

TestNG Introductions

Testing framework

- Postman
- Python unittest
- Junit
- TestNG

Why TestNG

- TestNG is a testing framework designed to simplify a broad range of testing needs, from unit testing to integration testing
- **Next Generation**

Three-step process

1. Write the business logic of your test and insert **TestNG annotations** in your code.
2. Add the information about your test (e.g. the class name, the groups you wish to run, etc...) in a **testng.xml** file or in build.xml.
3. **Run TestNG.**

Annotations

@BeforeSuite	The annotated method will be run before all tests in this suite have run.
@AfterSuite	The annotated method will be run after all tests in this suite have run.
@BeforeTest	The annotated method will be run before any test method belonging to the classes inside the <test> tag is run.
@AfterTest	The annotated method will be run after all the test methods belonging to the classes inside the <test> tag have run.
@BeforeGroups	The list of groups that this configuration method will run before. This method is guaranteed to run shortly before the first test method that belongs to any of these groups is invoked.

@AfterGroups	The list of groups that this configuration method will run after. This method is guaranteed to run shortly after the last test method that belongs to any of these groups is invoked.
@BeforeClass	The annotated method will be run before the first test method in the current class is invoked.
@AfterClass	The annotated method will be run after all the test methods in the current class have been run.
@BeforeMethod	The annotated method will be run before each test method .
@AfterMethod	The annotated method will be run after each test method .

<i>alwaysRun</i>	For before methods (beforeSuite, beforeTest, beforeTestClass and beforeTestMethod, but not beforeGroups): If set to true, this configuration method will be run regardless of what groups it belongs to. For after methods (afterSuite, afterClass, ...): If set to true, this configuration method will be run even if one or more methods invoked previously failed or was skipped .
<i>dependsOnGroups</i>	The list of groups this method depends on.
<i>dependsOnMethods</i>	The list of methods this method depends on.
<i>enabled</i>	Whether methods on this class/method are enabled.
<i>groups</i>	The list of groups this class/method belongs to.
<i>inheritGroups</i>	If true, this method will belong to groups specified in the @Test annotation at the class level.

@DataProvider	Marks a method as supplying data for a test method. The annotated method must return an Object[][] where each Object[] can be assigned the parameter list of the test method. The @Test method that wants to receive data from this DataProvider needs to use a dataProvider name equals to the name of this annotation.
<i>name</i>	The name of this data provider. If it's not supplied, the name of this data provider will automatically be set to the name of the method.
<i>parallel</i>	If set to true, tests generated using this data provider are run in parallel. Default value is false.
@ Factory	Marks a method as a factory that returns objects that will be used by TestNG as Test classes. The method must return Object[].
@Listeners	Defines listeners on a test class.
<i>value</i>	An array of classes that extend org.testng.ITestNGListener .

@ Parameters	Describes how to pass parameters to a @Test method.
<i>value</i>	The list of variables used to fill the parameters of this method.
@Test	Marks a class or a method as part of the test.
<i>alwaysRun</i>	If set to true, this test method will always be run even if it depends on a method that failed.
<i>dataProvider</i>	The name of the data provider for this test method.
<i>dataProviderClass</i>	The class where to look for the data provider. If not specified, the data provider will be looked on the class of the current test method or one of its base classes. If this attribute is specified, the data provider method needs to be static on the specified class.
<i>dependsOnGroups</i>	The list of groups this method depends on.
<i>dependsOnMethods</i>	The list of methods this method depends on.
<i>description</i>	The description for this method.
<i>enabled</i>	Whether methods on this class/method are enabled.

