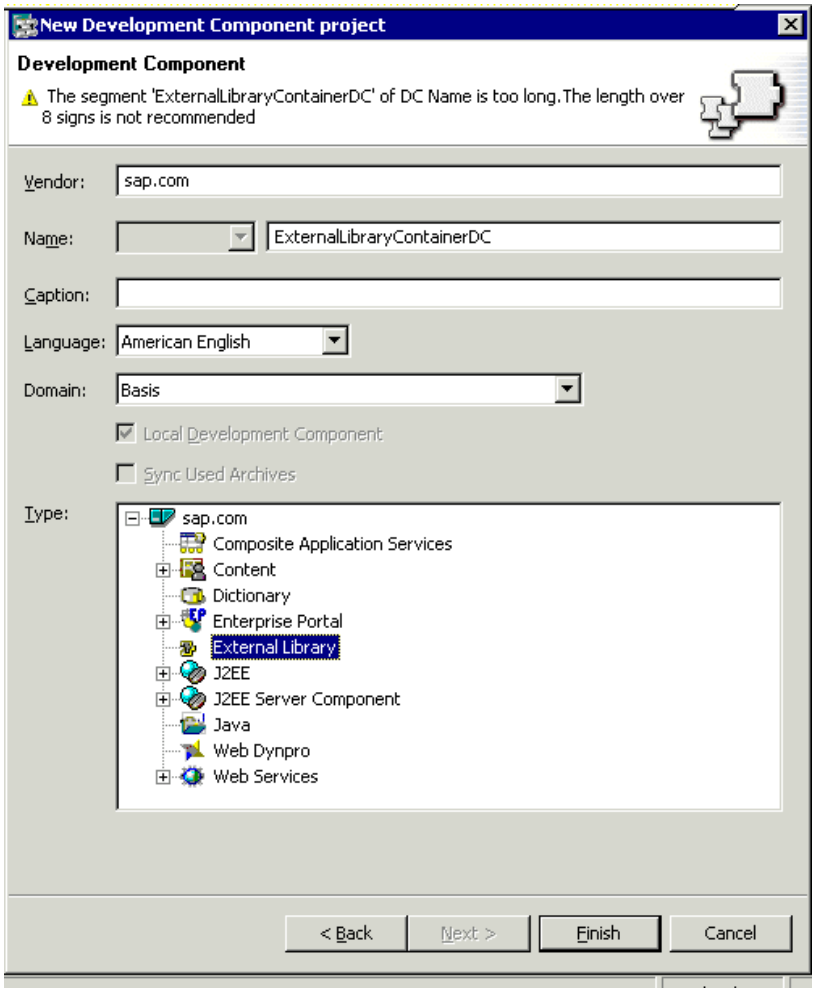


Select Development Component and click on Next.



Select Local Development -> My Components and click Next

Specify name of Vendor, Component name “ExternalLibraryContainerDC”. Select “External Library” as type of Development Component and click Finish



Step – Add JAR library file to “ExternalLibraryContainerDC”

Switch to “Package Explorer” in J2EE perspective. “Drag and Paste” your library in “Library folder” of ExternalLibraryContainerDC as shown below.

Note: Please note we are using “Apache POI” as an example. You can use any other JAR file as well

Expose library to be used by other Development Components.

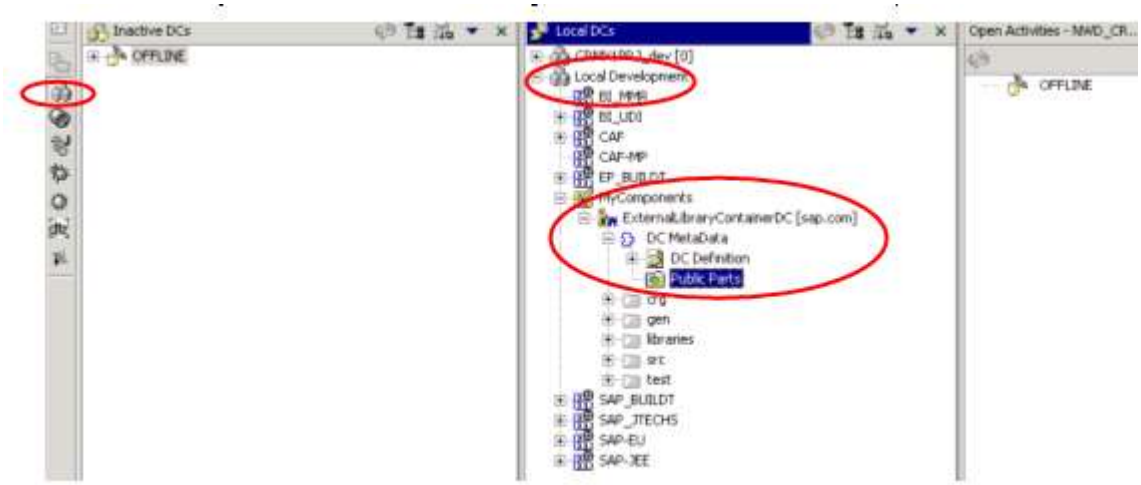
Till now we have successfully added our library to “ExternalLibraryContainerDC”. Now, we need to add this library to some place from where other Development Components (e.g. our web component “WebApplicationDC_WA”) can access it or reference it.

The place which we are referring to here is “Public Part”. As the name suggest it is a part of Development Component which can be accessed from other Development Components.

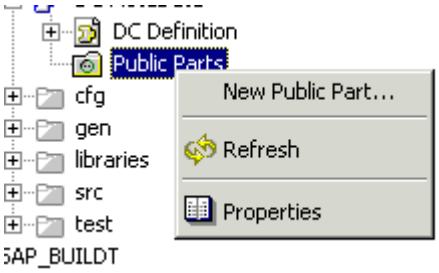
So now we will add “public parts” to “ExternalLibraryContainerDC”.

Step - Add “public parts” to “ExternalLibraryContainerDC”.

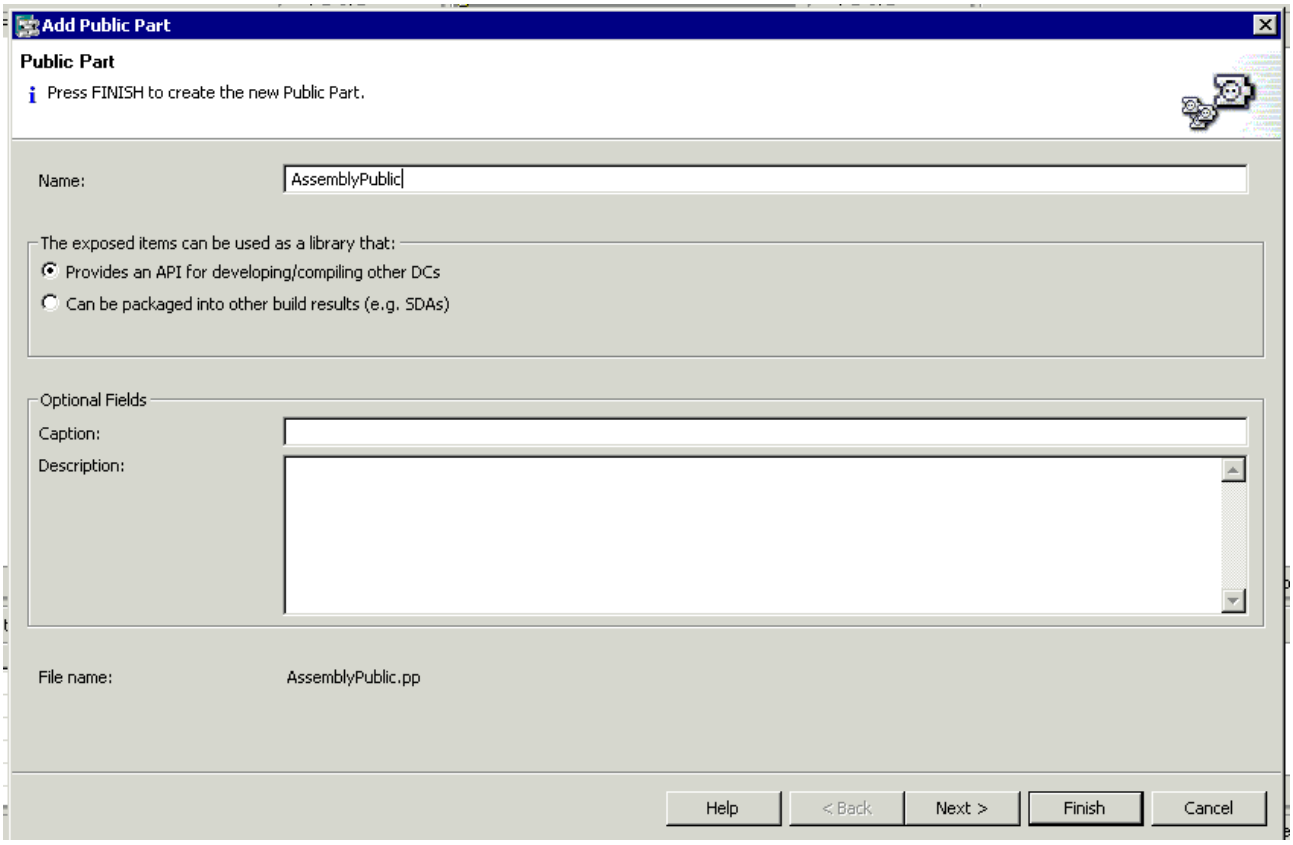
Open “Development Configuration Perspective” and select Local Development -> My Components -> “ExternalLibraryContainerDC”. Now, navigate to DC Metadata -> Public parts as shown below



