# Overview

To become familiar with repository concepts and basic CU Toolkit shell programs.

# Resources

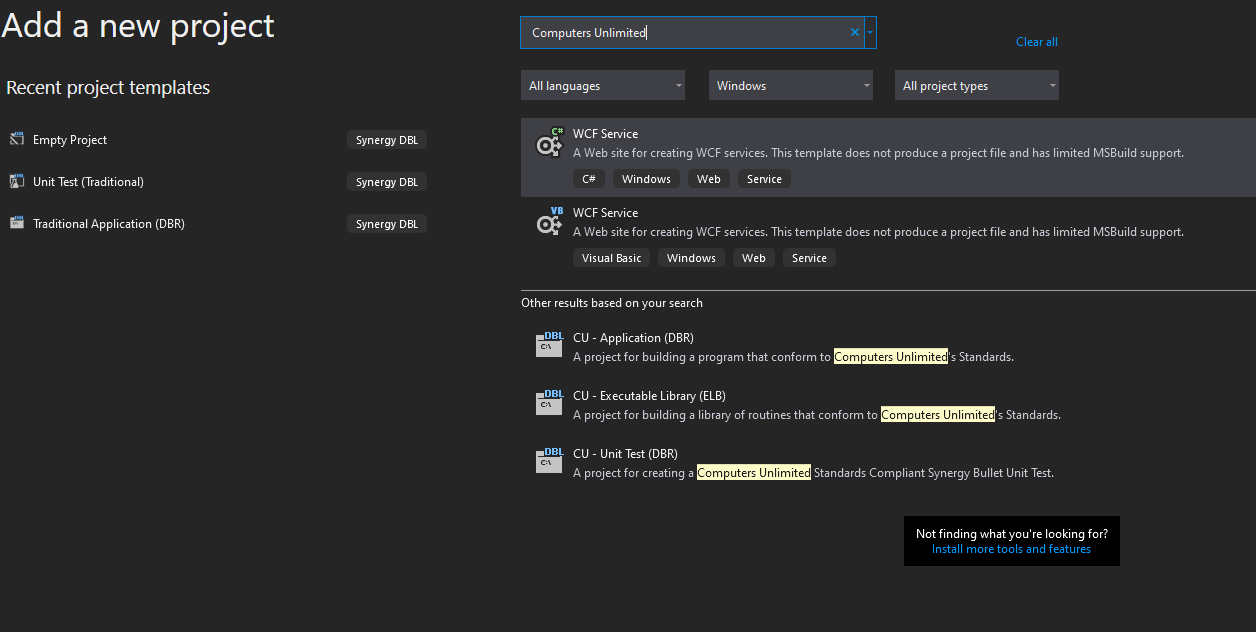
* [Synergy DBL Language Reference](https://www.synergex.com/docs/index.htm)
* [Synergy Best Practices - Coding Standards](http://jobfunc2.cu.net/Job%20Functions/Programmer/Programmer%20Handbook/Tims%20Best%20Practices%20-%20Standards/Synergy%20Best%20Practices%20-%20Coding%20Standards.docx)
* [Traditional Synergy in Visual Studio - CU Wiki](http://echo.cu.net/cuwiki/Traditional_Synergy_in_Visual_Studio)
* [Traditional Synergy in Visual Studio Common Terminology - CU Wiki](http://echo.cu.net/cuwiki/Traditional_Synergy_in_Visual_Studio_Common_Terminology)
* [Installing Traditional Synergy in Visual Studio Templates - CU Wiki](http://echo.cu.net/cuwiki/Installing_Traditional_Synergy_in_Visual_Studio_Templates)
* [Creating a New DBR Project (TSVS) - CU Wiki](http://echo.cu.net/cuwiki/Creating_a_New_DBR_Project_(TSVS))
* [Debugging (TSVS) - CU Wiki](http://echo.cu.net/cuwiki/Debugging_(TSVS))
* [CU Toolkit Manual](http://jobfunc2.cu.net/Job%20Functions/Programmer/Programmer%20Handbook/Synergy%20Reference%20-%20CUToolkit%20Reference%20Manual.doc)
* [Getting Started With Repository](https://www.synergex.com/docs/versions/v111/index.htm#rps/1_WELCOME_RPS.htm)
* [Synergy Data Language](https://www.synergex.com/docs/versions/v111/index.htm#rps/6_SDL.htm)

