iOS tutorials point

Yu
余加攀

Sep 21, 2015

Contents

1 iOS Objective-C basics **3**
1.1 Interface and Implementation . . . 3
1.2 Object Creation . . . 3
1.3 Methods . . . 3
1.3.1 Class Methods . . . 4
1.3.2 Instance Methods . . . 4
1.4 Important Data Types in Objective C . . . 4
1.5 Printing Logs . . . 4
1.6 Properties . . . 4
1.6.1 Accessing Properties . . . 4
1.7 Categories . . . 4
1.8 Arrays . . . 5
1.9 Dictionary . . . 5

2 iOS - Actions and Outlets **5**

3 iOS - Delegates You can refer to OCTP.tex 2.28 **5**

4 iOS - UI Elements **13**
4.1 UIImage and UIImageView . . . 13
4.2 UISwitch . . . 13
4.3 iOS - Text Field . . . 14
4.4 iOS - UITextView . . . 16
4.5 iOS - Label . . . 17
4.6 iOS - Toolbar . . . 18
4.7 iOS - Buttons . . . 18
4.8 iOS - Status Bar . . . 19
4.9 iOS - Navigation Bar . . . 20
4.10 iOS - Tab Bar . . . 22
4.11 UIPickerView . . . 23
4.12 UISlider . . . 24
4.13 UIAlertView . . . 25
4.14 UIActivityIndicator . . . 32
4.15 UIPageControl . . . 33
4.16 iOS - Table View . . . 35
4.17 UIScrollView . . . 51
4.18 NSTimer . . . 53
4.19 UIActionSheet . . . 54
4.20 UISegmentedControl . . . 55
4.21 UIImagePickerController . . . 56
4.22 UITapGestureRecognizer . . . 57
4.23 Draw a custom view . . . 59

4.24 programmatically-linking-uipagecontrol-to-uiscrollview . . . 61
4.25 Dismiss(resign) the keyboard . . . 62
4.26 Dealing with strings and NSLog . . . 63
4.27 Handling attributed string and text . . . 64
4.28 Dealing with sizes(all in points not pixels) . . . 66
4.29 what-is-the-difference-between-post-and-get . . . 66
4.30 Those characters refer to the source control. . . . 66
4.31 Bars . . . 67
4.32 NSNumber, id type . . . 67
4.33 NSURLConnection . . . 67
4.34 UIWebView . . . 69
4.35 NSUserDefaults UISwitch . . . 69
4.36 UIView Animation . . . 70
4.37 CALayer . . . 71
4.38 Push notifications (APNS) . . . 72
4.39 AFNetworking . . . 77
4.40 JSONModel . . . 79
4.41 Templates . . . 80
4.42 Install cocoapods . . . 80
4.42.1 Install cocoapods . . . 80
4.42.2 Setup project in xcode using cocoapods . . . 81
4.43 extern . . . 81

5 Important notes **82**
5.1 error-property-frame-not-found-on-object-of-type-uiview . . . 82
5.2 Working with stars . . . 82
5.3 how to add a view to the navigation bar . . . 83
5.4 how-to-detect-touches-in-status-bar . . . 83
5.5 Using Text Kit to Manage Text in Your iOS Apps . . . 83
5.6 Make a call . . . 83
5.7 Support different iOS versions . . . 83
5.8 compiler-error-initializer-element-is-not-a-compile-time-constant . . . 84
5.9 You can do the following . . . 85
5.10 Project structure . . . 85
5.11 removeFromSuperview method . . . 85
5.12 boundingRectWithSize method . . . 85
5.13 Difference between base sdk and deployment target . . . 85
5.14 For a info.plist on a lower version of ios see pic plistLowerVersion . . . 86
5.15 App Transport Security has blocked a cleartext HTTP (http://) resource load since it is insecure. 86
5.16 Prefix.pch . . . 86
5.17 Working with app images and icons . . . 87
5.18 why-do-i-need-1x-2x-and-3x-ios-images . . . 87
5.19 Working with Localized strings . . . 87
5.20 instance vs id . . . 87
5.21 video-tutorial-beginning-auto-layout . . . 87
5.22 Setter & getter . . . 87
5.23 Add Linker Flag . . . 89
5.24 how to check if array or dictionary or string is null or empty . . . 89
5.25 Status bar and navigation bar overlaps my view . . . 89
5.26 NULL and nil . . . 89
5.27 To copy paste, choose either copy/paste or option+drag . . . 90
5.28 Snippets . . . 90
5.29 Standford University ios 7 . . . 90
5.30 Blocks . . . 90
5.30.1 The __block Storage Type . . . 92
5.30.2 Blocks As Completion Handlers . . . 93

5.30.3 Typically, blocks are used for one of the following: . . . 94
5.30.4 A Simple Completion Handler Without BLocks . . . 94
5.30.5 A Simple Completion Handler Using Block Approach . . . 95
5.31 See those plugins installed on Mac . . . 96
5.32 how to fold methods xcode . . . 96
5.33 Accessing provisioning profile files . . . 97
5.34 raywenderlich . . . 97
5.35 About App store . . . 97

6 Things learned **97**
6.1 NSMutableParagraphStyle and boundingRectWithSize . . . 97

7 Things about Mac computer **99**
7.1 Mac bookmarks . . . 99
7.2 Command + option + esc to force quit . . . 99
7.3 How to Stop Photos App Launching Automatically in OS X . . . 99
7.4 The top most bar is called menu bar in Mac OSX . . . 100
7.5 how-to-hide-or-remove-icons-from-the-mac-desktop . . . 100
7.6 How to silence the startup chime on a Mac . . . 100
7.7 Use Texshop on Mac . . . 100
7.8 how-to-move-up-a-directory-with-terminal-in-osx . . . 100
7.9 how to change default apps for opening a specific file type os x . . . 100
7.10 Turn off auto updates OS X Yosemite . . . 101
7.11 how-to-fully-reset-safari-on-your-mac . . . 101
7.12 How to use iPhone, iPad earphones . . . 101

1 iOS Objective-C basics

1.1 Interface and Implementation

1. In Objective C, the file where the declaration of class is done is called the **interface file** and the file where the class is defined is called the **implementation file**.
2. A simple interface file **MyClass.h** would look like the following

```
@interface MyClass:NSObject{
// class variable declared here (self: also called instance variable)
}
// class properties declared here (self: also can be called instance variable)
// class methods and instance methods declared here
@end
```

The implementation file **MyClass.m** would be as follows

```
@implementation MyClass{
// instance variable declared here  (from https://www.youtube.com/watch?v=9VbZrdPneHM 2:10
}
// class methods defined here
@end
```

1.2 Object Creation

Object creation is done as follows

```
MyClass  *objectName = [[MyClass alloc]init] ;
```

1.3 Methods

1. Method is declared in Objective C as follows

```
-(returnType)methodName:(typeName) variable1 :(typeName)variable2;
```

An example is shown below.

```
-(void)calculateAreaForRectangleWithLength:(CGfloat)length
andBreadth:(CGfloat)breadth;
```

2. You might be wondering what the **andBreadth** string is for; actually it?s an optional string, which helps us read and understand the method easily, especially at the time of calling.
3. To call this method in the same class, we use the following statement
   `[self calculateAreaForRectangleWithLength:30 andBreadth:20];`
   Self is used to specify that it's a class method.

1.3.1 Class Methods

Class methods can be accessed directly without creating objects for the class. They don't have any variables and objects associated with it. An example is shown below.
`+(void)simpleClassMethod;`
It can be accessed by using the class name (let's assume the class name as MyClass) as follows
`[MyClass simpleClassMethod];`

1.3.2 Instance Methods

Instance methods can be accessed only after creating an object for the class. Memory is allocated to the instance variables. An example instance method is shown below.
`-(void)simpleInstanceMethod;`
It can be accessed after creating an object for the class as follows

```
MyClass  *objectName = [[MyClass alloc]init] ;
[objectName simpleInstanceMethod];
```

1.4 Important Data Types in Objective C

See Picture DataTypes

1.5 Printing Logs

NSLog - used for printing a statement. It will be printed in the device logs and debug console in release and debug modes respectively.

1.6 Properties

For an external class to access the class, variable properties are used. For example,
`@property(nonatomic , strong) NSString *myString;`

1.6.1 Accessing Properties

You can use dot operator to access properties. To access the above property, we will do the following.
`self.myString = @"Test";`
You can also use the set method as follows
`[self setMyString:@"Test"];`

1.7 Categories

Categories are used to add methods to the existing classes.

```
@interface MyClass(customAdditions)
- (void)sampleCategoryMethod;
@end

@implementation MyClass(categoryAdditions)
```

```
-(void)sampleCategoryMethod{
   NSLog(@"Just a test category");
}
```

1.8 Arrays

NSMutableArray and NSArray are the array classes used in objective C. As the name suggests, the former is mutable and the latter is immutable. An example is shown below.

NSMutableArray methods :

1. insertObject: atIndex: (index 0 is at the top)
2. addObject (add something at the bottom or end of the array)

```
NSMutableArray *aMutableArray = [[NSMutableArray alloc]init];
[aMutableArray addObject:[NSMutableArray arrayWithObjects:@"Craft tableview cell",
@"tableview cell dropdown list",@"<< View basic info", nil]];
[aMutableArray insertObject:[NSArray arrayWithObjects:@"WKSelfDiagnosisResultVCDropdownList",
@"UITableView reading from plist file", nil] atIndex:1];
[[aMutableArray objectAtIndex:0] replaceObjectAtIndex:2 withObject:@"<< View basic info"];

[anArray addObject:@"firstobject"];
NSArray *aImmutableArray = [[NSArray alloc]
initWithObjects:@"firstObject",@"secondObject",@"thirdObject",nil];
[aImmutableArray objectAtIndex:0];

self.viewControllers = [NSArray arrayWithObjects:_patientRootVC,_doctorRootVC,_accountRootVC,
nil];
```

1.9 Dictionary

```
NSMutableDictionary*aMutableDictionary = [[NSMutableArray alloc]init];
[aMutableDictionary setObject:@"firstobject" forKey:@"aKey"];
NSDictionary*aImmutableDictionary= [[NSDictionary alloc]initWithObjects:[NSArray arrayWithObjec
@"firstObject",nil] forKeys:[ NSArray arrayWithObjects:@"aKey"]];
```

2 iOS - Actions and Outlets

Actions and outlets in iOS are referred to as **ibActions** and **ibOutlets** respectively, where ib stands for interface builder.

3 iOS - Delegates You can refer to OCTP.tex 2.28

1. Categories don't allow you to add instance variables

Let's assume an object A calls an object B to perform an action. Once the action is complete, object A should know that B has completed the task and take necessary action. This is achieved with the help of delegates. The key concepts in the above example are

- A is a delegate object of B.
- B will have a reference of A.
- A will implement the delegate methods of B.
- B will notify A through the delegate methods.

1. If you created a `Single View Application` you will have AppDelegate and ViewController and `@property (strong, nonatomic) UIWindow *window;`automatically
2. If you created a new class you will have the `@interface className` declaration skeleton in the className.h file and `@implementation className` implementation skeleton in the className.m file

3. new a SampleProtocol class with subclass as NSObject
4. Add a protocol to the SampleProtocol.h file and the updated code is as follows

```
#import <Foundation/Foundation.h>
// Protocol definition starts here
@protocol SampleProtocolDelegate <NSObject>
@required
- (void) processCompleted;
@end
// Protocol Definition ends here
@interface SampleProtocol : NSObject

{
   // Delegate to respond back
   id <SampleProtocolDelegate> _delegate; // Or delegate only

}
@property (nonatomic,strong) id delegate;
//property for delegate
//@property (retain)id <SampleProtocolDelegate> delegate;

-(void)startSampleProcess; // Instance method

@end
```

Passing Data Between View Controllers Using Protocols iOS If not specify, defaults to required See project PassingDataUsingProtocolsYT and project PassingDataUsingProtocolsYTProgrammatically
(when use storyboard:
`[self dismissViewControllerAnimated:YES completion:nil];`
when not:
`[self.navigationController popViewControllerAnimated:YES];`
)

```
@protocol passNames <NSObject>
- (void)setFirstName:(NSString*)firstName;
- (void)setlastName:(NSString*)lastName;
@end

@interface SecondVC : UIViewController
@property (retain)id <passNames> delegate;  // the object type will be unknown at runtime
```

notice the instance variable delegate is of type id, as it will be unknown at compile time the type of class that will adopt this protocol. See project theBasicsOfProtocolsDelegates (passed a BOOL value) (compare with delegates project which didn't pass any value) (both are achieved programatically)

5. Implement the instance method by updating the SampleProtocol.m file as shown below.

```
#import "SampleProtocol.h"

@implementation SampleProtocol

-(void)startSampleProcess{

    [NSTimer scheduledTimerWithTimeInterval:3.0 target:self.delegate
selector:@selector(processCompleted) userInfo:nil repeats:NO]; // Or target:self only
}
@end
```

6. Add a UILabel in the ViewController.xib(mine is **Main.storyboard**),Create an IBOutlet for the label and name it as myLabel and update the code as follows to adopt SampleProtocolDelegate in ViewController.h.

```
#import <UIKit/UIKit.h>
#import "SampleProtocol.h"
```

```
@interface ViewController : UIViewController<SampleProtocolDelegate>
{
    IBOutlet UILabel *myLabel;
}
@end
```

Pay attention to the `#import "SampleProtocol.h"`

7. Implement the delegate method, create object for SampleProtocol and call the startSampleProcess method. The Updated ViewController.m file is as follows

```
#import "ViewController.h"

@interface ViewController () //This is a class extension

@end

@implementation ViewController

- (void)viewDidLoad
{
    [super viewDidLoad];
    SampleProtocol *sampleProtocol = [[SampleProtocol alloc]init];
    sampleProtocol.delegate = self;
    [myLabel setText:@"Processing..."];
    [sampleProtocol startSampleProcess];
// Do any additional setup after loading the view, typically from a nib.
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

#pragma mark - Sample protocol delegate
-(void)processCompleted{
    [myLabel setText:@"Process Completed"];
}

@end
```

Completed code

```
#import <Foundation/Foundation.h>
// Protocol definition starts here
@protocol SampleProtocolDelegate <NSObject>
@required
- (void) processCompleted;
@end
// Protocol Definition ends here
@interface SampleProtocol : NSObject

{
   // Delegate to respond back
   id <SampleProtocolDelegate> _delegate;

}
@property (nonatomic,strong) id delegate;

-(void)startSampleProcess; // Instance method
```

```
@end

#import "SampleProtocol.h"

@implementation SampleProtocol

-(void)startSampleProcess{

    //Protocol method gets called
    [NSTimer scheduledTimerWithTimeInterval:3.0 target:self.delegate
selector:@selector(processCompleted) userInfo:nil repeats:NO];
}
@end

#import <UIKit/UIKit.h>
#import "SampleProtocol.h"

//Class conforms to the protocol
@interface ViewController : UIViewController<SampleProtocolDelegate>
{
    IBOutlet UILabel *myLabel;
}
@end

#import "ViewController.h"

@interface ViewController ()

@end

@implementation ViewController

- (void)viewDidLoad
{
    [super viewDidLoad];
    SampleProtocol *sampleProtocol = [[SampleProtocol alloc]init];
    //sets the delegate to the current class(from TutorialsPoint UIElement textField)
    sampleProtocol.delegate = self;
    [myLabel setText:@"Processing..."];
    [sampleProtocol startSampleProcess];
// Do any additional setup after loading the view, typically from a nib.
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

#pragma mark - Sample protocol delegate
-(void)processCompleted{
    [myLabel setText:@"Process Completed"]; //Protocol method gets implemented
}

@end
```

PassingDataUsingProtocolsYTProgrammatically1 revamped from PassingDataUsingProtocolsYTProgrammatically

Create a SVA, delete Main.storyboard and the corresponding row in info.plist, delete ViewController. Add two classes named FirstVC and SecondVC by subclassing UIViewController respectively. (passing data from SecondVC to FirstVC) See Pic Passing Data 1 and Passing Data 2

AppDelegate.m

```
#import "FirstVC.h"
inside - (Bool)application:didFinishLaunchingWithOptions: method add
self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]];

FirstVC *first = [[FirstVC alloc]init];
UINavigationController *navController = [[UINavigationController alloc]initWithRootViewCon

self.window.rootViewController = navController;

[self.window makeKeyAndVisible];
```

SecondVC.h

```
@protocol passNames <NSObject>

- (void)setFirstName:(NSString *)firstName;
- (void)setlastName:(NSString *)lastName;

@end

@interface SecondVC : UIViewController

//property for delegate
@property (retain)id <passNames> delegate;

/**
 * string passed to us from firstVC
 */
@property (strong, nonatomic) NSString *passedMessage;

@end
```

SecondVC.m

```
@interface SecondVC (){
    UILabel *_messageLabel;
    UITextField *_firstNameText;
    UITextField *_lastNameText;
}

@end

@implementation SecondVC

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view.

    self.title = @"SecondVC";
```

```
    UIBarButtonItem *cancel = [[UIBarButtonItem alloc]initWithTitle:@"Cancel" style:UIBarB
    UIBarButtonItem *save = [[UIBarButtonItem alloc]initWithTitle:@"Save" style:UIBarButto

    [self.navigationItem setLeftBarButtonItem:cancel animated:YES];
    [self.navigationItem setRightBarButtonItem:save animated:YES];

    [self buildViews];

    self.view.backgroundColor = [UIColor colorWithRed:0 green:10 blue:250 alpha:1];
}

- (void)buildViews{
    UILabel *label1 = [[UILabel alloc]initWithFrame:CGRectMake(50, 80, 200, 20)];
    label1.text = @"Sent from firstVC";

    _messageLabel = [[UILabel alloc]initWithFrame:CGRectMake(50, 110, 200, 30)];
    _messageLabel.backgroundColor = [UIColor yellowColor];
    _messageLabel.text = self.passedMessage;

    UILabel *label2 = [[UILabel alloc]initWithFrame:CGRectMake(50, 150, 150, 20)];
    label2.text = @"Send to firstVC";

    _firstNameText = [[UITextField alloc]initWithFrame:CGRectMake(50, 180, 150, 30)];
    _firstNameText.backgroundColor = [UIColor whiteColor];

    _lastNameText = [[UITextField alloc]initWithFrame:CGRectMake(50, 220, 150, 30)];
    _lastNameText.backgroundColor = [UIColor whiteColor];

    [self.view addSubview:label1];
    [self.view addSubview:_messageLabel];
    [self.view addSubview:label2];
    [self.view addSubview:_firstNameText];
    [self.view addSubview:_lastNameText];
}

- (void)cancelAction{
    [self.navigationController popViewControllerAnimated:YES];
}

- (void)saveAction{
    if ([_firstNameText.text isEqualToString:@""]) {
        UIAlertController *aController = [UIAlertController alertControllerWithTitle:@"A S

        UIAlertAction *aAction = [UIAlertAction actionWithTitle:@"OK" style:UIAlertActionS
        [aController addAction:aAction];

        [self presentViewController:aController animated:YES completion:nil];

        return;
    }

    // assign the value of (_firstNameText field) to (firstNameString string)
    [[self delegate]setFirstName:_firstNameText.text];

    [[self delegate]setlastName:_lastNameText.text];
```

```
    [self.navigationController popViewControllerAnimated:YES];
}

#pragma mark -- hideKeyboardAuto
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
    // [self.phoneNumTextField resignFirstResponder];
    [self.view endEditing:YES];
}

@end


FirstVC.h
#import "SecondVC.h"

@interface FirstVC : UIViewController<passNames>

// passed back from secondVC
@property (nonatomic, strong) NSString *firstNameString;
@property (nonatomic, strong) NSString *lastNameString;

@end


FirstVC.m
@interface FirstVC (){
    UILabel *_firstNameLabel;
    UILabel *_lastNameLabel;
    UITextField *_messageBox;
}

@end

@implementation FirstVC

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view.

    self.title = @"FirstVC";

    [self buildViews];

    UIBarButtonItem *add = [[UIBarButtonItem alloc]initWithBarButtonSystemItem:UIBarButton

    [self.navigationItem setRightBarButtonItem:add animated:YES];

    self.view.backgroundColor = [UIColor greenColor];

}

- (void)buildViews{
    UILabel *label1 = [[UILabel alloc]initWithFrame:CGRectMake(50, 80, 200, 20)];
    label1.text = @"Data sent from SecondVC";
```

```
    _firstNameLabel = [[UILabel alloc]initWithFrame:CGRectMake(50, 110, 200, 30)];
    _firstNameLabel.backgroundColor = [UIColor yellowColor];
    //    self.firstNameLabel.text = self.firstNameString;

    _lastNameLabel = [[UILabel alloc]initWithFrame:CGRectMake(50, 150, 200, 30)];
    _lastNameLabel.backgroundColor = [UIColor yellowColor];
    //    self.lastNameLabel.text = self.lastNameString;

    UILabel *label2 = [[UILabel alloc]initWithFrame:CGRectMake(50, 190, 200, 20)];
    label2.text = @"Sending to SecondVC";

    _messageBox.borderStyle = UITextBorderStyleRoundedRect;
    _messageBox = [[UITextField alloc]initWithFrame:CGRectMake(50, 220, 200, 30)];
    _messageBox.backgroundColor = [UIColor whiteColor];
    _messageBox.placeholder = @"Write message";

    [self.view addSubview:label1];
    [self.view addSubview:_firstNameLabel];
    [self.view addSubview:_lastNameLabel];
    [self.view addSubview:label2];
    [self.view addSubview:_messageBox];
}

- (void)viewWillAppear:(BOOL)animated{

    _firstNameLabel.text = self.firstNameString;
    _lastNameLabel.text = self.lastNameString;
}

- (void)pushNewView{
    SecondVC *second = [[SecondVC alloc]init];
    second.delegate = self;
    second.passedMessage = _messageBox.text;
    [self.navigationController pushViewController:second animated:YES];
}

#pragma mark - protocol methods

- (void)setFirstName:(NSString *)firstName{
    self.firstNameString = firstName;

}

- (void)setlastName:(NSString *)lastName{
    self.lastNameString = lastName;

}

#pragma mark -- hideKeyboardAuto
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
    // [self.phoneNumTextField resignFirstResponder];
    [self.view endEditing:YES];
}

@end
```

4 iOS - UI Elements

4.1 UIImage and UIImageView

1. ```
   UIImageView *rightImage = [[UIImageView alloc]initWithImage:[UIImage imageNamed:
   @"common_icon_arrow"]];
   rightImage.size = CGSizeMake(10, 10);
   rightImage.centerY = _imageView.centerY;
   rightImage.centerX = kScreenWidth-rightImage.width-20;
   [view addSubview:rightImage];
   ```
2. **center the image:**
   ```
   UIImage *image = [UIImage imageNamed:@"biaoqing"];
   CGFloat width = image.size.width;
   CGFloat height = image.size.height;

   UIImageView *biaoqing = [[UIImageView alloc]initWithImage:image];
   biaoqing.frame = CGRectMake((kScreenWidth - width)/2 ,
   (self.view.height - height - 60)/2, width, height);
   ```
3. draw a coloured area See Pic goImage
   ```
   UIImageView *_goImage;
   _goImage = [[UIImageView alloc]initWithFrame:CGRectZero];
   _goImage.backgroundColor = [UIColor orangeColor];
   [self.contentView addSubview:_goImage];
   _goImage.frame = CGRectMake(width - 30, 56, 20, 20);
   ```
4. Round the image to a circle See pic RoundedImage
   ```
   _imageView.frame = CGRectMake(x, upperLineMaxY+10, 57, 57);
   _imageView.layer.masksToBounds = YES;
   _imageView.layer.cornerRadius = _imageView.bounds.size.width/2;
   _imageView.layer.borderWidth = 0.5;
   _imageView.layer.borderColor = [WKColorLine CGColor];
   ```

4.2 UISwitch

Create a SVA, make the ViewController class root view controller.

