Carbide.c++ Tutorial 1 - Helloworld

This tutorial provides a basic Symbian C++ "hello world" example. It demonstrates all the steps necessary to develop a helloword GUI based Symbian C++ application and how to use UI designer. The example code is also available #Source Code.

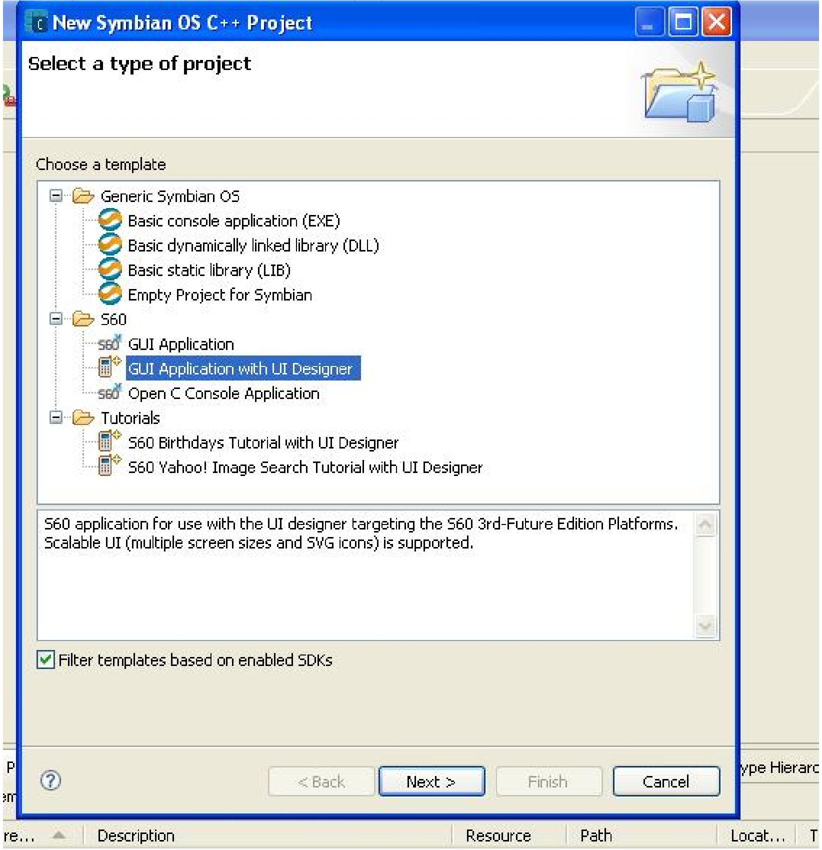
Introduction

Carbide.c++ is Nokia's IDE for developing application on various platforms, like Symbian C++, Open C/C++, Qt, etc. Symbian C++ development has really become easy (to good extent) after the invent of the IDEs. Carbide.c++ comes with a unique tool called as UI designer, allowing faster prototyping and deployment. This tutorial mainly deals with UI designer and it can be followed on all editions of Carbide.C++.

This tutorial assumes that S60 SDK and Carbide.c++ is already installed and setup. Check SDK and Carbide.c++ installation guide for more information.

Step 1 - Create new project

1. Start with a new Symbian C++ project using file menu of carbide c++.
2. Next select GUI application with UI designer as an option on next screen as shown in screenshot. This step enables the GUI editor for the project.



New Symbian OS C++ Project

Select a type of project

Choose a template

Generic Symbian OS
Basic console application (EXE)
Basic dynamically linked library (DLL)
Basic static library (LIB)
Empty Project for Symbian
S60
GUI Application
GUI Application with UI Designer
Open C Console Application
Tutorials
S60 Birthdays Tutorial with UI Designer
S60 Yahoo! Image Search Tutorial with UI Designer

S60 application for use with the UI designer targeting the S60 3rd-Future Edition Platforms. Scalable UI (multiple screen sizes and SVG icons) is supported.

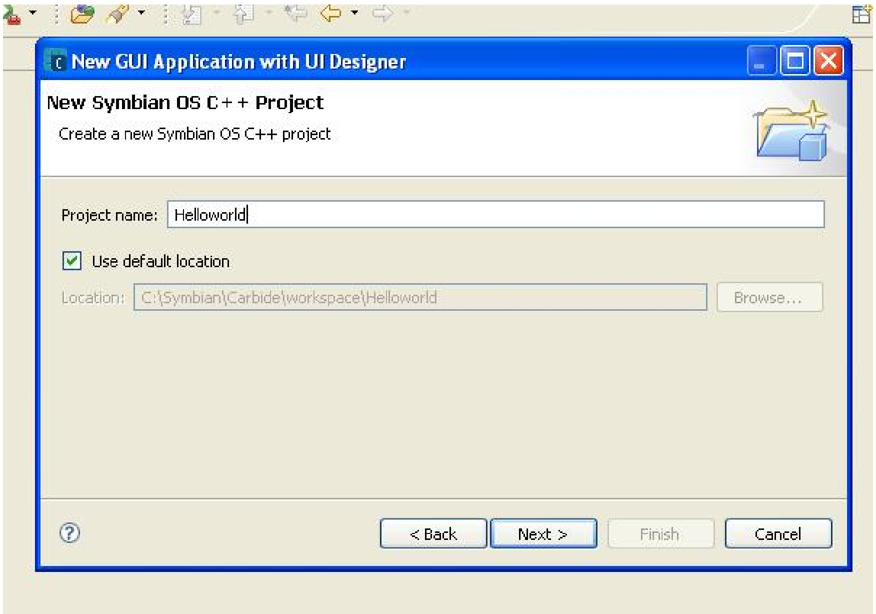
☑ Filter templates based on enabled SDKs

< Back
Next >
Finish
Cancel
Description
Resource
Path
Locat...

Step 2 - Enable UI designer

- Next enter the desired name of project. As a basic user, select the default location.

