个人整理超精密的 IOS 笔记

目录

退回输入键盘

```
- (BOOL) textFieldShouldReturn:(id)textField{
    [textField   resignFirstResponder];
}
```

CGRect

CGRect frame = CGRectMake (origin.x, origin.y, size.width, size.height)；矩形
NSStringFromCGRect(someCG) 把 CGRect 结构转变为格式化字符串；
CGRectFromString(aString) 由字符串恢复出矩形；
CGRectInset(aRect) 创建较小或较大的矩形（中心点相同），＋较小 －较大
CGRectIntersectsRect(rect1, rect2) 判断两矩形是否交叉，是否重叠
CGRectZero 高度和宽度为零的 / 位于（0，0）的矩形常量

CGPoint & CGSize

```
CGPoint aPoint = CGPointMake(x, y);
CGSize aSize = CGSizeMake(width, height);
```

设置透明度

```
[myView setAlpha:value];    (0.0 < value < 1.0)
```

设置背景色

```
[myView setBackgroundColor:[UIColor redColor]];
 (blackColor;darkGrayColor;lightGrayColor;
whiteColor;grayColor; redColor; greenColor;
blueColor; cyanColor;yellowColor;
magentaColor;orangeColor;purpleColor;
brownColor; clearColor; )
```

自定义颜色

```
UIColor *newColor = [[UIColor alloc]
 initWithRed:(float) green:(float) blue:(float) alpha:(float)];
     0.0~1.0
```

竖屏
320X480

横屏
480X320

状态栏高 (显示时间和网络状态)
20 像素

导航栏、工具栏高(返回)
44 像素

隐藏状态栏
[[UIApplication shareApplication] setStatusBarHidden: YES animated:NO]

横屏
[[UIApplication shareApplication]
setStatusBarOrientation:UIInterfaceOrientationLandscapeRight].

屏幕变动检测
orientation == UIInterfaceOrientationLandscapeLeft

全屏
window=[[UIWindow alloc] initWithFrame:[UIScreen mainScreen] bounds];

自动适应父视图大小：
```
aView.autoresizingSubviews = YES;
aView.autoresizingMask = (UIViewAutoresizingFlexibleWidth |
                          UIViewAutoresizingFlexibleHeight);
```

定义按钮
```
UIButton *scaleUpButton = [UIButton buttonWithType:UIButtonTypeRoundedRect];
[scaleUpButton setTitle:@"放  大" forState:UIControlStateNormal];
scaleUpButton.frame = CGRectMake(40, 420, 100, 40);
[scaleUpButton addTarget:self
  action:@selector(scaleUp)
forControlEvents:UIControlEventTouchUpInside];
```

设置视图背景图片
```
UIImageView *aView;
[aView setImage:[UIImage imageNamed:@”name.png”]];
view1.backgroundColor = [UIColor colorWithPatternImage:
[UIImage imageNamed:@"image1.png"]];
```

自定义 UISlider 的样式和滑块

我们使用的是 UISlider 的 setMinimumTrackImage，和 setMaximumTrackImage 方法来定义图片的，这两个方法可以设置滑块左边和右边的图片的，不过如果用的是同一张图片且宽度和控件宽度基本一致，就不会有变形拉伸的后果，先看代码，写在 viewDidLoad 中：

```
//左右轨的图片
UIImage *stetchLeftTrack= [UIImage imageNamed:@"brightness_bar.png"];
UIImage *stetchRightTrack = [UIImage imageNamed:@"brightness_bar.png"];
//滑块图片
UIImage *thumbImage = [UIImage imageNamed:@"mark.png"];

UISlider *sliderA=[[UISlider alloc]initWithFrame:CGRectMake(30, 320, 257, 7)];
sliderA.backgroundColor = [UIColor clearColor];
sliderA.value=1.0;
sliderA.minimumValue=0.7;
sliderA.maximumValue=1.0;

[sliderA setMinimumTrackImage:stetchLeftTrack forState:UIControlStateNormal];
[sliderA setMaximumTrackImage:stetchRightTrack forState:UIControlStateNormal];
//注意这里要加UIControlStateHightlighted的状态，否则当拖动滑块时滑块将变成原生的控件
[sliderA setThumbImage:thumbImage forState:UIControlStateHighlighted];
[sliderA setThumbImage:thumbImage forState:UIControlStateNormal];
//滑块拖动时的事件
[sliderA addTarget:self action:@selector(sliderValueChanged:) forControlEvents:UIControlEventValueChanged];
//滑动拖动后的事件
[sliderA addTarget:self action:@selector(sliderDragUp:) forControlEvents:UIControlEventTouchUpInside];

[self.view addSubview:sliderA];
```

为了大家实验方便，我附上背景图brightness_bar.png和滑块图mark.png

http://pic002.cnblogs.com/images/2011/162291/2011121611431816.png

http://pic002.cnblogs.com/images/2011/162291/2011121611432897.png

```
-(IBAction)sliderValueChanged:(id)sender{
UISlider *slider = (UISlider *) sender;
NSString *newText = [[NSString alloc] initWithFormat:@"%d", (int)(slider.value + 0.5f)];
label.text = newText;
}
```

活动表单

```
<UIActionSheetDelegate>

 - (IBActive) someButtonPressed:(id) sender
{
    UIActionSheet *actionSheet = [[UIActionSheet alloc]
                    initWithTitle:@”Are you sure?”
                    delegate:self
                    cancelButtonTitle:@”No way!”
                    destructiveButtonTitle:@”Yes, I’m Sure!”
                    otherButtonTitles:nil];
    [actionSheet showInView:self.view];
    [actionSheet release];
}
```

警告视图

```
 <UIAlertViewDelegate>

 - (void) actionSheet:(UIActionSheet *)actionSheet didDismissWithButtonIndex:(NSInteger)
buttonIndex
{
     if(buttonIndex != [actionSheet cancelButtonIndex])
     {
          NSString *message = [[NSString alloc] initWithFormat:@”You can
                breathe easy, everything went OK.”];
          UIAlertView *alert = [[UIAlertView alloc]
                              initWithTitle:@”Something was done”
                               message:message
                               delegate:self
                               cancelButtonTitle:@”OK”
                               otherButtonTitles:nil];
          [alert show];
          [alert release];
          [message release];
     }
}
```

动画效果

```
-(void)doChange:(id)sender
{
if(view2 == nil)
{
[self loadSec];
}
```

```
[UIView beginAnimations:nil context:NULL];
[UIView setAnimationDuration:1];
[UIView setAnimationTransition:([view1 superview]?UIViewAnimationTransitionFlipFromLeft:UIViewAnimationTransitionFlipFromRight)forView:self.view cache:YES];

    if([view1 superview]!= nil)
{
[view1 removeFromSuperview];
[self.view addSubview:view2];

}else {

[view2 removeFromSuperview];
[self.view addSubview:view1];
}
[UIView commitAnimations];
}

Table View    <UITableViewDateSource>
#pragma mark -
#pragma mark Table View Data Source Methods
//指定分区中的行数，默认为 1
- (NSInteger)tableView:(UITableView *)tableView
 numberOfRowsInSection:(NSInteger)section
{
return [self.listData count];
}

//设置每一行 cell 显示的内容
- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
static NSString *SimpleTableIndentifier = @"SimpleTableIndentifier";
UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:SimpleTableIndentifier];
if (cell == nil) {
cell = [[[UITableViewCell alloc]
initWithStyle:UITableViewCellStyleSubtitle
reuseIdentifier:SimpleTableIndentifier]
autorelease];
}
     UIImage *image = [UIImage imageNamed:@"13.gif"];
cell.imageView.image = image;
```

```
NSUInteger row = [indexPath row];
cell.textLabel.text = [listData objectAtIndex:row];
      cell.textLabel.font = [UIFont boldSystemFontOfSize:20];

      if(row < 5)
cell.detailTextLabel.text = @"Best friends";
else
     cell.detailTextLabel.text = @"friends";
return cell;
}
```

图像、文本标签和详细文本标签

图像：如果设置图像，则它显示在文本的左侧； 文本标签：这是单元的主要文本（UITableViewCellStyleDefault 只显示文本标签）；详细文本标签：这是单元的辅助文本，通常用作解释性说明或标签

```
UITableViewCellStyleSubtitle
UITableViewCellStyleDefault
UITableViewCellStyleValue1
UITableViewCellStyleValue2

<UITableViewDelegate>
#pragma mark -
#pragma mark Table View Delegate Methods
//把每一行缩进级别设置为其行号
- (NSInteger)tableView:(UITableView *)tableView
indentationLevelForRowAtIndexPath:(NSIndexPath *)indexPath
{
NSUInteger row = [indexPath row];
return row;
}
//获取传递过来的 indexPath 值
- (NSIndexPath *)tableView:(UITableView *)tableView
willSelectRowAtIndexPath:(NSIndexPath *)indexPath
{
NSUInteger row = [indexPath row];
if (row == 0)
return nil;
return indexPath;
}

- (void)tableView:(UITableView *)tableView didSelectRowAtIndexPath:(NSIndexPath
```

```
*)indexPath
{
NSUInteger row = [indexPath row];
NSString *rowValue = [listData objectAtIndex:row];
NSString *message = [[NSString alloc] initWithFormat:@"You selected %@",rowValue];
UIAlertView *alert = [[UIAlertView alloc] initWithTitle:@"Row Selected"
message:message
    delegate:nil
  cancelButtonTitle:@"Yes, I did!"
  otherButtonTitles:nil];
[alert show];
[alert release];
[message release];
[tableView deselectRowAtIndexPath:indexPath animated:YES];
}

//设置行的高度
-  (CGFloat)tableView:(UITableView  *)tableView  heightForRowAtIndexPath:(NSIndexPath
*)indexPath
{
return 40;
}

NavigationController 推出 push 推出 pop
[self.navigationController pushViewController:_detailController animated:YES];
[self.navigationController popViewControllerAnimated:YES];

Debug:
NSLog(@"%s %d", __FUNCTION__, __LINE__);
```

点击 textField 外的地方回收键盘

先定义一个 UIControl 类型的对象，在上面可以添加触发事件，令 SEL 实践为回收键盘的方法,最后将 UIControl 的实例加到当前 View 上。

```
UIControl *m_control = [[UIControl alloc] initWithFrame:CGRectMake(0, 0, 320, 480)];
[m_control addTarget:self action:@selector(keyboardReturn)
forControlEvents:UIControlEventTouchUpInside];
[self.view addSubview:m_control];

- (void) keyboardReturn
{
[aTextField resignFirstResponder];
}
```