# Exercise

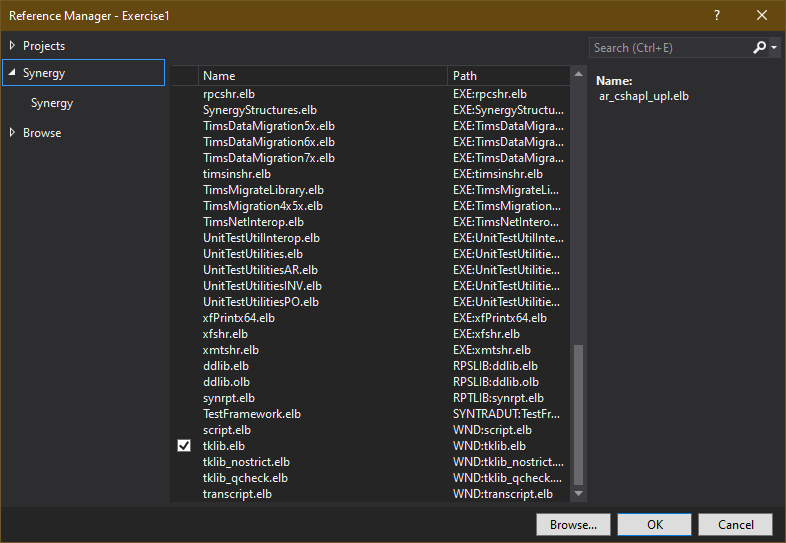
For this exercise you should complete the following steps in the order shown:

1. Create a “Training” folder under the “Synergy” folder of your local source tree.
2. Create a “CU Toolkit” folder under the “Training” folder.
3. Open Synergy\TIMS.ELBs.sln.
4. Add a new [CU – Application (DBR)](http://echo.cu.net/cuwiki/Creating_a_New_DBR_Project_(TSVS)) project “Orders” in \Synergy\Training\CU Toolkit”.

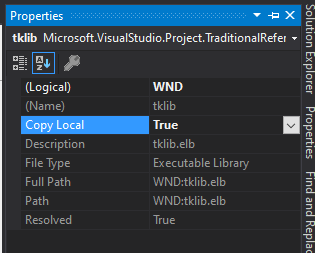
**Note:**  You will need to search for “Computers Unlimited” in the “Add a new project” dialog:



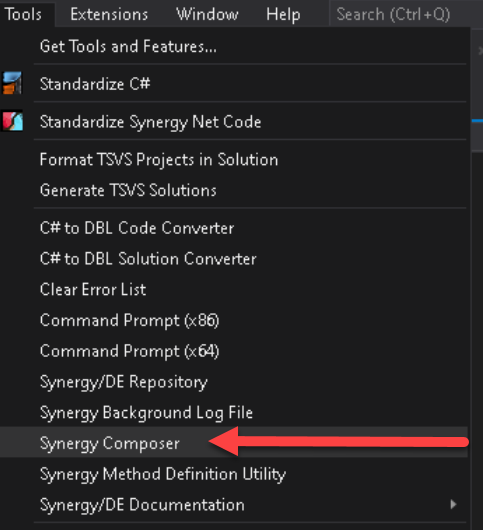
1. Due to current limitations in Visual Studio, the name of the project will always be Program. To work around this you need to select the project and Hit F2to rename the project.
2. Rename the project to “DemoMaintenance”.
3. Perform these same steps for the Program.dbv file which was generated for you.
4. Rename the file to “DemoMnt.dbv”.
5. Now open the Project Properties (Hotkey Alt+Enter).
6. Change the Output name under the Application Tab to “DemoMnt”.
7. Set “DemoMaintenance” as the startup project.
8. Add a reference to “tklib.elb” to the “DemoMaintenance” project:



