Quick Start

from August 20, 2009

Here it comes the first small example written in Objective-Basic by you. All you need to do is to follow the following steps and explanations. It will give you some feeling about writing in Objective-Basic and Cocoa – a whole new world to explore.

You will need some time to do the task. Plan to have time for about an hour before you have completed this quick start.

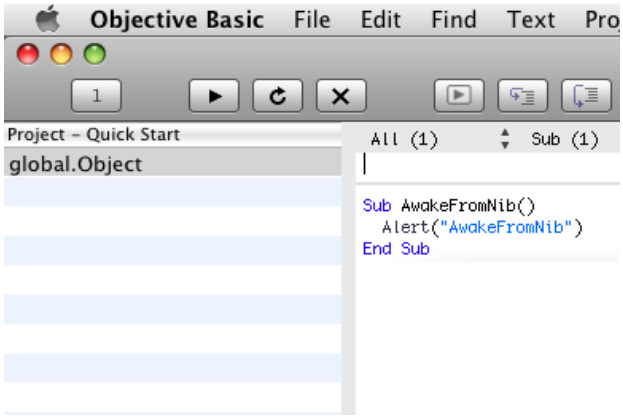
Let's go

Start Objective-Basic by double clicking on the symbol on your desktop.

Create a new project

Select in the menubar *File -> New Project...* Enter a short name used for your .app directory and your project overall. Don't use / . or other special characters. What about the name *Quick Start*?

After pressing OK, a new skeleton project will be created for you (normally on the Desktop of you the current user).



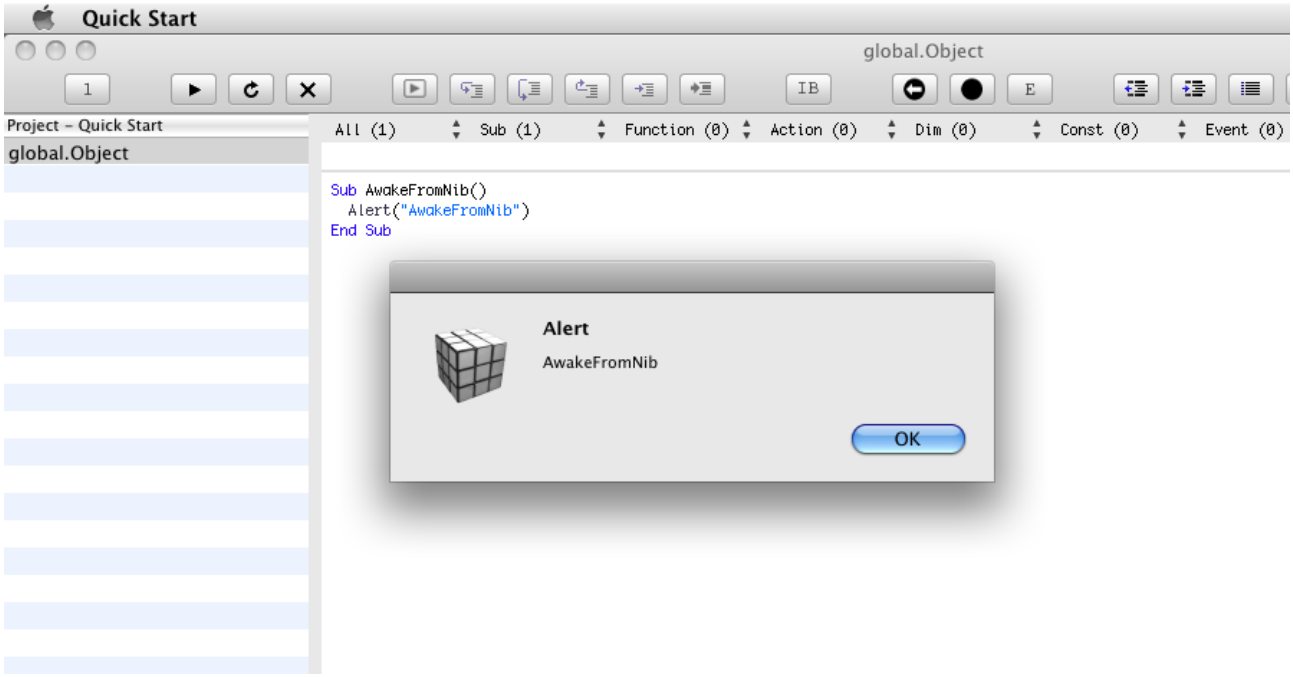
Screenshot 1: The main window of Objective-Basic showing the source code editor on the right and the project file list on the left

