FREE eBook

LEARNING osx

#osx

Table of Contents

About

Chapter 1: Getting started with osx

Remarks

This tag is for documentation on Apple-created Mac-specific programming APIs, such as AppKit.

Examples

Overview of Frameworks

Apps written for macOS are usually written with Apple's Frameworks. The frameworks that almost every app will use are:

- AppKit - for creating and managing UI elements
- Foundation - for common non-UI objects and operations

There are other common frameworks that are used in many, but not all apps:

- CoreData - for data storage
- Dispatch - for management of multiple threads
- CoreGraphics - for drawing a graphics related tasks
- CoreAnimation - for animation of UI elements

Read Getting started with osx online: https://riptutorial.com/osx/topic/2818/getting-started-with-osx

Chapter 2: File association

Examples

Set my app as default app for a file type

```
- (NSString *) UTIforFileExtension:(NSString *) extension {
    NSString * UTIString = (NSString
*)UTTypeCreatePreferredIdentifierForTag(kUTTagClassFilenameExtension,
                                                                        (CFStringRef)extension,

                                                                        NULL);

    return [UTIString autorelease];
}

- (BOOL) setMyselfAsDefaultApplicationForFileExtension:(NSString *) fileExtension {
    OSStatus returnStatus = LSSetDefaultRoleHandlerForContentType (
                                                                    (CFStringRef) [self
UTIforFileExtension:fileExtension],
                                                                    kLSRolesAll,
                                                                    (CFStringRef) [[NSBundle
mainBundle] bundleIdentifier]
                                                                    );

    if (returnStatus != 0) {
        NSLog(@"Got an error when setting default application - %d", returnStatus);
        // Please see the documentation or LSInfo.h
        return NO;
    }

    return YES;
}
```

source

Create association with new/custom file types via Info.plist

```
<key>CFBundleDocumentTypes</key>
<array>
    <dict>
        <key>CFBundleTypeIconFile</key>
        <string>Icon file for associated file</string>
        <key>CFBundleTypeName</key>
        <string>My file format</string>
        <key>CFBundleTypeRole</key>
        <string>Viewer</string> <!-- The value can be Editor, Viewer, Shell, or None. This key
is required.  -->
        <key>LSItemContentTypes</key>
        <array>
            <string>UTI of the file</string> <!-- Existing UTI or create a UTI for your new
file type -->
        </array>
        <key>LSHandlerRank</key>
        <string>Owner</string>
    </dict>
</array>
```

source

Read File association online: https://riptutorial.com/osx/topic/10926/file-association

Chapter 3: NSFont

Introduction

NSFont is the object that provides Mac applications with glyph information and font characteristics to be used for primarily for display. You'll learn how to create and use NSFont objects in a variety of ways, both common and uncommon.

Examples

Creating an NSFont object

The preferred and most common way of making an NSFont object is the following:

Objective-C

```
// Name is PostScript name of font; size is in points.
NSFont *essayFont = [NSFont fontWithName:@"Times New Roman" size:12.0];
```

Read NSFont online: https://riptutorial.com/osx/topic/8881/nsfont

Chapter 4: NSMenuItem

Remarks

See Apple's Documentation here:
https://developer.apple.com/library/mac/documentation/Cocoa/Reference/ApplicationKit/Classes/NSMenu

Examples

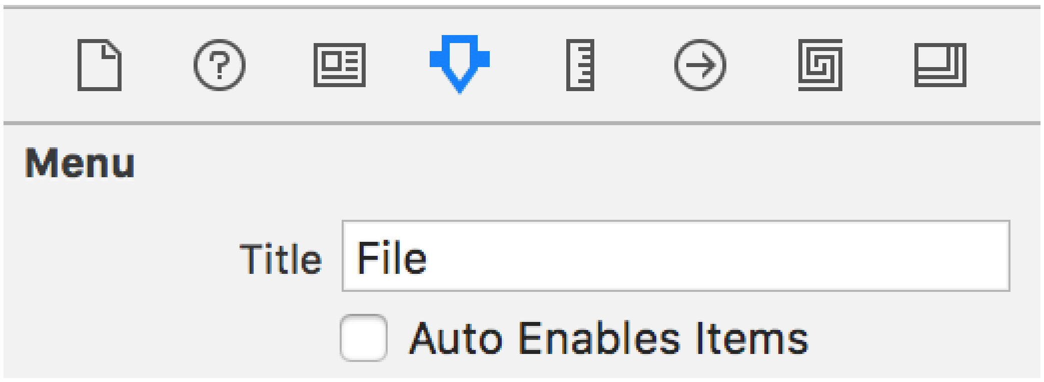
Enabling menu items

Manually enabling menu items

To manually control the enabled state of a menu items the menu that contains it must disable automatic enabling of its items

Menus can turn off automatic enabling in one of two ways:

1. In the Interface Builder


Menu

Title File

Auto Enables Items

2. In code

```
menu.autoenablesItems = false
```

Both of the mechanisms set the autoenablesItems property on NSMenu.