<i>expectedExceptions</i>	The list of exceptions that a test method is expected to throw. If no
<i>groups</i>	The list of groups this class/method belongs to.
<i>invocationCount</i>	The number of times this method should be invoked.
<i>invocationTimeOut</i>	The maximum number of milliseconds this test should take for the cumulated time of all the invocationcounts. This attribute will be ignored if invocationCount is not specified.
<i>priority</i>	The priority for this test method. Lower priorities will be scheduled first.
<i>successPercentage</i>	The percentage of success expected from this method
<i>singleThreaded</i>	If set to true, all the methods on this test class are guaranteed to run in the same thread, even if the tests are currently being run with parallel="methods". This attribute can only be used at the class level and it will be ignored if used at the method level. Note: this attribute used to be called sequential (now deprecated).
<i>timeOut</i>	The maximum number of milliseconds this test should take.
<i>threadPoolSize</i>	he size of the thread pool for this method. The method will be invoked from multiple threads as specified by invocationCount. Note: this attribute is ignored if invocationCount is not specified

TestNG testing

- `Exception Test(@Test(expectedExceptions = ?.class))`
- `Ignore Test(@Test(enabled = false))`
- `Timeout Test(@Test(timeOut = ?))`
- `Suit Test(@BeforeSuite, @AfterSuite...)`
- `Group Test(@Test(groups = ?))`
- `Parameterized Test(@Parameters,@dataProvider...)`
- `Dependency Test(@Test(dependsOnMethods = { ?,?}))`
- `Load Test(@Test(invocationCount = ?,threadPoolSize = ?))`

