# 一、常用控件

## 1 UITableView 的使用

### 1 设置数据源

#### 1.1 代码设置

添加< UITableViewDataSource >

self.tableView.dataSource = self;

#### 1.2连线法设置

dataSource 连到左边的View Controller Scene

reuseIdentifier – 可从用的标示 放到缓冲池中 – 提高程序的性能和效率

# 二 、常用第三方库

## 1 AFNetworking

## 2 MBProgressHUD

## 3 CocoaAsyncSocket

# 三 Object-C 语法

## 1 分类

### 1.1 分类格式

声明

@interface 类名 (分类名称)

@end

实现

@implementation类名 (分类名称)

@end

### 1.2 快捷创建分类

Command+N ->选左边的Source-> Objective-C File ->File Type -> Category

### 1.3 使用注意

1) 分类只能添加方法，不能添加成员变量

2) 分类可以访问原来类中的成员变量

3) 如果分类和原来类出现同名的方法，优先调用分类中的方法，原来类中的方法会被忽略

4) 方法调用的优先级(从高到低)

①分类(最后参与编译的分类优先)

②原来类

③父类

### 1.4 价值所在

1) 实现了类的相关方法的模块化

把不同的方法分配到了不同的分类文件中

可以用一个分类来代表一类功能(一个模块)

2) 支持团队协作

可以让团队内部的不同成员 共同扩充某个类的功能

可以为系统自带的类扩充功能

## 2 类扩展

### 1.1 什么是类扩展

可以为某个类扩充一些私有的成员变量和方法

写在.m文件中

英文名是Class Extension

### 1.2 书写格式

@interface 类名 ()

@end

### 1.3 和分类的区别

对比分类, 就少了一个分类名称, 因此也有人称它为”匿名分类”

## 3 Block

### 1.1 什么是Block

1. Block是iOS中一种比较特殊的数据类型
2. Block是苹果官方特别推荐使用的数据类型, 应用场景比较广泛

动画

多线程

集合遍历

网络请求回调

3) Block的作用

用来保存某一段代码, 可以在恰当的时间再取出来调用

功能有点类似于函数和方法

### 1.2 基本使用

1) Block的组成要素跟函数类似

返回值

形式参数

实际参数

2) Block的定义格式

返回值类型 (^block变量名)(形参列表) = ^(形参列表) {

};

调用Block保存的代码

block变量名(实参);

4) 使用typedef定义Block类型

typedef 返回值类型 (^block类型名称)(形参列表);

### 1.3 使用注意

1) 默认情况下, Block内部不能修改外面的局部变量

2) Block内部可以修改使用\_\_block修饰的局部变量

## 4 Protocol( 协议 )

### 1 什么是Protocol

1) Protocol翻译过来, 叫做”协议”

1. Protocol的作用

用来声明一些方法

也就说, 一个Protocol是由一系列的方法声明组成的

1. 任何类只要遵守了Protocol, 就相当于拥有了Protocol的所有方法声明

### 2 书写格式

Protocol 的定义

@Protocol 协议名称

// 方法声明列表

@end;

类遵守协议

@interface 类名 : 父类 <协议名称1, 协议名称2,…>

@end

协议中有2个关键字可以控制方法是否要实现(默认是@required，在大多数情况下，用途在于程序员之间的交流)

@required：这个方法必须要实现（若不实现，编译器会发出警告）

@optional：这个方法不一定要实现

### 3协议遵守协议

一个协议可以遵守其他多个协议

一个协议遵守了其他协议，就相当于拥有了其他协议中的方法声明

@protocol 协议名称 <协议1, 协议2>

@end

### 4 基协议

NSObject是一个基类，最根本最基本的类，任何其他类最终都要继承它

还有名字也叫NSObject的协议，它是一个基协议，最根本最基本的协议

NSObject协议中声明很多最基本的方法

description

retain

release

建议每个新的协议都要遵守NSObject协议

# 四 数据库

## 1 SQLite

# 五 -----------

## 1 键盘管理

resignFirstResponder

当叫出键盘的那个控件(第一响应者)调用这个方法时，就能退出键盘

endEditing

只要调用这个方法的控件内部存在第一响应者，就能退出键盘

## 2 控件大小和位置

1 frame属性是相对于父容器的定位坐标。

2 bounds属性针对于自己，指明大小边框，默认点为（0，0），而宽和高与frame宽和高相等。

3 center属性是针对与frame属性的中心点坐标。

4 当frame变化时，bounds和center相应变化。

5 当bounds变化时，frame会根据新bounds的宽和高，在不改变center的情况下，进行重新设定。

6 center永远与frame相关，指定frame的中心坐标！

## 3 解析 plist

获得Plist文件的全路径

NSBundle \*bundle = [NSBundle mainBundle];

NSString \*path = [bundle pathForResource:@"images" ofType:@"plist"];

加载plist文件

\_images = [NSArray arrayWithContentsOfFile:path];

- (NSArray \*)images

{

if (\_images == nil) {

NSBundle \*bundle = [NSBundle mainBundle];

NSString \*path = [bundle pathForResource:@"images" ofType:@"plist"];

\_images = [NSArray arrayWithContentsOfFile:path];

}

return \_images;

}

## 3 UIView常用属性和方法

### 属性

@property(nonatomic,readonly) UIView \*superview;

获得自己的父控件对象

@property(nonatomic,readonly,copy) NSArray \*subviews;

获得自己的所有子控件对象

@property(nonatomic) NSInteger tag;

控件的ID\标识，父控件可以通过tag来找到对应的子控件

@property(nonatomic) CGAffineTransform transform;

控件的形变属性（可以设置旋转角度、比例缩放、平移等属性）

### 方法

- (void)addSubview:(UIView \*)view;

添加一个子控件view

- (void)removeFromSuperview;

从父控件中移除

- (UIView \*)viewWithTag:(NSInteger)tag;

根据一个tag标识找出对应的控件（一般都是子控件）