```
ViewController.m
@interface ViewController (){
    UISwitch *_mySwitch;
}
@end

- (void)viewDidLoad {
    _mySwitch = [[UISwitch alloc]initWithFrame:CGRectMake(130, 235, 0, 0)];
    [_mySwitch addTarget:self action:@selector(changeSwitch) forControlEvents:UIControlEventVal
    [self.view addSubview:_mySwitch];
}

- (void)changeSwitch{
    if ([_mySwitch isOn]) {
        NSLog(@"Switch is ON");
    }else{
        NSLog(@"Switch is OFF");
    }
```
```

```
}

- (void)changeSwitch:(id)sender{
    if ([sender isOn]) {
        NSLog(@"Switch is ON");
    }else{
        NSLog(@"Switch is OFF");
    }
}
```

4.3 iOS - Text Field

Important Properties of Text Field

- Placeholder text which is shown when there is no user input
- Normal text
- Auto correction type
- Key board type
- Return key type
- Clear button mode
- Alignment myTextField.textAlignment = NSTextAlignmentCenter;
- Delegate

1. The default height of a UITextField is 31 You can't not set it's height less than 24 or this textfield will have a very different effect when its text is fullfilled
2. uitextfield-with-secure-entry-always-getting-cleared-before-editing
   Solution: By using UITextFieldDelegate. First set that textField's delegate to current viewController(self), and implement the following method. Also you have to set that textField's secureTextEntry to YES.
   ```
   - (BOOL)textField:(UITextField *)textField shouldChangeCharactersInRange:(NSRange)range
   replacementString:(NSString *)string {
       if ([string isEqualToString: @"\n"])
           return NO;
       NSString *updatedString = [textField.text stringByReplacingCharactersInRange:range
       withString:string];
       textField.text = updatedString;

       return NO;
   }
   ```
3. TextFieldPadding protocol
   ```
   UITextField+TextFieldPadding.h
   @interface UITextField (TextFieldPadding)

   /**
    *  设置textfield padding
    *
    *  @param paddingValue padding值
    */
   -(void) setLeftPadding:(int) paddingValue;

   -(void) setRightPadding:(int) paddingValue;

   @end


   UITextField+TextFieldPadding.m
   @implementation UITextField (TextFieldPadding)

   -(void) setLeftPadding:(int) paddingValue
   ```

```
{
    UIView *paddingView = [[UIView alloc] initWithFrame:CGRectMake(0, 0, paddingValue, sel
    self.leftView = paddingView;
    self.leftViewMode = UITextFieldViewModeAlways;
}

-(void) setRightPadding:(int) paddingValue
{
    UIView *paddingView = [[UIView alloc] initWithFrame:CGRectMake(0, 0, paddingValue, sel
    self.rightView = paddingView;
    self.rightViewMode = UITextFieldViewModeAlways;
}

@end
```

creating-text-fields-programmatically first create a SVA then inside ViewController.m file add the following code. See Pic clearButtonMode returnKeyType autocapitalizationType

UITextField's text is a string not merely UITextField

```
//first one
UITextField *tf=[[UITextField alloc]initWithFrame:CGRectMake(85,30,200,40)];
tf.textColor=[UIColor colorWithRed:0/256.0 green:84/256 blue:129/256 alpha:1.0];
tf.secureTextEntry = YES;
tf.borderStyle = UITextBorderStyleRoundedRect;
tf.keyboardType = UIKeyboardTypeNumberPad;
tf.clearButtonMode = UITextFieldViewModeWhileEditing;
tf.returnKeyType = UIReturnKeyDone;
passwordTextField.autocapitalizationType = UITextAutocapitalizationTypeWords;
tf.contentVerticalAlignment = UIControlContentVerticalAlignmentCenter;
tf.textAlignment = NSTextAlignmentLeft;
tf.font=[UIFont fontWithName:@"Helvetica-Bold" size:25];
// tf.font = [UIFont systemFontOfSize:14.0f];
tf.backgroundColor=[UIColor grayColor];
//tf.placeholder = @"Hello World";
tf.text=@"hello world";
textField.attributedPlaceholder = [[NSAttributedString alloc]initWithString:holder attributes:
@{NSForegroundColorAttributeName:WKColorGrayLight,NSFontAttributeName:WKFontNor_15}];

/*
textfield.leftView = _padding;
textfield.leftViewMode = UITextFieldViewModeAlways;
*/ // doesn't work very well

UIView *view=[[UIView alloc]initWithFrame:CGRectMake(100, 200, 400, 400)];
[view addSubview:tf];

[self.view addSubview:view];


[myTextField setPlaceholder:@"Pick a Sport"];
```

UITextFieldDelegate See project TextFieldARC See Pic UITextFieldDelegate Always remember to set the delegate to self after you adopt the this delegate protocol
when you first click the first textField following delegate methods are called

```
- (BOOL)textFieldShouldBeginEditing:
- (void)textFieldDidBeginEditing:
```

when you go to next textField following delegate methods are called sequentially

```
- (BOOL)textFieldShouldBeginEditing:
- (BOOL)textFieldShouldEndEditing:
- (void)textFieldDidEndEditing:
- (void)textFieldDidBeginEditing:
```

To restrict certain characters from a text field(see the picture), as in following code disallow the '#' symbol

```
- (BOOL)textField:(UITextField *)textField shouldChangeCharactersInRange:(NSRange)range
replacementString:(NSString *)string{
NSLog(@"textField:shouldChangeCharactersInRange:replacementString:");
     if ([string isEqualToString:@"#"]) {
         return NO;
     }else{
         return YES;
     }
}
```

resigning the keyboard and moving to the next text field
`textFieldShouldReturn:` is called when the user presses the return key on the keyboard.(in the project the return key is **Done**)

```
- (BOOL)textFieldShouldReturn:(UITextField *)textField{
     NSLog(@"textFieldShouldReturn:");
     if (textField.tag == 1) {
         UITextField *passwordTextField = (UITextField *)[self.view viewWithTag:2];
         [passwordTextField becomeFirstResponder];
     }else{
         [textField resignFirstResponder];
     }
     return YES;
}
```

4.4 iOS - UITextView

```
UITextView *myTextView;
myTextView = [[UITextView alloc]initWithFrame:CGRectMake(10, 50, 300, 200)];
[myTextView setText:@"something"];
[self.view addSubview:myTextView];
```

how-to-set-margins-padding-in-uitextview

```
myTextView.textContainerInset = UIEdgeInsetsMake(5, 5, 0, 0);
```

Wrap line in uitextview: See Pic wrapLine

UITextView placeholder:

```
AppDelegate.m
Make the ViewController root view controller and conforms to UITextViewDelegate

ViewController.m
@implementation ViewController{
    UITextField *_content;
}

- (void)viewDidLoad {
    [super viewDidLoad];
```

```
    self.view.backgroundColor = [UIColor grayColor];

    UIView *bgview = [[UIView alloc]initWithFrame:CGRectMake(10, 120, 320-20, 100)];
    bgview.backgroundColor = [UIColor whiteColor];
    bgview.layer.masksToBounds = YES;
    bgview.layer.cornerRadius = 5;
    bgview.layer.borderWidth = 0.5;
    bgview.layer.borderColor = [[UIColor grayColor]CGColor];
    [self.view addSubview:bgview];

    UITextView *myUITextView = [[UITextView alloc] initWithFrame:CGRectMake(10, 0, bgview.frame
    myUITextView.delegate = self;
    myUITextView.text = @"???60???????";
    myUITextView.textColor = [UIColor lightGrayColor];
    [bgview addSubview:myUITextView];

}

- (void)textViewDidBeginEditing:(UITextView *)textView
{
    if ([textView.text isEqualToString:@"???60???????"]) {
        textView.text = @"";
        textView.textColor = [UIColor blackColor]; //optional
    }
    [textView becomeFirstResponder];
}

- (void)textViewDidEndEditing:(UITextView *)textView
{
    if ([textView.text isEqualToString:@""]) {
        textView.text = @"???60???????";
        textView.textColor = [UIColor lightGrayColor]; //optional
    }
    [textView resignFirstResponder];
}
@end
```

4.5 iOS - Label

```
UILabel *titleLabel = [[UILabel alloc]initWithFrame:CGRectMake(0, 0, 100, 20)];
titleLabel.backgroundColor = [UIColor yellowColor];
titleLabel.text = @"Page 1";

[self.view addSubview:titleLabel];


CGRect labelFrame = CGRectMake(20, 20, 280, 150);
UILabel *myLabel = [[UILabel alloc] initWithFrame:labelFrame];
[myLabel setBackgroundColor:[UIColor orangeColor]];
NSString *labelText = @"I am the very model of a modern Major-General, I've information vegetab
[myLabel setText:labelText];
myLabel.font = [UIFont systemFontOfSize:15];
myLabel.layer.borderColor = [UIColor grayColor].CGColor;
myLabel.layer.borderWidth = 0.5;
myLabel.layer.cornerRadius = 5.0f;
myLabel.clipsToBounds = YES;  //These two must combined to make cornerRadius work on iOS 7.1
```

```
// and later Or myLabel.layer.masksToBounds = YES;
// Tell the label to use an unlimited number of lines
[myLabel setNumberOfLines:0];
myLabel.lineBreakMode = NSLineBreakByWordWrapping;
[myLabel sizeToFit];  // Sometimes can substitute vertical align(label has no vertical align)
```

1. Align two labels horizontally
   ```
   _treatment_dateLbl.frame = CGRectMake(x, lowerLineMaxY+10, kScreenWidth, 13);

   _feeLbl.frame = CGRectMake(0, 0, kScreenWidth-10, 13);
   _feeLbl.centerY = _treatment_dateLbl.centerY;  // will override the y value of the upper
   // line
   ```
2. By default label's text is left aligned and vertically centered, you can set the text horizontally alignment by using the label's textAlignment property. There's no way to set the vertical align on a UILabel, but you can get the same effect by changing the label's frame.
3. When a label's bounds change See 5.12, you can change its text vertical alignment using following method
   `contentLabel.contentMode = UIViewContentModeTop;`

4.6 iOS - Toolbar

See Pic Toolbar

4.7 iOS - Buttons

1. Image and text
   `titleEdgeInset`
   `contentHorizontalAlignment`

Creates and returns a new button of the specified type.

```
class property (inside ViewController.h file):
UIButton *navButton; // SO have to be enclosed in curly braces

then inside ViewController.m file
navButton = [UIButton buttonWithType:UIButtonTypeRoundedRect]; // Class method
Or navButton.layer.cornerRadius=2.0;
[navButton setFrame:CGRectMake(60, 50, 200, 40)]; // Instance method
[navButton setBackgroundColor:[UIColor blueColor]];
[eyeBtn setImage:[UIImage imageNamed:@"icon-sign-eye"] forState:UIControlStateNormal];
[navButton setTitle:@"Push Navigation" forState:UIControlStateNormal];
[navButton setTitleColor:[UIColor blackColor] forState:UIControlStateNormal];
navButton.titleLabel.font=[UIFont systemFontOfSize:20];
navButton.contentHorizontalAlignment = UIControlContentHorizontalAlignmentLeft;
navButton.layer.masksToBounds = YES;
navButton.layer.cornerRadius = 5.0;
[navButton addTarget:self action:@selector(pushNewView:) forControlEvents:
UIControlEventTouchUpInside];
[navButton setHidden:YES];  // Set button hidden


UIButton *getAuth1=[[UIButton alloc]init];
getAuthCodebutton1=getAuth1;
CGFloat width1=120;
getAuth1.frame = CGRectMake(120, 280, width1, 30);
[self.view addSubview:getAuth1];
```

This button will not have the click effect. It seems that if you don't specify a button this way :
`UIButton *PwdButton = [UIButton buttonWithType:UIButtonTypeRoundedRect];`
then it won't have the click effect

4.8 iOS - Status Bar

1. Status bar flick effect (temporary solution)

```
- (void)loadView{
    [super loadView];
    self.navigationController.navigationBar.translucent = YES;
}

- (void)viewWillAppear:(BOOL)animated
{
    [super viewWillAppear:animated];
    self.navigationController.navigationBar.translucent = NO;
}

- (void)viewDidAppear:(BOOL)animated{
    [super viewDidAppear:animated];
    self.navigationController.navigationBar.clipsToBounds = YES;
}

- (void) viewWillDisappear:(BOOL)animated
{
    [super viewWillDisappear:animated];
    self.navigationController.navigationBar.clipsToBounds = NO;
}
```

Permanent Solution: add following line to both view controller

```
[self.navigationController.navigationBar setShadowImage:[[UIImage alloc]init]];
[self.navigationController.navigationBar setBackgroundImage:[UIImage imageNamed:@"topbg"]
forBarMetrics:UIBarMetricsDefault];
```

setStatusBarHidden: In the controller in which you want to hide the status bar, call

```
- (BOOL)prefersStatusBarHidden{
    return YES;
}
```

preferredStatusBarStyle: In the controller in which you want to style the status bar, call

```
- (UIStatusBarStyle)preferredStatusBarStyle{
    return UIStatusBarStyleLightContent;
}
```

Not applied to iOS 9, in iOS 9 there are two ways to do this inside `ViewController.m`

1. `[self setNeedsStatusBarAppearanceUpdate];`
   `Or: [self prefersStatusBarHidden];  // These will be inside - (void)viewDidLoad`

```
- (BOOL)prefersStatusBarHidden{
    return YES;
}
```

2. `[self preferredStatusBarStyle];`

```
- (UIStatusBarStyle)preferredStatusBarStyle{
    return UIStatusBarStyleLightContent; // Another is default
}
```

4.9 iOS - Navigation Bar

1. some methods of navigation bar

```
self.automaticallyAdjustsScrollViewInsets = NO;
[self.navigationController.navigationBar setShadowImage:[[UIImage alloc]init]];
[self.navigationController.navigationBar setBackgroundImage:[self imageWithColor:
[[UIColor whiteColor] colorWithAlphaComponent:1]] forBarMetrics:UIBarMetricsDefault];

- (UIImage *)imageWithColor:(UIColor *)color
{
    // 描述矩形
    CGRect rect = CGRectMake(0.0f, 0.0f, 1.0f, 1.0f);

    // 开启位图上下文
    UIGraphicsBeginImageContext(rect.size);
    // 获取位图上下文
    CGContextRef context = UIGraphicsGetCurrentContext();
    // 使用color演示填充上下文
    CGContextSetFillColorWithColor(context, [color CGColor]);
    // 渲染上下文
    CGContextFillRect(context, rect);
    // 从上下文中获取图片
    UIImage *theImage = UIGraphicsGetImageFromCurrentImageContext();
    // 结束上下文
    UIGraphicsEndImageContext();

    return theImage;
}
```

2. navigationBar bottom line disappear

```
- (void)viewWillAppear:(BOOL)animated
{
    [super viewWillAppear:animated];
    self.navigationController.navigationBar.clipsToBounds = YES;
}
- (void) viewWillDisappear:(BOOL)animated
{
    [super viewWillDisappear:animated];
    self.navigationController.navigationBar.clipsToBounds = NO;
}
```

3. navigationItem has a titleView `super.navigationItem.titleView = titleLabel;`
4. setNavigationBarHidden: In the controller in which you want to hide the navigation bar, call

```
- (void) viewWillAppear:(BOOL)animated {
    [super viewWillAppear:animated];
    [self.navigationController setNavigationBarHidden:YES animated:animated];
}

- (void)viewWillDisappear:(BOOL)animated{
    [super viewWillDisappear:animated];
    [self.navigationController setNavigationBarHidden:NO animated:YES];
}
```

Create a view based app and create a TemplViewController class by subclassing UIViewController, following is a simple example (complex example is denoted with /* */ comment —use another button to make the transition) See Pic NavigationBarModified See project NavigationBarModified

1. Use tintColor to tint bar button items; use barTintColor to tint the bar background. See project UITabBarControllerBlogspot. (Combine tabbar controller with navigation controller)

Note:
Inside TempViewController.m and ViewController.m you can set its background color by:
self.view.backgroundColor = [UIColor someColor];

AppDelegate.h

```
#import "ViewController.h" //Or @class ViewController; better the first one

@interface AppDelegate : UIResponder <UIApplicationDelegate>

@property (strong, nonatomic) UIWindow *window;

@property (strong, nonatomic) ViewController *viewController;

@property (strong, nonatomic) UINavigationController *navController;

@end
```

AppDelegate.m

```
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)
    // Override point for customization after application launch.
    self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]];
    self.viewController = [[ViewController alloc]init];

    UINavigationController *navController = [[UINavigationController alloc]
    initWithRootViewController:self.viewController];

    self.window.rootViewController = navController;

    [self.window makeKeyAndVisible];

    return YES;
}
```

ViewController.h

```
#import "TempViewController.h"

@interface ViewController : UIViewController{
    UIButton *navButton;
}

- (void)addNavigationBarButton;

- (IBAction)pushNewView:(id)sender;
/* - (IBAction)myButtonClicked:(id)sender; */

@end
```

ViewController.m
inside - (void)viewDidLoad method:

```
[self addNavigationBarButton];

- (void)addNavigationBarButton{
```

```
    UIBarButtonItem *myNavBtn = [[UIBarButtonItem alloc]initWithTitle:@"MyButton"
    style:UIBarButtonItemStylePlain target:self action:@selector(pushNewView:)];
    /* action:@selector(myButtonClicked:) */

    [self.navigationController.navigationBar setBarStyle:UIBarStyleBlack];
    [self.navigationItem setRightBarButtonItem:myNavBtn animated:YES];

    /* // create a navigation push button that is initially hidden
    navButton = [UIButton buttonWithType:UIButtonTypeRoundedRect];
    [navButton setFrame:CGRectMake(60, 50, 200, 40)];
    [navButton setTitle:@"Push Navigation" forState:UIControlStateNormal];
    [navButton addTarget:self action:@selector(pushNewView:) forControlEvents:
    UIControlEventTouchUpInside];

    [self.view addSubview:navButton];
    [navButton setHidden:YES];  */
}

- (IBAction)pushNewView:(id)sender{
    TempViewController *tempVC = [[TempViewController alloc]init];
    [self.navigationController pushViewController:tempVC animated:YES];
}

/* - (IBAction)myButtonClicked:(id)sender{
    // toggle hidden state for navButton
    [navButton setHidden:!navButton.hidden];
} */
```

4.10 iOS - Tab Bar

1. It's generally used to switch between various subtasks, views or models within the same view.
2. Important Properties
   - backgroundImage
   - items
   - selectedItem
3. Tab bar controller is the root view controller not navigation controller when these two combined

Create a SVA, add three classes named RedViewController GreenViewController BlueViewController respectively and set these three classes background color to red green blue See Pic tabBar1 tabBar2 tabBar3

AppDelegate.m

```
#import "RedViewController.h"
#import "GreenViewController.h"
#import "BlueViewController.h"

inside - (Bool)application:didFinishLaunchingWithOptions: method add
self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]];

// Override point for customization after application launch.

UITabBarController *tabBars = [[UITabBarController alloc]init];
NSMutableArray *array = [[NSMutableArray alloc]init];

RedViewController *rVC = [[RedViewController alloc]init];
GreenViewController *gVC = [[GreenViewController alloc]init];
BlueViewController *bVC = [[BlueViewController alloc]init];

rVC.tabBarItem = [[UITabBarItem alloc]initWithTitle:@"Home" image:[UIImage imageNamed:@"tabBar_
```

```
// rVC.tabBarItem = [[UITabBarItem alloc]initWithTabBarSystemItem:UITabBarSystemItemBookmarks t

gVC.tabBarItem.title = @"Green";
// gVC.tabBarItem.badgeValue = @"some";
bVC.tabBarItem.title = @"Blue";

[array addObject:rVC];
[array addObject:gVC];
[array addObject:bVC];

tabBars.viewControllers = array;
tabBars.view.autoresizingMask=(UIViewAutoresizingFlexibleHeight);
self.window.rootViewController = tabBars;

[self.window makeKeyAndVisible];
```


1. UITabBarItem is a subclass of UIBarItem, so in addition to its title and image it inherits the ability to adjust the image position with imageInsets, plus the enabled and tag properties.

4.11 UIPickerView

Create a view based app and in Main.storyboard add pickerview to the design area then ctrl+drag the pickerview to the class name the outlet picker. Also you have to make the class conforms to `UIPickerViewDelegate` and `UIPickerViewDataSource` protocols.

```
ViewController.h
#import <UIKit/UIKit.h>

@interface ViewController : UIViewController<UIPickerViewDelegate,UIPickerViewDataSource>{
    NSArray *pickerData;
}

@end


ViewController.m
#import "ViewController.h"

@interface ViewController ()

@property (weak, nonatomic) IBOutlet UIPickerView *picker;


@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    pickerData = @[@[@"Item 1",@"Item2",@"Item 3",@"Item 4",@"Item5"],
    @[@"1",@"2",@"3",@"4",@"5"],
    @[@"a",@"b",@"c",@"d",@"e"],
                   @[@"w",@"x",@"y",@"z",@"s"],
                   @[@"",@"",@"",@"",@""]]; // I don't know why it only displays four columns,
```

```
                    // when not adding this line it only displays three columns, you also have
                    // to change numberOfcomponentsInPickerView method

    // Connect data
    self.picker.dataSource = self;
    self.picker.delegate = self;
}

- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

// The number of columns of data
- (int)numberOfComponentsInPickerView:(UIPickerView *)pickerView{
    return 5;
}

// The number of rows of data
- (int)pickerView:(UIPickerView *)pickerView numberOfRowsInComponent:(NSInteger)component{
    return pickerData.count;
}

// The data to return for the row and component (column) that's being passed in
- (NSString *)pickerView:(UIPickerView *)pickerView titleForRow:(NSInteger)row forComponent:(NS
    return pickerData[component][row];
}

- (void)pickerView:(UIPickerView *)pickerView didSelectRow:(NSInteger)row inComponent:(NSIntege
    // This method is triggered whenever the user makes a change to the picker selection
    // The parameter named row and component represents what was selected
}
@end
```

PickersTP project has similar function to Adjust the view while showing keyboard

4.12 UISlider

Using storyboard: Create a view based app, in Main.storyboard drag a label and slider and create **outlet for both of them named label and slider** (inside .h file not inside inspector) respectively and **last** you have to drag the slider to the ViewController.m file and create an IBAction for it:

```
ViewController.h
@interface ViewController : UIViewController
//@property (weak, nonatomic) IBOutlet UILabel *label;

@property (weak, nonatomic) IBOutlet UILabel *label;

@property (weak, nonatomic) IBOutlet UISlider *slider;

@end


ViewController.m
- (IBAction)sliderValueChanged:(id)sender {
    // Set the label text to the value of the slider as it changes
    self.label.text = [NSString stringWithFormat:@"%f",self.slider.value];
```

```
}
```

Programmatically:

```
mySlider = [[UISlider alloc]initWithFrame:CGRectMake(50, 100, 200, 30)];
[self.view addSubview:mySlider];
mySlider.maximumValue = 99.0;
mySlider.minimumValue = 10.0;
mySlider.continuous = NO;
[mySlider addTarget:self action:@selector(sliderChanged:) forControlEvents:
UIControlEventValueChanged];
```

4.13 UIAlertView

UIAlertControllerStyleActionSheet can use array to include each action

```
UIAlertController *alertController = [UIAlertController alertControllerWithTitle:@"请联系客服01
在线反馈" message:nil preferredStyle:UIAlertControllerStyleActionSheet];
[alertController.view setTintColor:[UIColor orangeColor]];

// create the actions
UIAlertAction *cancelAction = [UIAlertAction actionWithTitle:@"" style:UIAlertActionStyleCancel
    DLog(@"alert's cancel action occured");
}];
UIAlertAction *feedbackAction = [UIAlertAction actionWithTitle:@"我要反馈" style:UIAlertActionS
    DLog(@"我要反馈");
}];
UIAlertAction *callAction = [UIAlertAction actionWithTitle:@"拨打电话" style:UIAlertActionStyle
    DLog(@"拨打电话");
}];
UIAlertAction *action = [UIAlertAction actionWithTitle:@"取消" style:UIAlertActionStyleDestruct
    DLog(@"取消");
}];

// add the actions
[alertController addAction:cancelAction];
[alertController addAction:feedbackAction];
[alertController addAction:callAction];
[alertController addAction:action];

[self presentViewController:alertController animated:YES completion:nil];
```

First create a button —addTarget:action:forControlEvents and change the action part for each situation:

```
UIButton *button = [UIButton buttonWithType:UIButtonTypeRoundedRect];
button.frame = CGRectMake(60, 120, 120, 20);
[button setTitle:@"AlertController" forState:UIControlStateNormal];
[button setTitleColor:[UIColor blackColor] forState:UIControlStateNormal];
[button addTarget:self action:@selector(showSimpleAlert) forControlEvents:UIControlEventTouchUp

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

See Pic choosePay for following code

```
UIAlertView *alert = [[UIAlertView alloc]initWithTitle:@"???????" message:nil delegate:self
cancelButtonTitle:@"??" otherButtonTitles:@"????", @"?????", nil];
alert.tag = 1; // tag值不能设为0，要不然会崩
[alert show];
```

See Pic OkayAlert for following code

```
- (void)showSimpleAlert {
    NSString *title = NSLocalizedString(@"A Short Title Is Best", nil);
    NSString *message = NSLocalizedString(@"A message should be a short, complete sentence.", n
    NSString *cancelButtonTitle = NSLocalizedString(@"OK", nil);

    UIAlertController *alertController = [UIAlertController alertControllerWithTitle:title mess

    // Create the action.
    UIAlertAction *cancelAction = [UIAlertAction actionWithTitle:cancelButtonTitle style:UIAler
        NSLog(@"The simple alert's cancel action occured.");
    }];

    // Add the action.
    [alertController addAction:cancelAction];

    [self presentViewController:alertController animated:YES completion:nil];
}
```

See Pic OkayCancelAlert for following code

```
// Show an alert with an "Okay" and "Cancel" button.
- (void)showOkayCancelAlert {
    NSString *title = NSLocalizedString(@"A Short Title Is Best", nil);
    NSString *message = NSLocalizedString(@"A message should be a short, complete sentence.", n
    NSString *cancelButtonTitle = NSLocalizedString(@"Cancel", nil);
    NSString *otherButtonTitle = NSLocalizedString(@"OK", nil);

    UIAlertController *alertController = [UIAlertController alertControllerWithTitle:title mess

    // Create the actions.
    UIAlertAction *cancelAction = [UIAlertAction actionWithTitle:cancelButtonTitle style:UIAler
        NSLog(@"The \"Okay/Cancel\" alert's cancel action occured.");
    }];

    UIAlertAction *otherAction = [UIAlertAction actionWithTitle:otherButtonTitle style:UIAlertA
        NSLog(@"The \"Okay/Cancel\" alert's other action occured.");
    }];

    // Add the actions.
    [alertController addAction:cancelAction];
    [alertController addAction:otherAction];

    [self presentViewController:alertController animated:YES completion:nil];
}
```

See Pic TwoCustomButtonAlert for following code

```
// Show an alert with two custom buttons.
- (void)showOtherAlert {
    NSString *title = NSLocalizedString(@"A Short Title Is Best", nil);
    NSString *message = NSLocalizedString(@"A message should be a short, complete sentence.", n
    NSString *cancelButtonTitle = NSLocalizedString(@"Cancel", nil);
    NSString *otherButtonTitleOne = NSLocalizedString(@"Choice One", nil);
    NSString *otherButtonTitleTwo = NSLocalizedString(@"Choice Two", nil);

    UIAlertController *alertController = [UIAlertController alertControllerWithTitle:title mess
```

```
    // Create the actions.
    UIAlertAction *cancelAction = [UIAlertAction actionWithTitle:cancelButtonTitle style:UIAler
        NSLog(@"The \"Other\" alert's cancel action occured.");
    }];

    UIAlertAction *otherButtonOneAction = [UIAlertAction actionWithTitle:otherButtonTitleOne st
        NSLog(@"The \"Other\" alert's other button one action occured.");
    }];

    UIAlertAction *otherButtonTwoAction = [UIAlertAction actionWithTitle:otherButtonTitleTwo st
        NSLog(@"The \"Other\" alert's other button two action occured.");
    }];

    // Add the actions.
    [alertController addAction:cancelAction];
    [alertController addAction:otherButtonOneAction];
    [alertController addAction:otherButtonTwoAction];

    [self presentViewController:alertController animated:YES completion:nil];
}
```

See Pic TextEntryAlert for following code

```
// Show a text entry alert with two custom buttons.
- (void)showTextEntryAlert {
    NSString *title = NSLocalizedString(@"A Short Title Is Best", nil);
    NSString *message = NSLocalizedString(@"A message should be a short, complete sentence.", n
    NSString *cancelButtonTitle = NSLocalizedString(@"Cancel", nil);
    NSString *otherButtonTitle = NSLocalizedString(@"OK", nil);

    UIAlertController *alertController = [UIAlertController alertControllerWithTitle:title mess

    // Add the text field for text entry.
    [alertController addTextFieldWithConfigurationHandler:^(UITextField *textField) {
        textField.placeholder = @"-------------";
        // If you need to customize the text field, you can do so here.
    }];

    // Create the actions.
    UIAlertAction *cancelAction = [UIAlertAction actionWithTitle:cancelButtonTitle style:UIAler
        NSLog(@"The \"Text Entry\" alert's cancel action occured.");
    }];

    UIAlertAction *otherAction = [UIAlertAction actionWithTitle:otherButtonTitle style:UIAlertA
        NSLog(@"The \"Text Entry\" alert's other action occured.");
    }];

    // Add the actions.
    [alertController addAction:cancelAction];
    [alertController addAction:otherAction];

    [self presentViewController:alertController animated:YES completion:nil];
}
```

See Pic SecureTextEntryAlert for following code

