无限互联是国内唯一一家**专注**于iPhone和iPad软件开发培训机构，到目前为止为各大公司输送了一大批优秀的iOS高级软件研发人才。随着iOS7系统的发布，我们也在陆续发布**国内首套完整**的iOS开发的视频教程，手把手教您写代码，从入门到熟练再到精通。

高薪就业是检验一家培训机构质量的唯一标准，我们的学员高薪就业是对我们最好的肯定，也是我们前进的最强烈的动力，我们感谢同学们的努力，感谢你们对我们的支持！我们也将免费为你们提供最好的就业后的技术支持！

亲爱的同学们，你们的**高薪就业**才是我们最大的成功！

http://www.iphonetrain.com

流程控制

主讲：汪鸿俊

本节内容

- for、for-in
- while、do-while
- if-else
- switch
- continue、break、fallthrough、return

流程控制

- Swift 流程控制简介
  - Swift提供了类似 C 语言的流程控制结构，包括可以多次执行任务的for和while循环，基于特定条件选择执行不同代码分支的if和switch语句，还有控制流程跳转到其他代码的break和continue语句。
  - 除了 C 语言里面传统的for条件递增（for-condition-increment）循环，Swift 还增加了for-in循环，用来更简单地遍历数组（array），字典（dictionary），区间（range），字符串（string）和其他序列类型。
  - Swift 的switch语句比 C 语言中的更强大。在 C 语言中，如果某个 case 不小心漏写了break，这个 case 就会贯穿（fallthrough）至下一个 case，Swift 无需写break，所以不会发生这种贯穿的情况。case 还可以匹配更多的类型模式，包括区间匹配（range matching），元组（tuple）和特定类型。

無限互联 MOBILE STAR

循环语句

- For循环
  - for循环用来按照指定的次数多次执行一系列语句
  - Swift 提供两种for循环形式：for-in、for条件递增
- for条件递增（for-condition-increment）
  - 用来重复执行一系列语句直到达成特定条件，一般通过在每次循环完成后增加计数器的值来实现
- for-in
  - for-in用来遍历一个区间范围（range），序列（sequence），集合（collection），系列（progression）里面所有的元素
  - 如果不需要知道区间内每一项的值，你可以使用下划线（_）替代变量名来忽略对值的访问

無限互联
MOBILE STAR

循环语句

- While循环
  - Swift 提供两种while循环形式：while、do-while
- while
  - 先判断条件，再执行
- do-while
  - 先执行一次，再判断条件

无限互联网站：　http://www.iphonetrain.com
老师E-mail：　junewhj@qq.com
老师Blog：　http://blog.csdn.net/jaywon

视频讲解过程中如有不妥之处，欢迎大家将信息反馈到我的Email中，我们会努力完善！谢谢各位的支持。

视频持续更新中... 敬请期待！

无限互联是国内唯一一家**专注**于iPhone和iPad软件开发培训机构，到目前为止为各大公司输送了一大批优秀的iOS高级软件研发人才。随着iOS7系统的发布，我们也在陆续发布**国内首套完整**的iOS开发的视频教程，手把手教您写代码，从入门到熟练再到精通。

高薪就业是检验一家培训机构质量的唯一标准，我们的学员高薪就业是对我们最好的肯定，也是我们前进的最强烈的动力，我们感谢同学们的努力，感谢你们对我们的支持！我们也将免费为你们提供最好的就业后的技术支持！

亲爱的同学们，你们的**高薪就业**才是我们最大的成功！

http://www.iphonetrain.com

無限互联 MOBILE STAR

条件语句

- 条件语句
  - Swift 提供两种类型的条件语句：if语句和switch语句
  - 当条件较为简单且可能的情况很少时，使用if语句。而switch语句更适用于条件较复杂、可能情况较多且需要用到模式匹配（pattern-matching）的情况
- If
  - if语句最简单的形式就是只包含一个条件，当且仅当该条件为true时，才执行相关代码，当条件为false时，执行 else 语句

```
let 今天天气好 = true
if 今天天气好 {
    println("我们就去爬山")
} else {
    println("在教室学习")
}
```

条件语句

- Switch
  - switch语句会尝试把某个值与若干个模式（pattern）进行匹配。根据第一个匹配成功的模式，switch语句会执行对应的代码。
  - 当有可能的情况较多时，通常用switch语句替换if语句
  - switch语句都由多个 case 构成，每一个 case 都是代码执行的一条分支
  - 在某些不可能涵盖所有值的情况下，你可以使用默认（default）分支满足该要求，这个默认分支必须在switch语句的最后面

```
let charA: Character = "A"
switch charA {
    case "a", "A":  //如果想匹配多个条件，可以在一个case里面把多个条件用（，）隔开
        println("The letter a") //每一个 case 分支都必须包含至少一条语句
    case "A":
        println("The letter A")
    default:
        println("default")
}
```

