

Pocket Smalltalk Tutorial

By [Andrew Brault](#)

Updated by N. Chamfort (December 16, 2001).

This tutorial steps you through the development of a simple (one form, one button) application using Pocket Smalltalk. It is assumed you have at least a basic familiarity with Smalltalk, but that is not absolutely required.

In order to follow this tutorial you should have the following components installed:

- Latest version of [Pocket Smalltalk](#)
- HotSync software (or equivalent) for your PalmPilot, or else a [Palm OS Emulator](#) running on your PC
- The [PiIRC](#) resource compiler (you can use a different resource compiler as long as it lets you control assignment of resource IDs)

Make sure you have PiIRC (pilrc.exe) in your search path. Alternatively, you can simply place it in the Pocket Smalltalk directory and run it from there.

Once you have everything installed, follow these steps to create your first application:

1. [Start a new project](#)
2. [Using the Class Browser](#)
3. [Create application resources](#)
4. [Add application resources to your project](#)
5. [Create an Application class](#)
6. [Generate an executable application](#)
7. [Create bindings to PalmOS widgets](#)

1. Start a New Project

First, start Pocket Smalltalk. You will get a small "launcher" window. From here you can access various components of the development environment. Your first task is to create a new project for your application. Projects are a way to organize the code for your application, and to keep code you write separate from code in the standard library.

When Pocket Smalltalk first starts up it has an "empty" project with only a few basic Smalltalk classes and no methods. In order to get a useful class library for application development you must install one or more packages. A package is simply a text file containing Smalltalk source code, and several such packages are included in the Pocket Smalltalk distribution.

You need to install two packages for this tutorial: *core.st* and *forms.st*. The *core.st* package contains the base Smalltalk class library (collections, numbers, and so forth), and the *forms.st* package contains a framework for PalmPilot applications.

To install these packages, open a Package Browser from the launcher window (use the *Tools/Package Browser* menu option). The package browser window has two panes. The top pane has a list of currently installed packages. Right now there are no packages installed, so the only thing listed here is (uncommitted), which is a special package that contains source code that does not belong to any other package.

The bottom pane of the Package Browser window lists the contents of the selected package. You can switch between viewing the classes, methods, and constants in the package using the three tabs in the middle of the window.

Select the *Package/Install Package...* menu option in the Package Browser. You will be prompted for the filename of the package to install. You first need to install the *core.st* package, which is located in the same directory as the Pocket Smalltalk executable. After selecting this file there will be a momentary pause as the package is compiled and installed. You will then see *core.st* appear in the list of packages, indicating that it has been successfully installed. You can click on the package name to view the classes defined by the package.

Now install the *forms.st* package in the same way: choose the *Package/Install Package...* menu option and select the *forms.st* package. You should now have both packages installed and visible in the Package Browser.

There is an important step you should perform at this point. The two packages you have loaded are now part of your current project. When you save your project the packages will be written to disk along with any changes you have made to them. For packages you create yourself, this is appropriate, but for "system" packages such as the two you just installed it is not.

You can therefore tell the development environment to not save those packages along with your project. To do this, select one of the packages in the Package Browser and use the *Package/Don't save with project* menu option. Do this for both the *core.st* and *forms.st* packages.

Now you have your project set up to receive source code for the application you are about to write. But first, you need to create a package for yourself into which your new classes and methods will go. To do this, select the *Package/New package...* menu option in the Package Browser. Your new package will be called *tutorial.st*, so type that name into the file dialog box. You can place the new package in the Pocket Smalltalk directory, but to avoid clutter it is recommended that you create a new directory for each project.

After creating the new package, its name will appear in the Package Browser. Notice that the bottom status line in the Package Browser now indicates that your new package is the "default" package. This means that all new classes and methods you define will be put into your new package, which is what you want at this point.

Now you're ready to write an application! You should "save" your project now by selecting the *System/Save project* menu item in the Launcher window. Give it the filename *tutorial.prj* and put it in the same directory as your *tutorial.st* package.

2. Using the Class Browser

Most of your source code will be written using the Class Browser. This tool allows you to browse, edit, and search Smalltalk source code.

Open a Class Browser by selecting the *Tools/Class Browser* menu option from the Launcher window. Class browsers have three panes. The upper-left pane gives a hierarchical view of the classes present in your project. If a class has subclasses, a small square appears to the left of the class name; clicking on it hides or reveals the subclasses.

