****

**接口自动化测试**

**技术方案**

|  |  |  |  |
| --- | --- | --- | --- |
| 文档状态 | [√] 初稿 | 当前版本 |  |
| [ ] 评审通过 |
| [ ] 修改 |
| [ ] 发布 |
| [ ] 作废 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日期** | **版本** | **修订内容描述** | **作者** | **审核人** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**修 订 记 录**

**目 录**

[一、 概述 3](#_Toc519790529)

[1.1. 定义 4](#_Toc519790530)

[1.2. 意义 5](#_Toc519790531)

[1.3. 前提 5](#_Toc519790532)

[二、 框架 6](#_Toc519790533)

[2.1. 目标 6](#_Toc519790535)

[2.2. 框架 7](#_Toc519790536)

[2.2.1. HttpClient 7](#_Toc519790537)

[2.2.2. REST Assured 7](#_Toc519790538)

[2.2.3. TestNG 8](#_Toc519790539)

[2.2.4. Maven 8](#_Toc519790540)

[2.2.5. Jenkins 9](#_Toc519790541)

[三、 实现 9](#_Toc519790542)

[3.1. 设计思想 9](#_Toc519790544)

[3.2. 代码结构 9](#_Toc519790545)

[3.3. 框架特性 10](#_Toc519790546)

[3.4. 规则 10](#_Toc519790547)

[3.4.1. 用例规则 10](#_Toc519790548)

[3.4.2. 代码规则 10](#_Toc519790549)

[3.5. 代码实现 11](#_Toc519790550)

[3.5.1. 测试框架 11](#_Toc519790551)

[3.5.2. 测试用例结构 12](#_Toc519790552)

[四、 持续集成 12](#_Toc519790553)

[4.1. 设计思想 12](#_Toc519790555)

[4.2. Job执行架构 12](#_Toc519790556)

[4.3. 执行结果 12](#_Toc519790557)

[五、 测试效果 13](#_Toc519790558)

[六、 近期目标 14](#_Toc519790559)

[七、 远期目标 14](#_Toc519790560)

# 概述

有别于传统桌面应用开发，Web应用因为其迭代周期短，需求变化快的特性，越来越多的Web开发团队在拥抱敏捷。同时随着前端技术Ajax、JavaScript、Html5等不断被推出和完善，Web应用也变得越来越复杂和绚丽。这都给我们的测试带来了挑战。

* 持续变化的需求
* 复杂的UI（各种JavaScript插件，Ajax等）
* 往往一个测试团队需要应付多个项目的测试工作
* 非常有限的资源（人力，成本，等等…）
* 跨平台（Linux，Mac，windows）
* 跨浏览器（Firefox，Chrome，IE）

如何在有限的资源下，有效地解决目前我们测试团队所面临的挑战呢？最好最有效的方式即是通过工具及自动化手段实现。

**UI**

**API**

**Unit**

在分层测试的“金字塔”模型中， UI层（主要是WEB或APP）自动化测试因业务变更、升级等原因导致页面变化，脚本维护繁巨，成本较高；单元测试技术要求较高，成本较大；中间API自动化测试收益更大，且容易实现，维护成本低，有着更高的投入产出比。

**UI**

**API**

**Unit**

## 定义

API testing is a type of software testing that involves testing application programming interfaces (APIs) directly and as part of integration testing to determine if they meet expectations for functionality, reliability, performance, and security. Since APIs lack a GUI, API testing is performed at the message layer. API testing is now considered critical for automating testing because APIs now serve as the primary interface to application logic and because GUI tests are difficult to maintain with the short release cycles and frequent changes commonly used with Agile software development and DevOps).

API 测试是一种作为集成测试的一部分，通过直接控制被测应用的接口（API）来确定是否在功能、可靠性、性能和安全方面达到预期的[软件测试](http://lib.csdn.net/base/softwaretest" \o "软件测试知识库" \t "_blank)活动。由于 API 都没有 GUI 界面，API 测试都是在通讯层进行的。现在 API 测试在自动化测试中有着很重要的地位，因为 API 一般是应用逻辑的主要接口，同时 GUI 测试在[敏捷](http://lib.csdn.net/base/agile" \o "敏捷知识库" \t "_blank)开发和 DevOps 的快速迭代和频繁变更中很难维护。