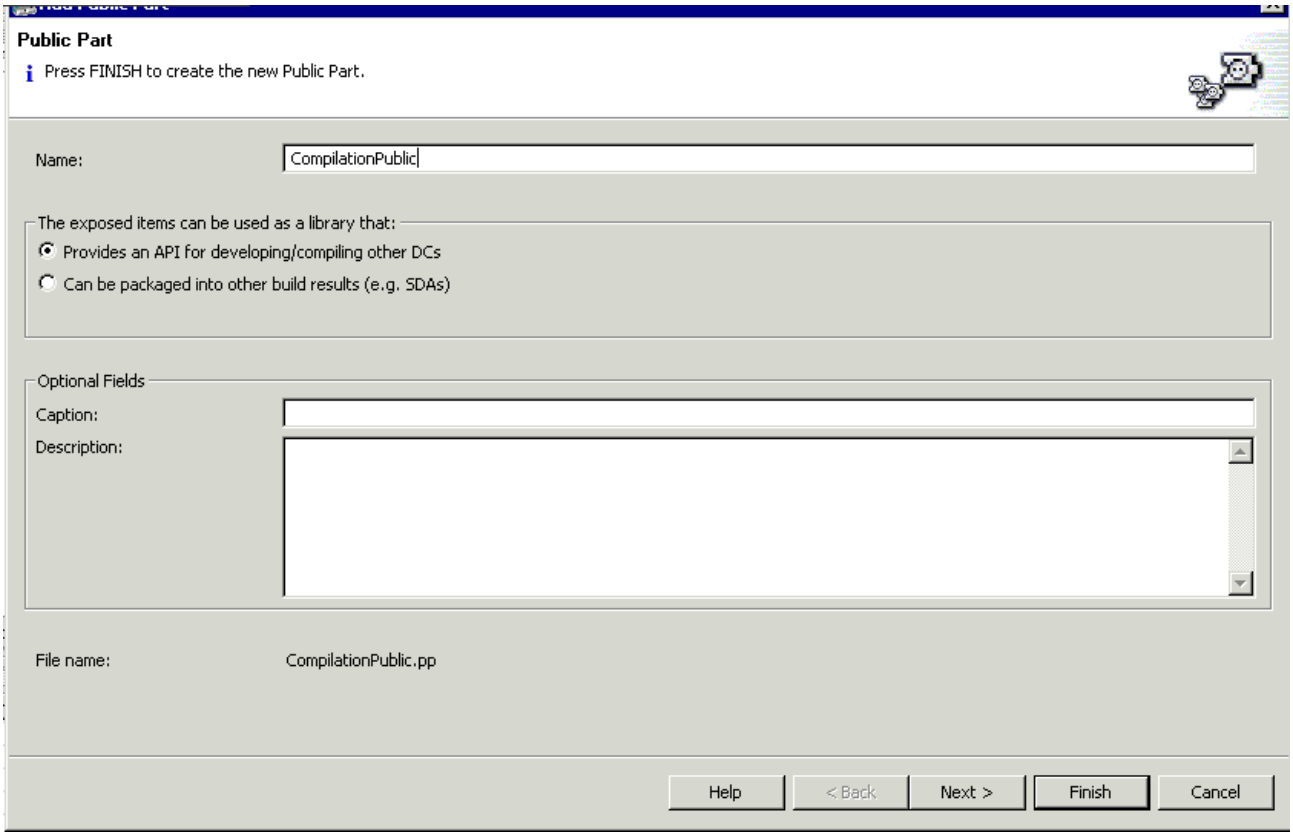
Right click on “Public parts” and select “New Public part”.



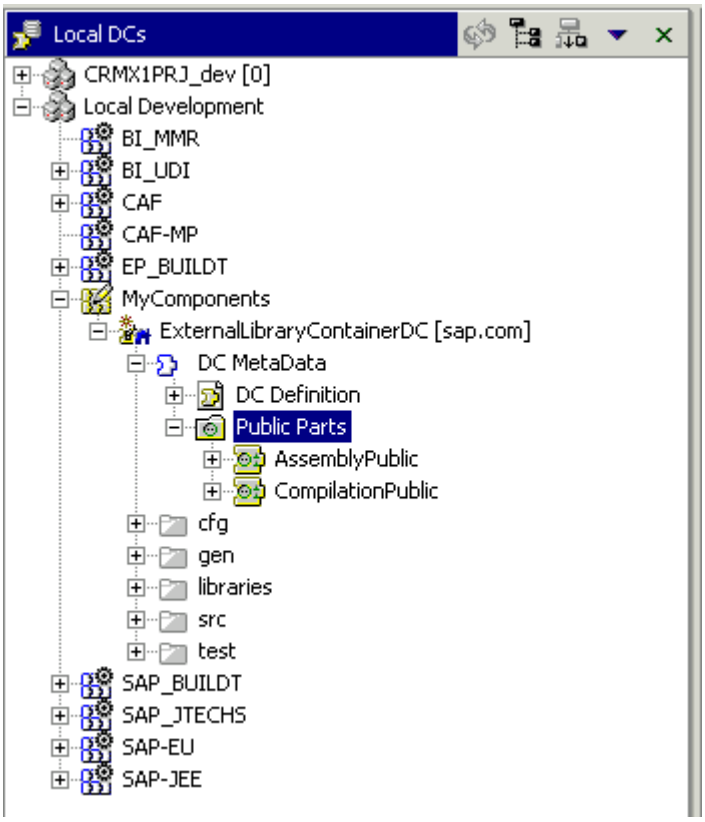
Enter name “AssemblyPublic” and click on “Finish”



Create one more public part similarly and name it “CompilationPublic”.