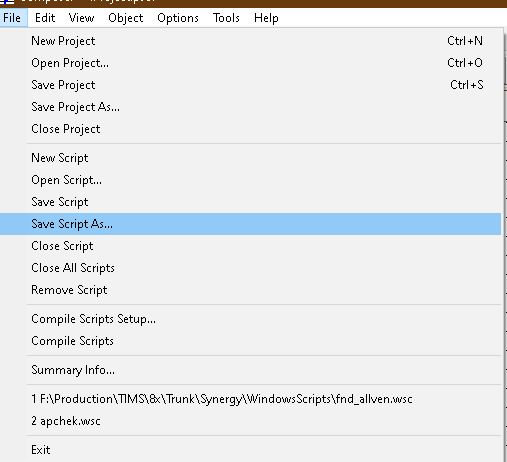
1. Open “Properties” of “tklib.elb” and set “Copy Local” to “True”:



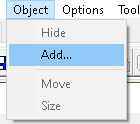
1. Add a “Projects” reference for:
   * “ComputersUnlimited.Utilities”
   * “CUToolkitInputLibrary”
   * “DataAccessLibrary”
   * “Repository”
   * “SynergyStructures”
2. Add the “DemoMas.schema” to the Repository project.
3. Using Visual Studio (or the text editor of choice), modify the repository schema “DemoMas.schema”:
   * Add appropriate prompts to all fields.
   * Make the “Active” field a checkbox.
   * Save the file.
   * Compile the schema using: “**…Synergy\BuildSynergyWindowsLibrary.bat**”
   * Rebuild the solution
4. Using Visual Studio, Tools, Synergy Composer:



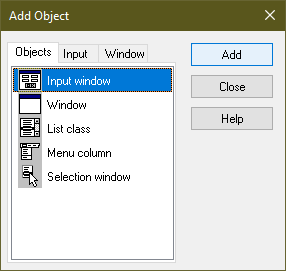
1. In Synergy Composer, File, Save Script As:



1. Save the script as **Synergy\WindowsScripts\DemoMnt.wsc**
2. In Synergy Composer, Object, Add:



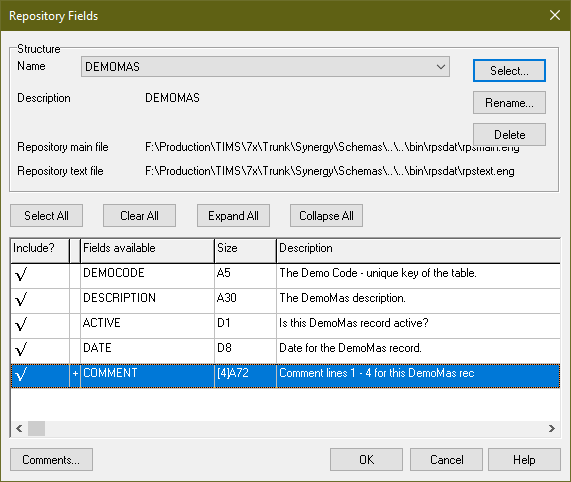
1. In Synergy Composer, Add Object, add an Input Window:



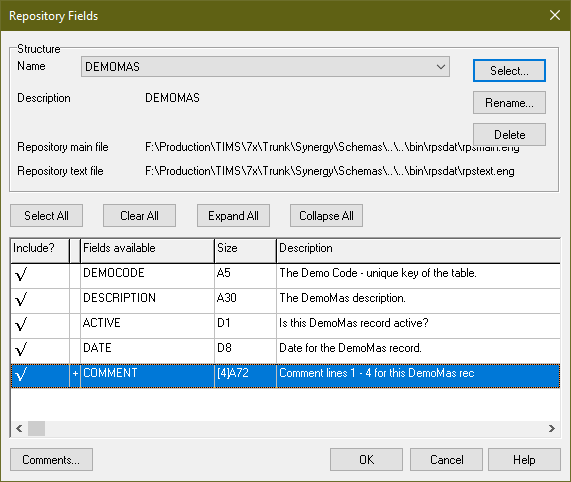
1. Select the “Long” tab in the “Properties” window and set the following:
   * Name: DemoMntG
   * Height: 8
   * Width: 74
   * Title: General
2. Save the script.
3. Close Synergy Composer.
4. Compile the script using: **Synergy\BuildSynergyWindowsLibrary.bat**
5. Using Visual Studio (or the text editor of choice), open “Demo.wsc” and add the following to the DemoMntG input window (after the **.title** key word):