The upper-right pane contains the message categories for the selected class. Clicking the square next to a category name hides or reveals the methods in that category. The *Instance* and *Class* tabs above the method list let you switch between "sides" of the class (i.e., class and metaclass).

The bottom pane is where most of your source code editing takes place. It usually shows the source code for the selected method, or the definition for the selected class. You can switch between four different text views using the tabs above the source code pane; these are, from left to right:

- Source: the source code for the selected method
- Class: the definition for the selected class
- Disassembly: the bytecode disassembly for the selected method
- Comment: the comment for the selected class

In any of these views except for Disassembly you can edit the displayed text and submit it for compilation by typing *Ctrl-S*.

All of the panes in Pocket Smalltalk windows have "context menus" associated with them. You can access these context menus by right-clicking in the desired pane. Context menus in the class browser allow you to perform a wide variety of operations, some of which will be covered in this tutorial.

For now, you should take a few minutes to experiment with the Class Browser and look at what classes are available. The next page will show you how to build the layout for your application's user interface.

3. Create Application Resources

You will need to create PalmOS resources for hold user interface layouts, bitmaps, an application icon, the application's version ID, and so forth. This is not part of Pocket Smalltalk; you need to use external tools to do this. I recommend using [PiIRC](#) for compiling resources. The following steps assume that you have PiIRC installed and can invoke it from the MS-DOS prompt.

The first step is to create a text file in your project's directory. If you do not have a good text editor you can use the Notepad program supplied with Microsoft Windows. Create a file named *tutorial.rcp* with the following contents:

```
FORM ID 3000 AT (0 0 160 160)
BEGIN
  TITLE "Smalltalk Tutorial"
  BUTTON "Push Me" ID 4000 AT (65 50 AUTO AUTO)
END
```

This is PiIRC syntax for a simple form layout. It specifies a form ID of 3000, and the form contains a title and a single button with an ID of 4000. You will use these ID numbers later to create bindings to these PalmOS user interface elements from Smalltalk.

Compile the file you just created with PiIRC by typing the following at the MS-DOS prompt (in the same directory as the file

you just created):

`pilrc tutorial.rcp`

This should yield a file named `tFRM0bb8.bin`. If you had put more than one form definition in the `.rcp` file this would have produced more than one `.bin` file.

If you make changes to the `.rcp` file containing your form definition(s), you must re-run PiIRC as shown above.

4. Add Application Resources to your Project

Pocket Smalltalk creates "executable" `.prc` resource databases by merging one or more existing resource databases with resources it generates containing your compiled Smalltalk code. You must therefore tell Pocket Smalltalk which resource databases to include in the final `.prc` file. At a minimum, you will need to include the virtual machine resource database. This can be found in the Pocket Smalltalk home directory. There are actually two virtual machine databases; the one you should use depends on whether you are planning to use double precision floating-point support:

- `vm-math.prc` - contains floating-point support
- `vm.prc` - no floating-point support

For this tutorial, you should use `vm.prc`, since you will not be using floating-point numbers and floating-point support requires you to install a special math library.

The way to tell Pocket Smalltalk which resource databases to include is to create a named constant for each database. Do this by opening a Constants Browser (*Tools/Constants Browser* from the Launcher window). Use the Constants Browser menu option *Constants/Add resource database...* and select the `vm.prc` file. A new constant named `##resourceDB1` is automatically created, and its value (shown in the bottom pane) is the filename you specified. Pocket Smalltalk will look for constants named `##resourceDB1`, `##resourceDB2`, and so forth, and it will read those files and integrate them into its final "executable" resource database.

Pocket Smalltalk distinguishes between files containing complete resource databases (with extension `.prc`) and files containing single resources (with extension `.bin`). PiIRC creates single-resource (`.bin`) files, so you could add a `##resourceDB` constant for each `.bin` file created by PiIRC. However, for nontrivial applications there are usually a great many `.bin` files, so you can instead specify a wildcard filename matching the `.bin` files you wish to include. The easiest way to do this is to again use the *Add resource database...* menu option and this time use the drop-down list at the bottom of the file selection dialog to select files of type "*PalmOS Resources*". Then select *one* of the `.bin` files that were created by PiIRC. The bottom pane of the Constants Browser will now contain a filename; you should edit the filename from `tFRM0d7a.bin` (or whatever it is) to `*.bin`. Accept with *Ctrl-S* or the right-button menu. This tells Pocket Smalltalk to search the directory for any `.bin` files and automatically add them to your finished application.