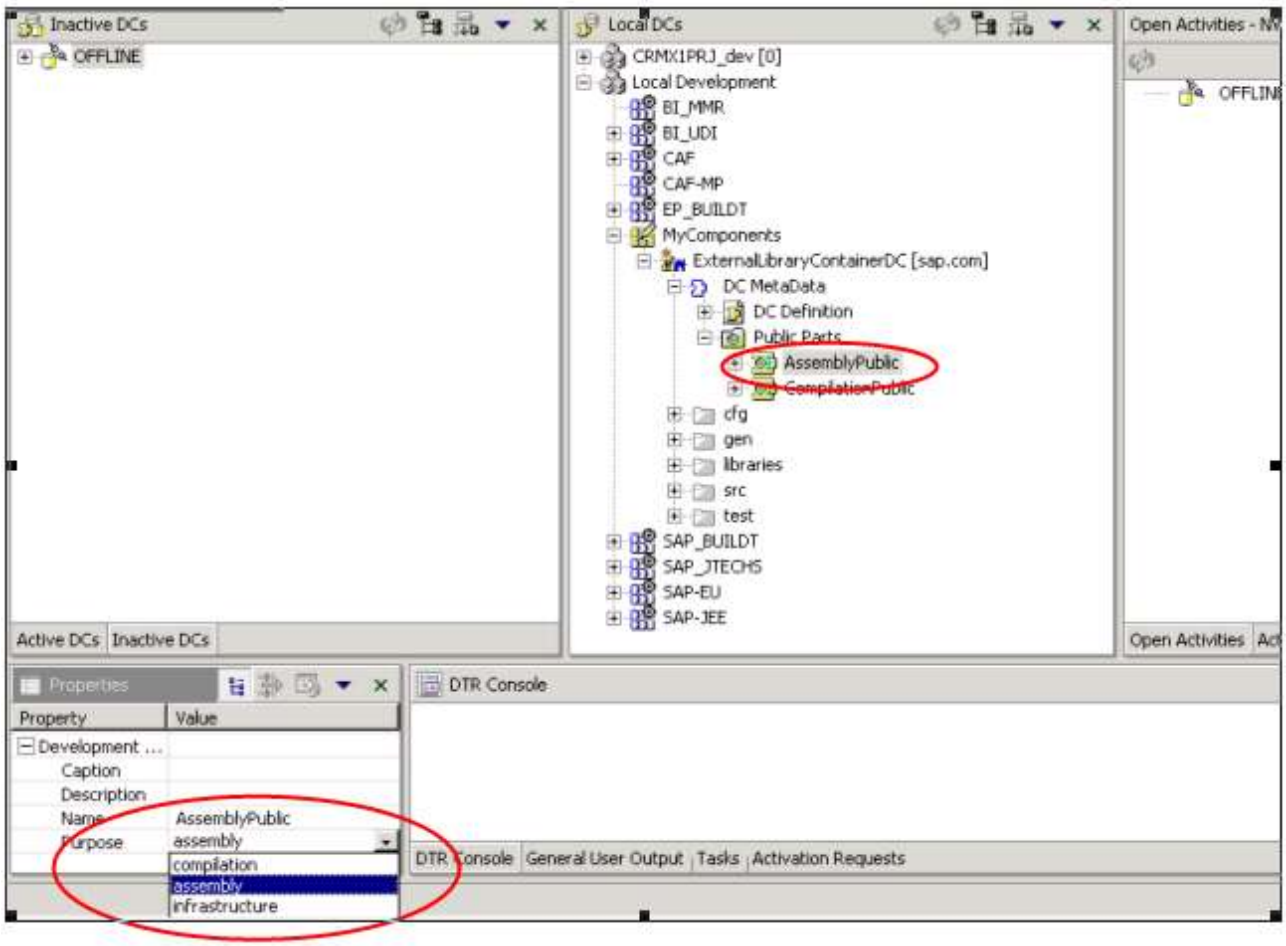


Now you will be able to see two public parts under DC Metadata -> Public parts as shown below.



Now we need to change an important property for public part “AssemblyPublic”.

Select “AssemblyPublic” and update its purpose to “Assembly” in properties window as shown below.

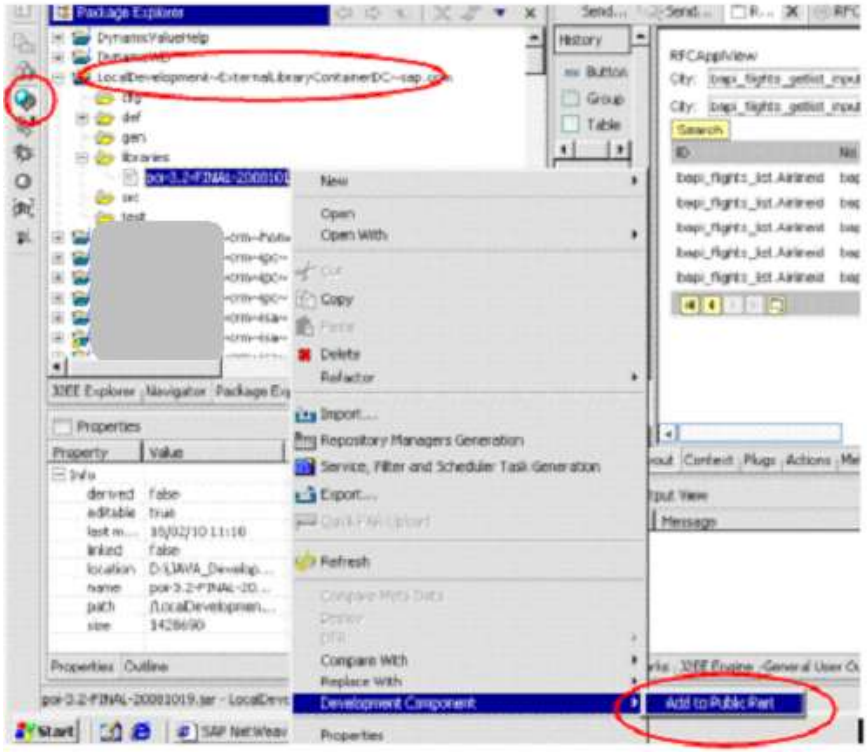


Next step will be to add your JAR file “Apache POI” to both these public parts (AssemblyPublic, CompilationPublic)

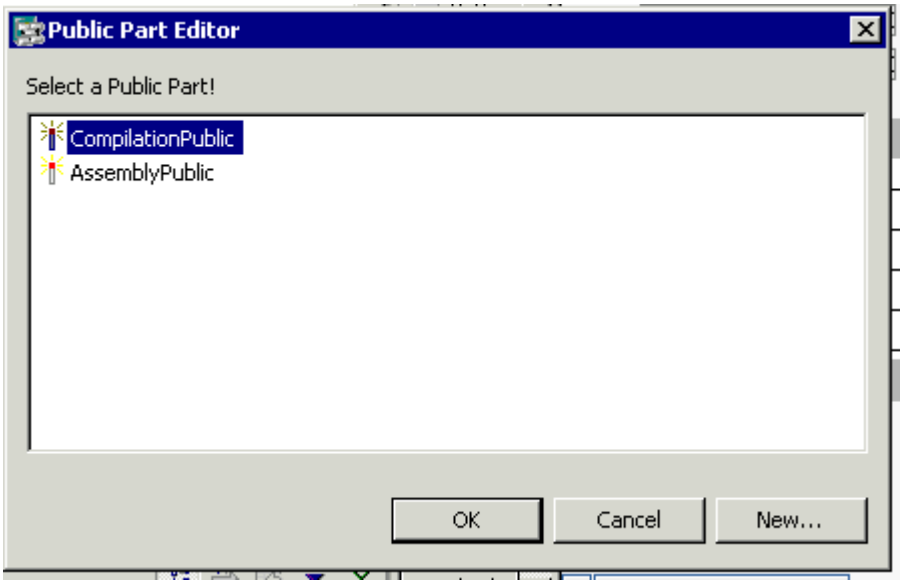
Step – Add JAR file to public part

Switch to “Package Explorer” in J2EE perspective. And open folder “libraries” under “ExternalLibraryContainerDC” Development Component. Here you will find your uploaded JAR file.

