OC基础二

方法

今日作业

- 为上次作业中的15个类：
- 添加初始化方法 和 指定初始化方法
- 添加便利构造器方法

课上任务

- 创建 Student类继承于 Person类
- 为Student类添加 初始化方法和便利构造器方法

课程目标

- 了解继承（面向对象三大特性之一）
- 掌握IOS中的方法（函数）

熟悉的画面

熟悉的画面

熟悉的画面

熟悉的画面

熟悉的画面




熟悉的画面

• 普通僵尸

领队僵尸

路障僵尸

铁桶僵尸

熟悉的画面

• 普通僵尸

领队僵尸

路障僵尸

铁桶僵尸

如何创建这些类?

类的特征和行为

类的特征和行为

普通僵尸

血量
攻击力
移动速度

-行走
-攻击
-死亡

路障僵尸

血量
防具
攻击力
移动速度

-行走
-攻击
-死亡
失去装备

铁桶僵尸

血量
弱点
防具
攻击力
移动速度

-行走
-攻击
-死亡
失去装备

类的特征和行为

普通僵尸

血量
攻击力
移动速度

-行走
-攻击
-死亡

路障僵尸

血量
防具
攻击力
移动速度

-行走
-攻击
-死亡
失去装备

铁桶僵尸

血量
弱点
防具
攻击力
移动速度

-行走
-攻击
-死亡
失去装备

存在部分相同特征和行为

分析

- 相同点：血量、攻击力、移动速度；移动、攻击、死亡
- 不同点：弱点、防具、失去装备。
- 结论：虽然相似，但不是同一类；类型多时，重复的代码比较多。

继承

- OC中，一个类可以继承另外一个类
- 被继承的类称为父类（super class）或超类
- 继承的类称为子类（subclass）
- 子类可直接“拥有”父类中除了@private实例变量之外的全部内容。

继承

- 实现继承很简单：在类的接口部分使用符号“：”

- 例如：

- OC只允许单继承-一个子类最多只能有一个直接父类。
- 没有父类的类称为根类，OC中的根类是NSObject。

继承

- 实现继承很简单：在类的接口部分使用符号“：”

```
@interface childClass： superClass {
}
```

- 例如：

```
@interface Person： NSObject {
}
```

- OC只允许单继承-一个子类最多只能有一个直接父类。
- 没有父类的类称为根类，OC中的根类是NSObject。

继承的实现

```
#import "Person.h"
@interface Student : Person
{
    int      number;        //学号
    float    score;         //分数
}
- (void)study;

@end
```

继承的实现

```
#import "Person.h"
@interface Student : Person
{
    int      number;        //学号
    float    score;         //分数
}
- (void)study;

@end
```

- Student继承于Person类，除了包含自身特征和行为外，还包含父类的特征和行为。

继承的实现

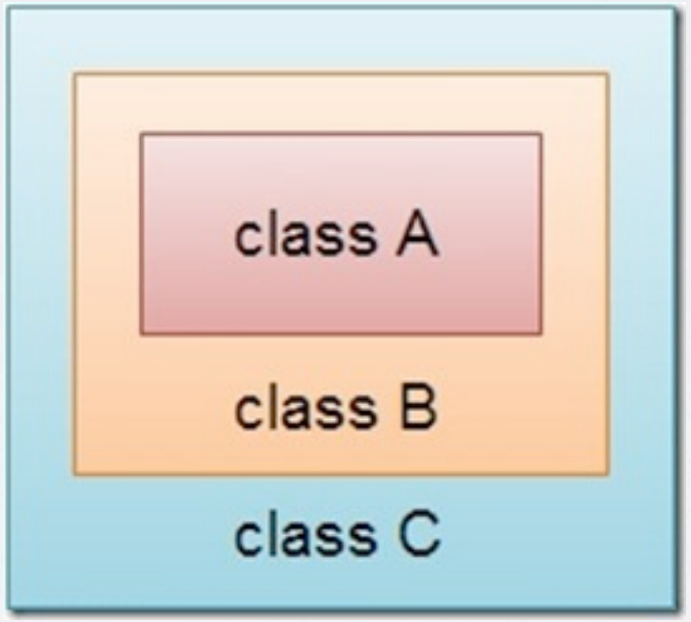
```
#import "Person.h"
@interface Student : Person
{
    int      number;        //学号
    float    score;         //分数
}
- (void)study;

@end
```