Once the menu is menu is no longer enabling and disabling menu items the menu items can be programmatically set in one of two ways

1. In Interface Builder
2. In code

```
menuItem.enabled = true
```

Automatically enabling menu items

Menu items can be automatically enabled by connecting menu items actions to the first responder and implementing the delivered action on an object in the responder chain as described by Apple's docs.

Supporting default menu actions

Menus act like all standard control items. They have an action which is the function to be called and a target which is the object to send the function to. If the target is set to an object then when a user selects a menu item it the action method will be sent to the target object. If the menu item has an action, but not a target then the target will be dynamically selected from the first object from the following that responds to the action:

1. The first responder
2. The view hierarchy
3. Window
4. Window controller
5. `NSApplication`
6. `NSApplication.delegate`
7. `NSApplication.nextResponder`

Implementing the default Open (O) menu item can be accomplished by implementing the `openDocument` method on any object in the above list.

```
- (IBAction)openDocument:(id)sender {

}
```

Adding and removing items to a menu

```
// add an item to a menu
menu.addItem(item)

// remove and item from a menu
menu.removeItem(item)
```

Read NSMenuItem online: https://riptutorial.com/osx/topic/6038/nsmenuitem

Chapter 5: NSRunLoop

Examples

simple daemon application

A daemon process executes a program in the background, usually without user interaction. The example below shows how to create a daemon and register a listener, which monitors all open applications. The main part is the function call `NSRunLoop.mainRunLoop().run()`, which starts the daemon.

```
class MyObserver: NSObject
{
    override init() {
        super.init()

        // app listeners
        NSWorkspace.sharedWorkspace().notificationCenter.addObserver(self, selector:
"SwitchedApp:", name: NSWorkspaceDidActivateApplicationNotification, object: nil)
    }

    func SwitchedApp(notification: NSNotification!)
    {
        print(notification)
    }
}


let observer = MyObserver()

// simply to keep the command line tool alive - as a daemon process
NSRunLoop.mainRunLoop().run()
```

You can also use this code as a basis for a server process.

Read NSRunLoop online: https://riptutorial.com/osx/topic/6667/nsrunloop

Chapter 6: NSStoryBoard

Examples

Open a New Window Controller

To open a new window, add the following code somewhere where you can keep a reference to the new window (I.E., the app delegate).

Swift

```
let storyboard:NSStoryboard = NSStoryboard(name: "Main", bundle: nil)
guard let controller:NSWindowController =
storyboard.instantiateControllerWithIdentifier("myWindowController") as? NSWindowController
else { return /*or handle error*/ }
controller.showWindow(self)
```

Objective-C

```
NSStoryboard *storyBoard = [NSStoryboard storyboardWithName:@"Main" bundle:nil]; // get a
reference to the storyboard
myController = [storyBoard instantiateControllerWithIdentifier:@"secondWindowController"]; //
instantiate your window controller
[myController showWindow:self];
```

Once you create your `controller` make sure you keep a reference it to somewhere outside of the function call. This can be done by creating a `NSWindowController` variable in your app delegate, and assigning your new controller to the variable.