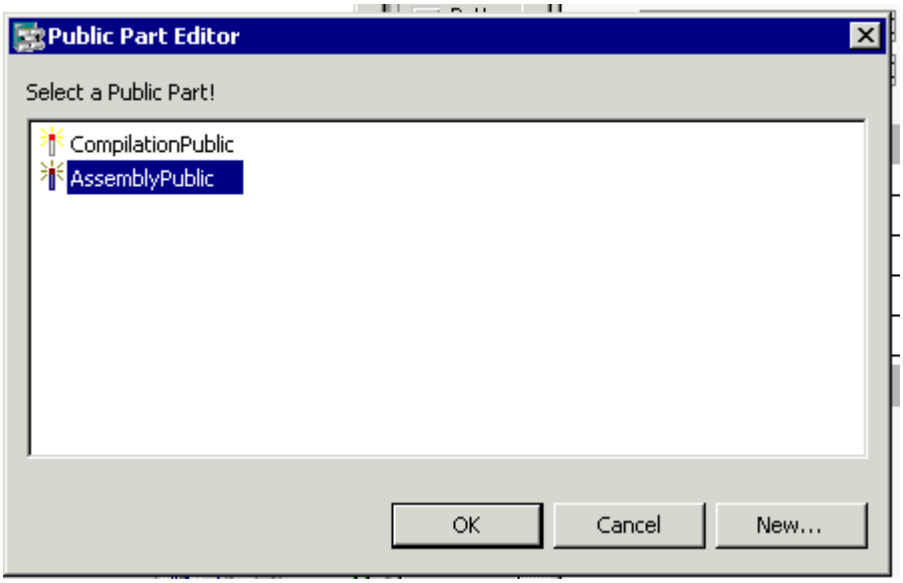
Right click on “Apache POI” JAR file and navigate to “Development Component” -> “Add To Public part”. Refer below figure.



You will be prompted to select the required public part. Select “CompilationPublic”



Repeat same process for “AssemblyPublic”.



Press “OK”

Now, you have successfully added your JAR file to both public parts and thus, it is now ready to be used by other Components.

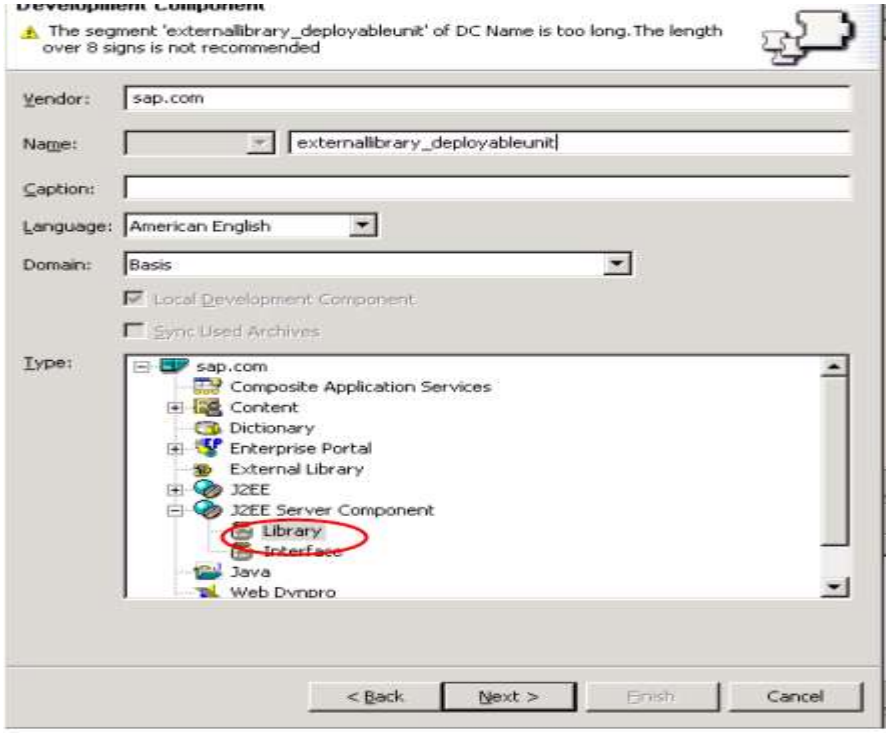
Create a deployable unit for “ExternalLibraryContainerDC”

Our library is properly configured in Netweaver Developer Studio. Now, we need to deploy this library to server. For this we need a deployable unit which references to “ExternalLibraryContainerDC”.

We will name this deployable unit as “ExternalLibrary_DeployableUnit”

Step – Create a Development Component of type “J2EE server component -> Library”

Create a new Development Component in your Local Development environment and select “J2EE server component -> Library” as type of project. Name it as “externallibrary_deployableunit”.

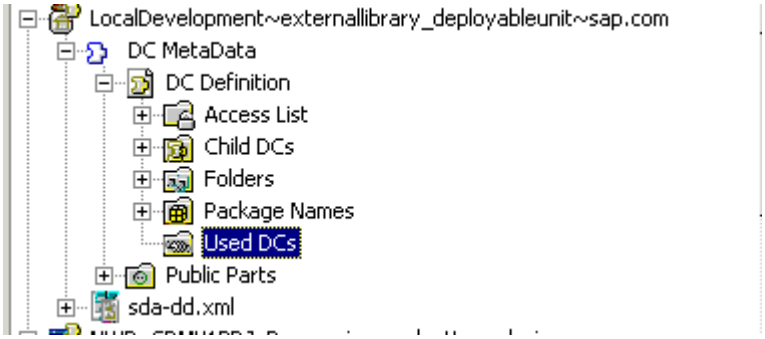


Click Next and Finish.

Step – Reference "ExternalLibraryContainerDC" from "externallibrary_deployableunit".

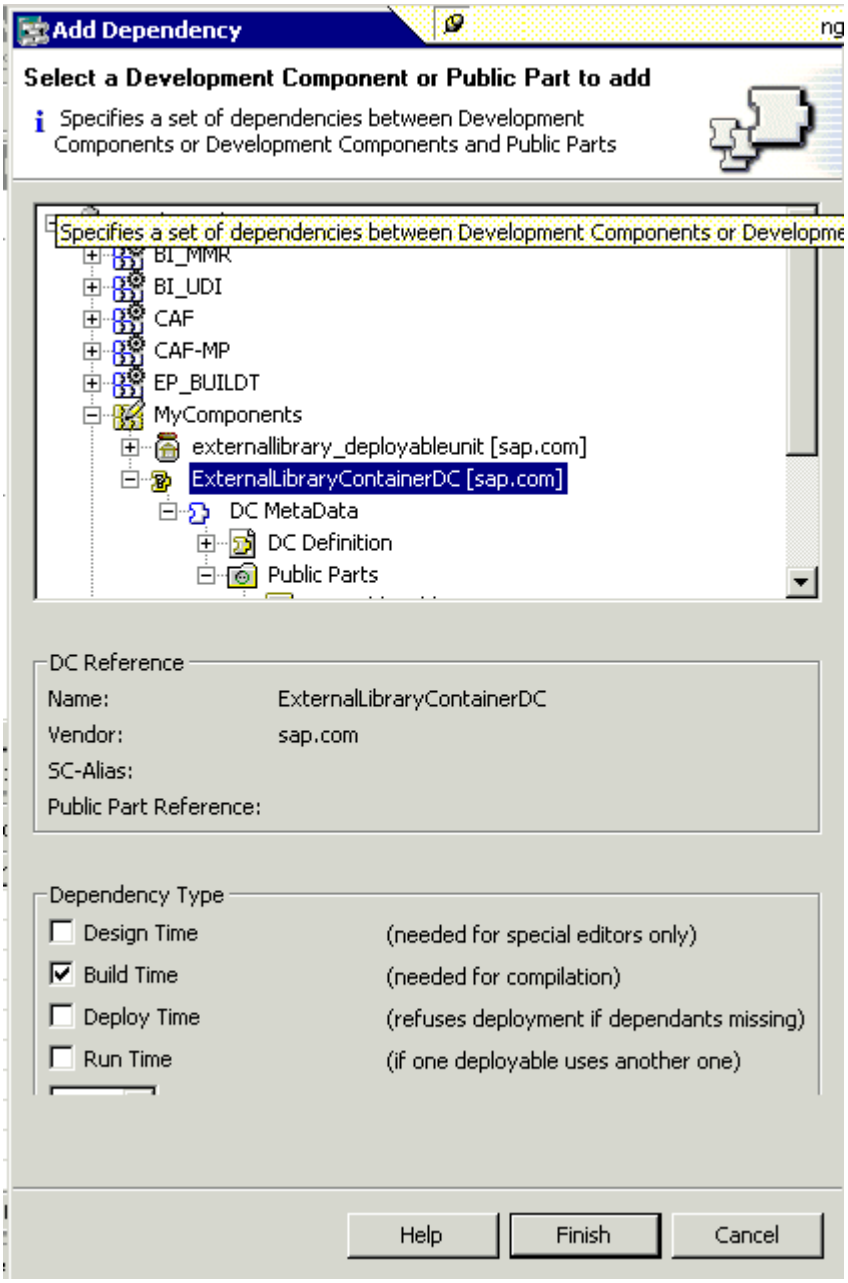
Add a "build" time dependency in "externallibrary_deployableunit" to assembly part "AssemblyPublic" of "ExternalLibraryContainerDC".

