Module objectives:

* Be able to create a z/OS Project
* Be able to effectively utilize the interactive debugger

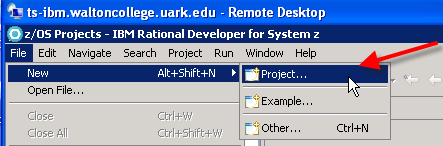
**Simple Cobol Program Illustrating Interactive Debugger**

This exercise extends Exercise 1 to include using the interactive debugger with Cobol; the interactive debugger requires creating a z/OS project. Creating a Cobol z/OS project also requires a number of PDSs as shown below.

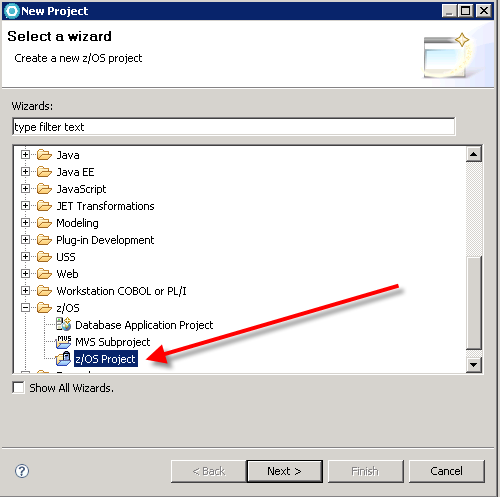
|  |  |  |
| --- | --- | --- |
| **Name** | **Category** | **Type** |
| UOASxxx.GENERTED.JCL | SOURCE | JCL |
| UOASxxx.WORK.JCL | SOURCE | JCL |
| UOASxxx.COBLIST.LISTING | LISTING | COBOL |
| UOASxxx.COBOL.SYSDEBUG | Others | SYSDEBUG |
| UOASxxx.COBOBJS.OBJ | Others | OJECT |
| UOASxxx.COBOL.COPYLIB | SOURCE | COBOL |
| UOASxxx.LOAD | Others | LOADMOD |
| UOASxxx.ERRCOB | LISTING | COBOl |
| UOASxxx.WORK.OUT | SOURCE | TEXT |
| UOSSxxx.WORK.DATA | SOURCE | TEXT |
| UOASxxx.WORK.COBOL | SOURCE | COBOL |

If you do not already have these PDSs, then create them.

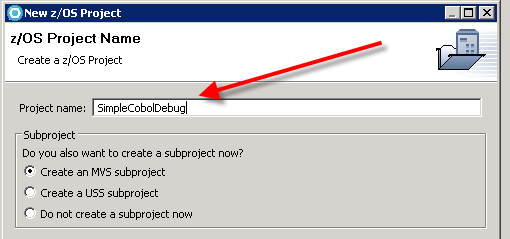
Begin creating the z/OS project by clicking the z/OS Projects tab (so it has the focus) and then from the main menu, File🡪New🡪 Project as shown below.



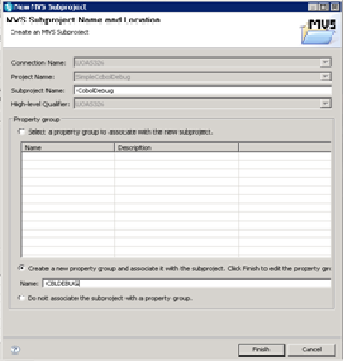
From the New Project dialog, scroll down and expand the z/OS node and select z/OS Project.



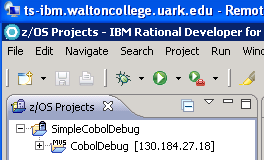
Click the Next button and enter a name for your project in the **Project name:** textbox as shown below and then click the Finish button.

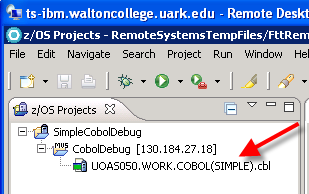


Clicking the Finish button opens the MVS Subproject Name and Location dialog. Enter a MVS Subproject name in the **Subproject Name:** textbox. If this is the first time, click on “create new property group” and specify a file name.



Click the Finish button—the z/OS Projects pane should appear as follows.

