IOS开发总结

by mhmwadm

2013/4/10

1 XCode快捷键 4

2 Objective-C 4

2.1 代码混编 4

2.2 代码中字符串换行 5

2.3 不要调用[super release] 5

2.4 判断一个字符串是否包含另一个字符串: 5

2.5 没有用到类的成员变量的，都写成类方法 5

2.6 category可以用来调试 5

2.7 Category与Extension 6

2.7.0 Categories Add Methods to Existing Classes 6

2.7.1 Class Extensions Extend the Internal Implementation 7

2.8 引用 7

2.9 block 7

2.10 property重命名 10

2.11 重写description 10

2.12 Archive and serialise 10

2.13 Mutability Determines Whether a Represented Value Can Be Changed 11

2.14 Use new to Create an Object If No Arguments Are Needed for Initialization 12

2.15 Literals Offer a Concise Object-Creation Syntax 12

2.16 property的属性 13

2.17 Represent Other Values Using Instances of the NSValue Class 14

2.18 Determining Equality of Objects 14

2.19 Working with nil 15

3 IOS 16

3.1 设置圆角 16

3.2 常用的有用方法 17

3.3 图片拉伸后模糊的原因 17

3.4 使用自定义字体 17

3.5 后台运行 18

3.6 让程序退出后台时继续运行10分钟 18

3.7 关于UITableView 20

3.7.0 任意设置Cell选中状态的背景色: 20

3.7.1 取table中控件的位置 21

3.7.2 flashScrollIndicators 21

3.7.3 点击Cell中的按钮时，如何取所在的Cell： 21

3.8 \_cmd 22

3.9 一个不停震动的方法： 22

3.10 去掉app图标的发光效果 23

3.11 UIImage:stretchableImageWithLeftCapWidth:topCapHeight: 23

3.12 UIPopoverController 23

3.13 UIColor colorWithRed:green:blue:alpha: 23

3.14 禁止textField和textView的复制粘贴菜单： 24

3.15 loadView 24

3.16 GestureRecognizer相关 24

3.17 如何进入软件在app store 的页面： 25

3.18 someview显示一断时间后自动消失 25

3.19 使提示窗口在任何界面都能显示： 26

3.20 禁止程序运行时自动锁屏 26

3.21 自定义UINavigationController的返回按钮 26

3.22 改变UIAlertView背景 27

3.23 浮动提示 28

3.24 改变UITextField的背景 29

3.25 CALayer高清显示 29

3.26 CGLayer高清显示 29

3.27 用于CALayer的动画 30

3.28 取常用的地址 30

3.29 所有可用的编码 30

3.30 有时float值计算不准确，要用double 31

3.31 UINavigationBar 31

3.32 如果NSString是文件地址 31

3.33 NSPredicate 31

3.34 NSDictionary,NSMutableDictionary 32

3.35 如何改变UINavigationBar的背景 32

3.36 自IOS 6.0，为了控制旋转，要给UINavigationController写个category 33

3.37 allSubviews, allApplicationViews, pathToView 35

3.38 键盘是带按钮的pickerview 36

3.39 +(void)showAlertWithTitle:(NSString \*)title message:(id)formatstring,... 38

3.40 修改UIAlertView 38

3.41 给CALayer设置animation 39

3.42 addSubview不支持Animation 40

3.43 给keyboard增加删除按钮 41

3.44 UITextField扩展 47

3.44.0 删除前一输入的字符 47

3.44.1 TextField只要有输入，马上清掉旧值 48

3.45 CGContext常用方法 49

3.46 设置线宽 50

3.47 在CGContext中输出汉字 51

3.48 可以现成用的比较好的类： 51

3.49 简化代码用的define 51

3.50 如何加大按钮的点击范围： 52

3.51 setNavigationBarHidden先调用 52

3.52 非常规退出 52

3.53 有时iPhone或iPad检测设备旋转不准确 52

3.54 如何重写isEqual 53

3.55 添加到navigationController.view 中的视图要手动removeFromSuperview 53

4 其它 53

4.1 比较版本号 53

4.2 如果确认软件升级了 54

4.3 B/S传输文件，如果本来约定的数据结构变了： 54

4.4 日期的使用 54

4.5 关于更新 54

4.6 时间相关 55

4.7 用#if defined控制不同版本协议的使用 56

4.8 设置字节对齐方式 56

# XCode快捷键 (git 测试)

Switches between the .m and .h files: ctrl+cmd+↑

Ctrl-left/Ctrl-right to navigate words within a variable or method name

Jump to selection: shift+cmd+L

Edit All in Scope: ctrl + cmd + E

Re-Indent : ctrl + i

code fold: Alt + cmd + ←

code unfold: Alt + cmd + →

shift left : cmd +［

shift right : cmd + ］

move line up:alt + cmd +［

move line down : alt + cmd + ］

debug:

pause/continue:ctrl+cmd+Y

continue to current line:ctrl + cmd + C

# Objective-C

## 代码混编

1）obj-c的编译器处理后缀为m的文件时，可以识别obj-c和c的代码，处理mm文件可以识别obj-c,c,c++代码，但cpp文件必须只能用c/c++代码，而且cpp文件include的头文件中，也不能出现obj-c的代码，因为cpp只是cpp

2) 在mm文件中混用cpp直接使用即可，所以obj-c混cpp不是问题

3）在cpp中混用obj-c其实就是使用obj-c编写的模块是我们想要的。

如果模块以类实现，那么要按照cpp class的标准写类的定义，头文件中不能出现obj-c的东西，包括#import cocoa的。实现文件中，即类的实现代码中可以使用obj-c的东西，可以import,只是后缀是mm。

如果模块以函数实现，那么头文件要按c的格式声明函数，实现文件中，c++函数内部可以用obj-c，但后缀还是mm或m。

总结：只要cpp文件和cpp include的文件中不包含obj-c的东西就可以用了，cpp混用obj-c的关键是使用接口，而不能直接使用实现代码，实际上cpp混用的是obj-c编译后的o文件，这个东西其实是无差别的，所以可以用。而obj-c混用cpp就简单了，直接用就可以，因为obj-c的编译器支持啊

## 代码中字符串换行

NSString \*string = @"ABCDEFGHIJKL" \

"MNOPQRSTUVsWXYZ";

## 不要调用[super release]

在dealloc里要调用[super dealloc]，千万不要调用[super release]

## 判断一个字符串是否包含另一个字符串:

[str1 rangeOfString:str2].length != 0 ? @"包含" : @"不包含"

## 没有用到类的成员变量的，都写成类方法

## category可以用来调试

除了隐藏私有方法外，我主要用它截住函数。   
例1：测试时我想知道TableViewCell有没有释放，就可以这样写

@implementation UITableViewCell(dealloc)   
-(void)dealloc   
{   
 NSLog(@"%@",NSStringFromSelector(\_cmd));   
    NSArray \*array = allSubviews(self); // allSubviews是cookBook里的函数，可以取一个view的所有subView ,在这个文档后面也有  
    NSLog(@"%@",array);   
  
    [super dealloc];   
}   
@end

其它的类也可以这样写，你随便输出什么   
例2：我调试程序，觉得table的大小变了，想找到在哪改变的，这样做：

@implementation UITableView(setframe)   
-(void)setFrame:(CGRect)frame   
{   
 NSLog(%"%@",self);   
    [super setFrame: frame];   
}   
@end

## Category与Extension

### Categories Add Methods to Existing Classes

@interface ClassName (CategoryName)

@end

A category is usually declared in a separate header file and implemented in a separate source code file.

Categories can be used to declare either instance methods or class methods but are not usually suitable for declaring additional properties.

（这段可以不看：）It’s valid syntax to include a property declaration in a category interface, but it’s not possible to declare an additional instance variable in a category. This means the compiler won’t synthesize any instance variable, nor will it synthesize any property accessor methods. You can write your own accessor methods in the category implementation, but you won’t be able to keep track of a value for that property unless it’s already stored by the original class

### Class Extensions Extend the Internal Implementation

@interface ClassName ()

@end

The methods declared by a class extension are implemented in the @implementation block for the original class

Unlike regular categories, a class extension can add its own properties and instance variables to a class.

you can declare the class extension in a separate header file and import it in the source files that need it. It’s not uncommon to have two header files for a class, for example, such as XYZPerson.h and XYZPersonPrivate.h. When you release the framework, you only release the public XYZPerson.h header file.

