

14 | 更接近业务的抽象：让自动化测试脚本更好地描述业务

2018-07-30 茹炳晟

软件测试52讲

[进入课程 >](#)



讲述：茹炳晟

时长 10:05 大小 4.62M



在上一篇文章中，我介绍了 GUI 自动化测试中的两个主要的概念“脚本与数据的解耦”以及“页面对象模型”。在引入“操作函数”封装时，我提到操作函数在改善测试脚本可读性问题的同时，也引入了两个新的问题，即：如何把控操作函数的粒度，以及如何衔接两个操作函数之间的页面。

现在，我就以这两个问题作为引子，为你介绍 GUI 自动化测试中“业务流程（business flow）”的概念、核心思想以及应用场景。

如何把控操作函数的粒度？

操作函数的粒度是指，一个操作函数到底应该包含多少操作步骤才是最合适的。

如果粒度太大，就会降低操作函数的可重用性。极端的例子就是，前面文章中涉及的百度搜索的案例，把“登录”“搜索”“登出”的操作作为一个操作函数。

如果粒度太小，也就失去了操作函数封装的意义。极端的例子就是，把每一个步骤都作为一个操作函数。

更糟糕的是，在企业实际自动化测试开发中，每个测试工程师对操作函数的粒度理解也不完全相同，很有可能出现同一个项目中脚本粒度差异过大，以及某些操作函数的可重用性低的问题。

那么，操作函数的粒度到底应该如何控制呢？其实这个问题，在很大程度上取决于项目的实际情况，以及测试用例步骤的设计，并没有一个放之四海而皆准的绝对标准。

但是，脚本粒度的控制还是有设计依据可以遵循的，即往往以完成一个业务流程（business flow）为主线，抽象出其中的“高内聚低耦合”的操作步骤集合，操作函数就由这些操作步骤集合构成。

比如，对于“用户注册”这个业务流程，其中的“信用卡绑定”操作就会涉及多个操作步骤，而这些操作在逻辑上又是相对独立的，所以就可以包装成一个操作函数。也就是说，业务流程会依次调用各个操作函数，来完成具体的业务操作。

如何衔接两个操作函数之间的页面？

完成一个业务流程操作，往往会需要依次调用多个操作函数，但是操作函数和操作函数之间会有页面衔接的问题，即前序操作函数完成后的最后一个页面，必须是后续操作函数的第一个页面。

如果连续的两个操作函数之间无法用页面衔接，那就需要在两个操作函数之间加入额外的页面跳转代码，或者是在操作函数内部加入特定的页面跳转代码。

业务流程抽象

在解决如何把控操作函数的粒度，以及如何衔接两个操作函数之间的页面这两个问题的过程中，我引入了业务流程的概念。那么，接下来我就跟你详细说说什么是业务流程。

业务流程抽象是，基于操作函数的更接近于实际业务的更高层次的抽象方式。基于业务流程抽象实现的测试用例往往灵活性会非常好，你可以很方便地组装出各种测试用例。

这个概念有点拗口，难以理解。但是，没关系，我举个例子，你就豁然开朗了。

假设，某个具体的业务流程是：已注册的用户登录电商平台购买指定的书籍。那么，基于业务流程抽象的测试用例伪代码，如图 1 所示。

```
1 // Business Flow - login flow
2 LoginFlowParameters loginFlowParameters = new LoginFlowParameters();
3 loginFlowParameters.setUserName("username");
4 loginFlowParameters.setPassword("password");
5 LoginFlow loginFlow = new LoginFlow(loginFlowParameters);
6 loginFlow.execute();
7
8 // Business Flow - Search book flow
9 SearchBookFlowParameters searchBookFlowParameters = new SearchBookFlowParameters();
10 searchBookFlowParameters.setBookName("bookname");
11 SearchBookFlow searchBookFlow = new SearchBookFlow(searchBookFlowParameters);
12 searchBookFlow.withStartPage(loginFlow.getEndPage()).execute();
13
14 // Business Flow - Checkout/buy book flow
15 CheckoutBookFlowParameters checkoutBookFlowParameters = new CheckoutBookFlowParameters();
16 checkoutBookFlowParameters.setBookID(searchBookFlow.getOutput().getBookID());
17 CheckoutBookFlow checkoutBookFlow = new CheckoutBookFlow(checkoutBookFlowParameters);
18 checkoutBookFlow.withStartPage(searchBookFlow.getEndPage()).execute();
19
20 // Business Flow - logout flow
21 LogoutFlow logoutFlow = new LogoutFlow();
22 logoutFlow.withStartPage(checkoutBookFlow.getEndPage()).execute();
```

