

## 07 WebDriver 常用API-2

# 本章大纲

7.1 JavascriptExecutor进阶

7.2 其他元素（上传/日期/富文本框/高亮/截屏）

7.3 操作html5元素

7.4 模拟键盘操作

7.5 自动化框架的基础

# 使用JavascriptExecutor单击元素

click无法生效时，可以使用这个方法

```
public void testjsClick() {
    WebElement btn = driver.findElement(By.id("btn1"));
    JavaScriptClick(btn);
}

public void JavaScriptClick(WebElement element) {
    try {
        if (element.isEnabled() && element.isDisplayed()) {
            System.out.println("使用Javascript单击元素");
            JavascriptExecutor js = (JavascriptExecutor) driver;
            js.executeScript("arguments[0].click();", element);
        } else {
            System.out.println("元素无法单击");
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

# 使用JavascriptExecutor操作页面的滚动条

- 滚动条到页面的最下面
- 滚动条到页面的某个元素
- 滚动条向下移动某个数量的像素

@Test

```
public void testScrolling() throws InterruptedException {  
    driver.get("http://news.baidu.com/");  
    WebElement link = driver.findElement(By  
        .LinkText("唐山迁安：苹果树“搬”进家 盆栽苹果正俏销"));  
    JavascriptExecutor js = (JavascriptExecutor) driver;  
    // 滚动条滑动至最下方  
    js.executeScript("window.scrollTo(0,document.body.scrollHeight)");  
    Thread.sleep(5000);  
    // 滚动条滑动到指定元素的位置  
    js.executeScript("arguments[0].scrollIntoView()", link);  
    // 滚动条滑动至某像素位置  
    js.executeScript("window.scrollBy(0,800)");  
}
```

# 高亮显示正在被操作的页面元素

```
public void highlightElement(WebElement element){
    JavascriptExecutor js = (JavascriptExecutor) driver;
    //使用JavascriptExecutor将传入参数的页面对象的背景颜色和边框颜色分别定为黄色和红色
    js.executeScript("arguments[0].setAttribute('style',arguments[1]);",
        element,"backgroud:yellow;border:2px solid red;");
}

@Test
public void testHighlight() throws InterruptedException{
    WebElement inputSearch = driver.findElement(By.id("kw"));
    highlightElement(inputSearch);
    Thread.sleep(3000);
}
```

# 设置页面对象的属性值

```
public void setAttribute(WebElement element, String attributeName,
    String value) {
    js.executeScript("arguments[0].setAttribute(arguments[1], " +
        "arguments[2])", element, attributeName, value);
}

@Test
public void testSetAttribute() {
    wd.get("http://localhost:8032/test/07.html");
    this.setAttribute(wd.findElement(By.id("uid")), "size", "10");
    this.setAttribute(wd.findElement(By.id("uid")), "value", "星期2");
}
```

可使用removeAttribute删除页面元素的属性

# 本章大纲

7.1 JavascriptExecutor进阶

7.2 其他元素（浮动框/上传/日期/富文本框/截屏）

7.3 操作html5元素

7.4 模拟键盘操作

7.5 自动化框架的基础

# Ajax产生的浮动框

```
@Test
public void testAjaxDivOption() {
    driver.get("https://www.sogou.com/");
    WebElement searchInputbox = driver.findElement(By.id("query"));
    searchInputbox.click();
    List<WebElement> options = driver.findElements(By
        .xpath("//*[@id='v1']/div/ul/li"));
    for (WebElement element : options) {
        if (element.getText().contains("傅园慧")) {
            System.out.println(element.getText());
            element.click();
            break;
        }
    }
}
```



# 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

# 上传附件

@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);  
}
```

# 富文本框

- 富文本框的实现常见技术用到了Frame标签，并且在Frame中实现了一个常见的HTML网页结构，所以使用普通定位方式无法获取
- 方法1：使用切换子页面语句实现富文本框的输入