testng.xml

- Tree

suite

--suite-files

--parameter

--test

----parameter

----groups

-----define

-----run

-----dependencies

----classes

-----parameter

-----class

-----parameter

-----methods

-----parameter

----packages

----listeners

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
3. <suite name="suite" junit="false" verbose="3" parallel="false" thread-count="5" configfailurepolicy="fail" annotations="javadoc" time-out="10000" skipfailedinvocationcounts="true" data-provider-thread-count="5" object-factory="classname" allow-return-values="true"> <!-- name参数为必须 -->
4.   <suite-files>
5.     <suite-file path="/path/to/suitefile1"/></suite-file> <!-- path参数为必须 -->
6.     <suite-file path="/path/to/suitefile2"/></suite-file>
7.   </suite-files>
8.   <parameter name="par1" value="value1"/></parameter> <!-- name, value参数为必须 -->
9.   <parameter name="par2" value="value2"/></parameter>
10.  <method-selectors>
11.    <method-selector>
12.      <selector-class name="classname" priority="1"/></selector-class> <!-- name参数为必须 -->
13.      <script language="java"/></script> <!-- language参数为必须 -->
14.    </method-selector>
15.  </method-selectors>
16.  <test name="testname" junit="false" verbose="3" parallel="false" thread-count="5" annotations="javadoc" time-out="10000" enabled="true" skipfailedinvocationcounts="true" preserve-order="true" allow-return-values="true"> <!-- name参数为必须 -->
17.    <parameter name="par1" value="value1"/></parameter> <!-- name, value参数为必须 -->
18.    <parameter name="par2" value="value2"/></parameter>
19.    <groups>
20.      <define name="xxx"> <!-- name参数为必须 -->
21.        <include name="" description="" invocation-numbers="" /> <!-- name参数为必须 -->
22.        <include name="" description="" invocation-numbers="" />
23.      </define>
24.    <run>
25.      <include name="" /> <!-- name参数为必须 -->
26.      <exclude name="" /> <!-- name参数为必须 -->
27.    </run>
28.    <dependencies>
29.      <group name="" depends-on=""></group> <!-- name, depends-on均为参数为必须 -->
```

reportNG

- reportng-1.1.4.jar

getAuditPlanListTest						Groups
	Duration	Passed	Skipped	Failed	Pass Rate	
w_LoginTest	0.048s	1	0	0	100%	
w_getAuditPlanListTest	0.350s	4	0	2	66%	
Total		5	0	2	71%	

getCheckAuditPlanNameTest						Groups
	Duration	Passed	Skipped	Failed	Pass Rate	
w_LoginTest	0.067s	1	0	0	100%	
w_getCheckAuditPlanNameTest	0.078s	2	0	0	100%	
Total		3	0	0	100%	

postAuditPlanTest						Groups
	Duration	Passed	Skipped	Failed	Pass Rate	
w_LoginTest	0.001s	0	1	0	0%	
w_postAuditPlanTest	0.001s	0	1	0	0%	
w_getAuditPlanListTest	0.001s	0	2	0	0%	
Total		0	4	0	0%	

HttpClient

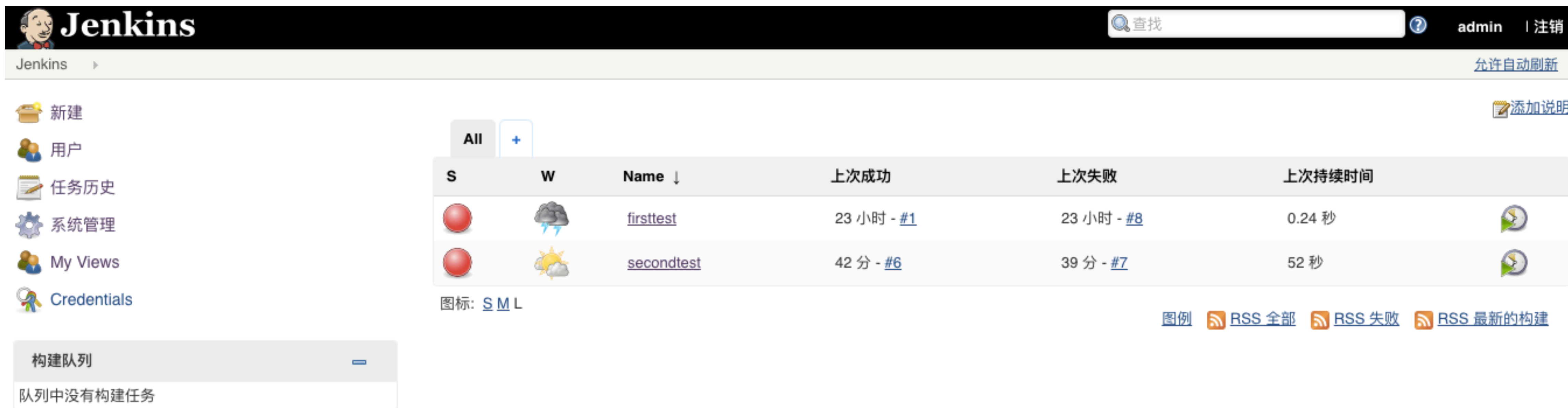
- URLConnection
- HttpURLConnection
- HttpClient

Selenium

- Python
- Java

Now

- Jenkins+maven+git+TestNG(+selenium)
- git clone <http://git.iparmacare.net/daijunjun/iparmacare-sf-webtest.git>



The screenshot shows the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and user information (admin | 注销). The left sidebar contains links for 新建 (New), 用户 (Users), 任务历史 (Task History), 系统管理 (System Management), My Views, and Credentials. The main content area displays a table of build jobs. The table has columns for status (S), weather icon (W), Name, last success (上次成功), last failure (上次失败), and last duration (上次持续时间). Two jobs are listed: 'firsttest' and 'secondtest'. Below the table, there are links for '图例' (Legend), 'RSS 全部' (RSS All), 'RSS 失败' (RSS Failure), and 'RSS 最新的构建' (RSS Latest Build). At the bottom left, there is a '构建队列' (Build Queue) section showing '队列中没有构建任务' (No build tasks in the queue).

S	W	Name ↓	上次成功	上次失败	上次持续时间
		firsttest	23 小时 - #1	23 小时 - #8	0.24 秒
		secondtest	42 分 - #6	39 分 - #7	52 秒

图标: [S](#) [M](#) [L](#)

[图例](#) [RSS 全部](#) [RSS 失败](#) [RSS 最新的构建](#)

Thank you