- Student继承于Person类，除了包含自身特征和行为外，还包含父类的特征和行为。

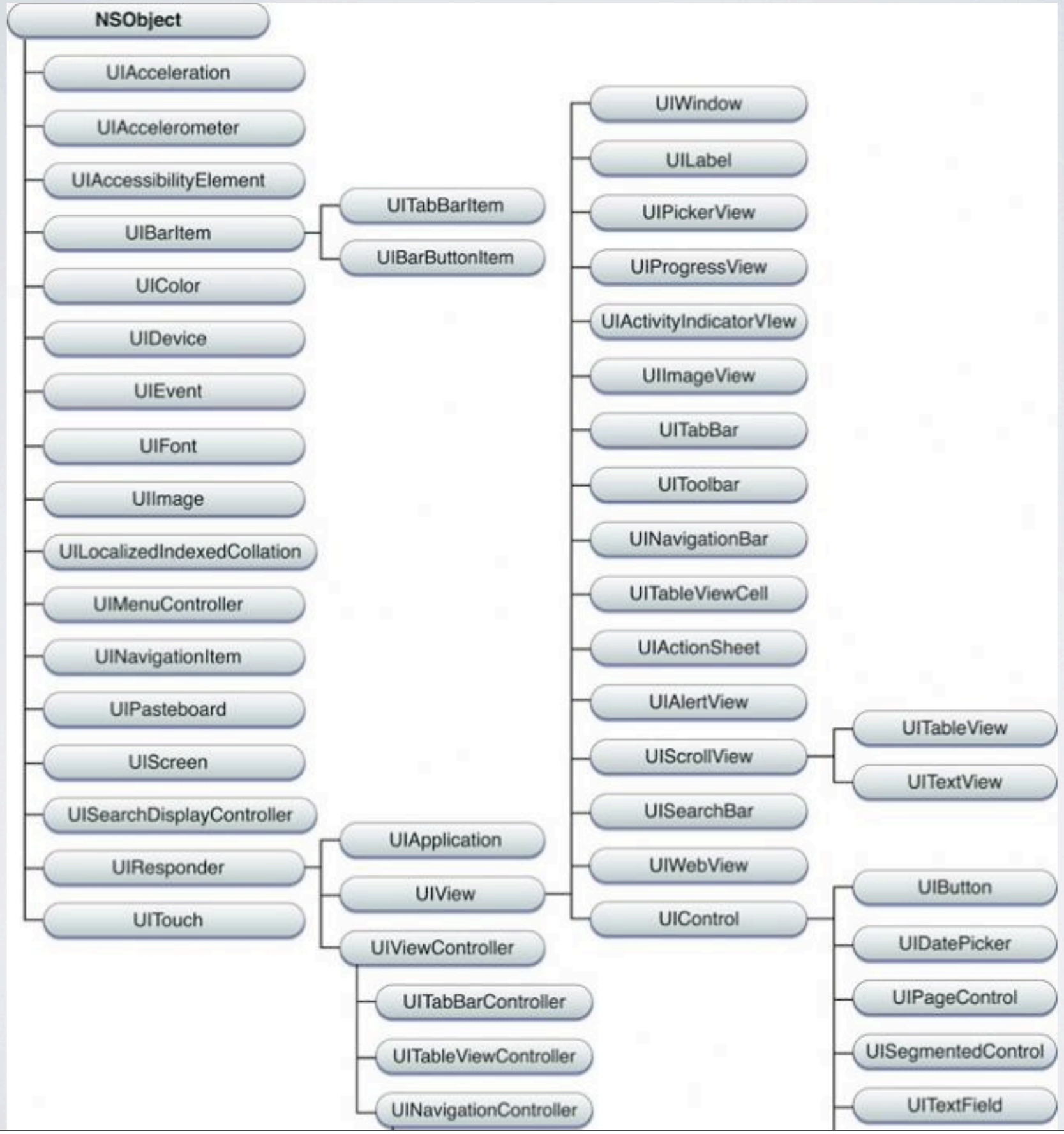
代码演示

继承示意图

- 父类-子类关系图
- 详情 见黑板草图



继承树


NSObject
UIAcceleration
UIAccelerometer
UIAccessibilityElement
UIBarItem
UITabBarItem
UIBarButtonItem
UIColor
UIDevice
UIEvent
UIFont
UIImage
UILocalizedIndexedCollation
UIMenuController
UINavigationItem
UIPasteboard
UIScreen
UISearchDisplayController
UIResponder
UIApplication
UIView
UIViewController
UITabBarController
UITableViewController
UINavigationController
UITouch
UIWindow
UILabel
UIPickerView
UIProgressView
UIActivityIndicatorView
UIImageView
UITabBar
UIToolbar
UINavigationBar
UITableViewCell
UIActionSheet
UIAlertView
UIScrollView
UITableView
UITextView
UISearchBar
UIWebView
UIControl
UIButton
UIDatePicker
UIPageControl
UISegmentedControl
UITextField

