**Objective**

The homework provides you with a basic experience in the ABAP development environment in SAP ERP to allow you to experience ABAP development workflow, to verify that your account is set up properly, and to create your first ABAP program.

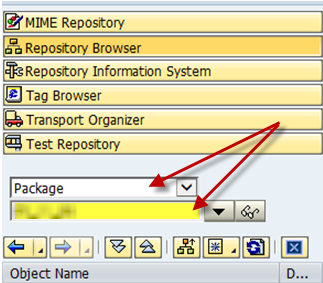
Start the SAPGUI and log in. Information on your User ID, the client, initial password, etc. can be found in an email sent to you prior to the first class.

**Task 1: Create a new package for ABAP Development**

Start the ABAP Development Workbench by executing transaction code **SE80**.

In the left screen region select the Repository Browser button (if it is not selected by default). This tool allows access the SAP development repository, which is where all programs and other programming objects reside. In the dropdown menu (shown below) select “**Package**”. If opening a pre-existing package, we could enter its name or search for it using the field below the dropdown menu. We want to create a new package for this lab exercise, so we will enter the name of this package in that field. For this class, package names, programs, and various other elements should always start with **#####\_** where the ##### is replaced by your login name *with the dash omitted*. This naming convention allows us to conform to the general requirements of all SAP development as well as making sure in this class that our work does not inadvertently collide with another developer.

Create a package named **#####\_PKG01** by typing that into the field under the package selection.



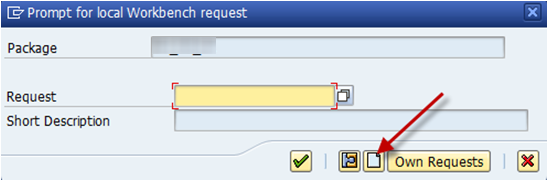
After entering the package name, press **enter**. The system will display a message indicating that this package does not exist and asking whether the package should be created. Choose “**Yes**”.

**Note**: If this package had been previously created, any items created within this package would have been retrieved and displayed in the Object Navigator after pressing enter. Since the package is a new creation, additional information needs to be supplied.

In the Create Package dialog, enter the **short description** “<lastname> package for ABAP development” where <lastname> is replaced by your last name. Leave all other fields as they are. Click the **continue (enter) button** in the lower right to create the package.

**Task 2: Create a transport request for your package**

The system places every package into a transport request (or simply a “transport”). You are prompted to specify which transport is to be used. As you do not currently have a transport request, click the ‘**Create Request’** button to create a new request.

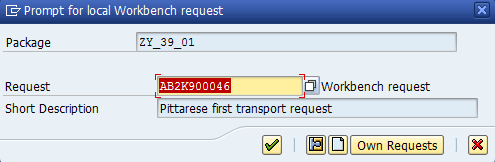


**Note:** In many corporate development processes, transport requests are created by system administrators. Developers therefore do not create new requests, but rather use ones already created. In this situation, you would enter or search for the appropriate request using the search option that is to the right of the Request field.

**Note:** If you wanted to see other transport requests you created previously, the “Own Requests” will show your active requests in the system.

To create a transport request, you must supply a **short description**. As your description use “<lastname> first transport request” where your last name is substituted in place of <lastname>. Leave all other fields as they are. Click the save icon in the lower right.

