10 页面对象（Page Object）模式

本章大纲

10.1 页面对象模式简介

10.2 单一PageObject实例

10.3 多个PageObject实例

10.4 PageObject注意事项

页面对象模式简介

- 使用面向对象的设计模式，页面对象模式将测试代码和被测试页面的页面元素及其操作方法进行分离，以此降低页面元素变化对测试代码的影响。
- 每一个测试页面都会被单独定义一个类，类中会定位所有需要参与测试的页面元素对象，并且定义操作每一个页面元素对象的方法。

PageObject 设计模式的优点

- 减少代码的重复
- 提高测试用例的可读性
- 提高测试用例的可维护性，特别是针对UI频繁变化的项目

本章大纲

10.1 页面对象模式简介

10.2 单一PageObject实例

10.3 多个PageObject实例

10.4 PageObject注意事项

PagetObject实例1

- 使用PageFactory类给测试类提供带操作的页面元素
- 以百度搜索为例：

  声明一个名为SearchPage的类，并且通过定位表达式找到搜索框和搜索按钮“百度一下”，分别定义输入的方法和单击的方法

  创建两个类：页面对象类，测试类

使用注解获取页面元素

```
@FindBy(id= "A")
private WebElement A;

@Findbys({
@FindBy(className = "A"),
@FindBy(id = "B") }
)
public WebElement AB;

@FindAll({
@FindBy(id = "A"),
@FindBy(id = "B")
 })
 public List<WebElement> AandB;
```

PageFactory类封装页面的元素

```
public class SearchPage {

    @FindBy(id="kw")
    public WebElement inputSearch;

    @FindBy(id="su")
    public WebElement btnSearch;

    public SearchPage(WebDriver driver){
        PageFactory.initElements(driver, this);
    }
}
```

```
public class SearchPageTest extends  BaseTest {
    @Test
    public void search(){
        SearchPage searchPage =new SearchPage(wd);
        wd.get("https://www.baidu.com/");
        searchPage.inputSearch.sendKeys("taobao");
        searchPage.btnSearch.click();
        assertTrue(wd.getPageSource().contains("购物"));
    }
}
```

PageObject实例2

- PageFactory类封装页面的元素的操作方法

```
public class SearchPage2 {
    public WebDriver _driver;
    String url = "https://www.baidu.com/";
    @FindBy(id="kw")
    public WebElement inputSearch;
    @FindBy(id="su")
    public WebElement btnSearch;

    public SearchPage2(WebDriver driver){
        this._driver = driver;
        PageFactory.initElements(_driver, this);
    }

    public void load(){
        this._driver.get(url);
    }

    public void search(String keyword){
        this.inputSearch.sendKeys(keyword);
        this.btnSearch.click();
    }
}
```

PageObject实例2

```
public class SearchPageTest2 extends  BaseTest {
    @Test
    public void search() throws InterruptedException{
        SearchPage2 searchPage =new SearchPage2(wd);
        searchPage.load();
        searchPage.search("taobao");
        Thread.sleep(3000);
        assertTrue(wd.getPageSource().contains("购物"));
    }
}
```

PageObject实例3

- 页面类继承LoadableComponent类可以在页面加载的时候判断是否加载了正确的页面，只需要重写load() 和isLoaded()两个方法。此方式有助于让页面对象的页面访问操作更加健壮。

PageObject实例3

```
public class SearchPage3 extends LoadableComponent<SearchPage3> {
    public WebDriver _driver;
    String url = "https://www.baidu.com/";
    @FindBy(id="kw")
    public WebElement inputSearch;
    @FindBy(id="su")
    public WebElement btnSearch;

    public SearchPage3(WebDriver driver){
        this._driver = driver;
        PageFactory.initElements(_driver, this);
    }

    public void load(){
        this._driver.get(url);
    }
     public void search(String keyword) {
         this.inputSearch.sendKeys(keyword);
         this.btnSearch.click();
     }

     @Override
     protected void isLoaded() throws Error {
         assertTrue(this._driver.getTitle().contains("百度一下"));
     }
}
```

PageObject实例3

```
public class SearchPageTest3 extends  BaseTest {
    @Test
    public void search() throws InterruptedException{
        SearchPage3 searchPage =new SearchPage3(wd);
        //继承LoadableComponent后，只要实现了覆盖的load方法
        //即使在没有定义get()方法的情况下，也可以调用get()
        //默认会调用页面类中的load()方法
        searchPage.get();
        searchPage.search("taobao");
        Thread.sleep(3000);
        assertTrue(wd.getPageSource().contains("购物"));
    }
}
```

本章大纲

10.1 页面对象模式简介

10.2 单一PageObject实例

10.3 多个PageObject实例

10.4 PageObject注意事项

多个Paget Object的自动化实现

- 使用正确的用户名错误的密码，判断是否出现“帐号或密码错误”
- 使用正确的用户名正确的密码，判断是否出现“退出”
- 登录后，成功添加影片信息
- 创建类四个类
  - ✓BaseTest 浏览器操作
  - ✓LoginPage 登录页面类
  - ✓ManageMovie 管理电影类
  - ✓MovieTest 测试类

本章大纲

10.1 页面对象模式简介

10.2 单一PageObject实例

10.3 多个PageObject实例

10.4 PageObject注意事项

PageObject注意事项

1. 在PageObject类中定义public方法来对外提供服务
2. 不在PageObject类中进行断言操作
3. 只需要在PageObject中定义需要的操作元素和方法
4. PageObject页面中的相同动作如果产生多个不同的结果，需要在PageObject类中定义多个操作方法

思考

怎样结合数据驱动模式和页面对象模式实现自动化测试