`driver.switchTo().frame(iframe);`

- 方法2：找到紧邻的元素，使用Tab键的方法

# 失败截屏，文件名为当前时间

```
public void screenShot(String imageName) {
    File source_file = ((TakesScreenshot) driver)
        .getScreenshotAs(OutputType.FILE);
    try {
        FileUtils.copyFile(source_file, new File("D://demo//" + imageName
            + ".jpg"));
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

@Test
public void testscreenShot() throws IOException {
    this.driver.get("https://www.baidu.com/");
    SimpleDateFormat df = new SimpleDateFormat("yyyyMMdd-HH:mm:ss");
    String fileName = df.format(new Date());
    screenShot(fileName);
}
```

# 精确比较网页截图图片

测试思路：首先截屏，然后将实际图片与预期的图片任一像素一一比对。如有不同，则视为失败的测试用例

@Test

```
public void testImageCompare() throws IOException {
    driver.get("https://www.jd.com/");
    screenshot("actual.jpg");
    File actualFile = new File("d:\\demo\\actual.jpg");
    File expectFile = new File("d:\\demo\\a1.jpg");
    /*
     * 以下部分为两个文件进行像素对比的算法实现，获取文件的像素个数大小，
     * 然后使用循环的方式将两张图的所有像素进行对比，如有任一像素不相同，则退出循环。
     */
    BufferedImage bufileActual = ImageIO.read(actualFile);
    DataBuffer daFileActual = bufileActual.getData().getDataBuffer();
    int sizeActual = daFileActual.getSize();

    BufferedImage bufileExpect = ImageIO.read(expectFile);
    DataBuffer daFileExpect = bufileExpect.getData().getDataBuffer();
    int sizeExpect = daFileExpect.getSize();
    boolean flag = true;
    if (sizeActual == sizeExpect) {
        for (int j = 0; j < sizeActual; j++) {
            if (daFileActual.getElem(j) != daFileExpect.getElem(j)) {
                flag = false;
                break;
            }
        }
    } else
        flag = false;
    assertTrue(flag, "实际截图与期望截图不一致");
}
```

# 本章大纲

7.1 JavascriptExecutor进阶

7.2 其他元素（浮动框/上传/日期/富文本框/截屏）

7.3 操作html5元素

7.4 模拟键盘操作

7.5 自动化框架的基础

- 需要使用JavaScript语句调用HTML5对象提供的内部变量和函数来实现各类操作，可以参考相关HTML5的接口文档来实现各种复杂的测试操作，HTML5的JS接口文档网址为<http://html5index.org/>



# HTML5中的视频播放器

- [http://www.w3school.com.cn/tiy/t.asp?f=html5\\_video](http://www.w3school.com.cn/tiy/t.asp?f=html5_video)

```
@Test
public void testVideo() throws InterruptedException {
    driver.get("http://videojs.com/");
    WebElement video = driver.findElement(By.tagName("video"));
    JavascriptExecutor js = (JavascriptExecutor) driver;
    //获取视频的网络存储地址
    String source = (String) js.executeScript(
        "return arguments[0].currentSrc;", video);
    System.out.println(source);
    assertEquals(source, "http://vjs.zencdn.net/v/oceans.mp4");
    //获取播放时长
    Double videoDuration = (Double) js.executeScript(
        "return arguments[0].duration", video);
    System.out.println(videoDuration.doubleValue());
    //播放
    js.executeScript("return arguments[0].play()", video);
    Thread.sleep(2000);
    //暂停
    js.executeScript("return arguments[0].pause()", video);
}
```

# HTML5中的绘画操作

- moveToElement(toElement,xOffset,yOffset)  
// 以鼠标当前位置或者 (0,0) 为中心开始移动到 (xOffset, yOffset) 坐标轴
- 测试地址<http://literallycanvas.com/>

```
@Test
public void testCanvas1() throws InterruptedException {
    driver.get("http://literallycanvas.com/");
    WebElement canvas = driver.findElement(By
        .xpath("//*[@id='literally-canvas']/div[1]/div[1]/canvas[2]"));
    Actions drawing = new Actions(driver);
    drawing.clickAndHold(canvas).moveByOffset(10, 50).moveByOffset(50, 10)
        .moveByOffset(-10, -50).moveByOffset(-50, -10).release()
        .perform();
    Thread.sleep(5000);
    screenshot("test12.jpg");
}
```