Browse to "Used DCs" part of "externallibrary_deployableunit"



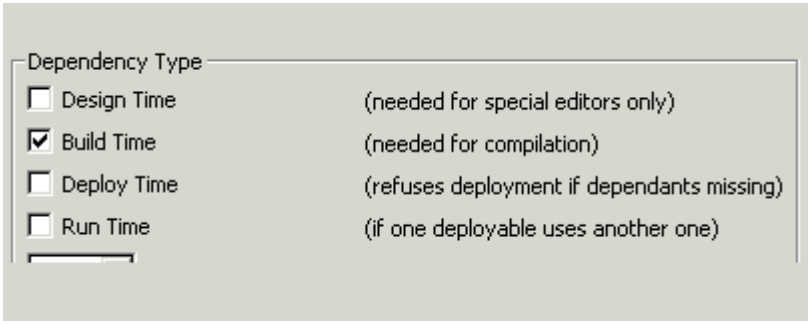
Right click on "Used DCs" and click on "Add Used DC"

Now in prompted window select Local Development -> "ExternalLibraryContainerDC" -> Public parts

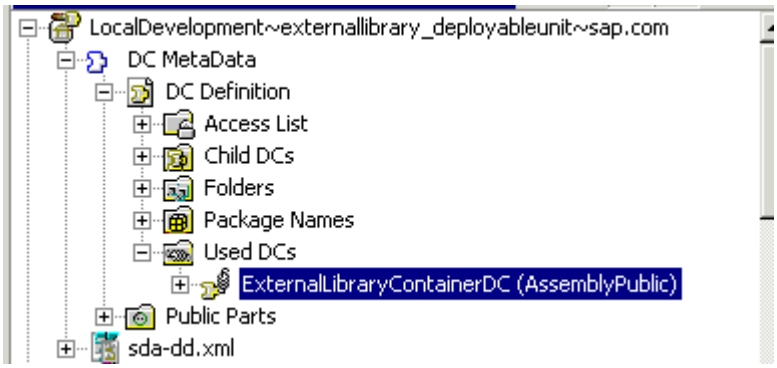


Select assembly public part “AssemblyPublic”

Do not forget to select “Build Time” as dependency type.

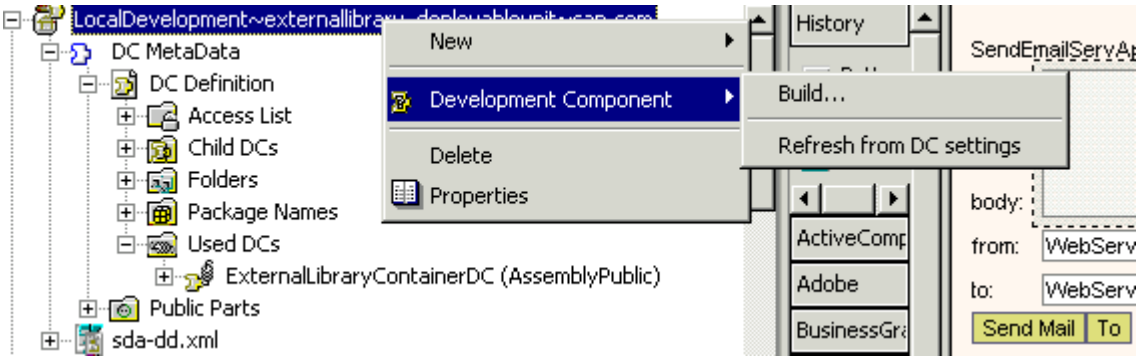


Click Finish and you will be able to see added public part to “Used DCs” portion of “ExternalLibraryContainerDC”.

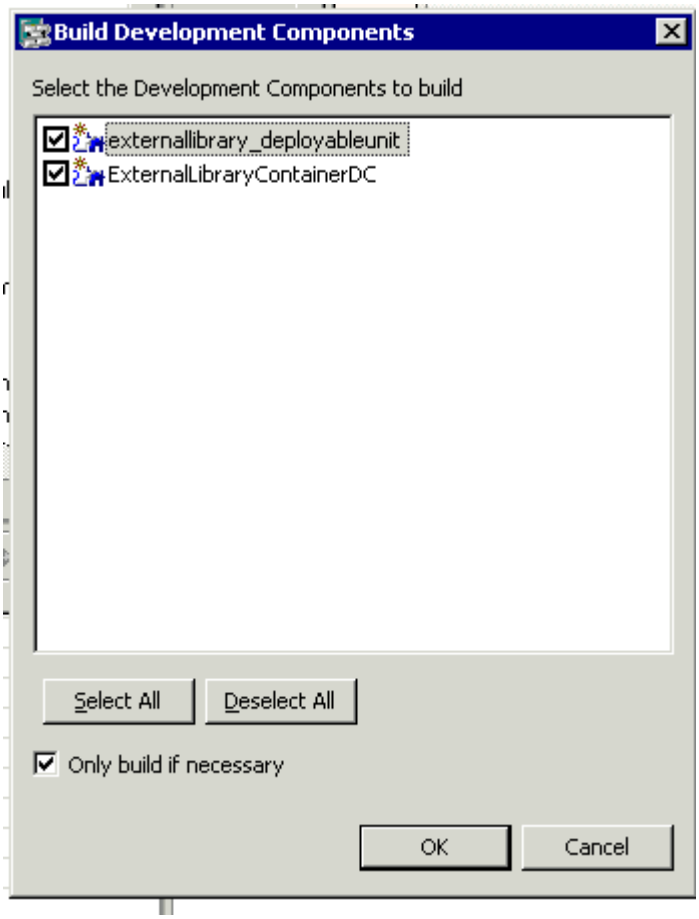


Step - Built and deploy “externallibrary_deployableunit”.

Built the deployment unit for “ExternalLibraryContainerDC” i.e. “externallibrary_deployableunit”.

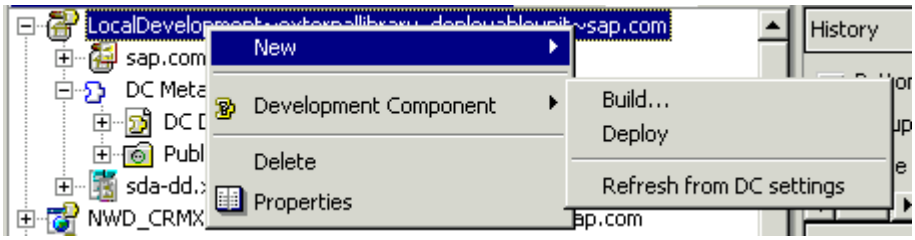


As you will see, because of dependency we created it automatically asks to build “ExternalLibraryContainerDC”.