The file **`global.Object`** is a special file created automatically for you with the special procedure **`Sub AwakeFromNib()`**. The global.Object file is actually the place where you may declare variables, constants, functions and so on, which may be used as global elements of your program without the need of a object creation before. It is similar to a module in VB. The Sub

AwakeFormNib() is called after the MainMenu.nib is automatically loaded by Cocoa on startup of your application. You may freely add code to it. But we won't do much of it in this small tutorial.

First start of your application

Anyway, let's just run this small example and take a look what happens when you hit the run button in the toolbar of the Objective-Basic IDE or in the menubar *Compiler -> Run*.



Screenshot 2: The screen of your Mac will look like this after running it

You probably noticed the command **`Alert("AwakeFromNib")`** in the Sub AwakeFromNib(), which makes the window appearing with the labels Alert and AwakeFromNib after you started your example.

Stopping your running example

When you hit the Stop button in the toolbar or in the menubar *Compiler -> Stop,* your application will immediately stop execution.

First change of your application

Let us change the label Main to something more meaningful like "Hello World!". To do so delete the line Alert("AwakeFromNib") and write Alert("Hello World!"). Run the example again, if you don't know how see the paragraph above. Now, Hello World! Will appear instead of AwakeFromNib. Hey, you just change your first application in Objective-Basic. Congratulations!

Windows and buttons

Time to get a more sophisticated way to get a window on screen. Now, I want you to start **Interface Builder** create a new window, with a button on it. Change the button's title when clicking on it and showing a message when it is pressed. Interface Builder is our friend. To start it, select in the menubar *Interface Builder -> MainMenu.nib*.

Interface Builder gets loaded with the default GUI file MainMenu.nib.

Interface Builder

To get a quick overview about Interface Builder, read the documentation provided for it, called "Interface Builder" downloadable on www.objective-basic.com.

How to create a new window? Select in the menubar of Interface Builder *Tools -> Library*. Search the list for the right element. Find *Library -> Cocoa -> Application -> Windows* and select in the icon list beneath the previous list the entry *Window* by double clicking on it. A new window will be inserted in the project list of MainMenu.nib.

New Window
Find the project file list named *MainMenu.nib* and double click on the newly created Window shown as *Window (Window)*. The window will appear as empty window on screen ready to be exteneded by dropping controls on it.

New Button
Place a button on the window. (All controls are listed in the library window you used to create the window). Search for *Library -> Cocoa -> Views & Cells -> Buttons* and select in the icon list beneath the previous list the entry *Push Button* by drag & drop it to the new created window. (Press the left button on the Push Button and draw your mouse onto the window and stop pressing the mouse button.)

Basic usage of Interface Builder
In Interface Builder you draw your windows and controls and connect them to variables and functions declared in your source code in the Objective-Basic IDE.

Next Step
Save the file opened in Interface Builder. In the menubar of Interface Builder: *File -> Save*

Now, you have created a window and a button on that window, you need to tell the Objective-Basic IDE what have to be done with them. But before that, it is time to run your example again. Change to IDE of Objective-Basic by clicking on the icon on the bottom of your screen and select run again.

What will happen? Your message will appear again, but after clicking on OK, your newly created window with the button will appear. Great! You managed to get a custom Cocoa based Window to be shown on your screen.