Read NSStoryBoard online: https://riptutorial.com/osx/topic/4287/nsstoryboard

Chapter 7: NSTextView

Introduction

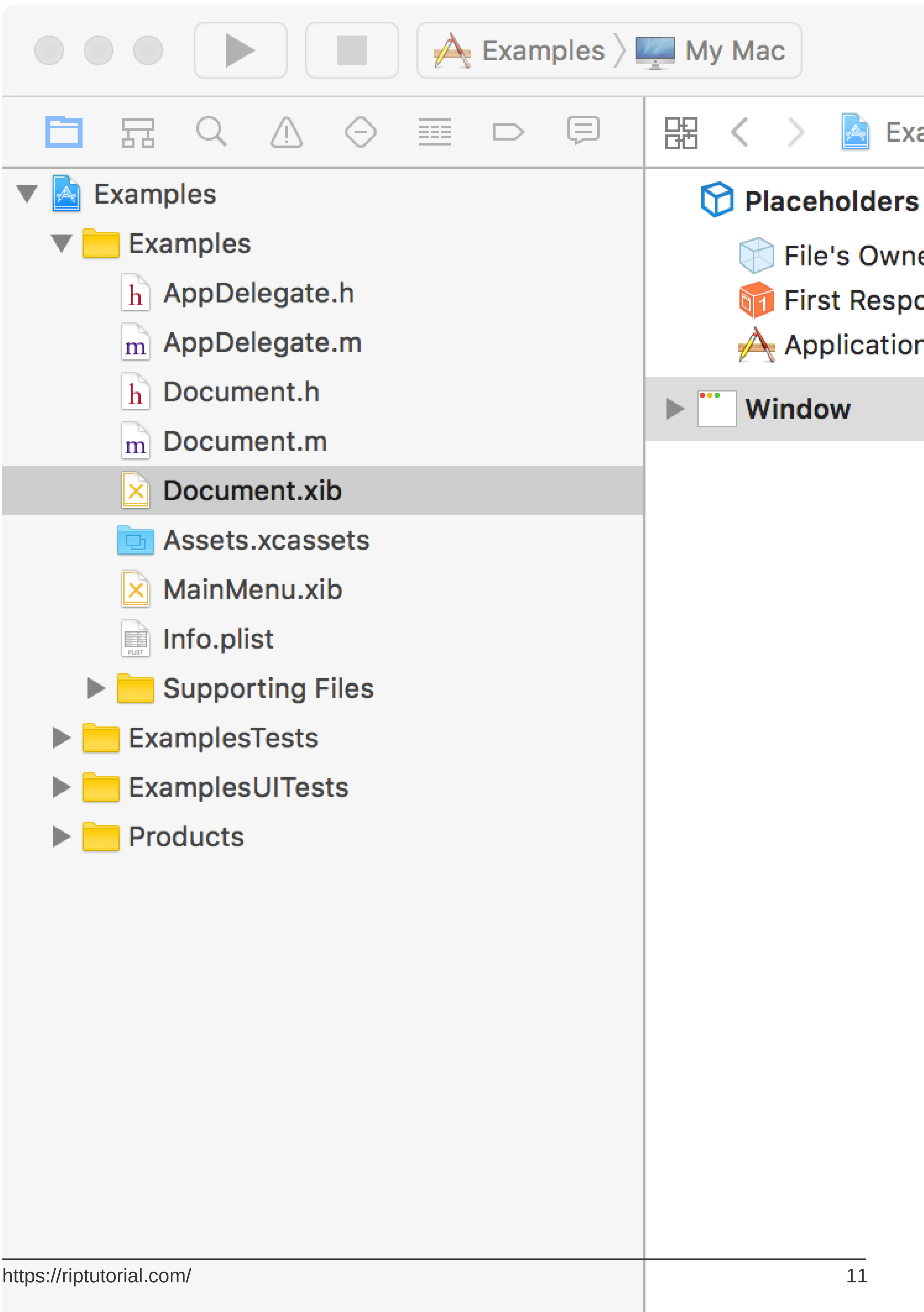
NSTextView is Apple's main handler of AppKit's text system. It contains everything you need to create a text viewer/editor space in OS X (renamed macOS) applications.

