类和对象

类和对象

- 面向对象和面向过程区别
- 类和对象定义
- 类的组成
- 对象创建

类和对象

面向过程和面向对象区别

	面向过程	面向对象
特点	分析解决问题的步骤,实现函数,依次调用函数	分析该问题需要参与的对象,各个对象的作用,完成该事件需要多个对象协同完成该任务.
侧重点	实现功能（步骤）	对象的设计（包含哪些特征和行为）
语言举例	C语言	OC,C++,Java等

类和对象

类和对象定义

- 类：一组具有相同属性和行为的事物的集合，就是一种数据类型
- 对象：类的具体实现，也即是类这种数据类型的变量

类和对象

类的组成

- @interface部分:

  1.对外声明类的特征和行为

  2.放在.h文件中

- @implementation部分：

  1.行为的具体实现

  2.放在.m文件中

类和对象

- @interface部分

```
@interface NewClassName: ParentClassName
    propertyAndMethodDeclarations;
@end
```

类和对象

- @implementation部分

```
@ implementation NewClassName
    methodDefinitions;
@end
```

类和对象

对象创建

- [ClassName alloc]init];

类的属性和方法

类的属性和方法

- 属性
- 实例方法和类方法

类的属性和方法

属性

- @property (...)
- nonatomic 和 atomic
- assign strong weak
- readonly readwrite

类的属性和方法

实例方法和类方法

- 实例方法(-):方法属于某个对象
- 类方法(+):方法属于类也就是所有对象共享的

继承

继承

- 父类和子类
- 增加新属性和方法
- 重载

循环语句

父类和子类

- NSObject是所有OC类的最终父类，也就是根类
- 子类继承父类的所有属性和方法

继承

增加新属性和方法

- 通过在子类中添加新属性和方法来扩展父类没有的功能

继承

重载

- 通过重载来更改父类中的方法的具体实现
- super来调用来自父类的方法

多态

多态

- 基本概念
- 好处
- 使用注意

多态

基本概念

- 多种形态，必须要有继承才有多态
- 父类指向子类对象，然后进行动态检测，以调用真实的对象方法

多态

好处

- 使用父类来代替多种子类来简化代码和操作

多态

使用注意

- Animal *animal = [[Dog alloc]init] animal对象到底是动物还是狗，让人疑惑
- 父类的指针变量不能直接调用子类特有的方法，需要强制转换成对应的子类才能调用

动态类型和动态绑定

动态类型和动态绑定

- 定义
- id类型
- 静态类型和动态类型
- SEL
- 常用方法

动态类型和动态绑定

定义

- 动态类型：程序直到执行时才能确定所属的类。
- 动态绑定：程序直到执行时才能确定实际要调用的方法。

动态类型和动态绑定

id类型

- 可以存储任何类的对象
- 可以向它发送任何消息，即它可以调用所有的方法

动态类型和动态绑定

id类型如何判断调用哪个类型的方法

- OC总是跟踪对象所属的类，然后动态调用方法
- 但这个过程是在运行时，而不是编译时进行的

动态类型和动态绑定

静态类型和动态类型

- 编译期检查与运行时检查
- 静态类型在编译期就能检查出错误
- 静态类型声明代码可读性好
- 动态类型只有在运行时才能发现错误

动态类型和动态绑定

SEL

- 获取类方法的编号, 等同C语言的函数指针

动态类型和动态绑定

常用方法

- isMemberOfClass：判断是否是这个类的实例
- isKindOfClass：判断是否是这个类或者这个类的父类的实例
- respondsToSelector：判读实例是否有这样方法
- performSelector：执行SEL指定的方法

对象初始化

对象初始化

- 定义
- 默认构造方法
- 自定义构造方法

对象初始化

定义

- 初始化(initialization)：从系统取得一块内存，准备用于存储对象

对象初始化

默认构造方法

- Person *person = [[Person alloc] init];
- 默认初始化完毕后，所有成员变量的值都为默认值。

对象初始化

- 自定义构造方法

属性作用域

属性作用域

- 实例变量作用域
- 外部变量
- static和const

属性作用域

实例变量作用域

- @protected : 该类和它的子类访问，是实例变量的默认情况
- @private ：只能被该类访问
- @public : 所有类都可以访问

属性作用域

外部变量

- 可被任何类访问
- 使用extern关键字声明就可以访问外部变量

属性作用域

static和const

- static表示变量在程序开始执行的时候初始化，之后会一直保留它的值
- const变量必须在定义时有初始值，变量在程序运行期间不能修改它的值。

分类和协议

分类和协议

- 分类
- 协议

分类和协议

分类

- 不必访问类的源代码，也无需创建子类，就可以扩展现有类

分类和协议

- 分类的格式

```
@interface 原类名 (分类名)
@end
@implementation 原类名(分类名)
@end
```

分类和协议

分类注意事项

- 可以访问实例变量，但不能添加任何变量
- 不要使用分类重载方法
- 如果需要添加变量或者重载方法，建议创建子类

分类和协议

协议

- 协议是多个类共享一个方法列表。它列出的方法没有对应的实现，由其他遵守这个协议的来实现。

分类和协议

- 协议的格式

@protocol 协议名
方法列表
@end

分类和协议

- @required 和 @optional