# HTML5中的存储对象

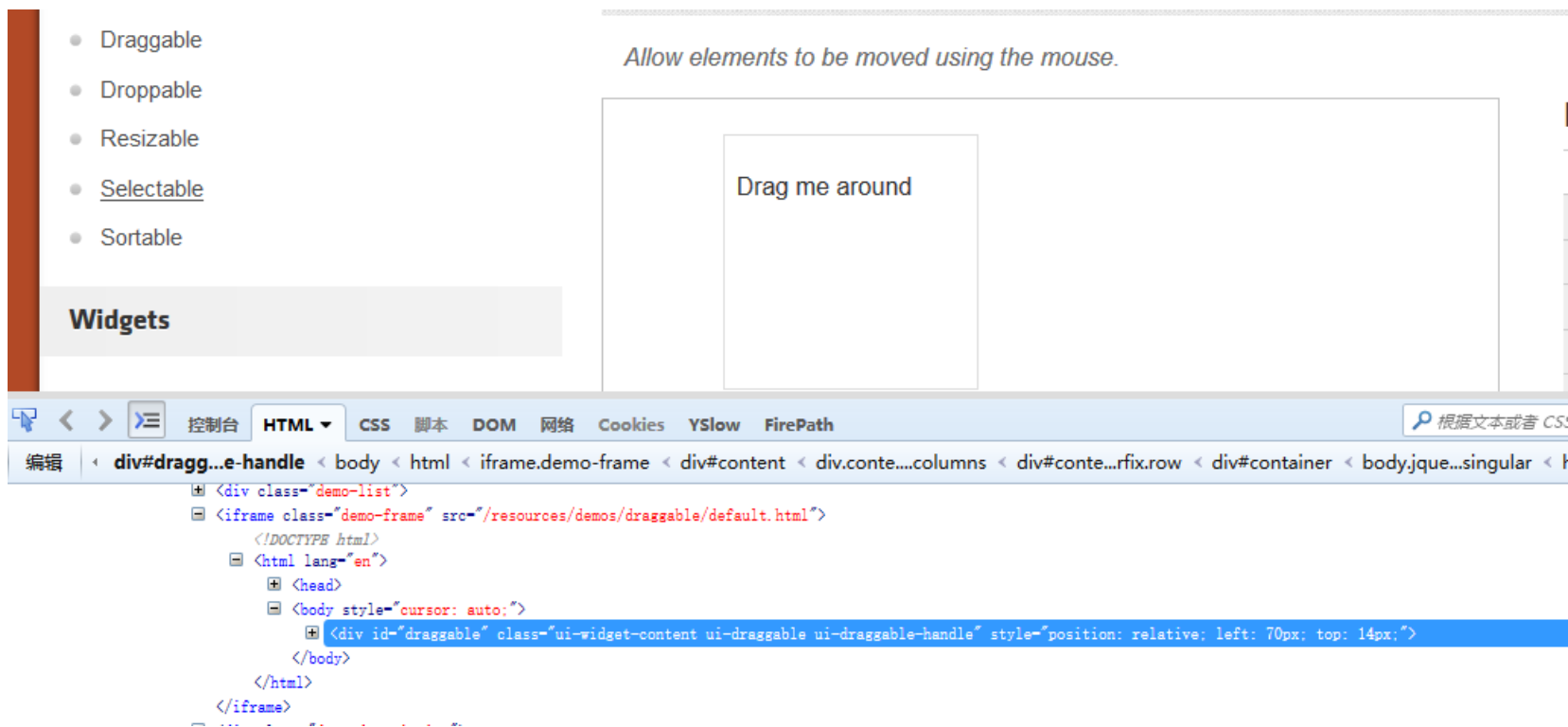
- [http://www.w3school.com.cn/html5/html\\_5\\_webstorage.asp](http://www.w3school.com.cn/html5/html_5_webstorage.asp)

```
@Test
public void testLocalStorage() throws InterruptedException {
    driver.get("http://www.w3school.com.cn/tiy/loadtext.asp?f=html5_webstorage_local");
    String name1 = executeJs("return localStorage.lastname");
    System.out.println(name1);
    assertEquals(name1, "Smith");
    executeJs("localStorage.clear();");
    String name2 = executeJs("return localStorage.lastname");
    System.out.println(name2);
}

@Test
public void testSessionStorage() throws InterruptedException {
    driver.get("http://www.w3school.com.cn/tiy/loadtext.asp?f=html5_webstorage_session");
    String name1 = executeJs("return sessionStorage.lastname");
    assertEquals(name1, "Smith");
    executeJs("sessionStorage.removeItem('lastname');");
    executeJs("sessionStorage.clear();");
}
```