## 引用

C++支持引用，Objective-C是从C衍变来的，不支持引用

## block

Blocks are often used to simplify common tasks such as collection enumeration, sorting and testing. They also make it easy to schedule tasks for concurrent or asynchronous execution using technologies like Grand Central Dispatch (GCD).

声明block：

void (^simpleBlock)(void);

定义block：

simpleBlock = ^{

NSLog(@"This is a block");

};

声明和定义在一起：

void (^simpleBlock)(void) = ^{

NSLog(@"This is a block");

};

调用block：

simpleBlock();

带返回值和参数的block:

double (^multiplyTwoValues)(double, double) =

^(double firstValue, double secondValue) {

return firstValue \* secondValue;

};

double result = multiplyTwoValues(2,4);

NSLog(@"The result is %f", result);

If you need to be able to change the value of a captured variable from within a block, you can use the \_\_block storage type modifier on the original variable declaration.

如果block是递归调用的，必须设置成 \_\_block。e.g.

\_\_block int (^recursiveBlock)(int) = ^(int param) {

if (param == 1) {

return 1;

}

return (param--) \* recursiveBlock(param);

};

recursiveBlock(5); // 计算5的阶乘

You Can Pass Blocks as Arguments to Methods or Functions

- (void)beginTaskWithCallbackBlock:(void (^)(void))callbackBlock;

The (void (^)(void)) specifies that the parameter is a block that doesn’t take any arguments or return any values. The implementation of the method can invoke the block in the usual way:

- (void)beginTaskWithCallbackBlock:(void (^)(void))callbackBlock {

...

callbackBlock();

}

A Block Should Always Be the Last Argument to a Method

Objects Use Properties to Keep Track of Blocks

The syntax to define a property to keep track of a block is similar to a block variable:

@interface XYZObject : NSObject

@property (copy) void (^blockProperty)(void);

@end

Note: You should specify copy as the property attribute, because a block needs to be copied to keep track of its captured state outside of the original scope.

A block property is set or invoked like any other block variable:

self.blockProperty = ^{

...

};

self.blockProperty();

typedef void(^BlockCC)(void);

// e.g.更新myTableview并显示最后一行

if (dataArray.count > 0) {

[myTableView reloadData];

// reloadData后不能直接调用scrollToRowAtIndexPath，有可能lastIndex在table中不存在，

// 所以要[NSObject performBlock: afterDelay:0];

[myTableView retain]; // 防止关闭视图，myTableView释放之后scrollToBottom出错

NSIndexPath \*lastIndex = [NSIndexPath indexPathForRow:dataArray.count-1 inSection:0];

void (^scrollToBottom)(void) = ^{

[myTableView scrollToRowAtIndexPath:lastIndex atScrollPosition:UITableViewScrollPositionBottom animated:YES];

[myTableView release];

};

[NSObject performBlock:scrollToBottom afterDelay:0];

}

## property重命名

@property (getter=isFinished) BOOL finished;

@property (readonly, getter=isFinished) BOOL finished;

In this case, the compiler will synthesize only an isFinished method, but not a setFinished: method.

@synthesize propertyName = instanceVariableName;

For example:

@synthesize firstName = ivar\_firstName;

In this case, the property will still be called firstName, and be accessible through firstName and setFirstName: accessor methods or dot syntax, but it will be backed by an instance variable called ivar\_firstName.

Important: If you use @synthesize without specifying an instance variable name, like this:

@synthesize firstName;

the instance variable will bear the same name as the property.

In this example, the instance variable will also be called firstName, without an underscore.

## 重写description

输出重要变量的值，因为调试窗口variableView有时候变量值显示不出来。

## Archive and serialise

You can use an archiver object, such as NSKeyedArchiver, to create an archive of the collected objects.

The only requirement to create an archive is that each object must support the NSCoding protocol.

If the superclass of your class does adopt NSCoding, you should invoke the superclass’s encodeWithCoder: and initWithCoder method first.

+(NSMutableDictionary \*)loadSmartPromptFromDocument

{

NSData \*data = [NSData dataWithContentsOfFile:[FilePathsManager smartPromptPathInDocument]];

NSKeyedUnarchiver \*keyedUnarchiver = [[NSKeyedUnarchiver alloc] initForReadingWithData:data];

NSMutableDictionary \* smartDic = [keyedUnarchiver decodeObjectForKey:@"SmartPrompt"];

[keyedUnarchiver finishDecoding];

[keyedUnarchiver release];

return smartDic;

}

+(void)saveSmartPrompt:(NSDictionary \*)smartDic

{

NSMutableData \*data = [NSMutableData data];

NSKeyedArchiver \*archiver = [[NSKeyedArchiver alloc] initForWritingWithMutableData:data];

[archiver encodeObject:smartDic forKey:@"SmartPrompt"];

[archiver finishEncoding];

[data writeToFile:[FilePathsManager smartPromptPathInDocument] atomically:YES];

[archiver release];

}

## Mutability Determines Whether a Represented Value Can Be Changed

Some classes define objects that are immutable. This means that the internal contents must be set when an object is created, and cannot subsequently be changed by other objects. In Objective-C, all basic NSString and NSNumber objects are immutable. If you need to represent a different number, you must use a new NSNumber instance.

Some immutable classes also offer a mutable version.

## Use new to Create an Object If No Arguments Are Needed for Initialization

It’s also possible to create an instance of a class using the new class method. This method is provided by NSObject and doesn’t need to be overridden in your own subclasses.

It’s effectively the same as calling alloc and init with no arguments:

XYZObject \*object = [XYZObject new];

// is effectively the same as:

XYZObject \*object = [[XYZObject alloc] init];

## Literals Offer a Concise Object-Creation Syntax

Some classes allow you to use a more concise, literal syntax to create instances.

You can create an NSString instance, for example, using a special literal notation, like this:

NSString \*someString = @"Hello, World!";

This is effectively the same as allocating and initializing an NSString or using one of its class factory methods:

NSString \*someString = [NSString stringWithCString:"Hello, World!"

encoding:NSUTF8StringEncoding];

The NSNumber class also allows a variety of literals:

NSNumber \*myBOOL = @YES;

NSNumber \*myFloat = @3.14f;

NSNumber \*myInt = @42;

NSNumber \*myLong = @42L;

Again, each of these examples is effectively the same as using the relevant initializer or a class factory method.

You can also create an NSNumber using a boxed expression, like this:

NSNumber \*myInt = @(84 / 2);

In this case, the expression is evaluated, and an NSNumber instance created with the result.

It’s also possible to create an array using an Objective-C literal, like this:

NSArray \*someArray = @[firstObject, secondObject, thirdObject];

You should not terminate the list of objects with nil when using this literal syntax, and in fact nil is an invalid value. You’ll get an exception at runtime if you try to execute the following code, for example:

id firstObject = @"someString";

id secondObject = nil;

NSArray \*someArray = @[firstObject, secondObject];

// exception: "attempt to insert nil object"

Objective-C also offers a literal syntax for dictionary creation, like this:

NSDictionary \*dictionary = @{

@"anObject" : someObject,

@"helloString" : @"Hello, World!",

@"magicNumber" : @42,

@"aValue" : someValue

};

## property的属性

NSString property的属性必须是copy

Delegate property的属性是assign

Dealloc 时，如果有delegate，要把delegate置成nilj,避免收到消息等

init 和dealloc中不要用self.XX=;

Any object that you wish to set for a copy property must support NSCopying, which means that it should conform to the NSCopying protocol.

