04 WebDriver 基础应用

本章大纲

4.1 浏览器多窗口的操作

4.2 常见控件的操作

4.3 操作Frame中的页面元素

4.4 模拟键盘的操作

句柄+网页title

使用getWindowHandle方法来获取当前浏览器窗口的句柄
使用getWindowHandles方法获取所有弹出的浏览器窗口的句柄

```
@Test
public void controlWindow() {
    wd.get("http://localhost:8032/MyMovie/");
    JavascriptExecutor js = (JavascriptExecutor) wd;
    js.executeScript("window.open('https://www.baidu.com/')");
    Set<String> windows = wd.getWindowHandles();
    for (String s : windows) {
        System.out.println(s);
        boolean status = wd.switchTo().window(s).getTitle().contains("My Movie");
        if (status) {
            wd.findElement(By.linkText("登录")).click();
        }
    }
    wd.close();
}
```

句柄+网页内容

```
public void identifyWindowByPageSource() {
    String pwindowHandle = driver.getWindowHandle();
    System.out.println(pwindowHandle);
    WebElement link1 = driver.findElement(By.tagName("a"));
    link1.click();
    Set<String> allwindowsHandles = driver.getWindowHandles();
    if (!allwindowsHandles.isEmpty()) {
        for (String windowHandle : allwindowsHandles) {
            if (driver.switchTo().window(windowHandle).getPageSource()
                    .contains("百度新闻")) {
                driver.findElement(By.tagName("a")).click();
                System.out.println("定位到第二个页面");
            }
        }
    }
    driver.switchTo().window(pwindowHandle);
    System.out.println(driver.getTitle());
}
```

本章大纲

4.1 浏览器多窗口的操作

4.2 常见控件的操作

4.3 操作Frame中的页面元素

4.4 模拟键盘的操作

输入框

```
public void inputText() throws InterruptedException {
    driver.get(baseUrl);
    WebElement input = driver.findElement(By.id("uname"));
    input.clear();// 清空
    Thread.sleep(3000);
    input.sendKeys("hello");// 输入
    input.getText();
    input.submit();
    Thread.sleep(3000);
}
```

按钮单击

```
@Test
public void buttonClick() throws InterruptedException {
    driver.get(baseUrl);
    WebElement button = driver.findElement(By.id("u_submit"));
    button.click();
}
```

下拉框的操作

```
@Test
public void operateDropList() {
    Select dropList = new Select
            (driver.findElement(By.name("fruit")));
    assertFalse(dropList.isMultiple());// 是否允许多选
    assertEquals(dropList.getFirstSelectedOption().getText(), "桃子");
    dropList.selectByIndex(2);// 0表示第一个选项
    assertEquals(dropList.getFirstSelectedOption().getText(),  "桔子");
    dropList.selectByValue("xigua");
    dropList.selectByVisibleText("猕猴桃");
}
```

多选列表的操作

```
@Test
public void operateMultipleDropList() {
    Select dropList = new Select
            (driver.findElement(By.name("course")));
    assertFalse(dropList.isMultiple());// 是否允许多选
    dropList.selectByIndex(0);
    dropList.selectByValue("tiyu");
    dropList.selectByVisibleText("程序设计基础");
    dropList.deselectAll();//取消所有选中的状态
    dropList.selectByIndex(3);
    dropList.deselectByValue("shuju");
}
```

单选框/复选框的操作

```
@Test
public void operateRadio() {
    WebElement radioSex = driver.findElement(By
            .xpath("//input[@value='1']"));
    if (!radioSex.isSelected())
        radioSex.click();
    assertTrue(radioSex.isSelected());
}
```

```
@Test
public void operateCheckBox() {
    WebElement checkBox = driver.findElement(By
            .xpath("//input[@value='11']"));
    if (!checkBox.isSelected())
        checkBox.click();
    assertTrue(checkBox.isSelected());
    List<WebElement> checkboxs = driver.findElements(By.name("hobby"));
    for (WebElement checkbox : checkboxs) {
        checkbox.click();//选中所有复选框
    }
}
```

检查文本内容是否出现

```
@Test
public void isElementTextPresent() {
    WebElement text = driver.findElement(By
            .xpath("//p[1]"));
    String contentText = text.getText();
    assertTrue(contentText.contains(contentText));
    assertEquals(contentText, "欢迎进入Selenium课程的学习");
}
```

查看页面元素的属性

```
@Test
public void  getWebElementAttribute() {
    WebElement input = driver.findElement
            (By.id("uname"));
    String inputText = input.getAttribute("value");
    assertEquals(inputText, "默认的内容");
}
```

查看页面元素的CSS属性值

```
@Test
public void  getWebElementCssValue() {
    WebElement div = driver.findElement
            (By.id("div2"));
    String divwidth = div.getCssValue("width");
    assertEquals(divwidth, "200px");
}
```

上传下载组件附件

```
@Test
public void testUpFile() {
    driver.get("file:///D:/demo/0801.html");
    driver.findElement(By.cssSelector("input[type=file]")).
    sendKeys("D:\\demo\\test1.html");
    WebDriverWait wait = new WebDriverWait(driver, 5);
    wait.until(ExpectedConditions.elementToBeClickable
            (By.id("filesubmit")));
    driver.findElement(By.id("filesubmit")).click();;
    boolean t= wait.until(ExpectedConditions.titleContains("成功"));
    assertTrue(t);
}
```

日期选择器

```
@Test
public void testDatepicker() throws InterruptedException {
    driver.get("http://jqueryui.com/datepicker/");
    driver.switchTo().frame(0);
    WebElement dateInputbox = driver.findElement(By.id("datepicker"));
    dateInputbox.sendKeys("08/02/2016");
    Thread.sleep(5000);
    System.out.println(dateInputbox.getText());
}
```