图 1 基于业务流程抽象的测试用例伪代码

这段伪代码的信息量很大，但是理解了这段代码的设计思想，你也就掌握了业务流程抽象的精髓。

首先，从整体结构上看，段伪代码顺序调用了 4 个业务流程，依次是完成用户登录的 LoginFlow、完成书籍查询的 SearchBookFlow、完成书籍购买的 CheckoutBookFlow、完成用户登出的 LogoutFlow。

这 4 个业务流程都是作为独立的类封装的，可以被很方便的重用并灵活组合，类的内部实现通常是调用操作函数。而操作函数内部，则是基于页面对象模型完成具体的页面控件操作。

然后，对于每一个业务流程类，都会有相应的业务流程输入参数类与之一一对应。具体的步骤通常有这么几步：

1. 初始化一个业务流程输入参数类的实例；
2. 给这个实例赋值；

3. 用这个输入参数实例来初始化业务流程类的实例;
4. 执行这个业务流程实例。

执行业务流程实例的过程，其实就是调用操作函数来完成具体的页面对象操作的过程。

为了让你更好地理解业务流程抽象提供了哪些功能，接下来我会为你逐行解读这段伪代码。

伪代码的第 2-6 行，调用的是 LoginFlow，完成了用户登录的操作。

 复制代码

```
1 2: LoginFlowParameters loginFlowParameters = new LoginFlowParameters();
2 3: loginFlowParameters.setUserName("username");
3 4: loginFlowParameters.setPassword("password");
4 5: LoginFlow loginFlow = new LoginFlow(loginFlowParameters);
5 6: loginFlow.execute();
```


第 2 行，初始化了 LoginFlow 对应的 LoginFlowParameters 的实例。

第 3-4 行，通过 setUsername 和 setPassword 方法将用户名和密码传入该参数实例。

第 5 行，用这个已经赋值的参数实例来初始化 LoginFlow。

第 6 行，通过 execute 方法发起执行。执行之后，LoginFlow 会调用内部的操作函数，或者直接调用页面对象方法，完成用户登录的操作。

伪代码的第 9-12 行，用和 2-6 行类似的方式调用了 SearchBookFlow，完成了书籍搜索的操作。

 复制代码

```
1 9: SearchBookFlowParameters searchBookFlowParameters = new SearchBookFlowParameters();
2 10: searchBookFlowParameters.setBookName("bookname");
3 11: SearchBookFlow searchBookFlow = new SearchBookFlow(searchBookFlowParameters);
4 12: searchBookFlow.withStartPage(loginFlow.getEndPage()).execute();
```


需要特别注意的是，第 12 行中 `withStartPage(loginFlow.getEndPage())` 的含义是，`SearchBookFlow` 的起始页面将会使用之前 `loginFlow` 的结束页面。显然，通过这种方式可以很方便地完成两个业务流程之间的页面衔接。

同时，从中还可以看出，其实每个业务流程都可以接受不同的起始页面。以 `SearchBookFlow` 为例，它的起始页面既可以是书籍首页，也可以是其他页面，但是需要在它的内部对不同的初始页面做出相应的处理，以保证这个业务流程真正开始的页面是在书籍搜索页面。

同样，由于业务流程存在分支的可能性，每个业务流程执行完成的最终页面也不是唯一的，你可以使用 `getEndPage` 方法拿到这个业务流程执行结束后的最后页面。

通过这段代码的解读，你可以很清楚地理解，业务流程之间的页面衔接是如何实现的。

伪代码的第 15-18 行，调用了 `CheckoutBookFlow`，完成了书籍购买操作。

 复制代码

```
1 15: CheckoutBookFlowParameters checkoutBookFlowParameters = new CheckoutBookFlowParamet
2 16: checkoutBookFlowParameters.setBookID(searchBookFlow.getOutPut().getBookID());
3 17: CheckoutBookFlow checkoutBookFlow = new CheckoutBookFlow(checkoutBookFlowParameters
4 18: checkoutBookFlow.withStartPage(searchBookFlow.getEndPage()).execute();
```


