数据类型，流程控制，指针，数组，函数。。。。。基本同C++。

**块**

匿名函数，类似lamdba表达式。

定义块：^[return type] (param type p1, ...)

如：

void (^block)(void) = ^(void); //定义块变量，并赋值

double(^b)(double, double) =

^(double d1, double d2)

{

return sqrt(d1 \* d2);

}

也可以先typedef：typedef void(^Bxx)(NString\*);

块变量无法修改局部变量。

但是这样定义可以：block type value = xxx;

**类**

.h

@interface CLASSNAME : BASECLASS

{

[@public/package/protected/private] @package 类的同一映像中，默认private

member

}

-(return type) FUNNAME:(PARAM type) PARAM name [FUNNAME:(type)name ... ];

+... 类方法

@end

.m

@implement CLASSNAME

static TYPE name = initValue; 类变量

{

.m中声明的变量

}

-...实现

-(void) test

{

.m中声明的函数

}

+...提供给类变量的方法

@end

方法调用

[调用者 方法名:参数 形参标签:参数...]

定义类对象

CLASSNAME\* class = [[CLASSNAME alloc] init];

调用方法

[class FUNNAME:参数 FUNNAME:参数...];

访问实例变量

class->变量名，若合成了存取方法，可以直接用 ：class.变量名。

**id类型**可以代表所有对象的类型

合成存取方法

@interface ...

@property (修饰) type name;

//...

修饰：

assign 不改引用计数，用在int，double等”小类型”

atomic，nonatomic：线程安全

copy：快照一份

readonly : 只合成get方法

strong，weak：引用的对象不会自动回收

键值编码

合成存取方式使你能够通过”.”语法访问属性，OC还提供了一种以字符串形式操作属性的方法，叫做Key Value Coding KVC。

NSObject 具有NSKeyValueCoding协议，该协议的两个方法：setValue，valueForKey。

键值监听

解决的问题：数据模型组件的数据发生变化，视图组件能动态更新自己。

KVO机制由NSKeyValueObserving协议提供。

OC的继承，重写和多态和C++大致相同，直接重写；id代表任意对象类型；super，self；

。。。

**类别**

为现有类添加新方法

@interface 已有类(category name)

//method

@end

@implmentation 已有类(category name)

//method implmentation

@end

用法：

1. 模块化设计
2. 调用私有方法（即.m中的方法）

只要在类别中声明私有方法，使用时#import这个类别.h

1. 实现非正式协议

扩展：

临时对某个类的接口进行扩展。

@interface 已有类 ()

{

变量

}

方法

@end

**协议**

定义类应该遵守的规范，即应该有哪些方法。

非正式协议：

@interface NSObject(catagory name)

方法

@end

因为NSObject是基类

正式协议：

@protocol name <父协议1，...>

{

@optional / @required

方法

}

实现协议：也就是重写方法。

使用协议定义变量：

NSObject<protocol 1, ...>\* name;

id<protocol1 ,...> name;

只能调用协议方法。

每个类都有一个Class

获取Class

1），Class NSClassFromString(NSString\* classname)

2)，[CLASSNAME class]

3)，CLASSNAME.class

检查继承关系（包括协议）

isKindOfClass 对象是否是类及其子类的实例

isMemberOfClass 对象是否是该类的实例

conformsToProtocol 对象是否实现了该协议

动态获取SEL对象：

1. @selector(完整的方法签名)
2. SEL NSSelectorFromString(NSString\*)

SEL对象

OC把方法称为选择器。

动态调用方法：

1. [NSObject performSelector] withObject传参
2. objc\_msgSend(receiver, selector,...) 函数

如：

[self performSelector:@selector(move:) withObject: 2]

objc\_msgSend(self, NSSelectorFromString(@”move:”), 2);

目前难度所在：

OC有大量的封装的类，光了解这些类的方法就够你懵逼的了，关键还要理解，要像使用for循环一样了解它能做什么，能适用哪些场景。。。这个更需要时间去积累。

现阶段只能积累了。

帮助文档是很好的学习和查阅的资料，少做回头功，看清类的本质，争取一次记住（或者链接它的网络）。

杂项：

判断指针实际类型：isMemberOfClass isKindOfClass isSubclassOfClass

NSNumber P205

打印对象 NSLog(@”%@”, p) -> [p description]

== 与 isEqual:(NSObject) 不同的变量，内容相等，重写isEqual:

copy mutableCopy NSObject 自定义的类需要实现NSCopying协议（copyWithZone方法），注意调用父类的copy，注意深浅拷贝。

