第5章

SoapUI脚本开发实战精要

本章将讲解 SoapUI 如何在项目实战中应用。由于 SoapUI 本身的学习资料少之又少，所以本章将尽可能详细地讲解，但可能不会讲述太多基础的知识，所以在学习之前最好有一定的基础，否则在基本的概念和操作上可能会有点“晕”。大家可以去作者的博客或者附录中的参考资料里进行提前学习。这里使用的是 SoapUI 5.0 Pro 版本。

5.1 SoapUI 介绍

SoapUI 是一款强大的接口测试工具，易用性极好，很多操作可以通过界面来完成，这也是它受到很多人喜欢的原因之一，另外它自带 Mock 服务，通过界面的操作就可以完成，大大降低了入门门槛。SoapUI 的版本分为开源版和 Pro 版，其中 Pro 版就是商业版，在功能上会比开源版强大很多。

SoapUI 可以轻松完成 SOAP 和 REST 的 WebService 测试并可自动生成测试报告，除此之外它还可以做接口级的压力测试和安全测试。尽管它如此强大，但最拿手的还是进行 SOAP WebService 接口的功能自动化测试，而本章内容也将主要围绕此点进行。更多 SoapUI 的介绍可以看官网 https://www.soapui.org/。

5.2　SOAP WebService 接口功能自动化测试

先来了解下什么是 SOAP 协议。本书尽可能地把教科书式的解释弱化，而是用更加通俗易懂的语言来解释，这样大家理解起来会更容易。你可以简单地理解为 SOAP 协议是基于 XML 的一个简易的协议，如果用一句话概况那就是：SOAP＝HTTP＋XML，协议中必须包括 Envelope、Body 等元素。

此处我们以 qqCheckOnline 的 WebService 接口为例进行讲解，接口的具体信息如下。

- 接口描述：获得腾讯 QQ 在线状态。
- 入参：qqCode，String 类型。默认 QQ 号码：8698053。
- 出参：qqCheckOnlineResult，String 类型。
  返回数据代表的含义为：Y ＝在线；N ＝离线；E ＝ QQ 号码错误；A ＝商业用户验证失败；V ＝免费用户超过数量。
- 返回格式：

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <qqCheckOnlineResponse xmlns="http://WebXml.com.cn/">
      <qqCheckOnlineResult>string</qqCheckOnlineResult>
    </qqCheckOnlineResponse>
  </soap:Body>
</soap:Envelope>
```

了解了接口信息之后我们来看看如何完成接口用例脚本的设计，大致步骤如图 5.1 所示。

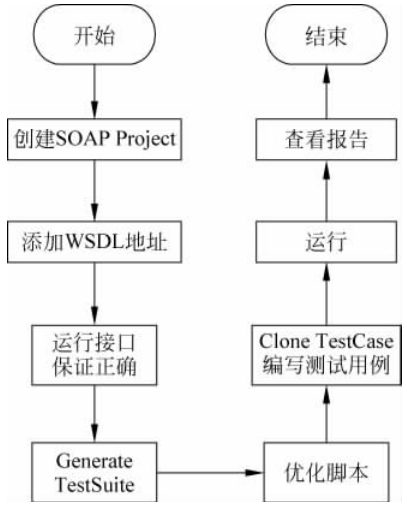

图 5.1　接口用例脚本设计步骤

5.2.1　单接口的测试方法