Examples

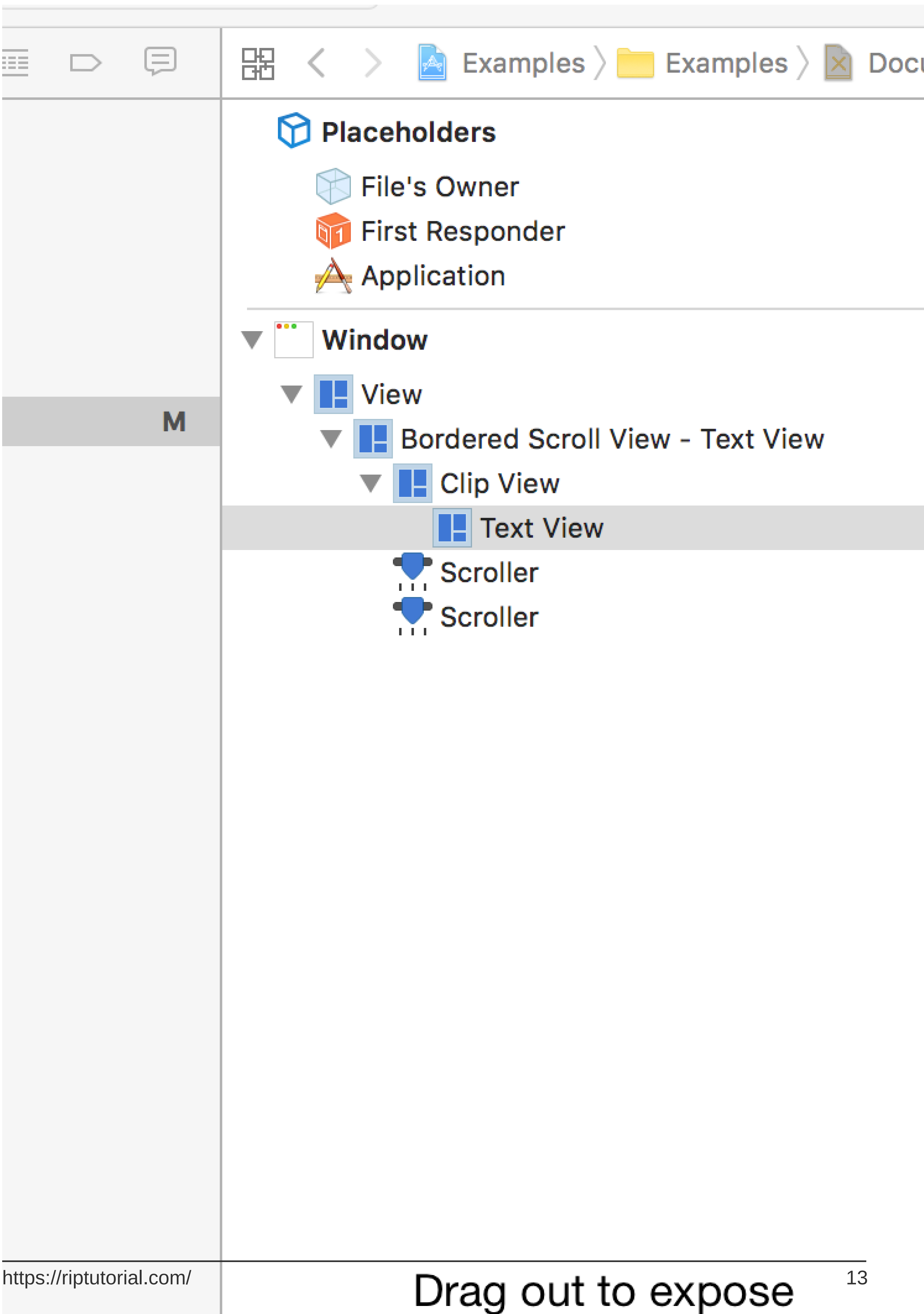
Creating an NSTextView

Graphically

In XCode a simple NSTextView can be created by dragging and dropping one from the Object Library.


Examples
My Mac
Exa
Examples
Examples
AppDelegate.h
AppDelegate.m
Document.h
Document.m
Document.xib
Assets.xcassets
MainMenu.xib
Info.plist
Supporting Files
ExamplesTests
ExamplesUITests
Products
Placeholders
File's Owne
First Respo
Applicatior
Window

that is automatically set to expand vertically with the text view. Make sure when option()-dragging you make connections to the text view and not the scroll view.


Examples 〉 Examples 〉 Docu

Placeholders

File's Owner

First Responder

Application

Window

View

Bordered Scroll View - Text View

Clip View

Text View

Scroller

Scroller

1. An NSLayoutManager - performs glyph/character layout.
2. An NSTextContainer - controls graphical space that glyphs/characters can inhabit.
3. An NSTextStorage - holds the actual string data that NSTextView displays.

- An NSTextStorage can have many NSLayoutManager's, but an NSLayoutManager can only have one NSTextStorage. This is useful if you wish to show the same data in different ways at the same time.

- NSLayoutManager's can have many NSTextContainer's. Useful for paginated text.

- NSTextView's can only have one NSTextContainer at a time.

- Certain things built into NSTextView are off limits at the time of writing. For example built-in Find-and-Replace functions are not able to be customized, but can be overridden with custom functions.