# Drag和Drop

- 使用Actions通过来拖动和移动元素位置
- 测试地址: <http://jqueryui.com/draggable/>



@Test

```
public void testDrag() throws InterruptedException {  
    driver.get("http://jqueryui.com/draggable/");  
    WebElement frame1 = driver.findElement(By.className("demo-frame"));  
    driver.switchTo().frame(frame1);  
    WebElement draggable = driver.findElement(By  
        .xpath("//div[@id='draggable']"));  
    new Actions(driver).dragAndDropBy(draggable, 200, 100).build()  
        .perform();  
    Thread.sleep(5000);  
}
```

# 本章大纲

7.1 JavascriptExecutor进阶

7.2 其他元素（浮动框/上传/日期/富文本框/截屏）

7.3 操作html5元素

7.4 模拟键盘操作

7.5 自动化框架的基础

# Robot对象操作键盘

- 通过Robot对象操作键盘上的按键完成复制、粘贴、切换焦点和回车的常用动作

```
@Test
public void testRobotOperateKeyboard() {
    WebDriverWait wait = new WebDriverWait(driver, 5); // 显示等待，页面是否有搜索框
    wait.until(ExpectedConditions.presenceOfAllElementsLocatedBy(By
        .id("kw")));
    setCtrlVClipboardData("奥运会");
    this.pressTabkey();
    this.pressEnterKey();
}
```

// 实现了Control+V的粘贴，把参数放入剪切板中，然后使用Robot对象的

```
public void setControlVClipboardData(String s) {  
    StringSelection str = new StringSelection(s);  
    Toolkit.getDefaultToolkit().getSystemClipboard().setContents(str, null);  
    robot.keyPress(KeyEvent.VK_COLON); // 按下Control  
    robot.keyPress(KeyEvent.VK_V);  
    robot.keyRelease(KeyEvent.VK_COLON); // 释放Control  
    robot.keyRelease(KeyEvent.VK_V);  
  
}  
public void pressTabkey() {  
    robot.keyPress(KeyEvent.VK_TAB);  
    robot.keyRelease(KeyEvent.VK_TAB);  
}  
public void pressEnterKey() {  
    robot.keyPress(KeyEvent.VK_ENTER);  
    robot.keyRelease(KeyEvent.VK_ENTER);  
}
```