按照图 5.1 所示的步骤完成初步设置后，脚本结构如图 5.2 所示，这是最简单的脚本状态，还有很多地方需要优化改进，下面我们就分别讲解常见的优化方法。注意：后续的操作都在 TestSuite 中完成。

图 5.2　脚本结构

我们在设计测试用例的时候根据接口的信息，可能需要考虑多种情况，包括但不限于正确的 QQ 号码、错误的 QQ 号码、处于在线状态的 QQ 号码和处于离线状态的 QQ 号码等来验证各种情况下的接口的正确性，具体的用例需要根据具体的接口信息来设计。此处我们只以正确且处于在线状态的 QQ 号码为例进行讲解。

1. 参数化

打开 TestSteps 下的 qqCheckOnline 接口，如图 5.3 所示，会发现其中的 qqCode 是写死的，显然这个不是我们希望的，我们希望这里是“活”的。

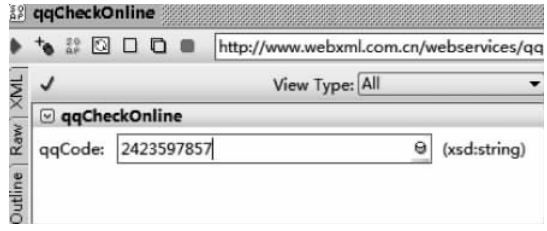

图 5.3　qqCode

那如何能使该参数变“活”呢？这时候就要利用 DataSource 这个强大的功能了。在 DataSource 中可以通过多种外部介质来实现参数化，比如：

- File：文本文件的形式。
- Excel：最好使用 2003 格式的 Excel。
- Grid：表格形式。
- JDBC：JDBC 数据源，就是从数据库中获取。
- XML：XML 格式。
- Groovy：Groovy 脚本形式。

这里我们使用 File 类型的文本文件形式进行参数化，大致实现步骤如下。

1）在本地电脑上新建一个文本文件 qq.txt，并在文件中输入如图 5.4 所示的内容。

图 5.4　qq.txt

2）新建一个 DataSource，填入相关的数据信息，注意它的顺序要位于接口之前，如图 5.5 所示。

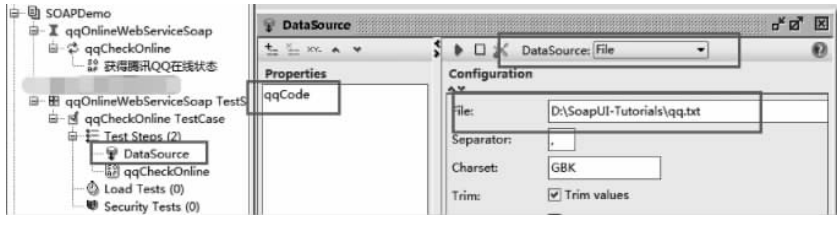

图 5.5　DataSource

部分字段的解释如下。

- DataSource：选择外部的存储介质。
- File：选择文件的路径。
- Properties：把从外部存储介质中获取的结果保存到这里。
- 其余的字段可以保持默认。

3）切换到 qqCheckOnline 接口，把之前写死的 qqCode 变“活”。只需在 qqCode 参数处右击选择 Get Data 下对应步骤中的 Property 即可，如图 5.6 所示。

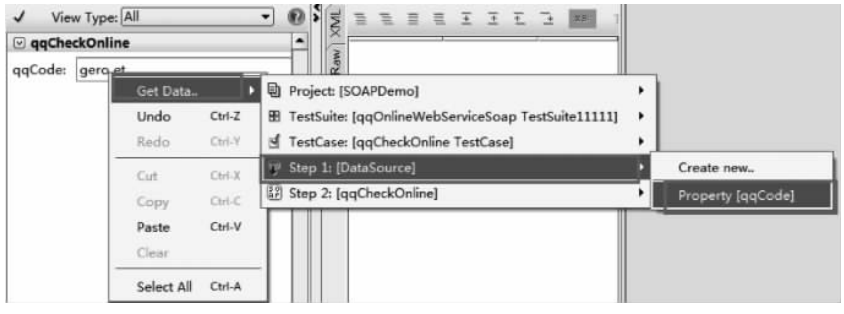

图 5.6　Get Data

4）增加 DataSource Loop，完成参数化的遍历，如果不添加这个则永远取出来的是第一个 QQ 号码，最终的脚本结构如图 5.7 所示。其中 DataSource Step 是选择的源数据，Target Step 是选择的目标步骤。这里需要特别注意 DataSource、接口、DataSource Loop 的顺序。

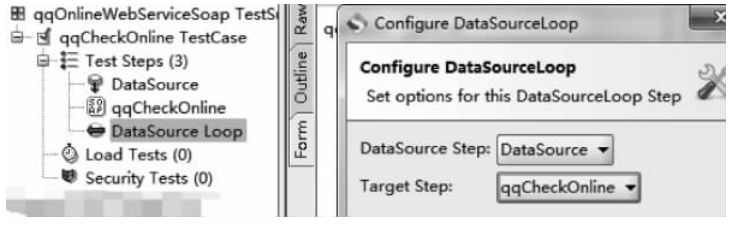

图 5.7　DataSource Loop

2. 断言（检查点）