第 15 行，初始化了 `CheckoutBookFlow` 对应的 `checkoutBookFlowParameters` 的实例。

第 16 行，通过 `setBookID(searchBookFlow.getOutPut().getBookID())`，将上一个业务流程 `searchBookFlow` 的输出参数，作为了当前业务流程的输入参数。这是典型的业务流程之间如何传递参数的示例，也是很多测试场景中都要用到的。

第 17 行，用 `checkoutBookFlowParameters` 参数实例来初始化 `checkoutBookFlow`。

第 18 行，通过 `execute` 方法发起执行。这里需要注意的是，`checkoutBookFlow` 的起始页面将会使用之前 `searchBookFlow` 的结束页面。开始执行后，`checkoutBookFlow` 会调用内部的操作函数，或者直接调用页面对象方法，完成书籍的购买操作。

伪代码的第 21-22 行，调用 LogoutFlow，完成了用户登出操作。

 复制代码

```
1 21: LogoutFlow logoutFlow = new LogoutFlow();  
2 22: logoutFlow.withStartPage(checkoutBookFlow.getEndPage()).execute();
```

第 21 行，由于 LogoutFlow 不带参数，所以直接初始化了 LogoutFlow。

第 22 行，通过 execute 方法发起执行。这里 LogoutFlow 的起始页面将会使用之前 CheckoutBookFlow 的结束页面。开始执行后，LogoutFlow 会调用内部的操作函数，或者直接调用页面对象方法，完成用户登出操作。

通过对这些代码的解读，我解释了业务流程是什么，并从使用者的角度分析了它的主要特点。比如，如何实现不同业务流程间的页面衔接，如何在不同的业务流程间传递参数等。

为了加深印象，我再来总结一下业务流程的优点：

1. 业务流程（Business Flow）的封装更接近实际业务；
2. 基于业务流程的测试用例非常标准化，遵循“参数准备”、“实例化 Flow”和“执行 Flow”这三个大步骤，非常适用于测试代码的自动生成；
3. 由于更接近实际业务，所以可以很方便地和 BDD 结合。BDD 就是 Behavior Driven Development，即行为驱动开发，我会在后续文章中详细讲解。

总结

我以如何把控操作函数的粒度，和如何衔接两个操作函数之间的页面，这两个问题为引子，为你介绍了业务流程的概念、核心思想和适用的场景。

业务流程抽象是，基于操作函数的更接近于实际业务的更高层次的抽象方式。基于业务流程抽象实现的测试用例往往具有较好的灵活性，可以根据实际测试需求方便地组装出各种测试用例。

业务流程的核心思想是，从业务的维度来指导测试业务流程的封装。由于业务流程封装通常很贴近实际业务，所以特别适用于组装面向终端用户的端到端（E2E）的系统功能测试用

例，尤其适用于业务功能非常多，并且存在各种组合的 E2E 测试场景。

思考题

你所在公司的 GUI 自动化测试是否已经运用了业务流程级别的封装？在使用过程中，你是否遇到什么瓶颈，是如何解决的？

欢迎你给我留言。

 极客时间

软件测试52讲

从小工到专家的实战心法



茹炳晟 eBay中国研发中心
测试基础架构技术主管

新版升级：点击「 请朋友读」，10位好友免费读，邀请订阅更有**现金**奖励。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 13 | 效率为王：脚本与数据的解耦 + Page Object模型

下一篇 15 | 过不了的坎：聊聊GUI自动化过程中的测试数据

精选留言 (25)

 写留言



图·美克尔
2018-08-01

代码自动生成会讲到吗？

 2

展开 ▾

作者回复: 代码细节不会讲, 只会讲基本的思路。

◀ ▶



II

2018-07-31

👍 2

和老师的思路不谋而合, 实际项目也用了基于业务流程封装的业务关键字, 写测试用例过程非常灵活好用, 功能变更只需要修改业务关键字内部结构, 不改用例; 数据的实例化对象也超级好用, 灵活改变数据, 数据内容丰富多样。期待老师的测试数据准备的课程.....



李真真

2018-07-30

👍 2

在我自己学自动化测试的过程中, 主要有三大疑问: 1. 元素的有效定位方法
2. 断言如何做到全面性
3. POM怎么颗粒化

这些都没有固定的标准和方法, 搞的有点无措。读了本篇, 第三个问题就有了很清楚的一种参考方法啦。非常感谢老师!

展开 ▾



Cynthia◆...