API测试的重点是检查数据的交换、传递和控制管理过程，以及系统间的逻辑依赖关系等。

API测试适用场景：

* 多系统间交互
* 高复杂性平台（拥有多个子系统交互）
* 提供服务的底层框架

## 意义

* 节省成本
* 站在用户的角度
* 效益更高

## 前提

* 接口文档：应包括

1. 功能说明
2. 接口URL
3. 接口类型
4. [参数说明](http://phalapi.oschina.mopaas.com/Public/demo/checkApiParams.php?service=User.GetBaseInfo)
5. 返回参数
6. 成功样例

* 业务流：系统业务逻辑交互关系
* 数据流：业务I/O数据流
* 协议：HTTP、WebService、Dubbo、Thrift、Socket等网络协议
* 数据库

# 框架

Maven

Java/Python

TestNG

Excel（API Test File）

SQL DB

Test Result Report

HttpClient/REST Assured

Jenkins

1. 概要设计方案
2. 详细代码说明
3. Jenkins部署

## 目标

* 测试框架可移植
* 脚本可配置
* 数据驱动
* 持续集成

## 框架

### HttpClient

HttpClient 是Apache Jakarta Common 下的子项目，可以用来提供高效的、最新的、功能丰富的支持 HTTP 协议的客户端编程工具包，并且它支持 HTTP 协议最新的版本和建议。基本特性：

* 基于标准、纯净的java语言，实现了Http1.0和Http1.1
* 实现了所有 HTTP 的方法（GET,POST,PUT,HEAD 等）
* 支持自动转向
* 支持 HTTPS 协议
* 支持代理[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "_blank)等
* 通过Http代理建立透明的连接。
* 利用CONNECT方法通过Http代理建立隧道的https连接。

### REST Assured

REST Assured是一个可以简化HTTP Builder顶层 基于REST服务的测试过程的Java **DSL**（针对某一领域，具有受限表达性的一种计算机程序设计语言）。它支持发起POST,GET,PUT,DELETE,OPTIONS,PATCH和HEAD请求，并且可以用来验证和校对这些请求的响应信息。

### TestNG

TestNG是一个测试框架，其灵感来自JUnit和NUnit，但引入了一些新的功能，使其功能更强大，使用更方便。其特点为：

* 注解性
* 参数化
* 数据驱动
* 灵活的运行时配置
* 支持依赖测试
* 灵活的插件API
* 支持多线程测试

### Maven

Maven是一个项目管理工具。[Maven](http://www.yiibai.com/maven" \t "_blank)提供了开发人员构建一个完整的生命周期框架。开发团队可以自动完成项目的基础工具建设，Maven使用标准的目录结构和默认构建生命周期。

在多个开发团队环境时，Maven可以设置按标准在非常短的时间里完成配置工作。由于大部分项目的设置都很简单，并且可重复使用，Maven让开发人员的工作更轻松，同时创建报表，检查、构建和测试自动化设置。

概括地说，Maven简化和标准化项目建设过程，增加可重用性并负责建立相关的任务。

### Jenkins

持续集成(CI)是现代软件开发领域的基石，它改变了团队对于整个开发过程的理解。CI架构能够使得从开发到部署顺序进行，更快地发现和修复bug，最终给客户带来更多的价值。Jenkins即为这样一款，使用java语言编写的持续集成工具，其支持各种语言开发实现的项目，比如.net、Java、ruby、PHP等。其特点：

* 开源
* 易于使用，界面简单、直观
* 扩展性
* 分布式

# 实现



## 设计思想

使用Java/Python作为开发语言， Excel作为用例管理工具，TestNG 作为测试框架，Maven作为项目管理工具，Jenkins作为持续集成手段，来实现整个项目的开发测试。

## 逻辑结构

测试用例  
（Excel）

测试报告（html）

测试数据

API 测试框架（Driver）

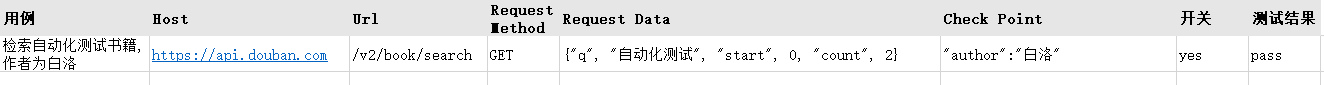
## 框架特性

* 脚本与框架分离
* 脚本框架组织、管理方便
* 脚本可重用、可配置

## 规则

### 用例规则

* API测试用例以模块划分，写入不同的excel
* 每个excel表中应包含如下列（仅作参考，根据项目实际情况可能会变更）



* 用例执行结果写入excel

### 代码规则

* 尽量少使用全局变量
* 自定义变量使用有意义的字符串
* 类名以大写字母开头
* 方法以小写字母开头
* 方法前添加注释及author
* 关键逻辑添加注释信息