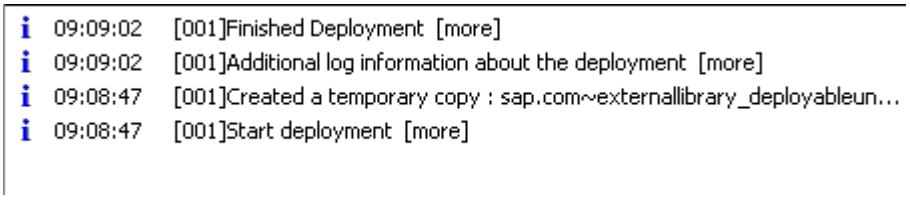


Click OK

Now we will deploy it to server so that our “Apache POI” jar is available in server to be used by other components.

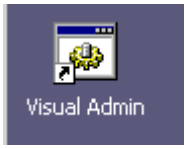


On successful deployment on server, it will give a message in deploy output view.



Step – Crosscheck whether it has reached server.

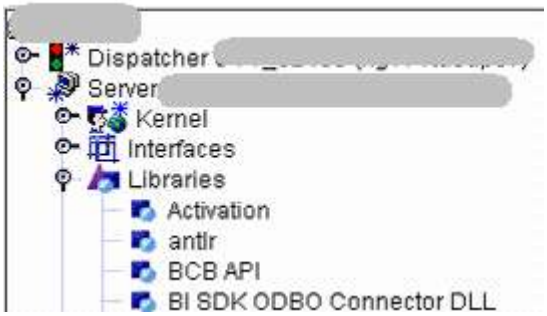
Open your J2EE server visual admin tool.



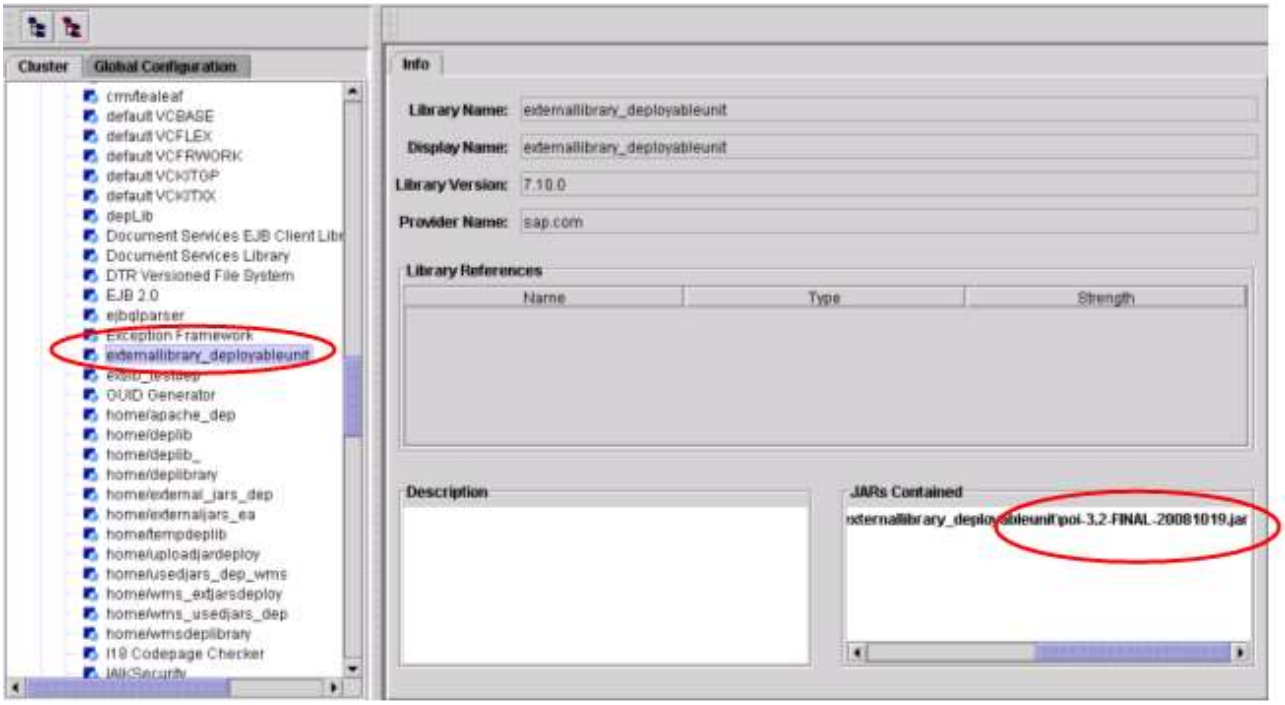
You will find it in

"<Installation Drive>:\Program Files\SAP\admin\go.bat"

Login and navigate to Server -> Libraries.



Here you should be able to find “externallibrary_deployableunit”. Click on it.



Using deployed library in your web application

By now we have successfully configured and deployed our library “Apache POI” on server. Now next step is to use this library in your web application.

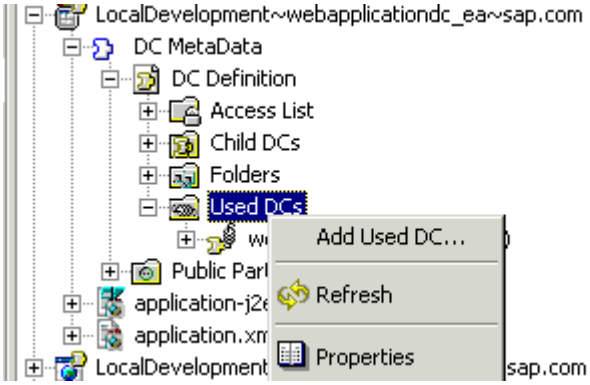
Note: I am assuming that you already have your web application and its deployable unit ready in your workspace.

I have named them:

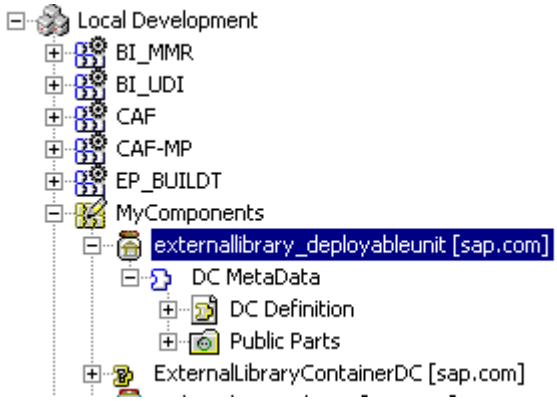
- WebApplicationDC_WA (Web Development Component)
- WebApplicationDC_EA (Enterprise Application Development Component)

Step – Create dependency to “externallibrary_deployableunit” from “WebApplicationDC_EA”

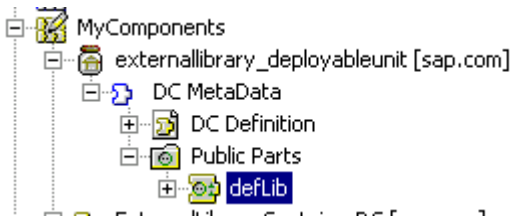
Open WebApplicationDC_EA in “J2EE DC Explorer” and navigate to “DC Metadata -> DC Definition -> Used DCs”. Right click there and select “Add Used DC”



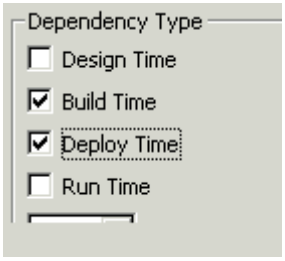
In prompted window navigate to “Local Development -> My Components -> externallibrary_deployableunit”