键盘覆盖输入框

当键盘调出时将输入框覆盖时，可以用下方法：

```
- (BOOL)textFieldShouldBeginEditing:(UITextField *)textField
{
[self.view setFrame:CGRectMake(0, -100, 320, 480) ];
return YES;
}
- (BOOL)textFieldShouldEndEditing:(UITextField *)textField
{
   [self.view setFrame:CGRectMake(0, 0, 320, 480)];
        return YES;
}
```

当准备输入时，将视图的位置上调 100，这样键盘就不能覆盖到输入框。

当依赖注入方法不好使时，可以在 AppDelegate 内申明一个全局的控制器实例 _anotherViewController，在另一个需要使用_anotherViewController 的地方定义以下委托方法，使用共享的 UIApplication 实例来获取该委托的引用

```
SomeAppDelegate *appDelegate = (SomeAppDelegate *)[[UIApplication sharedApplication] delegate];
_anotherViewController = appDelegate._anotherViewController;
```

UIViewController 内建 Table View

纯代码在 UIViewController 控制器内建 Table View

```
@interface RootViewController : UIViewController <UITableViewDelegate, UITableViewDataSource> {
NSArray *timeZoneNames;
}
@property (nonatomic,retain) NSArray *timeZoneNames;
@end

(void) loadView
{
UITableView *tableView = [[UITableView alloc] initWithFrame:[[UIScreen mainScreen] applicationFrame]] style: UITableViewStylePlain];
tableView.autoresizingMask = (UIViewAutoresizingFlexibleHeight | UIViewAutoresizingWidth);
tableView.delegate = self;
tableView.dataSource = self;
[tableView reloadData];

self.view = tableView;
[tableView release];
}
```

将 plist 文件中的数据赋给数组

```
NSString *thePath = [[NSBundle mainBundle] pathForResource:@"States" ofType:@"plist"];
NSArray *array = [NSArray arrayWithContentsOfFile:thePath];
```

UITouch

手指的触摸范围：64X64

```
#pragma mark -
#pragma mark Touch Events

- (void)touchesBegan:(NSSet *) touches withEvent:(UIEvent *) event {
originFrame = bookCover.frame;
NSLog(@"%s %d", __FUNCTION__,__LINE__);

if ([touches count] == 2)
{
NSArray *twoTouches = [touches allObjects];
UITouch *firstTouch = [twoTouches objectAtIndex:0];
UITouch *secondTouch = [twoTouches objectAtIndex:1];
CGPoint firstPoint = [firstTouch locationInView:bookCover];
CGPoint secondPoint = [secondTouch locationInView:bookCover];

CGFloat deltaX = secondPoint.x - firstPoint.x;
CGFloat deltaY = secondPoint.y - firstPoint.y;
initialDistance = sqrt(deltaX * deltaX + deltaY * deltaY );
frameX = bookCover.frame.origin.x;
frameY = bookCover.frame.origin.y;
frameW = bookCover.frame.size.width;
frameH = bookCover.frame.size.height;
NSLog(@"%s %d", __FUNCTION__,__LINE__);
}
}

- (void)touchesMoved:(NSSet *) touches withEvent:(UIEvent *) event {

if([touches count] == 2)
{
NSLog(@"%s %d", __FUNCTION__,__LINE__);

NSArray *twoTouches = [touches allObjects];
UITouch *firstTouch = [twoTouches objectAtIndex:0];
UITouch *secondTouch = [twoTouches objectAtIndex:1];
```

```
CGPoint firstPoint = [firstTouch locationInView:bookCover];
CGPoint secondPoint = [secondTouch locationInView:bookCover];

CGFloat deltaX = secondPoint.x - firstPoint.x;
CGFloat deltaY = secondPoint.y - firstPoint.y;
CGFloat currentDistance = sqrt(deltaX * deltaX + deltaY * deltaY );

if (initialDistance == 0) {
initialDistance = currentDistance;
}
else if (currentDistance != initialDistance)
{
CGFloat changedDistance = currentDistance - initialDistance;
NSLog(@"changedDistance = %f",changedDistance);
[bookCover setFrame:CGRectMake(frameX - changedDistance / 2,
frameY - (changedDistance * frameH) / (2 * frameW),
frameW + changedDistance,
frameH + (changedDistance * frameH) / frameW)];
}
}
}

- (void)touchesEnded:(NSSet *) touches withEvent:(UIEvent *) event {
UITouch *touch = [touches anyObject];
```

UITouch 双击图片变大/还原

```
if ([touch tapCount] == 2)
{
NSLog(@"%s %d", __FUNCTION__,__LINE__);

if (!flag) {
[bookCover setFrame:CGRectMake(bookCover.frame.origin.x - bookCover.frame.size.width / 2,
bookCover.frame.origin.y - bookCover.frame.size.height / 2,
2 * bookCover.frame.size.width,
2 * bookCover.frame.size.height)];
flag = YES;
}
else {
[bookCover setFrame:CGRectMake(bookCover.frame.origin.x + bookCover.frame.size.width / 4,
bookCover.frame.origin.y + bookCover.frame.size.height / 4,
bookCover.frame.size.width / 2, bookCover.frame.size.height / 2)];
flag = NO;
}
}
```

}

Get the Location of Touches
(CGPoint)locationInView:(UIView *)view
(CGPoint)previousLocationInView:(UIView *)view
view window

Getting Touch Attributes
tapCount(read only) timestamp(read only) phase(read only)

Getting a Touch Object's Gesture Recognizers
gestureRecognizers

Touch Phase
UITouchPhaseBegan
UITouchPhaseMoved
UITouchPhaseStationary
UITouchPhaseEnded
UITouchPhaseCancelled

从 Plist 里读内容
NSString *plistPath = [[NSBundle mainBundle] pathForResource:@"book" ofType:@"plist"];
NSDictionary *dictionary = [[NSDictionary alloc] initWithContentsOfFile:plistPath];
NSString *book = [dictionary objectForKey:bookTitle];
[textView setText:book];

(void) initialize {
NSUserDefaults = [NSUserDefaults standardUserDefaults];
NSDictionary *appDefaults = [NSDictionary dictionaryWithObject:@"YES" forKey:@"DeleteBackup"];
[defaults registerDefaults:appDefaults];
}

To get a value of a default, use the valueForKey: method:
[[theDefaultsController values] valueForKey:@"userName"];
To set a value for a default, use setValue:forKey:
[[theDefaultsController values] setValue:newUserName forKey:@"userName"];

[[NSUserDefaults standardUserDefaults] setValue:aVale forKey:aKey];
[[NSUserDefaults standardUserDefaults] valueForKey:aKey];

获取 Documents 目录
NSArray *paths = NSSearchPathForDictionariesInDomains(NSDocumentDirectory, NSUserDomainMask, YES);

```
NSString *documentsDirectory = [paths objectAtIndex:0];
NSString *filename = [documentsDirectory
stringByAppendingPathComponent:@"theFile.txt"];
```

获取 tmp 目录

```
NSString *tempPath = NSTemporaryDirectory();
NSString *tempFile = [tempPath stringByAppendingPathComponent:@"tempFile.txt"];

[[NSUserDefaults standardUserDefaults] setObject:data forKey:@"someKey"];
[[NSUserDefaults standardUserDefaults] objectForKey:aKey];
```

自定义 NavigationBar

```
navigationBar = [[UINavigationBar alloc] initWithFrame:CGRectMake(0, 0, 320, 44)];
[navigationBar setBarStyle:UIBarStyleBlackOpaque];

myNavigationItem = [[UINavigationItem alloc] initWithTitle:@"Setting"];
[navigationBar setItems:[NSArray arrayWithObject:myNavigationItem]];
[self.view addSubview:navigationBar];

backButton = [[UIBarButtonItem alloc] initWithTitle:@"Back" style:UIBarButtonItemStylePlain
target:self action:@selector(back)];
myNavigationItem.leftBarButtonItem = backButton;
```

利用 Safari 打开一个链接

```
NSURL *url = [NSURL URLWithString:@"http://www.cnblogs.com/tracy-e/"];
[[UIApplication sharedApplication] openURL:url];
```

利用 UIWebView 显示 pdf 文件、网页。。。

```
webView = [[UIWebView alloc] initWithFrame:CGRectMake(0, 0, 320, 480)];
[webView setDelegate:self];
[webView setScalesPageToFit:YES];
[webView setAutoresizingMask:UIViewAutoresizingFlexibleWidth |
UIViewAutoresizingFlexibleHeight];
[webView setAllowsInlineMediaPlayback:YES];
[self.view addSubview:webView];
NSString *pdfPath = [[NSBundle mainBundle] pathForResource:@"ojc" ofType:@"pdf"];
NSURL *url = [NSURL fileURLWithPath:pdfPath];
NSURLRequest *request = [NSURLRequest requestWithURL:url
cachePolicy:NSURLRequestUseProtocolCachePolicy
timeoutInterval:5];
[webView loadRequest:request];
```

```
[myWebView loadRequest:[NSURLRequest requestWithURL:[NSURL
                                URLWithString: @"http://www.cnblogs.com/tracy-e/"]]];

NSString *errorString = [NSString stringWithFormat:@"<html><center><font size=
+5 color ='red'>An Error Occurred:<br>%@</fone></center></html>",error];
[myWebView loadHTMLString:errorString baseURL:nil];

//Stopping a load request when the view is to disappear
- (void)viewWillDisappear:(BOOL)animate{
if ([myWebView loading]){
[myWebView stopLoading];
}
myWebView.delegate = nil;
[UIApplication shareApplication].networkActivityIndicatorVisible = NO;
}
```

汉字转码

```
NSString *oriString = @"\u67aa\u738b";
NSString *escapedString = [oriString
stringByReplacingPercentEscapesUsingEncoding: NSUTF8StringEncoding];
```

Checking for background support on earlier versions of iOS

```
UIDevice *device = [UIDevice currentDevice];
BOOL backgroundSupported = NO;
if ([device respondsToSelector:@selector(isMultitaskingSupported)]){
backgroundSupported = device.multitaskingSupported;
}
```

Being a Responsible,Multitasking-Aware Application

```
# Do not make any OpenGL ES calls from your code.
# Cancel any Bonjour-related services before being suspended.
# Be prepared to handle connection failures in your network-based sockets.
# Save your application state before moving to the background.
# Release any unneeded memory when moving to the background.
# Stop using shared system resources before being suspended.
# Avoid updating your windows and views.
# Respond to connect and disconnect notification for external accessories.
# Clean up resource for active alerts when moving to the background.
# Remove sensitive information from views before moving to the background.
# Do minimal work while running in the background.
```