既然我们是做接口的功能自动化，那一定会对返回的响应数据（出参）进行检查，只有符合我们预期结果才能认为该接口通过测试，要完成这件事

情就需要用到断言，即常说的检查点，大致实现步骤如下。

1）双击 TestSteps 中的接口并运行，在响应区域对你想检查的内容添加断言，右键选择 Add Assertion→for Content，如图 5.8 所示。

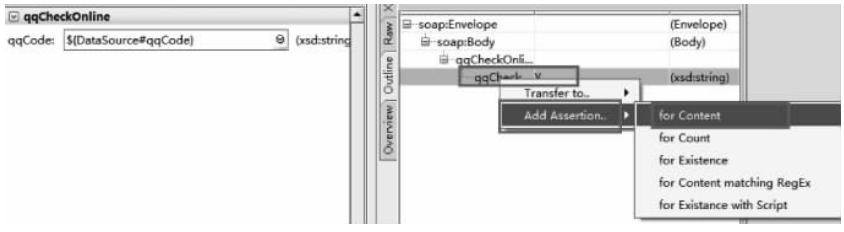

图 5.8　选择断言

2）在弹出的 XPath Expression 对话框中可以看到已经识别出来了要检查的内容就是 qqCheckOnlineResult 对应的值 Y，直接单击 save 按钮即可，如图 5.9 所示。

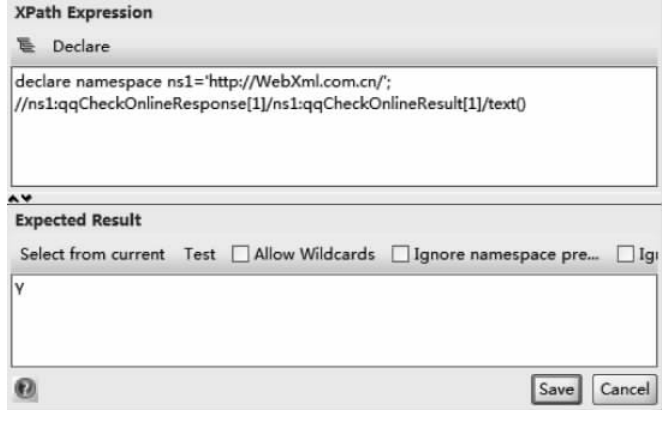

图 5.9　确认断言

3）最终完成后的效果如图 5.10 所示，其中 Assertions 表示的就是断言。

在 SoapUI 中有多种形式的断言，可谓功能十分强大，可以通过单击 Add Assertion 按钮来查看，如图 5.11 所示。

1）Property Content 类型的部分断言解释如下。

- Contains：包含。在本节内容 2.断言中已经讲解过。
- Not Contains：不包含。比如，在返回的正确结果中不应该包含什

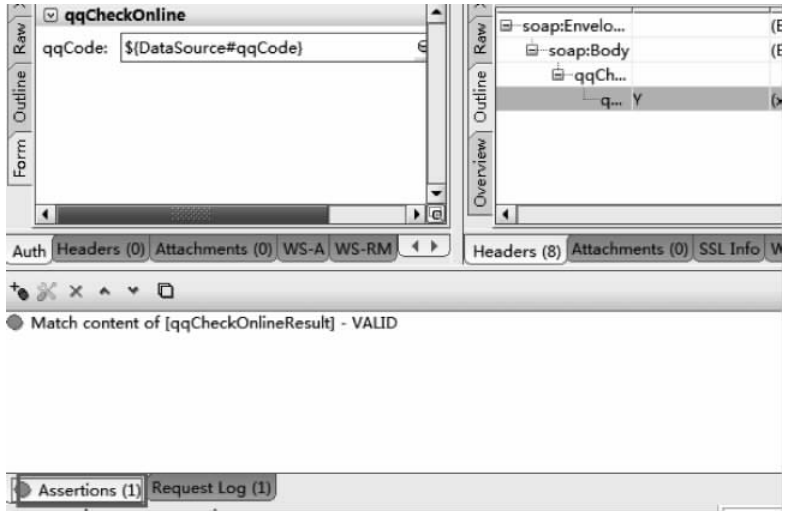

图 5.10 断言效果图

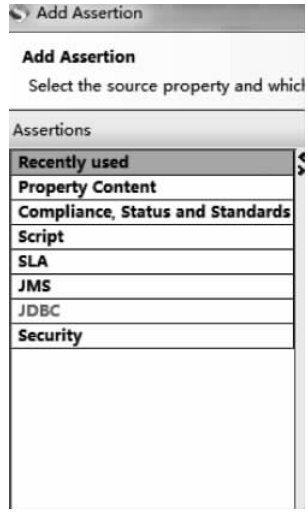

图 5.11 断言类型

么，就可以用该断言。

- XPath Match：可以利用 XPath 表达式进行断言，如果断言的内容是变化的，可以选中 Allow Wildcards 利用通配符来匹配，如图 5.12 所示。

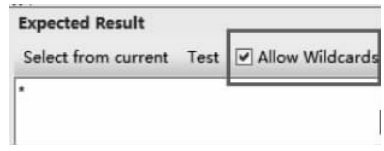

图 5.12　Allow Wildcards

2）SLA 类型的部分断言解释如下。

Response SLA：设置该断言后，如果超过设定时间仍没有收到返回信息，就表示请求失败了。默认为 200ms，如果超过这个数字就判定为失败。

3）其他的断言用法类似，因为都是界面操作，所以大家只要耐心看下每个断言下方的英文解释就能明白是干什么的了。SoapUI 的断言里还有一个群组断言的概念，意思就是可以设置多个单个断言，然后把这些断言组成一个群组，该群组里的断言都通过了才算成功，或者该群组里的部分断言通过了就算成功，这些都可以自由设定。

3. 运行与报告

完成上述步骤之后，就可以运行本用例脚本了，双击本 TestCase，在弹出的 qqCheckOnline TestCase 对话框中单击“绿色小箭头”即可，如图 5.13 所示。如果想看 SoapUI 生成的测试报告，单击“文档”形状的图标即可，测

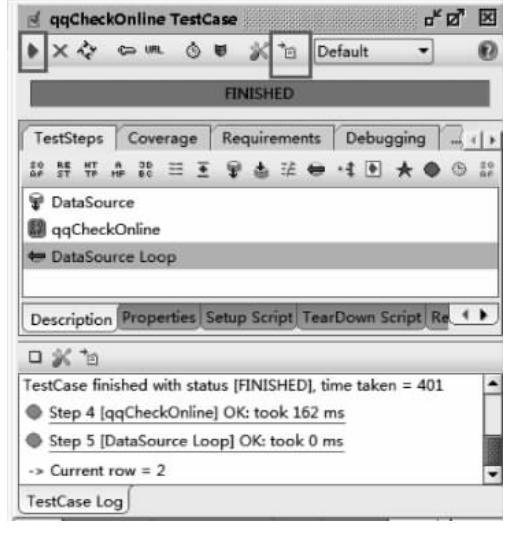

图 5.13　运行 TestCase

试报告样式如图 5.14 所示。

