OC

源代码—》编译预处理—》编译—》链接—》运行

编译预处理指令

1. 在编译前进行解析处理的指令。
2. 所有编译预处理指令都是以#开头的。
3. 所有预编译处理指令都是不需要分号的。
4. 预处理指令可出现在程序的任何位置，它的作用范围是从它出现的位置到文件尾。习惯上尽可能将预处理指令写在源程序开头，这种情况下，它的作用范围就是整个源程序文件。

宏定义

1. 宏名一般使用大写字母。
2. 程序中用双引号扩起来和注释中的宏名不会被替换。
3. 在预编译处理用字符串替换宏名时，不作语法检查，只是简单的字符串替换。只有在编译的时候才对已经展开宏名的源程序进行语法检查。
4. 宏名的有效范围从定义位置到文件结束。如果需要终止宏定义的作用域，可以用#undef命令。
5. 宏名和参数列表间不能有空格，否则空格后面的所有字符串都作为替换的字符串。
6. 宏定义不涉及存储空间的分配、参数类型匹配、参数传递、返回值问题；函数调用在程序运行时执行，而宏替换只在编译预处理阶段进行，所以带参数的宏比函数具有更高的执行效率。

文件包含include

1. #include <文件名>直接到c语言库函数头文件所在目录中寻找文件。
2. #include “文件名”系统会先在源程序当前目录下寻找，若找不到，再到操作系统的path路径中查找，最后才到c语言库函数头文件所在目录中查找。

static关键字

1. static定义的这一行代码仅仅会执行一次。

NSObject

1. 基类，所有类的祖先类，让子类具有创建对象的能力。
2. 类的声明以@interface开头，以@end结尾；类的实现以@implementation开头以@end结尾。

3. 若一个类只有声明没有实现，那么这个类在链接的时候就报错，如同C中只有声明没有定义一般。
4. 内部定义常用方法
   - (BOOL)isKindOfClass:(Class)aClass;判断是否为aClass或者aClass子类的实例。
   - (BOOL)isMemberOfClass:(Class)aClass;判断是否为aClass的实例(不包括aClass的子类)。
   - (BOOL)conformsToProtocol:(Protocol *)aProtocol;判断对象是否实现了aProtocol协议。
   - (BOOL)respondsToSelector:(SEL)aSelector;判断对象是否拥有参数提供的方法。
   - (void)performSelector:(SEL)aSelector withObject:(id)anArgument afterDelay:(NSTimeInterval)delay;延迟调用参数提供的方法，方法所需参数用withObject传入。

[类名 new]作用

1. 为该类创建一个对象并在堆中分配内存。
2. 初始化成员变量。
3. 返回指向刚刚创建出来的对象的指针。

NSLog与printf对比

1. printf是C语言提供的，在stdio.h头文件中；
   NSLog是Foundation框架提供的在NSObjCRuntime.h中。
2. NSLog包含日志输出日期以及对应的应用程序名称。
3. NSLog自动换行，末尾\n是无效的。
4. NSLog中的格式字符串不是普通C语言字符串，NSString对象@“”它是一个NSString对象的字面量表示。
5. printf中所有占位符在OC中都是支持的。
6. NSLog新增了格式符%@用于输出对象。

类方法与对象方法

1. 类方法声明和定义以+开头，对象方法以-开头。
2. 类方法只能通过类名调用它，对象方法只能通过对象调用。
3. 类方法中不能访问成员变量，对象方法可以直接访问成员变量。
4. 类方法中不可直接调用对象方法，可通过传入参数和局部变量来调用对象方法；
   对象方法可以调用对象方法，也可以通过类名调用类方法。
5. 类方法比对象方法执行速度更快，更节省内存空间，因为类方法是通过类对象调用的，可以通过类名直接找到，类对象在程序已启动就被创建并初始化，在整个程序