Handing the Keyboard notifications

```
//Call this method somewhere in your view controller setup code
```

```
- (void) registerForKeyboardNotifications{

[[NSNotificationCenter defaultCenter] addObserver:self
selector:@selector(keyboardWasShown:)
name:UIKeyboardDidShowNotification
object:nil];
[[NSNotificationCenter defaultCenter] addObserver:self
selector:@selector(keyboardWasHidden:)
name:UIKeyboardDidHideNotification
object:nil];

}

//Called when the UIKeyboardDidShowNotification is sent
- (void)keyboardWasShown:(NSNotification *) aNotification{
if(keyboardShown)
return;
NSDictionary *info = [aNotification userInfo];

//get the size of the keyboard.
NSValue *aValue = [info objectForKey:UIKeyboardFrameBeginUserInfoKey];
CGSize keyboardSize = [aValue CGRectValue].size;

//Resize the scroll view
CGRect viewFrame = [scrollView frame];
viewFrame.size.height -= keyboardSize.height;

//Scroll the active text field into view
CGRect textFieldRect = [activeField frame];
[scrollView scrollRectToVisible:textFieldRect animated:YES];

keyboardShown = YES;
}

//Called when the UIKeyboardDidHideNotification is sent
- (void)keyboardWasHidden:(NSNotification *) aNotification{
NSDictionary *info = [aNotification userInfo];

//Get the size of the keyboard.
NSValue *aValue = [info objectForKey:UIKeyboardFrameEndUserInfoKey];
CGSize keyboardSize = [aValue CGRectValue].size;

//Reset the height of the scroll view to its original value
CGRect viewFrame = [scrollView Frame];
```

```
viewFrame.size.height += keyboardSize.height;
scrollView.frame = viewFrame;

keyboardShown = NO;
}
```

点击键盘的 next 按钮，在不同的 textField 之间换行

```
//首先给不同的 textField 赋不同的且相邻的 tag 值
- (BOOL)textFieldShouldReturn:(UITextField *)textField
{
if ([textField returnKeyType] != UIReturnKeyDone)
{
NSInteger nextTag = [textField tag] + 1;
UIView *nextTextField = [[self tableView] viewWithTag:nextTag];
[nextTextField becomeFirstResponder];
}
else {
[textField resignFirstResponder];
}
return YES;
}
```

Configuring a date formatter

```
- (void)viewDidLoad {
[super viewDidLoad];
dateFormatter = [[NSDateFormatter alloc] init];
[dateFormatter setGeneratesCalendarDates:YES];
[dateFormatter setLocale:[NSLocale currentLocale]];
[dateFormatter setCalendar:[NSCalendar autoupdatingCurrentCalendar]];
[dateFormatter setTimeZone:[NSTimeZone defaultTimeZone]];
[dateFormatter setDateStyle:NSDateFormatterShortStyle];
DOB.placeholder = [NSString stringWithFormat:@"Example: %@",[dateFormatter
stringFromDate:[NSDate date]]];
}

- (void)textFieldDidEndEditing:(UITextField *)textField{
[textField resignFirstResponder];
if ([textField.text isEqualToString:@""])
return;
switch (textField.tag){
case DOBField:
NSDate *theDate = [dateFormatter dateFromString:textField.text];
if (theDate)
[inputDate setObject:theDate forKey:MyAppPersonDOBKey];
```

```
break;
default:
break;
}
}
```

tableView 的 cell 高度

tableView 的 cell 高度除了在 delegate 中指定外，还可以在任意位置以[tableView setRowHeight:44]的方式指定

```
[[self navigationItem] setLeftBarButtonItem:[self editButtonItem]];

- (void)setEditing:(BOOL)editing animated:(BOOL)animated{
[super setEditing:editing animated:animated];
if (editing){
.....
}
else{
.....
}
}
```

One added a subview to a view, release the subview to avoid the extra retain count of it, Because when you insert a view as a subview using addSubview:, the subview is retained by its superview. When you remove the subview from its superview using the removeFromSuperview: method, subview is autoreleased.

为 UINavigationBar 设置背景图片

在 iPhone 开发中，有时候我们想给导航条添加背景图片，实现多样化的导航条效果，用其他方法往往无法达到理想的效果，经过网上搜索及多次实验，确定如下最佳实现方案:

为 UINavigatonBar 增加如下 Category(类别:提供一种为某个类添加方法而又不必编写子类的途径,类别只能添加成员函数，不能添加数据成员):

```
@implementation UINavigationBar (CustomImage)
- (void)drawRect:(CGRect)rect {
   UIImage *image = [UIImage imageNamed: @"NavigationBar.png"];
   [image drawInRect:CGRectMake(0, 0, self.frame.size.width, self.frame.size.height)];
}
@end
```

例如，在我的项目中，添加如下代码:

```
//////////////////////////////////////////////////////////
/* input: The image and a tag to later identify the view */
```

```
@implementation UINavigationBar (CustomImage)
- (void)drawRect:(CGRect)rect {
    UIImage *image = [UIImage imageNamed: @"title_bg.png"];
    [image drawInRect:CGRectMake(0, 0, self.frame.size.width, self.frame.size.height)];
}
@end
//////////////////////////////////////////////////////////
@implementation FriendsPageViewController
// Implement viewDidLoad to do additional setup after loading the view, typically from a nib.
- (void)viewDidLoad {
    self.navigationBar.tintColor = [UIColor purpleColor];

    [self initWithRootViewController:[[RegPageViewController alloc] init]];
    [super viewDidLoad];
}
.....
```

实现的效果如下图:

转载，原文地址 http://blog.csdn.net/wave_1102/archive/2009/11/04/4768212.aspx

为 UINavigationBar 添加自定义背景

```
@implementation UINavigationBar (UINavigationBarCategory)

- (void)drawRect:(CGRect)rect {
        //颜色填充
//    UIColor *color = [UIColor redColor];
//    CGContextRef context = UIGraphicsGetCurrentContext();
//    CGContextSetFillColor(context, CGColorGetComponents( [color CGColor]));
```

```
//    CGContextFillRect(context, rect);
//    self.tintColor = color;
      //图片填充
      UIColor *color = [UIColor colorWithRed:46.0f/255.0f
      green:87.0f/255.0f blue:29.0f/255.0f alpha:1.0f];

      UIImage *img       = [UIImage imageNamed: @"bg.png"];
      [img drawInRect:CGRectMake(0, 0, self.frame.size.width, self.frame.size.height)];

      self.tintColor = color;
}

@end
```

加载图片要及时 release

你还在使用 myImage = [UIImage imageNamed:@"icon.png"]; 吗？

如题，是不是大家为了方便都这样加载图片啊

myImage = [UIImage imageNamed:@"icon.png"];

那么小心了

这种方法在一些图片很少，或者图片很小的程序里是 ok 的。

但是，在大量加载图片的程序里，请千万不要这样做。

为什么呢 ？？？？？？？

这种方法在 application bundle 的顶层文件夹寻找由供应的名字的图象。 如果找到图片，装载到 iPhone 系统缓存图象。那意味图片是(理论上)放在内存里作为 cache 的。

试想你图片多了，是什么后果？

图片 cache 极有可能不会响应 memory warnings and release its objects

所以，用图片的时候一定要小心的 alloc 和 release。

推荐使用 NSString *path = [[NSBundle mainBundle] pathForResource:@"icon" ofType:@"png"];

myImage = [UIImage imageWithContentsOfFile:path];

// Todo use of myImage

[myImage release];

From: http://www.cocoachina.com/bbs/simple/?t27420.html

uiwebview 打开 doc,pdf 文件

```
UIWebView *webView = [[UIWebView alloc]initWithFrame:CGRectMake(0, 55, 320, 300)];
    webView.delegate = self;
    webView.multipleTouchEnabled = YES;
    webView.scalesPageToFit = YES;

    NSArray *paths = NSSearchPathForDirectoriesInDomains(NSDocumentDirectory, NSUserDomainMask, YES);
    NSString *documentsDirectory = [paths objectAtIndex:0];
    NSString *docPath = [documentsDirectory stringByAppendingString:@"/doc2003_1.doc"];
NSLog(@"#######%@",docPath);

    NSURL *url = [NSURL fileURLWithPath:docPath];
    NSURLRequest *request = [NSURLRequest requestWithURL:url];
    [webView loadRequest:request];

    [self.view addSubview:webView];
    [webView release];
```

From:http://blog.csdn.net/dadalan/archive/2010/10/22/5959301.aspx

iPhone 游戏中既播放背景音乐又播放特效声音的办法

有时候在 iPhone 游戏中，既要播放背景音乐，同时又要播放比如枪的开火音效。此时您可以试试以下方法

```
        NSString *musicFilePath = [[NSBundle mainBundle] pathForResource:fileName ofType:@"wav"];          //创建音乐文件路径
        NSURL *musicURL = [[NSURL alloc] initFileURLWithPath:musicFilePath];
        AVAudioPlayer* musicPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:musicURL error:nil];
        [musicURL release];
        [musicPlayer prepareToPlay];
        //[musicPlayer setVolume:1];                    //设置音量大小
        //musicPlayer .numberOfLoops = -1;//设置音乐播放次数  -1 为一直循环
```

要导入框架 AVFoundation.framework，头文件中 #import <AVFoundation/AVFoundation.h>；做成类的话则更方便。

From: http://blog.csdn.net/dadalan/archive/2010/10/19/5950493.aspx

NSNotificationCenter 用于增加回调函数

[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(_willBecomeActive) name:UIApplicationDidBecomeActiveNotification object:nil];