```
@property (nonatomic, weak) UIAlertAction *secureTextAlertAction;
```

```
// Show a secure text entry alert with two custom buttons.
- (void)showSecureTextEntryAlert {
    NSString *title = NSLocalizedString(@"A Short Title Is Best", nil);
    NSString *message = NSLocalizedString(@"A message should be a short, complete sentence.", n
    NSString *cancelButtonTitle = NSLocalizedString(@"Cancel", nil);
    NSString *otherButtonTitle = NSLocalizedString(@"OK", nil);

    UIAlertController *alertController = [UIAlertController alertControllerWithTitle:title mess

    // Add the text field for the secure text entry.
    [alertController addTextFieldWithConfigurationHandler:^(UITextField *textField) {
        /*
         Listen for changes to the text field's text so that we can toggle the current
         action's enabled property based on whether the user has entered a sufficiently
         secure entry.
         */
        [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(handleTextFie

        textField.secureTextEntry = YES;
    }];

    // Create the actions.
    UIAlertAction *cancelAction = [UIAlertAction actionWithTitle:cancelButtonTitle style:UIAler
        NSLog(@"The \"Secure Text Entry\" alert's cancel action occured.");

        // Stop listening for text changed notifications.
        [[NSNotificationCenter defaultCenter] removeObserver:self name:UITextFieldTextDidChange
    }];

    UIAlertAction *otherAction = [UIAlertAction actionWithTitle:otherButtonTitle style:UIAlertA
        NSLog(@"The \"Secure Text Entry\" alert's other action occured.");

        // Stop listening for text changed notifications.
        [[NSNotificationCenter defaultCenter] removeObserver:self name:UITextFieldTextDidChange
    }];

    // The text field initially has no text in the text field, so we'll disable it.
    otherAction.enabled = NO;

    // Hold onto the secure text alert action to toggle the enabled/disabled state when the tex
    self.secureTextAlertAction = otherAction;

    // Add the actions.
    [alertController addAction:cancelAction];
    [alertController addAction:otherAction];

    [self presentViewController:alertController animated:YES completion:nil];
}

#pragma mark - UITextFieldTextDidChangeNotification

- (void)handleTextFieldTextDidChangeNotification:(NSNotification *)notification {
    UITextField *textField = notification.object;
```

```
    // Enforce a minimum length of >= 5 characters for secure text alerts.
    self.secureTextAlertAction.enabled = textField.text.length >= 5;
}
```

See project AlertDemo for both programmatically (status quo) and storyboard on alertViewDelegate method. import three images See Pic AlertDemo

```
AppDelegate.m
#import "ViewController.h"
inside - (Bool)application:didFinishLaunchingWithOptions: method add
self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]];

ViewController *vController = [[ViewController alloc]init];
self.window.backgroundColor = [UIColor whiteColor];

self.window.rootViewController = vController;

[self.window makeKeyAndVisible];


ViewController.m
In ViewController.h adopt <UIAlertViewDelegate>
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.
    UIImage *image = [UIImage imageNamed:@"background"];
    UIImageView *imageView = [[UIImageView alloc]initWithImage:image];
    imageView.frame = CGRectMake(0, 20, self.view.frame.size.width, self.view.frame.size.height

    UIButton *button = [UIButton buttonWithType:UIButtonTypeCustom];
    UIImage *buttonImage = [UIImage imageNamed:@"button"];
    CGFloat width = buttonImage.size.width;
    CGFloat height = buttonImage.size.height;
    button.frame = CGRectMake(60, 160, width, height);
    [button setImage:[UIImage imageNamed:@"button"] forState:UIControlStateNormal];
    [button setImage:[UIImage imageNamed:@"button-pressed"] forState:UIControlStateNormal];
    [button setBackgroundImage:[UIImage imageNamed:@"button"] forState:UIControlStateNormal];
    [button addTarget:self action:@selector(touched) forControlEvents:UIControlEventTouchUpInsi

    [self.view addSubview:imageView];
    [self.view addSubview:button];   //here button will show on top of imageView
}

- (void)touched{
    UIAlertView *message = [[UIAlertView alloc]initWithTitle:@"Hello World!" message:@"This is

    [message addButtonWithTitle:@"Button 2"];
    [message addButtonWithTitle:@"Button 3"];
    [message addButtonWithTitle:@"something added"];

    [message show];
}

- (void)alertView:(UIAlertView *)alertView clickedButtonAtIndex:(NSInteger)buttonIndex{
    NSString *title = [alertView buttonTitleAtIndex:buttonIndex];
    if ([title isEqualToString:@"Button 1"]) {
        NSLog(@"Button 1 was selected");
```

```
    }else if ([title isEqualToString:@"Button 2"]){
        NSLog(@"Button 2 was selected");
    }else if ([title isEqualToString:@"Button 3"]){
        NSLog(@"Button 3 was selected");
    }
}
```

Create a SVA, make the ViewController class root view controller(See 4.41).Create a new UIAlertView category named NSCookbook. Xcode will enforce the proper naming convention creating both `UIAlertView+NSCookbook.h` and `UIAlertView+NSCookbook.m`. Create a .strings file named Localizable See project alertViewBlock, See Pic alertBlockGoTo

```
Localizable.strings
NSCBSampleAlertTitle = "Sample Alert";
NSCBSampleAlertMessage = "This alert uses a block callback instead implementing the super
annoying delegate protocol that would normally be required just to handle a simple button
click in an alert such as this.";

NSCBSampleAlertCancel = "Cancel";
NSCBSampleAlertOk = "Ok";

NSCBSampleAlertCallbackFormat = "You selected %@";


UIAlertView+NSCookbook.h
@interface UIAlertView (NSCookbook)<UIAlertViewDelegate>

- (void)showWithCompletion:(void(^)(UIAlertView *alertView, NSInteger buttonIndex))completion;

@end


UIAlertView+NSCookbook.m
#import <objc/runtime.h>
#import "UIAlertView+NSCookbook.h"

@interface NSCBAlertWrapper : NSObject

@property (copy) void(^completionBlock)(UIAlertView *alertView, NSInteger buttonIndex);

@end

@implementation NSCBAlertWrapper

#pragma mark - UIAlertViewDelegate

// Called when a button is clicked. The view will be automatically dismissed after this call
// returns
- (void)alertView:(UIAlertView *)alertView clickedButtonAtIndex:(NSInteger)buttonIndex{
    if (self.completionBlock)
        self.completionBlock(alertView, buttonIndex);
}

// Called when we cancel a view (eg. the user clicks the Home button). This is not called when
// the user clicks the cancel button.
// If not defined in the delegate, we simulate a click in the cancel button
- (void)alertViewCancel:(UIAlertView *)alertView
```

```
{
    // Just simulate a cancel button click
    if (self.completionBlock) {
        self.completionBlock(alertView, alertView.cancelButtonIndex);
    }
}

@end

static const char kNSCBAlertWrapper;

@implementation UIAlertView (NSCookbook)

#pragma mark - Class Public

- (void)showWithCompletion:(void (^)(UIAlertView *, NSInteger))completion{
    NSCBAlertWrapper *alertWrapper = [[NSCBAlertWrapper alloc]init];
    alertWrapper.completionBlock = completion;
    self.delegate = alertWrapper;

    // Set the wrapper as an associated object
    objc_setAssociatedObject(self, &kNSCBAlertWrapper, alertWrapper,
    OBJC_ASSOCIATION_RETAIN_NONATOMIC);

    // Show the alert as normal
    [self show];
}

@end


ViewController.m
#import "UIAlertView+NSCookbook.h"
@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.
    UIButton *button = [UIButton buttonWithType:UIButtonTypeCustom];
    button.frame = CGRectMake(80, 200, 120, 40);
    button.backgroundColor = [UIColor orangeColor];
    [button setTitle:@"show alert" forState:UIControlStateNormal];
    [button addTarget:self action:@selector(showAlert) forControlEvents:
    UIControlEventTouchUpInside];

    [self.view addSubview:button];

}

- (void)showAlert{
    UIAlertView *alertView = [[UIAlertView alloc]initWithTitle:NSLocalizedString
    (@"NSCBSampleAlertTitle", @"")
                    message:NSLocalizedString(@"NSCBSampleAlertMessage", @"")
                    delegate:nil
            cancelButtonTitle:NSLocalizedString(@"NSCBSampleAlertCancel", @"")
            otherButtonTitles:NSLocalizedString(@"NSCBSampleAlertOk", @""), nil];
```

```
    [alertView showWithCompletion:^(UIAlertView *alertView, NSInteger buttonIndex) {
        UIAlertView *responseAlert = [[UIAlertView alloc]initWithTitle:NSLocalizedString
        (@"NSCBSampleAlertTitle", @"")
                        message:[NSString stringWithFormat:NSLocalizedString
                        (@"NSCBSampleAlertCallbackFormat", @""), buttonIndex ?
                        NSLocalizedString(@"NSCBSampleAlertOk", @""):
                        NSLocalizedString(@"NSCBSampleAlertCancel", @"")]
                        delegate:nil
              cancelButtonTitle:@"Ok"
              otherButtonTitles:nil, nil];
        [responseAlert showWithCompletion:NULL];
    }];
}
```

4.14 UIActivityIndicator

Using storyboard: Create a view based app, in Main.storyboard drag two **activity indicator view** and create outlet for both of them named grayStyleActivityIndicatorView and tintedAvtivityIndicatorView respectively inside **ViewController.m** file

```
ViewController.m
@interface ViewController ()
@property (weak, nonatomic) IBOutlet UIActivityIndicatorView *grayStyleActivityIndicatorView;
@property (weak, nonatomic) IBOutlet UIActivityIndicatorView *tintedAvtivityIndicatorView;

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    [self configureGrayActivityIndicatorView];
    [self configureTintedActivityIndicatorView];
}

#pragma mark - Configuration

- (void)configureGrayActivityIndicatorView {
    self.grayStyleActivityIndicatorView.activityIndicatorViewStyle = UIActivityIndicatorViewSty

    [self.grayStyleActivityIndicatorView startAnimating];

    self.grayStyleActivityIndicatorView.hidesWhenStopped = YES;
}

- (void)configureTintedActivityIndicatorView {
    self.tintedActivityIndicatorView.activityIndicatorViewStyle = UIActivityIndicatorViewStyleG

    self.tintedActivityIndicatorView.color = [UIColor redColor];

    [self.tintedActivityIndicatorView startAnimating];
}

- (void)didReceiveMemoryWarning {
```

```
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

@end
```

Programmatically

```
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    self.view.backgroundColor = [UIColor orangeColor];

    UIActivityIndicatorView *activityView = [[UIActivityIndicatorView alloc]initWithActivityInd
    Or:
    activityIndicator.center = CGPointMake(self.view.frame.size.width / 2.0, self.view.frame.si

    activityView.center=self.view.center;
    [activityView startAnimating];
    [self.view addSubview:activityView];
}
```

4.15 UIPageControl

Using storyboard: Create a view based app, in Main.storyboard drag **uiview View** and **Page Control** create outlet for both of them named colorView and pageControl respectively inside ViewController.m file

```
ViewController.m
@interface ViewController ()

@property (weak, nonatomic) IBOutlet UIPageControl *pageControl;

@property (weak, nonatomic) IBOutlet UIView *colorView;

@property (nonatomic, strong) NSArray *colors;

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    // Set a list of colors that correspond to the selected page.
    self.colors = @[[UIColor blackColor], [UIColor grayColor], [UIColor redColor], [UIColor gre

    [self configurePageControl];
    [self pageControlValueDidChange];
}

#pragma mark - Configuration

- (void)configurePageControl {
    // The total number of pages that are available is based on how many available colors we ha
    self.pageControl.numberOfPages = [self.colors count];
```

```
    self.pageControl.currentPage = 2;

    self.pageControl.tintColor = [UIColor redColor];
    self.pageControl.pageIndicatorTintColor = [UIColor blackColor];
    self.pageControl.currentPageIndicatorTintColor = [UIColor greenColor];

    [self.pageControl addTarget:self action:@selector(pageControlValueDidChange) forControlEver
}

#pragma mark - Actions

- (void)pageControlValueDidChange {
    NSLog(@"The page control changed its current page to %ld.", (long)self.pageControl.currentP

    self.colorView.backgroundColor = self.colors[self.pageControl.currentPage];
}
```

Programmatically (just for reference)

```
ViewController.m
@interface ViewController ()

@property (nonatomic, weak) UIPageControl *pageControl;

@property (nonatomic, weak) UIView *colorView;

@property (nonatomic,strong) NSArray *colors;

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    self.colors = @[[UIColor blackColor], [UIColor grayColor], [UIColor redColor], [UIColor gre

    UIView *colorView = [[UIView alloc]initWithFrame:CGRectMake(50, 200, 120, 100)];
    [colorView setBackgroundColor:[UIColor blackColor]];

    UIPageControl *pageControl = [[UIPageControl alloc]initWithFrame:CGRectMake(50, 400, 250, 2
    pageControl.numberOfPages = [self.colors count];
    pageControl.currentPage = 4;
    pageControl.tintColor = [UIColor redColor];
    pageControl.pageIndicatorTintColor = [UIColor blackColor];
    pageControl.currentPageIndicatorTintColor = [UIColor greenColor];

    [pageControl addTarget:self action:@selector(pageControlValueDidChange) forControlEvents:UI

    [self.view addSubview:colorView];
    [self.view addSubview:pageControl];

}

- (void)pageControlValueDidChange {
    NSLog(@"The page control changed its current page to %ld.", (long)self.pageControl.currentP
```

```
//    [self.colorView setBackgroundColor:self.colors[self.pageControl.currentPage];
    self.colorView.backgroundColor = self.colors[self.pageControl.currentPage];
}
```

4.16 iOS - Table View

It is used for displaying a vertically scrollable view which consists of a number of cells (generally reusable cells). It has special features like headers, footers, rows, and section.
UITabelView is a subclass of UIScrollView

1. Things about indexPath
   ```
   // current indexPath
   NSIndexPath *indexPath = [self.tableView indexPathForSelectedRow];
   // make a new indexPath and add 1 to the row of the previous one
   NSIndexPath *indexPath2 = [[NSIndexPath alloc]init];
   indexPath2 = [NSIndexPath indexPathForItem:(indexPath.row + 1) inSection:indexPath.section
   ```

2. tableView frequently used attributes and methods
   ```
   tableView.backgroundColor = WKColorVCBg;
   tableView.tableHeaderView = [[UIView alloc]initWithFrame:CGRectMake(0, 0, kScreenWidth,
   10)];
   self.tableView.sectionHeaderHeight  can be 0 value
   tableView.separatorStyle = UITableViewCellSeparatorStyleNone;
   tableView.separatorColor = WKColorLine;
   tableView.bounces = NO;
   self.tableView.tableHeaderView


   #pragma mark - UITableViewDelegate
   - (CGFloat)tableView:(UITableView *)tableView heightForHeaderInSection:(NSInteger)section{
   - (UIView *)tableView:(UITableView *)tableView viewForHeaderInSection:(NSInteger)section {
   These two must exist at the same time

   - (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:(NSIndexPath *)index
   - (CGFloat)tableView:(UITableView *)tableView heightForFooterInSection:(NSInteger)section{

   #pragma mark - UITableViewDataSource
   - (NSString *)tableView:(UITableView *)tableView titleForHeaderInSection:(NSInteger)sectio
   - (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section{}
   - (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView{}
   - (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath
   ```

3. In WaitingInfoCell.m, you can implement the following method
   ```
   - (void)layoutSubviews{
       [super layoutSubviews];
   CGRect bounds = self.contentView.bounds;
   CGFloat width = bounds.size.width;
   }
   ```
   Get the contentView's bounds. You can set UIElements frame here, in other place CGRectZero. Like in a UIViewController subclass, you can set UIElements frame in `- (void)viewWillLayoutSubviews` method and in other place CGRectZero

4. how-to-remove-empty-cells-in-uitableview
   Set a zero height table footer view, like so:
   ```
   tableView.tableFooterView = [[UIView alloc] initWithFrame:CGRectZero];
   ```
   Because the table thinks there is a footer to show, it doesn't display any cells beyond those you explicitly

asked for.

5. A table view with static cells is static, which means that the number of rows and sections is defined at compile time, not runtime. However, it does not mean that the cell's contents cannot be modified at runtime.
6. Adding a title to the table view

```
- (void)loadView
{
    CGRect titleRect = CGRectMake(0, 0, 300, 40);
    UILabel *tableTitle = [[UILabel alloc] initWithFrame:titleRect];
    tableTitle.textColor = [UIColor blueColor];
    tableTitle.backgroundColor = [self.tableView backgroundColor];
    tableTitle.opaque = YES;
    tableTitle.font = [UIFont boldSystemFontOfSize:18];
    tableTitle.text = [curTrail objectForKey:@"Name"];
    self.tableView.tableHeaderView = tableTitle;
    [self.tableView reloadData];
}
```

7. Add a switch to table cell

```
        health = [[UISwitch alloc]init];
        cell.textLabel.text = @"something on the cell";
        cell.accessoryView = health;
//        [cell setAccessoryView:hotline]; //This will work too
//        [cell.accessoryView addSubview:health];  //This won't work(didn't show),
// don't know why
//        [cell.contentView addSubview:health];  // This will work
```

8. how-to-set-the-full-width-of-separator-in-uitableview
   Just add this code on your TableViewController

```
// Set the full width of separator
-(void)tableView:(UITableView *)tableView willDisplayCell:(UITableViewCell *)cell forRowAt
{
    if ([cell respondsToSelector:@selector(setSeparatorInset:)]) {
        [cell setSeparatorInset:UIEdgeInsetsZero];
    }

    if ([cell respondsToSelector:@selector(setLayoutMargins:)]) {
        [cell setLayoutMargins:UIEdgeInsetsZero];
    }
}

-(void)viewDidLayoutSubviews
{
    if ([self.tableView respondsToSelector:@selector(setSeparatorInset:)]) {
        [self.tableView setSeparatorInset:UIEdgeInsetsZero];
    }

    if ([self.tableView respondsToSelector:@selector(setLayoutMargins:)]) {
        [self.tableView setLayoutMargins:UIEdgeInsetsZero];
    }
}
```

9. ios-7-changing-font-size-for-uitableview-section-headers Have some problems in implementing this method (cause section header overlaps cell text)
   Unfortunately, you may have to override this:

```
- (UIView *)tableView:(UITableView *)tableView viewForHeaderInSection:(NSInteger)section
```

   Add these codes to your TableViewController

```
- (UIView *)tableView:(UITableView *)tableView viewForHeaderInSection:(NSInteger)section {
    UILabel *myLabel = [[UILabel alloc] init];
```

```
    myLabel.frame = CGRectMake(15, (40-16)/2, 320, 40);
    myLabel.font = [UIFont systemFontOfSize:16];
    myLabel.text = [self tableView:tableView titleForHeaderInSection:section];
    [myLabel sizeToFit];
    UIView *headerView = [[UIView alloc] init];
    [headerView addSubview:myLabel];

    return headerView;
}
```

10. Specify `cell.selectionStyle = UITableViewCellSelectionStyleNone;` just before **return** in `- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath` method will dismiss the flick effect and make the cell not selectable (in other place may cause the flick effect)
11. Make specific rows not selectable

```
- (NSIndexPath *)tableView:(UITableView *)tableView willSelectRowAtIndexPath:(NSIndexPath
    // rows in section 0 should not be selectable
    if ( indexPath.section == 0 ) return nil;

    // first 3 rows in any section should not be selectable
//    if ( indexPath.row <= 2 ) return nil;

    // By default, allow row to be selected
    return indexPath;
}
```

12. In the network request method, you have to add `[self.tableView reloadData];` or the following method won't work

```
- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
{
return self.dataSource.count;
}
```

13. You can

```
NSString *dataToShow = [[self.tableData objectAtIndex:[indexPath section]] objectAtIndex:
[indexPath row]];
```

But can not

```
NSString *dataToShow = [[self.tableData objectAtIndex:[indexPath.section]] objectAtIndex:
[indexPath.row]];
```

This is OK

```
NSString *dataToShow = [[self.tableData objectAtIndex:[indexPath.section]] objectAtIndex:
indexPath.row];
```

14. See pic sectionsTestGoTo and ToSectionsTest for following code how-to-show-and-hide-toggle-uitableview

```
@interface WKSelfTestSecondVC ()<UITableViewDataSource,UITableViewDelegate>{
    BOOL showAllSections;
}

@property (nonatomic, strong) UITableView *tableView;
@property (nonatomic, strong) NSMutableArray *tableData;
@property (nonatomic, strong) NSMutableArray *subtitleData;

@end

@implementation WKSelfTestSecondVC
- (NSMutableArray *)tableData{
    if (_tableData == nil) {
        _tableData = [[NSMutableArray alloc]init];
    }
```

```
    return _tableData;
}

- (NSMutableArray *)subtitleData{
    if (_subtitleData == nil) {
        _subtitleData = [[NSMutableArray alloc]init];
    }
    return _subtitleData;
}

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view.

    self.title = @"UITableSectionsTestApp";

    showAllSections = NO;
    [self setUpTableView];

    [self.tableData addObject:[NSMutableArray arrayWithObjects:@"John",@"Somebody",@"View
    [self.tableData addObject:[NSMutableArray arrayWithObjects:@"email1@example.com",@"ano

    [self.subtitleData addObject:[NSMutableArray arrayWithObjects:@"First name",@"Last nam
    [self.subtitleData addObject:[NSMutableArray arrayWithObjects:@"E-mail address #1",@"E
}

- (void)setUpTableView{
    UITableView *tableView = [[UITableView alloc]initWithFrame:CGRectMake(0, 0, kScreenWid
    tableView.dataSource = self;
    tableView.delegate = self;
    tableView.backgroundColor = WKColorVCBg;
    self.tableView = tableView;
    [self.view addSubview:tableView];
}

#pragma mark - UITableViewDataSource
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView{
    if (showAllSections) {
        return 4;
    }else{
        return 2;
    }
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section{
    if (section == 0) {
        return 3;
    }else{
        return 2;
    }
}

- (NSString *)tableView:(UITableView *)tableView titleForHeaderInSection:(NSInteger)sectio
    NSString *headerTitle = @"";

    if (section == 0) {
```

```
        headerTitle = @"Basic info";
    }else{
        if (showAllSections) {
            switch (section) {
                case 1:
                    headerTitle = @"Phone numbers";
                    break;
                case 2:
                    headerTitle = @"Addresses";
                    break;
                case 3:
                    headerTitle = @"E-mail addresses";
                default:
                    break;
            }
        }else{
            headerTitle = @"E-mail addresses";
        }
    }

    return headerTitle;
}

- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath
    static NSString *CellIdentifier = @"Cell";

    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
    if (!cell) {
        cell = [[UITableViewCell alloc]initWithStyle:UITableViewCellStyleSubtitle reuseIde
    }

    cell.selectionStyle = UITableViewCellSelectionStyleNone;

    NSString *dataToShow = [[self.tableData objectAtIndex:indexPath.section] objectAtIndex
    NSString *detailText = [[self.subtitleData objectAtIndex:[indexPath section]] objectAt

    cell.textLabel.text = dataToShow;
    cell.textLabel.textColor = [UIColor blackColor];
    cell.detailTextLabel.text = detailText;

    if (indexPath.section == 0 && indexPath.row == 2) {
        cell.textLabel.textColor = [UIColor redColor];
        cell.selectionStyle = UITableViewCellSelectionStyleBlue;
    }

    return cell;
}

#pragma mark - UITableViewDelegate
- (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:(NSIndexPath *)index
    return 60;
}

- (void)tableView:(UITableView *)tableView didSelectRowAtIndexPath:(NSIndexPath *)indexPat
    if (indexPath.section == 0 && indexPath.row == 2) {
        if (!showAllSections) {
```

```
            [self.tableData insertObject:[NSArray arrayWithObjects:@"1234567890",@"0987654
            [self.subtitleData insertObject:[NSArray arrayWithObjects:@"Phone number #1",@

            [self.tableData insertObject:[NSArray arrayWithObjects:@"Somewhere out there S
            [self.subtitleData insertObject:[NSArray arrayWithObjects:@"Home address",@"Po

            [[self.tableData objectAtIndex:0] replaceObjectAtIndex:2 withObject:@"<< View
        }else{
            [self.tableData removeObjectAtIndex:2];
            [self.tableData removeObjectAtIndex:1];
            [self.subtitleData removeObjectAtIndex:2];
            [self.subtitleData removeObjectAtIndex:1];

            [[self.tableData objectAtIndex:0] replaceObjectAtIndex:2 withObject:@"View ext
        }

        showAllSections = !showAllSections;
        [self.tableView reloadData];
    }
}

@end
```

15. UITableViewCell dropdown list

```
@interface WKSelfDiagnosisResultVC ()<UITableViewDataSource,UITableViewDelegate>{
    NSString *dataForSection0;
    NSString *dataForSection2;
    NSMutableArray *demoData;
    int selectedValueIndex;
    bool isShowingList;
}

@property (nonatomic, strong) UITableView *tableView;

@end

@implementation WKSelfDiagnosisResultVC

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view.

    self.title = @"自测结果2";
    [self setUpTableView];

    // For the beginning, let's give some values to the strings we declared.
    // Those values will be displayed in the first and the third sections.
    dataForSection0 = @"This is some cell content";
    dataForSection2 = @"This is another cell content";

    // Let's preprare our actual test data.
    demoData = [[NSMutableArray alloc]init];
    // We'll give five values, from one to five.
    [demoData addObject:@"one"];
    [demoData addObject:@"two"];
    [demoData addObject:@"three"];
    [demoData addObject:@"four"];
```

```
    [demoData addObject:@"five"];

    // Initially, the isShowingList value will be set to NO.
    // We don't want the list to be dislplayed when the view loads.
    isShowingList = NO;

    // By default, when the view loads, the first value of the five we created
    // above will be set as selected.
    // We'll do that by pointing to the first index of the array.
    // Don't forget that for the five items of the array, the indexes are from
    // zero to four (0 - 4).
    selectedValueIndex = 0;
}

- (void)setUpTableView{
    UITableView *tableView = [[UITableView alloc]initWithFrame:CGRectMake(0, 0, kScreenWid
    tableView.dataSource = self;
    tableView.delegate = self;
    tableView.backgroundColor = WKColorVCBg;
    self.tableView = tableView;
    [self.view addSubview:tableView];
}

#pragma mark - UITableViewDataSource
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView{
    return 3;
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section{
    if (section == 0 || section == 2) {
        return 1;
    }else{
        // For the number of rows of the section with index 1 (our test section) there
        // are two cases.
        //
        // First case: If the isShowingList variable is set to NO, then no list
        // with values should be displayed (the values of the demoData array) and
        // it should exist only one row.
        //
        // Second case: If the isShowingList variable is set to YES, then the
        // demoData array values should be displayed as a list and the returned
        // number of rows should match the number of the items in the array.
        if (!isShowingList) {
            return 1;
        }else{
            return [demoData count];
        }
    }
}

- (NSString *)tableView:(UITableView *)tableView titleForHeaderInSection:(NSInteger)sectio
    if (section == 0) {
        return @"My first section";
    }else if (section == 1){
        return @"My demo section";
    }else{
```

```
        return @"Another section";
    }
}

#pragma mark - UITableViewDelegate
- (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:(NSIndexPath *)index
    return 60.0;
}

// Customize the appearance of table view cells.
- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath
    static NSString *CellIdentifier = @"Cell";

    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
    if (cell == nil) {
        cell = [[UITableViewCell alloc]initWithStyle:UITableViewCellStyleDefault reuseIden
    }

    // configure the cell
    cell.selectionStyle = UITableViewCellSelectionStyleBlue;
    cell.accessoryType = UITableViewCellAccessoryNone;

    // Let's set another font for the cells.
    [[cell textLabel]setFont:[UIFont fontWithName:@"Marker Felt" size:16.0]];

    // Depending on the section, the appropriate data will be displayed.
    // For the demo section especially, the data display requires a different
    // handling.
    if (indexPath.section == 0) {
        [[cell textLabel]setText:dataForSection0];
    }else if (indexPath.section == 2){
        [[cell textLabel]setText:dataForSection2];
    }else{
        // Depending on the isShowingList variable value, we'll display either
        // the selected value of the demoData array only, or the whole array's
        // contents.
        // Remember that if the isShowingList is set to NO, then only a single row
        // is displayed, containing the selected value.
        // If the isShowingList is set to YES, then a list of values is displayed
        // and all the items of the demoData array should be used.
        if (!isShowingList) {
            // Not a list in this case.
            // We'll only display the item of the demoData array of which array
            // index matches the selectedValueList.
            [[cell textLabel]setText:[demoData objectAtIndex:selectedValueIndex]];

            // We'll also display the disclosure indicator to prompt user to
            // tap on that cell.
            cell.accessoryType = UITableViewCellAccessoryDisclosureIndicator;
        }else{
            // Listing the array items
            [[cell textLabel]setText:[demoData objectAtIndex:indexPath.row]];

            if ([indexPath row] == selectedValueIndex) {
                cell.accessoryType = UITableViewCellAccessoryCheckmark;
            }else{
```

```
                cell.accessoryType = UITableViewCellAccessoryNone;
            }
        }
    }

    return cell;
}

- (void)tableView:(UITableView *)tableView didSelectRowAtIndexPath:(NSIndexPath *)indexPat
    if ([indexPath section] == 1) {
        // The job we have to do here is pretty easy.
        // 1. If the isShowingList variable is set to YES, then we save the
        //     index of the row that the user tapped on (save it to the selectedValueIndex
        // 2. Change the value of the isShowingList variable.
        // 3. Reload not the whole table but only the section we're working on.
        //
        // Note that only that last two steps are necessary when the isShowingList
        // variable is set to NO.