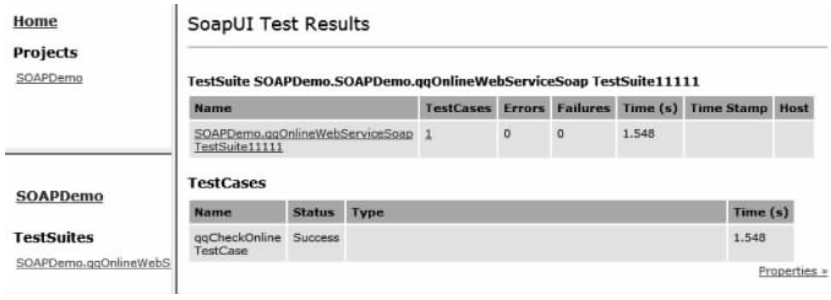

图 5.14 测试报告

所有类似这样单接口的测试大概都是这个过程，大家需要根据具体的接口信息做一定的调整，但整体的思路和方法是大同小异的，也希望大家在不断学习的过程中可以悟到“一通百通”的道理，这样即使你只有 2 年的工作经验也可能会超越有 5 年工作经验的朋友。

5.2.2 接口依赖的测试方法

在实际的应用中我们经常会遇到这样一种情况，现在有两个接口，分别是接口 1 和接口 2，其中接口 2 的入参要用到接口 1 中响应数据中的某个字段(出参)，这时候就产生了接口之间的依赖。在 SoapUI 中可以通过非常方便的操作解决这个问题，大大降低了实现难度。

下面我们就来讲解下常见的两种解决方案。

1. 在 TestSteps 中进行接口之间的数据传递

这种情况处理起来非常简单，为了方便说明，我们再增加一个 qqCheckOnline 的 step 并命名为 qqCheckOnline 2，然后把 qqCheckOnline 的响应数据中的 qqCheckOnlineResult 字段作为入参传递给 qqCheckOnline 2，大致实现步骤如下。

1) 在 qqCheckOnline 2 的入参处进行如图 5.15 所示的操作，即把 qqCheckOnline 响应中的字段传入此处。

2) 在弹出的 Select XPath 对话框中选中你需要的响应数据，这里需要的就是检查的结果 Y，单击 ok 按钮，如图 5.16 所示。

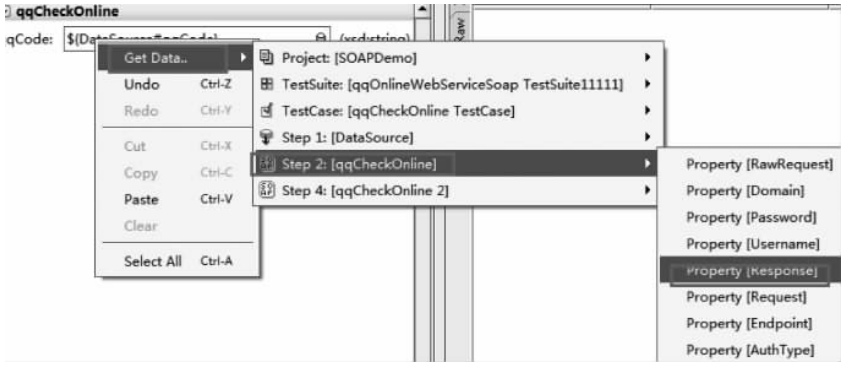

图 5.15　选择响应数据

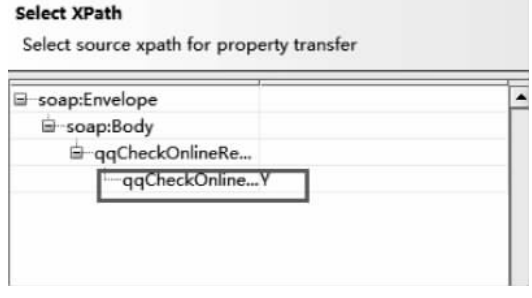

图 5.16　Select XPath

这样就完成了在 TestStep 之间的接口数据的传递了，也就解决了接口的依赖。如果运行，会提示你 Fail，原因就是我们从 qqCheckOnline 中获取的响应是 Y，传入 qqCheckOnline 2 中后不符合 QQ 号码的要求，自然就返回错误了。

2. 在 TestCase 中进行接口之间的数据传递

要解决这个问题，比在 TestStep 中稍微复杂一点，核心的思想就是：利用 TestSuite 中的 Properties 是可以用共享的这个特性来完成。更加通俗点说就是找了一个可以全局共享的中间人，让它来帮助我们做数据的传递，大致实现步骤如下。

1）为了方便讲解，增加一个 TestCase 并命名为 qqCheckOnline TestCase 2，如图 5.17 所示。

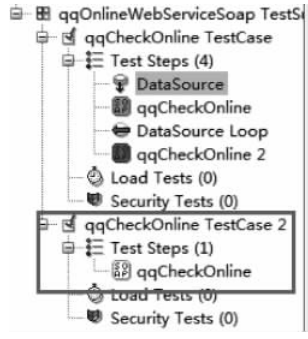

图 5.17 qqCheckOnline TestCase 2

2）新建一个 TestSuite 级别的 Properties，如图 5.18 所示，Value 留空即可。

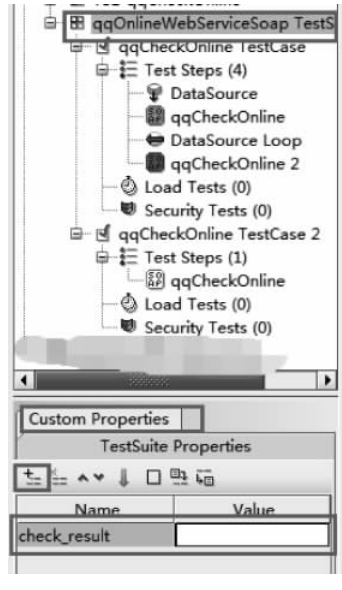

