NSCharacterSet

Mattt Thompson撰写、 Ricky Tan翻译、 发布于2012年9月17日

正如之前提前过的，基础类库（Foundation）拥有最好的、功能也最全的string类的实现。

但是仅当程序员熟练掌握它时，一个string的实现才是真的好。所以本周，我们将浏览一些基础类库的string生态系统中经常用到且用错的重要组成部分：`NSCharacterSet`。

> 如果你对什么是字符编码搞不清楚的话（即使你有很好的专业知识），那么你应该抓住这次机会反复阅读Joel Spolsky的这篇经典的文章"The Absolute Minimum Every Software Developer Absolutely, Positively Must Know About Unicode and Character Sets (No Excuses!)"。在头脑中保持新鲜感将对你理解我们将要探讨的话题非常有帮助。

`NSCharacterSet`，以及它的可变版本`NSMutableCharacterSet`，用面向对象的方式来表示一组Unicode字符。它经常与`NSString`及`NSScanner`组合起来使用，在不同的字符上做过滤、删除或者分割操作。为了给你提供这些字符是哪些字符的直观印象，请看看`NSCharacterSet` 提供的类方法：

- `alphanumericCharacterSet`
- `capitalizedLetterCharacterSet`
- `controlCharacterSet`
- `decimalDigitCharacterSet`
- `decomposableCharacterSet`
- `illegalCharacterSet`
- `letterCharacterSet`
- `lowercaseLetterCharacterSet`
- `newlineCharacterSet`
- `nonBaseCharacterSet`

- `punctuationCharacterSet`
- `symbolCharacterSet`
- `uppercaseLetterCharacterSet`
- `whitespaceAndNewlineCharacterSet`
- `whitespaceCharacterSet`

与它的名字所表述的相反，`NSCharacterSet` 跟 `NSSet` 一点关系都*没有*。

虽然底层实现不太一样，但是 `NSCharacterSet` 在概念上跟 `NSIndexSet` 还*有点*相似的。`NSIndexSet`，之前提到过，表示一个有序的不重复的无符号整数的集合。Unicode字符跟无符号整数类似，大致对应一些拼写表示。所以，一个 `NSCharacterSet +lowercaseCharacterSet` 字符集与一个包含97到122范围的 `NSIndexSet` 是等价的。

现在我们对理解 `NSCharacterSet` 的基本概念已经有了少许自信，让我们来看一些它的模式与反模式吧：

去掉空格

`NSString -stringByTrimmingCharactersInSet:` 是个你需要牢牢记住的方法。它经常会传入 `NSCharacterSet +whitespaceCharacterSet` 或 `+whitespaceAndNewlineCharacterSet` 来删除输入字符串的头尾的空白符号。

需要重点注意的是，这个方法 *仅仅* 去除了 *开头* 和 *结尾* 的指定字符集中连续字符。这就是说，如果你想去除单词之间的额外空格，请看下一步。

挤压空格

假设你去掉字符串两端的多余空格之后，还想去除单词之间的多余空格，这里有个非常简便的方法：

Objective-C

```
NSString *string = @"Lorem    ipsum dolar   sit  amet.";
string = [string stringByTrimmingCharactersInSet:[NSCharacterSet whitespaceCharacterSet]];

NSArray *components = [string componentsSeparatedByCharactersInSet:[NSCharacterSet whitespaceCh
components = [components filteredArrayUsingPredicate:[NSPredicate predicateWithFormat:@"self <>

string = [components componentsJoinedByString:@" "];
```

首先，删除字符串首尾的空格；然后用 `NSString -componentsSeparatedByCharactersInSet:` 在空格处将字符串分割成一个 `NSArray`；再用一个 `NSPredicate` 去除空串；最后，用 `NSArray -componentsJoinedByString:` 用单个空格符将数组重新拼成字符串。注意：这种方法仅适用于英语这种用空格分割的语言。

现在看看反模式吧。请先看看 the answers to this question on StackOverflow。

在写这篇文章的时候，排行第二的正确答案有 58 个顶和 2 个踩。排行第一的有 84 个顶和 24 个踩。

如今，排名第一的答案却不是正确答案是不太正常的，但是这个问题已经破了不重复答案数（10个）的记录，同时也破了不重复、完全错误的答案数（9个）的记录。

言归正传，这里有 9 个错误答案：

- "Use `stringByTrimmingCharactersInSet`" - *正如你所知道的，它只去掉首尾的空格。*
- "Replace ' ' with "" - *这个去除了**所有**的空格，劳而无功。*
- "Use a regular expression" - *有点用，但它没有处理首尾的空格。用正则表达式有点大材小用了。*
- "Use Regexp Lite" - *说真的，正则表达式真心没必要。同时为了这点功能增加第三方库很不值。*
- "Use OgreKit" - *同上，添加了第三方库。*
- "Split the string into components, iterate over them to find components with non-zero length, and then re-combine" - *很接近了，但是 `componentsSeparatedByCharactersInSet:` 已经让遍历变得没必要。*
- "Replace two-space strings with single-space strings in a while loop" - *错误且浪费计算资源。*
- "Manually iterate over each `unichar` in the string and use `NSCharacterSet -characterIsMember:`" - *用了一个复杂到让人吃惊的程度的方法，却忘了标准库中已经有方法可以用。*
- "Find and remove all of the tabs" - *有谁提到了制表符了？不过还是谢谢了吧。*