UINavigationBar 背景 Hack

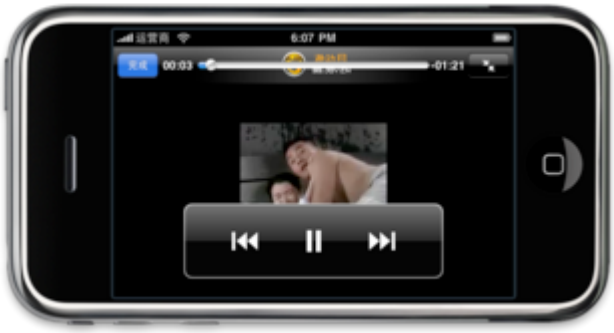
LOGO_320×44.png 图片显示在背景上，

```
@implementation UINavigationBar (UINavigationBarCategory)
- (void)drawRect:(CGRect)rect {
    //加入旋转坐标系代码
   // Drawing code
    UIImage *navBarImage = [UIImage imageNamed:@"LOGO_320×44.png"];
    CGContextRef context = UIGraphicsGetCurrentContext();
    CGContextTranslateCTM(context, 0.0, self.frame.size.height);
    CGContextScaleCTM(context, 1.0, -1.0);

    CGPoint center=self.center;

    CGImageRef cgImage= CGImageCreateWithImageInRect(navBarImage.CGImage, CGRectMake(0, 0, 1, 44));
    CGContextDrawImage(context, CGRectMake(center.x-160-80, 0, 80, self.frame.size.height), cgImage);
    CGContextDrawImage(context, CGRectMake(center.x-160, 0, 320, self.frame.size.height), navBarImage.CGImage);
    CGContextDrawImage(context, CGRectMake(center.x+160, 0, 80, self.frame.size.height), cgImage);
}
@end
```

old code

```
CGContextDrawImage(context, CGRectMake(0, 0, self.frame.size.width, self.frame.size.height), navBarImage.CGImage);
```

hack 过 logo 不再拉伸



From: http://blog.163.com/fengyi1103@126/blog/static/1383562742010627910267 1/

清除电话号码中的其他符号（源码）

最近从通讯录读取电话号码，读出得号码如：134－1814－****。
而我需要的为 11 位纯数字，一直找方法解决此问题，今天终于找到了。。
分享一下……

代码如下：

```
NSString *originalString = @"(123) 123123 abc";
NSMutableString *strippedString = [NSMutableString
        stringWithCapacity:originalString.length];

NSScanner *scanner = [NSScanner scannerWithString:originalString];
NSCharacterSet *numbers = [NSCharacterSet
        characterSetWithCharactersInString:@"0123456789"];

while ([scanner isAtEnd] == NO) {
   NSString *buffer;
   if ([scanner scanCharactersFromSet:numbers intoString:&buffer]) {
      [strippedString appendString:buffer];
   }
   // --------- Add the following to get out of endless loop
   else {
       [scanner setScanLocation:([scanner scanLocation] + 1)];
   }
   // --------- End of addition
}

NSLog(@"%@", strippedString); // "123123123"
```

From： http://stackoverflow.com/questions/1129521/remove-all-but-numbers-from-nsstring

正则判断：字符串只包含字母和数字

```
NSString *mystring = @"Letter1234";
NSString *regex = @"[a-z][A-Z][0-9]";

NSPredicate *predicate = [NSPredicate predicateWithFormat:@"SELF MATCHES %@", regex];

if ([predicate evaluateWithObject:mystring] == YES) {
        //implement
}
```

一行代码设置 UITableViewCell 与导航条间距

UITableView 的 cell 默认出现在 uitableview 的第一行，如果你想自定义 UITableViewCell 与导航条间距的话，可以使用下面这行代码

```
tableview.tableHeaderView = [[[UIView alloc] initWithFrame:CGRectMake(0, 0, 5, 20)]autorelease];
```

From: http://blog.163.com/fengyi1103@126/blog/static/13835627420101016111074 92/

修改 UITableview 滚动条颜色的方法

UITableview 的滚动条默认颜色是黑色的，如果 UItableview 背景也是深颜色，则滚动条会变的很不明显。您可以用下面这行代码来改变滚动条的颜色

```
self.tableView.indicatorStyle=UIScrollViewIndicatorStyleWhite;
```

当然，最后的 “White” 也可以换成其它颜色。

下文件之前获取到文件大小的代码

下面这段代码，能实现在下载文件之前获得文件大小，应用在软件里，能在很大程度上改善用户体验

```
[m_pASIHTTPRequest setDidReceiveResponseHeadersSelector:@selector(didReceiveResponseHeaders:)];

- (void)didReceiveResponseHeaders:(ASIHTTPRequest *)request
{
    NSLog(@"didReceiveResponseHeaders %@",[m_request.responseHeaders
```

valueForKey:@"Content-Length"]);
}

网络编程总结 iphone

一：确认网络环境 3G/WIFI

1. 添加源文件和 framework

开发 Web 等网络应用程序的时候，需要确认网络环境，连接情况等信息。如果没有处理它们，是不会通过 Apple 的审(我们的)查的。

Apple 的 例程 Reachability 中介绍了取得/检测网络状态的方法。要在应用程序程序中使用 Reachability，首先要完成如下两部：

1.1. 添加源文件：

在你的程序中使用 Reachability 只须将该例程中的 Reachability.h 和 Reachability.m 拷贝到你的工程中。如下图：

1.2.添加 framework：

将 SystemConfiguration.framework 添加进工程。如下图：

2. 网络状态

Reachability.h 中定义了三种网络状态：

```
typedef enum {
    NotReachable = 0,                 //无连接
    ReachableViaWiFi,                  //使用 3G/GPRS 网络
    ReachableViaWWAN                    //使用 WiFi 网络
} NetworkStatus;
```

因此可以这样检查网络状态：

```
Reachability *r = [Reachability reachabilityWithHostName:@“www.apple.com”];
switch ([r currentReachabilityStatus]) {
        case NotReachable:
                // 没有网络连接
                break;
        case ReachableViaWWAN:
                // 使用 3G 网络
                break;
        case ReachableViaWiFi:
```

```
                    // 使用 WiFi 网络
                    break;
        }
```

3.检查当前网络环境

程序启动时，如果想检测可用的网络环境，可以像这样

```
        // 是否 wifi
        + (BOOL) IsEnableWIFI {
            return ([[Reachability reachabilityForLocalWiFi] currentReachabilityStatus] != NotReachable);
        }

        // 是否 3G
        + (BOOL) IsEnable3G {
            return ([[Reachability reachabilityForInternetConnection] currentReachabilityStatus] != NotReachable);
        }
```

例子：

```
        - (void)viewWillAppear:(BOOL)animated {
        if (([Reachability reachabilityForInternetConnection].currentReachabilityStatus == NotReachable) &&
                ([Reachability reachabilityForLocalWiFi].currentReachabilityStatus == NotReachable)) {
                self.navigationItem.hidesBackButton = YES;
                [self.navigationItem setLeftBarButtonItem:nil animated:NO];
            }
        }
```

4. 链接状态的实时通知

网络连接状态的实时检查，通知在网络应用中也是十分必要的。接续状态发生变化时，需要及时地通知用户：

Reachability 1.5 版本

```
        // My.AppDelegate.h
        #import "Reachability.h"

        @interface MyAppDelegate : NSObject <UIApplicationDelegate> {
            NetworkStatus remoteHostStatus;
        }

        @property NetworkStatus remoteHostStatus;

        @end
```

```
// My.AppDelegate.m
#import "MyAppDelegate.h"

@implementation MyAppDelegate
@synthesize remoteHostStatus;

// 更新网络状态
- (void)updateStatus {
    self.remoteHostStatus = [[Reachability sharedReachability] remoteHostStatus];
}

// 通知网络状态
- (void)reachabilityChanged:(NSNotification *)note {
    [self updateStatus];
    if (self.remoteHostStatus == NotReachable) {
        UIAlertView *alert = [[UIAlertView alloc]
initWithTitle:NSLocalizedString(@"AppName", nil)
                              message:NSLocalizedString (@"NotReachable", nil)
                             delegate:nil cancelButtonTitle:@"OK" otherButtonTitles: nil];
        [alert show];
        [alert release];
    }
}

// 程序启动器，启动网络监视
- (void)applicationDidFinishLaunching:(UIApplication *)application {

    // 设置网络检测的站点
    [[Reachability sharedReachability] setHostName:@"www.apple.com"];
    [[Reachability sharedReachability] setNetworkStatusNotificationsEnabled:YES];
    // 设置网络状态变化时的通知函数
    [[NSNotificationCenter defaultCenter] addObserver:self
selector:@selector(reachabilityChanged:)

name:@"kNetworkReachabilityChangedNotification" object:nil];
    [self updateStatus];
}

- (void)dealloc {
    // 删除通知对象
    [[NSNotificationCenter defaultCenter] removeObserver:self];
    [window release];
    [super dealloc];
}
```

Reachability 2.0 版本

```
// MyAppDelegate.h
@class Reachability;

    @interface MyAppDelegate : NSObject <UIApplicationDelegate> {
        Reachability    *hostReach;
    }

@end

// MyAppDelegate.m
- (void)reachabilityChanged:(NSNotification *)note {
    Reachability* curReach = [note object];
    NSParameterAssert([curReach isKindOfClass: [Reachability class]]);
    NetworkStatus status = [curReach currentReachabilityStatus];

    if (status == NotReachable) {
        UIAlertView *alert = [[UIAlertView alloc] initWithTitle:@"AppName""
                              message:@"NotReachable"
                              delegate:nil
                              cancelButtonTitle:@"YES" otherButtonTitles:nil];
                              [alert show];
                              [alert release];
    }
}

- (void)applicationDidFinishLaunching:(UIApplication *)application {
    // ...

    // 监测网络情况
    [[NSNotificationCenter defaultCenter] addObserver:self
                              selector:@selector(reachabilityChanged:)
                              name: kReachabilityChangedNotification
                              object: nil];
    hostReach = [[Reachability reachabilityWithHostName:@"www.google.com"]
retain];
    hostReach startNotifer];
    // ...
}
```

二：使用 NSConnection 下载数据

1.创建 NSConnection 对象，设置委托对象

NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:[NSURL URLWithString:[self urlString]]];
[NSURLConnection connectionWithRequest:request delegate:self];

2. NSURLConnection delegate 委托方法

- (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response;
- (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error;
- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data;
- (void)connectionDidFinishLoading:(NSURLConnection *)connection;

3. 实现委托方法

```
- (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response {
    // store data
    [self.receivedData setLength:0];                //通常在这里先清空接受数据的缓存
}

- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data {
    /* appends the new data to the received data */
    [self.receivedData appendData:data];            //可能多次收到数据，把新的数据添加在现有数据最后
}

- (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error {
    // 错误处理
}

- (void)connectionDidFinishLoading:(NSURLConnection *)connection {
    // disconnect
    [UIApplication sharedApplication].networkActivityIndicatorVisible = NO;
    NSString *returnString = [[NSString alloc] initWithData:self.receivedData encoding:NSUTF8StringEncoding];
    NSLog(returnString);
    [self urlLoaded:[self urlString] data:self.receivedData];
    firstTimeDownloaded = YES;
```

}

三：使用 NSXMLParser 解析 xml 文件

1. 设置委托对象，开始解析

NSXMLParser *parser = [[NSXMLParser alloc] initWithData:data]; //或者也可以使用 initWithContentsOfURL 直接下载文件，但是有一个原因不这么做：

// It's also possible to have NSXMLParser download the data, by passing it a URL, but this is not desirable

// because it gives less control over the network, particularly in responding to connection errors.