继承

- 子类能直接继承父类的方法
- 还可以重写父类的方法（子类自己实现行为）
- 子类重写了父类的方法，调用过程中执行子类的方法
- 继承具有传递性：如果A是B的子类，B是C的子类，那么A也具有C的特征和行为。

继承

- 子类能直接继承父类的方法
- 还可以重写父类的方法（子类自己实现行为）
- 子类重写了父类的方法，调用过程中执行子类的方法
- 继承具有传递性：如果A是B的子类，B是C的子类，那么A也具有C的特征和行为。

代码演示

练习

- 创建一个Teacher类，继承于Person类

方法

- OC语言里，方法分两大类：类(+)方法和实例(-)方法。
- 类方法必须用类（名）来调用。
- 实例方法必须用对象来调用。
- 方法是类的行为，在接口文件中声明，在实现文件中实现。

方法

- setter、getter都是OC中的方法
- 除了上节课定义的那些方法
- OC中同样有多参方法
- 如何得知一个方法的方法名

方法

- setter、getter都是OC中的方法
- 除了上节课定义的那些方法
- OC中同样有多参方法
- 如何得知一个方法的方法名

代码演示

初始化方法

- 创建对象分两步：分配空间和初始化。

- 初始化方法的作用就是为对象赋初始值。

- 初始化方法通常以init开头，返回值为id类型。例如：

```
- (id)initWithName:(NSString *)name;
```

- 一个类可以有多个初始化方法。

初始化方法

- 创建对象分两步：分配空间和初始化。

- 初始化方法的作用就是为对象赋初始值。

- 初始化方法通常以init开头，返回值为id类型。例如：

```
- (id)initWithName:(NSString *)name;
```

- 一个类可以有多个初始化方法。

代码演示

SELF、SUPER

- self和super是OC语言中的关键字。
- super在OC中就一个作用：调用父类中的方法（非自身）
- self也只有一个作用：调用自身的方法。
- self始终代表调用方法的对象（very important）

指定初始化方法

- 一个类只有一个指定初始化方法
- 在众多的初始化方法中，无论你调用哪个初始化方法，指定初始化方法都会执行。
- 指定初始化方法会跟父类初始化方法有一定联系。

指定初始化方法

- 一个类只有一个指定初始化方法
- 在众多的初始化方法中，无论你调用哪个初始化方法，指定初始化方法都会执行。
- 指定初始化方法会跟父类初始化方法有一定联系。

代码演示

便利构造器

- 便利构造器是一种快速创建对象的方式。它本质上是把初始化方法做了一次封装，方便外界使用。

- 便利构造器是一个类方法。

- 通常以类名开头，返回值仍然是id类型。

便利构造器

- 便利构造器是一种快速创建对象的方式。它本质上是把初始化方法做了一次封装，方便外界使用。

- 便利构造器是一个类方法。

- 通常以类名开头，返回值仍然是id类型。

代码演示

总结

- OC只允许单继承，继承具有传递性。
- 子类从父类继承了除@private实例变量之外的所有东西。
- 子类可以重写父类方法，定制自己个性化的行为。
- OC中的方法：类方法和实例方法。
- 初始化方法和便利构造器都返回id类型数据，一个以init开头，一个以类名开头，前者实例方法，后者类方法。

术语和技巧

- 方法的调用 在OC中称为消息表达式[receiver message];开发主要依靠消息机制支撑。

- 方法名的确立方式很简单：去掉“+”或“-”，去掉返回值，去掉参数类型和参数，剩下的部分拼接起来。

- OC语法最大的特点就是“自然语言”，见名知意。

- 自己写代码一定要注意命名规范，勤写注释

OC中的几种类

- 父类：被继承的类
- 子类：继承的类

- 根类：没有父类的类
- 基类：也称基础类或者基本类，介于根类和父类之间，是提供某一功能的核心类，通常是一个抽象类。基类很多，UIView就是一个基类，提供视图相关的全部操作

- 本节重点：熟练使用OC中的方法
- 下节课讲：属性（@property）

- 决定你能否成为高手的因素：不是时间的长短，而是代码量的多少，闻道有先后

今日作业

- 为上次作业中的15个类：
- 添加初始化方法 和 指定初始化方法
- 添加便利构造器方法