**.repository\_structure** DemoMas

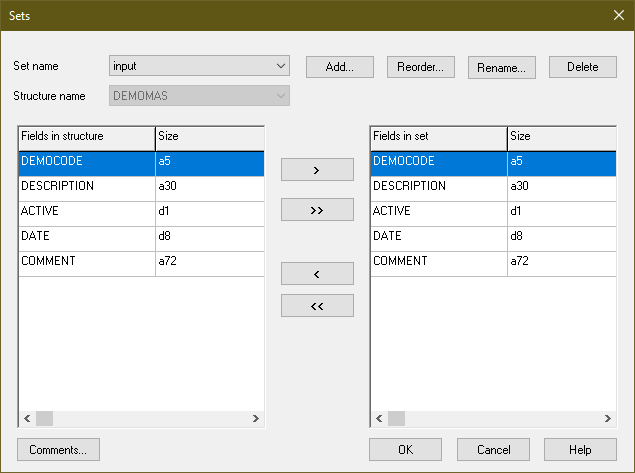
1. Save the script.
2. Compile the script using: **Synergy\BuildSynergyWindowsLibrary.bat**
3. Using Visual Studio, Tools, Synergy Composer, open the script file **Synergy\WindowsScripts\DemoMnt.wsc**
4. Select the “Long” tab in the “Properties” window and set the following:
   * Under Repository fields, hit the ellipsis (…) and add all fields in the DemoMas structure to the DemoMntG input window:



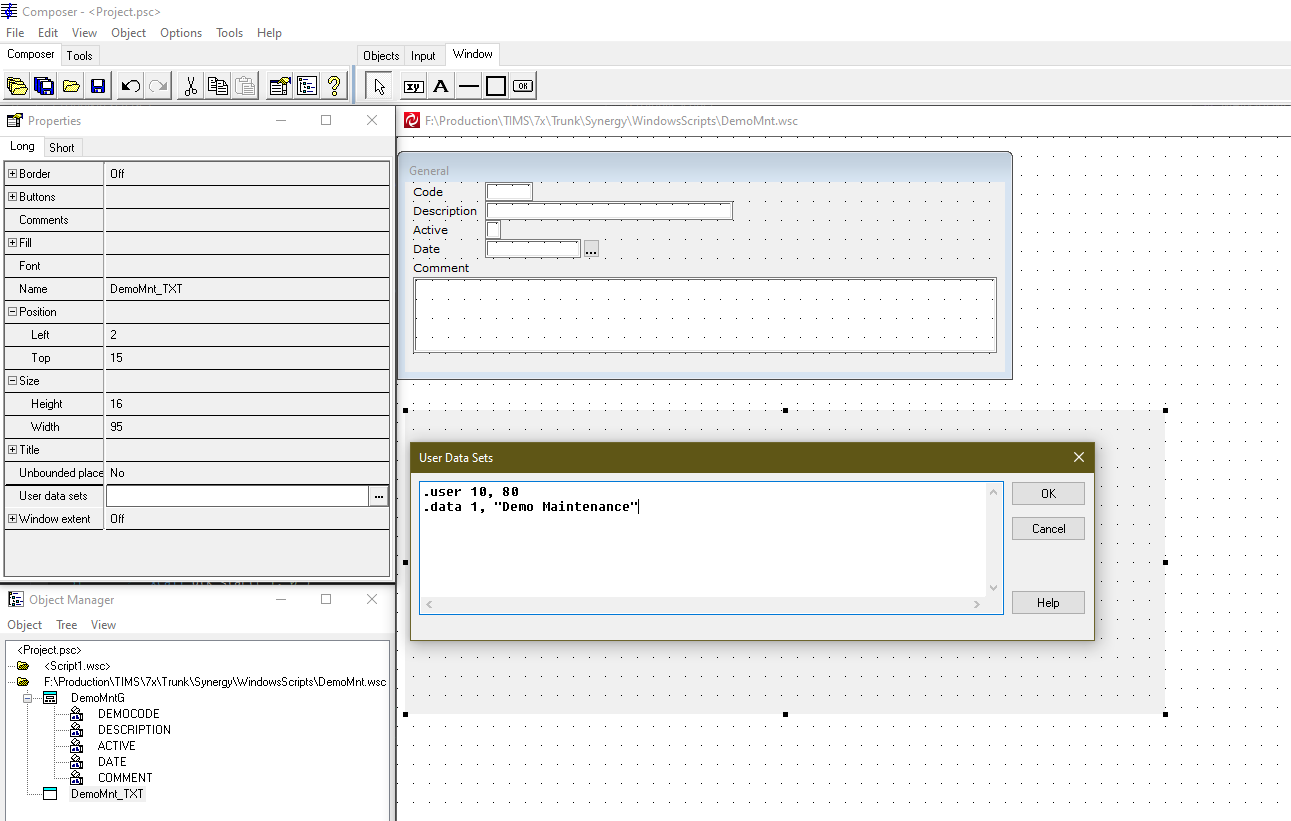
* + Under Sets, hit the ellipsis (…) and add a set named “Key” that contains the DemoCode field:



