

Flaws in Tutorials, Screencasts, and Samplecode

NSConference 7 Blitz Talk
Tim Mecking (@sptim)

Examples

I. File Read / Write with NSString

I. File Read / Write with NSString

```
-(IBAction)save:(id)sender {
    NSString *path=[NSString stringWithFormat:@"%s/Documents/text.txt",
                      NSHomeDirectory()];
    [self.textView.text writeToFile:path
                      atomically:YES
                      encoding:NSUTF8StringEncoding
                      error:nil];
}

-(IBAction)load:(id)sender {
    NSString *path=[NSString stringWithFormat:@"%s/Documents/text.txt",
                      NSHomeDirectory()];
    self.textView.text=[NSString stringWithContentsOfFile:path
                      encoding:NSUTF8StringEncoding
                      error:nil];
}
```

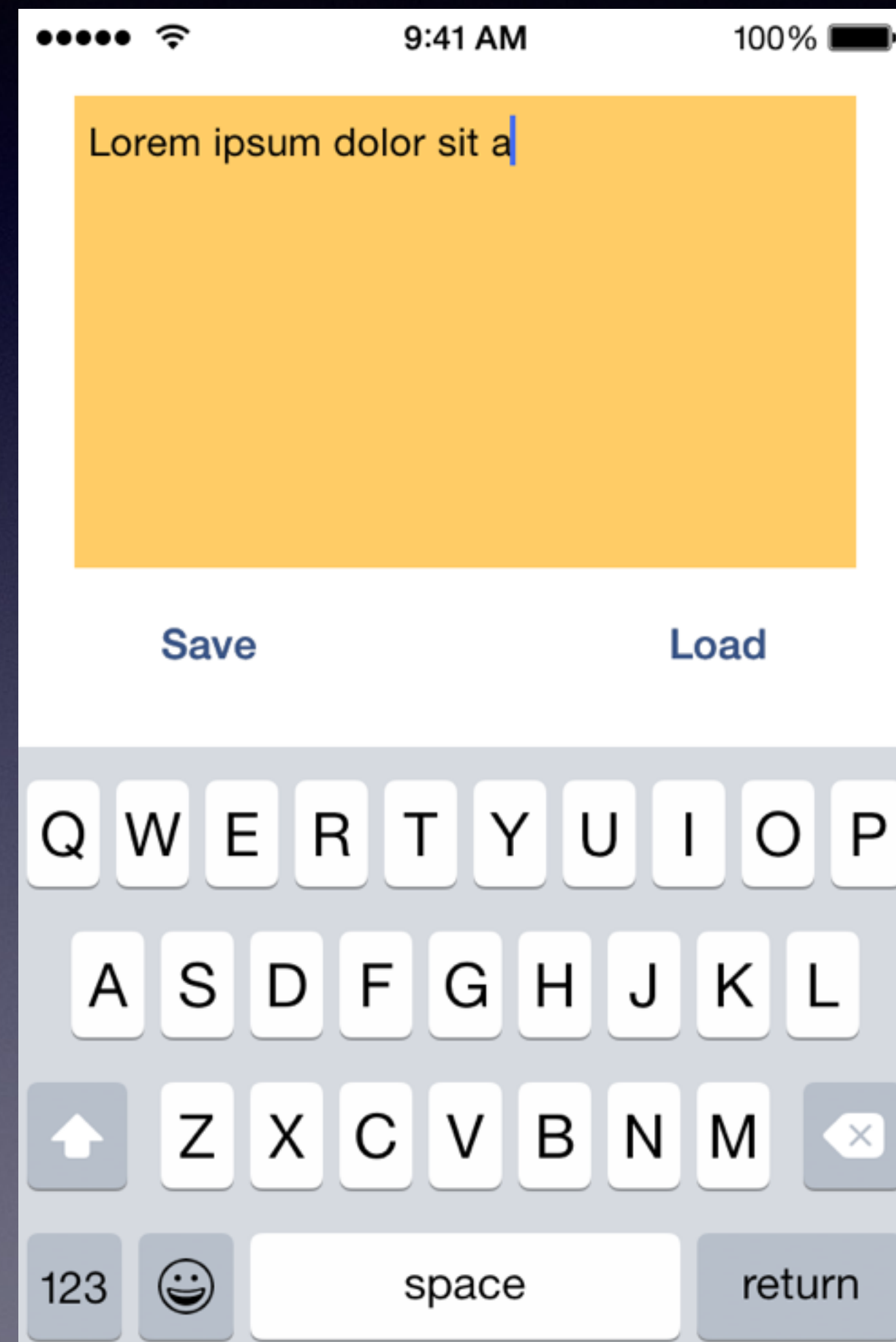
I. File Read / Write with NSString

```
-(NSString*)filePath {  
    NSArray* folderPaths=NSSearchPathForDirectoriesInDomains(NSDocumentDirectory,  
                                                                NSUserDomainMask,  
                                                                YES);  
  
    NSString* folderPath=[folderPaths firstObject];  
    NSString* filePath=[folderPath stringByAppendingPathComponent:@"text.txt"];  
    return filePath;  
}
```