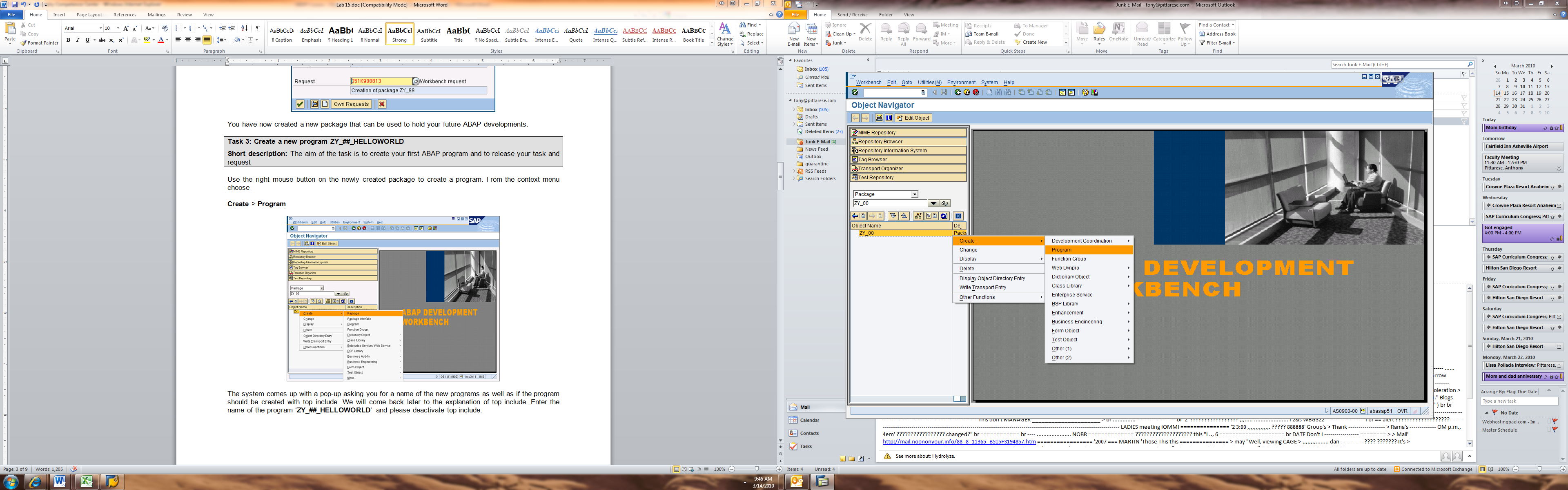
After you hit ‘**Save**‘ you will see the number of your newly created transport request supplied in the dialog box. The request number is assigned automatically by the system. Continue by pressing the **green check mark**.



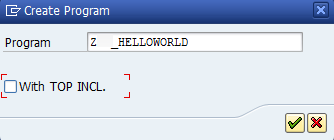
You have now created a new transport request and placed your development package within that transport request.

**Task 3: Create a new program**

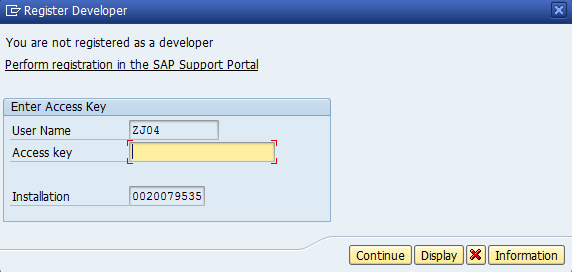
Right click on the package name (as shown below) and select **Create > Program**



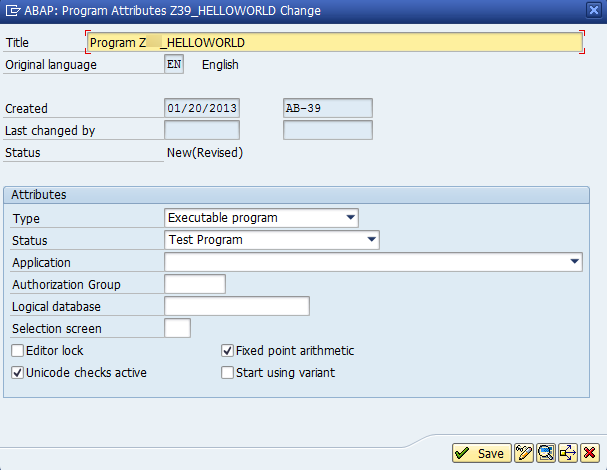
The system displays a create program dialog box. Enter the name of the program ‘**#####\_HELLOWORLD**’ (remember our naming convention). Make sure the Create with TOP include is **not** checked. Click the Continue icon or press enter.



As you are requesting the creation of a new program, and your user account is not yet registered in the system as a developer, you will be asked to supply an access key. Enter the key you have been provided (see the email sent to you previously where you can copy your developer key and paste it here) and select **continue**.