[parser setDelegate:self];

[parser parse];

2. 常用的委托方法

```
- (void)parser:(NSXMLParser *)parser didStartElement:(NSString *)elementName
                                    namespaceURI:(NSString *)namespaceURI
                                    qualifiedName:(NSString *)qName
                                    attributes:(NSDictionary *)attributeDict;
- (void)parser:(NSXMLParser *)parser didEndElement:(NSString *)elementName
                                    namespaceURI:(NSString *)namespaceURI
                                    qualifiedName:(NSString *)qName;
- (void)parser:(NSXMLParser *)parser foundCharacters:(NSString *)string;
- (void)parser:(NSXMLParser *)parser parseErrorOccurred:(NSError *)parseError;
```

static NSString *feedURLString = @"http://www.yifeiyang.net/test/test.xml";

3. 应用举例

```
- (void)parseXMLFileAtURL:(NSURL *)URL parseError:(NSError **)error
{
    NSXMLParser *parser = [[NSXMLParser alloc] initWithContentsOfURL:URL];
    [parser setDelegate:self];
    [parser setShouldProcessNamespaces:NO];
    [parser setShouldReportNamespacePrefixes:NO];
    [parser setShouldResolveExternalEntities:NO];
    [parser parse];
    NSError *parseError = [parser parserError];
    if (parseError && error) {
        *error = parseError;
    }
    [parser release];
}
```

- (void)parser:(NSXMLParser *)parser didStartElement:(NSString *)elementName

```
namespaceURI:(NSString *)namespaceURI
                                        qualifiedName:(NSString*)qName
attributes:(NSDictionary *)attributeDict{
        // 元素开始句柄
        if (qName) {
            elementName = qName;
        }
        if ([elementName isEqualToString:@"user"]) {
            // 输出属性值
            NSLog(@"Name is %@ , Age is %@", [attributeDict
objectForKey:@"name"], [attributeDict objectForKey:@"age"]);
        }
    }

    - (void)parser:(NSXMLParser *)parser didEndElement:(NSString *)elementName
namespaceURI:(NSString *)namespaceURI
                                        qualifiedName:(NSString *)qName
    {
        // 元素终了句柄
        if (qName) {
            elementName = qName;
        }
    }

    - (void)parser:(NSXMLParser *)parser foundCharacters:(NSString *)string
    {
        // 取得元素的 text
    }

    NSError *parseError = nil;
    [self parseXMLFileAtURL:[NSURL URLWithString:feedURLString]
parseError:&parseError];
```

Iphone 实现画折线图

iphone 里面要画图一般都是通过 CoreGraphics.framwork 和 QuartzCore.framwork 实现，apple 的官方 sdk demon 中包含了 QuartzCore 的基本用法，

具体 demo 请参考 http://developer.apple.com/library/ios/#samplecode/QuartzDemo/

折线图

要实现折线图也就把全部的点连起来，movePointLineto，具体的调用里面的 api 就可以实现了，但是画坐标就比较麻烦了，里面需要去转很多，好在国外有人开源了一个画折线图的开发包，首先看看效果吧，具体怎么用可以参考作者 git 版本库中的 wiki。

http://github.com/devinross/tapkulibrary/wiki/How-To-Use-This-Library

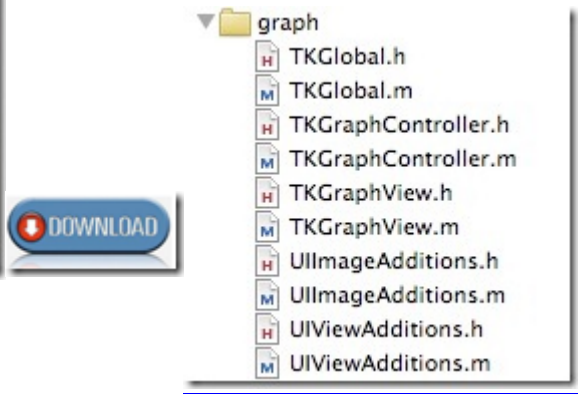
这个包还提供了其他的很好看的 UI，都可以调来用，但是我们只需要一个画图要把整个包都导进去，工程太大了，既然是开源的那就想办法提取出来吧，原先之前也有人干过这样的事 。

http://duivesteyn.net/2010/03/07/iphone-sdk-implementing-the-tapku-graph-in-your-application/

我对源代码进行简单的修改，使其显示坐标之类的，更加符合工程的需要，但是还没有实现画多组数据，只能画一组数据，不用 viewContol，而使用 addsubview，直接添加到当前的窗

口，最终效果如下。
使用方法：

1.工程添加 tk 库里面的如下文件



2. 添加 QuartzCore framework
#import <QuartzCore/QuartzCore.h>
添加 TapkuLibrary.bundle 资源文件
3.代码中完成实例，数据初始化就可以用了

```
graph = [[TKGraphView alloc] initWithFrame:CGRectMake(10,100, 300, 150)];

graph.title.text = @"Data Graph";
[graph setPointDistance:15];

data = [[NSMutableArray alloc] init];

srand((int)time(0));
for(int i=0;i<20;i++){
    int no = rand() % 100 + i;

    GraphPoint *gp = [[GraphPoint alloc] initWithID:i value:[NSNumber numberWithFloat:no]];
    [data addObject:gp];
    [gp release];
}


[graph setGraphWithDataPoints:data];
```

下载修改后的版本。下次有时间在整理一个工程版本出来。

让 iPhone 屏幕常亮不变暗的方法

如果您希望运行自己开发的 App 时，iPhone 的屏幕不再自动变暗，可以使用以下方法让屏幕常亮： iPhone OS 用一个布尔值用来控制是否取消应用程序空闲时间：@property(nonatomic, getter=isIdleTime

如果您希望运行自己开发的 App 时，iPhone 的屏幕不再自动变暗，可以使用以下方法让屏幕常亮：

iPhone OS 用一个布尔值用来控制是否取消应用程序空闲时间：@property(nonatomic, getter=isIdleTimerDisabled) BOOL idleTimerDisabled。这个值的默认属性是"NO"。当大多数应用程序没有接收到用户输入信息的时候，系统会把设备设置成“休眠”状态，iPhone 屏幕也会变暗。这样做是为了保存更多电量。事实上，应用程序在运行加速度游戏的时候是不需

要用户输入的，当然这里只是一个假设，把这个变量设置为"YES"，来取消系统休眠的“空闲时间”。

重点是：你必须当真正需要的时候才打开这个属性当你不用的时候马上还愿成"NO"。大多数应用程序在休眠时间到的时候让系统关闭屏幕。这个包括了有音频的应用程 序。在 Audio Session Services 中使用适当的回放和记录功能不会被间断当屏幕关闭时。只有地图应用程序，游戏或者一些不间断的用户交互程序可以取消这个属性。

苹果开发网络编程知识总结

以下苹果开发网络编程知识由 CocoaChina 会员 cocoa_yang 总结，希望能为苹果开发新手梳理知识脉络，节省入门时间。一：确认网络环境 3G/WIFI 1. 添加源文件和 framework 开发 Web 等网络应用程序

以下苹果开发网络编程知识由 CocoaChina 会员 “cocoa_yang” 总结，希望能为苹果开发新手梳理知识脉络，节省入门时间。

一：确认网络环境 3G/WIFI

1. 添加源文件和 framework

开发 Web 等网络应用程序的时候，需要确认网络环境，连接情况等信息。如果没有处理它们，是不会通过 Apple 的审查的。

Apple 的 例程 Reachability 中介绍了取得/检测网络状态的方法。要在应用程序程序中使用 Reachability，首先要完成如下两部：

1.1. 添加源文件：

在你的程序中使用 Reachability 只须将该例程中的 Reachability.h 和 Reachability.m 拷贝到你的工程中。如下图：

1.2.添加 framework：

将 SystemConfiguration.framework 添加进工程。如下图：

2. 网络状态

Reachability.h 中定义了三种网络状态：

```
typedef enum {
    NotReachable = 0,                 //无连接
    ReachableViaWiFi,                  //使用 3G/GPRS 网络
    ReachableViaWWAN                    //使用 WiFi 网络
} NetworkStatus;
```

因此可以这样检查网络状态：

```
Reachability *r = [Reachability reachabilityWithHostName:@“www.apple.com”];
switch ([r currentReachabilityStatus]) {
        case NotReachable:
                // 没有网络连接
                break;
        case ReachableViaWWAN:
                // 使用 3G 网络
                break;
        case ReachableViaWiFi:
                // 使用 WiFi 网络
                break;
}
```

3.检查当前网络环境

程序启动时，如果想检测可用的网络环境，可以像这样

```
// 是否 wifi
+ (BOOL) IsEnableWIFI {
    return ([[Reachability reachabilityForLocalWiFi] currentReachabilityStatus] != NotReachable);
}

// 是否 3G
+ (BOOL) IsEnable3G {
    return ([[Reachability reachabilityForInternetConnection] currentReachabilityStatus] != NotReachable);
}
```

例子：

```
- (void)viewWillAppear:(BOOL)animated {
if (([Reachability reachabilityForInternetConnection].currentReachabilityStatus == NotReachable) &&
        ([Reachability reachabilityForLocalWiFi].currentReachabilityStatus == NotReachable)) {
        self.navigationItem.hidesBackButton = YES;
        [self.navigationItem setLeftBarButtonItem:nil animated:NO];
    }
}
```

4. 链接状态的实时通知

网络连接状态的实时检查，通知在网络应用中也是十分必要的。接续状态发生变化时，需要及时地通知用户：

Reachability 1.5 版本

```
// My.AppDelegate.h
#import "Reachability.h"

@interface MyAppDelegate : NSObject <UIApplicationDelegate> {
    NetworkStatus remoteHostStatus;
}

@property NetworkStatus remoteHostStatus;

@end

// My.AppDelegate.m
#import "MyAppDelegate.h"

@implementation MyAppDelegate
@synthesize remoteHostStatus;

// 更新网络状态
- (void)updateStatus {
    self.remoteHostStatus = [[Reachability sharedReachability] remoteHostStatus];
}

// 通知网络状态
- (void)reachabilityChanged:(NSNotification *)note {
    [self updateStatus];
    if (self.remoteHostStatus == NotReachable) {
        UIAlertView *alert = [[UIAlertView alloc]
initWithTitle:NSLocalizedString(@"AppName", nil)
                                message:NSLocalizedString (@"NotReachable", nil)
                               delegate:nil cancelButtonTitle:@"OK" otherButtonTitles: nil];
        [alert show];
        [alert release];
    }
}

// 程序启动器，启动网络监视
- (void)applicationDidFinishLaunching:(UIApplication *)application {

    // 设置网络检测的站点
    [[Reachability sharedReachability] setHostName:@"www.apple.com"];
    [[Reachability sharedReachability] setNetworkStatusNotificationsEnabled:YES];
    // 设置网络状态变化时的通知函数
    [[NSNotificationCenter defaultCenter] addObserver:self
```

```
selector:@selector(reachabilityChanged:)

name:@"kNetworkReachabilityChangedNotification" object:nil];
        [self updateStatus];
    }

    - (void)dealloc {
        // 删除通知对象
        [[NSNotificationCenter defaultCenter] removeObserver:self];
        [window release];
        [super dealloc];
    }
```