Select “DC metadata -> Public Part -> <Public Part Name>”



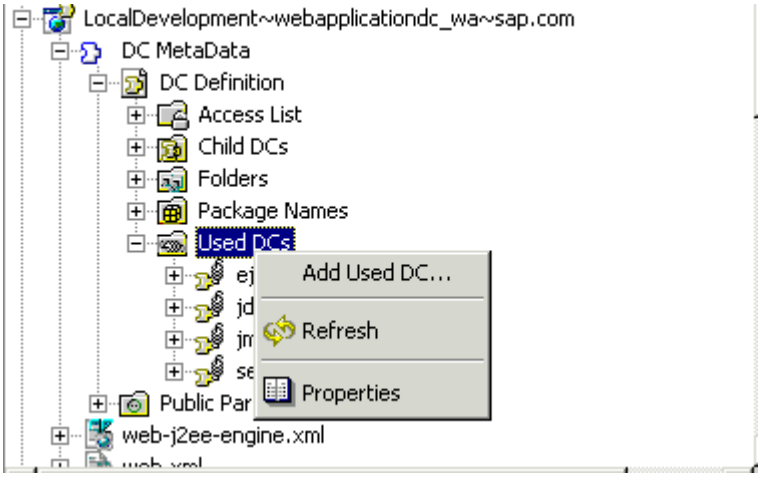
Select “Built time” and “Deploy time” as dependency type



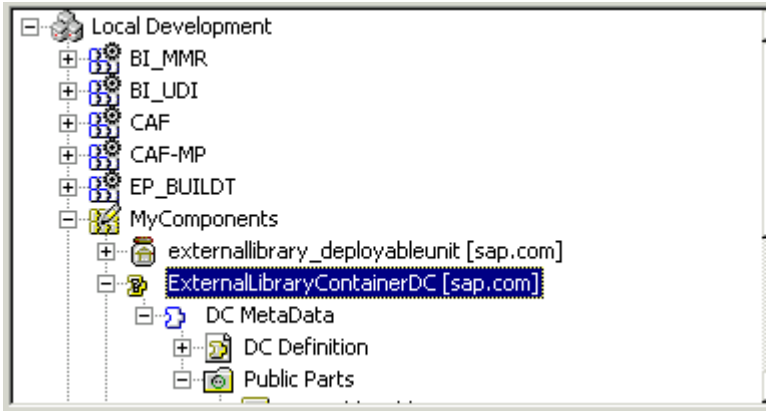
Click Finish.

Step – Create dependency to "ExternalLibraryContainerDC" from "WebApplicationDC_WA"

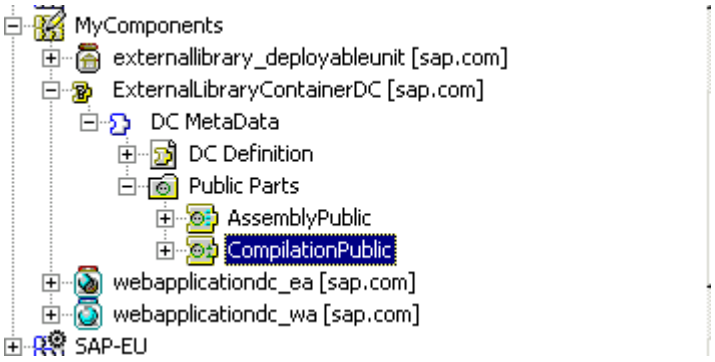
Open WebApplicationDC_WA in "J2EE DC Explorer" and navigate to "DC Metadata -> DC Definition -> Used DCs". Right click there and select "Add Used DC".



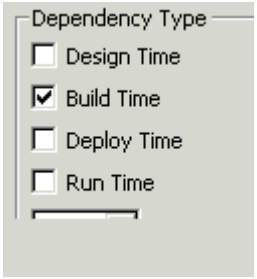
In prompted window navigate to "Local Development -> My Components -> ExternalLibraryContainerDC"



Select "DC metadata -> Public Part -> **CompilationPublic**"



Select “Built time” as dependency type

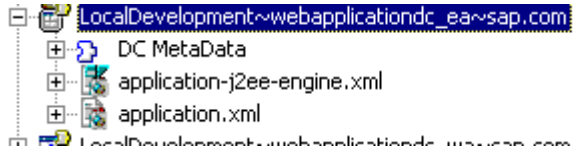


Click Finish.

Step –Add reference to “Apache POI” in “WebApplicationDC_EA”

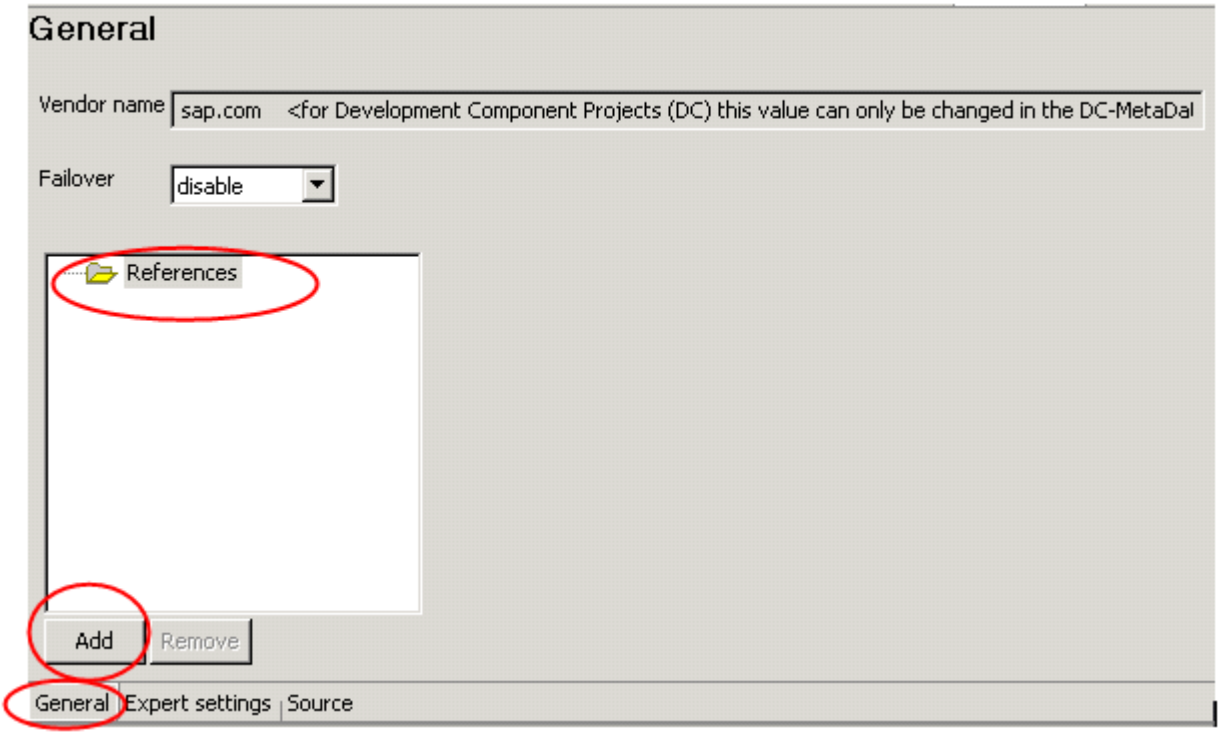
Open WebApplicationDC_EA in “J2EE DC Explorer”.