Once you get some expertise you can select any other workspace. Basically carbide.c++ manages the projects using workspaces. Workspace is a collection of projects. One can group relevant projects in one workspace. In carbide it is possible to switch between multiple workspaces. One can set the default location and can explore more there.



New GUI Application with UI Designer

New Symbian OS C++ Project

Create a new Symbian OS C++ project

Project name: Helloworld

☑ Use default location

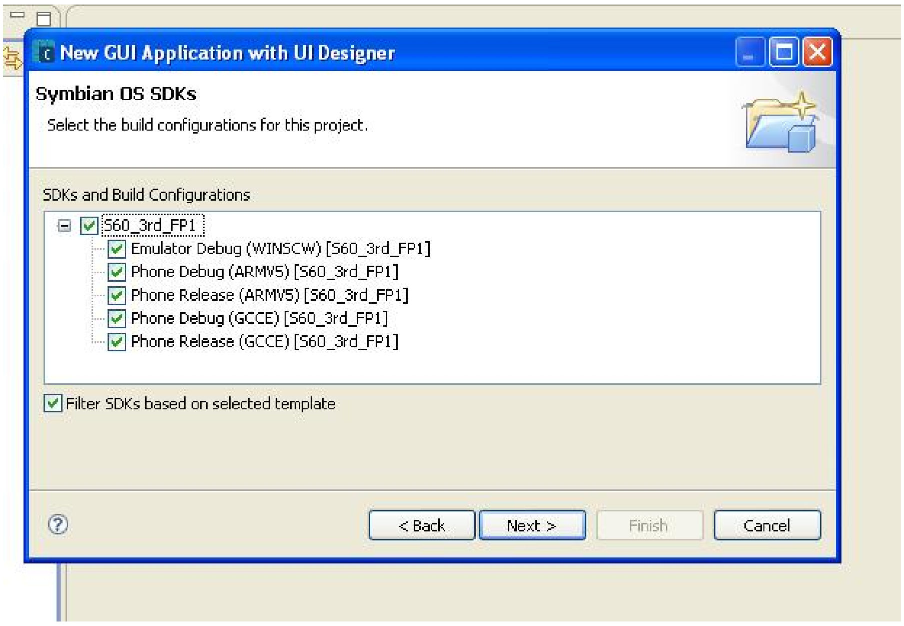
Location: C:\Symbian\Carbide\workspace\Helloworld

Step 3 - Select the build configuration

In the next step, the wizard asks to select the build configurations to be added for the project. Usually, by default, all the configurations are selected.

- For this project and for most of the projects, Emulator debug (WINSCW) and one Phone release configuration is required.

Once this is confirmed click on next.



New GUI Application with UI Designer

Symbian OS SDKs

Select the build configurations for this project.

SDKs and Build Configurations

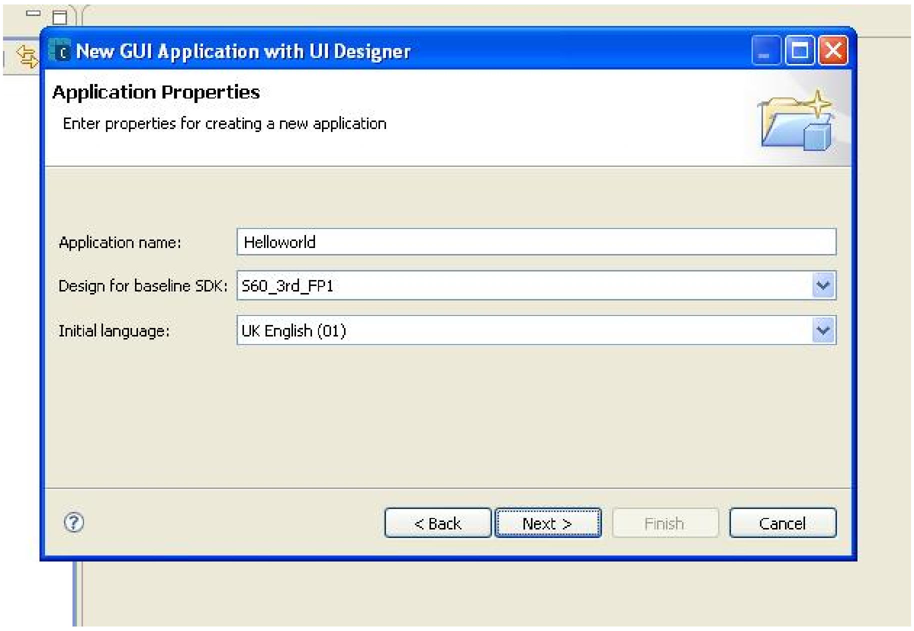
☑ S60_3rd_FP1
- ☑ Emulator Debug (WINSCW) [S60_3rd_FP1]
- ☑ Phone Debug (ARMV5) [S60_3rd_FP1]
- ☑ Phone Release (ARMV5) [S60_3rd_FP1]
- ☑ Phone Debug (GCCE) [S60_3rd_FP1]
- ☑ Phone Release (GCCE) [S60_3rd_FP1]

☑ Filter SDKs based on selected template

Step 4 - Application properties

In the next step, you will be prompted to enter few application properties.

- Enter the application name here as *Helloworld*. You can keep rest of the things as default.



