macoslib for Newbies

Document for **macoslib** v146.

You do not know anything about OS X guts ? Nevermind, **macoslib** can still be helpful for you because it implements some classes and many new methods that you can use without any specific knowledge of OS X, even if you will not be able to fully unleash the power of **macoslib**.

Installing macoslib into your Project

1. Open the *macoslib.rbvcp* file with Xojo or Real Studio.
2. Select the "macoslib" folder inside the project.
3. Copy it by typing ⌘-C.
4. Create a new project.
5. Paste "macoslib" into it by typing ⌘-V.
6. Go back to the *macoslib.rbvcp* project and copy the "Additional Modules" folder.[1]
7. Paste it into your own project.

You are ready to go !

Note: you **cannot** drag&drop macoslib files from the Finder to the project window.

Using macoslib

Unless you want to learn how the different functions are implemented, you will probably never need to look into the "macoslib" folder.

By contrast, looking in the "Additional Modules" folder will show you all the available improvements, especially in the "Class Extensions" sub-folder. All the methods take and return pure Xojo-types, whenever possible.

Finding Additional Functions

Unfortunately, there is no full documentation yet, but it is on its way. In between:

1. Look for methods and new classes in the "Additional Modules/Class Extensions" folder.
2. Most extension methods will appear in the autocompletion menu.
3. When you run the *macoslib.rbvcp* project, the *Examples* menu has a "Convenience Extensions" sub-menu which contains examples which are only based on the *Additional Functions* (this is what you are looking for).

Contents of the **Additional Modules** folder:

Class Extensions: a collection of convenience extensions which take and return only Xojo native types.
Cocoa MenuItems: implementation of the common Cocoa MenuItems.
TT's SmartPreferences: handles app preferences. Works on OS X, Win and Linux.
KT's PList Stuff: Kem Tekinay's property list helper.
DebugReport Cross-Platform: a debugging system to follow execution of your code.

[1] The "Additional Modules" folder is what you want but, as it uses classes and functions from the "macoslib" folder, you need to paste the latter first.

If you Compile your Application for Carbon

Many functions in **macoslib** use Cocoa instead of Carbon. However, a Carbon application can use Cocoa seemlessly if your application executes the following code at startup in your *App.Open* event:

```
Cocoa.Initialize
```

Nothing more necessary and there is no problem calling it inside a Cocoa application or calling it several times.