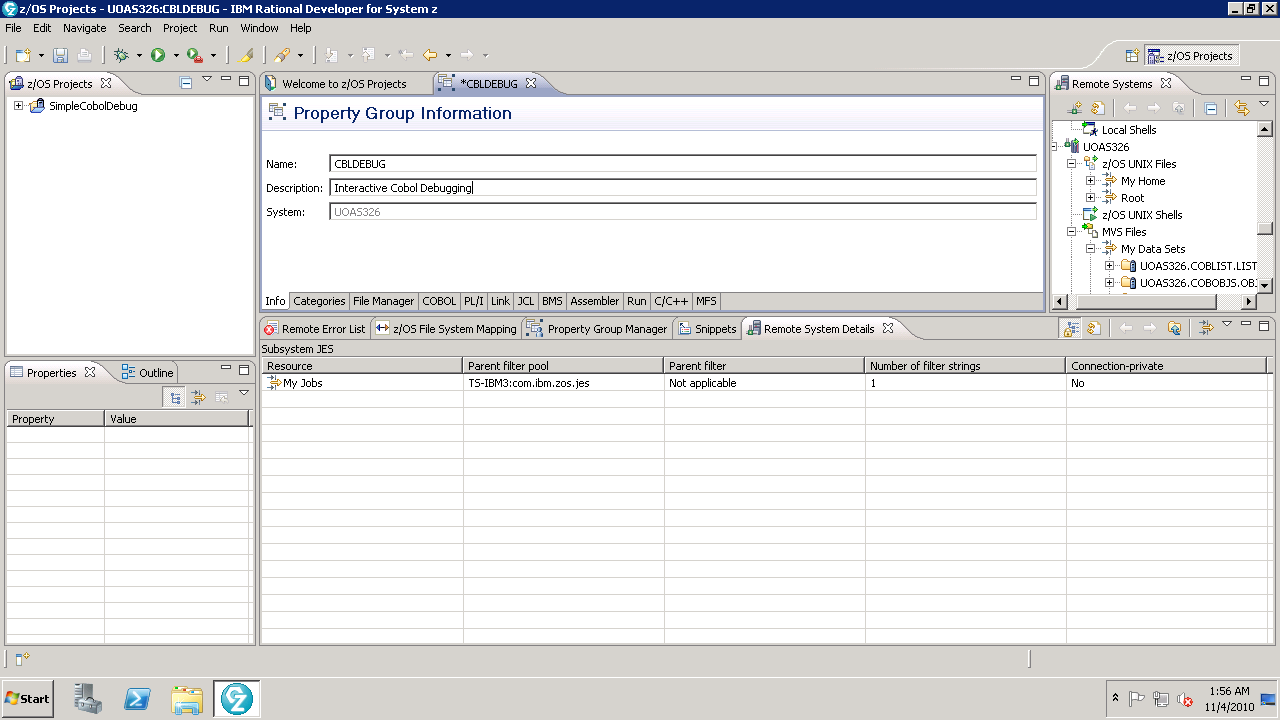


Based on the Simple Cobol exercise, you should have all the files needed to complete this project. This example will use the PDS structure, UOASxxx.WORK.xxx whereas the initial project documentation used a UOASxxx.RDZ.xxxx PDS structure.

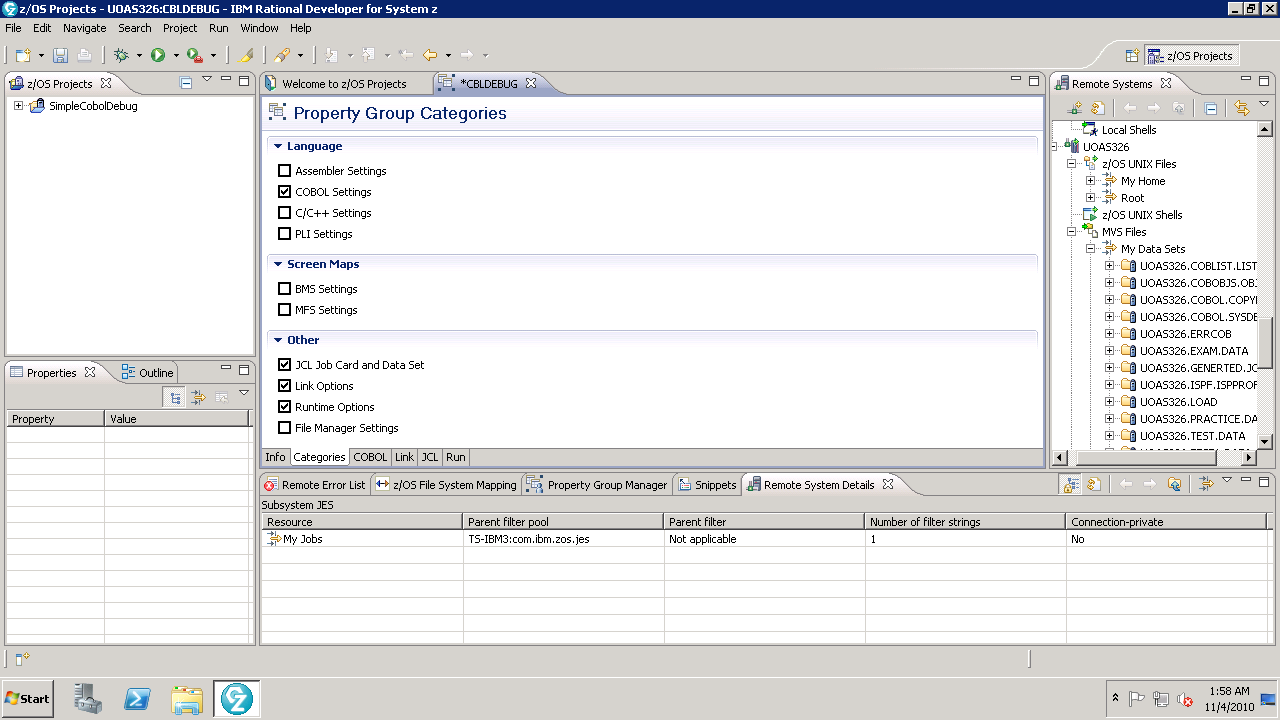
Drag your Cobol member and drop on your z/OS MVS Subproject—see image at right.