New GUI Application with UI Designer

Application Properties

Enter properties for creating a new application

Application name: Helloworld

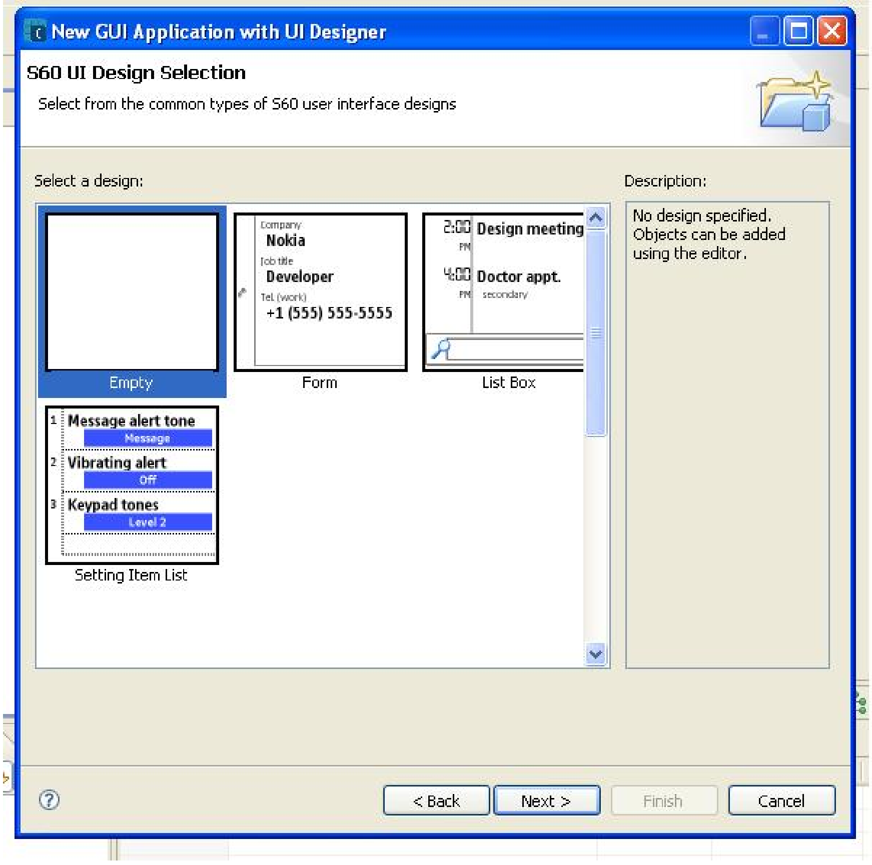
Design for baseline SDK: S60_3rd_FP1

Initial language: UK English (01)

Step 5 - Select the view type

In the next step, the wizard prompts to select a view type for the application.

- For our application select an empty container as we will not have any control in it.



New GUI Application with UI Designer

S60 UI Design Selection

Select from the common types of S60 user interface designs

Select a design: Empty

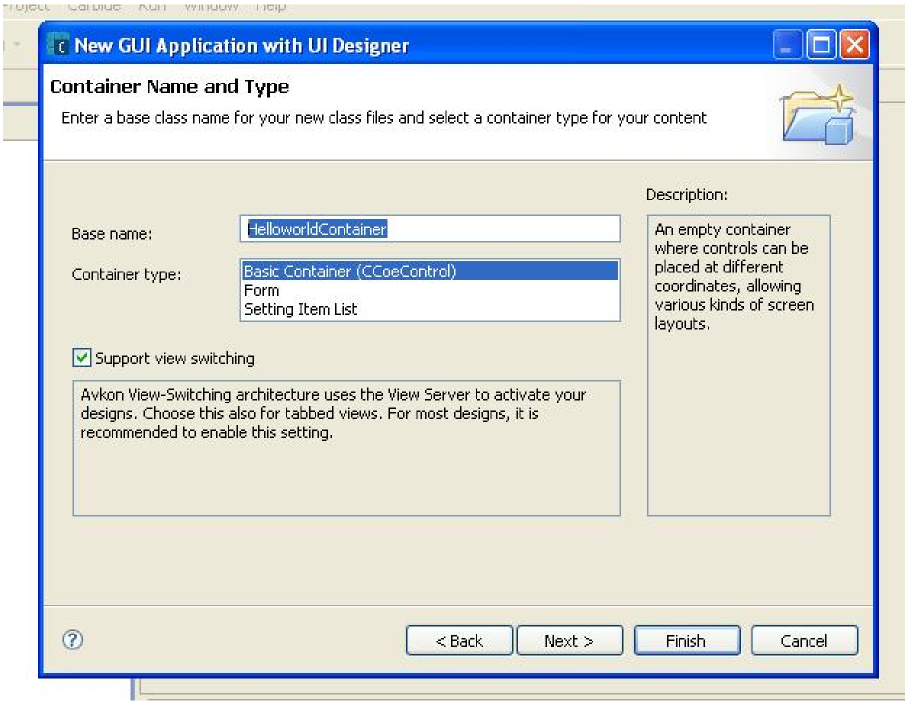
Description:

No design specified. Objects can be added using the editor.

-

Step 6 - Select name for the view

- The next step involves naming the container class and the type of the class, as shown in figure.
  One more important thing that is selected in this step is the support for view switching ability.



Container Name and Type

Enter a base class name for your new class files and select a container type for your content

Base name: HelloworldContainer

Container type:
Basic Container (CCoeControl)
Form
Setting Item List

Description:
An empty container where controls can be placed at different coordinates, allowing various kinds of screen layouts.