If you need to set a copy property’s instance variable directly, for example in an initializer method, don’t forget to set a copy of the original object:

- (id)initWithSomeOriginalString:(NSString \*)aString {

self = [super init];

if (self) {

\_instanceVariableForCopyProperty = [aString copy];

}

return self;

}

## Represent Other Values Using Instances of the NSValue Class

you can create an NSValue instance by providing a pointer to the structure as well as an encoded Objective-C type. The @encode() compiler directive is used to create the correct Objective-C type, like this:

struct MyIntegerFloatStruct aStruct;

aStruct.i = 42;

aStruct.f = 3.14;

NSValue \*structValue = [NSValue value:&aStruct

withObjCType:@encode(MyIntegerFloatStruct)];

The standard C reference operator (&) is used to provide the address of aStruct for the value parameter.

## Determining Equality of Objects

If you need to compare whether one object represents a greater or lesser value than another object, you can’t use the standard C comparison operators > and <. Instead, the basic Foundation types, like NSNumber, NSString and NSDate, provide a compare: method:

if ([someDate compare:anotherDate] == NSOrderedAscending) {

// someDate is earlier than anotherDate

}

## Working with nil

It’s always a good idea to initialize scalar variables at the time you declare them, otherwise their initial values will contain garbage from the previous stack contents:

BOOL success = NO;

int magicNumber = 42;

This isn’t necessary for object pointers, because the compiler will automatically set the variable to nil if you don’t specify any other initial value:

XYZPerson \*somePerson;

// somePerson is automatically set to nil

A nil value is the safest way to initialize an object pointer if you don’t have another value to use, because it’s perfectly acceptable in Objective-C to send a message to nil. If you do send a message to nil, obviously nothing happens.

Note: If you expect a return value from a message sent to nil, the return value will be nil for object return types, 0 for numeric types, and NO for BOOL types.

If you need to check to make sure an object is not nil (that a variable points to an object in memory), you can either use the standard C inequality operator:

if (somePerson != nil) {

// somePerson points to an object

}

or simply supply the variable:

if (somePerson) {

// somePerson points to an object

}

If the somePerson variable is nil, its logical value is 0 (false). If it has an address, it’s not zero, so evaluates as true.

Similarly, if you need to check for a nil variable, you can either use the equality operator:

if (somePerson == nil) {

// somePerson does not point to an object

}

# IOS

## 设置圆角

myView.layer.cornerRadius = 6;

myView.layer.masksToBounds = YES;

设置圆角和阴影：(必须分两层)

CALayer \*shadowLayer = [CALayer layer];

shadowLayer.shadowColor = [UIColor blackColor].CGColor;

shadowLayer.shadowOffset = CGSizeMake(0, 0);

shadowLayer.shadowRadius = 5;

shadowLayer.shadowOpacity = 1;

shadowLayer.frame = self.bounds;

shadowLayer.backgroundColor = [UIColor clearColor ].CGColor;

shadowLayer.cornerRadius = 5;

shadowLayer.borderColor = [UIColor whiteColor].CGColor;

shadowLayer.borderWidth = 2.0;

[self.layer addSublayer:shadowLayer];

CALayer \*borderLayer = [CALayer layer];

borderLayer.cornerRadius = 5;

borderLayer.masksToBounds = YES;

borderLayer.frame = shadowLayer.bounds;

[shadowLayer addSublayer:borderLayer];

## 常用的有用方法

NSClassFromString

NSStringFromSelector

NSSelectorFromString

if ([objectA class] == [objectB class]) { //...

conformsToProtocol

## 图片拉伸后模糊的原因

UIImage stretchableimagewithleftcapwidth 有时blur：原因，像素没有和device pixel对齐.使用instrument 的Core Animation可以检测这个，勾选color misaligned images，Puts a magenta overlay over images whose source pixels aren't aligned to destination pixels， while yellow overlays are caused by stretching. Prior to iOS 4, Instruments used magenta for both.

## 使用自定义字体

1.添加对应的字体(.ttf或.odf)到工程的resurce，例如my.ttf。

2.在info.plist中添加一项 Fonts provided by application (item0对应的value为my.ttf，添加多个字体依次添加就可以了)。

3.使用时aLabel.font=[UIFontfontWithName:@"XXX" size:30]; 注意XXX不一定是my，这里是RETURN TO CASTLE。

可以用如下方法查看familyname和fontname：

NSArray \*familyNames = [UIFont familyNames];

    for( NSString \*familyNameinfamilyNames ){

        printf( "Family: %s \n", [familyNameUTF8String] );

        NSArray \*fontNames = [UIFont fontNamesForFamilyName:familyName];

        for( NSString \*fontNameinfontNames ){

            printf( "\tFont: %s \n", [fontNameUTF8String] );

        }

    }

## 后台运行

IOS允许长时间在后台运行的情况有7种：

audio

VoIP

GPS

下载新闻

和其它附属硬件进行通讯时

使用蓝牙进行通讯时

使用蓝牙共享数据时

除以上情况，程序退出时可能设置短暂运行10分钟

## 让程序退出后台时继续运行10分钟

在XXAppDelegate中增加：UIBackgroundTaskIdentifier bgTask;

- (void)applicationDidEnterBackground:(UIApplication \*)application

{

bgTask = [application beginBackgroundTaskWithExpirationHandler:^{

// 10分钟后执行这里，应该进行一些清理工作，如断开和服务器的连接等

// ...

// stopped or ending the task outright.

[application endBackgroundTask:bgTask];

bgTask = UIBackgroundTaskInvalid;

}];

if (bgTask == UIBackgroundTaskInvalid) {

NSLog(@"failed to start background task!");

}

// Start the long-running task and return immediately.

dispatch\_async(dispatch\_get\_global\_queue(DISPATCH\_QUEUE\_PRIORITY\_DEFAULT, 0), ^{

// Do the work associated with the task, preferably in chunks.

NSTimeInterval timeRemain = 0;

do{

[NSThread sleepForTimeInterval:5];

if (bgTask!= UIBackgroundTaskInvalid) {

timeRemain = [application backgroundTimeRemaining];

NSLog(@"Time remaining: %f",timeRemain);

}

}while(bgTask!= UIBackgroundTaskInvalid && timeRemain > 0); // 如果改为timeRemain > 5\*60,表示后台运行5分钟

// done!

// 如果没到10分钟，也可以主动关闭后台任务，但这需要在主线程中执行，否则会出错

dispatch\_async(dispatch\_get\_main\_queue(), ^{

if (bgTask != UIBackgroundTaskInvalid)

{

// 和上面10分钟后执行的代码一样

// ...

// if you don't call endBackgroundTask, the OS will exit your app.

[application endBackgroundTask:bgTask];

bgTask = UIBackgroundTaskInvalid;

}

});

});

}

- (void)applicationWillEnterForeground:(UIApplication \*)application

{

// 如果没到10分钟又打开了app,结束后台任务

if (bgTask!=UIBackgroundTaskInvalid) {

[application endBackgroundTask:bgTask];

bgTask = UIBackgroundTaskInvalid;

}

}

后台时，如果某些代码你不希望执行，可以加以下条件：

UIApplication \*application = [UIApplication sharedApplication];

if( application.applicationState == UIApplicationStateBackground) {

return;

}

有的app虽然我们不允许通知，但还是会弹出消息，应该是设置了定时器，到某一时间就让程序后台运行一会，从服务器更新数据，然后显示出来。

## 关于UITableView

### 任意设置Cell选中状态的背景色:

UIView \*bgView = [[UIView alloc] init];

bgView.backgroundColor = [UIColor orangeColor];

self.selectedBackgroundView = bgView;

[bgView release];

该方法设置的是纯色， 也可以使用任何图片，把selectedBackgroundView设成UIImageView。

除了上面的方法，前几天发现了一个新方法：重写UITableViewCell 的 setSelected:animated:方法

- (void)setSelected:(BOOL)selected animated:(BOOL)animated  
{  
    [super setSelected:selected animated:animated];  
 if (selected) {  
 self.backgroundColor = RGB(224, 152, 40);  
 }  
 else {  
 self.backgroundColor = [UIColor clearColor];  
 }  
}

### 取table中控件的位置

如果Table中有控件，这里以switch为例（适合其它可修改值的各种控件），要在switch的UIControlEventValueChanged事件的处理方法里把值记录下来。以下方法是不可取的：在执行的最后把所有cell遍历一遍，处理各控件的值。因为没显示出来的cell，是取不到的，当然也就取不到该cell里的控件。所以正确的做法是，在控件可见时，如果值变了，立即处理。当然，如果你的Cell少，不会出现隐藏的情况就随便了。

### flashScrollIndicators

这个很有用，闪一下滚动条，暗示是否有可滚动的内容。可以在ViewDidAppear或[table reload]之后调用。

### 点击Cell中的按钮时，如何取所在的Cell：

-(void)OnTouchBtnInCell:(UIButton \*)btn   
{   
 CGPoint point = btn.center;   
 point = [table convertPoint:point fromView:btn.superview];   
 NSIndexPath\* indexpath = [table indexPathForRowAtPoint:point];   
 UITableViewCell \*cell = [table cellForRowAtIndexPath:indexpath];   
 ...