In the next step you have to define the program attributes. You should see where the system has filled in the Title based on your program name. Leave that as it is. Select ‘**Test Program’** as the status of the program. The rest of the attributes can be left as they are. Use the button ‘**Save**’ to continue.



In the next step the SAP system asks for a package. Usually the SAP system defaults to displaying the package you most recently created. Verify the SAP system automatically filled in the correct package name created previously (it should be #####\_PKG01). **Click Save.**

The SAP system now asks for a transport request. This was previously created and so is supplied by the system. Click the check button to continue.

The SAP system now provides you a program workspace.

When creating programs in the future, you will not have to go through all the steps in this process. You can reuse packages. (It is common for students to put all their programs for this class in one package.) Given that we are not going to be actually transporting out programs (except in this lab), you can reuse the same transport all semester. Lastly, the developer key only has to be entered one time.

At the top of the workspace, the first few lines are comment lines. The first actual line of code is generally found on line 6 and begins with the word REPORT followed by the name of the program.

On the line immediately after the REPORT line, enter the following (**with capitalization, etc. exactly as shown**). Replace <fullname> with your full name:

**write ‘Hello World from <fullname>!’.**

Pay attention to the dot at the end of the line as this indicates the end of the line of code. The dot in ABAP is like the semicolon in PHP, Java, and C#.

Save your new program using the top menu bar path **Program > Save** (or use the disk icon in the menu bar). Check your program for any syntax errors. Checking can be done via menu path **Program > Check > Syntax** (or use the balance scale icon in the menu bar). The system will respond with a message in the lower left taskbar.

**Task 4: Activate and Transport your program**

After creating a program, the finished program must be activated. Activation in this context means updating the SAP repository (where programs are stored in the database) and checking in the new version of the program. Activate the program via **Program > Activate** (or use the toolbar icon for activation).

To test your new program, use the top menu path **Program > Execute > Direct Processing** (or use the wrench icon in the toolbar). The result should show your first program. **Take a screen shot of your program output and paste it into a Word document for lab submission.** To get back to the ABAP Development Workbench, use the ‘**Back**’ button.

At this point other system users could now run your program. To see an example of this, open a new SAPGUI instance by clicking the New GUI Window icon in the toolbar. Execute transaction SA38. As the program name, enter ZN048\_HELLOWORLD and click the Execute button.

For our class development work we will not transport our programs as there is not a real testing or production system in use for us to transport into. However, for this lab exercise you will perform a transport to see how this would be done by a system administrator.

We are now ready to release the program for transportation to other SAP systems. In the same SAPGUI instance you executed ZN05\_HELLOWORLD navigate back to the main SAP Easy Access menu. Execute the transaction below.

**Tools >** **ABAP Workbench > Overview > Transport Organizer**

You may also use the transaction code **SE09** for direct access.

Type in your username into the **User field** (if it is not already present) and **deselect the Customizing Request checkbox** (only Workbench Requests should be selected). Hit **Enter** or press the green check to load your transports.

The Transport Organizer shows all your transport requests. You should see your transport request appear in the hierarchy (“<lastname> first transport request”). If it is not already expanded, expand the node for your transport by pressing the plus in a folder symbol that appears to its left. Under that node you will see a node for Development/Correction. Open/expand that node. Open/expand the Program node. You will see that the transport contains only one program which is named #####\_HELLOWORLD. In a real-world situation, a request might contain many tasks from different developers.

**Take a screen shot showing your transport and paste it into your Word document for lab submission.**

You will now transport your request. This is a two-step process. Find the line on your screen that corresponds to your transport request (it should be labelled “<lastname> first transport request”). The Development/Correction node inside your transport request (the line immediately below it) must be released first. Right mouse click on this Development/Correction node and select ‘**Release Directly’** from the context menu. After the above is released you will see a message in the lower left status bar. Now, click on the node <lastname> first transport request, right click, and release this directly.

The SAP system now creates the export of the transport request so that the developments are available to other SAP systems. This may take a few seconds.

**Take a screen shot showing that your work has been successfully transported and paste it into your Word document for lab submission.**

Click the refresh button at the top left of the screen. The screen should be empty except for a header line since all of your programs have been transported.