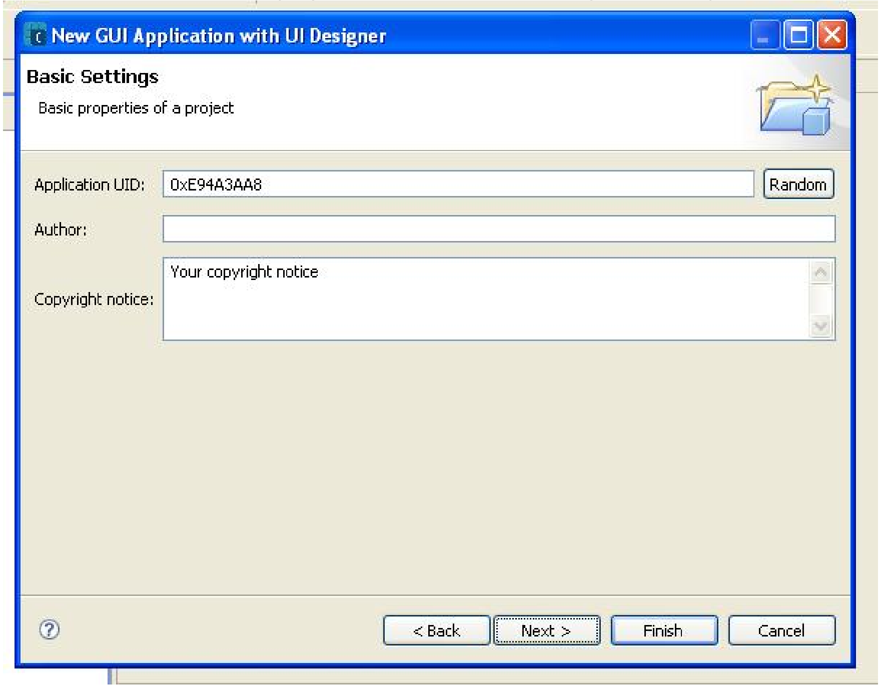
☑ Support view switching

Avkon View-Switching architecture uses the View Server to activate your designs. Choose this also for tabbed views. For most designs, it is recommended to enable this setting.

Step 7 - Some more properties

In the next step, we will need to input some more information like UID .

- If you are not aware of these concepts you can leave the defaults and go to next step.



Basic Settings

Basic properties of a project

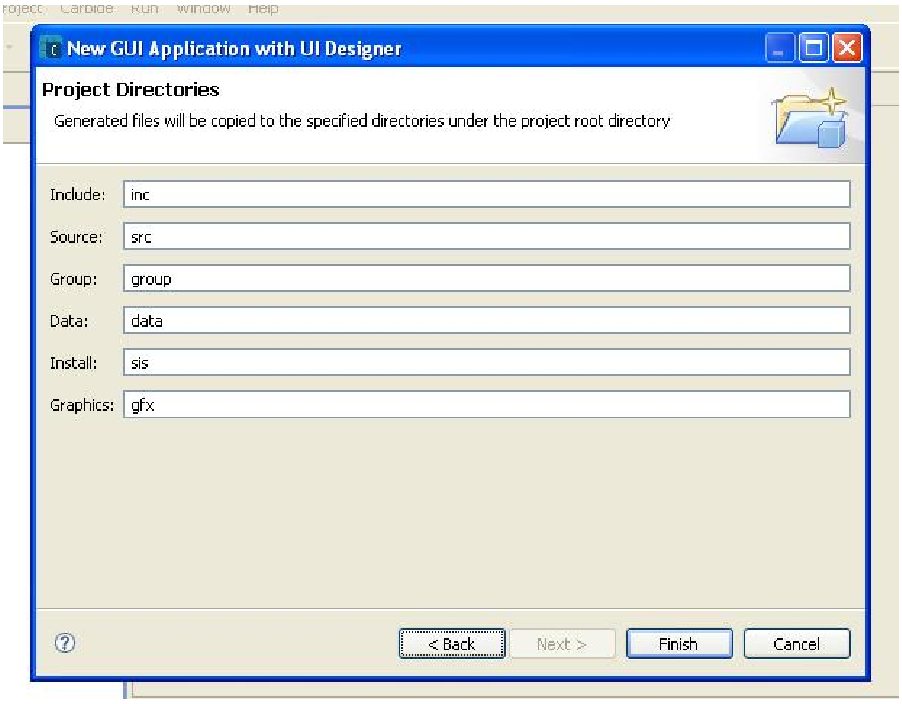
Application UID: 0xE94A3AA8

Author:

Copyright notice: Your copyright notice

Step 8 - Select / change project directories.

- Accept default suggestion for directories in the next step. Finish the wizard by clicking the finish button.



New GUI Application with UI Designer

Project Directories

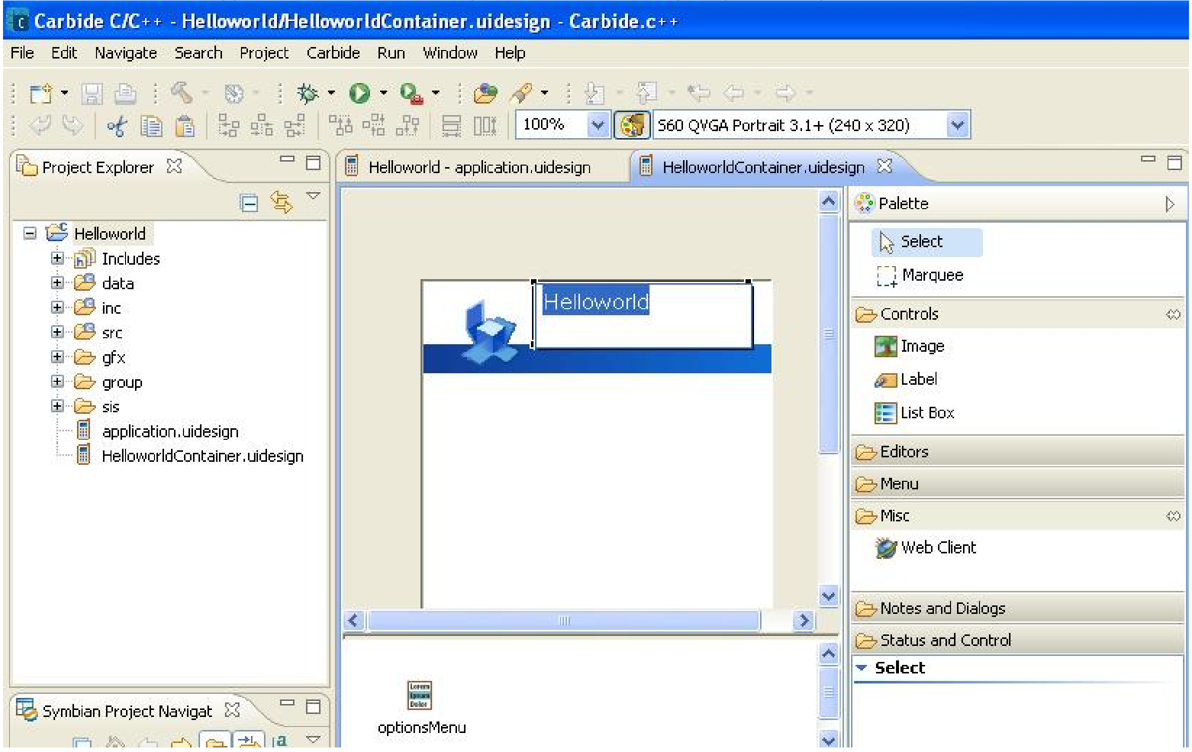
Generated files will be copied to the specified directories under the project root directory

Include: inc
Source: src
Group: group
Data: data
Install: sis
Graphics: gfx

Step 9 - Changing title of application window

Now we will have the UI designer window which showcases the application window in it.