// 也可以通过一路取btn的父窗口取到cell，但如果cell下通过好几层subview才到btn,就要取好几次 superview

// 所以我用上面的方法，比较通用。这种 方法也适用于其它控件。   
}

## \_cmd

表示该方法的selector，可以赋值给SEL类型的变量，可以做为参数传递。

例如一个显示消息的方法：

-(void)ShowNotifyWithString:(NSString \*)notifyString fromMethod:(SEL) originalMethod;

originalMethod就是调用这个方法的selector。

调用：

NSString \*stmp = @"test";

[self ShowNotifyWithString:stmp fromMethod:\_cmd];

如何记录当前方法名称：

NSLog(NSStringFromSelector(\_cmd));

## 一个不停震动的方法：

// 定义一个回调函数，震动结束时再次发出震动

void MyAudioServicesSystemSoundCompletionProc (SystemSoundID  ssID,void \*clientData)

{

      BOOL\* iShouldKeepBuzzing = clientData;

      if (\*iShouldKeepBuzzing) {        AudioServicesPlaySystemSound(kSystemSoundID\_Vibrate);

      } else {

           //Unregister, so we don't get called again...

           AudioServicesRemoveSystemSoundCompletion(kSystemSoundID\_Vibrate);

      }

}

以下为调用的代码：

BOOL iShouldKeepBuzzing = YES;

AudioServicesAddSystemSoundCompletion (

  kSystemSoundID\_Vibrate,

  NULL,

  NULL,

  MyAudioServicesSystemSoundCompletionProc,

&iShouldKeepBuzzing );

AudioServicesPlaySystemSound (kSystemSoundID\_Vibrate);

## 去掉app图标的发光效果

info.plist里增加Icon already includes gloss effects，值设为YES

## UIImage:stretchableImageWithLeftCapWidth:topCapHeight:

有时图片模糊(blur)的原因:像素没有和device pixel对齐.使用instrument 的Core Animation可以检测这个，勾选"color misaligned images"，如果图片显示为红紫色，就是没有对齐

## UIPopoverController

如果是用presentPopoverFromBarButtonItem显示的，设备旋转时，popover可以自动调整位置；如果是用presentPopoverFromRect显示的， 需要present again

-(void)didRotateFromInterfaceOrientation:(UIInterfaceOrientation)fromInterfaceOrientation

{

[aPopover presentPopoverFromRect:targetRect.frame inView:self.view permittedArrowDirections:UIPopoverArrowDirectionAny animated:YES];

}

## UIColor colorWithRed:green:blue:alpha:

这个方法的参数必须用浮点型。

假如使用Xcode自带的取颜色的工具，取到的RGB值分别为：25，25，25，

传给上述方法的参数应为25/255.0或25.0/255。如果用整型25/255，经过取整，小数部分没有了，显示出来的颜色和取到的是不一样的。可以定义一个宏：

#define RGB(A,B,C) [UIColor colorWithRed:A/255.0 green:B/255.0 blue:C/255.0 alpha:1.0]

然后用RGB(25,25,25)就可以了

## 禁止textField和textView的复制粘贴菜单：

-(BOOL)canPerformAction:(SEL)action withSender:(id)sender

{

     if ([UIMenuController sharedMenuController]) {

       [UIMenuController sharedMenuController].menuVisible = NO;

     }

     return NO;

}

## loadView

如果重载loadView，一定要在这个方法里产生一个self.view。可以调用[super loadView]，也可以使用alloc+init。

错误情况举例：loadView 直接调用self.view.alpha = 0.5; 因为self.view为nil，self.view.alpha这句又会调用loadView，也就是loadView不断调用loadView，进入了死循环

## GestureRecognizer相关

1．一个View有GestureRecognizer又有按钮（或其它需要处理action event的控件）时，有时按钮不灵敏，解决办法：

-(BOOL)gestureRecognizer:(UIGestureRecognizer \*)gestureRecognizer shouldReceiveTouch:(UITouch \*)touch

{

     CGPoint pt      = [touch locationInView:baseView];

     UIView \*btn     = [baseView viewWithTag:TAG\_MYBTN];

     CGPoint ptInbtn = [baseView convertPoint:pt toView:btn];

     return ![btn pointInside:ptInbtn withEvent:nil];

}

2．实现某个view点一下就移除时，要防止移除两次。（此方法适用于希望GestureRecognizer只执行一次的情况）

-(void)OnTapViewTobeRemoved:(UITapGestureRecognizer \*)sender

{

     if (!sender.enabled) {

           return;

     }

     sender.enabled = NO;

     [sender.view removeFromSuperview];

}

## 如何进入软件在app store 的页面：

先用iTunes Link Maker找到软件在访问地址，格式为itms-apps://ax.itunes.apple.com/...，然后

#define  ITUNESLINK   @"itms-apps://ax.itunes.apple.com/..."

NSURL \*url = [NSURL URLWithString:ITUNESLINK];

if([[UIApplication sharedApplication] canOpenURL:url]){

     [[UIApplication sharedApplication] openURL:url];

}

如果把上述地址中itms-apps改为http就可以在浏览器中打开了。可以把这个地址放在自己的网站里，链接到app store。

iTunes Link Maker地址：<http://itunes.apple.com/linkmaker>

## someview显示一断时间后自动消失

[self performSelector:@selector(dismissView:) withObject:someview afterDelay:2];

这么写比用NSTimer代码少，不过哪种都行的，这里只是提供一种不同的方法

## 使提示窗口在任何界面都能显示：

［self.navigationController.view addSubview:(自定义的提示窗口)］

或用UIAlertView

## 禁止程序运行时自动锁屏

[[UIApplication sharedApplication] setIdleTimerDisabled:YES];

## 自定义UINavigationController的返回按钮

navigationItem的backBarButtonItem的action是不会执行的.无论怎么改，除了popViewController什么都不执行。   
例如：   
UIBarButtonItem \*backButton = [[UIBarButtonItem alloc] initWithTitle:@"返回" style:UIBarButtonItemStylePlain target:self action:@selector(onComingback)];   
self.navigationItem.backBarButtonItem= backButton;   
在下一级视图中点“返回”，onComingback也是不会执行的。target和action都被忽略了，所以参数用nil就行了   
要想在点“返回”时执行某段代码，只能自己做一个像返回按钮那样的UIBarButtonItem,图片是需要自己做的。self.navigationItem.leftBarButtonItem= custombackButton; // custombackButton的方法中包含popViewController和你想加的其它代码

+(UIBarButtonItem \*)backBarBtnItemWithTitle:(NSString \*)title

target:(id)target

selector:(SEL)selector

frame:(CGRect)frame

{

UIButton\* backBtn = [UIButton buttonWithType:UIButtonTypeCustom];

backBtn.titleLabel.font = [UIFont boldSystemFontOfSize:[UIFont smallSystemFontSize]];

backBtn.titleLabel.textColor = [UIColor whiteColor];

backBtn.titleEdgeInsets = UIEdgeInsetsMake(0, 8.0, 0, 3.0);

[backBtn setTitle:title forState:UIControlStateNormal];

[backBtn addTarget:target action:selector forControlEvents:UIControlEventTouchUpInside];

UIImage\* buttonImage = [[UIImage imageNamed:@"backBtn"] stretchableImageWithLeftCapWidth:14.0 topCapHeight:0.0];

backBtn.frame = frame;