Recall that the Cobol member in the z/OS Project is the same member as the one in the Remote System view. Thus, if you delete it from your z/OS Project, it is **gone**. Thus, you probably need to almost always remove ***source*** files from the z/OS project rather than delete from the z/OS Project.

Next, the Properties Group will need to be configured correctly—mostly copy and paste. Click on the Info tab on the Property Group Information and enter a description.

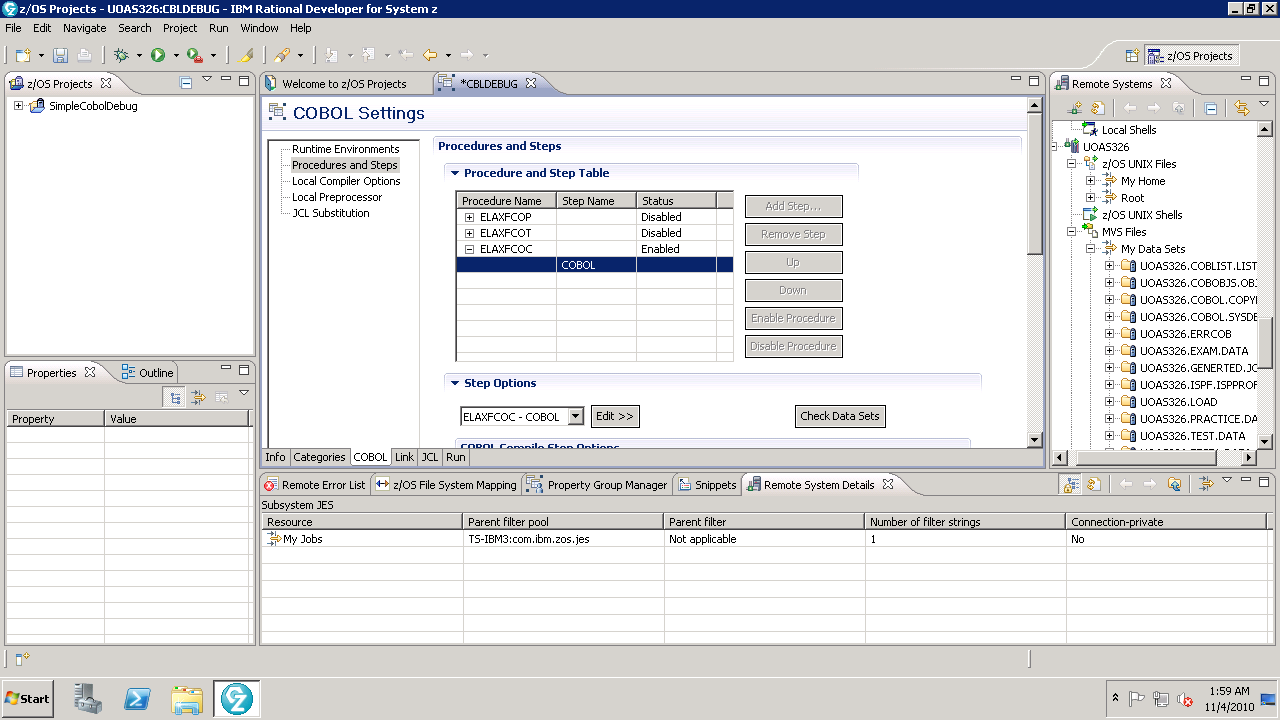


In the Categories tab, set the appropriate setting for a Cobol program as shown below