图 5.18 Custom Properties

3）在第一个用例脚本中新建一个 Property Transfer 用来传递数据，如图 5.19 所示。其中 Source 代表要从哪里获取数据，Target 代表要把获取的数据存到哪里。根据之前的思路，我们就是要从第一个用例脚本中的接口获取数据，然后存到 TestSuite 中的 Properties 里。

图 5.19　Property Transfer

4）在第二个用例脚本中的接口入参处替换即可，如图 5.20 所示。

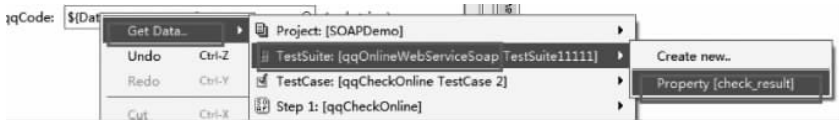

图 5.20　替换参数

以上讲解，基本已经涵盖了在实际应用中常见的情况，也可以应对大部分的接口测试，不过具体的实现还需要根据接口的信息做灵活的调整，大家在学习的时候不要过分死板要学会变通，这样学习才能有效率。同时，细心的朋友会发现，我们每次在解决一个问题的时候并不是急于去操作，而是先把主要的思路整理好，然后再去实践，不然就会像没头的苍蝇到处乱碰，这个也是很多小白和经验缺乏的朋友都需要注意的地方。

5.3　SOAP WebService 接口负载测试

SoapUI 可以完成简单的接口负载测试，但这个并不是它的强项，能够提供的数据也非常有限，不过用起来还是很方便的，大致实现步骤如下。

1）在 Load Tests 处单击右键选择 New TestLoad 即可。

2）创建完成后如图 5.21 所示。部分字段解释如下。

- Limit 表示负载要持续的时间。

- Threads 表示并发数。
- Test Delay 表示从完成一次用例后，到开始下一次前，休息多长时间，单位是毫秒。
- Random 的设置代表的是 Test Delay 的浮动范围，如果设置为 0.5 则代表 Test Delay 在“Test Delay ＊（1－0.5）～ Test Delay ＊（1＋0.5）”毫秒之间，如果设置为 0 则表示不会进行浮动。

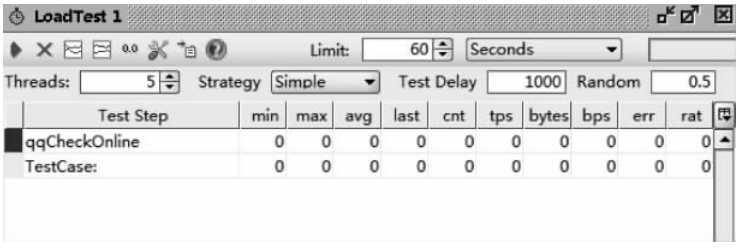

图 5.21 LoadTest

3）运行中你也可以单击“折线图”按钮切换到图形模式，结束之后单击“文档”按钮即可生成测试报告。

SoapUI 在负载测试中也可以设置断言，单击 LoadTest 弹出框下方切换到 LoadTest Assertions 标签，然后单击“添加”按钮会弹出断言的设置，如图 5.22 所示。

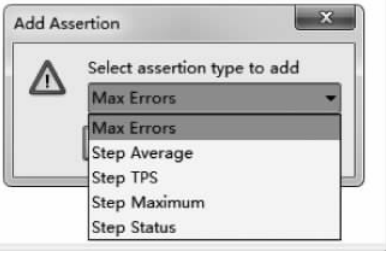

图 5.22 断言

负载测试中的断言解释如下。

- Max Errors：当 Error 的数量超过设定的值时就结束测试，不论测试时间是否结束。
- Step Average：可以用于对任何一个请求的平均响应时间做断言。
- Step TPS ：可以用于对任何一个请求的每秒处理的事务数做断言。

- Step Maximum：如果超过设置的最长时间就报错。
- Step Status：状态码断言。