## 代码实现

### 测试框架

* 测试框架类（Framework）
  + HttpClient/Restful Assure API API
  + 常用方法集Utils，包括string处理、xml处理、char处理、excel处理、csv处理、文件处理、数据库处理、zip包处理等
  + TestNG类（TestNGCaseBaseAPI），包括测试用例初始化，测试用例执行前准备beforeSuite、beforeTest、beforeClass、beforeGroup、beforeMethod；测试用例执行后的操作afterMethod、afterGroup、afterClass、afterTest、afterSuite等
* 公共方法类（MicroseerAPICommon）
  + 项目基本方法类
  + 模块特殊方法类
* 测试用例断言类（assert）
* 测试用例异常处理类
* 日志及报告配置类

### 测试用例结构

* Script

（继承于TestNGCaseBaseAPI）中添加一条测试用例。包括@Test标签、组信息、限时，case断言、失败截图及异常

* Xml

testng.Xml、pom.xml，在class中引用script

# 持续集成



## 设计思想

定时执行、多节点分布式执行、模块化、代码自动部署

## Job执行架构

Jenkins（Master）

Client2

Client1

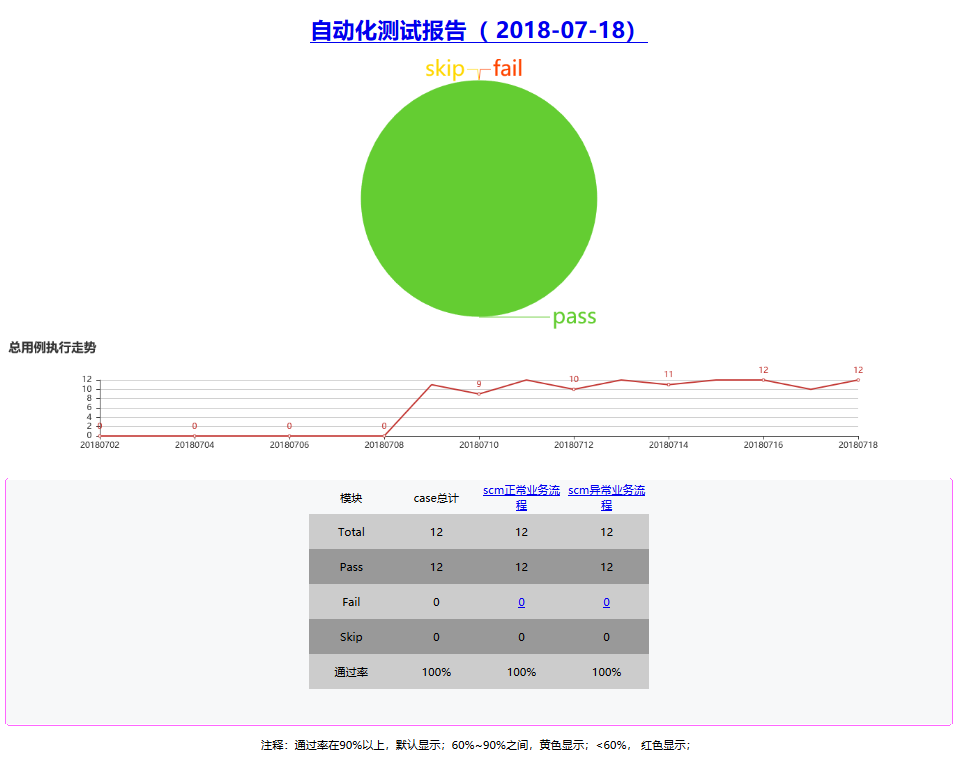
（client可添加）……

Job随机

Job随机

## 执行结果

* + - 概括化，即实现Jenkins结果的总结、统计
    - 可视化，即通过输出报告，查看测试失败、通过、跳出情况
    - 模块化，即通过输出报告，查看各模块测试失败、通过、跳出情况



# 测试效果

* 通过Jenkins，实现了用例持续执行
* 验证一些逻辑较复杂、繁琐的case，提高了测试效率
* 通过对相应模块xml文件的操作，即模块下某一组测试开关的打开关闭，实现回归测试
* 通过调整Jenkins配置，实现测试某单一模块或几个模块的集合

# 近期目标

* API测试框架搭建

# 远期目标

* 敏捷开发过程中API测试常规化
* API自动化测试框架满足更多需求