Writing code for the window

We need a place to write code for the window. There are several ways to do so, but we will create a new source code file in the IDE of Objective-Basic. Please select in the menubar Project -> New File... enter *code.Object* as name and add there a new line:

```
IBOutlet mybutton As Button
```

Where you add this line is regardless.
It declares a variable used for a button in a nib file, which can be accessed and used in source code within your project for us we use it for the button in the newly created window in the nib file.

After that create the following lines as well.

```
IBAction myevent(sender As Object)
```

```
  mybutton.Title = "Just changed the title”
End IBAction
```

It creates a event procudure used for Interface Builder objects, which are executed then the event is triggered (in our case when the button is triggered).

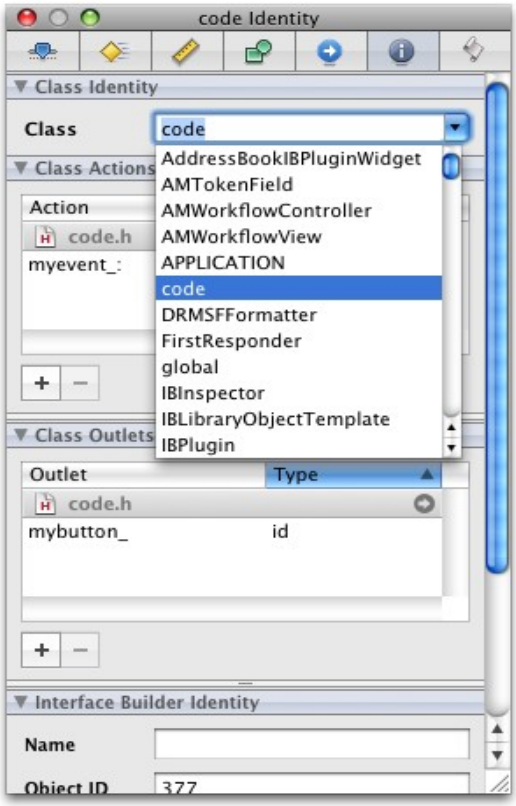
Writing those lines is part one, part two is to doing something in Interface Builder again.

Hit the "Make Objective-Basic" to inform Interface Builder about the changes in your source code, whenever you change code by adding or removing Interface Builder related code in the Objective-Basic IDE. In never version of the IDE of Objective-Basic, it automatically updates the changes for IB, when you use the Interface Builder Button in the toolbar or the related items in the menubar to switch to Interface Builder.

Telling Interface Builder about our source code
So switch to Interface Builder again. We need to make the code file *code.Object* available in Interface Builder by telling it to use it when the MainMenu.nib file is loaded. To do so, we need to create a new empty object and add to the list of objects.

How to create a new object? Select in the menubar of Interface Builder *Tools -> Library*. Search the list for the right element. Find *Library -> Cocoa -> Objects & Controllers -> Controllers* and select in the icon list beneath the previous list the entry *Object* by double clicking on it. A new object will be inserted in the project list of MainMenu.nib.

Well, done. Select this newly created object and change the class of that object. It is not important that you know what exactly class means here. We may see it as the file name used for the object, in our case our file name is *code.Object* or short for Interface Builder *code*. Actually, our class name we need is *code*.



Screenshot 3: The property window used to change the class name of the object.

Change the page to Identity of the property window to see the class identity

Telling Interface Builder about your created IBOutlet and IBAction

We are nearly finished with your example, the last task we need to do is to connect the IBOutlet written in the source code file with the button created in the window.

For the IBOutlet: Select the newly created object *code* in the MainMenu.nib window. After that keep pressing the control button on your keyboard and keep pressing the left button of your mouse and move onto the button. A selection marker will appear, leave the buttons and select *mybutton_:* showing up. You just did it. That's it.

For the IBAction: The other way around. Select the button. After that keep pressing the control button on your keyboard and keep pressing the left button of your mouse and move onto the object *code* in the MainMenu.nib window. A selection marker will appear, leave the buttons and select *myevent_:* showing up. You just did it again.

Save the nib file. Switch to the IDE of Objective-Basic and **build & run your example**. After the first message is shown, press on the button appearing on screen and so that the title of the button is changed after that.

To be continued...