[backBtn setBackgroundImage:buttonImage forState:UIControlStateNormal];

return [[[UIBarButtonItem alloc] initWithCustomView:backBtn] autorelease];

}

## 改变UIAlertView背景

UIAlertView默认是半透明的，会透出背景的内容，有时看着有些混乱。可以写个改变背景的方法changeBackground。  
@interface UIAlertView (changeBK)  
- (void)changeBackground;  
@end

@implementation UIAlertView (changeBK)  
- (void)changeBackground  
{  
 for(UIView \* v in [self subviews]){  
 if ([v isKindOfClass:[UIImageView class]]) {  
 UIImage \*theImage = [UIImage imageNamed:@"AlertView.png"];  
 ((UIImageView\*)v).image = theImage;  
 break;  
 }  
 }  
}  
@end  
在[alertView show]之后或willPresentAlertView:中调用即可。  
// UIAlertView \*alertView = [UIAlertView alloc] init ...  
...  
[alertView show];  
[alertView changeBackground];

## 浮动提示

有时需要显示一个视图，几秒后再消失的功能，这个功能也可以写成category

@interface UIView (AutoRemove)  
- (void)removeAfterDelay:(NSTimeInterval)time;  
- (void)removeImediatly;  
@end  
  
@implementation UIView (AutoRemove)  
- (void)removeAfterDelay:(NSTimeInterval)time  
{  
 [self performSelector:@selector(removeFromSuperview) withObject:nil afterDelay:time];  
}  
  
- (void)removeImediatly  
{  
 [NSObject cancelPreviousPerformRequestsWithTarget:self selector:@selector(removeFromSuperview) object:nil];  
 [self removeFromSuperview];  
}  
@end

## 改变UITextField的背景

可以给textField加好看的边框等

@interface UITextField (changeBK)  
-(void)orangeBackground;  
@end  
  
@implementation UITextField (changeBK)  
-(void)orangeBackground  
{  
 self.background = [[UIImage imageNamed:@"fieldBK.png"] stretchableImageWithLeftCapWidth:5 topCapHeight:5];  
}  
@end

## CALayer高清显示

if ([myLayer respondsToSelector:@selector(setContentsScale:)]) {

myLayer.contentsScale = [[UIScreen mainScreen] scale];

}

## CGLayer高清显示

This is how to draw a CGLayer correctly for all resolutions.

1. When first creating the layer, you need to calculate the correct bounds by multiplying the dimensions with the scale:

int width = 25;

int height = 25;

float scale = [self contentScaleFactor];

CGRect bounds = CGRectMake(0, 0, width \* scale, height \* scale);

CGLayer layer = CGLayerCreateWithContext(context, bounds.size, NULL);

CGContextRef layerContext = CGLayerGetContext(layer);

1. You then need to set the correct scale for your layer context:

CGContextScaleCTM(layerContext, scale, scale);

1. If the current device has a retina display, all drawing made to the layer will now be drawn twice as large.
2. When you finally draw the contents of your layer, make sure you use CGContextDrawLayerInRect and supply the unscaled CGRect:
3. CGRect bounds = CGRectMake(0, 0, width, height);

CGContextDrawLayerInRect(context, bounds, layerContext);

## 用于CALayer的动画

CALayer有默认动画效果，如果想完全用自已的，先要removeAllAnimations，再加上自设的animation。如果UIView上有一个CALayer，UIView的的内容没有更新，只有CALayer,则只重绘CALayer就可以，调用[layer setNeedsDisplay]。之前CALayer的内容会自动清除。比VC++高级，对于要更新的区域，不用手动擦除旧的

## 取常用的地址

Document：

NSArray \*paths = NSSearchPathForDirectoriesInDomains(NSDocumentDirectory, NSUserDomainMask, YES);

NSString \*documentsDirectory = [paths objectAtIndex:0];

Library：

NSArray \*paths = NSSearchPathForDirectoriesInDomains(NSLibraryDirectory, NSUserDomainMask, YES);

NSString \*libraryDirectory = [paths objectAtIndex:0];

## 所有可用的编码

[NSString availableStringEncodings]

## 有时float值计算不准确，要用double

## UINavigationBar

在UINavigationController中，一般是用不着navigationBar的，只需用navigationItem即可，只有设置tintColor,bayStyle时才会用到。

## 如果NSString是文件地址

[string pathExtension]

[string lastPathComponent]

## NSPredicate

1.从 数组1中过滤出数组2中没有的对象

NSArray \*arrayFilter = [NSArray arrayWithObjects:@"abc1", @"abc2", nil];

NSArray \*arrayContent = [NSArray arrayWithObjects:@"a1", @"abc1", @"abc4", @"abc2", nil];

NSPredicate \*thePredicate = [NSPredicate predicateWithFormat:@"NOT (SELF in %@)", arrayFilter];

[arrayContent filterUsingPredicate:thePredicate];

这样arrayContent过滤出来的就是不包含 arrayFilter中的所有item了。

**match的用**法

1. 简单比较

NSString \*match = @"imagexyz-999.png";

NSPredicate \*predicate = [NSPredicate predicateWithFormat:@"SELF == %@", match];

NSArray \*results = [directoryContents filteredArrayUsingPredicate:predicate];﻿

2. match里like的用法（类似Sql中的用法）

NSString \*match = @"imagexyz\*.png";

NSPredicate \*predicate = [NSPredicate predicateWithFormat:@"SELF like %@", match];

NSArray \*results = [directoryContents filteredArrayUsingPredicate:predicate];﻿

3. 大小写比较

［c］表示忽略大小写，［d］表示忽略重音，可以在一起使用，如下：

NSString \*match = @"imagexyz\*.png";

NSPredicate \*predicate = [NSPredicate predicateWithFormat:@"SELF like[cd] %@", match];

NSArray \*results = [directoryContents filteredArrayUsingPredicate:predicate];﻿

4.使用正则

NSString \*match = @"imagexyz-\\d{3}\\.png"; //imagexyz－123.png

NSPredicate \*predicate = [NSPredicate predicateWithFormat:@"SELF matches %@", match];

NSArray \*results = [directoryContents filteredArrayUsingPredicate:predicate];﻿

总结：

1） 当使用聚合类的操作符时是可以不需要循环的

2）当使用单个比较类的操作符时可以一个循环来搞定

PS，例子 一中尝试使用[@"SELF CONTAINS %@", arrayFilter] 来过滤会挂调，因为CONTAINS时字符串比较操作符，不是集合操作符。

## NSDictionary,NSMutableDictionary

NSMutableArray \*objects = [myDictionary objectsForKeys:myKeys notfoundMarker:[NSNull null]]; // 这里不能用nil

## 如何改变UINavigationBar的背景

针对较旧的IOS,大概是IOS 4.3及以下：

@implementation UINavigationBar (CustomImage)

- (void)drawRect:(CGRect)rect

{

UIImage \*image = [[UIImage imageNamed: @"navBar"] stretchableImageWithLeftCapWidth:5 topCapHeight:0];

[image drawInRect:CGRectMake(0, 0.5, self.frame.size.width, self.frame.size.height)];

}

@end

针对所有版本IOS:

@implementation UINavigationBar (selBackground)

-(void)setToCustomImage

{

if ([[UIDevice currentDevice] systemVersion].floatValue >= 5.0 ) {

[self setBackgroundImage:[UIImage imageNamed:@"navBar"] forBarMetrics:UIBarMetricsDefault];

}

else

{

self.layer.contents = (id)[UIImage imageNamed:@"navBar"].CGImage;

}

}

@end

## 自IOS 6.0，为了控制旋转，要给UINavigationController写个category

**（原因见以下黄色背景的内容）**

@interface UINavigationController (Rotate)

@end

@implementation UINavigationController (Rotate)

- (NSUInteger)supportedInterfaceOrientations

{

return [self.topViewController supportedInterfaceOrientations];

}

- (BOOL)shouldAutorotate

{

return [self.topViewController shouldAutorotate];

}

@end

对程序里所有不希望旋转的viewController,

- (BOOL)shouldAutorotate

{

return NO;