注意：有些日期选择器不允许输入，可以设置页面属性的方式，设定日期字段可以编辑的属性

处理Alert弹窗

```
public void testAlert() {
    WebElement btn = driver.findElement(By.id("su"));
    btn.click();
    // 获取Alert对象
    Alert alert = driver.switchTo().alert();
    // 使用alert.getText()获取对话框的文字
    assertEquals(alert.getText(), "这是一个alert弹出框");
    alert.accept();
}
```

处理Alert弹窗

```
public void testConfirm() {
    WebElement btn = driver.findElement(By.id("btn2"));
    btn.click();
    // 获取Alert对象
    Alert alert = driver.switchTo().alert();
    // 使用alert.getText()获取对话框的文字
    assertEquals(alert.getText(), "这是一个confirm弹出框");
    alert.dismiss();//模拟【取消】按钮的操作
    //alert.accept();模拟【确定】按钮的操作
}
```

```
@Test
public void testPrompt() {
    WebElement btn = driver.findElement(By.id("btn3"));
    btn.click();
    Alert alert = driver.switchTo().alert();
    // 使用alert.getText()获取对话框的文字
    assertEquals(alert.getText(), "这是一个prompt弹出框");
    alert.sendKeys("hello");
    alert.accept();
}
```

本章大纲

4.1 浏览器多窗口的操作

4.2 常见控件的操作

4.3 操作Frame中的页面元素

4.4 模拟键盘的操作

iframe元素定位

```
public void testFrame() {
    // 进入左侧的frame页面
    driver.switchTo().frame("leftframe");
    WebElement pElement1 = driver.findElement(By.tagName("p"));
    assertEquals(pElement1.getText(), "左侧frame页面的文字");
    // 跳出左侧的frame页面，回到frameSet页面
    driver.switchTo().defaultContent();
    // 如果frame不存在id呢，可以使用索引号，从0开始
    driver.switchTo().frame(2);
    WebElement pElement2 = driver.findElement(By.xpath("//p"));
    assertEquals(pElement2.getText(), "右侧frame页面的文字");
}
```

iframe元素定位

```
public void testFrameByPageSource() {
    // 找到页面上所有的frame对象，并储存到frames容器中
    List<WebElement> frames = driver.findElements(By.tagName("frame"));
    // 使用for循环遍历frames的所有frame页面，查找包含“左侧”frame的页面
    for (WebElement frame : frames) {
        driver.switchTo().frame(frame);
        if (driver.getPageSource().contains("左侧")) {
            WebElement pElement = driver.findElement(By.xpath("//p"));
            assertEquals(pElement.getText(), "左侧frame页面的文字");
            break;
        } else {
            // 如果没有找到指定的frame，则调用此行代码，返回到frameset页面，实现下一次循环
            driver.switchTo().defaultContent();
        }
    }
    driver.switchTo().defaultContent();
}
```

本章大纲

4.1 浏览器多窗口的操作

4.2 常见控件的操作

4.3 操作Frame中的页面元素

4.4 模拟键盘的操作

模拟键盘的操作

在webdriver的使用过程中，应该会用到使用工具来模拟用的鼠标、键盘的一些输入操作，比如说：

1、鼠标的左键点击、双击、拖拽、右键点击等；

2、键盘的回车、回退、空格、ctrl、alt、shift等；

```
@Test
public void clickKeys() {
    Actions action = new Actions(driver);
    action.keyDown(Keys.CONTROL);//按下
    action.keyDown(Keys.SHIFT);
    action.keyDown(Keys.ALT);
    action.keyUp(Keys.CONTROL);//释放
    action.keyUp(Keys.SHIFT);
    action.keyUp(Keys.ALT);
    action.keyDown(Keys.SHIFT).sendKeys("abcd").perform();
    //在浏览器的搜索框中输入大写的acd
}
```

模拟键盘的操作

```
public void testKeyOperate() throws InterruptedException{
    driver.get("http://localhost:8032/MyMovie/admin.php/Login/index.html");
    driver.findElement(By.name("username")).sendKeys("admin");
    Actions action  =new Actions(driver);
    action.sendKeys(Keys.TAB);
    action.sendKeys("admin").perform();
    action.sendKeys(Keys.ENTER).perform();
    Thread.sleep(5000);
}
```

模拟鼠标右键事件

```
@Test
public void rightClickMouse() {
    Actions action = new Actions(driver);
    action.contextClick(driver.findElement
            (By.id("uname"))).perform();
}
```

在指定元素进行鼠标悬浮

```
@Test
public void mouseoverElement() {
    WebElement link1 = driver.findElement
            (By.xpath("//a[@id='link1']"));
    Actions action = new Actions(driver);
    action.moveToElement(link1).perform();
}
```

鼠标单击左键和释放的操作

```
public void mouseclickAndRelease() throws InterruptedException {
    driver.get("file:///D:/demo/0901api.html");
    WebElement div = driver.findElement(By.xpath("//a[@id='div2']"));
    Actions action = new Actions(driver);
    action.clickAndHold(div).perform();// 在指定控件单击左键不释放
    Thread.sleep(3000);
    action.release(div).perform();// 释放鼠标左键
    // 注意，clickAndHold和release 两个方法连起来使用，会被认为执行了一次事件click
    Thread.sleep(3000);
}
```

拖拽页面元素

```
public void dragPageElement() throws InterruptedException {
    driver.get("file:///D:/demo/0901api.html");
    WebElement div = driver.findElement(By.xpath("//div"));
    Actions action = new Actions(driver);
    action.dragAndDropBy(div, 0, 50).build().perform();
    Thread.sleep(5000);
}
```