無限互联 MOBILE STAR

条件语句switch

- 不存在隐式的贯穿（No Implicit Fallthrough）
  - Swift 中的switch，当匹配的 case 分支中的代码执行完毕后，程序会终止switch语句，而不会继续执行下一个 case 分支。这也就是说，不需要在 case 分支中显式地使用break语句
  - 每个 case 分支都必须包含至少一条语句；如果想匹配多个条件，可以在一个case里面把多个条件用逗号隔开

```
//每一个 case 分支都必须包含至少一条语句；如果想匹配多个条件，可以在一个case里面把多个条件用（，）隔开
let anotherCharacter: Character = "a"
switch anotherCharacter {
    case "a":   //编译报错
    case "A":
        println("The letter A")
    default:
        println("Not the letter A")
}
```

条件语句switch

- 区间范围匹配（Range Matching）
  - case 分支的模式也可以是一个值的范围

```
//使用范围匹配来输出任意数字对应的自然语言格式
let count = 3_000_000_000_000
let countedThings = "stars in the Milky Way"
var naturalCount: String
switch count {
    case 0:
        naturalCount = "no"
    case 1...3:
        naturalCount = "a few"
    case 4...9:
        naturalCount = "several"
    case 10...99:
        naturalCount = "tens of"
    case 100...999:
        naturalCount = "hundreds of"
    case 1000...999_999:
        naturalCount = "thousands of"
    default:
        naturalCount = "millions and millions of"
}
println("There are \(naturalCount) \(countedThings).")
```

条件语句switch

- 匹配元组Tuple
  - 可以使用元组在同一个switch语句中测试多个值。元组中的元素可以是值，也可以是范围。另外，使用下划线（_）来匹配所有可能的值
  - Swift 允许多个 case 匹配同一个值(C语言不支持)。如果存在多个匹配，只会执行第一个被匹配到的 case 分支

```
//使用一个(Int, Int)类型的元组来分类下图中的点(x, y)
let somePoint = (1, 1)
switch somePoint {
    case (0, 0):
        println("(0, 0) is at the origin")
    case (_, 0):
        println("(\(somePoint.0), 0) is on the x-axis")
    case (0, _):
        println("(0, \(somePoint.1)) is on the y-axis")
    case (-2...2, -2...2):
        println("(\(somePoint.0), \(somePoint.1)) is inside the box")
    default:
        println("(\(somePoint.0), \(somePoint.1)) is outside of the box")
}
```

条件语句switch

- 值绑定（Value Bindings）
  - case 分支允许将匹配的值绑定到一个临时的常量或变量，这些常量或变量在该 case 分支里就可以被引用了——这种行为被称为值绑定（value binding）

```
//在一个(Int, Int)类型的元组中使用值绑定来分类下图中的点(x, y)
let anotherPoint = (2, 0)
switch anotherPoint {
    case (let x, 0):    //此时x只是一个占位符，用来临时的获取switch条件中的一个或多个值
        println("on the x-axis with an x value of \(x)")
    case (0, let y):
        println("on the y-axis with a y value of \(y)")
    case let (x, y):
        println("somewhere else at (\(x), \(y))")
}
```

条件语句switch

- Where 附加条件
  - case 分支可以使用where语句来判断额外的条件

```
//case 分支可以使用where语句来判断额外的条件
let yetAnotherPoint = (1, -1)
switch yetAnotherPoint {
    case let (x, y) where x == y:   //把值赋给x,y，并且要求x等于y
        println("(\(x), \(y)) is on the line x == y")
    case let (x, y) where x == -y:
        println("(\(x), \(y)) is on the line x == -y")
    case let (x, y):
        println("(\(x), \(y)) is just some arbitrary point")
}
```

无限互联网站：　http://www.iphonetrain.com
老师E-mail：　junewhj@qq.com
老师Blog：　http://blog.csdn.net/jaywon

视频讲解过程中如有不妥之处，欢迎大家将信息反馈到我的Email中，我们会努力完善！谢谢各位的支持。

视频持续更新中... 敬请期待！

无限互联是国内唯一一家**专注**于iPhone和iPad软件开发培训机构，到目前为止为各大公司输送了一大批优秀的iOS高级软件研发人才。随着iOS7系统的发布，我们也在陆续发布**国内首套完整**的iOS开发的视频教程，手把手教您写代码，从入门到熟练再到精通。