}

* Autorotation is changing in iOS 6. In iOS 6, the shouldAutorotateToInterfaceOrientation: method of [UIViewController](file:///Users/wenhua/Library/Developer/Shared/Documentation/DocSets/com.apple.adc.documentation.AppleiOS6.1.iOSLibrary.docset/Contents/Resources/Documents/documentation/UIKit/Reference/UIViewController_Class/Reference/Reference.html#//apple_ref/occ/cl/UIViewController) is deprecated. In its place, you should use the supportedInterfaceOrientationsForWindow: and [shouldAutorotate](file:///Users/wenhua/Library/Developer/Shared/Documentation/DocSets/com.apple.adc.documentation.AppleiOS6.1.iOSLibrary.docset/Contents/Resources/Documents/documentation/UIKit/Reference/UIViewController_Class/Reference/Reference.html#//apple_ref/occ/instm/UIViewController/shouldAutorotate) methods.
  + More responsibility is moving to the app and the app delegate. Now, iOS containers (such as UINavigationController) do not consult their children to determine whether they should autorotate. By default, an app and a view controller’s supported interface orientations are set to UIInterfaceOrientationMaskAll for the iPad idiom and UIInterfaceOrientationMaskAllButUpsideDown for the iPhone idiom.

A view controller’s supported interface orientations can change over time—even an app’s supported interface orientations can change over time. The system asks the top-most full-screen view controller (typically the root view controller) for its supported interface orientations whenever the device rotates or whenever a view controller is presented with the full-screen modal presentation style. Moreover, the supported orientations are retrieved only if this view controller returns YES from its shouldAutorotate method. The system intersects the view controller’s supported orientations with the app’s supported orientations (as determined by the Info.plist file or the app delegate’s application:supportedInterfaceOrientationsForWindow: method) to determine whether to rotate.

* + The system determines whether an orientation is supported by intersecting the value returned by the app’s supportedInterfaceOrientationsForWindow: method with the value returned by the supportedInterfaceOrientations method of the top-most full-screen controller.
  + The setStatusBarOrientation:animated: method is not deprecated outright. It now works only if the [supportedInterfaceOrientations](file:///Users/wenhua/Library/Developer/Shared/Documentation/DocSets/com.apple.adc.documentation.AppleiOS6.1.iOSLibrary.docset/Contents/Resources/Documents/documentation/UIKit/Reference/UIViewController_Class/Reference/Reference.html#//apple_ref/occ/instm/UIViewController/supportedInterfaceOrientations) method of the top-most full-screen view controller returns 0. This makes the caller responsible for ensuring that the status bar orientation is consistent.
* The willRotateToInterfaceOrientation:duration:, willAnimateRotationToInterfaceOrientation:duration:, and didRotateFromInterfaceOrientation: methods are no longer called on any view controller that makes a full-screen presentation over itself—for example, presentViewController:animated:completion:.
  + You should make sure that your apps are not using these methods to manage the layout of any subviews. Instead, they should use the view controller’s viewWillLayoutSubviews method and adjust the layout using the view’s bounds rectangle.

In iOS 6, the viewWillUnload and viewDidUnload methods of UIViewController are now deprecated. If you were using these methods to release data, use the didReceiveMemoryWarning method instead. You can also use this method to release references to the view controller’s view if it is not being used. You would need to test that the view is not in a window before doing this.

## allSubviews, allApplicationViews, pathToView

NSArray \*allSubviews(UIView \*aView)

{

NSArray \*results = [aView subviews];

for (UIView \*eachView in [aView subviews])

{

NSArray \*riz = allSubviews(eachView);

if (riz) results = [results arrayByAddingObjectsFromArray:riz];

}

return results;

}

// Return all views throughout the application

NSArray \*allApplicationViews()

{

NSArray \*results = [[UIApplication sharedApplication] windows];

for (UIWindow \*window in [[UIApplication sharedApplication] windows])

{

NSArray \*riz = allSubviews(window);

if (riz) results = [results arrayByAddingObjectsFromArray: riz];

}

return results;

}

// Return an array of parent views from the window down to the view

NSArray \*pathToView(UIView \*aView)

{

NSMutableArray \*array = [NSMutableArray arrayWithObject:aView];

UIView \*view = aView;

UIWindow \*window = aView.window;

while (view != window)

{

view = [view superview];

[array insertObject:view atIndex:0];

}

return array;

}

## 键盘是带按钮的pickerview

UIView \*keyBoardView = [[UIView alloc] initWithFrame:CGRectMake(0, 0, CGRectGetWidth(self.view.bounds), 220)];

//键盘上方加工具条

UIToolbar \* keyToolBar = [[UIToolbar alloc]initWithFrame:CGRectMake(0, 0, CGRectGetWidth(self.view.bounds), 38)];

[keyToolBar setTranslucent:NO];

[keyToolBar setTintColor:RGB(212, 214, 218)];

UIButton \*closeButton = [UIButton buttonWithType:UIButtonTypeCustom];

closeButton.frame = CGRectMake(260, 5.0, 50.0f, 28.0);

closeButton.titleLabel.font = [UIFont fontWithName:FONT\_ARIAL size:14.0];

[closeButton addTarget:self action:@selector(closeKeyboard:) forControlEvents:UIControlEventTouchUpInside];

[closeButton setTitle:T\_CONFIRM forState:UIControlStateNormal];

[closeButton setBackgroundImage:[[UIImage imageNamed:@"selBack"] stretchableImageWithLeftCapWidth:8 topCapHeight:13]

forState:UIControlStateNormal];

[keyToolBar addSubview:closeButton];

[keyBoardView addSubview:keyToolBar];

[keyToolBar release];

UIPickerView \*picker = [[UIPickerView alloc] initWithFrame:CGRectMake(0, CGRectGetMaxY(keyToolBar.frame), 0, 200)];

picker.showsSelectionIndicator = YES;

picker.dataSource = self;

picker.delegate = self;

[keyBoardView addSubview:picker];

self.pickerView = picker;

[picker release];

companyTF.inputView = keyBoardView;

[keyBoardView release];

## +(void)showAlertWithTitle:(NSString \*)title message:(id)formatstring,...

+(void)showAlertWithTitle:(NSString \*)title message:(id)formatstring,...

{

id outstring = nil;

if (formatstring) {

va\_list arglist;

va\_start(arglist, formatstring);

outstring = [[[NSString alloc] initWithFormat:formatstring arguments:arglist] autorelease];

va\_end(arglist);

}

NSLog(@"%@:%@",title,outstring);

UIAlertView \*av = [[UIAlertView alloc] initWithTitle:title

message:outstring

delegate:[Adapter instance]

cancelButtonTitle:T\_CONFIRM

otherButtonTitles:nil];

[av show];

[av changeBackground];

[av release];

}

## 修改UIAlertView

-(void)willPresentAlertView:(UIAlertView \*)alertView

{

[alertView changeBackground];

switch (alertView.tag) {

case TAG\_ALERT:

for (UIView \*view in alertView.subviews) {

if ([view isKindOfClass:[UILabel class]]) {

UILabel \* label = (UILabel \*)view;

if ([label.text isEqualToString:@"今天天气真好"]) {

label.textAlignment = UITextAlignmentLeft;

// 其它的属性也可以改

}

}

}

break;

default:

break;

}

}

## 给CALayer设置animation

CATransition \*animation = [CATransition animation];

[animation setDuration:0.35];

[animation setTimingFunction:UIViewAnimationCurveEaseInOut];

animation.type = kCATransitionPush;

animation.subtype = kCATransitionFromLeft;

animation.fillMode = kCAFillModeForwards;

[animation setRemovedOnCompletion:YES];

[queryLayer addAnimation:animation forKey:kCAOnOrderIn];

animation = [CATransition animation];

[animation setDuration:0.35];

[animation setTimingFunction:UIViewAnimationCurveEaseInOut];

animation.type = kCATransitionPush;

animation.subtype = kCATransitionFromLeft;

animation.fillMode = kCAFillModeForwards;

[animation setRemovedOnCompletion:YES];