- Double click the title of the application window and change it to *Helloworld* or anything else that you wish.

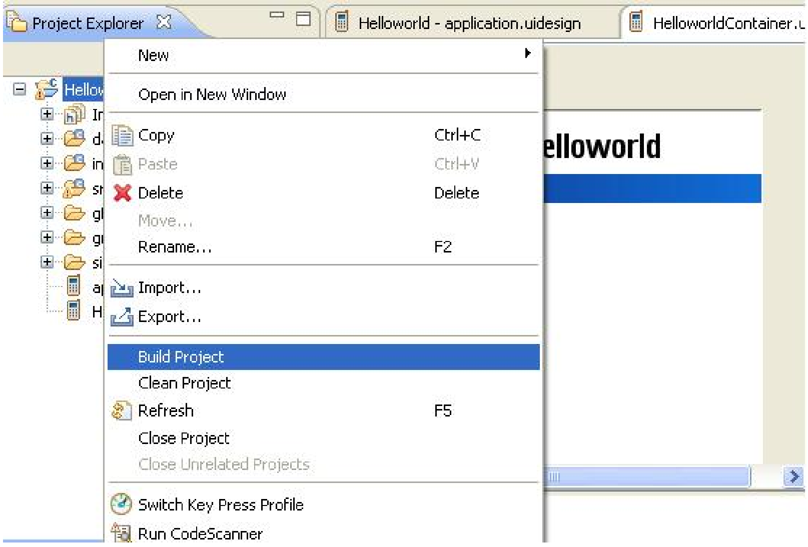


Carbide C/C++ - Helloworld/HelloworldContainer.uidesign - Carbide.c++

Step 10 - Building the application

Now following these steps, you can run your first Helloworld application.

1. Go to Project Explorer window right click your own project name.
2. From the popup menu select the *Build Project* option.
   If everything goes well you will have no errors in bottom output window.

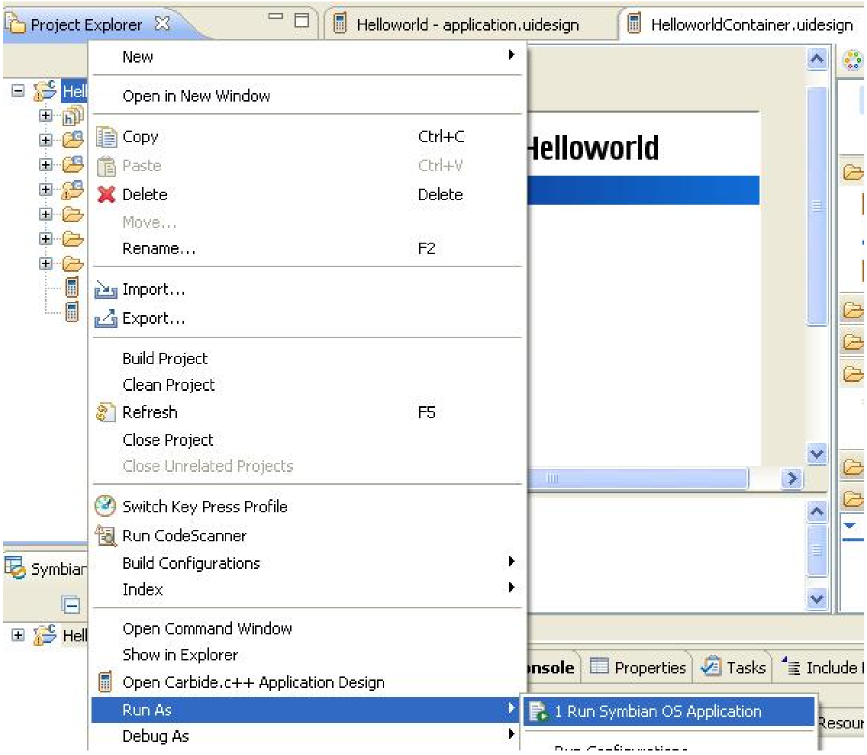


Step 11 - Running the application

Once the build is successful, run the application with the help of pop up menu.

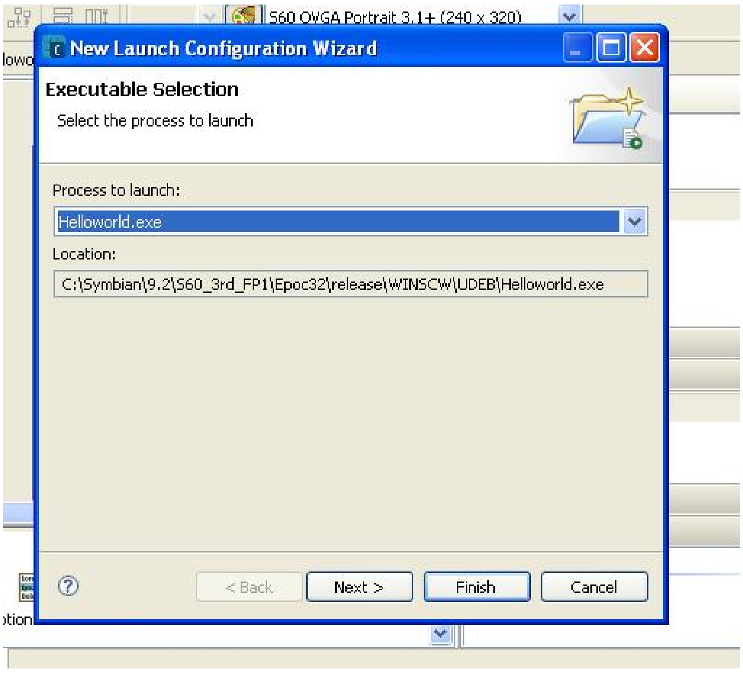
1. To get the popup, right click on the project as show in the figure.