I. File Read / Write with NSString

```
- (IBAction)save:(id)sender {  
    [self.textView.text writeToFile:self.filePath  
        atomically:YES  
        encoding:NSUTF8StringEncoding  
        error:nil];  
}  
  
- (IBAction)load:(id)sender {  
    self.textView.text=[NSString stringWithContentsOfFile:self.filePath  
        encoding:NSUTF8StringEncoding  
        error:nil];  
}
```


I. File Read / Write with NSString





9:41 AM

100%

Lorem ipsum dolor sit a


Save

Load





9:41 AM

100% 

Edit

Lorem ipsum dolor

I. File Read / Write with NSString

```
-(void)viewDidLoad {
    [super viewDidLoad];
    self.navigationItem.rightBarButtonItem=self.editButtonItem;
    [self load:self];
}
-(void)setEditing:(BOOL)editing animated:(BOOL)animated {
    [super setEditing:editing animated:animated];
    if(editing) {
        [self.textView becomeFirstResponder];
    }
    else {
        [self.textView resignFirstResponder];
    }
}
```


I. File Read / Write with NSString

```
-(void)textViewDidBeginEditing:(UITextView *)textView {
    self.editing=YES;
}
-(void)textViewDidChange:(UITextView *)textView {
    self.navigationItem.leftBarButtonItem=self.undoButtonItem;
}
-(void)textViewDidEndEditing:(UITextView *)textView {
    self.editing=NO;
    [self save:self];
}
```


I. File Read / Write with NSString

```
-(IBAction)save:(id)sender {
    [self.textView.text writeToFile:self.filePath
                        atomically:YES
                        encoding:NSUTF8StringEncoding
                        error:nil];
    self.navigationItem.leftBarButtonItem=nil;
}
-(IBAction)load:(id)sender {
    self.textView.text=[NSString stringWithContentsOfFile:self.filePath
                                                encoding:NSUTF8StringEncoding
                                                error:nil];
    self.navigationItem.leftBarButtonItem=nil;
}
```


II. NotificationCenter

II. NSNotificationCenter

[illegible][illegible]

II. NotificationCenter

```
-(void)viewDidLoad {
    [super viewDidLoad];
    self.navigationItem.rightBarButtonItem=self.editButtonItem;
    [self load:self];
    NotificationCenter* notificationCenter=[NSNotificationCenter defaultCenter];
    [notificationCenter addObserver:self.textView
                               selector:@selector(resignFirstResponder)
                               name:UIApplicationDidEnterBackgroundNotification
                               object:nil];
}
-(void)dealloc {
    [[NSNotificationCenter defaultCenter] removeObserver:self.textView];
}
```


III. Display File Size

III. Display File Size

```
NSFileManager *manager=[NSFileManager defaultManager];

NSDictionary *mydic;
if((mydic=[manager attributesOfItemAtPath:@"/etc/hosts" error:NULL])==nil){
    NSLog(@"could not get file attributes");
    return 4;
}
else{
    NSLog(@"Size %i bytes", [[mydic objectForKey:NSFileSize]intValue]);
}
```


III. Display File Size

```
NSFileManager *fileManager=[NSFileManager defaultManager];

NSDictionary *fileAttributes=[fileManager attributesOfItemAtPath:@"/etc/hosts"
                                error:NULL];

if(fileAttributes==nil){
    NSLog(@"could not get file attributes");
    return 4;
}
else{
    NSLog(@"Size %i bytes", [[fileAttributes objectForKey:NSFileSize]intValue]);
}
```


III. Display File Size

```
NSFileManager *fileManager=[NSFileManager defaultManager];

NSError *error;
NSDictionary *fileAttributes=[fileManager attributesOfItemAtPath:@"/etc/hosts"
                                                                    error:&error];

if(fileAttributes==nil){
    NSLog(@"%@",error.localizedDescription);
    return 4;
}
else{
    NSLog(@"Size %llu bytes", fileAttributes.fileSize);
}
```


III. Display File Size

```
let fileManager=NSFileManager defaultManager()

var error:NSError?
if let fileAttributes=fileManager.attributesOfItemAtPath("/etc/hosts",
                                                         error:&error) {
    println("Size \ (fileAttributes[NSFileSize]!) bytes")
}
else {
    println(error!.localizedDescription)
    exit(4)
}
```