# 本章大纲

7.1 JavascriptExecutor进阶

7.2 其他元素（浮动框/上传/日期/富文本框/截屏）

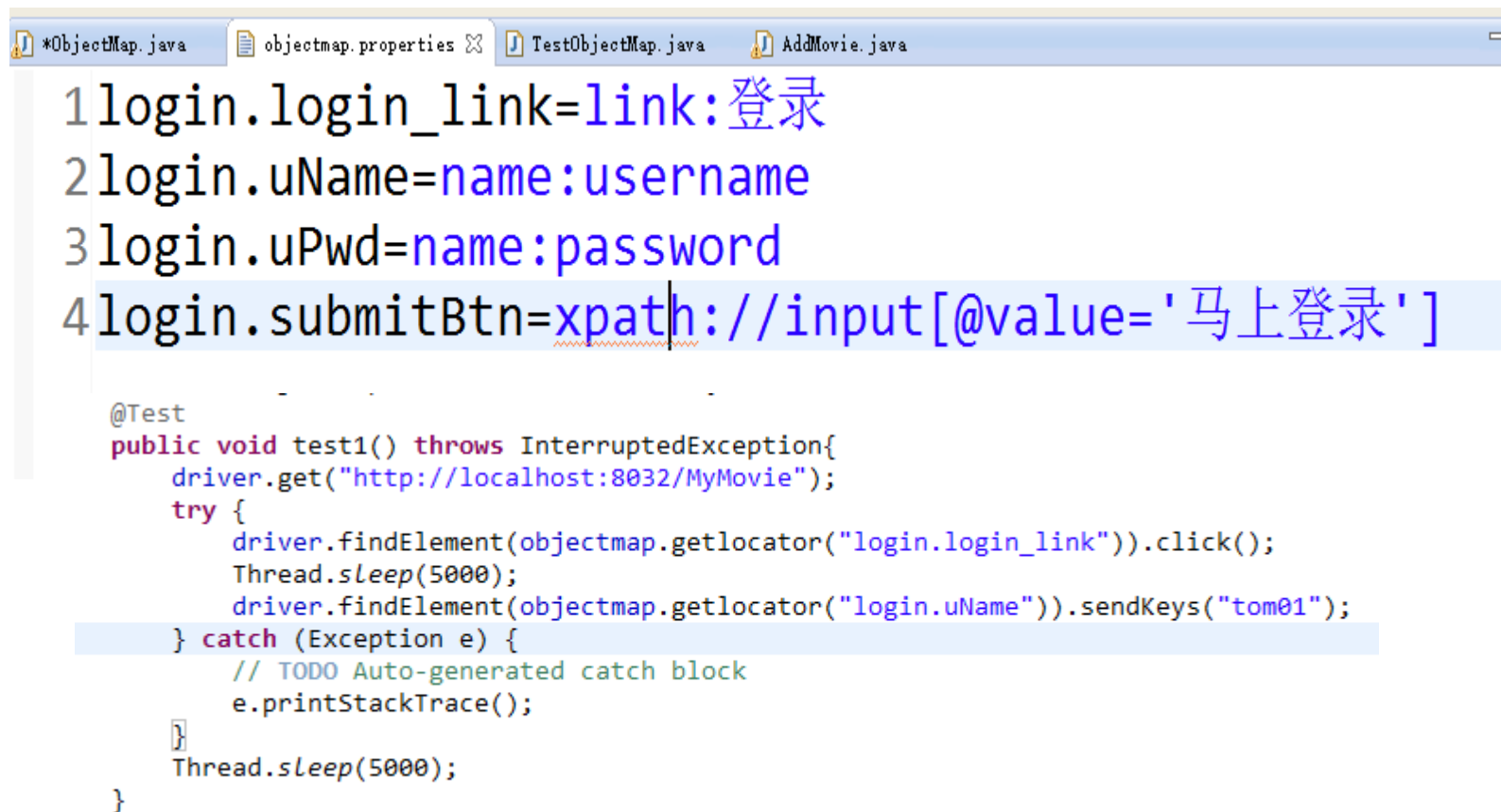
7.3 操作html5元素

7.4 模拟键盘操作

7.5 自动化框架的基础

# 对象库

- 使用配置文件存储被测试页面上的元素的定位方式和定位表达式，做到定位数据和定位程序的分离。



```
*ObjectMap.java  objectmap.properties  TestObjectMap.java  AddMovie.java

1login.login_link=link:登录
2login.uName=name:username
3login.uPwd=name:password
4login.submitBtn=xpath://input[@value='马上登录']

@Test
public void test1() throws InterruptedException{
    driver.get("http://localhost:8032/MyMovie");
    try {
        driver.findElement(objectmap.getlocator("login.login_link")).click();
        Thread.sleep(5000);
        driver.findElement(objectmap.getlocator("login.uName")).sendKeys("tom01");
    } catch (Exception e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    Thread.sleep(5000);
}
```

# 封装操作表格的公用类

```
public class Table {  
  
    WebElement table;  
  
    public Table(WebElement table) {  
        this.table = table;  
    }  
  
    public int getRowCount() {  
        List<WebElement> rows = this.table.findElements(By.tagName("tr"));  
        return rows.size();  
    }  
  
    public int getColCount() {  
        List<WebElement> rows = this.table.findElements(By.tagName("tr"));  
        return rows.get(0).findElements(By.tagName("td")).size();  
    }  
  
    // 获得单元格某行某列的所有内容  
    public WebElement getCell(int rowNo, int ColNo) {  
        List<WebElement> rows = this.table.findElements(By.tagName("tr"));  
        WebElement currentRow = rows.get(rowNo - 1);  
        List<WebElement> cols = currentRow.findElements(By.tagName("td"));  
        WebElement currentCell = cols.get(ColNo - 1);  
        return currentCell;  
    }  
}
```