        // Step 1
        if (isShowingList) {
            selectedValueIndex = [indexPath row];
        }

        // Step 2
        isShowingList = !isShowingList;

        [tableView reloadSections:[NSIndexSet indexSetWithIndex:1] withRowAnimation:UITabl
    }else{
        return;
    }

}

@end
```

Using storyboard: Create a SVA (view based), in Main.storyboard drag **Table View** then **Set delegate and dataSource to file owner for tableview by ctrl-clicking and selecting datasource and delegate**. Create an outlet for tableview name it myTableView. Last in ViewController.h adopt the **UITableViewDataSource** and **UITableViewDelegate** protocols. See Pic tableView and fileOwner

```
ViewController.h
@interface ViewController : UIViewController<UITableViewDataSource,UITableViewDelegate>

@property (weak, nonatomic) IBOutlet UITableView *myTableView;

@end



ViewController.m
@interface ViewController ()

@property (strong, nonatomic) NSMutableArray *myData;

@end

@implementation ViewController
```

```
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    self.myData = [[NSMutableArray alloc]initWithObjects:@"Data 1 in array",@"Data 2 in array",
}

- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

// Default is 1 if not implemented
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView{
    return 2;
}

#pragma mark - Table View Data source
- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section{
    return [self.myData count]/2;
}

- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)in
    static NSString *cellIdentifier = @"cellID";

    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:cellIdentifier];
    if(cell == nil){
        cell = [[UITableViewCell alloc]initWithStyle:UITableViewCellStyleDefault reuseIdentifie
    }
    NSString *stringForCell;
    if(indexPath.section == 0){
        stringForCell = [self.myData objectAtIndex:indexPath.row];
    }else if(indexPath.section == 1){
        stringForCell = [self.myData objectAtIndex:indexPath.row+[self.myData count]/2];
    }
    [cell.textLabel setText:stringForCell];
    return cell;
}

- (NSString *)tableView:(UITableView *)tableView titleForHeaderInSection:(NSInteger)section{
    NSString *headerTitle;
    if(section==0){
        headerTitle = @"Section 1 Header";
    }
    else{
        headerTitle = @"Section 2 Header";
    }

    return headerTitle;
}

- (NSString *)tableView:(UITableView *)tableView titleForFooterInSection:(NSInteger)section{
    NSString *footerTitle;
    if(section==0){
        footerTitle = @"Section 1 Footer";
```

```
    }
    else{
        footerTitle = @"Section 2 Footer";
    }

    return footerTitle;
}

#pragma mark - TableView delegate

- (void)tableView:(UITableView *)tableView didDeselectRowAtIndexPath:(NSIndexPath *)indexPath{
    [tableView deselectRowAtIndexPath:indexPath animated:YES];
    UITableViewCell *cell = [tableView cellForRowAtIndexPath:indexPath];
    NSLog(@"Section:%ld Row:%ld selected and its data is %@",(long)indexPath.section,(long)inde
}
@end
```

Pay close attention to these. Their same value will cause different height. The same goes for footer See Pic sectionHeight

```
self.tableView.sectionHeaderHeight  can be 0 value
- (CGFloat)tableView:(UITableView *)tableView heightForHeaderInSection:(NSInteger)section
can't be 0 value
```

Programmatically The code indentation i did myself
Create a SVA, make ViewController adopt the `UITableViewDataSource` protocol See Pic tableView1

```
AppDelegate.m
#import "ViewController.h"
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)
    self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]];
    // Override point for customization after application launch.

    ViewController *vController = [[ViewController alloc]init];

    self.window.rootViewController = vController;

    [self.window makeKeyAndVisible];

    return YES;
}


ViewController.m
@implementation ViewController
{
    NSMutableArray *_dataSource;
}

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    [self prepareData];

    UITableView *tView = [[UITableView alloc]initWithFrame:CGRectMake(0, 0, 320, 480) style:UI
    tView.dataSource = self;
```

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

}

- (void)prepareData
{
    _dataSource = [[NSMutableArray alloc]init];
    for(NSInteger i = 0; i<20; i++){
        NSString *info = [NSString stringWithFormat:@"Student %ld",(long)i];
        [_dataSource addObject:info];
    }
}

- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
{
    return 1;
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
{
    return _dataSource.count;
}

- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)in
{
    static NSString *cellId = @"cellId";

    NSLog(@"cellForRow: %ld",(long)indexPath.row);

    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:cellId];
    if(!cell){
        NSLog(@"alloc: %ld",(long)indexPath.row);
        cell = [[UITableViewCell alloc]initWithStyle:UITableViewCellStyleDefault reuseIdentifi
    }

    cell.textLabel.text = _dataSource[indexPath.row];
    return cell;
}
@end
```

Create a SVA, and add Person.h and Person.m file, make the ViewController adopt UITableViewDataSource and UITableViewDelegate add QQImages to project. Always remember add `#import "ViewController.h"` to AppDelegate.m file See Pic tableViewEdit

AppDelegate.m

```
#import "ViewController.h"
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)
    self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]];
    // Override point for customization after application launch.

    ViewController *vController = [[ViewController alloc]init];
    UINavigationController *navController = [[UINavigationController alloc]initWithRootViewCon

    self.window.backgroundColor = [UIColor whiteColor];

    self.window.rootViewController = navController;
```

```
    [self.window makeKeyAndVisible];

    return YES;
}
```

Person.h

```
#import <Foundation/Foundation.h>

@interface Person : NSObject

@property (nonatomic, copy)NSString *name;//??
@property (nonatomic, assign)NSInteger age;//??
@property (nonatomic, copy)NSString *iconName;//???

@end
```

Person.m

```
#import "Person.h"

@implementation Person

@end
```

ViewController.m

```
#import "Person.h"
@implementation ViewController
{
    NSMutableArray *_persons;
    UITableView *tableView;
}

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    [self prepareData];
    [self customUI];
    [self customNavigationBar];

}

- (void)customUI
{
    self.automaticallyAdjustsScrollViewInsets = NO;
    tableView = [[UITableView alloc]initWithFrame:CGRectMake(0, 64, 320, 416) style:UITableVie
    tableView.dataSource = self;
    tableView.delegate = self;
    [self.view addSubview:tableView];
}

- (void)customNavigationBar
{
    UIBarButtonItem *btnEdit = [[UIBarButtonItem alloc]initWithTitle:@"Edit" style:UIBarButton
```

```
    self.navigationItem.leftBarButtonItem = btnEdit;

    UIBarButtonItem *btnReload = [[UIBarButtonItem alloc]initWithTitle:@"Reload" style:UIBarBu
    self.navigationItem.rightBarButtonItem = btnReload;
}

- (void)onBtnEditClick:(UIBarButtonItem *)sender
{
    if([sender.title isEqualToString:@"Edit"]){
        sender.title = @"Done";

        [tableView setEditing:YES animated:YES];
    }else{
        sender.title = @"Edit";

        [tableView setEditing:NO animated:YES];
    }
}

- (void)onBtnReloadClick
{
    [tableView reloadData];
}

- (void)prepareData
{
    _persons = [[NSMutableArray alloc]init];
    NSMutableArray *classmates = [[NSMutableArray alloc]init];
    NSMutableArray *colleagues = [[NSMutableArray alloc]init];

    [_persons addObject:classmates];
    [_persons addObject:colleagues];

    for(NSInteger i=0; i<=20; i++){
        Person *person = [[Person alloc]init];
        if(i%2){
            person.name = [NSString stringWithFormat:@"Classmates%d",i/2];
            person.iconName = [NSString stringWithFormat:@"p_%d",i];
            person.age = arc4random()%5 + 20;
            [classmates addObject:person];
        }else{
            person.name = [NSString stringWithFormat:@"Colleagues:%d",i/2];
            person.iconName = [NSString stringWithFormat:@"p_%d",i];
            person.age = arc4random()%30 + 20;
            [colleagues addObject:person];
        }
    }
}

- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
{
    return _persons.count;
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
{
```

```
    NSArray *array = _persons[section];
    return array.count;
}

- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)in
{
    NSLog(@"cellForRowAtIndexPath section:%d, row:%d",indexPath.section,indexPath.row);
    static NSString *cellId = @"CellId";
    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:cellId];
    if (!cell) {
        cell = [[UITableViewCell alloc]initWithStyle:UITableViewCellStyleSubtitle reuseIdentif
    }

    Person *person = _persons[indexPath.section][indexPath.row];
    cell.textLabel.text = person.name;
    cell.detailTextLabel.text = [NSString stringWithFormat:@"%d",person.age];
    cell.imageView.image = [UIImage imageNamed:person.iconName];
    return cell;
}

//- (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:(nonnull NSIndexPath *)
//{
// return 50;
//}

- (NSString *)tableView:(UITableView *)tableView titleForHeaderInSection:(NSInteger)section
{
    if(section == 0){
        return @"Classmates";
    }
    return @"Colleagues";
}

- (CGFloat)tableView:(UITableView *)tableView heightForHeaderInSection:(NSInteger)section
{
    return 40;
}

- (void)tableView:(UITableView *)tableView didSelectRowAtIndexPath:(nonnull NSIndexPath *)index
{
    NSLog(@"array %d select section: %d row: %d",tableView.numberOfSections, indexPath.section
}

- (void)tableView:(UITableView *)tableView didDeselectRowAtIndexPath:(nonnull NSIndexPath *)ind
{
    NSLog(@"deselect section: %d row: %d",indexPath.section,indexPath.row);
}

- (UITableViewCellEditingStyle)tableView:(UITableView *)tableView editingStyleForRowAtIndexPath
{
    if(indexPath.section == 0){
        return UITableViewCellEditingStyleDelete;
    }
    return UITableViewCellEditingStyleInsert;
}
```

```
- (void)tableView:(UITableView *)tableView commitEditingStyle:(UITableViewCellEditingStyle)edit
{
if (editingStyle == UITableViewCellEditingStyleDelete) {
NSMutableArray *array = _persons[indexPath.section];
[array removeObjectAtIndex:indexPath.row];
#if 0

[tableView reloadData];
#else

NSArray *indexPaths = [NSArray arrayWithObject:indexPath];
[tableView deleteRowsAtIndexPaths:indexPaths withRowAnimation:UITableViewRowAnimationTop];
#endif
}else{
NSMutableArray *array = _persons[indexPath.section];
Person *person = [[Person alloc]init];
person.name = @"New Colleagues";
person.age = 30;
person.iconName = @"p_0";
[array insertObject:person atIndex:indexPath.row];

#if 0
[tableView reloadData];
#else
NSArray *indexPaths = [NSArray arrayWithObject:indexPath];

[tableView insertRowsAtIndexPaths:indexPaths withRowAnimation:UITableViewRowAnimationAutomatic]
#endif
}
}

- (NSString *)tableView:(UITableView *)tableView titleForDeleteConfirmationButtonForRowAtIndexP
{
    return @"Delete";
}

- (BOOL)tableView:(UITableView *)tableView canMoveRowAtIndexPath:(NSIndexPath *)indexPath
{
    if (indexPath.row) {
        return YES;
    }
    return NO;
}

- (void)tableView:(UITableView *)tableView moveRowAtIndexPath:(NSIndexPath *)sourceIndexPath to
{
    NSMutableArray *sourceArray = _persons[sourceIndexPath.section];

    NSMutableArray *destArray = _persons[destinationIndexPath.section];
    Person *person = sourceArray[sourceIndexPath.row];

    [sourceArray removeObjectAtIndex:sourceIndexPath.row];

    [destArray insertObject:person atIndex:destinationIndexPath.row];
}
```

```
- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

@end
```

UITableViewMultiLineDeleteBaoguo See Pic MultiLineDelete

CustomizingTableViewCellsYoutube: See project CustomizingTableViewCellsYoutubeStoryboard and CustomizingTableViewCellsYoutube and CustomizingTableViewCellsYoutube2 See Pic CustomizingTableViewCells Subclassing went wrong before,should subclass UIViewController, shouldn't subclass UITableView but UIViewController (even though the video drags a tableview to the canvas's view but it's still on top of a ViewController There is no difference between initialize a view in viewDidLoad method and initialize a view in a separate method then call it in viewDidLoad

4.17 UIScrollView

Programmatically horizontally
Create a SVA, delete ViewController .h .m file and create four class(ctrl + n) name them MainViewController, RedViewController, GreenViewController, BlueViewController respectively(subclassing UIViewController) See Pic UIScrollViewHorizontally

```
AppDelegate.m
Make the mainViewController root view controller

#import "MainViewController.h"
inside - (Bool)application:didFinishLaunchingWithOptions: method add
self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]];

// Override point for customization after application launch.

MainViewController *mainViewController = [[MainViewController alloc]init];

self.window.rootViewController = mainViewController;

[self.window makeKeyAndVisible];

return YES;


MainViewController.m
#import "RedViewController.h"
#import "GreenViewController.h"
#import "BlueViewController.h"

@interface MainViewController ()

@property (strong, nonatomic) UIScrollView *scrollView;

@end

@implementation MainViewController

- (void)viewDidLoad {
        [super viewDidLoad];
        // Do any additional setup after loading the view.
```

```
    self.scrollView = [[UIScrollView alloc]initWithFrame:CGRectMake(0, 0, self.view.frame.size

    RedViewController *redVC = [[RedViewController alloc] init];
    GreenViewController *greenVC = [[GreenViewController alloc] init];
    BlueViewController *blueVC = [[BlueViewController alloc] init];

    self.scrollView.contentSize = CGSizeMake(self.scrollView.frame.size.width * 3, self.scroll
    redVC.view.frame = CGRectMake(0, 0, self.scrollView.frame.size.width, self.scrollView.fram
    greenVC.view.frame = CGRectMake(self.scrollView.frame.size.width, 0, self.scrollView.frame
    blueVC.view.frame = CGRectMake(self.scrollView.frame.size.width * 2, 0, self.scrollView.fr

    redVC.view.backgroundColor = [UIColor redColor];
    greenVC.view.backgroundColor = [UIColor yellowColor];
    [blueVC.view setBackgroundColor:[UIColor blueColor]];

    [self.scrollView addSubview:redVC.view];
    [self.scrollView addSubview:greenVC.view];
    [self.scrollView addSubview:blueVC.view];
    [self.view addSubview:self.scrollView];
    [self.scrollView setPagingEnabled:YES];
    self.scrollView.showsHorizontalScrollIndicator = NO;
}
```

Vertically

Create a SVA, create a class named ScrollViewController by subclassing UIViewController. (/* */make the scroll(both contentInset and scrollIndicatorInsets) not underlaps an interface element) See Pic UIScrollViewVertically

```
AppDelegate.m
Make the ScrollViewController root view controller

#import "ScrollViewController.h"
Inside - (Bool)application:didFinishLaunchingWithOptions: method add
self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]];

ScrollViewController *scrollVC = [[ScrollViewController alloc]init];
self.window.rootViewController = scrollVC;

[self.window makeKeyAndVisible];

return YES;


ScrollViewController.m
@implementation ScrollViewController

- (void)viewDidLoad {
        [super viewDidLoad];
        // Do any additional setup after loading the view.

    self.view.backgroundColor = [UIColor whiteColor];

    [self scrollViewMethod];
}

- (void)scrollViewMethod{
    UIScrollView *scrollView = [[UIScrollView alloc]initWithFrame:CGRectMake(0, 0, self.view.f
```

```
    /*CGFloat top = [[UIApplication sharedApplication]statusBarFrame].size.height;
    NSLog(@"Status bar frame height is %f",top);
    scrollView.contentInset = UIEdgeInsetsMake(top, 0, 0, 0);
    [scrollView scrollRectToVisible:CGRectMake(0, 0, 1, 1) animated:NO];
    scrollView.scrollIndicatorInsets = UIEdgeInsetsMake(top, 0, 0, 0);*/

    float y = 20;
    for (int i = 1; i < 21; i++) {
        UILabel *label = [[UILabel alloc]initWithFrame:CGRectMake(0, y, 320, 30)];
        label.text = [NSString stringWithFormat:@"Row %d",i];
        label.textColor = [UIColor whiteColor];
        label.textAlignment = NSTextAlignmentCenter;
        label.backgroundColor = [UIColor magentaColor];
        [scrollView addSubview:label];
        y += 100;
    }

    scrollView.showsVerticalScrollIndicator = NO;

    [scrollView setContentSize:CGSizeMake(self.view.frame.size.width, 2010)];

    [self.view addSubview:scrollView];
}
```


1. In effect, the contentSize is how large the scrollable content is.
2. A UIEdgeInsets struct (four CGFloats in the order **top, left, bottom, right**) specifying margins around the content. A typical use for this would be that your scroll view underlaps(self: under) an interface element, such as a translucent status bar, navigation bar, or toolbar, and you want your content to be visible even when scrolled to its limit.

4.18 NSTimer

Create a SVA, make the ViewController class root view controller(See 4.41). delete Main.storyboard and the corresponding row in info.plist. See Pic TimerCountdown. See project NSTimer

```
ViewController.m
@interface ViewController (){
    UILabel *timerLabel;
    NSTimer *timer;
}

@end

@implementation ViewController
int timeTick = 5;

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    timerLabel = [[UILabel alloc]initWithFrame:CGRectMake(60, 200, 200, 80)];
    timerLabel.textAlignment = NSTextAlignmentCenter;
    timerLabel.font = [UIFont systemFontOfSize:40];
    timerLabel.textColor = [UIColor blackColor];
    timerLabel.text = [NSString stringWithFormat:@"%d",timeTick];
```

```
    UIButton *button = [UIButton buttonWithType:UIButtonTypeCustom];
    button.frame = CGRectMake(120, 380, 100, 40);
    [button setTitleColor:[UIColor blueColor] forState:UIControlStateNormal];
    button.titleLabel.font = [UIFont systemFontOfSize:24];
    [button setTitle:@"Start" forState:UIControlStateNormal];
    [button addTarget:self action:@selector(startTimer) forControlEvents:
    UIControlEventTouchUpInside];

    [self.view addSubview:button];
    [self.view addSubview:timerLabel];
}

- (void)startTimer{
    [timer invalidate];
    timer = [NSTimer scheduledTimerWithTimeInterval:1.0f target:self selector:@selector(tick)
    userInfo:nil repeats:YES];

}

- (void)tick{
    if (timeTick == 0) {
        [timer invalidate];
        timeTick = 5;
        timerLabel.text = [NSString stringWithFormat:@"%d",timeTick];
    }else{
        timeTick--;
        timerLabel.text = [NSString stringWithFormat:@"%d",timeTick];
    }
}

@end
```

4.19 UIActionSheet

Create a SVA, make ViewController adopt UIActionSheetDelegate. See Pic ActionSheet. See project ActionSheetA

```
ViewController.m
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    UIButton *btn = [[UIButton alloc]initWithFrame:CGRectMake(50, 200, 180, 80)];
    btn.backgroundColor = [UIColor blackColor];
    btn.center = self.view.center;
    [btn setTitle:@"Click to upload photo" forState:UIControlStateNormal];
    [btn addTarget:self action:@selector(clickEvent) forControlEvents:
    UIControlEventTouchUpInside];
    [self.view addSubview:btn];
}

- (void)clickEvent{
    UIActionSheet *myActionSheet;
    myActionSheet = [[UIActionSheet alloc]initWithTitle:nil delegate:self cancelButtonTitle:
    @"Cancel" destructiveButtonTitle:nil otherButtonTitles:@"Photos",@"Take photo",@"Mine",
    nil];
```

```
    [myActionSheet showInView:self.view];
}

#pragma mark - ActionSheetDelegate
- (void)actionSheet:(UIActionSheet *)actionSheet clickedButtonAtIndex:(NSInteger)buttonIndex{
    UIAlertView *alertView;
    switch (buttonIndex) {
        case 0:
            alertView = [[UIAlertView alloc]initWithTitle:@"A short title is best" message:
            @"Photos" delegate:self cancelButtonTitle:nil otherButtonTitles:nil, nil];
            [alertView show];
            [self performSelector:@selector(dismissAlertView:) withObject:alertView
            afterDelay:2.0f];
            break;

        case 1:
            alertView = [[UIAlertView alloc]initWithTitle:@"A short title is best"
            message:@"Take photo" delegate:self cancelButtonTitle:nil otherButtonTitles:nil,
            nil];
            [alertView show];
            [self performSelector:@selector(dismissAlertView:) withObject:alertView
            afterDelay:2.0];

        default:
            break;
    }
}

#pragma mark - dismiss alertView
- (void)dismissAlertView:(UIAlertView *)alertView{
    if (alertView) {
        [alertView dismissWithClickedButtonIndex:[alertView cancelButtonIndex] animated:YES];
        alertView.hidden = YES;
    }
}
```

4.20 UISegmentedControl

```
- (void)buildSubviews{
    UIView *headerWhiteView = [[UIView alloc]initWithFrame:CGRectMake(0, 0, kScreenWidth, 50)];
    headerWhiteView.backgroundColor = [UIColor whiteColor];
    [self.view addSubview:headerWhiteView];

    NSArray *arr = @[@"随访问卷",@"检查检验"];
    UISegmentedControl *segment = [[UISegmentedControl alloc]initWithItems:arr];
    segment.frame = CGRectMake((kScreenWidth-200)/2, 10, 200, 30);
    segment.selectedSegmentIndex = 0;
    segment.selected = YES;
    segment.tintColor = WKColorApp;
    [segment addTarget:self action:@selector(onSegmentClick:) forControlEvents:UIControlEvent
    ValueChanged];
    [headerWhiteView addSubview:segment];
}
```

```
- (void)onSegmentClick:(UISegmentedControl *)segment{
    if (segment.selectedSegmentIndex == 0) {
        DLog(@"index zero selected");
    }else if (segment.selectedSegmentIndex == 1){
        DLog(@"index one selected");
    }
}
```

4.21 UIImagePickerController

Create a SVA, make the ViewController class root view controller(See 4.41). Delete Main.storyboard and the corresponding row in info.plist. Make ViewController adopt UIImagePickerControllerDelegate, UINavigation-ControllerDelegate. See project CameraAppProgrammatically

```
ViewController.m
@interface ViewController (){
    UIImageView *imageView;
}

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.
    if (![UIImagePickerController isSourceTypeAvailable:UIImagePickerControllerSourceTypeCamera
        UIAlertView *myAlertView = [[UIAlertView alloc]initWithTitle:@"Error"
        message:@"Device has no camera" delegate:nil cancelButtonTitle:@"OK"
        otherButtonTitles:nil, nil];
        [myAlertView show];
    }

    imageView = [[UIImageView alloc]initWithFrame:CGRectMake(0, 20, self.view.frame.size.width,

    UIButton *takePhotoBtn = [UIButton buttonWithType:UIButtonTypeSystem];
    takePhotoBtn.frame = CGRectMake(150, 300, 80, 40);
    [takePhotoBtn setTitle:@"take photo" forState:UIControlStateNormal];
    [takePhotoBtn addTarget:self action:@selector(takePhoto) forControlEvents:
    UIControlEventTouchUpInside];

    UIButton *selectPhotoBtn = [UIButton buttonWithType:UIButtonTypeSystem];
    selectPhotoBtn.frame = CGRectMake(150, 350, 100, 40);
    [selectPhotoBtn setTitle:@"select photo" forState:UIControlStateNormal];
    [selectPhotoBtn addTarget:self action:@selector(selectPhoto) forControlEvents:
    UIControlEventTouchUpInside];

    [self.view addSubview:imageView];
    [self.view addSubview:takePhotoBtn];
    [self.view addSubview:selectPhotoBtn];
}

- (void)takePhoto{
    UIImagePickerController *picker = [[UIImagePickerController alloc]init];
    picker.delegate = self;
    picker.allowsEditing = YES;
    picker.sourceType = UIImagePickerControllerSourceTypeCamera;
```

```
    [self presentViewController:picker animated:YES completion:NULL];
}

- (void)selectPhoto{
    UIImagePickerController *picker = [[UIImagePickerController alloc]init];
    picker.delegate = self;
    picker.allowsEditing = YES;
    picker.sourceType = UIImagePickerControllerSourceTypePhotoLibrary;

    [self presentViewController:picker animated:YES completion:NULL];
}

#pragma mark - UIImagePickerControllerDelegate
- (void)imagePickerController:(UIImagePickerController *)picker didFinishPickingMediaWithInfo:(
    UIImage *chosenImage = info[UIImagePickerControllerEditedImage];
    imageView.image = chosenImage;

    [picker dismissViewControllerAnimated:YES completion:NULL];
}

- (void)imagePickerControllerDidCancel:(UIImagePickerController *)picker{
    [picker dismissViewControllerAnimated:YES completion:NULL];
}
```