Reachability 2.0 版本

```
    // MyAppDelegate.h
    @class Reachability;

        @interface MyAppDelegate : NSObject <UIApplicationDelegate> {
            Reachability    *hostReach;
        }

    @end

    // MyAppDelegate.m
    - (void)reachabilityChanged:(NSNotification *)note {
        Reachability* curReach = [note object];
        NSParameterAssert([curReach isKindOfClass: [Reachability class]]);
        NetworkStatus status = [curReach currentReachabilityStatus];

        if (status == NotReachable) {
            UIAlertView *alert = [[UIAlertView alloc] initWithTitle:@"AppName""
                                  message:@"NotReachable"
                                  delegate:nil
                                  cancelButtonTitle:@"YES" otherButtonTitles:nil];
                                  [alert show];
                                  [alert release];
        }
    }

    - (void)applicationDidFinishLaunching:(UIApplication *)application {
        // ...
```

```
    // 监测网络情况
    [[NSNotificationCenter defaultCenter] addObserver:self
                                             selector:@selector(reachabilityChanged:)
                                             name: kReachabilityChangedNotification
                                             object: nil];
    hostReach = [[Reachability reachabilityWithHostName:@"www.google.com"] retain];
    hostReach startNotifer];
    // ...
}
```

二：使用 NSConnection 下载数据

1.创建 NSConnection 对象，设置委托对象

```
NSMutableURLRequest *request = [NSMutableURLRequest requestWithURL:[NSURL URLWithString:[self urlString]]];
[NSURLConnection connectionWithRequest:request delegate:self];
```

2. NSURLConnection delegate 委托方法

```
- (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response;
- (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error;
- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data;
- (void)connectionDidFinishLoading:(NSURLConnection *)connection;
```

3. 实现委托方法

```
- (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response {
    // store data
    [self.receivedData setLength:0];                //通常在这里先清空接受数据的缓存
}

- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data {
    /* appends the new data to the received data */
    [self.receivedData appendData:data];            //可能多次收到数据，把新的数据添加在现有数据最后
}

- (void)connection:(NSURLConnection *)connection didFailWithError:(NSError
```

```
*)error {
        // 错误处理
    }

    - (void)connectionDidFinishLoading:(NSURLConnection *)connection {
        // disconnect
        [UIApplication sharedApplication].networkActivityIndicatorVisible = NO;
        NSString *returnString = [[NSString alloc] initWithData:self.receivedData encoding:NSUTF8StringEncoding];
        NSLog(returnString);
        [self urlLoaded:[self urlString] data:self.receivedData];
        firstTimeDownloaded = YES;
    }
```

三：使用 NSXMLParser 解析 xml 文件

1. 设置委托对象，开始解析

```
NSXMLParser *parser = [[NSXMLParser alloc] initWithData:data];    //或者也可以使用 initWithContentsOfURL 直接下载文件，但是有一个原因不这么做：
// It's also possible to have NSXMLParser download the data, by passing it a URL, but this is not desirable
// because it gives less control over the network, particularly in responding to connection errors.
[parser setDelegate:self];
[parser parse];
```

2. 常用的委托方法

```
- (void)parser:(NSXMLParser *)parser didStartElement:(NSString *)elementName
                                    namespaceURI:(NSString *)namespaceURI
                                    qualifiedName:(NSString *)qName
                                    attributes:(NSDictionary *)attributeDict;
- (void)parser:(NSXMLParser *)parser didEndElement:(NSString *)elementName
                                    namespaceURI:(NSString *)namespaceURI
                                    qualifiedName:(NSString *)qName;
- (void)parser:(NSXMLParser *)parser foundCharacters:(NSString *)string;
- (void)parser:(NSXMLParser *)parser parseErrorOccurred:(NSError *)parseError;

static NSString *feedURLString = @"http://www.yifeiyang.net/test/test.xml";
```

3. 应用举例

```
- (void)parseXMLFileAtURL:(NSURL *)URL parseError:(NSError **)error
{
    NSXMLParser *parser = [[NSXMLParser alloc] initWithContentsOfURL:URL];
    [parser setDelegate:self];
```

```
        [parser setShouldProcessNamespaces:NO];
        [parser setShouldReportNamespacePrefixes:NO];
        [parser setShouldResolveExternalEntities:NO];
        [parser parse];
        NSError *parseError = [parser parserError];
        if (parseError && error) {
            *error = parseError;
        }
        [parser release];
    }

    - (void)parser:(NSXMLParser *)parser didStartElement:(NSString *)elementName
namespaceURI:(NSString *)namespaceURI
                                        qualifiedName:(NSString*)qName
attributes:(NSDictionary *)attributeDict{
        // 元素开始句柄
        if (qName) {
            elementName = qName;
        }
        if ([elementName isEqualToString:@"user"]) {
            // 输出属性值
            NSLog(@"Name is %@ , Age is %@", [attributeDict
objectForKey:@"name"], [attributeDict objectForKey:@"age"]);
        }
    }

    - (void)parser:(NSXMLParser *)parser didEndElement:(NSString *)elementName
namespaceURI:(NSString *)namespaceURI
                                        qualifiedName:(NSString *)qName
    {
        // 元素终了句柄
        if (qName) {
            elementName = qName;
        }
    }

    - (void)parser:(NSXMLParser *)parser foundCharacters:(NSString *)string
    {
        // 取得元素的 text
    }

    NSError *parseError = nil;
    [self parseXMLFileAtURL:[NSURL URLWithString:feedURLString]
parseError:&parseError];
```

如何隐藏状态栏

[UIApplication sharedApplication].statusBarHidden = YES;

.m 文件与.mm 文件的区别

.m 文件是 object-c 文件

.mm 文件相当于 c++或者 c 文件

NSLog(@"afd")与 NSLog("afd")

细微差别会导致程序崩溃。

但是我不太明白为何苹果要把编译器做的对这两种常量有区别。

不过值得一提的是可能为了方便苹果自身的 NSObject 对象的格式化输出。

safari 其实没有把内存的缓存写到存储卡上

NSURLCache doesn't seem to support writing to disk on iPhone. The documentation for NSCachedURLResponse says that the NSURLCacheStoragePolicy "NSURLCacheStorageAllowed" is treated as "NSURLCacheStorageAllowedInMemoryOnly" by iPhone OS.

官方文档是这么说的。

为了证明这个，我找到了一个目录。

/private/var/mobile/Library/Caches/Safari/Thumbnails

随机数的使用

头文件的引用

```
#import <time.h>
#import <mach/mach_time.h>
```

srandom()的使用

```
srandom((unsigned)(mach_absolute_time() & 0xFFFFFFFF));
```

直接使用 random() 来调用随机数

在 UIImageView 中旋转图像

```
float rotateAngle = M_PI;
CGAffineTransform transform =CGAffineTransformMakeRotation(rotateAngle);
```

imageView.transform = transform;

以上代码旋转 imageView，角度为 rotateAngle，方向可以自己测试哦！

在 Quartz 中如何设置旋转点

UIImageView *imageView = [[UIImageView alloc] initWithImage:[UIImage imageNamed:@"bg.png"]];
imageView.layer.anchorPoint = CGPointMake(0.5, 1.0);

这个是把旋转点设置为底部中间。记住是在 QuartzCore.framework 中才得到支持。

创建.plist 文件并存储

```
NSString *errorDesc;  //用来存放错误信息
NSMutableDictionary *rootObj = [NSMutableDictionary dictionaryWithCapacity:4]; //NSDictionary, NSData 等文件可以直接转化为 plist 文件
NSDictionary *innerDict;
NSString *name;
Player *player;
NSInteger saveIndex;

for(int i = 0; i < [playerArray count]; i++) {
    player = nil;
    player = [playerArray objectAtIndex:i];
    if(player == nil)
        break;
    name = player.playerName;// This "Player1" denotes the player name could also be the computer name
    innerDict = [self getAllNodeInfoToDictionary:player];
    [rootObj setObject:innerDict forKey:name]; // This "Player1" denotes the person who start this game
}
player = nil;
NSData *plistData = [NSPropertyListSerialization dataFromPropertyList:(id)rootObj format:NSPropertyListXMLFormat_v1_0 errorDescription:&errorDesc];
```

红色部分可以忽略，只是给 rootObj 添加一点内容。这个 plistData 为创建好的 plist 文件，用其 writeToFile 方法就可以写成文件。下面是代码：

```
/*得到移动设备上的文件存放位置*/
```

```
NSString *documentsPath = [self getDocumentsDirectory];
NSString *savePath = [documentsPath stringByAppendingPathComponent:@"save.plist"];

/*存文件*/
if (plistData) {
    [plistData writeToFile:savePath atomically:YES];
 }
 else {
    NSLog(errorDesc);
    [errorDesc release];
}

- (NSString *)getDocumentsDirectory {
    NSArray *paths = NSSearchPathForDirectoriesInDomains(NSDocumentDirectory, NSUserDomainMask, YES);
    return [paths objectAtIndex:0];
}
```

读取 plist 文件并转化为 NSDictionary

```
NSString *documentsPath = [self getDocumentsDirectory];
NSString *fullPath = [documentsPath stringByAppendingPathComponent:@"save.plist"];
NSMutableDictionary* plistDict = [[NSMutableDictionary alloc] initWithContentsOfFile:fullPath];
```

读取一般性文档文件

```
NSString *tmp;
NSArray *lines; /*将文件转化为一行一行的*/
lines = [[NSString stringWithContentsOfFile:@"testFileReadLines.txt"]
                componentsSeparatedByString:@"\n"];

 NSEnumerator *nse = [lines objectEnumerator];

 // 读取<>里的内容
 while(tmp = [nse nextObject]) {
        NSString *stringBetweenBrackets = nil;
        NSScanner *scanner = [NSScanner scannerWithString:tmp];
        [scanner scanUpToString:@"<" intoString:nil];
        [scanner scanString:@"<" intoString:nil];
        [scanner scanUpToString:@">" intoString:&stringBetweenBrackets];
```

```
        NSLog([stringBetweenBrackets description]);
    }
```

对于读写文件，还有补充，暂时到此。随机数和文件读写在游戏开发中经常用到。所以把部分内容放在这，以便和大家分享，也当记录，便于查找。

隐藏 NavigationBar

```
[self.navigationController setNavigationBarHidden:YES animated:YES];
```

