# Overview

In this exercise you will modify the **Orders.dbv** program that you created in [Exercise – 1](http://jobfunc2.cu.net/Job%20Functions/Programmer/Programmer%20Handbook/Synergy%20Training%20-%20Synergy%20Language%20Essentials/Exercise%20-%201.docx). You will extend the functionality of the application to display a main menu, from which the user may select a required option. The menu will be displayed in a Synergy Window. Currently, we have the code required to create Synergy Windows and accept key inputs in our application, but the functionality is embedded in the DisplayMessage subroutine (**DisplayMessage.dbl**). To avoid duplicating this code, you will first “encapsulate” some of the existing functionality into three new external subroutines. Having done this, you will be able to use that functionality easily whenever you need it.

# Resources

* [Synergy Best Practices - Coding Standards](http://jobfunc2.cu.net/Job%20Functions/Programmer/Programmer%20Handbook/Tims%20Best%20Practices%20-%20Standards/Synergy%20Best%20Practices%20-%20Coding%20Standards.docx)

# Exercise

1. Using Visual Studio, open the previously created “Orders” project.
2. Using Visual Studio, create a new file.
3. Create a new external function called CreateWindow. Code the function to create a Synergy Window, and return the Window ID of the new window to the calling routine. The subroutine should have the following arguments:

**Name** Name of the window to create (a).

**Height** Height of the window (n).

**Width** Width of the window (n).

**Row position** Screen row for window placement (n).

**Column position** Screen column for window placement (n).

**Title** Window title (Optional) (a).

1. Save the new file as **CreateWindow.dbl**.
2. Add **CreateWindow.dbl** and **DisplayMessage.dbl** to your project.
3. Modify the DisplayMessage subroutine (**DisplayMessage.dbl**) to use the new CreateWindow function.
4. Compile, link, and run the program.
5. Using Visual Studio, create a new file.
6. Create a new external subroutine called DisplayText. Code the subroutine to display a simple text message at a specified position in a specified Synergy Window. The subroutine should have the following arguments:

**Window ID** Window ID of the window to receive the text (n).

**Row** Row position for the text display (n).

**Column** Column position for text display (n).

**Text** The text to be displayed (a).

1. Save the new file as **DisplayText.dbl**. Add the file to your project.
2. Modify the DisplayMessage subroutine (**DisplayMessage.dbl**) to use the new DisplayText subroutine.
3. Compile, link, and run the program.
4. Using Visual Studio, create a new file.
5. Create a new external subroutine called ClearBuffer. Code the subroutine to clear any pending input from the terminal buffer. The subroutine should have the following arguments:

**Window ID** Window ID of the window to be used for input (n).

1. Save the new file as **ClearBuffer.dbl**. Add the file to your project.
2. Using Visual Studio, create a new file.
3. Create a new external subroutine called WaitKey. Code the subroutine to accept a single keystroke at a specified position in a specified Synergy Window, and return the key pressed to the calling program. The subroutine should have the following arguments:

**Window ID** Window ID of the window to be used for input (n).

**Row** Row position where input should take place (n).

**Column**  Column position where input should take place (n).

**Character** Key pressed (returned to calling routine) (a).

1. Save the new file as **WaitKey.dbl**. Add the file to your project.
2. Modify the DisplayMessage subroutine (**DisplayMessage.dbl**) to use the new WaitKey subroutine.
3. Compile, link, and run the program.
4. Using Visual Studio, create a new file.
5. Create a new external subroutine called DeleteWindow. Code the subroutine to delete a specified Synergy Window. The subroutine should have the following arguments:

**Window ID** Window ID of the window to be deleted (n).

1. Save the new file as **DeleteWindow.dbl**. Add the file to your project.
2. Modify the DisplayMessage subroutine (**DisplayMessage.dbl**) to use the new DeleteWindow subroutine.
3. Compile, link, and run the program.
4. Using Visual Studio, open the **Orders.dbv** source file.
5. Modify the pProcess routine. Create a new window for the main menu, the window should have a title of “Order Processing Menu”.
6. In the window, display options for “Add and order”, “View an order”, and “Exit”.
7. In the menu window, display a message that prompts the user to select one of the available options.
8. Use the WaitKey external subroutine to prompt the user for a menu entry.
9. Decode which menu entry was selected, and display an appropriate message using the DisplayMessage external subroutine.
10. Continue processing this menu until “Exit” is selected then close the program.

# Discussion

This exercise is significantly more complicated than earlier ones. Complete the exercise one step at a time and test the application as you go. Make sure you complete each step full before moving on to the next step.