One this is done, Carbide launches the emulator and the application is started in the emulator.

1. 

Step 12 - Emulator output

The application usually starts in the emulator (S60 3rd FP2 onwards), once run option is selected. If not then it can be started manually from installation menu in the emulator. When you start to run the application it asks for the process to run make sure that it is name of the process is the one which we want to run.


S60 QVGA Portrait 3.1+ (240 x 320)
New Launch Configuration Wizard
< Back
Next >
Finish
Cancel

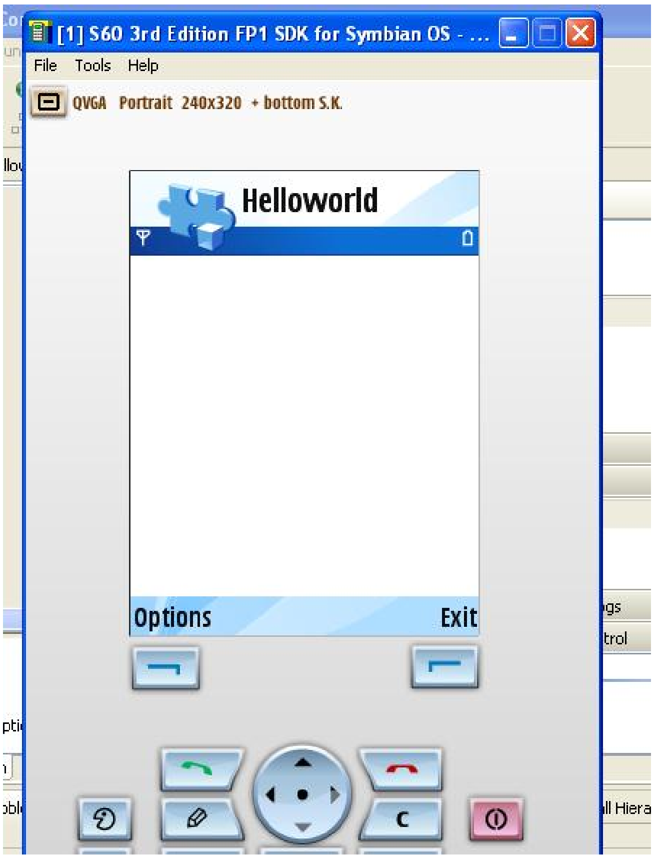
Executable Selection

Select the process to launch

Process to launch: Helloworld.exe

Location: C:\Symbian\9.2\S60_3rd_FP1\Epoc32\release\WINSCW\UDEB\Helloworld.exe

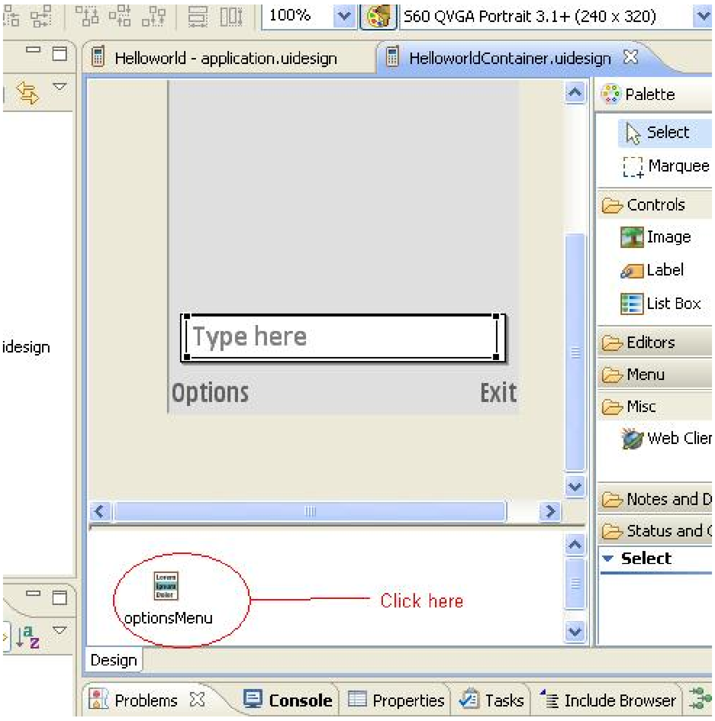
The output can be observed in the emulator.


[1] S60 3rd Edition FP1 SDK for Symbian OS - ...
File
Tools
Help
QVGA Portrait 240x320 + bottom S.K.
Helloworld
Options
Exit

Step 13 - Add a menu option

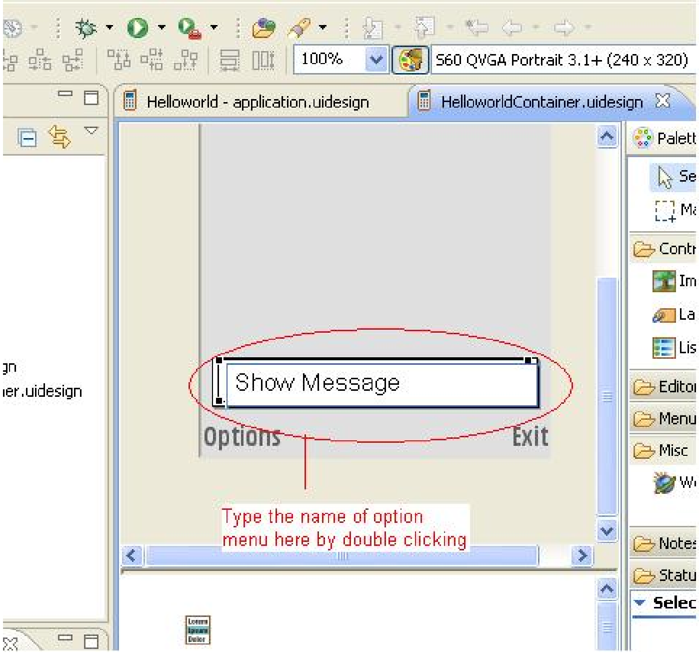
Now we have the basic Symbian C++ helloworld application running, doing nothing great.

