SRake DSL

SenseTime Ruby Training Program

问题

DSL (Domain Specific Language)是一种专注于某一特定领域的语言，使用通用语言（如Ruby本身）当然可以得到与DSL相同的功能。 但是这样会产生大量繁琐的代码并导致大量的领域知识被隐藏在通用语言构造中（如for循环，if条件，方法调用等）。

由于Ruby语言语法的灵活性和其强大的元编程能力，Ruby社区开发者设计了各种不同用途的DSL：

- Rakefile: 与GNU make功能类似的工具，解决任务依赖问题
- RSpec: 使用类似自然语言的方式编写Unittest
- Gemfile: Ruby Gem描述文件
- Sinatra: 简洁的编写Webserver的DSL
- Rails: 万能的Web框架，Ruby的杀手应用。实际上可以看作是一种DSL，如ActiveRecord中的Model描述

使用Ruby编写DSL的好处是，DSL更偏向与“描述”和“配置”，隐藏特定领域中的通用逻辑。由于DSL本身也是 合法的Ruby文件，DSL中可以使用普通Ruby的语法和第三方库实现逻辑和功能。

本次作业要求实现一个简化的Rake DSL，`SRake`。`SRake`是Rakefile语法的子集，以下是一个 简单的`SRake`文件：

```
task :default => :test3

desc 'task1'
task :test1 do
  sh 'echo task1'
end

desc 'task2'
task :test2 => :test1 do
  sh 'echo task2'
end

desc 'task3'
task :test3 => [:test1, :test2] do
  sh 'echo task3'
end

task :test4 => :test5 do
  sh 'echo task4'
end
```

参考输出：

```
./simplerake.rb test/test1.rake
task1
task2
task3
```

```
./simplerake.rb -T test/test1.rake
test1           # task1
test2           # task2
test3           # task3
test4           #
```

请参考给出的测试输入文件，了解你的DSL解析器需要处理的各种情况。 实现本次作业基本要求代码量约为100行。

评分标准

你的SRake实现应正确解析/运行我们给出的所有测试输入文件。注意*不需要* 实现 所有Rakefile支持的语法。

项目要求

- 可不考虑用户给出的SRakefile中，任务循环依赖的问题，如：

```
task :test1 => :test2
task :test2 => :test1
```

- 使用如下命令行运行你的SRake解析器：

```
Usage: ./simplerake.rb [options] srake_file [task]
    -T                               list tasks
    -h                               print help
```

- SRake DSL需要支持的语法：
    - `desc 'some description'` 设置当前`task`的描述，将出现在`simplerake -T`中
    - `task` 需要支持第一节中给出实例文件的几种形式，task名字的类型都是Symbol
    - `sh 'echo hello'` 运行指定的Shell命令
    - 当用户不指定目标task时，自动运行`default`目标
- 你应该把DSL的主要实现放到`SimpleRake`模块中
- 项目中可能会用到的语法点：
    - OptionParser
    - `class`及变量/闭包作用域
    - Proc/Block
    - `instance_eval`

加分项

- 如果用户给出的Rakefile中任务依赖有环，给出报错信息
- 用`dot`命令画出任务依赖关系图

演示和作业提交

在Github上建立一个名为`sensetime_hw`的仓库，其中建立一个名为`002-simple-rake`的文件夹，其中 最少包含两个文件：

- simplerake.rb
- README.md

请使用Markdown格式编写你的文档．

Reference

- http://jasonseifer.com/2010/04/06/rake-tutorial
- http://jroller.com/rolsen/entry/building_a_dsl_in_ruby