虽然在负载测试方面确实比不上专业的工具，但是功能测试完成之后顺便看看性能如何也是蛮方便的。

5.4　SOAP WebService 接口安全测试

本来安全测试的知识不在本书的范围内，但是考虑到介绍 SoapUI 的完整性，还是要写一下。对于安全方面的相关知识本节不会讲述，大家可以到 SoapUI 官网查看帮助文档，地址：https://www.soapui.org/security-testing/security-scans.html。

SoapUI 的安全测试是通过对被测接口进行内置的安全策略遍历攻击来进行的。也就是说，SoapUI 内置了一些安全策略和测试数据，然后它会按照设定的策略把测试数据注入接口中进行测试，最后给出测试报告。

利用 SoapUI 进行接口安全测试的大致实现步骤如下。

1）在任意一个工程下的 SecurityTest 处右键创建 SecurityTest1，之后选择测试策略，如图 5.23 所示。

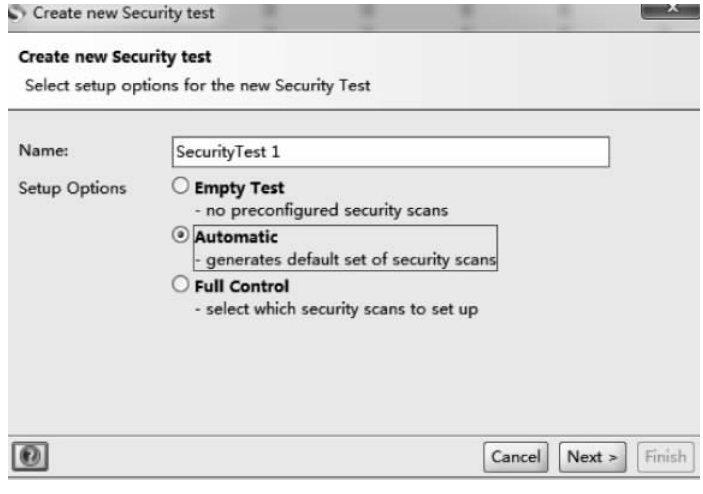

图 5.23　Create new Security test

Setup Options 中的字段解释如下。

- Empty Test：创建一个空白的测试策略，需要手动去配置安全扫描策略，如果你对安全测试非常熟悉可以使用此方法，否则建议不要选择。
- Automatic：自动使用内置的一些默认的安全扫描策略。此选项为推荐。
- Full Control：进行更加全面、细致的安全扫描。

2）之后根据提示一直单击 Next 按钮，直到最后单击 Finish 按钮，如图 5.24 所示。

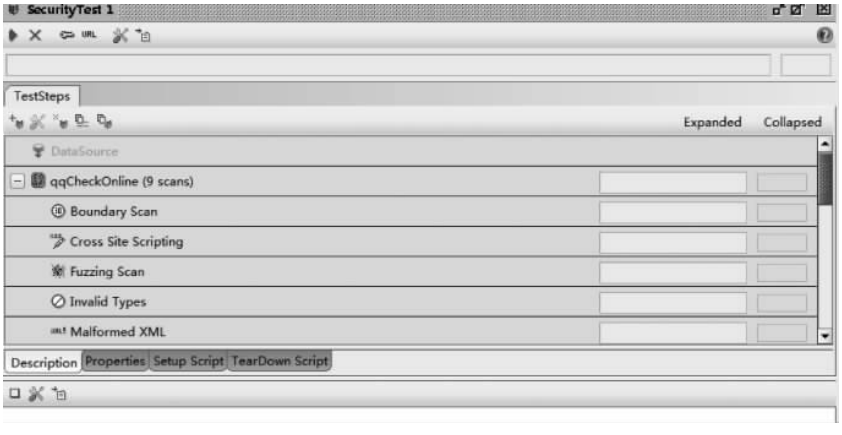

图 5.24　SecurityTest

3）单击导航上的绿色小箭头即可进行测试，测试结束后单击文档图标可生成报告。扫描结果没问题的显示绿色，有问题的显示非绿色。这里需要注意的是，即使 SoapUI 给出提示某个安全扫描项有问题，也要进行排查。

最后普及一下典型的安全测试知识，具体介绍如下。

1. SQL 注入

SQL 注入的通俗解释就是通过把特殊的或者恶意的 SQL 代码注入表单进行提交，提交后后台程序运行了该段代码，最终把隐私信息暴露了出来。比如，前几年各大网站的信息泄露。一般预防的方法有如下几种：

- 对用户的输入一定要进行校验，尤其是对单引号和双引号要进行转换；

- 为每个应用使用单独的数据库连接权限；
- 坚决不要使用动态方式来拼接 SQL 语句；
- 所有隐私的信息尽可能不要暴露，包括提示也是。有的网站访问出了问题并没有对错误页做统一处理，而是把错误以及服务器信息完全暴露了。

2. CSS 跨站式脚本攻击（又名 XSS）

跨站式脚本攻击是指攻击者在页面中插入了恶意的 HTML 代码，当用户触发时，这段恶意代码就会被执行。比如，常见的 Cookie 盗取。图 5.25 为早期新浪微博存在的 XSS 漏洞。

图 5.25　XSS

一般的预防方法也是要对前端和后端做双端验证、过滤特殊字符、对 HTML 的属性进行过滤等。