在想隐藏的 ViewController 中使用就可以了。

如何在 iPhone 程序中调用外部命令

下面是如何在 iPhone 非官方 SDK 程序中调用外部命令的方法。

- (NSString *) executeCommand : (NSString *) cmd { NSString * output = [NSString string] ; FILE * pipe = popen ([cmd cStringUsingEncoding : NSASCIIStringEnc

下面是如何在 iPhone 非官方 SDK 程序中调用外部命令的方法。

```
- (NSString *)executeCommand: (NSString *)cmd
{
    NSString *output = [NSString string];
    FILE *pipe = popen([cmd cStringUsingEncoding: NSASCIIStringEncoding], "r");
    if (!pipe) return;

    char buf[1024];
    while(fgets(buf, 1024, pipe)) {
    output = [output stringByAppendingFormat: @"%s", buf];
}

pclose(pipe);
return output;
}

NSString *yourcmd = [NSString stringWithFormat: @"your command"];
[self executeCommand: yourcmd];
```

如何在 iPhone 程序读取数据时显示进度窗

下面代码说明如何使用 iPhone 非官方 SDK 在读取数据时显示进度条。

以下代码参考了 MobileRss。

定义头文件：

```
#import "uikit/UIProgressHUD.h"

@interface EyeCandy : UIApplication {
  UIProgressHUD *progress;
}

- (void) showProgressHUD:(NSString *)label withWindow:(UIWindow *)w withView:(UIView *)v withRect:(struct CGRect)rect;
- (void) hideProgressHUD;

.@end
```

上面的引号要改成<>。

```
import "EyeCandy.h"

@implementation EyeCandy
- (void)showProgressHUD:(NSString *)label withWindow:(UIWindow *)w withView:(UIView *)v withRect:(struct CGRect)rect
{
  progress = [[UIProgressHUD alloc] initWithWindow: w];
  [progress setText: label];
  [progress drawRect: rect];
  [progress show: YES];

  [v addSubview:progress];
}

- (void)hideProgressHUD
{
  [progress show: NO];
  [progress removeFromSuperview];
}

@end
```

使用下面代码调用：

```
// Setup Eye Candy View
_eyeCandy = [[[EyeCandy alloc] init] retain];

// Call loading display
```

[_eyeCandy showProgressHUD:@"Loading …" withWindow:window withView:mainView withRect:CGRectMake(0.0f, 100.0f, 320.0f, 50.0f)];

// When finished for hiding the "loading text";
[_eyeCandy hideProgressHUD];

WebKit 的基本用法

WebKit 是苹果开发中比较常用的浏览器引擎，Safari 使用的正是 WebKit 引擎。WebKit 基于 KDE 的 KHTML 加以再开发，解析速度超过了以往所有的浏览器。这里简单记录一下 WebKit 的基本用法。

WebKit 由下面的结构组成：

- DomCore
- JavaScriptCore
- WebCore

一般浏览

要打开网页，可以这样做：

1.[[webView mainFrame] loadRequest:[NSURLRequest requestWithURL:[NSURL URLWithString:urlText]]];
DomCore

DomCore 用于处理 DOM 文档，包括：

- DOMDocument
- DOMNamedNodeMap
- DOMNode
- DOMNodeList

要获取一个 DOMDocument，可以这样做：

1.DOMDocument *myDOMDocument = [[webView mainFrame] DOMDocument];
要用于 HTML 处理，可以使用 DOMHTMLDocument（Mac OS X 10.4 之后），获取方式相同：

1.DOMHTMLDocument *myDOMDocument = (DOMHTMLDocument*)[[webView mainFrame] DOMDocument];
方法定义：

苹果的 WebKit 更新说明

JavaScriptCore

在 WebKit 中执行脚本的方法：

```
1.WebScriptObject *myscript = [webView windowScriptObject];
2.NSString *script = @"alert('hello');";
3.[myscript evaluateWebScript script];
```

参考：

http://www.macgood.com/thread-24636-1-1.html

http://www.cocoadev.com/index.pl?WebKit

为什么不要做 iPhone 上面的应用

简单来说就是因为两国的文化不同，或者说生活方式的不同。美国不管多穷的人都有车，他们平时的生活方式和国内绝对是完全不同的。做应用和做游戏不一样，应用需要满足人们某一

简单来说就是因为两国的文化不同，或者说生活方式的不同。美国不管多穷的人都有车，他们平时的生活方式和国内绝对是完全不同的。做应用和做游戏不一样，应用需要满足人们某一部分的需求，比如，一个计算小费的软件，在国内不会有市场，可是美国人都有一个。

大家可以设身处地的想一下，谁会需要你做的软件，这样的人有多少，这样的人又有 iPhone 的又有多少。

对于应用来说，针对商务人士的又比针对普通人的好，基本上商务人士不太在乎几块钱一个软件，这也是 backup assistant 卖得最好的一个原因。这个软件一年的年费 24 美元，大约有数千万美元一年的收入。什么样的应用软件是这些人需要的？连笔者自己也不太清楚，笔者虽然已经在美国工作了多年，但是对于美国文化的了解还处于一知半解状态，更不用说正在留学的学生了。

还有一个能成功的应用软件是你已经有非常多的数据，比如你有当地的所有加油站的信息，做一个油价的地图软件，显然市场会不错。不过数据要是美国的数据，国内的没有太大的帮助。

综上所述，游戏比应用好做很多，如果要作应用的话，可以从单机的小应用开始。要在美国运营一个支持 10 万人的网络应用，没有 30 万美元绝对没戏。如果非要上，只能早死早超生了。

获取 iPhone 用户手机号

使用下面的函数可以返回用户的手机号：

```
extern NSString *CTSettingCopyMyPhoneNumber();
```

然后调用即可。

由于这个函数是包含在 CoreTelephony 中，所以只能用于非官方 iPhone SDK。

在程序中关闭 iPhone

首先在程序中引用 #include sys/reboot.h 然后使用 reboot(RB_HALT); 就可以直接将 iPhone 关机。

首先在程序中引用

#include <sys/reboot.h>

然后使用

reboot(RB_HALT);

就可以直接将 iPhone 关机。

convert the contents of an NSData object to an NSString

1. NSString *stringFromASC = [NSString stringWithCString:[ascData bytes] length:[ascData length]];

If the NSData object contains unichar characters then do this:

NSString *stringFromUnichar = [NSString stringWithCharacters:[unicharData bytes] length:[unicharData length] / sizeof(unichar)];

2. - (id)initWithData:(NSData *)data encoding:(NSStringEncoding)encoding

iPhone 的特殊 URL

在 iPhone 中，可以直接用 UIApp 打开 URL 地址。如下所示：

1.[UIApp openURL: [NSURL URLWithString:@"http://www.apple.com"]];
或者：

1.[UIApp openURL: [NSURL URLWithString:@"mailto:apple@mac.com?Subject=hello"]];
与此同时，iPhone 还包含一些其他除了 http://或者 mailto:之外的 URL：

sms:// 可以调用短信程序

tel:// 可以拨打电话

itms:// 可以打开 MobileStore.app

audio-player-event:// 可以打开 iPod

audio-player-event://?uicmd=show-purchased-playlist 可以打开 iPod 播放列表

video-player-event:// 可以打开 iPod 中的视频

get iphone uniqueIdentifier

I also find that I can get uniqueIdentifier using:

UIDevice *myDevice = [UIDevice currentDevice];NSString *identifier = myDevice.uniqueIdentifier;

打开本地网页，与远程网页

fileURLWithPath:Initializes and returns a newly created NSURL object as a file URL with a specified path.

+ (id)fileURLWithPath:(NSString *)path

URLWithString:
Creates and returns an NSURL object initialized with a provided string.

+ (id)URLWithString:(NSString *)URLString

教你如何使用 UIWebView

Start by opening up the WebBrowserTutorialAppDelegate.h file and editing the @interface line to read:

@interface WebBrowserTutorialAppDelegate : NSObject <UIWebViewDelegate> {
What we have done is to make the main AppDelegate a delegate for the UIWebView as well.

Now we need to set our webView to have the main AppDelegate as its delegate, you can do this by opening up WebBrowserTutorialAppDelegate.m and putting the following line just inside theapplicationDidFinishLaunching function:

webView.delegate = self;
That is all pretty self explanatory, it just sets the delegate of our webView to self, which in this case is our main application delegate.

Now we are pretty much done, we just need to add the function to catch the link clicks. To do this we need to add a new function, copy the content below to the WebBrowserTutorialAppDelegate.m file:

```
- (BOOL)webView:(UIWebView*)webView
shouldStartLoadWithRequest:(NSURLRequest*)request
navigationType:(UIWebViewNavigationType)navigationType {
    NSURL *url = request.URL;
    NSString *urlString = url.absoluteString;
    NSLog(urlString);
    return YES;
}
```

This function will catch all requests and allow you to either manipulate them and pass them on or to perform your own custom action and stop the event from bubbling.

The first line gets the URL of the request, this is the contents inside the href attribute in the anchor tag.

The next line converts the URL to a string so we can log it out. You can access many parts of the NSURL, here are some of them and brief description of what they do.

* absoluteString - An absolute string for the URL. Creating by resolving the receiver's string against its base.
* absoluteURL - An absolute URL that refers to the same resource as the receiver. If the receiver is already absolute, returns self.
* baseURL - The base URL of the receiver. If the receiver is an absolute URL, returns nil.
* host - The host of the URL.
* parameterString - The parameter string of the URL.
* password - The password of the URL (i.e. http://user:pass@www.test.com would return pass)
* path - Returns the path of a URL.
* port - The port number of the URL.
* query - The query string of the URL.
* relativePath - The relative path of the URL without resolving against the base URL. If the receiver is an absolute URL, this method returns the same value as path.
* relativeString - string representation of the relative portion of the URL. If the receiver is an absolute URL this method returns the same value as absoluteString.
* scheme - The resource specifier of the URL (i.e. http, https, file, ftp, etc).
* user - The user portion of the URL.

Then the third line simply logs the URL to the console, so you will new to open up the console while you run this in the simulator to see the results.

Finally the forth line returns YES, this will allow the UIWebView to follow the link, if you

would just like to catch a link and stop the UIWebView from following it then simply return NO.