高薪就业是检验一家培训机构质量的唯一标准，我们的学员高薪就业是对我们最好的肯定，也是我们前进的最强烈的动力，我们感谢同学们的努力，感谢你们对我们的支持！我们也将免费为你们提供最好的就业后的技术支持！

亲爱的同学们，你们的**高薪就业**才是我们最大的成功！

http://www.iphonetrain.com

無限互联 MOBILE STAR

控制传递语句

- 控制传递语句（Control Transfer Statements）
  - 控制转移语句改变你代码的执行顺序，通过它你可以实现代码的跳转
  - Swift有四种控制转移语句：
    - continue
    - break
    - fallthrough
    - return

無限互联 MOBILE STAR

控制传递语句

- Continue
  - continue语句告诉一个循环体立刻停止本次循环迭代，重新开始下次循环迭代

```
//continue语句告诉一个循环体立刻停止本次循环迭代，重新开始下次循环迭代。就好像在说“本次循环迭代我已经执行完了,执行下一次吧”
let puzzleInput = "great minds think alike"
var puzzleOutput = ""
for character in puzzleInput {
    switch character {
        case "a", "e", "i", "o", "u", " ":
            continue
        default:
            puzzleOutput += character
    }
}
println(puzzleOutput)
```

無限互联 MOBILE STAR

控制传递语句

- Break
  - break语句会立刻结束整个控制流的执行。当你想要更早的结束一个switch代码块或者一个循环体时，可以使用break语句
- 循环语句中的 break
  - 当在一个循环体中使用break时，会立刻中断该循环体的执行，然后跳转到表示循环体结束的大括号(})后的第一行代码。不会再有本次循环迭代的代码被执行，也不会再有下次的循环迭代产生。
- Switch 语句中的 break
  - 当在一个switch代码块中使用break时，会立即中断该switch代码块的执行，并且跳转到表示switch代码块结束的大括号(})后的第一行代码。

控制传递语句

- 贯穿（Fallthrough）
  - Swift 中的switch不会从上一个 case 分支自动落入到下一个 case 分支，如果确实需要这种效果，可以在每个需要该特性的 case 分支中使用fallthrough关键字

```
let integerToDescribe = 5
var description = "The number \(integerToDescribe) is"
switch integerToDescribe {
    case 2, 3, 5, 7, 11, 13, 17, 19:
        description += " a prime number, and also"
        fallthrough
    default:
        description += " an integer."
}
println(description)
```

無限互联 MOBILE STAR

控制传递语句

- 标签语句（Labeled Statements）
  - 可以使用标签来标记一个循环体或者switch代码块，当使用break或者continue时，带上这个标签，可以控制该标签代表对象的中断或者执行，适合复杂的控制流结构

```
//根据分数评等级，超过100分跳过，遇到负数停止循环
var score = [96, 83, 43, 101, 66, 70, -5, 99]

First: for s in score {      //定义标签First
    switch s/10 {
        case 10:
            continue First  //使用标签
        case 9:
            println("\(s)分为优秀")
        case 8:
            println("\(s)分为良好")
        case 6...7:
            println("\(s)分为中等")
        case 0:
            break First      //使用标签，终止for循环。如果这里没使用标签，break的将是switch
        default:
            println("\(s)分为没及格")
    }
}
```

无限互联网站：　http://www.iphonetrain.com
老师E-mail：　junewhj@qq.com
老师Blog：　http://blog.csdn.net/jaywon

视频讲解过程中如有不妥之处，欢迎大家将信息反馈到我的Email中，我们会努力完善！谢谢各位的支持。

视频持续更新中... 敬请期待！

无限互联是国内唯一一家**专注**于iPhone和iPad软件开发培训机构，到目前为止为各大公司输送了一大批优秀的iOS高级软件研发人才。随着iOS7系统的发布，我们也在陆续发布**国内首套完整**的iOS开发的视频教程，手把手教您写代码，从入门到熟练再到精通。

高薪就业是检验一家培训机构质量的唯一标准，我们的学员高薪就业是对我们最好的肯定，也是我们前进的最强烈的动力，我们感谢同学们的努力，感谢你们对我们的支持！我们也将免费为你们提供最好的就业后的技术支持！

亲爱的同学们，你们的**高薪就业**才是我们最大的成功！

http://www.iphonetrain.com