5.5　SoapUI 轻量级接口自动化测试框架

也不知道什么时候横空出世了一个“轻量级”的概念，我理解为“重量级”的反面，也就是说相对比较轻便。那么本节就和大家分享下如何利用

SoapUI 来构建一个轻量级接口测试框架。说到这里，也许有朋友会觉得很高大上，其实这都是概念包装而已，只要理解了本质你就不会这么想了。

任何测试框架都有一个基本的思想，那就是脚本和数据的分离，好处是业务测试人员可以专心地设计测试用例，并且方便管理数据。这里我们就用 SoapUI 和最常用的 Excel 来完成轻量级的接口测试框架。实现过程并不高大上，但可以给迷茫的朋友提供一种思路，我觉得这就是价值所在。

实现本框架的大致思路为：利用 SoapUI 完成接口的请求处理等，Excel 完成入参和结果数据的记录。这里的 Excel 建议使用 2003 版，其他版本的可能会有问题。此处我们继续以获得腾讯 QQ 在线状态的 WebService 接口为例，大致实现步骤如下。

1）完成最基本的接口调试，并保证可以正常执行。

2）在外部建立 Excel，需要的字段为 qqCode（QQ 号码）、expected_result（期望结果）、actual_result（实际结果）、is_pass（是否通过），当然这里的字段大家可以根据实际情况自行扩展，我这里用的都是基本的字段并未进行扩展。

3）确定 Excel 中的字段后再把需要的测试数据写入对应的字段中即可，其中 actual_result（实际结果）、is_pass（是否通过）留空，它们会在脚本执行完成后自动填写。

4）创建 DataSource，用来读取 Excel 中的数据，如图 5.26 所示。其中 File 是文件路径；Worksheet 是工作簿；Start at Cell 是要开始的单元格；左侧的两个 Properties 用来接收数据。

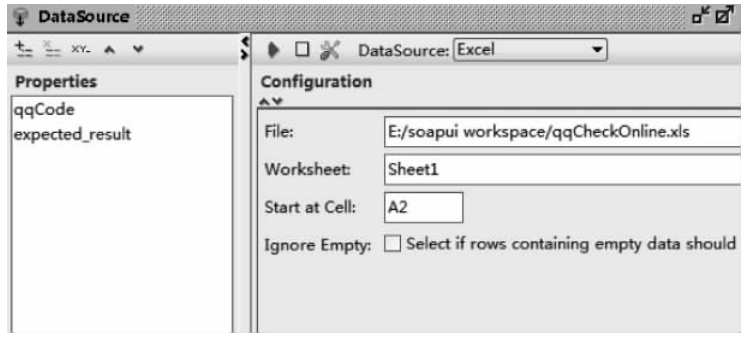

图 5.26 DataSource

5）请求中的入参 qqCode 替换为 DataSource 中的，这样就可以从 Excel 中读取数据了。

6）创建 PropertyTransfer，获取响应中的结果用于后续判断是否成功，如图 5.27 所示。其中左侧的 Transfers 就是用来接收响应中指定的数据的，也就是 qqCheckOnlineResult 的值。

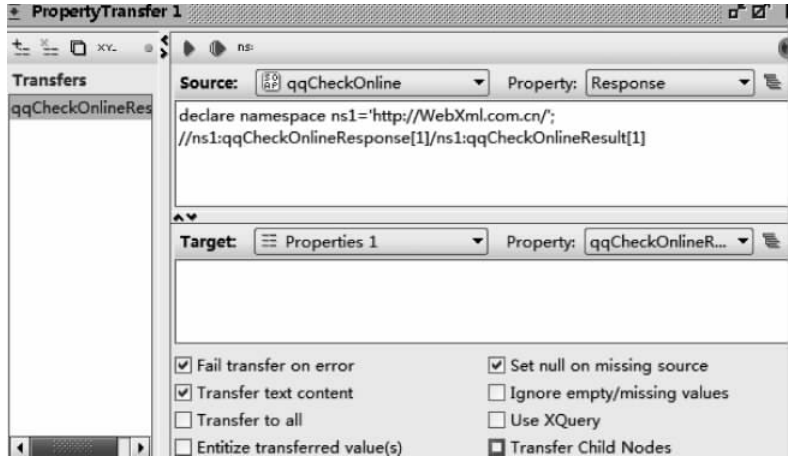

图 5.27　PropertyTransfer

7）创建 Groovy Script，在这里需要自己编写脚本，主要是完成从 Excel 里读取预期结果然后和实际结果做对比，并把对比结果返回，代码如下，里面有部分注释。

```
//从 DataSource 中获取 expected_result 的值
def expected_result = context.expand('${DataSource#expected_result}')