4.22 UITapGestureRecognizer

Create a SVA, make the ViewController class root view controller(See 4.41). See Pic LabelTap

```
ViewController.m
inside - (void)viewDidLoad method
UILabel *label = [[UILabel alloc]initWithFrame:CGRectMake(50, 200, 120, 30)];
label.backgroundColor = [UIColor orangeColor];
label.text = @"Tap me";
label.textColor = [UIColor blackColor];
label.font = [UIFont fontWithName:@"Helvetica Neue" size:12];
label.textAlignment = NSTextAlignmentCenter;
label.userInteractionEnabled = YES;
[self.view addSubview:label];

UITapGestureRecognizer *tapGR = [[UITapGestureRecognizer alloc]initWithTarget:self
action:@selector(labelTapped:)];

[label addGestureRecognizer:tapGR];


- (void)labelTapped:(UITapGestureRecognizer *)gestureRecognizer{
    UIAlertView *alert = [[UIAlertView alloc]initWithTitle:@"Warning" message:@"Simple text
    is best" delegate:nil cancelButtonTitle:@"OK" otherButtonTitles:nil, nil];
    [alert show];
}
```

Create a SVA, make the ViewController class root view controller(See 4.41).Create two classes named RoundedRect and TriangleShape by subclassing UIView respectively. Delete Main.storyboard and the corresponding info.plist file. See project UIGestureRecognizerYT for storyboard (also partly contains programmatically) and UIGestureRecognizerYTProgrammatically. See Pic SwipePanGesture
Pay close attentionn to initWithTarget, whether it is self or other object

ViewController.m

```
#import "TriangleShape.h"
#import "RoundedRect.h"
@interface ViewController (){
    TriangleShape *triangleShape;
}
@end

inside - (void)viewDidLoad method
UIView *view = [[UIView alloc]initWithFrame:CGRectMake(0, 0, self.view.frame.size.width, self.v
view.backgroundColor = [UIColor whiteColor];

triangleShape = [[TriangleShape alloc]init];
triangleShape.frame = CGRectMake(0, 0, 200, 200);
triangleShape.backgroundColor = [UIColor whiteColor];
triangleShape.userInteractionEnabled = YES;
[view addSubview:triangleShape];
[triangleShape addGestureRecognizer:[[UISwipeGestureRecognizer alloc]initWithTarget:self
action:@selector(swipe:)]];

RoundedRect *roundedRect = [[RoundedRect alloc]init];
roundedRect.frame = CGRectMake(200, 300, 80 , 120);
roundedRect.backgroundColor = [UIColor whiteColor];
[roundedRect addGestureRecognizer:[[UIPanGestureRecognizer alloc]initWithTarget:roundedRect
action:@selector(pan:)]];

[self.view addSubview:view];
[self.view addSubview:roundedRect];

- (void)swipe:(UISwipeGestureRecognizer *)sender{
    triangleShape.alpha = triangleShape.alpha - .2;
}
```

RoundedRect.m

```
- (void)drawRect:(CGRect)rect{
    UIBezierPath *roundedRect = [UIBezierPath bezierPathWithRoundedRect:self.bounds cornerRadiu
    [roundedRect addClip];  //make sure it's inside our rectangle not outside
    [[UIColor redColor]setFill];
    [[UIColor blackColor]setStroke];
    UIRectFill(self.bounds);
    [roundedRect stroke];
}

- (void)pan:(UIPanGestureRecognizer *)gesture{
    CGPoint translation = [gesture translationInView:gesture.view];
    gesture.view.center = CGPointMake(gesture.view.center.x + translation.x, gesture.view.cente

    [gesture setTranslation:CGPointZero inView:gesture.view];
}
```

TriangleShape.m

```
- (void)drawRect:(CGRect)rect{
    UIBezierPath *trianglePath = [[UIBezierPath alloc]init];
    [trianglePath moveToPoint:CGPointMake(25, 10)];
```

```
    [trianglePath addLineToPoint:CGPointMake(120, 140)];
    [trianglePath addLineToPoint:CGPointMake(7, 120)];
    [trianglePath closePath];
    [[UIColor greenColor] setFill];
    [[UIColor redColor] setStroke];
    [trianglePath fill];
    [trianglePath stroke];
}
```

4.23 Draw a custom view

1. Steps to draw something (See 4.22 project)
   1. Add drawRect method in .m file
      ```
      - (void)drawRect:(CGRect)rect{
      }
      ```
   2. Create an instance of UIBezierPath
   3. Define your custom path
   4. Set stroke and fill

Chapter 15. Drawing Create a SVA. See Pic Tiling the entire image of Mars
Make following changes:
`UIEdgeInsetsMake(mars.size.height/4.0, mars.size.width/4.0, mars.size.height/4.0, mars.size.wid`
See Pic Tiling the interior of Mars
Keep the above, change following part
`resizingMode:UIImageResizingModeStretch`
See Pic Stretching the interior of Mars

```
AppDelegate.m
#import "ViewController.h"
inside - (Bool)application:didFinishLaunchingWithOptions: method add
self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]];

ViewController *vController = [[ViewController alloc]init];
self.window.backgroundColor = [UIColor whiteColor];

self.window.rootViewController = vController;

[self.window makeKeyAndVisible];


ViewController.m
- (void)viewDidLoad {
[super viewDidLoad];
// Do any additional setup after loading the view, typically from a nib.

UIImage *mars = [UIImage imageNamed:@"activity_login.png"];
UIImage *marsTiled = [mars resizableImageWithCapInsets:UIEdgeInsetsZero resizingMode:UIImageRes

UIImageView *iv = [[UIImageView alloc]initWithFrame:CGRectMake(20, 25, mars.size.width*2, mars.
iv.image = marsTiled;

[self.view addSubview:iv];
}
```

Drawing a circle Create a SVA, make the ViewController class root view controller. See Pic BezierCircle

`inside - (void)viewDidLoad method add`

```
self.view.backgroundColor = [UIColor whiteColor];

UIBezierPath *bezierPath = [UIBezierPath bezierPath];
[bezierPath addArcWithCenter:CGPointMake(80, 240) radius:50 startAngle:0 endAngle:2 * M_PI cloc

CAShapeLayer *shape = [CAShapeLayer layer];
shape.path = bezierPath.CGPath;
shape.fillColor = [UIColor colorWithRed:255/255.0 green:25/255.0 blue:147/255.0 alpha:1].CGColo

[self.view.layer addSublayer:shape];
```

See Pic BezierCircle1

```
inside - (void)viewDidLoad method add
self.view.backgroundColor = [UIColor whiteColor];

UIBezierPath *path = [UIBezierPath bezierPath];
[path addArcWithCenter:CGPointMake(160, 240)
radius:50.0
startAngle:0.0
  endAngle:2.0 * M_PI
 clockwise:YES];

CAShapeLayer *shape = [CAShapeLayer layer];
shape.path = path.CGPath;
shape.fillColor = [UIColor colorWithRed:255/255.0 green:25/255.0 blue:147/255.0 alpha:1].CGColo

[self.view.layer addSublayer:shape];
```

See Pic BezierPath and BezierPath1

```
Inside - (void)viewDidLoad method add
self.view.backgroundColor = [UIColor whiteColor];

// 1
static CGFloat thickness = 13;
static CGFloat shapeSize = 43;

// 2
CGPoint startingPoint = CGPointMake(50, 50); // top-left corner
CGPoint points[6];
points[0] = startingPoint;
points[1] = CGPointMake(points[0].x + shapeSize, points[0].y);
points[2] = CGPointMake(points[0].x + shapeSize, points[0].y + thickness);
points[3] = CGPointMake(points[0].x + thickness, points[0].y + thickness);
points[4] = CGPointMake(points[0].x + thickness, points[0].y + shapeSize);
points[5] = CGPointMake(points[0].x, points[0].y + shapeSize);

// 3
CGMutablePathRef cgPath = CGPathCreateMutable();
CGPathAddLines(cgPath, &CGAffineTransformIdentity, points, sizeof points / sizeof *points);
CGPathCloseSubpath(cgPath);

UIBezierPath *path = [UIBezierPath bezierPathWithCGPath:cgPath];

// 4
CAShapeLayer *shape = [CAShapeLayer layer];
shape.path = path.CGPath;
```

```
// shape.path = bezierPath.CGPath;
shape.fillColor = [UIColor colorWithRed:255/255.0 green:25/255.0 blue:147/255.0 alpha:1].CGColo

// 5
[self.view.layer addSublayer:shape];
```

4.24 programmatically-linking-uipagecontrol-to-uiscrollview

Stackoverflow: programmatically-linking-uipagecontrol-to-uiscrollview
Create a SVA, add four classes named RedViewController GreenViewController BlueViewController respectively. See Pic UIPageControlUIScrollView

```
AppDelegate.m
Make the ViewController root view controller

#import "ViewController.h"
Inside - (Bool)application:didFinishLaunchingWithOptions: method add
self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]];

ViewController *viewController = [[ViewController alloc]init];

self.window.rootViewController = viewController;

[self.window makeKeyAndVisible];


ViewController.h
Conforms to <UIScrollViewDelegate>


ViewController.m
#import "RedViewController.h"
#import "GreenViewController.h"
#import "BlueViewController.h"

@interface ViewController ()

- (IBAction)changePage:(id)sender;

@property (nonatomic, strong) UIScrollView *scrollView;
@property (nonatomic, strong) UIPageControl * pageControl;

@end
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    self.view.backgroundColor = [UIColor whiteColor];

    RedViewController *redVC = [[RedViewController alloc]init];
    GreenViewController *greenVC = [[GreenViewController alloc]init];
    BlueViewController *blueVC = [[BlueViewController alloc]init];

    self.scrollView = [[UIScrollView alloc] initWithFrame:CGRectMake(0, 0, 320, 480)];
    [self.view addSubview:self.scrollView];

    self.scrollView.contentSize = CGSizeMake(self.scrollView.frame.size.width * 3, self.scroll
```

```
    self.scrollView.pagingEnabled = YES;
    self.scrollView.showsHorizontalScrollIndicator = NO;
    self.scrollView.delegate = self;

    redVC.view.frame = CGRectMake(self.scrollView.frame.size.width * 0, 0, self.scrollView.fra
    greenVC.view.frame = CGRectMake(self.scrollView.frame.size.width * 1, 0, self.scrollView.f
    blueVC.view.frame = CGRectMake(self.scrollView.frame.size.width * 2, 0, self.scrollView.fr

    redVC.view.backgroundColor = [UIColor redColor];
    greenVC.view.backgroundColor = [UIColor greenColor];
    blueVC.view.backgroundColor = [UIColor blueColor];

    [self.scrollView addSubview:redVC.view];
    [self.scrollView addSubview:greenVC.view];
    [self.scrollView addSubview:blueVC.view];

    self.pageControl = [[UIPageControl alloc] initWithFrame:CGRectMake(120, 420, self.scrollVi
    [self.pageControl setPageIndicatorTintColor:[UIColor blackColor]];
    [self.pageControl setCurrentPageIndicatorTintColor:[UIColor whiteColor]];
    self.pageControl.numberOfPages = self.scrollView.contentSize.width/self.scrollView.frame.s
   // self.pageControl.currentPage = 1;
    [self.pageControl addTarget:self action:@selector(changePage:) forControlEvents:UIControlE

    [self.view addSubview:self.pageControl];
}

- (IBAction)changePage:(id)sender {
    CGFloat x = self.pageControl.currentPage * self.scrollView.frame.size.width;
    [self.scrollView setContentOffset:CGPointMake(x, 0) animated:YES];
}

-(void) scrollViewDidEndDecelerating:(UIScrollView *)scrollView  {
    NSInteger pageNumber = lround(scrollView.contentOffset.x / (scrollView.frame.size.width));
    self.pageControl.currentPage = pageNumber;

    NSLog(@"End decelerate");
}
Or:
- (void)scrollViewDidScroll:(UIScrollView *)scrollView
{
    CGFloat doublePage = scrollView.contentOffset.x / scrollView.frame.size.width;
    int intPage = (int)(doublePage + 0.5);

    self.pageControl.currentPage = intPage;
}
```

See gif scrollViewPageControl for the difference

4.25 Dismiss(resign) the keyboard

refer to 4.3

```
- (void)touchesBegan:(NSSet<UITouch *> *)touches withEvent:(UIEvent *)event{
    [super touchesBegan:touches withEvent:event];
    [passwordTextField resignFirstResponder];
    // [self.view endEditing:YES];
```

```
    NSLog(@"touchesBegan:withEvent");
}

#pragma mark -- hideKeyboardAuto
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
  // [self.phoneNumTextField resignFirstResponder];
    [self.view endEditing:YES];
}
```

4.26 Dealing with strings and NSLog

1. `NSString *footerTitle;`
   `footerTitle = @"Section 1 Footer";`
2. `NSLog(@"Section:%ld Row:%ld selected and its data is %@",(long)indexPath.section,`
   `(long)indexPath.row,cell.textLabel.text);`
   NSData and NSError are in `%@`
3. `NSInteger i = 0;`
   `NSString *info = [NSString stringWithFormat:@"Student %ld",(long)i];`
   `(_waitDoctorLabel.text = [NSString stringWithFormat:@"%@  %@%@", _waitDoctorLabel.text,`
   `_doctorLabel.text,_doctorNumberLabel.text];` the font goes with the first arg)
4. `_waitDoctorLabel.text = waitLM.waitName;`
   `_waitDoctorLabel.text = [_waitDoctorLabel.text stringByAppendingString:@"DoctorNumber:"];`
   `_waitDoctorLabel.text = [_waitDoctorLabel.text stringByAppendingFormat:@"%@%@",`
   `_doctorLabel.text,_doctorNumberLabel.text];`
5. `@property (nonatomic, strong) NSString *str1;`
   `@property (nonatomic, strong) NSString *str2;`
   `@property (nonatomic, strong) NSString *str3;`

   ```
   self.str1 = @"string one ";
   self.str2 = @"string two";
   self.str3 = [self.str1 stringByAppendingString:self.str2];
   // You have to initialize str3 first to make it work otherwise it will log null
   /*self.str3 = @"";
   self.str3 = [self.str3 stringByAppendingFormat:@"(%@) -> **%@**",self.str1, self.str2];*/
   NSLog(@"%@", self.str3);
   ```
6. `NSString *http = [[kWebServiceURL substringToIndex:28]stringByAppendingString:@"appended"]`
7. The solution for me was to replace occurrences of `"\\n"` with `"\n"`.
   `myString = [myString stringByReplacingOccurrencesOfString:@"\\n" withString:@"\n"];`
8. Add space inside NSString
   In the same class add following method
   `-(NSString*)stringByAddingSpace:(NSString*)stringToAddSpace spaceCount:(NSInteger)`
   `spaceCount atIndex:(NSInteger)index{`
   ```
       NSString *result = [NSString stringWithFormat:@"%@%@",[@" " stringByPaddingToLength:
       spaceCount withString:@" " startingAtIndex:0],stringToAddSpace];
       return result;
   }
   ```
   then somewhere in your code call this method
   ```
   NSString *str = [self stringByAddingSpace:self.hospital spaceCount:5 atIndex:0];
   hospital.text = [NSString stringWithFormat:@"Hospital: %@",str];
   [view1 addSubview:hospital];
   ```
9. componentsSeparatedByString method
   `@property (nonatomic, strong) NSArray *dataSource;`
   `self.dataSource = [self.resultModel.depart_names componentsSeparatedByString:@"、"];`
10. NSMutableParagraphStyle
    `NSMutableAttributedString *mutableAttrString = [[NSMutableAttributedString alloc]`

```
initWithString:self.dataSource[indexPath.section][indexPath.row]];
NSMutableParagraphStyle *mutableParaStyle = [[NSMutableParagraphStyle alloc]init];
[mutableParaStyle setLineSpacing:8];
[mutableAttrString addAttribute:NSParagraphStyleAttributeName value:mutableParaStyle
range:NSMakeRange(0, [self.dataSource[indexPath.section][indexPath.row] length])];
cell.textLabel.attributedText = mutableAttrString;
```

4.27 Handling attributed string and text

In iOS 6+ you can render attributed strings using the attributedText property of UILabel. See Pic ConcatenateAttrString

1. **Once and for all**
```
UIColor *red = [UIColor redColor];
NSAttributedString *attrStr = [[NSAttributedString alloc]initWithString:@"已指导"
attributes:@{NSBaselineOffsetAttributeName:@(0.0),NSUnderlineStyleAttributeName:
@(NSUnderlineStyleSingle), NSStrokeWidthAttributeName:@(-3.0),
NSUnderlineColorAttributeName:red}];
```
2. **Attention**
Following will not work
```
NSDictionary *underlineAttribute = @{NSUnderlineStyleAttributeName:
@(NSUnderlineStyleSingle)};
NSAttributedString *attrStr = [[NSAttributedString alloc]initWithString:@"已指导"
attributes:@{NSUnderlineStyleAttributeName: underlineAttribute,
NSBaselineOffsetAttributeName:@(0.0)}];
```
But this does work
```
//NSDictionary *underlineAttribute = @{NSUnderlineStyleAttributeName:
@(NSUnderlineStyleSingle)};
NSAttributedString *attrStr = [[NSAttributedString alloc]initWithString:@"已指导"
attributes:@{NSUnderlineStyleAttributeName: @(NSUnderlineStyleSingle),
NSBaselineOffsetAttributeName:@(0.0)}];
```
3.
```
NSDictionary *attrDict = @{NSForegroundColorAttributeName: [UIColor blueColor]};
NSAttributedString *attrString = [[NSAttributedString alloc]initWithString:
waitLM.waitSequence attributes:attrDict];
NSMutableAttributedString *mutableAttrString = [[NSMutableAttributedString alloc]
initWithString:@"waitSequenceString:  "];
[mutableAttrString appendAttributedString: attrString];
_waitSequenceLabel.attributedText = mutableAttrString;
```

4.
```
NSDictionary *dict = self.patientArr[section];
NSDictionary *attrDict = @{NSForegroundColorAttributeName: WKColorGray};
NSMutableAttributedString *mutableAttrString = [[NSMutableAttributedString alloc]
initWithString:@"something: " attributes:attrDict];
NSDictionary *attrFontDict = @{NSFontAttributeName: WKFontNor_15};
NSAttributedString *attrString = [[NSAttributedString alloc]initWithString:[NSString
stringWithFormat:@"%@",dict[@"jzrName"]] attributes:attrFontDict];
[mutableAttrString appendAttributedString:attrString];

myLabel.attributedText = mutableAttrString;
```
5. **Underline**
```
NSDictionary *underlineAttribute = @{NSUnderlineStyleAttributeName:
@(NSUnderlineStyleSingle)};
myLabel.attributedText = [[NSAttributedString alloc] initWithString:@"Test
string" attributes:underlineAttribute];
```

Create a SVA, make the ViewController class root view controller(See 4.41). See Pic NSAttributedString See project NSAttributedString

`ViewController.m`

```
#define FONT_SIZE 20
#define FONT_HELVETICA @"Helvetica-Light"
#define BLACK_SHADOW [UIColor colorWithRed:40.0f/255.0f green:40.0f/255.0f blue:40.0f/255.0f
alpha:0.4f]
inside - (void)viewDidLoad method
//How to set the color of an attributed text
    UIColor *fgColor = [UIColor greenColor];
    NSDictionary *styleFG = @{NSForegroundColorAttributeName: fgColor};
    NSAttributedString *myFGString = [[NSAttributedString alloc]initWithString:
    @"Touch Code Magazine" attributes:styleFG];
    UILabel *FGlabel = [[UILabel alloc]initWithFrame:CGRectMake(80, 100, 200, 40)];
    FGlabel.attributedText = myFGString;

    //How to set the background color of an attributed text
    UIColor *bgColor = [UIColor orangeColor];
    NSDictionary *styleBG = @{NSBackgroundColorAttributeName: bgColor};
    NSAttributedString *myBGString = [[NSAttributedString alloc]initWithString:
    @"Touch Code Magazine" attributes:styleBG];
    UILabel *BGlabel = [[UILabel alloc]initWithFrame:CGRectMake(80, 150, 200, 40)];
    BGlabel.attributedText = myBGString;

    //How to set the font of an attributed text
    UIFont *myFont = [UIFont fontWithName:@"Zapfino" size:18];
    NSDictionary *styleFont = @{NSFontAttributeName: myFont};
    NSAttributedString *myFontString = [[NSAttributedString alloc]initWithString:
    @"Touch Code Magazine" attributes:styleFont];
    UILabel *fontLabel = [[UILabel alloc]initWithFrame:CGRectMake(40, 200, 320, 60)];
    fontLabel.attributedText = myFontString;

    NSMutableAttributedString *string = [[NSMutableAttributedString alloc]initWithString:
    @"firstsecondthird"];
    [string addAttribute:NSForegroundColorAttributeName value:[UIColor redColor]
    range:NSMakeRange(0, 5)];
    [string addAttribute:NSForegroundColorAttributeName value:[UIColor greenColor]
    range:NSMakeRange(5, 6)];
    [string addAttribute:NSForegroundColorAttributeName value:[UIColor blueColor]
    range:NSMakeRange(11, 5)];
    UILabel *stringLabel = [[UILabel alloc]initWithFrame:CGRectMake(40, 280, 320, 60)];
    stringLabel.attributedText = string;

    /*NSDictionary *wordToColorMapping = [[NSDictionary alloc]initWithObjects:[NSArray arrayWit
    NSMutableAttributedString *mutableString = [[NSMutableAttributedString alloc]initWithString
    for (NSString *word in wordToColorMapping){
        UIColor *color = [wordToColorMapping objectForKey:word];
        NSDictionary *attributes = [NSDictionary dictionaryWithObject:color forKey:NSForeground
        NSAttributedString *subString = [[NSAttributedString alloc]initWithString:word attribut
        [mutableString appendAttributedString:subString];
    }
    UILabel *mutableStringLabel = [[UILabel alloc]initWithFrame:CGRectMake(40, 350, 320, 60)];
    mutableStringLabel.attributedText = mutableString;*/

    NSString *myNSString = @"This is my string.\n It goes to a second line.";
    NSMutableParagraphStyle *paragraphStyle = [[NSMutableParagraphStyle alloc] init];
    paragraphStyle.alignment = NSTextAlignmentCenter;
    paragraphStyle.lineSpacing = FONT_SIZE/2;
    UIFont *labelFont = [UIFont fontWithName:FONT_HELVETICA size:FONT_SIZE];
```

```
    UIColor *labelColor = [UIColor colorWithWhite:1 alpha:1];
    NSShadow *shadow = [[NSShadow alloc]init];
    [shadow setShadowColor:BLACK_SHADOW];
    [shadow setShadowOffset:CGSizeMake(1.0, 1.0)];
    [shadow setShadowBlurRadius:1];
    NSAttributedString *labelText = [[NSAttributedString alloc]initWithString:myNSString
        attributes:@{NSParagraphStyleAttributeName:paragraphStyle,
                                NSKernAttributeName:@2.0,
                                NSFontAttributeName:labelFont,
                    NSForegroundColorAttributeName:labelColor,
                             NSShadowAttributeName:shadow}];
    UILabel *myNSStringLabel = [[UILabel alloc]initWithFrame:CGRectMake(40, 250, 320, 360)];
    myNSStringLabel.numberOfLines = 0;  // This line is crucial
    myNSStringLabel.attributedText = labelText;

    [self.view addSubview:FGlabel];
    [self.view addSubview:BGlabel];
    [self.view addSubview:fontLabel];
    [self.view addSubview:stringLabel];
//    [self.view addSubview:mutableStringLabel];  // Doesn't work
    [self.view addSubview:myNSStringLabel];
```

4.28 Dealing with sizes(all in points not pixels)

```
NSLog(@"Navframe Height=%f, %f",
      self.navigationController.navigationBar.frame.size.height,
      [UIApplication sharedApplication].statusBarFrame.size.height);

// statusBar height is 20 (40px). navigationBar height is 44 (88px).

NSLog(@"Cell height: %f",cell.size.height); // default cell height is 44 (inside
// cellForRowAtIndexPath) method
```

1. Get the current screen size
   ```
   CGRect screenRect = [[UIScreen mainScreen]bounds];
   CGFloat screenWidth = screenRect.size.width;
   CGFloat screenHeight = screenRect.size.height;
   Or:
   #define kScreenBounds [UIScreen mainScreen].bounds
   #define kScreenWidth [UIScreen mainScreen].bounds.size.width
   #define kScreenHeight [UIScreen mainScreen].bounds.size.height
   ```

4.29 what-is-the-difference-between-post-and-get

According to Wikipedia:

GET requests a representation of the specified resource.

POST submits data to be processed (e.g., from an HTML form) to the identified resource. The data is included in the body of the request. This may result in the creation of a new resource or the updates of existing resources or both.

So essentially GET is used to retrieve remote data, and POST is used to insert/update remote data.

