

iOS 程序员的 Ruby 初体验



在技术浪潮不断更新发展的时代下，身为程序员的我们不断的在学习新技术，不断的更新自己的技能，不断的和伙伴们一起成长，梦想着有天能航行到技术的顶端。就像海贼王里最恶世代们一样，只有不断的挑战，不断的成长抱团打倒 BOSS，才可能到达终点得到 ONE PIECE。

前言

一开始去学习 [Ruby](#) 只是想了解 Podfile 里写的东西到底是什么，到后面对 CocoaPod 和 fastlane 源码和原理的好奇，再到想通过学习到的东西改善自己的工作流。

本篇文章记录了自己从小白到 Ruby 简单入门的旅程。

Ruby 学习

做为一个 Ruby 小白，没人教，啥都不懂该如何去学习这样一门新的语言。没有别的，只能靠自己程序员的大脑和折腾。

对于新的编程语言我需要去了解他的语法，所以找到这几个网址：

1. [ruby基础教程（中文第四版）](#)

2. [Ruby | 菜鸟教程](#)
3. [Ruby China社区](#)

对于 Ruby 语法刚接触有几个地方有较深的映像：

1. 完全面向对象：任何东西都是对象，没有基础类型。平时我们使用的1, 2, 3, 4, 5, 6在 Ruby都是 ‘Fixnum’ 类的实例对象，有着属于自己的方法。

```
1.to_s    // 将 1 转为字符串
```

2. 代码块之间不是用{}来表示代码块的范围 而是 end 来表示块的结尾。比如定义一个方法 OC的做法是：

```
- (void) method {  
    这里是方法实现代码  
}
```

而在ruby里：

```
def method  
    这里是方法实现代码  
end
```

3. 可以用换行替换“;”，即代码中可以省略“;”。

OC中：

```
- (void) method {  
    第一行代码;  
    第二行代码;  
}
```

Ruby中：

```
def method  
    第一行代码  
    第二行代码  
end
```

4. 每行代码都有返回值。

我们在 irb 中输入下面代码（irb内置在shell的ruby解释器）。

```
irb(main):001:0> puts 'hello, world'
hello, world
=> nil
```

puts 'hello, world'意为打印'hello, world'字符串。所以我们得到的第一个输出为正常方法执行的效果，打印出'hello, world'字符串。而第二行

```
=> nil
```

则是意味着该行代码的放回值为nil。

如果我们的代码是给一个变量，那这行代码的返回值将是变量的值，如下：

```
irb(main):002:0> a = 1
=> 1
```

5. 一个有返回值的方法中如果没有return，则最后一行代码的返回值就是该方法的返回值。如：

```
def method
  a = 1
end
```

该方法的返回值为 1 。

实践

学习任何编程语言最快的方式就是上手敲。

在了解Ruby大致的语法后，便需要开始寻找工具来进行 Coding 实战。最后基于工具的学习时间成本先挑选了 irb 和 Sublime Text

irb

全名“interactive ruby”，一个命令行工具，也可以看作是Ruby的解释器。

MacOs下系统自带，打开终端输入：

```
irb
```

即可启动。启动后就可以输入Ruby代码的，注意每输入一行代码，irb都会给你返回该行代码的返回值，如刚刚出示的代码log

```
irb(main):001:0> puts 'hello, world'
hello, world
=> nil
```

Sublime Text

Sublime Text 是一套很好跨平台的文字编辑器，如果想要编写很长的 Ruby 代码在 irb 不是很方便，所以选择了它。（[Sublime Text 维基百科](#)）

阅读源码

通过一段时间的实践理解，大致对 Ruby 有初步的认识，可以尝试阅读 [fastlane](#), [cocoapod](#) 源码来对它们进行理解了（[fastlane源码](#)，[CocoaPods源码](#)）。

通过源码的阅读对他们大致总结为：通过 Ruby 来调用终端命令。

这里尤其是fastlane，读 fastlane 的很多的工具集（[actions](#)）的源码你会发现里面基本是提供了各种 shell 命令的合集。如 action：[pod_lib_lint](#) 的 [源码](#)中可看到：

```
command << "pod lib lint"

if params[:verbose]
  command << "--verbose"
end

if params[:sources]
  sources = params[:sources].join(",")
  command << "--sources='#{sources}'"
end

if params[:allow_warnings]
  command << "--allow-warnings"
end
```

源码中可以看到里面提供了我们平时在 Shell 中主动输入的一些命令。

所以如果你想做一个 fastlane 的 action，其实很简单，里面多数都是往command 数组中插入命令，你只要注意遵从 fastlane 对你们的要求格式进行提交你的 action 代码审核就行。