//从响应结果中获取 qqCheckOnlineResult 的值
def response = context.expand('${Properties 1#qqCheckOnlineResult}')
//把预期结果和实际响应做对比，成功返回 pass，失败返回 fail
if(expected_result == response)
{
    return "pass"
}
else
{
    return "fail"
}
```

小 强 课 堂

此处使用了 SoapUI 中的 Groovy 脚本编程，它和 Java 类似，但又有些不一样。比如，Groovy 脚本中不需要显式声明变量类型，它可以自动识别。关于更多 Groovy 脚本的介绍可以到官网查看，地址：http://groovy-lang.org/learn.html。

在 SoapUI 中尝试用的方法有 getPropertyValue、setPropertyValue、context.expand、getXmlHolder、getNodeValue 等。

8）创建 DataSink，把对比结果写到 Excel 里，如图 5.28 所示。其中左侧的 Value 就是代码中的获取方法。

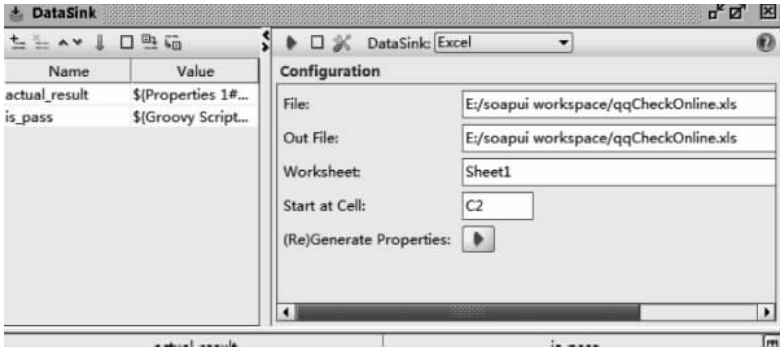

图 5.28　DataSink

小 强 课 堂

如果说 DataSource 是从外部介质读取数据，那么 DataSink 就是把内部的数据存储到外部介质中。此处就是把获取的实际结果和是否通过两个数值写入到外部介质 Excel 中。

9）创建 DataSource Loop，完成数据的循环操作。同样，需要注意它们的顺序。

10）执行 testcase 并查看 Excel 结果，如图 5.29 所示。

到此我们就完成了一个轻量级接口测试框架的构建，一个基础的框架

	A	B	C	D
1	qqCode	expected_result	actual_result	is_pass
2	2423597857	Y	Y	pass
3	2083503238	Y	Y	pass

图 5.29　运行结果

就此诞生了。是不是比你想象的要简单呢？其实正如在第一章中和大家分享的一样，自动化测试重要的是有思路，有了思路之后万事都可以想办法实现。当然，这个轻量级框架只是一个基本的雏形，还有很多地方可以改进和完善，大家感兴趣的可以自己研究，也欢迎与我交流分享。

5.6　本章小结

本章对如何使用 SoapUI 工具进行接口级的各类测试做了讲解，并对大家经常遇到的接口依赖的问题做了解答，最后的轻量级接口测试框架也可以很好地应用到实际工作中，虽然篇幅不算很多，但内容是比较实用的。有时候大家认为只有内容多、字数多才有价值，但是本来一句能解释清楚的为什么非要用一段内容来解释呢？我不知道有多少人看过《软件测试的艺术》这本书，它非常薄，可以说比现在任何测试类的书都要薄很多，但却是非常有价值的，也是我一直推荐小白朋友们学习测试的必看书籍。所以，也希望大家能正确地看待“量”这个概念。