4.30 Those characters refer to the source control.

```
? Unversioned
M Modified
```

```
A Added
A+ Moved / renamed
U Newer version of a file on source control
```

4.31 Bars

1. Status Bar
2. Navigation Bar
3. Toolbar
4. Tab Bar
5. Search Bar
6. Scope Bar

4.32 NSNumber, id type

```
@interface ViewController ()
@property (nonatomic, strong) NSNumber *myNumber;
@end

@implementation ViewController
@synthesize myNumber = _myNumber;  // only myNumber (self)
- (void)viewDidLoad {
[super viewDidLoad];
    NSMutableArray *localArray = [[NSMutableArray alloc]init];
    int myInt = 2;
    self.myNumber = [NSNumber numberWithInt:myInt];
    [localArray addObject:self.myNumber];
    //primitive type:
    //int
    //BOOL
    //float
    //char
    //double
    //short
    //long
    //NSInteger (self concluded)
    float myFloat = [self.myNumber floatValue]; // convert to float value
    NSString *myString = [self.myNumber stringValue]; // convert to string value
```

id is for the time that we don't know the type of the object. `(id) sender` $\mapsto$ `(UIButton *) button`

NSNumber Literal
Many times it becomes necessary to put primitive types in a collection object such as an NSArray or NSDictionary. The problem is you quickly find out that NSArray or NSDictionary only accept Objective-C objects which excludes primitives such as numbers!
The way to get around this is to box the primitive number with an NSNumber like so

```
NSInteger myLuckyNumber = 17;
NSNumber *myLuckyNumberObject = [NSNumber numberWithInteger:myLuckyNumber];
NSArray *luckyNumbers = [NSArray arrayWithObject:myLuckyNumberObject];
```

4.33 NSURLConnection

A few thing to notice: For example here are a few of the properties defined on NSURLRequest
URL the url of the server to which the request should be sent

HTTPMethod The http protocol defines a set of methods that can be performed, the most common are GET, POST, PUT, and DELETE. For our example we will be using GET

HTTPBody this defines the data to be sent with the request/response, in the APP.NET example above the JSON response received was the HTTP body

Using storyboard, details on how to implement NSURLConnection See project urlFetcher.

```
- (IBAction)makeRequest:(id)sender{
    // disable the fetch button when making request
    self.fetchButton.enabled = NO;

    // begin animating the spinner
    [self.spinner startAnimating];

    // create the request
    NSURLRequest *request = [NSURLRequest requestWithURL:[NSURL URLWithString:@"https:
    //alpha-api.app.net/stream/0/posts/stream/global"]];

    // create the connection
    self.connection = [NSURLConnection connectionWithRequest:request delegate:self];

    // ensure the connection was created
    if (self.connection) {
        // initialize the buffer
        self.buffer = [NSMutableData data];

        // start the request
        [self.connection start];
    }else{
        self.textField.text = @"Connection failed";
    }
}

- (IBAction)clear:(id)sender{
    self.textField.text = @"";
}

- (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error{
    // clear the connection and the buffer
    self.connection = nil;
    self.buffer = nil;

    self.textField.text = [error localizedDescription];
    NSLog(@"Connection failed ! Error - %@ %@", [error localizedDescription],
    [[error userInfo]objectForKey:NSURLErrorFailingURLStringErrorKey]);
}

- (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response{
    // reset the buffer length each time this method is called
    [self.buffer setLength:0];
}

- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data{
    // append data to the buffer
    [self.buffer appendData:data];
}

- (void)connectionDidFinishLoading:(NSURLConnection *)connection{
```

```
    // dispatch off the main queue for json processing
    dispatch_async(dispatch_get_global_queue(DISPATCH_QUEUE_PRIORITY_DEFAULT, 0), ^{
        NSError *error = nil;
        NSString *jsonString = [[NSJSONSerialization JSONObjectWithData:_buffer options:0
        error:&error]description];

        // dispatch back to the main queue for UI
        dispatch_async(dispatch_get_main_queue(), ^{

            // check for a JSON error
            if (!error) {
                self.textField.text = jsonString;
            }
            else
            {
                self.textField.text = [error localizedDescription];
            }

            // stop animating and re-enable the fetch button
            [self.spinner stopAnimating];
            self.fetchButton.enabled = YES;

            // clear the connection and the buffer
            self.connection = nil;
            self.buffer = nil;
        });

    });
}
```

4.34 UIWebView

```
NSURL *url = [NSURL URLWithString:@"http://m.365pad.net/test/yiyuantong/followupdetails.html"];
NSURLRequest *request = [NSURLRequest requestWithURL:url];
UIWebView *webView = [[UIWebView alloc]initWithFrame:CGRectMake(0, 0, kScreenWidth, kScreenHeig
[webView loadRequest:request];
webView.delegate = self;
[self.view addSubview:webView];
```

4.35 NSUserDefaults UISwitch

See project NSUserDefaults (in storyboard)

1. Can only store following types (light data):
    - NSArray
    - NSDictionary
    - NSNumber
    - NSString
    - NSDate
    - NSData

Create a SVA, make the ViewController class root view controller. See Pic userDefaults See project NSUserDefaultsProgrammatically

```
ViewController.m
@interface ViewController (){
    UISwitch *_rememberSwitch;
    UITextField *_usernameField;
}
```

```
@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    UILabel *username = [[UILabel alloc]initWithFrame:CGRectMake(20, 80, 150, 20)];
    username.text = @"Username";

    _usernameField = [[UITextField alloc]initWithFrame:CGRectMake(20, 110, 200, 30)];
    _usernameField.borderStyle = UITextBorderStyleRoundedRect;


    UILabel *rememberMe = [[UILabel alloc]initWithFrame:CGRectMake(20, 270, 120, 20)];
    rememberMe.text = @"Remember Me";

    _rememberSwitch = [[UISwitch alloc]initWithFrame:CGRectMake(150, 265, 0, 0)];

    UIButton *login = [UIButton buttonWithType:UIButtonTypeRoundedRect];
    login.frame = CGRectMake(160, 320, 50, 20);
    [login setTitle:@"Login" forState:UIControlStateNormal];
    [login addTarget:self action:@selector(signin) forControlEvents:UIControlEventTouchUpInside

    [_rememberSwitch setOn:[[NSUserDefaults standardUserDefaults]boolForKey:@"switchState"]];
    if (_rememberSwitch.isOn) {
        _usernameField.text = [[NSUserDefaults standardUserDefaults]stringForKey:@"username"];
    }


    [self.view addSubview:username];
    [self.view addSubview:_usernameField];
    [self.view addSubview:rememberMe];
    [self.view addSubview:_rememberSwitch];
    [self.view addSubview:login];
}

- (void)signin{
    if (_rememberSwitch.isOn) {
        [[NSUserDefaults standardUserDefaults]setValue:_usernameField.text forKey:@"username"];
    }
    [[NSUserDefaults standardUserDefaults]setBool:_rememberSwitch.isOn forKey:@"switchState"];
}
```

4.36 UIView Animation

Create a SVA, import two images to the project named `door_bottom.png` and `door_top.png` (pay attention to the image size fitting the iphone 4s screen size), create two IBOutlet for these two images named basketBottom and basketTop respectively in ViewController.m class extension area. In Main.storyboard drag two UIImageView to canvas and set one on top and one on bottom, then set these two imageView's image to the imported two images and view mode top and bottom respectively. Adjust the two images to look good See pic Lookgood. Finally link the two IBOutlet to the corresponding images on the canvas (go to storyboard to do this). See project Animation. See pic UIViewAnimation
There are two ways to do this, one is not using block and one using block

`ViewController.m`

```
#import "ViewController.h"

@interface ViewController ()

@property (nonatomic, weak) IBOutlet UIImageView *basketTop;
@property (nonatomic, weak) IBOutlet UIImageView *basketBottom;

@end

@implementation ViewController

- (void)viewDidAppear:(BOOL)animated
{
    CGRect basketTopFrame = self.basketTop.frame;
    basketTopFrame.origin.y = -basketTopFrame.size.height;

    CGRect basketBottomFrame = self.basketBottom.frame;
    basketBottomFrame.origin.y = self.view.bounds.size.height;

    [UIView beginAnimations:nil context:nil];
    [UIView setAnimationDuration:1];
    [UIView setAnimationDelay:3.0];
    [UIView setAnimationCurve:UIViewAnimationCurveEaseOut];

    self.basketTop.frame = basketTopFrame;
    self.basketBottom.frame = basketBottomFrame;

    [UIView commitAnimations];

//    [UIView animateWithDuration:1 delay:3.0 options:UIViewAnimationCurveEaseOut animations:^{
//        self.basketTop.frame = basketTopFrame;
//        self.basketBottom.frame = basketBottomFrame;
//    } completion:^(BOOL finished) {
//        NSLog(@"Done!");
//    }];
}
```

For a more advanced use of UIView Animation see project AnimationASecondLayerOfFun

4.37 CALayer

```
self.view.layer.backgroundColor = [UIColor orangeColor].CGColor;
self.view.layer.cornerRadius = 20.0;
self.view.layer.frame = CGRectInset(self.view.layer.frame, 20, 20);
```

- To get a pointer to the layer for the view, you just use self.view.layer. Remember, you can get the root view for a View Controller by calling self.view. Once you have the view, you can get its default layer (created automatically) by using view.layer. By default, the layer is a CALayer (not a subclass)
- Next you set the layer's background color to orange. Note that the backgroundColor property actually takes a CGColor, but it's easy to convert a UIColor to a CGColor by using the CGColor property
- Next you round the corners a bit by setting the corner radius. The value you pass in is the radius of the circle which makes up the corner, and it chooses 20 here for a nice rounded edge
- Finally, you shrink the frame a bit so it's easier to see, by using the handy CGRectInset function. The CGRectInset function takes a frame and an amount to shrink it by (for both the X and Y), and returns a new frame

Note: remember that these coordinates are relative to the parent layer's frame means its coordinate is in the parent view coordinate not the screen

For info on how to round the corners of an image see project LayerFun (you may need to create an SVA and then add the source code to it in order to run successfully due to location change of pch file), See pic LayerFun

4.38 Push notifications (APNS)

1. There are three things a push notification can do:
   1. Display a short text message
   2. Play a brief sound
   3. Set a number in a badge on the app's icon
2. See pic APNSProcedure
   1. An app enables push notifications. The user has to confirm that he wishes to receive these notifications
   2. The app receives a "device token". You can think of the device token as the address that push notifications will be sent to
   3. The app sends the device token to your server
   4. When something of interest to your app happens, the server sends a push notification to the Apple Push Notification Service, or APNS for short
   5. APNS sends the push notification to the user's device
3. Local notifications are limited to scheduling timed events and unlimited background processing is only available to apps that do VOIP, navigation or background audio
4. What You Need for Push Notifications
   1. An iPhone or iPad. Push notifications do not work in the simulator, so you will need to test on the device
   2. An iOS Developer Program membership. You need to make a new App ID and provisioning profile for each app that uses push, as well as an SSL certificate for the server. You do this at the iOS Provisioning Portal
   3. A server that is connected to the internet. Push notifications are always sent by a server. For development you can use your Mac as the server (which you'll do in this tutorial) but for production use, you need at least something like a VPS (Virtual Private Server)
5. Anatomy of a Push Notification
   1. A push notification is a short message that consists of the device token, a payload, and a few other bits and bytes. The payload is what you are interested in, as that contains the actual data you will be sending around
   2. Your server should provide the payload as a JSON dictionary. The payload for a simple push message looks like this

```
{
    "aps":
    {
        "alert": "Hello, world!",
        "sound": "default"
    }
}
```

   3. There are other items you can add to the "aps" section to configure the notification. For example:

```
{
    "aps":
        {
        "alert":
        {
            "action-loc-key": "Open",
            "body": "Hello, world!"
        },
        "badge": 2
    }
}
```

   4. Push notifications are intended to be small; the payload size can be no more than 256 bytes

5. A smart push server won't waste space on newlines and whitespace and generates something that looks like:
   `{"aps":{"alert":"Hello, world!","sound":"default"}}`
   It's less easy to read for us humans, but it saves enough bytes to make it worth it. Push notifications whose payload exceeds 256 bytes will not be accepted by APNS
6. Push Notification Gotchas
   1. They are not reliable! There is no guarantee that push notifications will actually be delivered, even if the APNS server accepted them
   2. As far as your server is concerned, push notifications are fire-and-forget; there is no way to find out what the status of a notification is after you've sent it to APNS. The delivery time may also vary, from seconds up to half an hour
   3. Also, the user's iPhone may not be able to receive push notifications all the time. They could be on a WiFi network that does not allow connections to be made to APNS because the required ports are blocked. Or the phone could be turned off
   4. APNS will try to deliver the last notification it received for that device when it comes back online, but it will only try for a limited time. Once it times out, the push notification will be lost forever!
   5. They can be expensive! Adding push functionality to your app is fairly easy and inexpensive if you own the data, but can be expensive if you have a lot of users or data you need to poll
7. Provisioning Profiles and Certificates, Oh My!
   As you know, apps use different provisioning profiles for development and distribution. There are also two types of push server certificates
   - **Development** If your app is running in Debug mode and is signed with the Development provisioning profile (Code Signing Identity is "iPhone Developer"), then your server must be using the Development certificate
   - **Production** Apps that are distributed as Ad Hoc or on the App Store (when Code Signing Identify is "iPhone Distribution") must talk to a server that uses the Production certificate. If there is a mismatch between these, push notifications cannot be delivered to your app
8. Generating the Certificate Signing Request (CSR)
   1. but you do need to be aware that a certificate always works in combination with a private key. The certificate is the public part of this key pair. It's important to know that you can't use the certificate if you don't have the private key.
   2. Whenever you apply for a digital certificate, you need to provide a Certificate Signing Request, or CSR for short
   3. Open Keychain Access on your Mac (it is in Applications/Utilities) and choose the menu option Request a Certificate from a Certificate Authority...
      If you do not have this menu option or it says "Request a Certificate from a Certificate Authority with key", then download and install the WWDR Intermediate Certificate first. Also make sure no private key is selected in the main Keychain Access window
   4. You should now see the following window See pic CertificateAssistant
      a. Enter your email address here. I've heard people recommended you use the same email address that you used to sign up for the iOS Developer Program, but it seems to accept any email address just fine
      b. Enter "PushChat"(here PushChatTest) for Common Name. You can type anything you want here, but choose something descriptive. This allows us to easily find the private key later
      c. Check Saved to disk and click Continue. Save the file as "PushChat.certSigningRequest" (here PushChatTest)
      d. If you go to the Keys section of Keychain Access, you will see that a new private key has appeared in your keychain. Right click it and choose Export See pic Keys
      e. Save the private key as PushChatKey.p12 and enter a passphrase (here PushChatKeyTest and passphrase is in the next step (passphrase:pushchattest))
9. Making the App ID and SSL Certificate
   1. Log in to the iOS Dev Center and "Select the Certificates, Identifiers and Profiles"
   2. select Certificates in the iOS Apps section. Now, you are going to make a new App ID. Each push app needs its own unique ID because push notifications are sent to a specific application. (You cannot use a wildcard ID.)
   3. Go to App IDs in the sidebar and click the + button

4. Fill the following details
   App ID Description : PushChat (here PushChatTest)
   App Services Check the Push Notifications Checkbox
   Explicit App ID com.hollance.PushChat (here com.jiapan.you) It is probably best if you choose your own Bundle Identifier here - com.yoursite.PushChat - instead of using mine. You will need to set this same bundle ID in your Xcode project.
5. After you're done filling all the details press the Continue button. You will be asked to verify the details of the app id, if everything seems okay click Submit
6. Hurray! You have successfully registered a new App ID
7. After you have made the App ID, it shows up like this in the list See pic AppID
8. Select the PushChat (here PushChatTest) app ID from the list. This will open up an accordion as shown See pic PushChatTestID
9. Click on the Setting button to configure these settings (actually Edit button)
10. Scroll down to the Push Notifications section and select the Create Certificate button in the Development SSL Certificate section See pic PushNotificationsSection
11. The "Add iOS Certificate" wizard comes up See pic iOSCertificateWizard
12. The first thing it asks you is to generate a Certificate Signing Request. You already did that, so click Continue. In the next step you upload the CSR. Choose the CSR file that you generated earlier and click Generate
13. It takes a few seconds to generate the SSL certificate. Click Continue when it's done
14. Now click Download to get the certificate - it is named "aps_development.cer"
15. As you can see, you have a valid certificate and push is now available for development. You can download the certificate again here if necessary. The development certificate is only valid for 3 months See pic CertificateExpires (it says 12 months)
16. When you are ready to release your app, repeat this process for the production certificate. The steps are the same
17. **Note:** The production certificate remains valid for a year, but you want to renew it before the year is over to ensure there is no downtime for your app
18. You don't have to add the certificate to your Keychain, although you could if you wanted to by double-clicking the downloaded aps_development.cer file. If you do, you'll see that it is now associated with the private key

10. Making a PEM File
    1. So now you have three files
       - The CSR
       - The private key as a p12 file (PushChatKey.p12)
       - The SSL certificate, aps_development.cer
    2. Store these three files in a safe place. By re-using the CSR you can keep using your existing private key and only the .cer file will change
    3. You have to convert the certificate and private key into a format that is more usable. Because the push part of our server will be written in PHP, you will combine the certificate and the private key into a single file that uses the PEM format
    4. If you write your push server in another language, these following steps may not apply to you. Open a Terminal and execute the following steps
       a. Go to the folder where you downloaded the files
       b. Convert the .cer file into a .pem file
          ```
          $ openssl x509 -in aps_development.cer -inform der -out PushChatCert.pem
          ```
       c. Convert the private key's .p12 file into a .pem file
          ```
          $ openssl pkcs12 -nocerts -out PushChatKey.pem -in PushChatKey.p12
          Enter Import Password:
          MAC verified OK
          Enter PEM pass phrase:
          Verifying - Enter PEM pass phrase:
          ```
          You first need to enter the passphrase for the .p12 file so that openssl can read it. Then you need to enter a new passphrase that will be used to encrypt the PEM file. Again for this tutorial I used "pushchat" as the PEM passphrase (here pushchattest). You should choose something more secure
       d. Finally, combine the certificate and key into a single .pem file

```
$ cat PushChatCert.pem PushChatKey.pem > ck.pem
```

e. At this point it's a good idea to test whether the certificate works. Execute the following command

```
$ telnet gateway.sandbox.push.apple.com 2195
Trying 17.172.232.226...
Connected to gateway.sandbox.push-apple.com.akadns.net.
Escape character is '^]'.
```

This tries to make a regular, unencrypted, connection to the APNS server. If you see the above response, then your Mac can reach APNS. Press Ctrl+C to close the connection. If you get an error message, then make sure your firewall allows outgoing connections on port 2195

f. Let's try connecting again, this time using our SSL certificate and private key to set up a secure connection

```
$ openssl s_client -connect gateway.sandbox.push.apple.com:2195 -cert
PushChatCert.pem -key PushChatKey.pem
Enter pass phrase for PushChatKey.pem:
```

You should see a whole bunch of output, which is openssl letting you know what is going on under the hood

g. If the connection is successful (See pic ConnectionSuccessful), you should be able to type a few characters. When you press enter, the server should disconnect. If there was a problem establishing the connection, openssl will give you an error message but you may have to scroll up through the output to find it

h. **Note:** There are two different APNS servers: the "sandbox" server that you can use for testing, and the live server that you use in production mode. Above, we used the sandbox server because our certificate is intended for development, not production use.

11. Making the Provisioning Profile

You're not yet done with the iOS Dev Center. Click the Provisioning Profiles button in the sidebar and click the + button

Step 1: Select Type Select the "iOS App development" option button in the first step of the wizard and press Continue

Step 2: Configure Select the PushChat app id (here PushChatTest) that you created in the previous section. This will ensure that this provisioning profile is explicitly tied to the PushChat app

Step 3: Generate In this step you select the certificates you want to include in this provisioning profile. This step should be quite routine by now

Step 4: Select devices Select the devices you want to include in this provisioning profile

Step 5: Name this profile Set the provisioning profile name as "PushChat Development" (here PushChat Development Test)

1. You're almost done! Finally press the Download button, this will download the newly created Development provisioning profile
2. Add the provisioning profile to Xcode by double-clicking it or dragging it onto the Xcode icon
3. If you're ready to release your app to the public, you will have to repeat this process to make an Ad Hoc or App Store distribution profile

12. A Very Basic App

1. Fire up Xcode and choose File, New Project. In the assistant, pick Single View Application and continue to the next step

I filled in these fields as follows:

- Product Name: PushChat (here: you)
- Company Identifier: com.hollance (here: jiapan)

You should choose a Product Name and Company Identifier that correspond to the App ID that you made earlier in the Provisioning Portal

i. open AppDelegate.m. Change the application:didFinishLaunchingWithOptions: method to look like this

```
// Let the device know we want to receive push notifications
if ([[[UIDevice currentDevice]systemVersion]floatValue] >= 8.0) {
    [[UIApplication sharedApplication]registerUserNotificationSettings:
    [UIUserNotificationSettings settingsForTypes:
    (UIUserNotificationTypeAlert|UIUserNotificationTypeBadge|UIUserNotificationTypeS
    [[UIApplication sharedApplication]registerForRemoteNotifications];
```

```
}else{
    [[UIApplication sharedApplication]registerForRemoteNotificationTypes:
    UIUserNotificationTypeBadge|UIUserNotificationTypeSound|UIUserNotificationTypeAl
}

return YES;
```

Build & Run the app. You need to do this on your device because the simulator does not support push notifications. Xcode should automatically have selected the new provisioning profile. If you get a code sign error, then make sure the proper profile is selected in the Code Signing build settings

ii. When the app starts and registers for push notifications, it shows a message to inform the user that this app wishes to send push notifications See pic informUserNotification

iii. The app asks this only once. If the user picks "OK", then we should be all set. However, if they choose "Don't Allow" then our app will never receive push notifications. The user can reverse their decision in the phone's Settings.

iv. The name of your app will be added to the phone's Settings, under Notifications. The user can enable or disable the notifications for your app here, including individual settings for badges, sounds, and alerts.

v. Your app can find out which types of push notifications are enabled through

```
if ([[[UIDevice currentDevice]systemVersion]floatValue] >= 8.0) {
    if ([[UIApplication sharedApplication]isRegisteredForRemoteNotifications]) {
        UIUserNotificationSettings *settings = [[UIApplication sharedApplication]
        currentUserNotificationSettings];
        NSLog(@"%@", settings);
    }else{
        NSLog(@"not registered for remote notifications");
    }
}else{
    UIRemoteNotificationType enabledTypes = [[UIApplication sharedApplication]
    enabledRemoteNotificationTypes];
    if (enabledTypes == UIRemoteNotificationTypeAlert) {
        NSLog(@"alert enabled");
    }
}
```

vi. There is one more thing you need to add in order to be able to receive push notifications. Add the following to AppDelegate.m

```
- (void)application:(UIApplication *)application didRegisterForRemoteNotificationsWi
    NSLog(@"My token is: %@", deviceToken);
}

- (void)application:(UIApplication *)application didFailToRegisterForRemoteNotificat
    NSLog(@"Failed to get token, error: %@", error);
}
```

When your app registers for remote (push) notifications, it tries to obtain a "device token". This is a 32-byte number that uniquely identifies your device

2. Sending Your First Push Notification

i. Download the SimplePush code and unzip it. You need to make some changes to simplepush.php

```
// Put your device token here (without spaces):
$deviceToken = '0f744707bebcf74f9b7c25d48e3358945f6aa01da5ddb387462c7eaf61bbad78';

// Put your private key's passphrase here:
$passphrase = 'pushchat';

// Put your alert message here:
$message = 'My first push notification!';
```

You should copy the device token from the app into the $deviceToken variable. Be sure to leave out the spaces and brackets; it should just be 64 hexadecimal characters. Put your private key' s

passphrase into $passphrase, and the text you wish to send in $message

ii. Copy your ck.pem file into the SimplePush folder. Remember, the ck.pem file contains both your certificate and the private key. Then open a Terminal and type:
`$ php simplepush.php`

iii. If all goes well, the script should say
`Connected to APNS`
`Message successfully delivered`

13. Viewing a Certificate Signing Request (CSR)
Part of configuring your App ID for push notifications is creating a certificate signing request, or CSR. Occasionally it's useful to look at the contents of a CSR, which you can do with the OpenSSL req command
`openssl req -noout -text -in PushChatTest.certSigningRequest`

4.39 AFNetworking

First import it from github

```
WKBaseService.h
/**
 *  发送http POST请求 处理成功和失败的回调
 *
 *  @param urlStr  请求地址
 *  @param body    请求参数
 *  @param success 成功回调
 *  @param failure 失败回调
 */
+ (void) postUrlStr:(NSString *)urlStr
               body:(NSDictionary *)body
            success:(void (^)(id))success
            failure:(void (^)(NSError *))failure;

WKBaseService.m
/**
 *  发送http POST请求 处理成功和失败的回调
 *
 *  @param urlStr  请求地址
 *  @param body    请求参数
 *  @param success 成功回调
 *  @param failure 失败回调
 */
+ (void) postUrlStr:(NSString *)urlStr
               body:(NSDictionary *)body
            success:(void (^)(id))success
            failure:(void (^)(NSError *))failure
{
    AFHTTPRequestOperationManager *manager = [AFHTTPRequestOperationManager manager];
    // 设置超时时间
    [manager.requestSerializer willChangeValueForKey:@"timeoutInterval"];
    manager.requestSerializer.timeoutInterval = 8.0f;
    [manager.requestSerializer didChangeValueForKey:@"timeoutInterval"];
    // 不让AFNetworking自动解析下载下来的内容
    manager.responseSerializer = [AFHTTPResponseSerializer serializer];

    [manager POST:urlStr parameters:body success:^(AFHTTPRequestOperation *operation, id respon
        if (success){
            DLog(@"success json = %@",operation.responseString);
            id result = nil;
            result = [NSJSONSerialization JSONObjectWithData:responseObject options:NSJSONReadi
```

```
            if (result == nil) {
                [[UIApplication sharedApplication].keyWindow makeToast:@"服务器君有异常啦" dura
            }
            success(result);
        }
    } failure:^(AFHTTPRequestOperation *operation, NSError *error) {
        if (failure) {
            DLog(@"error = %@",error);
            failure(error);
        }
    }];
}

WKMainService.h
/**
 *  获取可能疾病详细信息
 *
 *  @param diseaseId 可能疾病id
 *  @param success   成功回调
 *  @param failure   错误回调
 */
+ (void) requestPossibleDiseaseDetailsWithDiseaseId:(NSString *)diseaseId
                                            success:(void (^)(id result))success
                                            failure:(void (^)(id error))failure;

WKMainService.m
/**
 *  获取可能疾病详细信息
 *
 *  @param diseaseId 可能疾病id
 *  @param success   成功回调
 *  @param failure   错误回调
 */
+ (void) requestPossibleDiseaseDetailsWithDiseaseId:(NSString *)diseaseId
                                            success:(void (^)(id result))success
                                            failure:(void (^)(id error))failure{
    NSDictionary *dic = @{@"task":@"GET_PROPERTY", @"disease_id":diseaseId};
    NSString *url = [kServiceURL stringByAppendingString:@"smart_guide_treat"];

    [self postUrlStr:url body:dic success:success failure:failure];
}

WKSelfDiagnosisResultVC.m
- (void)requestPossibleDiseaseDetailData
{
    [self showHandleDataDefaultMessage];
    [WKMainService requestPossibleDiseaseDetailsWithDiseaseId:self.disease_id success:^(id resu

        NSDictionary *dict = result[@"data"];
        if (dict.count == 0) {
            [self showInfo:YES];
        }else{
            [self showInfo:NO];
        }

        if ([result[@"ret"] intValue] == 1) {
```

```
            for (int i=0; i<self.bodyArr.count; i++) {

                NSString *body = dict[self.bodyArr[i]];
                if (body != nil) {
                    WKSelfDiagnosisResultModel *model = [[WKSelfDiagnosisResultModel alloc]init
                    model.body = body;
                    model.title = self.titleArr[i];
                    [self.dataSource addObject:model];
                }
            }
            [self.tableView reloadData];
        } else {
            [self.view makeToast:result[@"msg"] duration:ToastTime position:CSToastPositionCent
        }
        [self hidenHandleDataDefaltMessage];
    } failure:^(id error) {
        [self showInfo:YES];
        [self.view makeToast:@"网络错误" duration:ToastTime position:CSToastPositionCenter];
        [self hidenHandleDataDefaltMessage];
    }];
}
```