NSString NSMutableString 字符串

NSDate 日期时间

NSDateFormatter 日期格式器

NSCalender NSDateComponents 提取年月日时分秒，合成NSDate

NSTimer 定时器

NSArray，NSSet，NSDictionary :

isEqual

调用集合中的方法makeObjectsPerformSelector:；以集合为参数调用方法 enumerateObjectsUsingBlock:

排序sortedArrayUsingFunction:context:

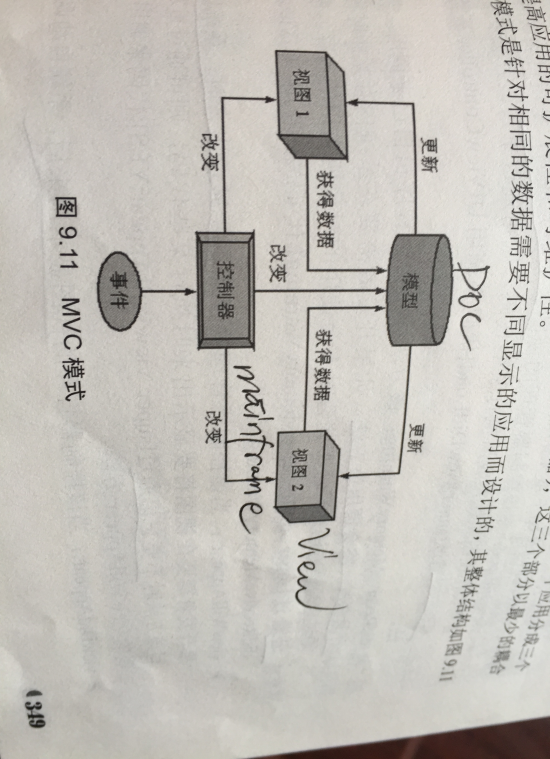
遍历 objectEnumerator: for-in

NSArray的KVC与KVO

NSSet，NSDictionary查手册。

文件I/O 手册

**IOS的MVC**



大部分控制器都会继承UIKit的UIViewController类，该类包含大量可重写的方法：

注意调用基类的方法。

viewDidLoad:视图被加载完成后，系统自动调用。

didReceiveMemoryWarning：系统检测到可用内存紧张时。

viewWillAppear：视图将要显示。

。。。

程序获取控件的两种方式：

1，IBOutlet连接

2，为控件指定Tag

1. 点辅助编辑器，ctrl+拖控件到ViewControl.h，可看到

@property(strong, nonatomic) IBOutlet UILabel\* xxx;

self.xxx.text = @”xxx”; // 使用

1. view属性有个Tag

UILabel\* myLb = (UILabel\*)[self.view viewWithTag:12]

程序事件处理

1. IBAction绑定

同IBOutlet一样，拖动按钮它就有Action选项，然后选择事件。操作界面时，只要触发了响应的操作，就会触发。

1. 继承自UIControl的都有些方法注册，删除事件

addTarget:action:forControlEvents:

removeTarget:action:forControlEvents:

1. 像UITableView不继承自UIControl，你要用代理对象（拥有某种协议），实现这些协议。

用代码控制UI，好处是维护方便；

自定义UI控件，你需要继承UIView，然后改写方法；

右键工程，新增类；

定制IOS图标，启动画面；

界面编程

UIControl: UIView : UIResponder

4种不同的状态：普通，高亮，禁用，选中；

UIButton 按钮

了解：

1. 属性面板
2. 外观，布局
3. 事件

UITextField 文本框

背景也是控件

UITextView 多行文本

没有继承UIControl，它的委托对象必须实现UITextViewDelegate协议，该协议的方法有：。。。

UIScrollView 可滚动控件

UINavigationBar 导航条 UINavigationController

UIToolbar 工具条

UIMenuItem 菜单项 UIMenuController

UISwitch 开关按钮

UISegmentedControl 分段控件

UIImageView 图像控件

没有继承UIControl

UIProcessView 进度条 UIActivityIndicatorView 旋转齿轮

UISlider 拖动条

UIAlertView 警告框 UIActionSheet 按钮列表

UIDatePicker 日期选择器

UIPickerView 选择器

委托对象完成事件处理

自定义选择器

UIVisualEffectView 为控件添加效果，如模糊

UIStepper 微调器

UIWebView 网页控件

UITableView 表格 UITableViewController 表格控制器

UISearchBar 搜索条 UISearchDisplayController

segue 控制过渡

UICollectionView 网格 UICollectionViewController 网格控制器

UITabBar UITabBarController

UIPageControl 页控件 UIPageViewControl

iphone 4 ,4s屏幕大小320 \* 480 ，由于它们采用了Retina高分辨率屏幕，因此其屏幕实际分辨率640pix \* 960pix

5，6，6plus 同理

配置屏幕方向

指定视图控制器支持的方向

为自动布局添加约束

图形，图像，动画