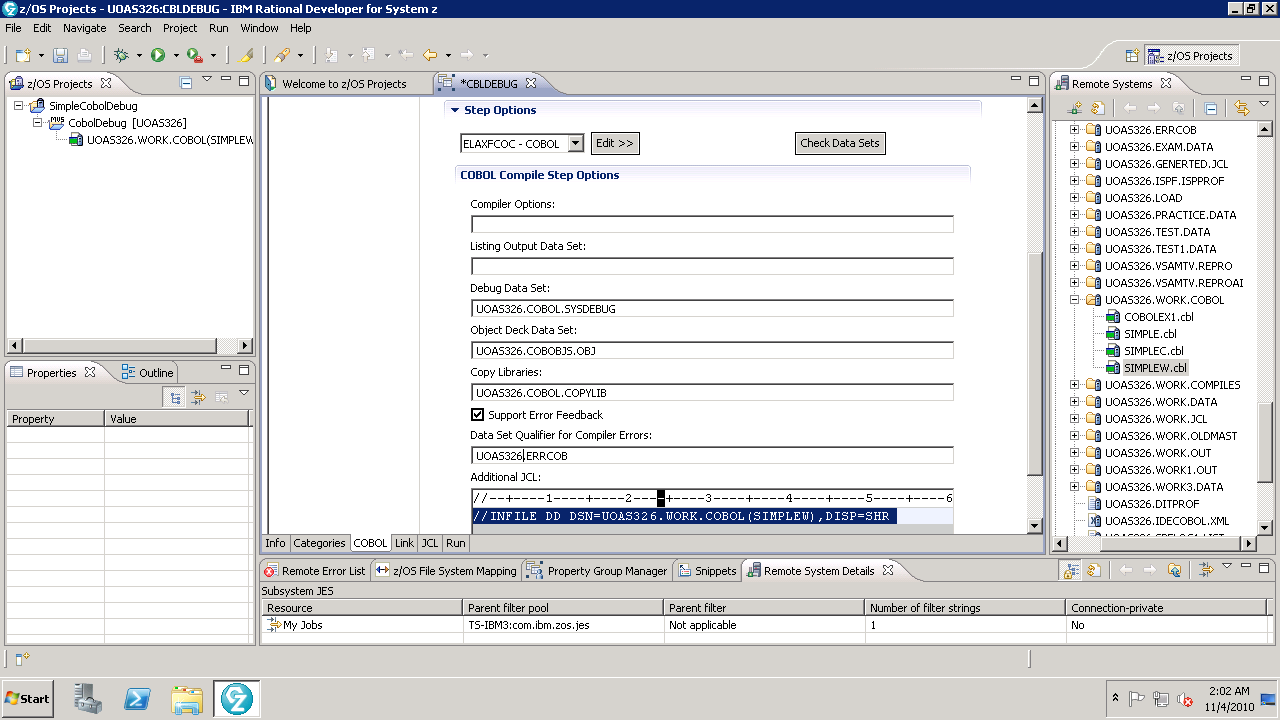
You will need to edit the properties of several steps. Click on the COBOL tab to edit the Cobol Settings.

