Swift 3 的第一印象

变量和常量 (https://www.boxueio.com/series/swift-up-and-running/ebook/1)

忘记旧有的"C风格"字符串吧 (https://www.boxueio.com/series/swift-up-and-running/ebook/106)

整数和浮点数

Back to series (/series/swift-up-and-running)

Swift里，数字分为整数（例如：1 / 10 / 100 / 1000等）和浮点数（例如：3.14 / 1.44 / 2.71等）。我们先来看整数。

整数 - Int & UInt

根据一个整数变量占据的内存空间（8 / 16 / 32 / 64-bit）以及整数是否带有符号（Unsinged）,Swift一共定义了8种不同的整数类型：

- Int(8 / 16 / 32 / 64)
- UInt(8 / 16/ 32/ 64)

第一行的四个类型，分别表示8 / 16 / 32 / 64-bit的有符号整数，第二行则是对应的无符号整数类型。

但通常，我们不会直接在代码中使用这些具体的整数类型，我们只使用 `Int` 来定义有符号整数，使用 `UInt` 来定义无符号整数。Swift编译器会根据目标编译平台，把 `Int` 或 `UInt` 转换成对应的整数类型。例如：

在我们的64位平台上，我们分别使用 `min` 和 `max` 方法，来查看 `Int` 和 `Int64` 可以表达的数值范围：

```
Int.min   // -9223372036854775808
Int.max   // 9223372036854775807

Int64.min // -9223372036854775808
Int64.max // 9223372036854775807
```

通过上面的对比，我们就能发现， `Int` 和 `Int64` ，它们可以表达的数值范围，是一样的。

整数的常用表达方式

在Swift里，我们可以使用多种方式来表达一个整数。包括使用10进制、16进制、8进制、2进制：

```
//: Number literal

let fifteenInDecimal = 15
let fifteenInHex = 0xF
let fifteenInOctal = 0o17
let fifteenInBinary = 0b1111
```

以及，我们可以在数字中，使用分隔符：

```
//: Number literal
let million = 1_000_000
```

> “除非我们所在的硬件平台有特别明确的需求需要我们使用 `UInt` 来定义无符号整数， 我们总是应该尽可能使用 `Int` 来表达所有的整数类型，哪怕我们确定一个整数一定是一个非负数。 这会给我们减少很多不必要的类型转换的麻烦。”

浮点数 Float & Double

在Swift里，根据可以表达的精度范围，有两种不同的浮点数类型：

- `Float` : 最多表达6位精度的浮点数；
- `Double` : 至少可以表达15位精度的浮点数；

我们用 `print` 分别打印 `Float` 和 `Double` ：

```
var oneThirdInFloat: Float = 1/3
var oneThirdInDouble: Double = 1/3

print(oneThirdInFloat)  // 0.333333
print(oneThirdInDouble) // 0.333333333333333
```

除了使用常规的十进制表达浮点数之外，我们还可以使用科学计数法。例如，表示浮点数PI：

```
var PI = 0.314e1
PI = 314e-2
```

> “如不是有明确的需求，我们应该统一使用 Double 来定义浮点数。”

和数字有关的Type Inference

在Swift里，我们使用一个整数，编译器会把它推导成 Int ，使用一个浮点数，编译器会把它推导成 Double ，例如：

```
var three = 3
type(of: three) // Int.Type

var zeroPointForteen = 0.14
type(of: zeroPointForteen) // Double.Type
```

我们可以使用 type(of:) 来查看一个变量的类型，从上面的结果我们就可以看到 three 的类型是 Int ， zeroPointForteen 的类型是 Double 。

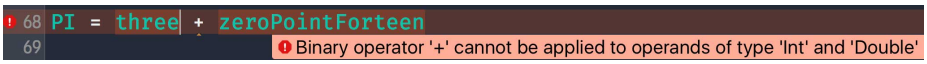
在Swift里，我们可以把不同类型数字的字面值直接进行运算：

```
PI = 3 + 0.14
type(of: PI)
```

我们可以看到，整数值3和浮点数0.14可以直接相加，Swift把相加的结果转换成一个 Double 。但是，在Swift里，我们不能把不同数字类型的变量直接进行算数运算，例如：

```
PI = three + zeroPointForteen
```

就会看到下面这样的编译错误：



当对变量进行算数运算的时候，所有变量的类型必须是相同的，如果变量类型不同，我们必须明确将其中的一些变量进行类型转换。像这样：

```
PI = Double(three) + zeroPointForteen
```

从上面的代码可以看到，我们使用： Double(Value) 把一个 Int 类型的 three ，"转换"成了浮点数。在这里，之所以我们要对转换加引号，是因为我们并没有真的把 three 的类型从 Int 转换成 Double ，而是用 three 的值，初始化了一个新的值为3的 Double ，并用这个新的 Double 和 zeroPointForteen 相加而已。

What's next?

以上就是我们这一节的内容。在了解了不同整数和浮点数定义、用法以及type inference规则之后，在下一节，我们将深入一个略显复杂，但是又非常重要的类型： String 。为了可以在unicode环境下正常工作，Swift中的 String 在设计上进行了诸多方面的考量。我们在接下来的几节中，就来向大家介绍这套复杂系统里最重要的几个概念。

⏮ 变量和常量 (https://www.boxueio.com/series/swift-up-and-running/ebook/1)

忘记旧有的"C风格"字符串吧 ⏭ (https://www.boxueio.com/series/swift-up-and-running/ebook/106)

职场漂泊的你，每天多学一点。

从开发、测试到运维，让技术不再成为你成长的绊脚石。我们用打磨产品的精神去传播知识，把最新的移动开发技术，通过简单的图表， 清晰的视频，简明的文字和切实可行的例子一一向你呈现。让学习不仅是一种需求，也是一种享受。

泊学动态

一个工作十年PM终创业的故事（二） (https://www.boxueio.com/after-the-full-upgrade-to-swift3)
Mar 4, 2017

人生中第一次创业的"10有" (https://www.boxueio.com/founder-chat)
Jan 9, 2016

猎云网采访报道泊学 (http://www.lieyunwang.com/archives/144329)
Dec 31, 2015

What most schools do not teach (https://www.boxueio.com/what-most-schools-do-not-teach)
Dec 21, 2015

一个工作十年PM终创业的故事（一） (https://www.boxueio.com/founder-story)
May 8, 2015

泊学相关

关于泊学 ›

加入泊学 ›

泊学用户隐私以及服务条款 (HTTPS://WWW.BOXUEIO.COM/TERMS-OF-SERVICE)

版权声明 (HTTPS://WWW.BOXUEIO.COM/COPYRIGHT-STATEMENT)

联系泊学

Email: 10[AT]boxue.io (mailto:10@boxue.io)

QQ: 2085489246

友情链接 SwiftV (http://www.swiftv.cn) | Seay信息安全博客 (http://www.cnseay.com) | Swift.gg (http://swift.gg/) | Laravist (http://laravist.com/) | SegmentFault (https://segmentfault.com) | 靛青K的博客 (http://blog.dianqk.org/)