**Take a screen shot showing that your work has been successfully transported and paste it into your Word document for lab submission.**

**Task 4: Modify your program**

If you followed these instructions precisely, you should be working in one instance of the SAPGUI and have another instance that is displaying your program. Return to your other open session where your program is being shown.

If you closed development session, start the ABAP Workbench from the SAP Easy Access Menu by using transaction code **SE80**.

Open your program double clicking on the program name in the left tree. The SAP system opens the program now in the display mode. If necessary, switch to change mode (the screen text at the top indicates if you are in display or change mode).

**Hint:**

Generally, there are two modes: **display mode** for only having a look at a program and the **change mode** for changing a program. You can switch between both modes by using the little pencil/eyeglasses symbol: .



Save your existing program (without making changes to it). Notice that the SAP system asks you for a new transport request! This is because you have released the original transport request and so this program revision must be placed into a new/different transport request. Follow the same process as given previously to create a new transport request. Give this transport request the description “<lastname> work for Enterprise Programming” where <lastname> is your last name.

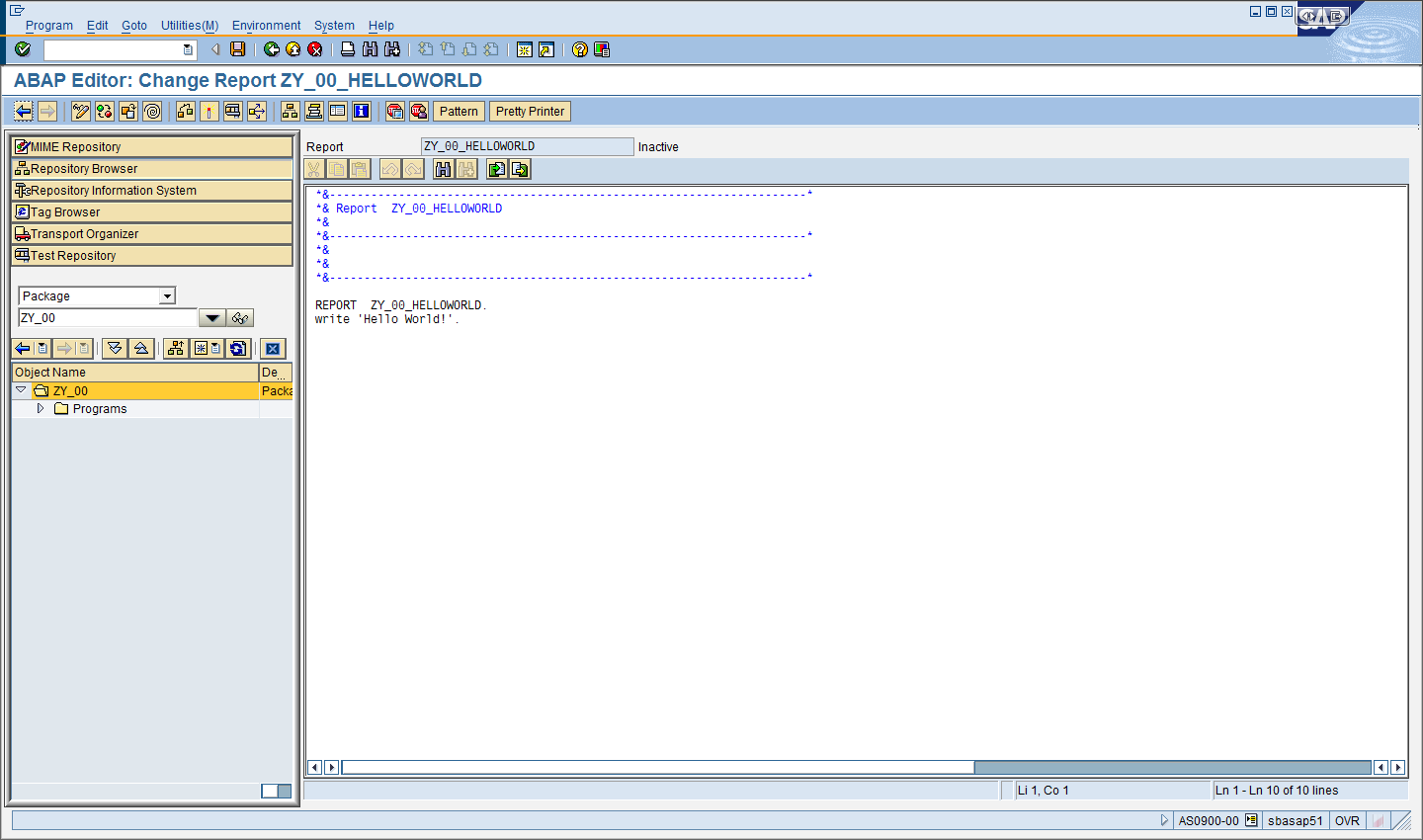
When the request is displayed, press enter or click the green check.