1. Further, click on options menu in the bottom window. It will bring up the menu in the main application window.



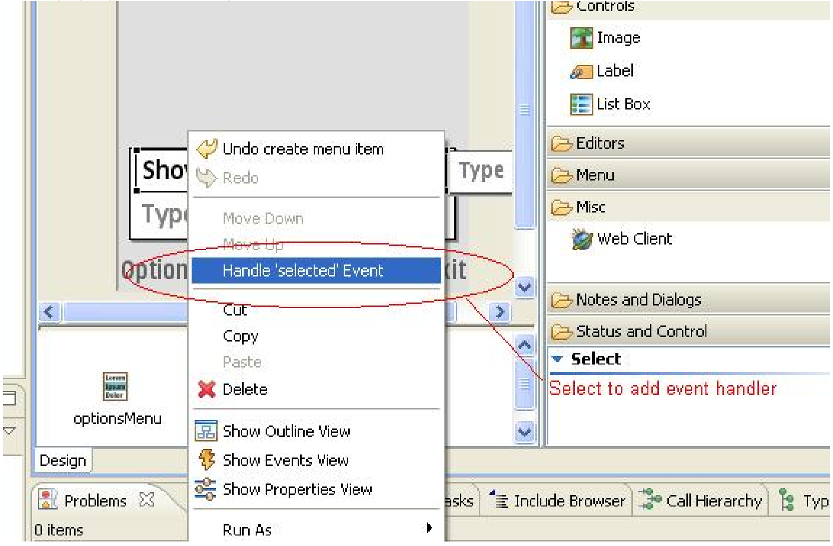
Click here

2. Next, in the menu type down the name as shown in figure.
   This will add a new menu option to the "Option" menu.


100%
S60 QVGA Portrait 3.1+ (240 x 320)
Helloworld - application.uidesign
HelloworldContainer.uidesign
Palett
Show Message
Options
Exit

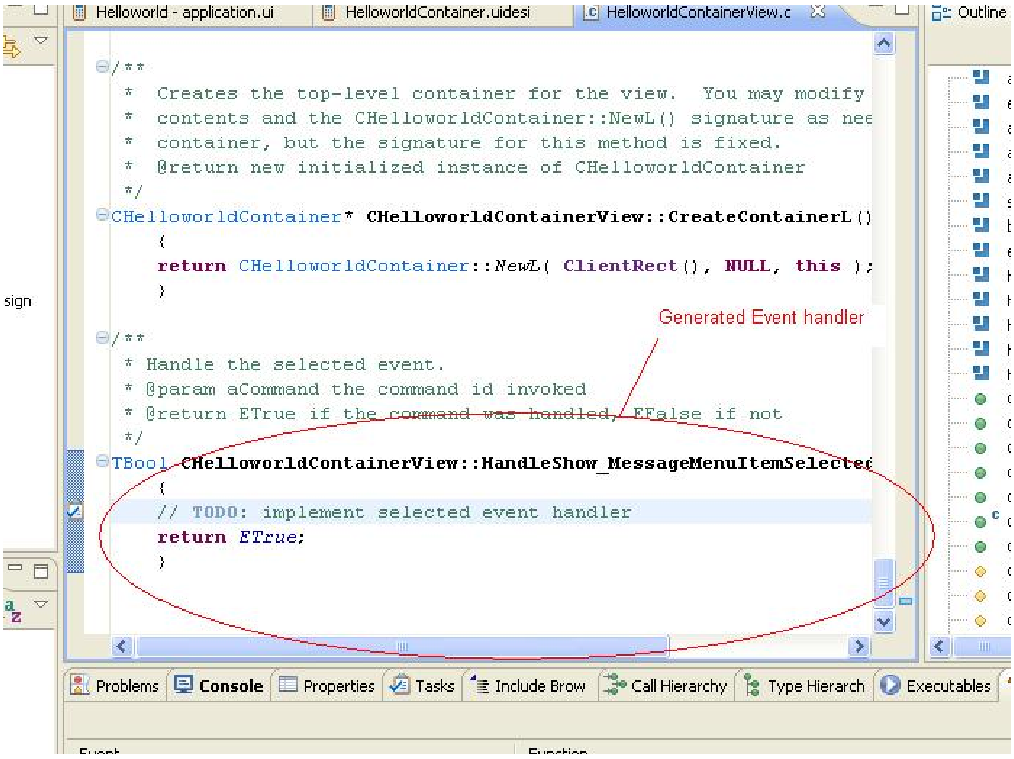
Type the name of option menu here by double clicking

3. Next, right click on the menu option itself and a pop will appear.
4. From this pop up select the option called "Handle Selected Event".