Navigate to “WebApplicationDC_EA -> Application-J2EE-Engine.xml”

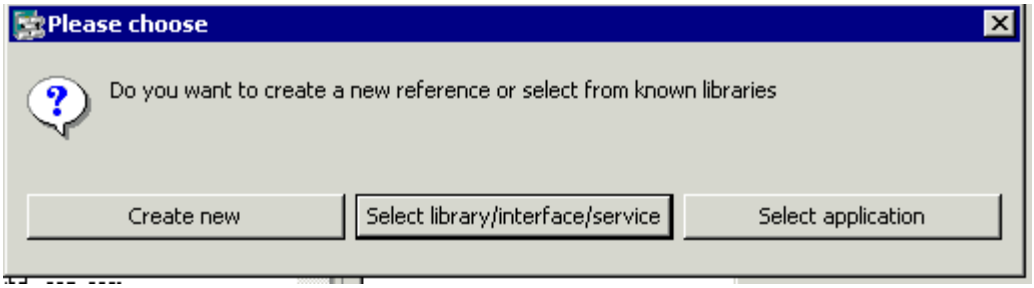


Open “Application-j2ee-engine.xml”.

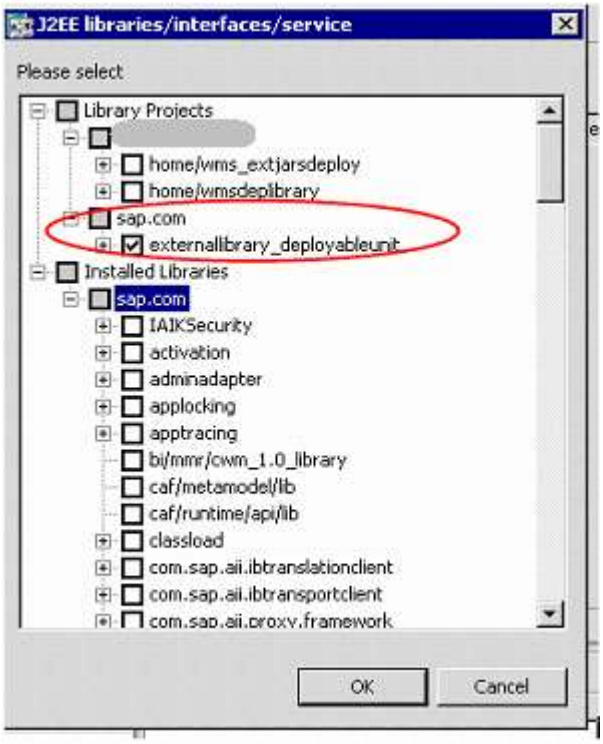
In General tab select “References” and click on “Add”



In prompted window select “Select library/interface/service”

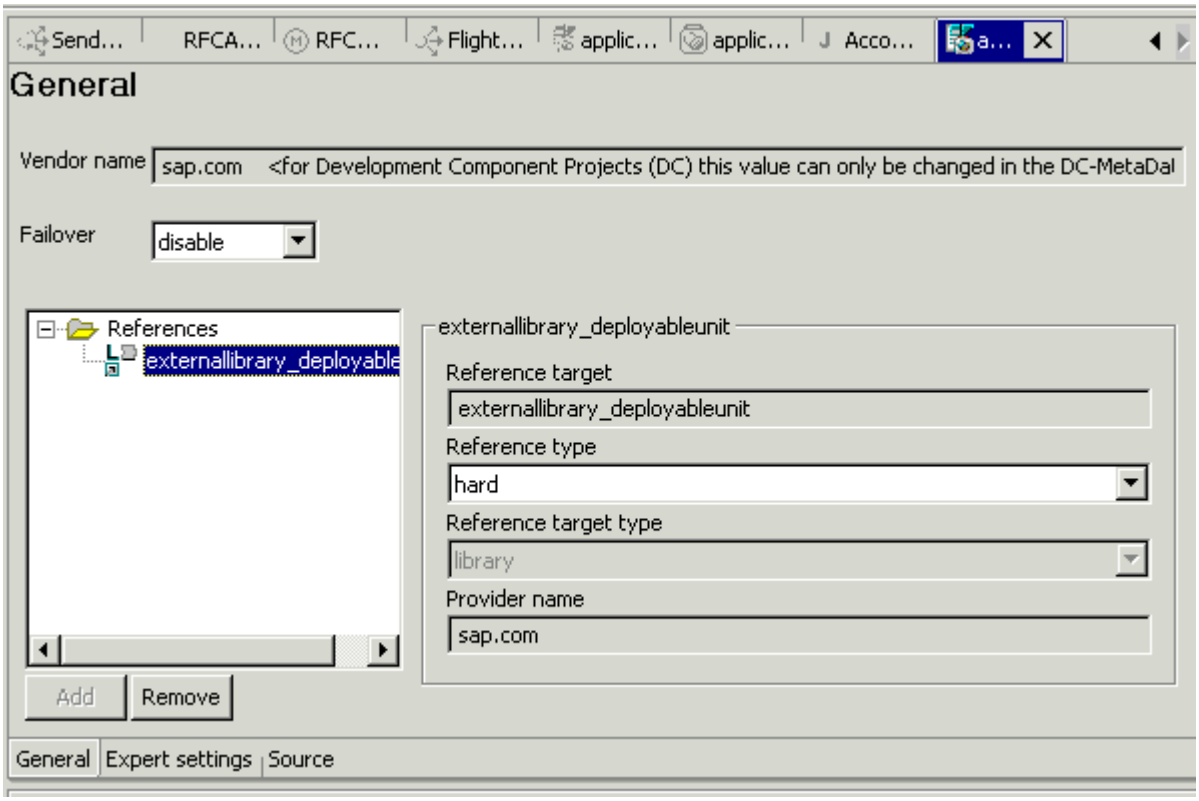


Now select “externallibrary_deployableunit”. This is deployable unit for “ExternalLibraryContainerDC” which contains “Apache POI” jar file.



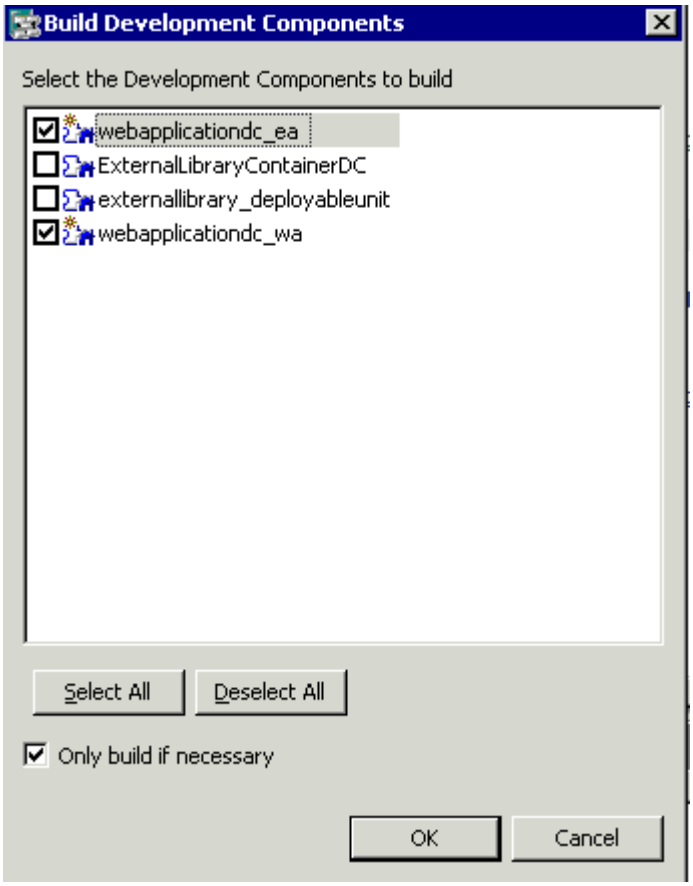
Click OK.

Reference to jar file is now available with this development component.



Note: Do not forget to save this application-j2ee-engine.xml

Build and deploy “WebApplicationDC_EA”



Your jar is now ready to be used by this web application.

Related Contents

For more information, visit the Java 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.