You are free to create a new package or transport whenever desired. For packages, make sure they adhere to our naming convention. It is very common, however, for students to use one package for all their coursework. You are welcome to use this current package for that purpose.

We will now configure the Pretty Printer. From the top menu bar choose **Utilities > Settings.** If prompted that the system is attempting to retrieve the environment variable select “Remember My Decision” and then choose “Allow”.

Select the ‘**ABAP Editor**’ top tab and navigate to the ‘**Pretty Printer**’ lower tab. Activate the Indent and the Convert Uppercase/Lowercase checkboxes, Select the ‘**Uppercase**’ radio button. This will have the Pretty Printer set all ABAP code except for string literals in uppercase. Click the check button to activate these settings.

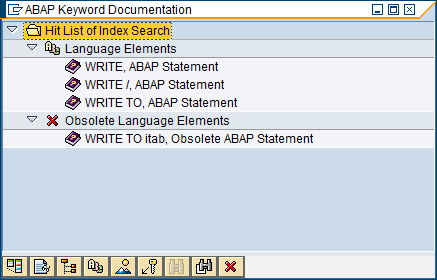
Go back to the Object Navigator and test the Pretty Printer by clicking on the ‘**Pretty Printer**’ button in the toolbar.



You will see the result of the Pretty Printer work instantly. The Pretty Printer has changed all program text (except for literal strings inside quotes) to uppercase. This is a common ABAP coding convention.

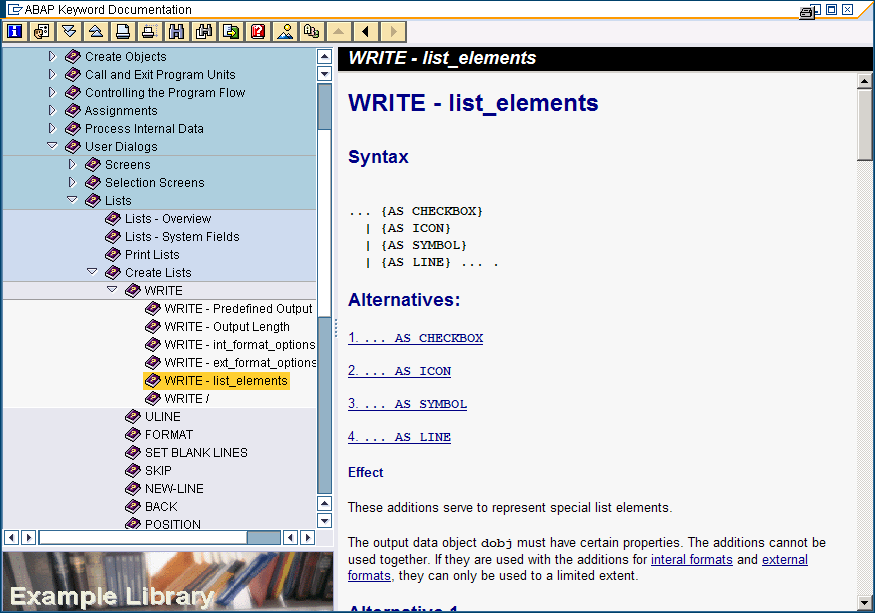
**Task 5: ABAP help**

In this task we want to use ABAP help to gain knowledge about the write instruction. ABAP keywords have many features and options. The built-in ABAP help is often the best way to learn what options are available. Help is activated by positioning the cursor on a keyword you wish to learn more about and pressing F1. Click one time on the keyword WRITE in your program to place the cursor on that word and then press the ‘**F1**’ button. (Note that as you work in the SAP GUI Editor you want to break yourself of the habit of using double clicking to select a word. Double clicking triggers a different activity in the ABAP editor.) The system will search for all entries in the ABAP help system related to the WRITE statement.