Expand last entry in the middle COBOL Setting pane and double-click COBOL

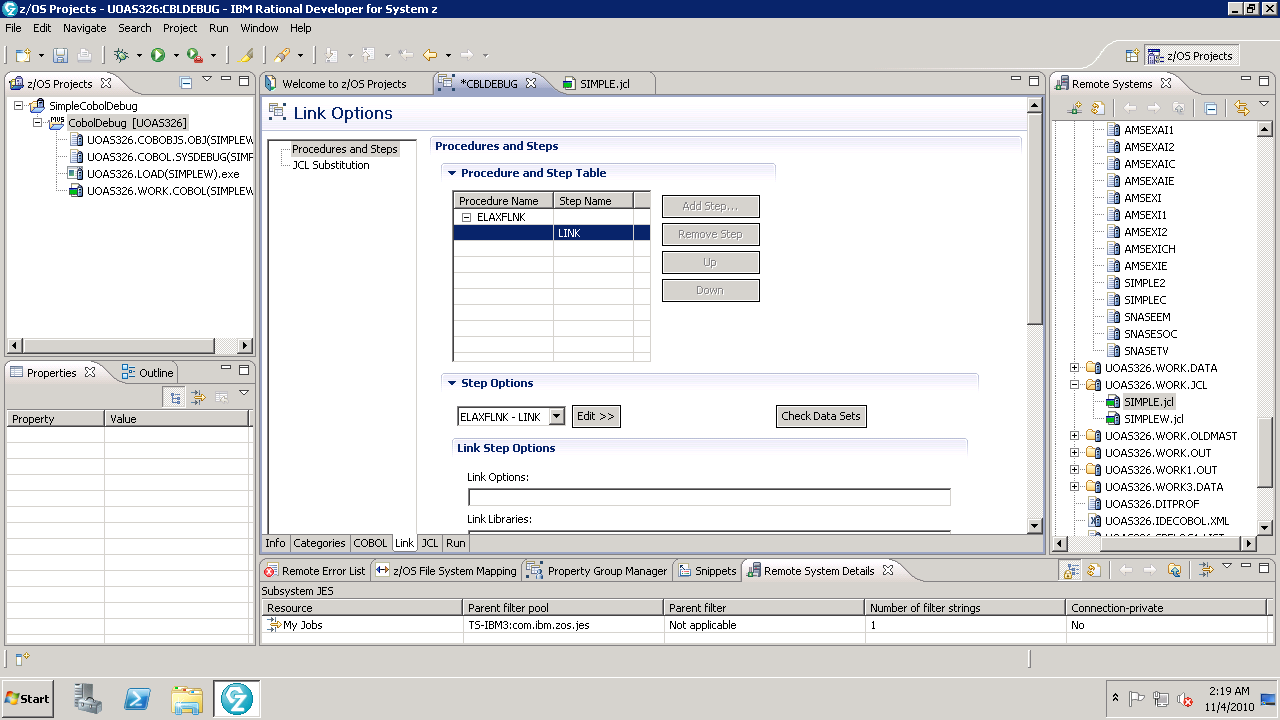


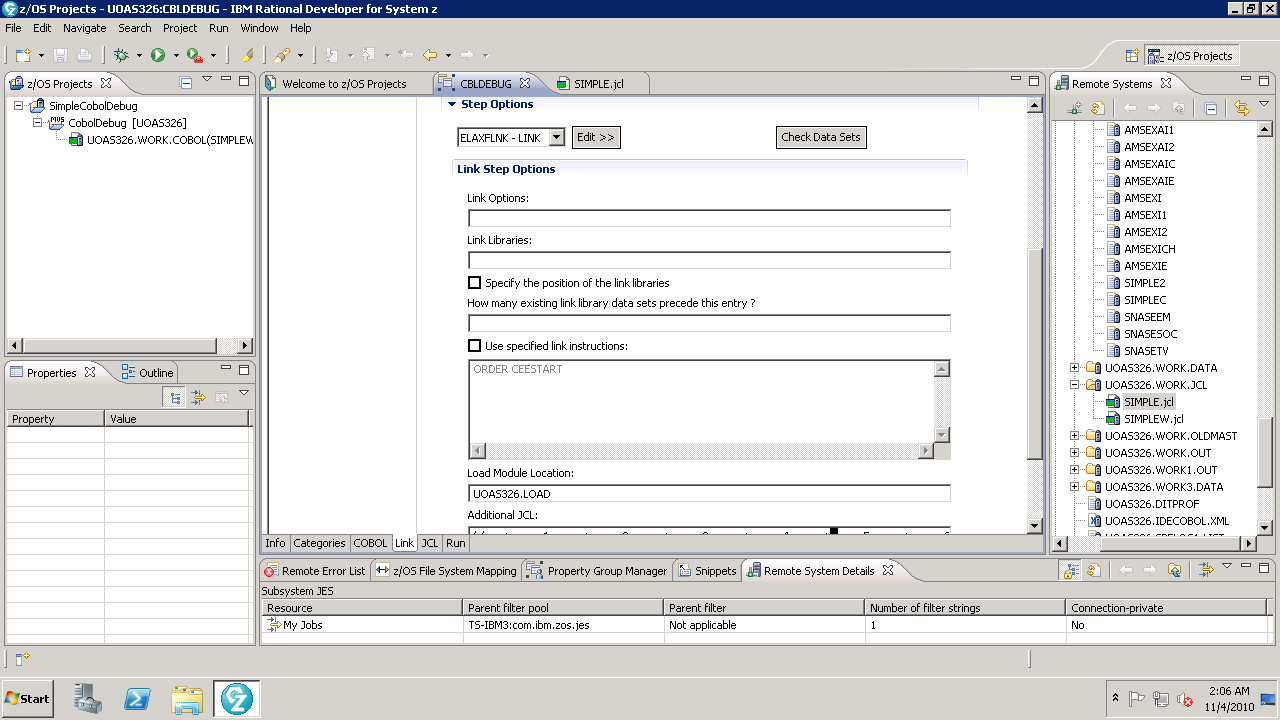
Change all the <HLQ> (High Level Qualifiers) to your account id. Then Enter the JCL needed to find the COBOL program—in this case it is:

