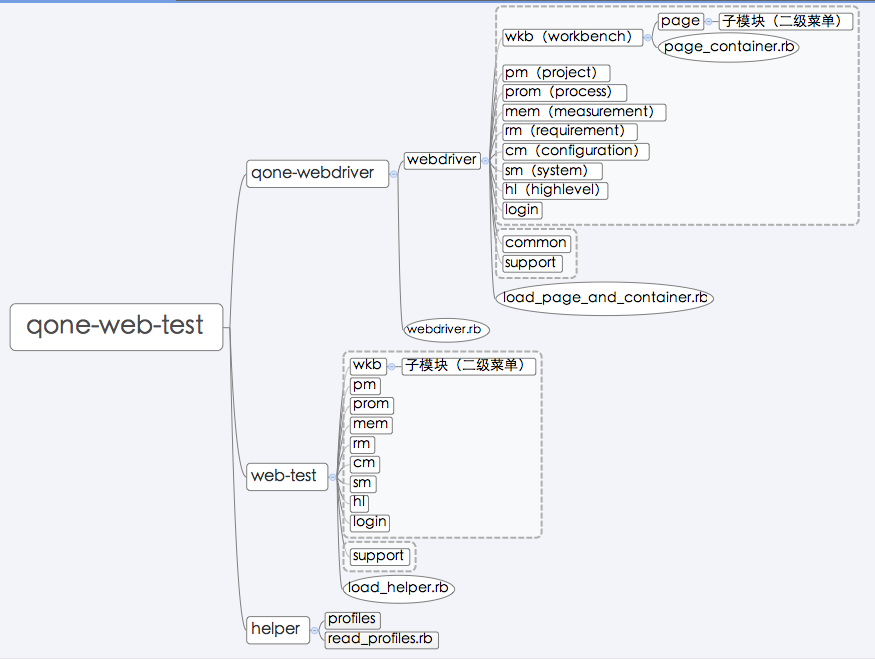
Qone自动化测试框架结构说明

采用Page Object Pattern设计模式，以每个Page页面为对象，Page对象包括页面的元素和页面的行为（功能），写测试用例时，直接调用Page对象的行为即可。这种设计可以把测试代码和页面的元素、行为完全分开，便于以后的维护和扩展。

如图所示，框架主要分成3部分，第一部分为qone-webdriver，主要是页面的行为，第二部分为web-test，主要是测试用例，第三部分是前2个模块共享的信息，如国际化文件。



1. qone-webdrvier

此部分主要是以selenium-webdriver为基础封装而成，它分为：页面（page）、公用（common）、支持（support）、辅助部分

* 1. 公用（common）

此包主要是PageObject和PageContainer的父类，或者说是“抽象部分”

1.1.1 browser\_container：浏览器容器类

包括浏览器的信息

1.1.2 base\_page：父页面类

继承browser \_container，主要作用是约定页面的公共行为

1.1.3 page\_container：页面容器父类

继承browser \_container，主要作用是约定页面容器类的公共行为

1.1.4 page\_container\_factory：页面容器工厂类

生产不同模块的页面容器

* 1. 页面（page）

1.2.1 PageObjectClass

每个页面是一个page对象，继承BasePage类，去实现具体页面的行为

1.2.2 PageContainerClass

每个大模块（如pm、sm）会有一个自己的PageContainer，用来管理这个模块下的PageObject

* 1. 支持（support）

1.3.1 common module

存放n个PageObject之间公用的部分，也就是会把公用部分封装成module，被PageObject重用

1.3.2 utils

存放一些公共的utils帮助类

* 1. 辅助部分

1.4.1 load files

主要作用是引用文件，如load\_page\_and\_container.rb，这个类的作用是自动扫描并引用xx\_page.rb和xx\_container.rb文件

1.4.2 webdirver.rb（供外部调用）

主要作用是创建每个模块自己的PageContainer，供测试用例使用，可以说是qone-webdriver的入口、对外访问方法

1. web-test

测试部分，主要分为测试用例部分和辅助部分

* 1. 测试用例部分

此部分主要是用RSpec行为驱动测试框架为基础封装而成，会按照大模块-子模块（二级菜单）/测试用例的结构去写用例，每个用例是针对于一个页面对象的。

* 1. 辅助部分
     1. 支持（support）

主要是测试用例中需要用到的辅助类或者说是帮助类，如table组件、qone部门树组件。

* + 1. load files

主要用来解决ruby引用文件的问题

1. helper
   * 1. 配置文件 profiles

主要是国际化文件和登录信息文件组成，到后期应该还会加入其他配置文件，测试代码中不允许出现中文信息。