Select to add event handler

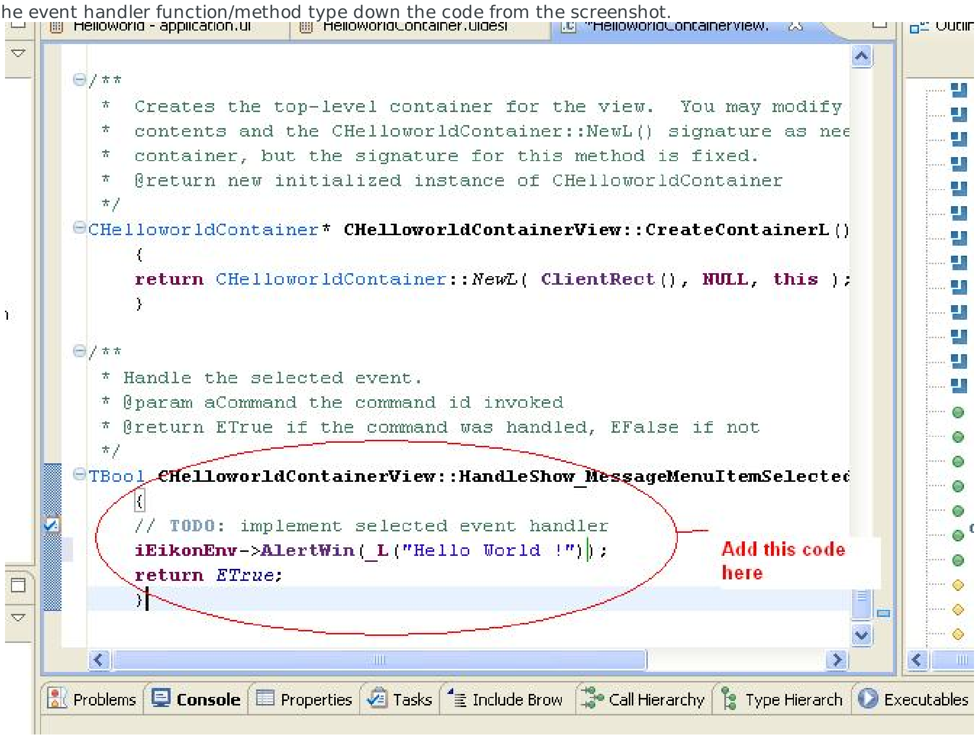
This should bring up the code window, where you can edit the code for the event. This is basically a method which will get called when the menu will be selected.



```
/**
 *  Creates the top-level container for the view.  You may modify
 *  contents and the CHelloworldContainer::NewL() signature as nee
 *  container, but the signature for this method is fixed.
 *  @return new initialized instance of CHelloworldContainer
 */
CHelloworldContainer* CHelloworldContainerView::CreateContainerL()
    {
    return CHelloworldContainer::NewL( ClientRect(), NULL, this );
    }

/**
 * Handle the selected event.
 * @param aCommand the command id invoked
 * @return ETrue if the command was handled, EFalse if not
 */
TBool CHelloworldContainerView::HandleShow_MessageMenuItemSelecte
    {
    // TODO: implement selected event handler
    return ETrue;
    }
```

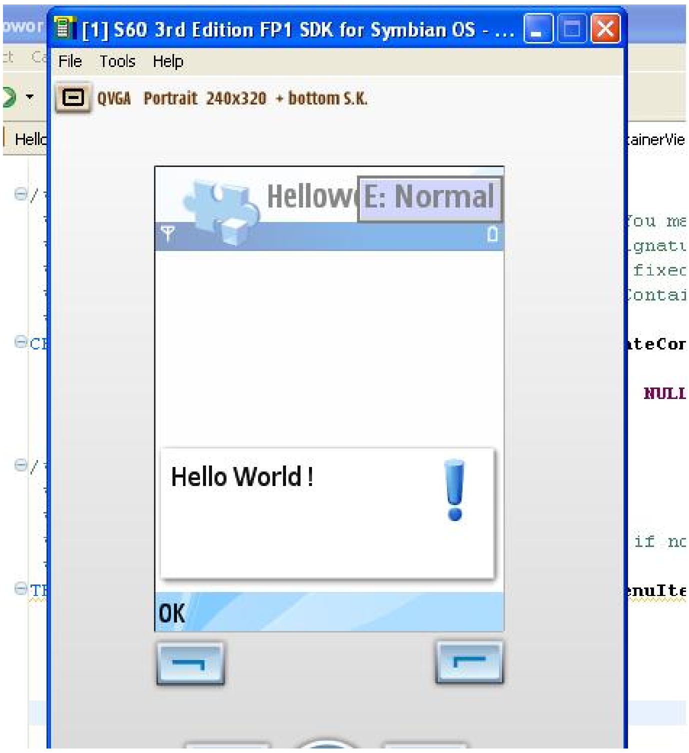
5. In the event handler function/method type down the code from the screenshot.



```
/**
 *  Creates the top-level container for the view.  You may modify
 *  contents and the CHelloworldContainer::NewL() signature as nee
 *  container, but the signature for this method is fixed.
 *  @return new initialized instance of CHelloworldContainer
 */
CHelloworldContainer* CHelloworldContainerView::CreateContainerL()
    {
    return CHelloworldContainer::NewL( ClientRect(), NULL, this );
    }

/**
 * Handle the selected event.
 * @param aCommand the command id invoked
 * @return ETrue if the command was handled, EFalse if not
 */
TBool CHelloworldContainerView::HandleShow_MessageMenuItemSelected
    {
    // TODO: implement selected event handler
    iEikonEnv->AlertWin(_L("Hello World !"));
    return ETrue;
    }
```

6. Now, save the file and run the application the way described above.
This should give the output as below.



[1] S60 3rd Edition FP1 SDK for Symbian OS - ...
File Tools Help
QVGA Portrait 240x320 + bottom S.K.
HelloworlE: Normal
OK

Hello World !

Conclusion

This way, a basic helloworld application can be developed using Carbide c++. Few steps are unnecessary in the wizard but sometimes they turn to be useful. Following the same steps one can develop his own application. This tutorial can be followed by any person with minimal knowledge of Symbian. But for further things a good tutorial on Symbian C++ is necessary.

Source Code

File:Carbid tut1 Helloworld.zip

Back to the tutorial

- Carbide.c++ Tutorials