## 软件分析技术大作业

**算法设计和实现部分**

1. **算法设计**

在Marionette框架中，网页整体由许多view组合而成。各个view之间和view内部的逻辑由许多的JavaScript文件给定。在Marionette打包的过程中，Marionette将这些文件整理统一，制作成单个的html文件供网页显示。

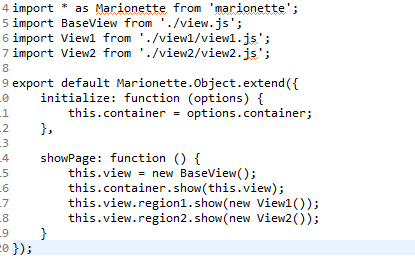
因此，要想了解各个SCSS文件具体影响到了哪些view和哪些view中的哪些元素，就要模拟Marionette打包的过程，生成一个简化版的html代码。之后，我们对用户选中的SCSS中的语句做分，首先获得它的文件位置和对应位置的Style类型，之后找到相应的Selector，从DOM树中找到对应元素并显示。

1. **代码实现**

本次项目的算法部分代码在buildTreeForCase2.java中实现。

* 1. 建立DOM树

网页由许多view组成，统一由controller.js控制。View由Backbone.View定义，指定其模板hbs文件。每个view有一个hbs设定了view的html模板。实例如下：



Controller.js

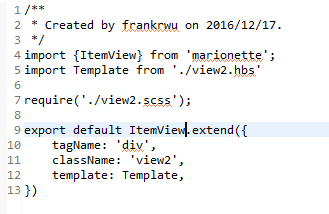
上面的Controller例子首先引入了view1.js和view2.js两个文件，并且在后面使用了这两个页面在region1和region2中。其中，region1和region2在layout.hbs中定义，controller将这两个label中添加相应的view代码。

具体在实现时，我们使用getController函数通过entry.js找到项目中的controller路径，之后通过BuildTree函数完成建树过程。

BuildTree函数的工作过程是这样的。首先通过findImport函数找到引用的javascript文件的路径，之后调用shell函数解析各个对应hbs并包“壳”。



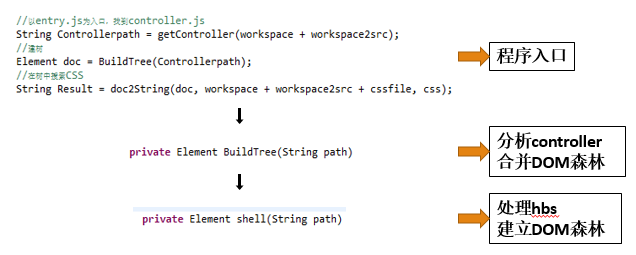
View2.hbs



View2.js

如上面实例所示，每个view的hbs的结构和一般html文件的结构是一致的。view2.js如上所示，为每个view外面添加了一些html节点，比如上面添加的就是classname为view2的div节点。因此套“壳”就是在解析hbs文件后，将其加上一层合适的div标签作为html节点返回，另外为了得到各个文件对应的地址，将路径作为一个atrribute加入div中。这部分工作就是在shell函数中完成的。

总的来说，上面的各个过程汇总起来，我们就完成了建立DOM树的过程。大概流程如下：



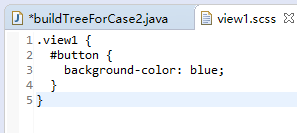
获得的dom树简单展示如下：



* 1. 分析SCSS

这部分的工作主要是解析插件返回的SCSS文件，在这些文件中找到插件返回的SCSS语句。之后找到对应的selector，从DOM树中找到selector对应的所有子节点，这些子节点都可能被相关的SCSS语句影响。

这部分功能在findSelector函数中实现。首先通过正则匹配大括号递归找到每个style和其对应的selector，比如在下图的SCSS代码中，background-color的selector就是view1中的button标签。我们记录下这些style-selector对并返回。



* 1. 结果返回

这部分我们得到了刚才的DOM树和Selector，功能就比较简单了，只需要从树中找到selector对应的所有子节点就可以了。这部分功能在doc2String函数中实现，最后将所有的结果整理成给出规则的字符串，规则为：对每一个selector进行统计每一行是一个tag和其路径，用逗号隔开，实例如下：