* + Under Sets, hit the ellipsis (…) and add a set named “Input” that contains all field:



1. Select the “Long” tab in the “Properties” window and set the following:
   * Border: Off
   * Name: DemoMnt\_TXT
   * Under User data sets, hit the ellipsis (…).
   * In the User Data Sets dialog add the following:
     + .user 10, 80
     + .data 1, "Demo Maintenance"



1. Make the “Active” field a checkbox.
2. Save the script.
3. Compile the script using: **Synergy\BuildSynergyWindowsLibrary.bat**
4. Using Visual Studio, open “DemoMnt.dbv”.
5. Add the following import statements at the physical top of the “DemoMnt.dbv” file:
   * import ComputersUnlimited
   * import ComputersUnlimited.DataAccessLibrary
   * import ComputersUnlimited.Structures
6. Add code using CU Toolkit commands to create a single tab maintenance program similar to the following:

**import ComputersUnlimited**

**import ComputersUnlimited.DataAccessLibrary**

**import ComputersUnlimited.Structures**

**.include "DemoMas" repository, structure = "sDemoMas", end**

**;;; <summary>**

**;;; Synergy Training: CU Toolkit - Exercise 3**

**;;; </summary>**

**main DemoMnt**

**.include "tims.def" library "cu\_records"**

**.define MAX\_DATA\_OBJECTS ,1**

**.define MAX\_TABS ,1**

**.define MAX\_WINDOWS ,1**

**.define TABSET\_COLUMNS ,74**

**.define TABSET\_ROWS ,10**

**.align**

**record WorkVars**

**hprogctl ,int**

**mDemomas ,sDemoMas**

**endrecord**

**proc**

**xcall dtk\_start( 1, 0 )**

**hprogctl = %cue\_maintCtl(**

**& "DemoMnt",**

**& mDemoMas,**

**& MAX\_TABS,**

**& MAX\_DATA\_OBJECTS,**

**& TABSET\_ROWS,**

**& TABSET\_COLUMNS )**

**xcall cuts\_create( hprogctl, "DemoMntG" )**

**xcall cutb\_maint( hprogctl )**

**while( ( G\_ENTNAM .NES. "CU\_EXIT" ) && ( G\_ENTNAM .NES. "CU\_CLOSE" ) )**

**begin**

**xcall cue\_process( hprogctl )**

**end**

**if( G\_ENTNAM .EQS. "CU\_CLOSE" )**

**begin**

**clear G\_SELECT**

**end**

**hprogctl = %cue\_destroyhprogctl( hprogctl )**

**xcall stop( "MENU"," " )**

**endmain**

# Discussion

This is obviously a complex exercise, and the key to success is to take it slowly and step by step. Don’t proceed to another section until you are satisfied that the current section is working.

There is no “correct approach” to input processing – if you find a better way to do something then go with it, so long as the results are similar to those discussed in the above text.