//INFILE DD DSN=UOAS050.WORK.COBOL(SIMPLEW),DISP=SHR

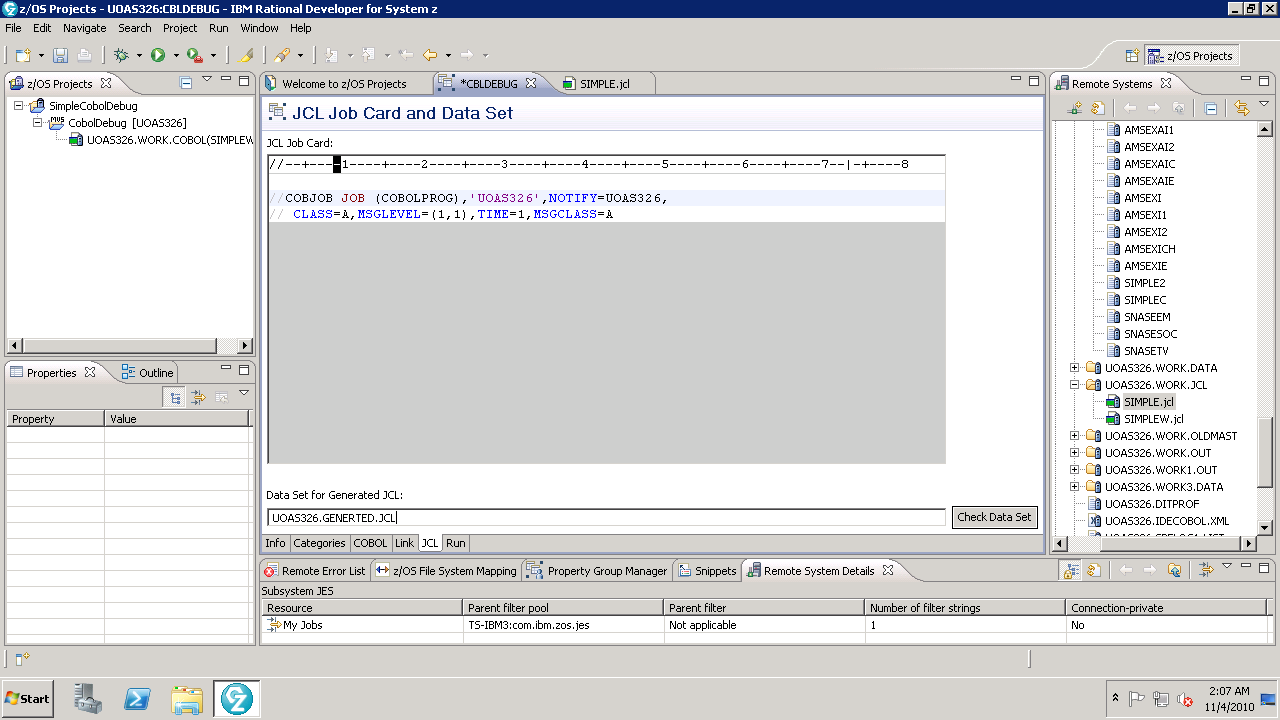


Click Link Tab and expand the entry and double-click LINK. The only addition to this page to change the high level qualifier to you account id on the Load Module Location.