Now is a good time to save your project (*System/Save project* in the Launcher window). This will record your work so you will not have to do it again if you leave Pocket Smalltalk and come back later.

5. Create an Application Class

Now you are ready to actually write some Smalltalk code. In this tutorial you will only write a small amount of code, but in a real project most of your time will be spent writing Smalltalk code (rather than getting things set up as you did in the last few pages).

The package `forms.st` which you installed contains a lightweight application framework which takes most of the tedium out of writing PalmOS applications. There are classes such as `Button` and `TextField` which represent PalmOS user interface elements, and an `Application` class which implements an event loop and methods for managing bindings between PalmOS and Smalltalk.

The first step is to create a subclass of `Application` for your new application. Open a Class Browser and select the `Application` class. Use the menu option *Class/Make subclass...* to create a new subclass; name the new class `TutorialApplication`. This adds a new class to your project, which you can confirm by opening a Package Browser and selecting the `tutorial.st` package.

To tell PalmOS to use the form layout you defined earlier, you must override the class-side method `#formID`. Make sure the new `TutorialApplication` class is selected in the Class Browser, then set the *Instance/Class* tab in the upper right to *Class*. You can now add class-side methods.

Right-click in the upper-right pane and select *New method* from the context menu. The source code pane at the bottom clears, allowing you to enter a new method. Type the following into the source code pane, then hit *Ctrl-S* to save it:

```
formID
  ^3000.
```

This tells the application framework to use Form ID 3000 for this Application subclass.

This is the only *required* method; the others are optional. To actually launch the application you must override the method `#start` on the class side of the class `Smalltalk`. Replace the default method definition with the following:

```
start
    TutorialApplication show.
```

Now you are ready to test the application on the PalmPilot.

6. Generate an Executable Application

Generating an executable `.prc` file is very simple: just select the *System/Generate Code* menu option in the Launcher. Enter an appropriate filename for the `.prc` file (for example, `tutorial.prc`) and, after a short delay, the file is written to disk. You can then install the file on your PalmPilot with HotSync or an equivalent tool. Since you have not given the application a name yet, it shows up as just "PST App". You can set the application's name by changing the value of the `##applicationTitle` constant in the Constants Browser. You can also change the application's Creator ID via the `##creatorID` constant.

Note that you need to use the *System/Generate Code* option every time you want to test your application on the PalmPilot, in order to have your changes take effect. You also need to re-generate the code if you change a form layout or some other included resource.

Try running the application on your PalmPilot; you should see the form layout defined earlier. Pushing the button will not have any effect, because you have not yet told Pocket Smalltalk about the button. This is covered in the next page.

7. Create Bindings to PalmOS Widgets

The application framework in Pocket Smalltalk provides an easy way to associate Smalltalk objects with PalmOS controls. To specify an action to be performed when the "Push Me" button is tapped, you need to override the `#createComponents` method in the `TutorialApplication` class (make sure the *Class/Instance* tab is set to *Instance*). Define it as follows:

```
createComponents
    self add: Button id: 4000 name: nil aspect: #buttonClicked.
```

The message `#add:id:name:aspect` is the main way to create bindings between Smalltalk objects and PalmOS controls. The first argument is the class of the Smalltalk object to create. The *id:* argument is the control ID you specified in the form layout. The *name:* argument can be a symbol which names the object for later reference. This button does not need a name so this argument is `nil`. Finally, the *aspect:* argument is interpreted differently depending on what kind of form control is being defined. The full set of aspect interpretations is specified in the User's Manual, but for a button the aspect is simply a selector that is sent to the Application instance when the button is clicked.

Since you specified `#buttonClicked` as the aspect of the button, you must create a `#buttonClicked` method in the `TutorialApplication` class (*Instance* side). Define it as:

```
buttonClicked
    Form notify: 'Hello, world!'.
```

The `Form notify:` method will pop up a dialog with the message.

Now that the binding is created, try re-generating the `.prc` (*System/Generate code* in the Launcher) and reloading it into your PalmPilot. This time the message should be displayed when you tap the button.

Congratulations! You have written your first application in Pocket Smalltalk. You have also been introduced to many of the components of the development environment. Your next steps should be to read the User's Guide and experiment with modifications to this tutorial application.

Last updated: Dec 16, 2001