我个人并不是想责怪回答问题的人——只是指出完成这个功能有多少种不同的方法，而这些方法有多少是完全错误的。

字符串分词

不要用 `NSCharacterSet` 来分词。 用 `CFStringTokenizer` 来替代它。

你用 `componentsSeparatedByCharactersInSet:` 来清理用户输入是可以谅解的，但是用它来做更复杂的事情，你将陷入痛苦的深渊。

为什么？请记住，语言并不是都用空格作为词的分界。虽然实际上以空格分界的语言使用非常广泛。但哪怕只算上中国和日本就已经有十多亿人，占了世界人口总量的 16%。

……即使是用空格分隔的语言，分词也有一些模棱两可的边界条件，特别是复合词汇和标点符号。

以上只为说明：如果你想将字符串分成有意义的单词，那么请用 `CFStringTokenizer` （或者 `enumerateSubstringsInRange:options:usingBlock:`）吧。

从字符串解析数据

`NSScanner` 是个用以解析任意或半结构化的字符串的数据的类。当你为一个字符串创建一个扫描器时，你可以指定忽略哪些字符，这样可以避免那些字符以各种各样的方式被包含到解析出来的结果中。

例如，你想从这样一个字符串中解析出开门时间：

Text

```
Mon-Thurs:  8:00 - 18:00
Fri:        7:00 - 17:00
Sat-Sun:    10:00 - 15:00
```

你会 `enumerateLinesUsingBlock:` 并像这样用一个 `NSScanner` 来解析：

Objective-C Swift

```
let skippedCharacters = NSMutableCharacterSet()
skippedCharacters.formIntersectionWithCharacterSet(NSCharacterSet.punctuationCharacterSet())
skippedCharacters.formIntersectionWithCharacterSet(NSCharacterSet.whitespaceCharacterSet())

string.enumerateLines { (line, _) in
    let scanner = NSScanner(string: line)
    scanner.charactersToBeSkipped = skippedCharacters

    var startDay, endDay: NSString?
    var startHour: Int = 0
    var startMinute: Int = 0
    var endHour: Int = 0
    var endMinute: Int = 0

    scanner.scanCharactersFromSet(NSCharacterSet.letterCharacterSet(), intoString: &startDay)
    scanner.scanCharactersFromSet(NSCharacterSet.letterCharacterSet(), intoString: &endDay)

    scanner.scanInteger(&startHour)
    scanner.scanInteger(&startMinute)
    scanner.scanInteger(&endHour)
```

```
        scanner.scanInteger(&endMinute)
    }
```

我们首先从空格字符集和标点符号字符集的并集构造了一个 `NSMutableCharacterSet`。告诉 `NSScanner` 忽略这些字符以极大地减少解析这些字符的必要逻辑。

`scanCharactersFromSet:` 传入字母字符集得到每项中一星期内的开始和结束（可选）的天数。`scanInteger` 类似地，得到下一个连续的整型值。

`NSCharacterSet` 和 `NSScanner` 让你可以快速而充满自信地编码。这两者真是完美组合。

`NSCharacterSet` 是基础类库中字符串处理系统中的一员，可能是最容易被用错或是误解的一员。在脑中记住这些模式与反模式，你将不仅能做一些很有用的诸如管理空格及从字符串中读信息之类的事情，更重要的是，你将避免误入歧途。

如果“不出错”对一个 NSHipster 来说不是最重要的事情，那我也不想成为正确的了！

> Ed. Speaking of (not) being wrong, the original version of this article contained errors in both code samples. These have since been corrected.

作者

Mattt Thompson

Mattt Thompson (@mattt) is the creator & maintainer of AFNetworking and other popular open-source projects, including Postgres.app, ASCIIwwdc and Nomad.

翻译者

Ricky Tan

Ricky Tan 是iZJU iOS 版 3.1.3 以前版本及后台、Xcode 插件 RTImageAssets 的开发者，另有浙大网址导航 iStudy Chrome 插件等。更多请移步。

下一篇文章

UICollectionView

从现在起，UICollectionView凭一己之力改变我们将要设计和开发iOS应用的方式。这并不是说，collection views是未知或模糊的。作为一个NSHipster，不仅仅是知道名不见经传的石头，更多是在它们家喻户晓、售罄一空之前就知道有前途。

相关文章

- UIAlertController
- NSIndexSet
- NSOrderedSet
- NSCoding / NSKeyedArchiver

本站文章由 Croath Liu 、 、 Delisa Mason 、 Jack Flintermann 、 Mattt Thompson 、 、 Mike Lazer-Walker 、 Natasha Murashev 和 Nate Cook 撰写、 Andrew Yang 、 April Peng 、 Bob Liu 、 Candyan 、 Chester Liu 、 Croath Liu 、 Daniel Hu 、 David Liu 、 GWesley 、 Henry Lee 、 JJ Mao 、 Lin Xiangyu 、 Ricky Tan 、 Sheldon Huang 、 Tiny Tian 、 Tony Li 、 Yifan Xiao 、 Yu Jin 和 Zihan Xu 翻译。