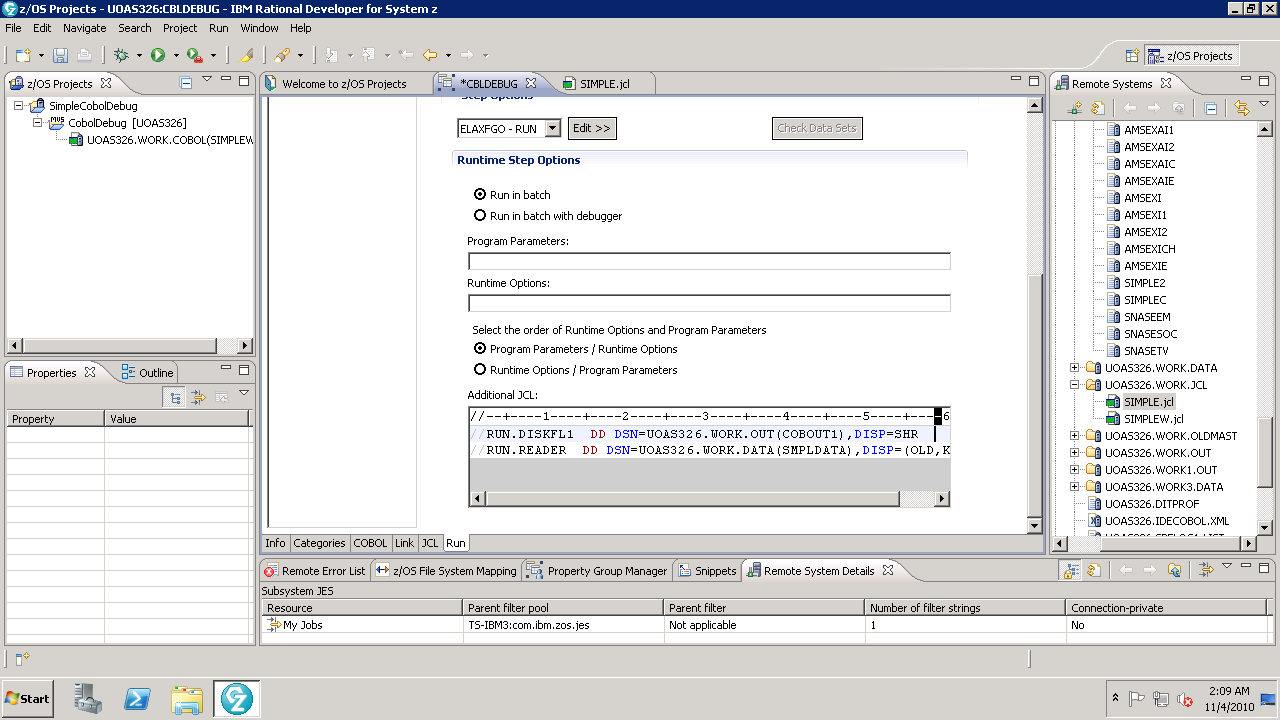




Enter any necessary changes to the JCL tab to enter job card information. Although not needed, enter a data set to hold generated JCL.



Click the Run tab to edit the Run-time Options. As before, expand the entry and double-click RUN. You will need to copy in the JCL to locate the input and output PDS members. However, replace the GO with RUN as that is the default Run Procedure Setup Name.



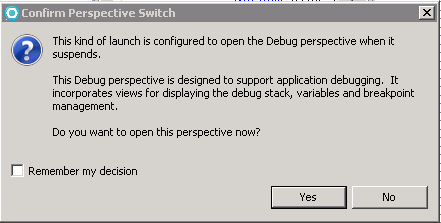
Click the OK button, the Finish button and then the OK button to return from the Property Group editing process to the z/OS Project.

Now, right click the MVS Subproject and select Rebuild Subproject.

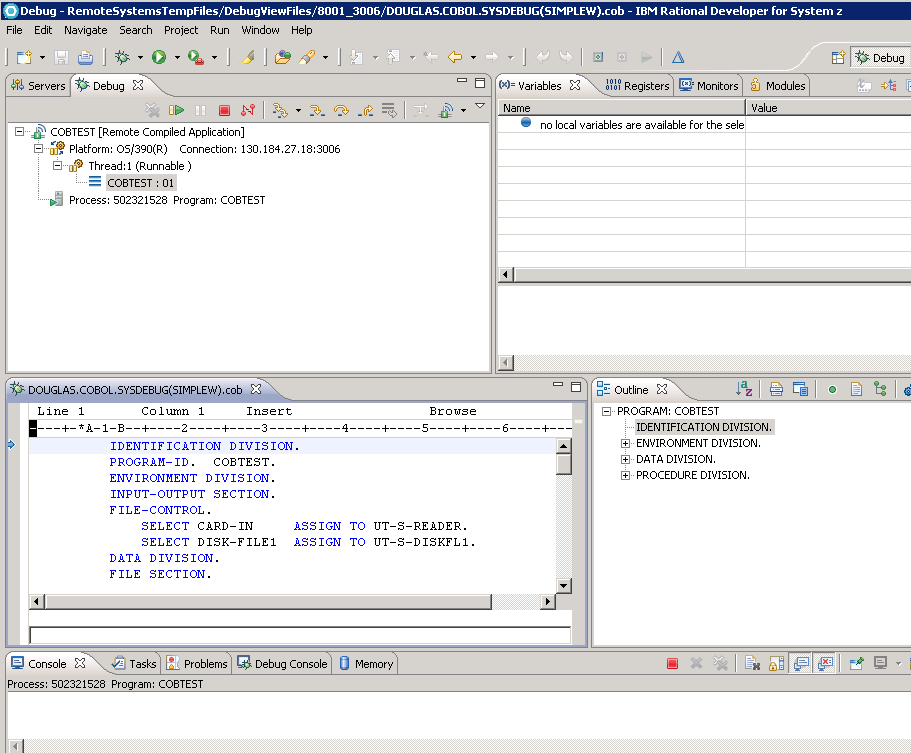
If the project builds correctly, it should have a xxxx.exe (in the z/OS Projects pane) file that you can run or debug. If you right-click the .exe file and select Run Application, then the project will run. You should do this to ensure that it runs correctly and check the output to see if it is correct.