运行期间类对象只有一个，不需要开辟存储空间。而对象方法必须通过对象调用，需要通过isa指针找到类对象，然后才能找到对象方法。

6. 类方法和对象方法可以重名。

继承

1. 类方法是可继承可覆盖的。
2. 子类中不能定义和父类同名的成员变量。
3. private成员变量只能在定义类中访问。

方法存储位置

1. 每个类的方法列表都存储在类对象中。
2. 每个方法都有一个与之对应的SEL类型的对象。
3. 根据一个SEL对象就可以找到方法的地址，进而调用方法。
4. SEL类型的定义typedef struct objc_selector *SEL;

self

1. self在对象方法中，它是调用这个方法的对象，可以访问其它对象方法。
2. self在类方法中，它是调用这个方法的类，可以访问其它类方法。

SEL对象基本使用

1. 对象创建SEL s = @selector(test); SEL s2 = NSSelectorFromString(@“test”);
2. 对象转为NSString对象 NSString *str = NSStringFromSelector(@selector(test));
3. 检测对象是否实现了test方法
   Person *p = [Person new];
   [p respondsToSelector:@selector(test)];
4. 调用对象的test方法[p performSelector:@selector(test)];

Category分类

1. 可以访问原始类的实例变量，但不能添加变量，只能添加方法。想添加变量可通过继承创建子类。
2. 可实现原始类的方法，但不推荐，因为它是直接替换掉原来的方法，这么做的后果是再也不能访问原来的方法。
3. 多个Category中如果实现了相同的方法，只有最后一个参与编译的才会有效，Compile Source中从上往下依次是先到后参加编译，因此下面分类中的方法会覆盖掉上面分类中同名的方法。

4. 分类可以访问原来类中的成员变量。
5. @property在分类中只能生成getter与setter方法的声明，不能生成setter与getter方法的实现和成员变量。

Extensions类扩展

1. 类扩展是没有名称的，直接在类后加括号，相对于分类缺少括号中名称。
2. 类扩展可以扩充方法和成员变量。
3. 一扩展一般写在.m文件中，用来扩充私有的方法和成员变量。

Protocol协议

1. 用来声明一些方法，由一序列的方法声明组成。
2. 任何类只要遵守了Protocol，就相当于拥有了Protocol的所有方法声明。
3. 协议中方法默认是@required必须实现的,遵守该协议的类中不实现会有警告，也可通过@optional说明为可选非必需实现，遵守该协议类中不实现不会有警告。在@required后面的都是@required的直到遇到@optional，在@optional后面的都是@optional直到遇到@required，与成员变量访问权限类型相似。
4. 一个协议可遵守其他多个协议。
5. 建议每个新的协议都要遵守NSObject协议。
6. 通过id<协议名称>定义出来的指针变量可以指向任意实现这个协议的类的实例对象，通过该变量可以调用协议中声明的方法，从而实现多态。

Block

1. iOS中一种比较特殊的数据类型，用来保存某一段代码，可以在恰当的时间再取出来调用。
2. 默认情况下，Block内部不能修改外面的局部变量。
3. Block内部可以修改使用__block修饰的局部变量，__block修饰的局部变量的地址和Block内部使用的同名变量地址不相同。
4. 作为参数格式 返回值类型 (^)(形参列表)。
5. 定义格式 返回值类型 (^block变量名称)(形参列表);
6. 实现格式 ^ 返回值类型 (形参列表){功能语句……}。
7. 无参无返回值 void (^block1)() = ^{NSLog(@“aaaaa”);}。
8. 有参无返回值
   void (^block2)(NSString *name) = ^(NSString *name){NSLog(@“%@”,name)}。
9. 有参有返回值
   void (^sum)(int num1, int num2) = ^(int num1, int num2){return num1 + num2;}。
10. 当在block内部使用对象时block内部会对这个对象增加一个强引用。

+load
1. 在程序启动的时候会加载所有的类和分类，并调用所有类和分类的+load方法。
2. 先加载父类，再加载子类；也就是先调用父类的+load，再调用子类的+load。
3. 先加载原始类，再加载分类Category。
4. 不管程序运行过程中有没有用到这个类，都会调用+load加载。

+initialize
1. 在第一次使用某个类时(比如创建对象等)，就会调用一次+initialize方法。
2. 一个类只会调用一次+initialize方法，先调用父类的，再调用子类的。

dealloc
1. 如果是ARC记得不能再调用[super dealloc];

ARC中强指针
1. __strong标示，默认所有的指针都是强指针。
2. 只要有强指针指向一个对象，那么这个对象就不会被释放，只要没有强指针指向对象，那么这个对象就会被立即释放。