You see documentation related to different uses of the write statement—WRITE by itself, WRITE used with /, WRITE TO, and then an obsolete version of the WRITE TO statement used with internal tables. Double click **WRITE, ABAP Statement** line. The SAP system now opens the ABAP help system and jumps directly to the entry for the WRITE statement. (The first time you run this, the system may take a while to load this up as help is being initially indexed.)

The display provides information on all the optional statement elements (contained in square brackets] that can be used with WRITE. It also provides documentation ‘extras’ on common uses.

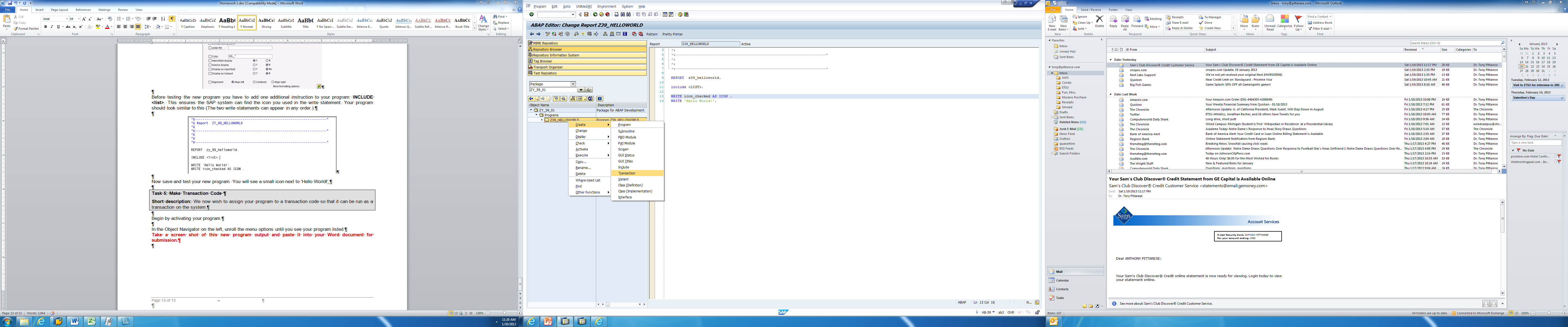


Close the documentation window and return to the ABAP editor.

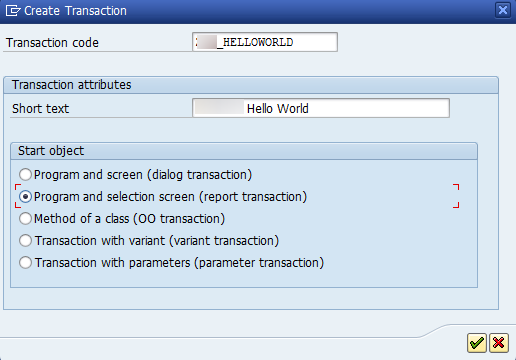
**Task 6: Make your program runnable by Transaction Code**

Begin by activating the current version of your program.

If necessary, unroll the package displayed in the Object Browser on the left until you see the name of your program listed. (See screen shot below.) Right click on your program name and select Create 🡪 Transaction.



Enter #####\_HELLOWORLD as the transaction code. Enter “<lastname> Hello World” as the short text, where <lastname> is your last name. Choose to make this a report transaction. Click the continue check mark.



In the Create Report Transaction screen that appears, enter the name of your program in the program field (#####\_HELLOWORLD). Choose the save icon from the menu bar and select save and continue until you are back at the main screen.

Press back until you are at the easy access menu. Enter your newly created transaction code in the command field and verify that your program executes.

**In the Word document you created, place the name of your transaction code at the end of the document. Save and upload this to the Lab 1 D2L dropbox by the due date posted in D2L.**