**A first tour of Cobol debugging**

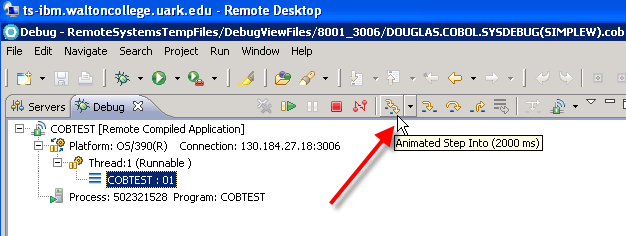
Right-click the .exe file and select Debug Application. An dialog window prompts you for Yes to launch the Debug perspective. Click Yes.



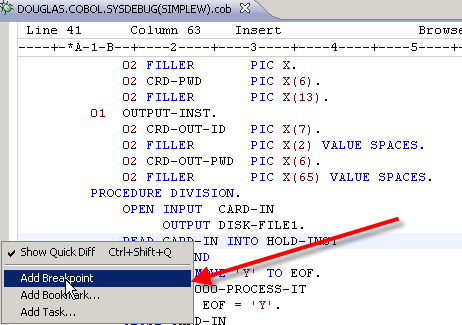
Your initial interface will be somewhat like the one shown below.



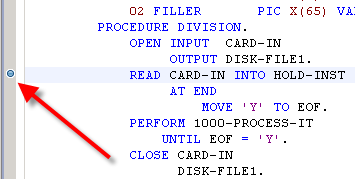
Note the tabs from upper left to lower right—Debug, Variables, the Cobol program and the Outline view. As throughout RDz, tooltips appear when moving the mouse over the icons – see example below.



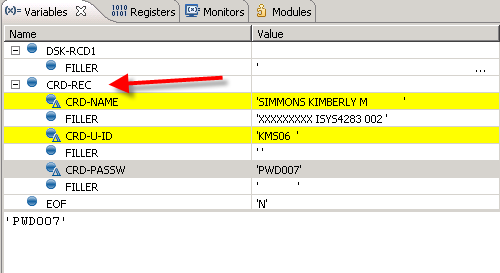
In the bottom left pane, scroll down the page until you get to the first READ statement in the Procedure Division. Right-click on the line and select Add Breakpoint.



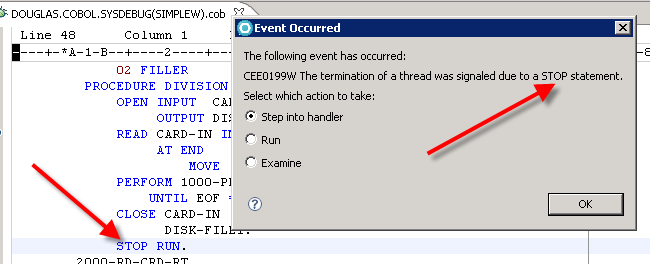
Notice that a round circle on the left edge of the program statement which indicates there is a break-point.



From above, click the Step-Into icon on the upper left pane that was shown above. Notice that in the upper right pane, you can expand the records to see the variables. Also, note that the Step Into continues until you click it again—thus, it will walk through the records until it ends unless you click the icon to halt it.



When you let it run to fruition, the program eventually reaches the STOP STATEMENT. The Debugger then prompts you with information that the program is stopping and allows you to click the OK button.



Interactively stepping through the code to find logical errors provides magnitudes of productivity increases. For this simple program, you may want to interject a logical error such as the program record format not match the input record for a particular field and inspect the variable values in the debugger as the records are read and processed.