More information on ways to use the text system can be found here.

Now for the code. This code will create a simple NSTextView, with not even scrolling. Such things like scrolling and pagination will be in another example.

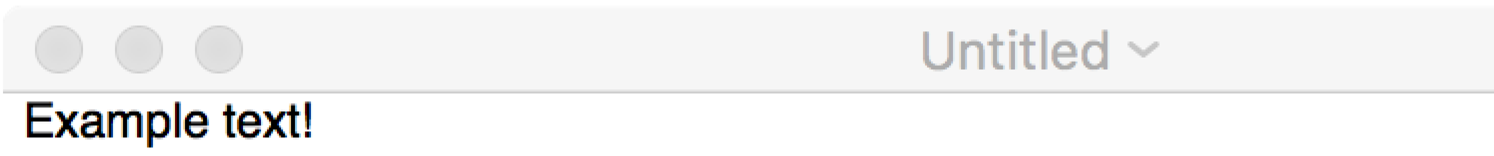
Objective-C

```
// This code resides in an NSDocument object's windowControllerDidLoadNib:(NSWindowController
*)windowController method.
// This is done simply because it is easy and automatically gets called upon.

// This method is also where the following NSRect variable gets size information. We need this
information for this example.
NSRect windowFrame = windowController.window.contentView.frame;
NSTextStorage *textStorage = [[NSTextStorage alloc] initWithString:@"Example text!"];
NSLayoutManager *manager = [[NSLayoutManager alloc] init];
NSTextContainer *container = [[NSTextContainer alloc]
initWithContainerSize:NSMakeSize(windowFrame.size.width, windowFrame.size.height)];
NSTextView *textView = [[NSTextView alloc] initWithFrame:windowFrame textContainer:container];

[textStorage addLayoutManager:manager];
[manager addTextContainer:container];
[windowController.window setContentView:textView];
```

Congratulations! You have made an NSTextView programmatically!



Example text!

Read NSTextView online: https://riptutorial.com/osx/topic/8880/nstextview

Chapter 8: Prompting the user for a file

Introduction

NSOpenPanel provides an API for prompting the user for a file to open. This menu is the standard UI presented by the Open (O) menu item.

Examples

Opening files

Opening any file

```
NSOpenPanel *openPanel = [NSOpenPanel openPanel];
[openPanel beginWithCompletionHandler:^(NSInteger result) {
    NSURL *url = openPanel.URL;
    if (result == NSFileHandlingPanelCancelButton || !url) {
        return;
    }
    // do something with a URL
}];
```

Allowing opening multiple files

```
NSOpenPanel *openPanel = [NSOpenPanel openPanel];
openPanel.allowsMultipleSelection = YES;
[openPanel beginWithCompletionHandler:^(NSInteger result) {
    NSArray <NSURL *>*urls = openPanel.URLs;
    // do things
}];
```

Limiting to specific file types

```
NSOpenPanel *openPanel = [NSOpenPanel openPanel];
openPanel.allowedFileTypes = @[@".png", @".jpg"];
[openPanel beginWithCompletionHandler:^(NSInteger result) {
    NSURL *url = openPanel.URL;
    if (result == NSFileHandlingPanelCancelButton || !url) {
        return;
    }
    // do something with a picture
}];
```

Read Prompting the user for a file online: https://riptutorial.com/osx/topic/9438/prompting-the-user-for-a-file

Chapter 9: Set environment variables

Introduction

In Mac OS X, you can set the environment variables in one of the following files :

~/.bashrc

~/.bash_profile

~/.profile

By default, Mac OS X does not has above files, you need to create it manually.

Examples

Add a path

1.`vim ~/.bash_profile`

The file may not exist (if not, you can just create it).

2.type in this and save the file:

```
export PATH=$PATH:YOUR_PATH_HERE
```

Read Set environment variables online: https://riptutorial.com/osx/topic/10162/set-environment-variables

Credits

S. No	Chapters	Contributors
1	Getting started with osx	Andrew Hoos, Community, l'L'l, tbodt
2	File association	bikram990
3	NSFont	malicedShade
4	NSMenuItem	Andrew Hoos, Barlow Tucker
5	NSRunLoop	Marco Pashkov
6	NSStoryBoard	Barlow Tucker
7	NSTextView	malicedShade
8	Prompting the user for a file	Andrew Hoos
9	Set environment variables	Kuhan