另外我们在使用 fastlane 进行打包的时候，能清楚的看到其答应出所使用的 xcodebuild 的命令。

其实这些也可以通过 Shell 脚本来实现，那为什么 fastlane，Cocoapods 是用 Ruby 来实现的，笔者猜想可能是因为 fastlane，Cocoapods 本身就是基于 Ruby 开发中的工具 rake 和 bundle 思想而开发出来的吧。

继续实践

程序员就是要折腾自己，既然大致原理是通过 Ruby 执行 Shell 命令，那他们能写很多工具来帮助程序员开发，我自己能写一个吗？不求写出来的高大上，但求学到的东西能实践应用，写出来的东西能方便自己。

于是就有了这个项目 [pod_updater](#)，这个项目意在帮助自己摆脱每次提交自己的私有 pod 库时，总是要在终端输入一堆 git 命令然后在 push 到 podspec 仓库等一系繁琐的操作。

开始实践

首先感觉 irb 不够用了，不能对代码进行调试，而 Sublime Text 只是个编辑器，于是又找到了 RubyMine。

RubyMine 是一个用于 Ruby 开发的 IDE，可见[资料](#)。

1. 有了 IDE 还不够，有时我们需要使用些成熟的三方库，所以我们需要了解这些 Gem，bundle，Gemfile。

Gem：Ruby 應用程式或程式庫。可以理解为Framework。

bundle：是管理、解決 gem 之間相依性的工具，它會根據 Gemfile 裡頭的設定自動下載和安裝 gem，而且幫你解決相依問題。可以理解他为iOS开发中的Cocoapod。

Gemfile：理解为podfile即可，用于描述程序的相关库安装与依赖关系。

[更多资料](#)

2. Ruby 执行shell 命令

Ruby执行 shell 命令方式有很多种，可见[资料](#)，而我选择是的 IO.popen 方法。

```
$ irb
>> IO.popen("date") { |f| puts f.gets }
Wed Jul  4 22:02:31 CST 2012
=> nil
```

3. 接着就是各种资料查找和折腾，最后模仿 fastlane 和 CocoaPods 将自己的工具做成 gem

的形式，于是我的第一个 RubyGem `pod_updater` 诞生了，使用效果如下：

```
WZ>HelloKit — git-remote-https • pod_updater -v 5.5.1 rvm_bin_path=/Users/qwkj/.rvm/bin TERM_PROGRAM=Apple_Termi...
git-remote-https • pod_updater -v 5.5.1 rvm_bin_path=/Users/qwkj/.rvm...minal GEM_HOME=/Users/qwkj/.rvm/gems/ruby-2.2.0 SHELL=/bin/bash
s.summary      = "The package of useful tools, include categories and classes"
s.homepage     = "https://github.com/hwzss/WZ>HelloKit"
s.license      = { :type => "MIT", :file => "LICENSE" }
s.authors      = { 'hwzss' => '1833361588@qq.com' }
s.platform     = :ios, "7.0"
s.source       = { :git => "https://github.com/hwzss/WZ>HelloKit.git", :tag => s.version
}
# s.source      = { :git => "/Users/qwkj/Documents/WZ_GitHub/WZ>HelloKit" }
s.source_files = 'WZ>HelloKit', 'WZ>HelloKit/**/*.{h,m}'
s.requires_arc = true
end
cd /Users/qwkj/Documents/WZ_GitHub/WZ>HelloKit && git add . && git commit -m "for pod version:5.5.1" && git push && git tag -a 5.5.1 -m "for pod version:5.5.1" && git push --tags
[master c4631e1] for pod version:5.5.1

1 file changed, 1 insertion(+), 1 deletion(-)

Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 293 bytes | 0 bytes/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/hwzss/WZ>HelloKit.git
   0b03e11..c4631e1  master -> master
█
```

总结

对于一门新语言，下手实践才是真道理。

对于自己，学习语言最大的动力莫过于他能帮助自己更好的生活和工作。

今天是 2018 除夕，祝各位程序员同胞新年快乐，过年多吃肉 😊。

学习资料

1. [ruby基础教程（中文第四版）](#)
2. [Ruby | 菜鸟教程](#)
3. [Ruby China社区](#)
4. [簡單筆記一下 RubyGem, Gem, RVM, Gemfile, bundler](#)
5. [Mastering ruby blocks in less than 5 minutes](#)
6. [在 Ruby 中执行 Shell 命令的 6 种方法](#)
7. [如何开发一个自己的 Gem](#)
8. [How to Use OptionParser: Ruby's Powerful Parsing Tool](#)