[queryLayer addAnimation:animation forKey:kCAOnOrderOut];

## addSubview不支持Animation

addSubView和removeFromSuperview是不支持animation的。要实现动画效果，可以利用alpha 这个变量。

// 显示：

floatView = [[FloatNotifyView alloc] initWithFrame:CGRectMake(35, 315, 250, 50)];

floatView.tag = TAG\_FLOAT\_NOTIFY;

floatView.alpha = 0;

[self.view addSubview:floatView];

[UIView animateWithDuration:0.2f

delay:0

options:UIViewAnimationOptionCurveEaseIn

animations:^{

floatView.alpha = 1;

}

completion:^(BOOL finished) {

}];

[floatView release];

// 消失：

-(void)hideInfo

{

[NSObject cancelPreviousPerformRequestsWithTarget:self selector:@selector(hideInfo) object:nil];

[UIView animateWithDuration:0.4f

delay:0

options:UIViewAnimationOptionCurveEaseOut

animations:^{

self.alpha = 0;

}

completion:^(BOOL finished) {

[self removeFromSuperview];

}];

}

## 给keyboard增加删除按钮

数字键盘没有关闭键盘的按钮，以下代码把”Done”拆成两个按钮了。

（后来觉得这个办法太麻烦了，不如在键盘上加一行工具栏，工具栏上加关闭按钮）

[[NSNotificationCenter defaultCenter] addObserver:self

selector:@selector(keyboardWillShowOnDelay:)

name:UIKeyboardWillShowNotification

object:nil];

[[NSNotificationCenter defaultCenter] addObserver:self

selector:@selector(keyboardWillHide:)

name:UIKeyboardWillHideNotification

object:nil];

- (void)keyboardWillShowOnDelay:(NSNotification \*)notification

{

[self performSelector:@selector(keyboardWillShow:) withObject:nil afterDelay:0];

}

- (void)keyboardWillShow:(NSNotification \*)notification

{

UIView \*foundKeyboard = nil;

UIWindow \*keyboardWindow = nil;

for (UIWindow \*testWindow in [[UIApplication sharedApplication] windows])

{

if (![[testWindow class] isEqual:[UIWindow class]])

{

keyboardWindow = testWindow;

break;

}

}

if (!keyboardWindow) return;

for (UIView \*possibleKeyboard in [keyboardWindow subviews])

{

//NSLog(@"%@",[possibleKeyboard description]);

//iOS3

if ([[possibleKeyboard description] hasPrefix:@"<UIKeyboard"])

{

foundKeyboard = possibleKeyboard;

break;

}

else

{

// iOS 4 sticks the UIKeyboard inside a UIPeripheralHostView.

if ([[possibleKeyboard description]hasPrefix:@"<UIPeripheralHostView"]) {

BOOL didFound = NO;

for (UIView \*tmpView in [possibleKeyboard subviews]) {

if ([[tmpView description] hasPrefix:@"<UIKeyboard"]) {

foundKeyboard = tmpView;

didFound = YES;

break;

}

}

if (didFound) {

break;

}

}

}

}

if (foundKeyboard)

{

UIButton \*backButton = [UIButton buttonWithType:UIButtonTypeCustom];

backButton.tag = 110;

backButton.frame = CGRectMake(215, 163, 52, 53);

backButton.adjustsImageWhenHighlighted = NO;

UIImage \*darkImage = [[UIImage imageNamed:@"dark.png"] stretchableImageWithLeftCapWidth:5 topCapHeight:0];

UIImage \*darkCornerImage = [[UIImage imageNamed:@"darkCorner.png"] stretchableImageWithLeftCapWidth:5 topCapHeight:0];

UIImage \*brightImage = [[UIImage imageNamed:@"bright.png"] stretchableImageWithLeftCapWidth:5 topCapHeight:0];

UIImage \*brightCornerImage = [[UIImage imageNamed:@"brightCorner.png"] stretchableImageWithLeftCapWidth:5 topCapHeight:0];

[backButton setBackgroundImage:darkImage forState:UIControlStateNormal];

[backButton setBackgroundImage:brightImage forState:UIControlStateHighlighted];

[backButton addTarget:self action:@selector(backButton:)forControlEvents:UIControlEventTouchUpInside];

[backButton setTitle:@"删除" forState:UIControlStateNormal];

[backButton setTitleColor:[UIColor blackColor] forState:UIControlStateHighlighted];

[foundKeyboard addSubview:backButton];

UIButton \*doneButton = [UIButton buttonWithType:UIButtonTypeCustom];

doneButton.tag = 111;

doneButton.frame = CGRectMake(215+52, 163, 53, 53);

doneButton.adjustsImageWhenHighlighted = NO;

[doneButton setBackgroundImage:darkCornerImage forState:UIControlStateNormal];

[doneButton setBackgroundImage:brightCornerImage forState:UIControlStateHighlighted];

[doneButton addTarget:self action:@selector(doneButton:)forControlEvents:UIControlEventTouchUpInside];

[doneButton setTitle:@"隐藏" forState:UIControlStateNormal];

[doneButton setTitleColor:[UIColor blackColor] forState:UIControlStateHighlighted];

[foundKeyboard addSubview:doneButton];

}

}

- (void)keyboardWillHide:(NSNotification \*)notification

{

UIView \*foundKeyboard = nil;

UIWindow \*keyboardWindow = nil;

for (UIWindow \*testWindow in [[UIApplication sharedApplication] windows])

{

if (![[testWindow class] isEqual:[UIWindow class]])

{

keyboardWindow = testWindow;

break;

}

}

if (!keyboardWindow) return;

for (UIView \*possibleKeyboard in [keyboardWindow subviews])

{

//NSLog(@"%@",[possibleKeyboard description]);

//iOS3

if ([[possibleKeyboard description] hasPrefix:@"<UIKeyboard"])

{

foundKeyboard = possibleKeyboard;

break;

}

else

{

// iOS 4 sticks the UIKeyboard inside a UIPeripheralHostView.

if ([[possibleKeyboard description]hasPrefix:@"<UIPeripheralHostView"]) {

BOOL didFound = NO;

for (UIView \*tmpView in [possibleKeyboard subviews]) {

if ([[tmpView description] hasPrefix:@"<UIKeyboard"]) {

foundKeyboard = tmpView;

didFound = YES;

break;

}

}

if (didFound) {

break;

}

}

}

}

if (foundKeyboard)

{

UIView \*back = [foundKeyboard viewWithTag:110];

[back removeFromSuperview];

UIView \*done = [foundKeyboard viewWithTag:111];

[done removeFromSuperview];

}

}

## UITextField扩展

### 删除前一输入的字符

@implementation UITextFieldBack

- (void)myDeleteBackward {

if ([self conformsToProtocol:@protocol(UITextInput)]){

// iOS5 and later

[self deleteBackward];

// Or do below line if you are not deploy-targeting 5.0 or above and want to avoid warnings

//[textField performSelector:@selector(deleteBackward)];

} else {

@try {

//check current selected range

NSRange selectedRange = [[self valueForKey:@"selectionRange"] rangeValue];

if (selectedRange.location == NSNotFound){

selectedRange = NSMakeRange([[self text] length], 0);

}

if (selectedRange.location < 1){

return;

}

//delete one char

NSRange deleteRange = (selectedRange.length > 0) ? selectedRange : NSMakeRange(selectedRange.location - 1, 1);

self.text = [self.text stringByReplacingCharactersInRange:deleteRange withString:@""];

//adjust the selected range to reflect the changes

selectedRange.location = deleteRange.location;

selectedRange.length = 0;

[self setValue:[NSValue valueWithRange:selectedRange] forKey:@"selectionRange"];

} @catch (NSException \*exception) {

NSLog(@"deleteBackword failed but catched. %@", exception);

} @finally {}

}

}

@end

### TextField只要有输入，马上清掉旧值

@interface UITextFieldEx : UITextField

{

BOOL deleteFirst;

}

@property(nonatomic)BOOL deleteFirst;

@end

@implementation UITextFieldEx

@synthesize deleteFirst;

-(id)initWithFrame:(CGRect)frame

{

self = [super initWithFrame:frame];

if (self) {

deleteFirst = YES;