4.40 JSONModel

For Pic WaitingInfo

```
if ([result[@"ret"] intValue] == 1) {
    NSArray *arr = result[@"data"];
    for (NSDictionary *dic in arr) {
        DLog(@"dic%@",dic[@"waitingList"]);

        NSMutableArray *mutableArr = [NSMutableArray array];
        for (NSDictionary *dict in dic[@"waitingList"]) {
            DLog(@"dict:%@", dict);

            WKWaitingInfoModel *model = [[WKWaitingInfoModel alloc]init];
            [model setValuesForKeysWithDictionary:dict];
            [mutableArr addObject:model];
        }

        WKWaitingModel *waitingModel = [[WKWaitingModel alloc]init];
        waitingModel.jzrName = dic[@"jzrName"];
        waitingModel.waitingList = mutableArr;
        [self.dataSource addObject:waitingModel];
    }
    [self.tableView reloadData];
}
```

Remember to add the following code to initialize dataSource

```
- (NSMutableArray *)dataSource {
    if (_dataSource == nil) {
        _dataSource = [NSMutableArray array];
    }
    return _dataSource;
}
```

For Pic selfTest

```
NSDictionary *dict = result[@"data"];
if (dict.count == 0) {
    [self showInfo:YES];
}else{
    [self showInfo:NO];
}

if ([result[@"ret"] intValue] == 1) {

    for (NSDictionary *dic in dict[@"disease_list"]) {
        DLog(@"dic values are %@", dic);
        WKSelfTestResultModel *model = [[WKSelfTestResultModel alloc]init];
        [model setValuesForKeysWithDictionary:dict];
    }
    for (NSDictionary *dic in dict[@"departs_list"]) {
        DLog(@"dic values are %@", dic);
        WKSelfTestModel *model = [[WKSelfTestModel alloc]init];
        [model setValuesForKeysWithDictionary:dict];
    }
}
```

4.41 Templates

1. Create a SVA, make the ViewController class root view controller(See \ref{template}). delete Main.storyboard and the corresponding row in info.plist
2. inside - (void)viewDidLoad method
3. make the ViewController class root view controller
   AppDelegate.m
   #import "ViewController.h"
   inside - (Bool)application:didFinishLaunchingWithOptions: method add
   self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]];

   ViewController *vController = [[ViewController alloc]init];
   self.window.backgroundColor = [UIColor whiteColor];

   self.window.rootViewController = vController;

   [self.window makeKeyAndVisible];

4.42 Install cocoapods

4.42.1 Install cocoapods

1. sudo gem update --system
2. sudo gem install cocoapods // this step failed try next
3. gem source -a http://rubygems.org/
4. y
5. sudo gem source -a http://rubygems.org/
6. sudo gem install cocoapods

This doesn't work on my computer

1. sudo gem update --system
2. gem source -r https://rubygems.org/
3. gem source -a http://rubygems.org/
4. sudo gem install cocoapods

5. pod setup

4.42.2 Setup project in xcode using cocoapods

First you have to have textmate installed ↦ download textmate and install it then go to TextMate preferences, click on Terminal icon then click on 'install' shell command if it's not already installed

1. Create a xcode project
2. Then close your xcode project
3. In terminal, navigate to the project you just created (See Pic podTest)
4. `pod init` let cocoapods have a slightly filled out version created for us **Or** `touch Podfile` start with an empty file
5. then `mate Podfile` to open Podfile in textMate
6. In textmate type
   `platform :ios, '7.0'`
   We don't know for sure AFNetworking pod specs is named, use the pod search command to figure it out.
   In command line type `pod search AFNetworking`
   Then in textmate type
   `pod 'AFNetworking'`
   Save it.
7. Go to cocoapods website and try the big search area, type any third party libraries and copy to clipboard and then paste to textmate in step 5.
8. In command line type `pod install`
9. If you want to update the pod specs simply `pod update` in the command line.If not working try remove Podfile.lock run `rm Podfile.lock` then `pod update`
10. When you want to include the added file's .h file, you need to import this way `#import <>` not the quotation way

Note: You have to be in the same file path as your project to do these thing or it may get messed up

4.43 extern

Using extern is the most frequent workaround for the lack of "class variables" (like those declared with static in Java) in Objective-C, It allows you to expand the scope in which you can reference a symbol beyond the compilation unit where it is declared

eg. create global notification names extern:

```
extern NSString* const XYYourNotification;
```

You then define the actual NSString* in your implementation

```
// MONClass.h
extern NSString* const MONClassDidCompleteRenderNotification;
// MONClass.m
NSString* const MONClassDidCompleteRenderNotification = @"MONClassDidCompleteRenderNotificatior
```

there is no case where the extern keyword affects visibility (public/protected/private/package). to use the symbol (e.g. the constant or C function), simply include the header it is declared in.

```
@interface MONClass : NSObject

extern const size_t MaximumThreads;

@end
```

has the same scope (global) and visibility (public) as:

```
@interface MONClass : NSObject

@end

extern const size_t MaximumThreads;
```

i recommend placing these in the same header as the interface, outside @interface/@end and @implementation/@end and prefixing the name with the class it is associated with, like so:

```
@interface MONClass : NSObject

@end

extern const size_t MONClassMaximumThreads;
// MONClass.m
const size_t MONClassMaximumThreads = 23;
```

and if you want that constant to be private, just declare and define it like this:

```
// MONClass.m
static const size_t MONClassMaximumThreads = 23;

@implementation MONClass

@end
```

5 Important notes

5.1 error-property-frame-not-found-on-object-of-type-uiview

```
(lldb) expr @import UIKit
(lldb) po _hosIntroLbl.frame.size.height
```

5.2 Working with stars

1. See pic staticStar for following code
   ```
   UIView *backView = [[UIView alloc]initWithFrame:CGRectMake(10, jobDescriptionMaxY+25, 64,
   backView.centerX = kScreenWidth/6 + kScreenWidth/6*i*2;
   [self.view addSubview:backView];
   int gap = 0;
   for (int i = 0; i<5; i++) {
       UIImageView *starImgView = [[UIImageView alloc]initWithFrame:CGRectMake(gap, 0, 12, 12
       UIImage *norImg = [UIImage imageNamed:@"icon_sta2r_evaluate"];
       UIImage *selImg = [UIImage imageNamed:@"icon_star_evaluate"];
       if (4 < i+1) {
           starImgView.image = norImg;
       }else{
           starImgView.image = selImg;
       }
       [backView addSubview:starImgView];
       gap += 13;
   }
   ```
2. See pic dynamicStar for following code
   ```
   for (int i=0; i<5; i++) {
       UIButton *btn = [UIButton buttonWithType:UIButtonTypeCustom];
       btn.frame = CGRectMake(label1X+i*20, 10, 20, 20);
       btn.tag = 11+i;

       if (i < 3) {
           [btn setImage:[UIImage imageNamed:@"wuxing_nor"] forState:UIControlStateNormal];
       } else {
           [btn setImage:[UIImage imageNamed:@"wuxing_sel"] forState:UIControlStateNormal];
       }
   ```

```
        [btn addTarget:self action:@selector(onSuduBtnClick:) forControlEvents:UIControlEventT
        [view addSubview:btn];
    }

    - (void) onSuduBtnClick:(UIButton *) sender {
        NSInteger tag = sender.tag;
        for (int i=11; i<16; i++) {

            UIButton *btn = (UIButton *)[self.view viewWithTag:i];
            if (i <= tag) {
                [btn setImage:[UIImage imageNamed:@"wuxing_nor"] forState:UIControlStateNormal
            } else {
                [btn setImage:[UIImage imageNamed:@"wuxing_sel"] forState:UIControlStateNormal
            }
        }
        _tag1 = tag-10;
        if (_tag1 == 1) {
            _suduState.text = @"很差";
        } else if (_tag1 == 2) {
            _suduState.text = @"不满意";
        } else if (_tag1 == 3) {
            _suduState.text = @"一般";
        } else if (_tag1 == 4) {
            _suduState.text = @"满意";
        } else if (_tag1 == 5) {
            _suduState.text = @"非常满意";
        }
    }
```

5.3 how to add a view to the navigation bar

```
UIWindow *window = [UIApplication sharedApplication].keyWindow;
[window addSubview:addView];
```

5.4 how-to-detect-touches-in-status-bar

5.5 Using Text Kit to Manage Text in Your iOS Apps

See project TextKitDemo

5.6 Make a call

```
[[UIApplication sharedApplication] openURL:[NSURL URLWithString:@"tel://010-62658746"]];
```

5.7 Support different iOS versions

In the place before **import** add following code

```
#ifndef NSFoundationVersionNumber_iOS_7_0
#define NSFoundationVersionNumber_iOS_7_0 1047.20
#endif

#ifndef NSFoundationVersionNumber_iOS_8_0
#define NSFoundationVersionNumber_iOS_8_0 1134.10
#endif

#ifndef iOS7
```

```
#define iOS7 (NSFoundationVersionNumber >= NSFoundationVersionNumber_iOS_7_0)
#endif

#ifndef iOS8
#define iOS8 (NSFoundationVersionNumber >= NSFoundationVersionNumber_iOS_8_0)
#endif
```

Then in other part of your code, for example

```
if (iOS8) {
    UIAlertController *alertController = [UIAlertController alertControllerWithTitle:@"请联系
客服010-62658746或在线反馈" message:nil preferredStyle:UIAlertControllerStyleActionSheet];
    [alertController.view setTintColor:[UIColor orangeColor]];

    // create the actions
    UIAlertAction *cancelAction = [UIAlertAction actionWithTitle:@"" style:UIAlertActionStyleCa
        DLog(@"alert's cancel action occured");
    }];
    UIAlertAction *feedbackAction = [UIAlertAction actionWithTitle:@"我要反馈" style:UIAlertAct
        DLog(@"我要反馈");
    }];
    UIAlertAction *callAction = [UIAlertAction actionWithTitle:@"拨打电话" style:UIAlertActionS
        DLog(@"拨打电话");
    }];
    UIAlertAction *action = [UIAlertAction actionWithTitle:@"取消" style:UIAlertActionStyleDest
        DLog(@"取消");
    }];

    // add the actions
    [alertController addAction:cancelAction];
    [alertController addAction:feedbackAction];
    [alertController addAction:callAction];
    [alertController addAction:action];

    [self presentViewController:alertController animated:YES completion:nil];
}else{
    UIActionSheet *actionSheet = [[UIActionSheet alloc]initWithTitle:@"请联系客服010-62658746或
在线反馈" delegate:self cancelButtonTitle:nil destructiveButtonTitle:@"我要反馈" otherButtonTit
打电话",@"取消", nil];
    actionSheet.destructiveButtonIndex = 2;

    [actionSheet showInView:self.view];
}
```

5.8 compiler-error-initializer-element-is-not-a-compile-time-constant

```
SEL selector = NSSelectorFromString(@"_alertController");
if ([actionSheet respondsToSelector:selector])
{
    UIAlertController *alertController = [actionSheet valueForKey:@"_alertController"];
    if ([alertController isKindOfClass:[UIAlertController class]])
    {
        alertController.view.tintColor = [UIColor blueColor];
    }
}
```

Change to

```
SEL selector = nil;
+ (void)initialize{
    if (!selector) {
        SEL selector = NSSelectorFromString(@"_alertController");
    }
}
if ([actionSheet respondsToSelector:selector]) {
    UIAlertController *alertController = [actionSheet valueForKey:@"_alertController"];
    if ([alertController isKindOfClass:[UIAlertController class]]) {
        alertController.view.tintColor = [UIColor redColor];
    }
}
```

5.9 You can do the following

```
[clockBtn addSubview:_redRemindLabel]
```

5.10 Project structure

See pic projectStructure1 and projectStructure2

5.11 removeFromSuperview method

```
for (UIView *subview in [self subviews]) {
    if ([subview isKindOfClass:[UIImageView class]]) {
        if (subview.tag == 1) {
            [subview removeFromSuperview]; // also add subview = nil; if doesn't work properly
            [self showInfo:YES];
        }
    }
}
```

sometimes you may have to replace the first self with `_tableView.tableFooterView`

5.12 boundingRectWithSize method

```
In (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:(NSIndexPath *)
indexPath method

CGSize size = [model.body boundingRectWithSize:CGSizeMake(kScreenWidth-30, 300) options:
NSStringDrawingTruncatesLastVisibleLine|NSStringDrawingUsesFontLeading|
NSStringDrawingUsesLineFragmentOrigin attributes:@{NSFontAttributeName:WKFontNor_15}
context:nil].size;
CGRect frame = contentLabel.frame;
frame.size.height = ceil(size.height);//ceil表示进1函数,为整数
contentLabel.frame = frame;
```

If the the content's height exceeds 300, then there will be some dots appended to the end of the content, if its height is less than 300 then the rest space will not show. 10 is used to add some other space to the row to allow other parts of the row to show properly

5.13 Difference between base sdk and deployment target

1. whats-the-meaning-of-base-sdk-ios-deployment-target-target-and-project-in-xc See pic SDKDevelopment-Timeline
    1. The base SDK is what you build your app against (i.e. include and library files and frameworks). As you say, it doesn't affect the deployment target, except that base sdk >= deployment target.

2. You specify build settings on 2 levels as each project can have multiple targets and you might not want the same settings for all targets. The project-level settings override the default settings and the target-level settings override the project-level settings.
3. Base SDK is the SDK you link against. Deployment Target is the minimum required iOS version you application needs to run. You can build an application with SDK 7 that runs under iOS 6. But then you have to take care to not use any function or method that is not available on iOS 6. If you do, your application will crash on iOS 6 as soon as this function is used

2. whats-the-real-difference-between-base-sdk-and-deployment-target-or-deployment
   1. Base SDK = What you're building against. If it's set to 4.1, then you build against the 4.1 SDK
   2. Deployment OS Version / Deployment Target = The lower end, or the oldest platform your app is aimed to support. If different than Base SDK or the SDK you build against, then you must do conditional checks so that older OS versions won't see a crash when your app calls something of newer ones, up to the Base SDK

5.14 For a info.plist on a lower version of ios see pic plistLowerVersion

5.15 App Transport Security has blocked a cleartext HTTP (http://) resource load since it is insecure.

You have to set the NSAllowsArbitraryLoads key to YES under NSAppTransportSecurity dictionary in your .plist file. Hope this helps! See pic appTransport

1. Request failed: unacceptable content-type: text/plain See pic serialization, you need to add text/plain to the end of acceptableContentTypes and the same goes for text/html

5.16 Prefix.pch

To create a prefix header (pch) file you'll have to do the following: See pic pchFilePath(add the name of the pch file as well)

1. Go to File ↦ New ↦ File ↦ Other (in iOS or OSX)
2. Select PCH File
3. Give it a name and store it somewhere in your project
4. Go to your project's target ↦ Build Settings
5. Search for prefix header
6. Under Apple LLVM - Language you will find the Prefix Header key
7. Add the path to your pch file
8. Clean your project and build

Every .pch file should have the following code and also pay attention to the order of the imported file(if you messed it up, then it won't work). If you moved the project to a different place be sure to change the pch path in the build settings

```
#import <Availability.h>  //optional

#ifndef __IPHONE_5_0
#warning "This project uses features only available in iOS SDK 5.0 and later."
#endif

#ifdef __OBJC__
    #import <UIKit/UIKit.h>
    #import <Foundation/Foundation.h>

    #import "AppConfig.h"  // import self created file like so, angle bracket will not work
    #import "BaseFramework.h"

#endif

/Users/A053/Library/Developer/Xcode/DerivedData
```

5.17 Working with app images and icons

Icon and Image Sizes
A Xcode plugin to automatically generate @2x, @1x image from @3x image for you, or upscale to @3x from @2x
See pic changeableImageSizes

1. xcode-launch-images-and-full-screen-issues See pic launchImage
   See pic launchImageSizesAs you can see in the attribute inspector, you can choose which device launch image you want and it will give you that. Check by clicking the place holder what image size or resolution it requires and then you just drag and drop those images from finder. So you can use this method to fix black screen issue.

5.18 why-do-i-need-1x-2x-and-3x-ios-images

1. In general - for an image too look perfect, you want to avoid both downscaling and upscaling
2. You can always go with only @2x or @3x images and add other scales only if you see visual problems
3. Using higher resolution won't improve downscaling. High resolutions are used only to avoid upscaling. Downscaling from a high scale (e.g. @100x) to @1x won't create better results than downscaling from @3x
4. See pic iosImages

5.19 Working with Localized strings

1. Note: Localizable.strings is the default filename iOS uses for localized text. Resist the urge to name the file something else, otherwise you will have to type the name of your .strings file every time you reference a localized string
2. You need to follow a specific, but fairly simple, format like this
   `"KEY" = "CONTENT";`
3. Key/content pairs can also contain format strings:
   `"Yesterday you sold %@ apps" = "Yesterday you sold %@ apps";`

5.20 instance vs id

See pic instanceTypeId

5.21 video-tutorial-beginning-auto-layout

video-tutorial-beginning-auto-layout
using-auto-layout-in-uitableview-for-dynamic-cell-layouts-variable-row-heights
self-sizing-cells

5.22 Setter & getter

1. One way:
   `.h`
   ```
   @interface RootViewController : UITableViewController {
   ```

```
    NSArray * _sushiTypes;
    NSString * _lastSushiSelected;
}
- (NSArray *)sushiTypes;
- (void)setSushiTypes:(NSArray *)sushiTypes;

- (NSString *)lastSushiSelected;
- (void)setLastSushiSelected:(NSString *)lastSushiSelected;
@end

.m
- (NSArray *)sushiTypes{
    return _sushiTypes;
}

- (void)setSushiTypes:(NSArray *)sushiTypes{
    [sushiTypes retain];
    [_sushiTypes release];
    _sushiTypes = sushiTypes;
}

- (NSString *)lastSushiSelected{
    return _lastSushiSelected;
}

- (void)setLastSushiSelected:(NSString *)lastSushiSelected{
    [lastSushiSelected retain];
    [_lastSushiSelected release];
    _lastSushiSelected = lastSushiSelected;
}
```

2. Another way:

```
.h
@interface RootViewController : UITableViewController {
    NSArray * _sushiTypes;
    NSString * _lastSushiSelected;
}
@property (nonatomic, retain) NSArray * sushiTypes;
@property (nonatomic, retain) NSString * lastSushiSelected;

@end

.m
@implementation RootViewController
@synthesize sushiTypes = _sushiTypes;
@synthesize lastSushiSelected = _lastSushiSelected;
```

5.23 Add Linker Flag

See Pic linkerFlag

5.24 how to check if array or dictionary or string is null or empty

array: how-to-check-if-array-is-null-or-empty

```
if (!array || !array.count){
  ...
}
```

That checks if array is not nil, and if not - check if it is not empty.
If the array is nil, it will be 0 as well, as nil maps to 0; therefore checking whether the array exists is unnecessary.

dictionary: `[response[@"data"] count] == 0`

That checks if an dictionary is empty

string:

```
if (waitLM.waitDateTime == nil) {
    _waitDateTimeLabel.text = @"some string";
}
if ([_waitStateString isEqualToString:@"0"])
```

5.25 Status bar and navigation bar overlaps my view

status-bar-and-navigation-bar-appear-over-my-views-bounds-in-ios-7
Two ways to handle this

1. `self.navigationController.navigationBar.translucent = NO;` This will fix the view from being framed underneath the navigation bar and status bar. You can set this in the navigationController combined with tabbar controller, such as `WKMainTabBarController.m` in Doctor edition (This may cause the nav bar's bar tint color not showing and back bar button item not showing as well so you need to be cautious about this and you may need to use third party library. You may use ↦ `[self.tabBarItem setTitlePositionAdjustment:UIOffsetMake(0, -2)];`)
2. In your apps plist file add a row, call it "View controller-based status bar appearance" and set it to NO. See Pic BasedStatusBarAppearance
3. **Storyboard based app See Pic ExtendEdges**
   This can also be accomplished programmatically through the usage of -[UIViewController edgesForExtendedLayout].
   `[self setEdgesForExtendedLayout:UIRectEdgeNone];`

5.26 NULL and nil

```
UIImagePickerController *picker = [[UIImagePickerController alloc]init];
picker.delegate = self;
picker.allowsEditing = YES;
picker.sourceType = UIImagePickerControllerSourceTypeCamera;

[self presentViewController:picker animated:YES completion:NULL];
```

```
if (![UIImagePickerController isSourceTypeAvailable:UIImagePickerControllerSour
    UIAlertView *myAlertView = [[UIAlertView alloc]initWithTitle:@"Error" messa
    @"Device has no camera" delegate:nil cancelButtonTitle:@"OK" otherButtonTit
}
```

5.27 To copy paste, choose either copy/paste or option+drag

5.28 Snippets

1. In Xcode, from the menu bar browse to View > Utilities > Show Code Snippet Library or press control + option + cmd + 2 to expose the code snippet library in the bottom right
2. Start by typing the code you want generated when you call your snippet
3. Select all the code to be added to the snippet, then click and hold for a second (the text will then become draggable). Once the text is draggable, go ahead and drag it onto the Snippets Library
4. Delete a code snippet, by selecting it and then press delete key on the keyboard
5. Input bubble
   (a) Open the snippet be selecting it in the Snippets Library and waiting for the popover to show up, then select Edit
   (b) To create an input bubble, you need to wrap the input text in angle brackets and pound signs. Here is an example `<#input text#>`
6. Created snippets
   mark for `#pragma mark - <#input text#>`
   some

5.29 Standford University ios 7

1. Primitive don't need to be declared strong or weak, just nonatomic is enough, it's not stored in the heap

5.30 Blocks

1. A block declaration follows the next syntax pattern:
   `ReturnType(^blockName)(Parameters)`
   When you pass no parameters to block, then the void keyword should be always set. A special characteristic that the block declaration has, is that the parameter names can be omitted, and just keep the parameter types.
2. Let's focus now on the block definition. Here is the pattern been followed:
   ```
   ^(Parameters){
      ...block body...
      return ReturnValue(or nothing if the block return type is void)
   };
   ```
   As you see, no block name exists here. In block definition, `parameter names are mandator` unlike the block declaration where they are optional. Especially note that after the block body closing, the semicolon ; is added, as the whole block is considered as a variable.

3. Let's see now a couple of examples on how you can assign block results to a block variable:
```
int (^howMany)(int, int) =^(int a, int b){
    return a + b;
};
```
In this example, the parameter names have been omitted in the declaration, but they mandatorily exist on the definition.
4. Let me clarify that it's not necessary the declaration and the definition to take place at the same time. For example:
```
// Declare a block variable.
void (^xyz)(void);

//Some other code...

//Define the block
xyz = ^(void){
    NSLog(@"What's up, Doc?");
};
```
5. Even more, a block can be declared as a class member variable, just like any other member variables. For example, in the private section of the interface you can write this..
```
@interface ViewController ()
@property (nonatomic, strong) NSString *(^blockAsAMemberVar)(void);
@end
```
... and then inside the viewDidLoad add this:
```
_blockAsAMemberVar = ^(void){
    return @"This block is declared as a member variable!";
};
```
6. A block variable is called just like a C function, using its name and providing any required arguments. Here is an example that clears it:
```
int (^howMany)(int, int) =^(int a, int b){
    return a + b;
};
NSLog(@"a + b = %d", howMany(5,10));
```

Let's see one more example
```
NSDate *(^today)(void);

today = ^(void){
    return [NSDate date];
};
NSLog(@"today's date is %@", today());
```

I hope this one won't confuse you
```
float results = ^(float value1, float value2, float value3){
    return value1*value2*value3;
```

```
}(1.2, 3.4, 5.6);
NSLog(@"%f", results);
```

This is a more direct approach, as using the same command we define the block, we pass the parameter values and we assign the results to a variable.

One last example, using a variable existing out of the block definition:

```
int factor = 5;
int (^newResult)(void) = ^(void){
    return factor*10;
};
NSLog(@"variable existing out of the block definiton: %d", newResult());
```

5.30.1 The __block Storage Type

The code existing in a block can access a variable that belongs to the scope of the method that the block is defined in, but by default this access is read-only. That means that the code of the block cannot change a variable's value from the "outer world", a fact that's useful for protecting us by doing any unwanted modifications, however the ability to assign new values in such variables is many times necessary. See project BlockStorageType.

```
@interface ViewController ()

- (void)testBlockStorageType;

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    [self testBlockStorageType];
}

- (void)testBlockStorageType{
    __block int someValue = 10;
    int (^myOperation)(void) = ^(void){
        someValue += 5;

        return someValue + 10;
    };

    NSLog(@"%d", myOperation());
}
@end
```

5.30.2 Blocks As Completion Handlers

1. A completion handler is the way (technique) for implementing **callback** functionality using blocks. In the introduction of this tutorial I mentioned that blocks can be used in place of delegate methods that work as callbacks, and that's the way this can be done; through completion handlers.
2. A completion handler is nothing more than a simple block declaration passed as a parameter to a method that needs to make a callback at a later time.
3. One important notice before moving forward is that you should never forget that the completion handler should always be the last parameter in a method. A method can have as many arguments as you want, but always have the completion handler as the last argument in the parameters list.
4. Here is the declaration pattern of a method which contains a completion handler to make callbacks:

```
-(returnType)methodNameWithParams:(parameterType)parameterName ...<more par
andCompletionHandler:(void(^)(<any block params>))completionHandler;
```

   On its implementation

```
-(returnType)methodNameWithParams:(parameterType)parameterName ...<more par
andCompletionHandler:(void(^)(<any block params>))completionHandler{
    ...
    ...

    // When the callback should happen:
    completionHandler(<any required parameters>);
}
```

   The completion handler block is always defined when the method is invoked:

```
[self methodNameWithParams:parameter1 ...<more params>...
andCompletionHandler:^(<any block params>){
    // The completion handler code is added here after the method has finis
    // and has made a callback.
}];
```

5. A couple of typical examples of using completion handlers from framework methods are:
   1. When showing a modal view controller, if you want to handle something after the view controller has been presented:

```
[self presentViewController:viewController animated:YES completion:^{
    NSLog(@"View Controller was presented...");

    // Other code related to view controller presentation...
}];
```

   2. When performing UIView animations:

```
[UIView animateWithDuration:0.5 animations:^{
    // Animation-related code here...
    [self.view setAlpha:0.5];
} completion:^(BOOL finished) {
    // Any completion handler related code here...
    NSLog(@"Animation is over.");
```

```
}];
```

6. Let's see a quite simple example on how we can create a method with a completion handler, how we call it and how to deal with the completion handler.
   To keep it simple, we will implement a method which adds two integers, and instead of returning the result to the caller, it will send it back using the completion handler.
   1. On the ViewController.m file, go to the private section of the interface and make the following method declaration:
      ```
      @interface ViewController ()
      - (void)addNumber:(int)number1 withNumber:(int)number2 andCompletionHandl
      (void(^)(int result))completionHandler;
      @end
      ```
      As you see, we declare a method with three parameters. The first two are the numbers we want to add, and the last one is the completion handler.
   2. Let's keep going with its implementation, which is going to be really easy:
      ```
      - (void)addNumber:(int)number1 withNumber:(int)number2 andCompletionHandl
      (void (^)(int))completionHandler{
          int result = number1 + number2;
          completionHandler(result);
      }
      ```
   3. Let's invoke it now in the viewDidLoad method as follows:
      ```
      [self addNumber:5 withNumber:7 andCompletionHandler:^(int result) {
          // We just log the result, no need to do anything else.
          NSLog(@"The result is %d", result);
      }];
      ```
      When invoking the method, we pass the values 5 and 7 as parameters, and we define the completion handler block here. It's obvious that is much more helpful to handle any callback actions here, than creating protocols, delegate methods, and so on, scattered all over the place.
7. Completion Handlers: A Practical Example. See project completionHandler.