IV. Photo Database / Camera

IV. Photo Database / Camera

```
@interface UIImage (Redacted)

// makes a copy at a different size
- (UIImage *)imageByScalingToSize:(CGSize)size;

// applies filter as described in Friday section
- (UIImage *)imageByApplyingFilterNamed:(NSString *)filterName;

@end
```


IV. Photo Database / Camera

```
@interface UIImage (Redacted)

// makes a copy at a different size
- (UIImage *)redacted_imageByScalingToSize:(CGSize)size;

// applies filter as described in Friday section
- (UIImage *)redacted_imageByApplyingFilterNamed:(NSString *)filterName;

@end
```


V. Blurring Images

V. Blurring Images

```
@interface UIImage (ImageEffects)
```

- (UIImage *)applyLightEffect;
- (UIImage *)applyExtraLightEffect;
- (UIImage *)applyDarkEffect;
- (UIImage *)applyTintEffectWithColor:(UIColor *)tintColor;
- (UIImage *)applyBlurWithRadius:(CGFloat)blurRadius
 tintColor:(UIColor *)tintColor
 saturationDeltaFactor:(CGFloat)saturationDeltaFactor
 maskImage:(UIImage *)maskImage;

```
@end
```


V. Blurring Images

```
@interface UIImage (ImageEffects)
```

- (UIImage *)aapl_applyLightEffect;
- (UIImage *)aapl_applyExtraLightEffect;
- (UIImage *)aapl_applyDarkEffect;
- (UIImage *)aapl_applyTintEffectWithColor:(UIColor *)tintColor;
- (UIImage *)aapl_applyBlurWithRadius:(CGFloat)blurRadius
tintColor:(UIColor *)tintColor
saturationDeltaFactor:(CGFloat)saturationDeltaFactor
maskImage:(UIImage *)maskImage;

@end

VI. UICatalog

VI. UICatalog

```
AppDelegate.h
AppDelegate.m
Constants.h
main.m
Picker/CustomPickerDataSource.h
Picker/CustomPickerDataSource.m
Picker/CustomView.h
Picker/CustomView.m
ViewControllers/AlertsViewController.h
ViewControllers/AlertsViewController.m
ViewControllers/ButtonsViewController.h
ViewControllers/ButtonsViewController.m
ViewControllers/ControlsViewController.h
ViewControllers/ControlsViewController.m
ViewControllers/ImagesViewController.h
ViewControllers/ImagesViewController.m
ViewControllers/MainViewController.h
```


VI. UICatalog

```
Objective-C/UICatalog/AAPLActivityIndicatorViewController.h
Objective-C/UICatalog/AAPLActivityIndicatorViewController.m
Objective-C/UICatalog/AAPLAlertControllerViewController.h
Objective-C/UICatalog/AAPLAlertControllerViewController.m
Objective-C/UICatalog/AAPLAppDelegate.h
Objective-C/UICatalog/AAPLAppDelegate.m
Objective-C/UICatalog/AAPLButtonViewController.h
Objective-C/UICatalog/AAPLButtonViewController.m
Objective-C/UICatalog/AAPLCustomSearchBarViewController.h
Objective-C/UICatalog/AAPLCustomSearchBarViewController.m
Objective-C/UICatalog/AAPLCustomToolbarViewController.h
Objective-C/UICatalog/AAPLCustomToolbarViewController.m
Objective-C/UICatalog/AAPLDatePickerController.h
Objective-C/UICatalog/AAPLDatePickerController.m
Objective-C/UICatalog/AAPLDefaultSearchBarViewController.h
Objective-C/UICatalog/AAPLDefaultSearchBarViewController.m
Objective-C/UICatalog/AAPLDefaultToolbarViewController.h
```


VII. Building a Framework

VII. Building a Framework

```
@interface RWRibbonView : UIView  
  
@end
```


VII. Building a Framework

```
@interface RWUIRibbonView : UIView  
  
@end
```


Conclusions

Things to Avoid

- Undocumented Tricks / Shortcuts
 - Relying on Filesystem Hierarchy (or View Hierarchy)
- Non HIG Compliant UI
- Doubling System Functionality

Naming Rules & Conventions

- Follow Apple's "Coding Guidelines for Cocoa"
- Always choose verbose & descriptive names
- Prefixing class names is recommended
 - 2 letter prefixes are reserved for Apple Frameworks. Use 3 or more uppercase letters.
- Never omit a prefix in a category method of a class you don't control
 - Use lowercase letters and an underscore as prefix.

Missing Topics

- Testing
- Discussion on Designated Initializers
- Special Cases in Subclassing
 - Class Clusters
 - Delegates or other classes implementing an interface with optional methods

Thank you

<https://github.com/sptim/flaws>