}

return self;

}

@end

在textFieldDelegate中实现：

-(void)textFieldDidBeginEditing:(UITextField \*)textField

{

UITextFieldEx \*field = (UITextFieldEx \*)textField;

field.deleteFirst = YES;

}

-(BOOL)textField:(UITextField \*)textField shouldChangeCharactersInRange:(NSRange)range replacementString:(NSString \*)string

{

UITextFieldEx \*field = (UITextFieldEx \*)textField;

if (field.deleteFirst) {

field.text = @"";

field.deleteFirst = NO;

}

return YES;

}

## CGContext常用方法

（可以不看，是我自己refresh memory的）

CGContextSaveGState(context);

CGContextSetShouldAntialias(context, YES);

CGContextSetLineDash(context, 0, NULL, 0);

SETRGBSTROKECOLOR(context, 108, 0, 0);

SETLINEWIDTH(context,1.0);

CGPoint gridLines[] =

{

CGPointMake(0.0, CGRectGetMaxY(frame)),

CGPointMake(xRight, CGRectGetMaxY(frame)),

};

CGContextStrokeLineSegments(context, gridLines, sizeof(gridLines)/sizeof(gridLines[0]));

CGContextSaveGState(context);

CGContextSetShouldAntialias(context, YES);

CGContextSelectFont(context, C\_FONT\_ARIAL, 12.0, kCGEncodingMacRoman);

CGContextSetTextMatrix(context, CGAffineTransformMake(

1.0f, 0.0f,

0.0f, -1.0f,

0.0f, 0.0f ));

CGContextSetTextDrawingMode(context, kCGTextFill);

CGContextRestoreGState(context);

## 设置线宽

如果是retina屏，lineWidth设为1,实际显示的宽度是2个像素，这里进行一下处理：

#define SETLINEWIDTH(ctx,w) CGContextSetLineWidth(ctx, w/[UIScreen mainScreen].scale)

## 在CGContext中输出汉字

CGContextShowTextAtPoint是不支持汉字的，需要用NSString的drawAtPoint或drawInRect方法

## 可以现成用的比较好的类：

（可以不看）

PageControlView

IKSegmentedControl

UICheckBox

MKHorizMenu

TitleSlideView

TradeNotifyView

FloatNotifyView

BubbleNotify

UIButton (hitTest)

NSObject (Blocks)

## 简化代码用的define

#define SETRGBSTROKECOLOR(ctx,R,G,B) CGContextSetRGBStrokeColor(context, R/255.0, G/255.0, B/255.0, 1.0)

#define SETRGBFILLCOLOR(ctx,R,G,B) CGContextSetRGBFillColor(context, R/255.0, G/255.0, B/255.0, 1.0)

#define \_ADDOBSERVER(TITLE, SELECTOR) [[NSNotificationCenter defaultCenter] addObserver:self selector:SELECTOR name:TITLE object:nil]

#define \_REMOVEOBSERVER(id) [[NSNotificationCenter defaultCenter] removeObserver:id]

#define \_POSTNOTIFY(TITLE,OBJ,PARAM) [[NSNotificationCenter defaultCenter] postNotificationName:TITLE object:OBJ userInfo:PARAM]

## 如何加大按钮的点击范围：

1. 把UIButton的frame 设置的大一些，然后给UIButton设置一个小些的图片

[tmpBtn setImageEdgeInsets:UIEdgeInsetsMake(5, 5, 5, 5)];

// 注意这里不能用setBackgroundImage

[tmpBtn setImage:[UIImage imageNamed:@"testBtnImage"] forState:UIControlStateNormal];

2.使用类UIButton (hitTest)。这是一个开源类

## setNavigationBarHidden先调用

要隐藏NavigationBar，更改视图的布局的话，必须先调用setNavigationBarHidden再调整其它控件。否则控件是按有NavigationBar的条件分布的，再隐藏NavigationBar,就达不到预期效果了。

~~CFTree 保存和读取~~

~~Method name不能用copy或者new，不然影响自动分析时计数~~

## 非常规退出

苹果不建议程序主动退出，但还是有一个函数可以实现这个效果：

exit(0)，

不过这个函数不触发applicationWillResignActive等AppDelegate method

## 有时iPhone或iPad检测设备旋转不准确

加上这个通知就可以了：

\_ADDOBSERVER(UIDeviceOrientationDidChangeNotification, @selector(didRotateNotification));

## 如何重写isEqual

## 添加到navigationController.view 中的视图要手动removeFromSuperview

最好在dealloc中检查一下，没remove的一定要手动remove.

~~DeviceDetection~~

~~connectedToNetwork~~

~~table.backgroundColor = [UIColor clearColor];~~

~~table.backgroundView = nil;~~

~~[self unregisterFromNotifications];~~

~~[self unregisterFromKVO];~~

# 其它

## 比较版本号

NSString \*currSysVer = [[UIDevice currentDevice] systemVersion];

if ([currSysVer compare:minRequirement options:NSNumericSearch] != NSOrderedAscending)

{

return YES;

}else{

return NO;

}

## 如果确认软件升级了

保存当前版本的版本号到Document中，每次启动和Bundle中的版本号比较，版本号小Bundle中的被更新了。

这时可以做一些更新资源文件的操作：

1. 直接覆盖到Document
2. 把Document中的内容经过比较，移植到Bundle文件中，再把Bundle中的文件拷贝到Document

## B/S传输文件，如果本来约定的数据结构变了：

一般来说，使用XML是不会对旧程序产生不良影响的，但版本号已经变了，更新后的程序要用新的数据，但由于当前版本号与服务器中的一样，新文件是申请不下来的。可以在Bundle中带着最新的文件，直接覆盖旧文件

## 日期的使用

NSDate \*someDate = [NSDate dateWithTimeIntervalSince1970:some\_time\_t]

time\_t unixTime = (time\_t) [[NSDate date] timeIntervalSince1970];

## 关于更新

iPhone自动保存document中的内容，如果你把文件放在document中，以后开发又改了这个文件的内容或格式，那更新之后运行很可能出错。解决的办法是，配置文件放bundle里，或者改个文件名。每次更新前都要从App store 下载旧版本，运行一段一时间后，再此基础上编译新版，运行不出错才能上传

## 时间相关

NSDate需要设置calendar，使用不方便也因为服务器传过来的是time\_t格式，所以我在客户端对时间的操作主要用的C语言的方法。

需要注意的是，有的函数不是线程安全的，也就是说在同一个范围内调用多次时，需要调用线程安全的版本，这样的函数有：

localtime\_r

asctime\_r

ctime\_r

gmtime\_r

localtime\_r

另外，可以直接给struct tm各成员变量赋值，例如（注意顺序）

struct tm tmStart = {second,minute,hour,day, mon, year};

struct tm的各成员是不能的加减的，因为超过了各变量的范围，可能出错，需要先转成time\_t，再加减相应的时间

struct tm\* tmp = localtime(&Time);

asctime(tmp)

time\_t now;

time(&now);

struct tm \* tmNow = localtime(&now);

struct tm tm = {0,nMinute,nHour,tmNow->tm\_mday, tmNow->tm\_mon, tmNow->tm\_year};

time\_t \_time = mktime(&tm);

time\_t tt1 = status.time;

time\_t tt2 = status2.time;

struct tm tm1 = {0};

struct tm tm2 = {0};

localtime\_r(&tt1, &tm1);

localtime\_r(&tt2, &tm2);

## 用#if defined控制不同版本协议的使用

- (void)Req\_Login\_Wh\_Proc:(id)reqMsg

{

#if defined(EDITION\_0\_009\_)

[self Req\_Login\_Wh\_0\_009\_Proc:reqMsg];

#elif defined (EDITION\_0\_010\_) || defined(EDITION\_0\_011\_)

[self Req\_Login\_Wh\_0\_011\_Proc:reqMsg];

#else

[self Req\_Login\_Wh\_0\_012\_Proc:reqMsg];

#endif

}

## 设置字节对齐方式

#pragma pack(1)//设置字节对齐方式为1字节对齐