2018-07-30

👍 2

我曾经做过的一个自动化项目, 整体代码实现方式与思路和这里介绍的不太一样。对业务流程也做了封装, 但每一个封装都是以子用例的形式来做的, 例如:
从登录login到a业务到b业务再到c业务算是一条完整的用例
而另一条用例是login a业务 x业务
那么就封装一个子用例1为: login-a业务。...

展开 ▾



口水窝

2019-03-28

👍 1

继续打卡, 还没做到深入GUI自动化封装的阶段, 加油, 先学习体系, 在操作实践!



sy lan215

👍 1



2019-02-13

1.上一篇我提到的 selenium 实现的 4 层实现，恰好和老师说的「业务流程」的层次性和逐层抽象的理念相吻合，看来从实际业务场景触发得出的结论是大同小异的；

2.如果按「业务流程」的概念来解释我之前说的把登陆和退出操作实现在了第二层的函数层就很好理解了，因为登陆和退出操作不是完全固定的具有公共属性的原子操作，所以...
展开 ▾



subona

2018-10-16

👍 1

老师有类似的实现源码供参考吗？伪代码看不懂具体该怎么实现

展开 ▾



Allen

2018-07-31

👍 1

公司的业务流程比较复杂，需要在接口层覆盖业务流程的自动化测试。最近正在设计接口自动化的测试方案，看了这篇文章，很有启发。

作者回复: 能有收获就好，其实我后面还会介绍更好的办法，就是通过gui来捕捉后端的api调用列表，后面的文章会有具体例子，希望可以帮到你



图·美克尔

2018-07-30

👍 1

然后，对于每一个业务流程类，都会有相应的业务流程输入参数类与之一一对应。具体的步骤通常有这么几步：

初始化一个业务流程输入参数类的实例；

...

展开 ▾

作者回复: 非常高质量的留言，你说的方法非常好，而且我们也曾经实际尝试了，和你说的完全一样的思路，但是最终我们放弃了，主要原因是技术上的实现难度有点大，我们需要知道哪些flow是可以衔接的，并且还要做到ide中可以自动提示，同时flow之间的测试数据传递写出来也会比较难看，还有就是两个flow之间在实际用例中经常需要插入很多额外的操作，而且由于我们后面基于BDD做了代码自动生成，所以我们没有采用全链的方式。





Geek_7d396...

2019-05-14



按照13章给的抽象模式：“XXXPage.YYYComponent.ZZZOpera...”，这里的YYYComponent 是指啥呢？

展开 ▾



后乐

2019-05-09



总结的很到位，很喜欢这种授人以渔的课程！学到了不少，得好好实践下~

展开 ▾

作者回复: 感谢支持，我一直反对直接教工具的使用，只有理解了背后的原理，才能做的更好



johnny

2018-11-19



第13节的内容能理解，我已经将伪代码实现了。但是这节的内容不好理解，老师可以给我发一个完整的示例吗（不是用伪代码描述的，是真正用java语言实现的代码示例）？简单的业务流程，只要能说明第14节内容就行。我的邮箱是cijnjk@163.com

作者回复: 实际代码可能没法发，因为不来源，实在不好意思



小老鼠

2018-10-24



如果某个用户场景别的测试用例肯定用不上，是否也需要封装，主要目的是便于阅读？



晶晶

2018-09-27



其实我觉得测试人员还是应该掌握面相对象设计思想才能更好的写出自动化测试工具，基础不打好，只能知其然而不能知其所以然。



江鸟川

2018-09-17



做人用RF做界面自动化吗？感觉还是只会个皮毛

展开 ▾



Tall Gira...

2018-09-14



老师，我不太清楚LoginFlowParameters这个业务流程参数类的作用是什么，可以解释一下吗。这处有点看不懂



Kuzaman

2018-08-14



精彩的文章，理清了思路，准备通用到接口测试中，把前一个响应的参数对全部抽取出来，做类似Java 里的get方法，方便下一个接口使用。期待与老师9月2号的见面。

展开 ▾



Sunshine

2018-08-03



感谢老师讲解，现在脑子有了一个更清晰的思路

展开 ▾

作者回复: 能够有收获就是最好的



涅槃Ls

2018-08-01



打卡14

展开 ▾



雪哥

2018-07-31



新手问下，有什么好的论坛，心得交流平台吗，或者测试经常浏览的技术网站

展开 ∨

作者回复: 这类网站是有很多, 不会如果你想学某一种工具, 我强烈推荐你上官网