UIBUtton title image 不能同时显示

[leftbutton setTitle:_(@"About") forState:UIControlStateNormal];

[leftbutton setImage:image forState:UIControlStateNormal];

不能同时显示。

其他控件如：UINavigatonItem

不要在语言包里面设置空格

有时，为了界面的需要，我们不要在语言包里面加空格，要在程序中进行控制。

buttonTitle = [NSString stringWithFormat:@" %@", _(@"updateWeb")];

NSNotificationCenter 带参数发送

MPMoviePlayerController* theMovie = [[MPMoviePlayerController alloc]initWithContentURL:[NSURL fileURLWithPath:[[[tableTitles objectForKey:keyIndex] objectAtIndex:row] objectAtIndex:3]]];

[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(myMovieFinishedCallback:) name:MPMoviePlayerPlaybackDidFinishNotification object:theMovie];

[theMovie play];

-(void)myMovieFinishedCallback:(NSNotification*)aNotification

{

MPMoviePlayerController *theMovie = [aNotification object];

[[NSNotificationCenter defaultCenter] removeObserver:self name:MPMoviePlayerPlaybackDidFinishNotification object:theMovie];

// Release the movie instance [theMovie release];

}

－－－－－－－－－－－－－

MPMoviePlayerController* theMovie = [[MPMoviePlayerController alloc]initWithContentURL:[NSURL fileURLWithPath:[[[tableTitles objectForKey:keyIndex] objectAtIndex:row] objectAtIndex:3]]];

[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(myMovieFinishedCallback:) name:MPMoviePlayerPlaybackDidFinishNotification object:theMovie userInfo:dic];

[theMovie play];

-(void)myMovieFinishedCallback:(NSNotification*)aNotification

{

MPMoviePlayerController *theMovie = [aNotification object];

[[NSNotificationCenter defaultCenter] removeObserver:self name:MPMoviePlayerPlaybackDidFinishNotification object:theMovie];

// Release the movie instance [theMovie release];

}

延时一段时间执行某一函数

[self performSelector:@selector(dismissModal) withObject:self afterDelay:1.0];

无 99 美金证书联机开发

第一步：进入 cd /Developer/Platforms/iPhoneOS.platform/Developer/SDKs/iPhoneOS4.1.sdk/ sudo vi SDKSettings.plist,将 CODE_SIGNING_REQUIRED 的值改成 NO. 保存后退出.

第二步:重新启动 XCode 项目.

第三步:右击项目 GetInfo.将 Code Signing 下的 Code Signing Identity 值设置成 Don't Code Sign, 将 Code Signing Identity 下的 Any iOS Device 的值设置成空.

获取 IOS 设备的基本信息

系统唯一标识

是什么设备：iPad 还是 iPhone 等

iOS 版本号

系统名称

[[UIDevice currentDevice] uniqueIdentifier],

```
[[UIDevice currentDevice] localizedModel],
[[UIDevice currentDevice] systemVersion],
[[UIDevice currentDevice] systemName],
[[UIDevice currentDevice] model]];
```

用 NSDateFormatter 调整时间格式的代码

在开发 iOS 程序时，有时候需要将时间格式调整成自己希望的格式，这个时候我们可以用 NSDateFormatter 类来处理。

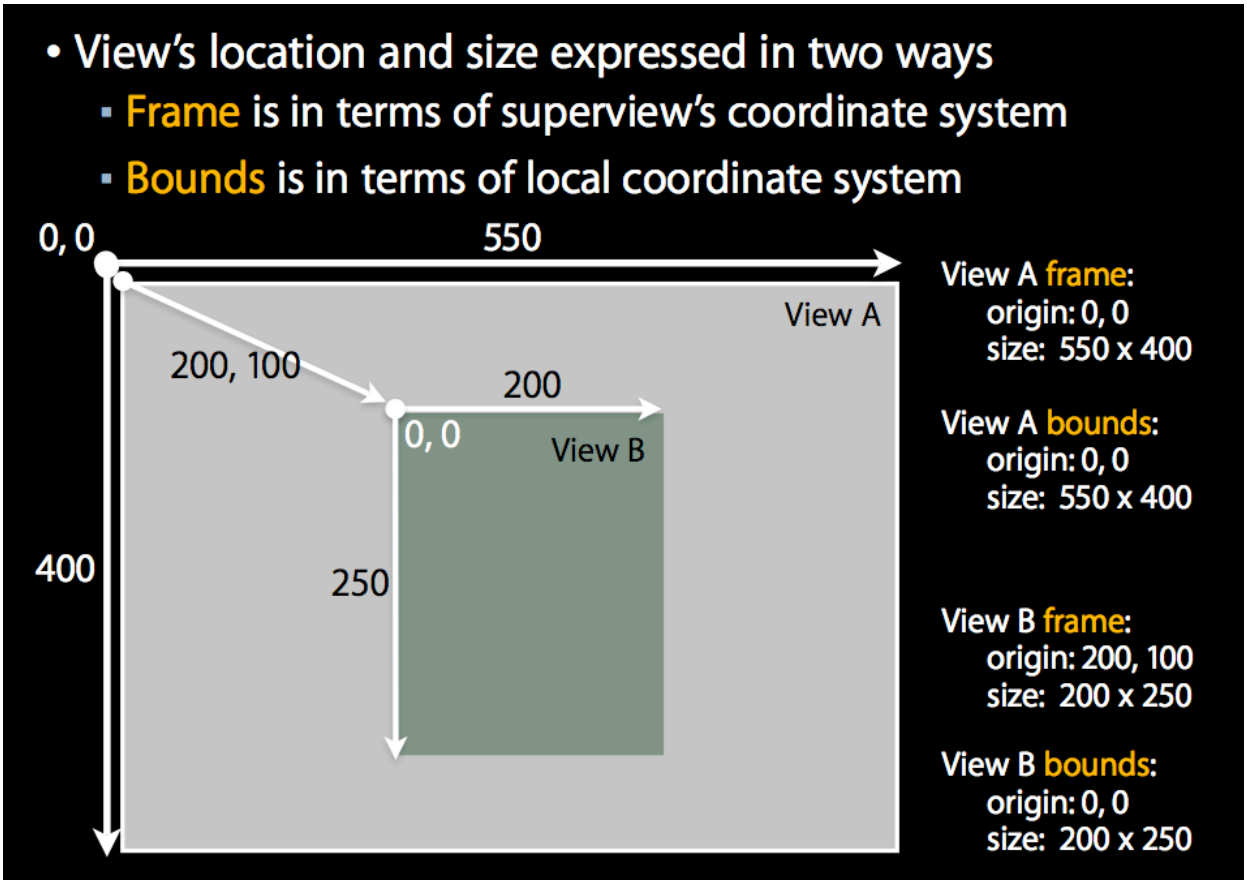
例如：

```
//实例化一个 NSDateFormatter 对象

NSDateFormatter *dateFormatter = [[NSDateFormatter alloc] init];

//设定时间格式,这里可以设置成自己需要的格式

[dateFormatter setDateFormat:@"yyyy-MM-dd HH:mm:ss"];

//用[NSDate date]可以获取系统当前时间

NSString *currentDateStr = [dateFormatter stringFromDate:[NSDate date]];

//输出格式为：2010-10-27 10:22:13

NSLog(@"%@",currentDateStr);

//alloc 后对不使用的对象别忘了 release

[dateFormatter release];
```

UIView 设置成圆角方法

```
m_mainImgView.layer.cornerRadius = 6;
m_mainImgView.layer.masksToBounds = YES;
```

iPhone 里的 frame 和 bounds 区别



Objective-C 内存管理

在使用 Objective-C 的工作中内存管理是首先要学会的一项技能，是如此重要，就好比是男人就要追漂亮姑娘一样～～下面就来聊聊 Apple 官网上的内存管理的事情。

Objective-C 的对象内存管理是一件非常有意思的事情，由其是在 iPhone 嵌入式设备中.

想玩的省心点，就得熟知它的管理规则，由其是内存的管理机制。了解它的品性了才能在 Cocoa 的世界里如鱼得水。否则，反之（如水得鱼!! ^_^）。

首先，要牢记 Apple 的官网上的内存管理三定律：

1，一个对象可以有一个或多个拥有者

2，当它一个拥有都都没有时，它就会被回收

3，如果想保留一个对象不被回收，你就必需成为它的拥有者

所有内存管理的原则全在这里!!

简单？？哈哈!

名人曰：“大道至简”

这儿玩意儿说起来比过家家还容易，但其实有些事情真正做起来并不是简单的事儿~~

咱们首先来说怎么样才能成为一个对象的拥有者。Cocoa 提供了一个机制叫"reference counting"，翻译过来就是“关联记数器”（自己翻译的，真不知叫啥，如果有官方的翻译请通知我）。每一个对象都有一个关联记数的值。当它被创建时，它的值为“1”。当值减少到“0”时，就会被回收(调用它的 deallocate 方法，如果没有写，则调用从 NSObject 继承而来的回收方法，下文有说，一定要重写该方法)。以下几个方法可以操作这个记数：

1，alloc
为对象分配内存，记数设为“1”，并返回此对象。

2，copy
复制一个对象，此对象记数为“1”，返回此对象。你将成为此克隆对象的拥有者

3，retain
对象“关联记数”加“1”，并成为此对象的拥有者。

4，release
对象“关联记数”减“1”，并丢掉此对象。

5，autorelease

在未来的某一时刻，对象“关联记数”减“1”。并在未来的某个时间放弃此对象。

有了上面的几个方法（当然这也是所有的内存操作的方法，简单吧，哈哈哈）你就可以随意操作一个对象的记数。并部分或完全的控制它的生命周期。但实际应用中，随意乱写上面的任何一个方法都可能会带来严重的内存泄露。混乱的内存分配等于没完没了的麻烦工作，你不想在情人节的日子还在为记数之类的鸟问题而丢了老婆吧~~哈哈哈，为了美丽温柔贤惠又善解人意的准老婆请牢记以下四条：

1，一个代码块内要确保 copy, alloc 和 retain 的使用数量与 release 和 autorelease 的数量。

2，在使用以“alloc”或“new”开头或包含“copy”的方法，或“retain”一个对象时，你就会变为它的拥有者。

3，实现“dealloc”方法，并施放所有的实例变量。（其实这里还有很多的巧儿门!!）

4，永不自己调用“dealloc”方法，这是系统当“retain”减到“0”时，自动调用的。手动调用会引起 retain count 记数错误（多一次的 release）。

其实做到这些也不难，

retain count 增加与减少的方法对应，板丁板做到了就行了。

来自：http://blog.csdn.net/dboylx/archive/2009/02/13/3888746.aspx

iphone 更改键盘右下角按键的 type

以 UISearchBar 为例。

创建 mySearchBar:

```
mySearchBar = [[UISearchBar alloc] initWithFrame:CGRectMake(0.0, 0,320, SEARCH_HEIGHT)];
mySearchBar.placeholder = curPath;
[mySearchBar setDelegate:self];
//tableView.tableHeaderView =mySearchBar;
[self.view addSubview:mySearchBar];
```

更改按键的 keyType(默认是 return,这里将它更改成 done，当然还可以更改成其他的):

```
UITextField *searchField = [[mySearchBar subviews] lastObject];
[searchField setReturnKeyType:UIReturnKeyDone];
[mySearchBar release];
```