5.30.3 Typically, blocks are used for one of the following:

- Completion Handlers
- Error Handlers
- Notification Handlers
- Enumeration
- View animation and transitions
- sorting

The next two examples see project OldSimpleCounter

5.30.4 A Simple Completion Handler Without BLocks

Create a SVA, creating a new NSObject class and title it oldSimpleCounter.

```
oldSimpleCounter.h
@interface oldSimpleCounter : NSObject
```

```
- (void)countToTenThousandAndNotifyObserver:(id)object withSelector:(SEL)select
@end


oldSimpleCounter.m
@implementation oldSimpleCounter
- (void)countToTenThousandAndNotifyObserver:(id)object withSelector:(SEL)select
    int x = 1;
    while (x < 10) {
        NSLog(@"%i", x);
        x++;
    }

    [object performSelector:selector withObject:object];
}

@end


ViewController.m
#import "ViewController.h"
#import "oldSimpleCounter.h"
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    [self startCountingToTenThousand];
}

- (void)startCountingToTenThousand{
    oldSimpleCounter *counter = [[oldSimpleCounter alloc]init];
    [counter countToTenThousandAndNotifyObserver:self withSelector:
    @selector(tenThousandCounts:)];
}

- (void)tenThousandCounts:(id)sender{
    NSLog(@"Ten thousand counts finished");
}
```

5.30.5 A Simple Completion Handler Using Block Approach

Create a SVA, create a new NSObject class and title it NewSimpleCounter.

```
NewSimpleCounter.h
@interface NewSimpleCounter : NSObject
```

```
- (void)countToTenThousandAndReturnCompletionBLock:(void (^)(BOOL))completed;
@end


NewSimpleCounter.m
@implementation NewSimpleCounter
- (void)countToTenThousandAndReturnCompletionBLock:(void (^)(BOOL))completed{
    int i = 1;
    while (i<1001) {
        NSLog(@"%i", i);
        i++;
    }

    completed(YES);
}

@end


ViewController.m
#import "ViewController.h"
#import "NewSimpleCounter.h"
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    NewSimpleCounter *counter = [[NewSimpleCounter alloc]init];
    [counter countToTenThousandAndReturnCompletionBLock:^(BOOL completed) {
        if (completed) {
            NSLog(@"Ten thousand counts finished");
        }
    }];
}
```

5.31 See those plugins installed on Mac

```
/Users/A053/Library/Application Support/Developer/Shared/Xcode/Plug-ins
RTImageAssets.xcplugin
VVDocumenter-Xcode.xcplugin
```

5.32 how to fold methods xcode

1. To fold/unfold current methods or if structures use command-alt-left arrow/right arrow
2. First, place the cursor inside a method/function, but outside any block. Then, use Shift Option Command + ← to fold it. Quickly fold all methods and functions with Shift

Option Command + ←
3. Unfold them all with Shift Option Command + →

5.33 Accessing provisioning profile files

```
/Users/A053/Library/MobileDevice/Provisioning Profiles
```

5.34 raywenderlich

1. http://www.raywenderlich.com/store/ios-9-by-tutorials
2. The iOS Apprentice Fourth Edition, Beginning iOS Development with Swift 2 :This is a sister book to the Swift Apprentice; the Swift Apprentice focuses on the Swift 2 language itself, while the iOS Apprentice focuses on making apps
3. iOS 9 by Tutorials: Learning the new iOS 9 APIs with Swift 2. This book is for intermediate iOS developers who already know the basics of iOS and Swift 2 development but want to learn the new APIs introduced in iOS 9.

5.35 About App store

1. But keep in mind, Apple doesn' t want you accessing private API properties in your App Store apps.

6 Things learned

6.1 NSMutableParagraphStyle and boundingRectWithSize

See pic NSMutableParagraphStyleFirst and NSMutableParagraphStyleSecond

```
@interface WKDynamicsDetail (){
    UIScrollView *_scrollView;
    UILabel *_contentLbl;
}

@end

@implementation WKDynamicsDetail

- (void)viewDidLoad{
    [super viewDidLoad];
    self.title = @"动态详情";

    [self buildSubviews];
}

- (void)buildSubviews{
    _scrollView = [[UIScrollView alloc]initWithFrame:CGRectZero];
    _scrollView.backgroundColor = [UIColor whiteColor];
```

```
    _scrollView.showsVerticalScrollIndicator = YES;
    [self.view addSubview:_scrollView];

    UILabel *headlineLbl = [[UILabel alloc]initWithFrame:CGRectMake(10, 0, kScr
    headlineLbl.font = [UIFont systemFontOfSize:20];
    headlineLbl.textColor = WKColorBlack;
    headlineLbl.text = @"2月10日李医生停诊通知";
    [_scrollView addSubview:headlineLbl];

    CGFloat headlineLblMaxY = CGRectGetMaxY(headlineLbl.frame);
    UILabel *dateLbl = [[UILabel alloc]initWithFrame:CGRectMake(10, headlineLbl
    dateLbl.font = WKFontNor_12;
    dateLbl.textColor = WKColorGrayLight;
    dateLbl.text = @"2015/12/10";
    [_scrollView addSubview:dateLbl];

    CGFloat dateLblMaxY = CGRectGetMaxY(dateLbl.frame);
    _contentLbl = [[UILabel alloc]initWithFrame:CGRectMake(10, dateLblMaxY+10,
    _contentLbl.font = WKFontNor_15;
    _contentLbl.textColor = WKColorGray;
    _contentLbl.numberOfLines = 0;
    _contentLbl.contentMode = UIViewContentModeTop;
    _contentLbl.lineBreakMode = NSLineBreakByWordWrapping;

    NSString *string = @"你好，床位已经安排好请您于5月10准时入住，哈哈哈哈看
价格行情卡我的家哈你好，床位已经安排好请您于5月10准时入住，哈哈哈哈看价格行情
卡我的家哈你好，床位已经安排好请您于5月10准时入住，哈哈哈哈看价格行情卡我的家
哈你好，床位已经安排好请您于5月10准时入住，哈哈哈哈看价格行情卡我的家哈你好，
床位已经安排好请您于5月10准时入住哈哈哈哈看价格行情卡我的家哈你好，床位已经
安排好请您于5月10准时入住，哈哈哈哈看价格行情卡我的家哈你好床位已经安排好请您
于5月10准时入住，哈哈哈哈看价格行情卡我的家哈你好，床位已经安排好请您于5月10准
时入住，哈哈哈哈看价格行情卡我的家哈你好，床位已经安排好请您于5月10准时入住
哈哈哈哈看价格行情卡我的家哈你好，床位已经安排好请您于5月10准时入住，哈哈哈
哈看价格行情卡我的家哈你好床位已经安排好请您于5月10准时入住，哈哈哈哈看价格行
情卡我的家哈你好，床位已经安排好请您于5月10准时入住，哈哈哈哈看价格行情卡我的
家哈你好，床位已经安排好请您于5月10准时入住哈哈哈哈看价格行情卡我的家哈你好，
床位已经安排好请您于5月10准时入住，哈哈哈哈看价格行情卡我的家哈你好床位已经安
排好请您于5月10准时入住，哈哈哈哈看价格行情卡我的家哈你好，床位已经安排好请您
于5月10准时入住，哈哈哈哈看价格行情卡我的家哈你好，床位已经安排好请您于5月10准
时入住哈哈哈哈看价格行情卡我的家哈你好，床位已经安排好请您于5月10准时入住，哈
哈哈哈看价格行情卡我的家哈你好结束。";
    NSMutableAttributedString *mutableAttrStr = [[NSMutableAttributedString all
    NSMutableParagraphStyle *mutableParaStyle = [[NSMutableParagraphStyle alloc
    [mutableParaStyle setLineSpacing:8];
    [mutableAttrStr addAttribute:NSParagraphStyleAttributeName value:mutablePar
```

```
    _contentLbl.attributedText = mutableAttrStr;
    [_scrollView addSubview:_contentLbl];
}

- (void)viewWillLayoutSubviews{
    _scrollView.frame = CGRectMake(0, 0, kScreenWidth, kScreenHeight);
    CGSize size = [_contentLbl.text boundingRectWithSize:CGSizeMake(kScreenWidt
    _scrollView.contentSize = CGSizeMake(kScreenWidth, size.height + 400);

    CGRect rect = _contentLbl.frame;
    rect.size.height = ceil(size.height);
    _contentLbl.frame = rect;
    [_contentLbl sizeToFit];
}
```

7 Things about Mac computer

7.1 Mac bookmarks

1. http://product.dangdang.com/1233407239.html#filtertype:1
2. http://bj.58.com/shuma/24626420581690x.shtml?psid=128679169190378254960467 entinfo=24626420581690_0&iuType=p_0&PGTID=0d3036d0-0000-117a-5abd-64d9a7b9e ClickID=1
3. http://52ss.xyz
4. http://www.code4app.com/index.php
5. http://www.cocoachina.com
6. http://www.12306.cn/mormhweb/zxdt/201305/t20130516_600.html
7. http://huoche.tianqi.com/dingpiaorili/
8. https://github.com/AFNetworking/AFNetworking
9. http://bug.120.io:88/Public/?c=index&a=login&bgurl=http%3A%2F%2Fbug.120.io% 3A88%2FPublic%2F%3Fc%3Dbug%26a%3Dindex%26projectid%3D0%26bugopt%3D1%26statu 3D0%26priority%3D0%26verid%3D0%26moduleid%3D0%26userid%3D0%26usergroup%3D0% 26usertype%3Dcreateuid%26lastdate%3D%26searchkey%3D
10. http://boss.365pad.net/login
11. http://bbs.zol.com.cn/nbbbs/d544_8235.html
12. https://detail.tmall.com/item.htm?spm=a230r.1.14.6.9dXBgy&id=520874434029& cm_id=140105335569ed55e27b&abbucket=12&sku_properties=5919063:6536025
13. https://item.taobao.com/item.htm?spm=a230r.1.14.1.FKvKCY&id=41992133165&ns= 1&abbucket=12#detail
14. http://www.mono-project.com/docs/database-access/providers/sqlite/

7.2 Command + option + esc to force quit

7.3 How to Stop Photos App Launching Automatically in OS X

http://osxdaily.com/2015/05/31/stop-photos-opening-automatically-mac-os-x/

1. Connect the iPhone, camera, SD card, etc to the Mac and let Photos app launch itself as usual
2. Under the "Import" tab of Photos app, look in the upper left corner to find the device name, this will indicate which hardware will no longer automatically activate Photos app
3. Click the checkbox so that "Open Photos for this device" is no longer selected
4. Quit out of Photos app, the change is immediate for that device - this can be switched back at any time in the same OS X Photos app screen

7.4 The top most bar is called menu bar in Mac OSX

7.5 how-to-hide-or-remove-icons-from-the-mac-desktop

Click on the Finder menu and select preferences, click on the general tab. See pic finderPreferences

7.6 How to silence the startup chime on a Mac

https://github.com/teored90/nobootsound

7.7 Use Texshop on Mac

1. Here to download click on the quick installation instruction
2. chinese typing on mac
   ```
   % !TEX encoding = UTF-8 Unicode
   \usepackage{CJK}
   \begin{CJK}{UTF8}{gbsn}
   \end{CJK}
   ```

   Unicode can not be omitted, you can choose from `Macros Encoding` then UTF8 encoding
3. `!TEX root = C#.tex  prepend with % sign`
   for managing a large document Warning: Here you should't put the percent sign in front or `Use one window` option can't be used
4. Go to TexShop preferences, in the source tab select Tags menu in Menubar to show the Tags menu and see those section or subsection there

7.8 how-to-move-up-a-directory-with-terminal-in-osx

See pic lsDirectory
1. cd ..
2. cd /usr

7.9 how to change default apps for opening a specific file type os x

See pic defaultFileOpenApp

7.10 Turn off auto updates OS X Yosemite

https://discussions.apple.com/thread/6663064?start=0&tstart=0
GO to System Preferences ↦ App Store ↦ Uncheck 'Automatically Check for Updates' and anything else in there you don't want

7.11 how-to-fully-reset-safari-on-your-mac

1.

7.12 How to use iPhone, iPad earphones

Play/Pause: A single tap on the middle (depressed) region of the headphone controller is all it takes to Play or Pause your music, video or podcast

Rewind: Similarly, to rewind a song tap the Play/Pause button three times and hold the control down for the track to skip back

Skip to next track: Bored of the song that's playing? Just tap the Play/Pause button twice to skip to the beginning of the next track

Skip to previous track: You can also skip to a previous track – all you need to do is tap the Play/Pause button three times

Activate Siri: Press and hold the Play/Pause button and you'll activate Apple's Siri

Take photo: Once you' ve launched the Camera app you can tap the + (volume up) button to take a photo

Reject incoming call: Receiving a call and unable to take it? Just press and hold the Play/Pause button for two seconds until you hear two low beeps. The call will be sent to voicemail

Answer and/or hang up incoming call: If someone calls you, answer the call by tapping the Play/Pause button. Once the call completes, tap the button once again to hang up.

Switch calls: If you receive a call while you are talking to someone else, press the Play/Pause button once to switch to the next call. Press it again to end the second call and return to the original

1. xcode-asking-username-password-everytime-i-compile-to-device For solution go to Keychain Access, and in the top left navigation, move your Developer certificate from "System" to "Login".
2. Set background images in code
   You can also set background images in code by control-dragging your button to the `@interface` section of the main view controller as an `IBOutlet`:
   `@property (weak, nonatomic) IBOutlet UIButton *fullSizeImage;`
   then calling the `setBackgroundImage` method inside `viewDidLoad` for each state:

```
- (void)viewDidLoad
{
    [super viewDidLoad];
    [self.fullSizeImage setBackgroundImage:[UIImage imageNamed:@"button.png
    UIControlStateNormal];
    [self.fullSizeImage setBackgroundImage:[UIImage imageNamed:@"buttonHigh
    forState:UIControlStateHighlighted];
}

Another way:
UIImage *image1 = [UIImage imageNamed:@"navigationbar_white.png"];
```

3. Error: Control reaches end of non-void function

```
-(id)initWithCoder:(NSCoder *)aDecoder{
    //our custom CALayer drawing will go here
    self=[super initWithCoder:aDecoder];

    //Custom drawing methods
    if(self)
    {
        [self drawButton];
        [self drawInnerGlow];
        [self drawBackgroundLayer];
        [self drawHighlightBackgroundLayer];
    }

//    return self;
}
```

   You have to add `return self;`
4. Things learned :
   - UIViews are sent initWithFrame, UIViewControllers are sent initWithcoder since they conform to NSCoding and all other objects are just sent init.
   - All outlets and action connections are established (Your IBOUtlets and IBActions) using setValue:forKey: and setTarget:action: respectively.
   - Also remember that a storyboard is just a collection of nib files with some metadata describing how they are related.
5. Thread 1:breakpoint 1.1: You set a breakpoint accidently
6. **A common mistake**: If you made a mistake in connecting the elements or naming

your properties and you want to re-do it, you can delete the property in the .h file but you also have to break the connection by going to your storyboard, right-clicking the element and clicking the "x" beside the outlet reference.

7. xcode-sentestingkit-not-found: Checking the Build Phases tab and looking in the "Compile Sources" disclosure
8. Whenever using CALayer, remember to link the QuartzCore framework in the Build phases tab of the project manager and choose `Link Binary With Libraries`. You'll also need to import its header file in your subclass using this directive `#import <QuartzCore` inside a .h file
9. In iOS 9, a storyboard can be split up into multiple smaller storyboards. Apple refers to this process as **refactoring**.
10. In Storyboard Reference you have the `Storyboard ID` under `show the Identity inspect` and
    `Storyboard Referenced ID Bundle` under `show the attribute inspector`
11. `navigationController` is declared in `UINavigationController.h` but has the `UIViewCont` reference as following:
    ```
    @interface UIViewController (UINavigationControllerItem)

    @property(nonatomic,readonly,strong) UINavigationItem *navigationItem; // C
    on-demand so that a view controller may customize its navigation appearance
    @property(nonatomic) BOOL hidesBottomBarWhenPushed; // If YES, then when th
    controller is pushed into a controller hierarchy with a bottom bar (like a
    the bottom bar will slide out. Default is NO.
    @property(nullable, nonatomic,readonly,strong) UINavigationController
    *navigationController; // If this view controller has been pushed onto a na
    controller, return it.

    @end
    ```
    If it(navigationController) has a UIViewController class referenceSee Pic UIViewControllerRef, then any subclass of UIViewController can access it use the dot syntax and through it to access any other properties or methods(including the referenced class↦ UIViewController properties and methods) in the class it declared in
12. There are four views : go to second view (viewController.count is 4 not 3)
    ```
    [self.navigationController popToViewController:self.navigationController.
    viewControllers[1] animated:YES];
    ```
13. how-to-create-an-empty-application-in-xcode-6-without-storyboard maybe can be applied to xcode 7
    1. Create an Single View Application with XCode6
    2. Remove Main.storyboard and LaunchScreen.xib (select them, right-click, and choose to either remove them from the project, or delete them completely)
    3. Remove "Main storyboard file base name" and "Launch screen interface file base name" entries in Info.plist file.
    4. Open AppDelegate.m, and edit applicationDidFinishLaunchingWithOptions so that it looks like this:
       ```
       - (BOOL)application:(UIApplication *)application didFinishLaunchingWithOp
       ```

```
{
    self.window = [[UIWindow alloc] initWithFrame:[[UIScreen mainScreen]
    // Override point for customization after application launch.
    self.window.backgroundColor = [UIColor whiteColor];
    [self.window makeKeyAndVisible];
    return YES;
}
```

It seems that you also have to set its rootViewController or it won't work

14. You also have to manually set the following inside a className.m file (when you are creating empty .m or .h file)

```
@implementation className

@end
```

15. What-is-the-difference-between-strong-retain-nonatomic-etc-in-the-Objective-C-iOS-prope

Roman Hernandez, Computer Science student at The University of Texas at Austin.

1. Strong and Weak help with retain-release cycles (RRC), a form of memory leak. iOS uses something called Automatic Reference Countin (ARC) to know when an object is in use and should be kept in memory, or is no longer in use and should be deleted to gain back resources.
2. Issues arise when you have two objects that hold references to each other. Because object A holds a reference to object B, and B to A, the reference count for both A and B will never be 0, this A and B will always be in memory. It's also possible that there are no other objects holding references to A or B, so we've just created a memory leak.
3. Getting back to Strong and Weak, these keywords are used to "denote ownership", if you will. They help you eliminate retain-release cycles by limiting what objects increment the reference count for another object.
4. A strong property is one where you increment the reference count of the object. If object A has a strong reference to B, and no other object is referencing B, B has count 1 (A owns, or needs to exist B). Now, if B wants to have a reference to A, we would want to use a weak reference. Weak references don't increment the reference count of the object. So in this particular case, if A has no other objects referencing it but B, A's count would be 0 given B's weak reference.

Aritra Das, iOS Developer

1. List of attributes of @property :
   - atomic.
   - nonatomic.
   - retain.
   - copy.
   - readonly.
   - readwrite.
   - assign.
   - strong.

atomic : It is the default behaviour. If an object is declared as atomic then it becomes thread-safe. Thread-safe means, at a time only one thread of a particular instance

of that class can have the control over that object.

nonatomic: It is not thread-safe. You can use the nonatomic property attribute to specify that synthesized accessors simply set or return a value directly, with no guarantees about what happens if that same value is accessed simultaneously from different threads. For this reason, it's faster to access a nonatomic property than an atomic one.

retain: is required when the attribute is a pointer to an object.

copy: If you use copy, you can't use retain. Using copy instance of the class will contain its own copy.

assign: will generate a setter which assigns the value to the instance variable directly, rather than copying or retaining it. This is best for primitive types like NSInteger and CGFloat, or objects you don't directly own, such as delegates.

strong: is a replacement for retain.

unsafe_unretained: There are a few classes in Cocoa and Cocoa Touch that don't yet support weak references, which means you can't declare a weak property or weak local variable to keep track of them. These classes include NSTextView, NSFont and NSColorSpace,etc. If you need to use a weak reference to one of these classes, you must use an unsafe reference.

An unsafe reference is similar to a weak reference in that it doesn't keep its related object alive, but it won't be set to nil if the destination object is deallocated.

```
@property (unsafe_unretained) NSObject *unsafeProperty;
```

16. Angle bracket and the content within it will not show in Calibre edit mode
17. **CGPointMake CGSizeMake CGRectMake** corresponding to CGPoint, CGSize, CGRect class

    CGPoint: It represents a point in view

    ```
    CGPoint myCGPoint = CGPointMake(x,y)
    ```

    CGSize: It represents width and height of the view

    ```
    CGSize myCGSize = CGSizeMake(width, height)
    ```

    CGRect: We can specify a rectangle with x, y, width and height **Like bounds**

    ```
    CGRect myCGRect = CGRectMake(x, y, width, height)
    ```

    ```
    CGPointMake(0,0)    CGSizeMake(0,0)    CGRectMake(0,0,0,0)

    [self.scrollView setContentOffset:CGPointMake(x, 0) animated:YES];

    self.scrollView = [[UIScrollView alloc]initWithFrame:CGRectMake(0, 0,
    self.view.frame.size.width, self.view.frame.size.height)];
    Or:
    self.window = [[UIWindow alloc]initWithFrame:[[UIScreen mainScreen]bounds]]

    self.scrollView.contentSize = CGSizeMake(self.scrollView.frame.size.width *
    self.scrollView.frame.size.height);
    ```

18. **NSInteger CGFloat CGPoint CGRect CGSize has no asterisk** (also those primitiveTypes See 4.32)
19. See Pic ViewProperty for the difference

    bounds: Your view's internal drawing space's origin and size. The bounds property is

what you use inside your view's own implementation

center: The center of your view in your superview's coordinate space

frame: A rectangle in your superview's coordinate space which entirely contains your views bounds.size

20. Warning: Implicit conversion loses integer precision: 'long' to 'int'

```
float floatVal = 1.53456;
int roundedVal = lroundf(floatVal); // lround, lroundf, round, roundf. The
// warning, last two no warning. All rounded to the closest value
int roundedUpVal = ceil(floatVal); // ceilf, ceill
int roundedDownVal = floor(floatVal); // floorf, floorl
// f refers to as float, l refers to as long

NSLog(@"roundedVal is %d",roundedVal);
NSLog(@"roundedUpVal is %d",roundedUpVal);
NSLog(@"roundedDownVal is %d",roundedDownVal);
```

21. A UIEdgeInsets struct (four CGFloats in the order **top, left, bottom, right**) specifying margins around the content. A typical use for this would be that your scroll view underlaps(self: under) an interface element, such as a translucent status bar, navigation bar, or toolbar, and you want your content to be visible even when scrolled to its limit.
22. The moral of the story is to avoid direct comparisons between BOOL types and the YES constant.

    (a)
    ```
    BOOL b = 37;
    if (b) {
        printf("b is YES!\n");
    }
    if (b != YES) {
        printf("b is not YES!\n");
    }
    ```
    The output from this would be as follows:
    ```
    b is YES!
    b is not YES!
    ```

    (b) Note that ObjC (assuming the default C99 setting is enabled) also supports the bool data type, which is an intrinsic boolean type and can only be set to true or false. A safe way to convert between BOOL and bool is as follows:
    ```
    // from BOOL to bool
    bool b = myBOOL ? true : false;
    // and back
    myBOOL = b ? YES : NO;
    ```

23. In your examples you use @synthesize for synthesize properties, but synthesizing goes automatically in XCode5 and you don't need to write @synthesize anymore or maybe I'm wrong?

    Yes, Xcode will automatically generate the getter and setter methods but you need to access them with an underscore. When you use the @synthesize statement, you can access the object directly without an underscore.
24. object that's used to send the event is called event sender, the object captures the event

and respond to it is called event receiver

25. **Storyboard Tranform to code like button**
    property, init
    action, add.
26. action:@selector(changeSwitch:)

```
@interface ViewController (){
    UISwitch *_mySwitch;
}
@end

- (void)viewDidLoad {
_mySwitch = [[UISwitch alloc]initWithFrame:CGRectMake(130, 235, 0, 0)];
[_mySwitch addTarget:self action:@selector(changeSwitch) forControlEvents:U
[self.view addSubview:_mySwitch];
}
```

    With colon:

```
- (void)changeSwitch:(id)sender{
    if ([sender isOn]) {
        NSLog(@"Switch is ON");
    }else{
        NSLog(@"Switch is OFF");
    }
}
```

    Without colon:

```
- (void)changeSwitch{
    if ([_mySwitch isOn]) {
        NSLog(@"Switch is ON");
    }else{
        NSLog(@"Switch is OFF");
    }
}
```

27. Install **network link conditioner** to prevent iphone simulator from accessing network directly(also default)
    Xcode $\mapsto$ Open Developer Tool $\mapsto$ More Developer Tools and get Hardware IO Tools for Xcode
28. Outlets are used to create properties that reference your Interface Builder objects, and actions are used to connect methods to IB object actions.
29. Change the iOS deployment target to 7.0 in the build settings will result in UIAlertView not showing any warning
30. option + y for Chinese currency
31. alpha value from 1 to 0 goes from completely visible to fully transparent, like deep red go to light red
32. Remember that this is one of the only places you should do this-in the rest of your methods you should be using self.model and self.odometer. http://rypress.com/tutorials/objective-c/classes

33. Disable autolayout: select MainStoryboard.storyboard, under File inspector, uncheck "Use Autolayout"

Bugzilla

1. 1ec0ab is the navigation bar color patient edition
2. 108c04 is the green color, patient edition
3. password append, not delete completely
4. pragma mark change to chinese
5. loading failed scene and first time askDoctor scene mixed

from Baoguo

1. gar
2. other linker
3. result[@"data"][@"headimgurl"]

My bugzilla

1. In Common.h line
   `#define KBackgroundColor [UIColor colorWithHexString:@"EEEEEE"]`
   error:No known class method for selector 'colorWithHexString:'