ARC中弱指针
1. __weak标示。
2. 弱指针不影响对象的回收。
3. 不要对新创建的对象使用弱指针，因为这样的话对象以创建出来就被回收了没有意义。
4. 循环引用中必须要有一个是弱指针。

自动释放池
1. 系统中存在一个自动释放池栈，当遇到@autoreleasepool{时会将这个自动释放池放入栈中，当遇到与之对应的}时，自动释放池出栈，自动释放池出栈时会对池中所有对象进行一次release操作。
2. 自动释放池栈中，只有栈顶自动释放池是活动的，其它的都在休眠。
3. 当调用对象的autorelease时会把这个对象放入栈顶的自动释放池中。
4. 只要是从方法中返回一个对象都要使用自动释放池。

常用宏

1. __FILE__: 当前源文件名。
2. __LINE__:当前行号。
3. __FUNCTION__:当前函数名称。
4. __VA_ARGC__:可变参数。
5. _cmd: 代表着当前方法的SEL。

NSString

1. 通过类名的字符串形式实例化对象
   Class class = NSClassFromString(@“Person”);
   Person *person = [[class alloc] init];
2. 将类名变成字符串
   Class class = [Person class];
   NSString *className = NSStringFromClass(class);
3. 通过方法的字符串形式实例化方法
   SEL selector = NSSelectorFromString(@“setName:”);
   [person performSelector:selector withObject:@“Mike”);
4. 将方法变成字符串
   NSStringFromSelector(@selector(setName:));
5. substringFromIndex从哪个索引开始截取到字符串的末尾(包含索引位置字符)，开始位置索引为0。
6. substringToIndex从开始位置截取到索引位置(不包含索引位置字符)。
7. [str stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding];
   把UTF8编码的字符串，编码成为URL中可用的字符串，中文会转换为%形式字符串，常用于有中文路径。
8. [str stringByReplacingPercentEscapesUsingEncoding:NSUTF8StringEncoding];
   把URL中可用的字符串，编码成为UTF8编码的字符串。

集合

1. 就是能用来容纳OC对象的容器。NSArray,NSDictionary等都是。
2. 只要将一个对象添加到集合中，这个对象计数器就会加1(做一次retain操作)。
3. 只要将一个对象从集合中移除，这个对象计数器就会减1(做一次release操作)。
4. 如果集合被销毁了，集合里面的所有对象计数器都会减1(做一次release操作)。

NSArray数组排序

1. selector选择器
   array = [array sortedArrayUsingSelector: @selector(compare:)];

2. 比较器
   array = [array sortedArrayUsingComparator:^NSComparisonResult(Person *p1, Person *p2){return [p1.name compare:p2.name];}];。
3. 属性描述器
   NSSortDescriptor *desc1 = [NSSortDescriptor sortDescriptorWithKey:@“name” ascending: YES];
   NSSortDescriptor *desc2 = [NSSortDescriptor sortDescriptorWithKey:@“age” ascending: YES];
   array = [array sortedArrayUsingDescriptor:@[desc1,desc2]];。
4. 构造数组时不要在数组中间放nil，nil表示数组结束了，数组中可放不同类型对象，但建议放同一种类型对象。

常见问题

创建对象步骤

1. 开辟内存空间。
2. 初始化参数。
3. 返回内存地址值。

NSSet、NSArray、NSDictionary区别

都属于不可变集合类，在创建完集合类后就不能够对他们进行修改，在集合类里只能添加对象元素不能添加基本数据类型。

NSSet

1. 无序集合。
2. 在内存中存储的地址是不连续的。
3. 添加进去的元素是不可重复的。

NSArray

1. 有序集合。
2. 在内存中存储的地址是连续的，添加的元素是可重复的。
3. 支持通过下标访问元素。
4. 如果想知道一个元素是否存在这个数组当中的话，则需要遍历整个数组一个个去判断，效率低下。

NSDictionary

1. 无序集合。
2. 数据存储方式是key value键值对方式进行存储的。
3. key在整个NSDictionary里是唯一的，如果key发生重复，那么后添加的元素会覆盖之前的。

UIButton的imageView属性是只读属性不能赋值，故不能通过该属性更改图片。

self.presentViewController不起作用，显示的还是原始vc，将其放到异步线程中

```
DispatchQueue.main.async {
            self.present(vc, animated: true, completion: nil)
        }
```