**selenium武林秘籍**

**一、软件测试比赛提交答案模板部分内容**

**@Test**

**public** **void** testUI() {

takeScreenshot("12306\_search\_UI\_001.png");

}

**@ParameterizedTest**

**@CsvSource({**

//时间分别为当前日期的前一天日期

// 当前日期

// 预售期内任意日期

// 预售期最后一天日期

// 超出预售期期限的后一天

//数据按顺序分别对应from,to,from time,screenShot,代表相同内容

"北京北,南京,2024-10-14,12306\_search\_001.png",

"北京北,南京,2024-10-15,12306\_search\_002.png",

"北京北,南京,2024-10-20,12306\_search\_003.png",

"北京北,南京,2024-10-29,12306\_search\_004.png",

"北京北,南京,2024-10-30,12306\_search\_005.png"

**})**

**public void test\_12306\_search(String from, String to, String fromTime, String screenShot) throws InterruptedException** {

WebElement fromField = driver.findElement(By.*id*("fromStationText"));

fromField.clear();

fromField.sendKeys(from + Keys.***ENTER***);

WebElement toField = driver.findElement(By.*id*("toStationText"));

toField.clear();

toField.sendKeys(to + Keys.***ENTER***);

WebElement fromTimeField = driver.findElement(By.*id*("train\_date"));

fromTimeField.clear();

fromTimeField.sendKeys(fromTime);

WebElement query = driver.findElement(By.*id*("search\_one"));

query.click();

driver.switchTo().window(driver.getWindowHandles().toArray()[1].toString());

Thread.*sleep*(1000);

takeScreenshot(screenShot);

}

**二、以上为比赛模板需求大致格式的展现，接下来写比较重要的几个内容**

**1. 初始化格式**

WebDriver driver =new ChromeDriver();

driver.quit();

最重要的初始化格式，定义driver为类的元素，driver.quit()为结束进程之后释放内存

**2.打开某个浏览器地址（浏览器窗口操作）**

eg:driver.get(写地址内容);

driver.refresh();刷新一次页面,driver.close()为关闭窗口（不代表结束进程结束进程为quit）

*重要的内容：time.sleep(暂停的秒数)用来做到延时等待，尽量打开新网页之后的语句都停顿一下来保证浏览器有足够的反应时间，一般情况：sleep(1)*

//时间停顿也可以用Thread.sleep(2000);

driver.navigate().back();回到上一个页面

driver.manage().window().maximize();放大窗口全屏

driver.manage().window().minimize();最小化窗口

driver.manage().window().setSize(new Dimension(800, 600));自由设置窗口大小

**3.selenium定位方法（我少写一点只加重要内容，说不定会补充）**

大致有八大定位方法，只写重要的

css selector,id,name,xpathhttps://click.meituan.com/t?t=1&c=2&p=4NCVx75zRC08

首先：找到元素的方法是右键需要点到的内容，点击检查，之后点copy，在弹出的栏目里面即能发现这几个元素，一般情况下xpath是定位最准的，省事可能会直接写xpath

driver.findElement(By.cssSelector("#fname"));

driver.findElement(By.id("lname"));

driver.findElement(By.tagName("a"));

driver.findElement(By.xpath("//input[@value='f']"));

检验XPath 定位是否正确可以使用 chrome 的检查模式 -> Console，输入$x('XPath 表达式')例如：$x("//input[@value='f']");



**4.等待方式，分为显式等待、隐式等待、强制等待**

最好用的理所应当的是**强制等待：**Thread.sleep(2000//时间戳);

**隐式等待：***driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);*

当在 DOM 结构中查找元素，且元素处于不能立即交互的状态时，将会触发隐式等待。

简单来说就是，隐式等待的作用就是让这一次测试的整体在因为网速等客观原因没法正常查找的时候，用来等待浏览器刷新完成的中间人。，作用于整个测试周期，也就是说一次测试只需要设置一次隐式等待，若是再次设置会覆盖之前的效果

**显式等待：**简单来说就是专注于等待某个元素，在这个元素出现之前保持等待状态

WebElement element = (new WebDriverWait(driver, Duration.ofSeconds(5)))

.until(ExpectedConditions.presenceOfElementLocated(By.id("kw")));

**5.Selenium常见操作**

A.输入点击清除，分别是send\_keys(‘ ’) click() clear()

driver.findElement(By,ID,’//查找的id内容’).send\_keys(‘’);//括号中内容即需要输入内容

driver.findElement(By,ID,’//查找的id内容’).cllick();

driver.findElement(By,ID,’//查找的id内容’).clear();

B.获取元素属性的方法，logger库

import java.util.logging.Logger;//声明库

private static final Logger logger = Logger.getLogger(demo.class.getName());//定义对象

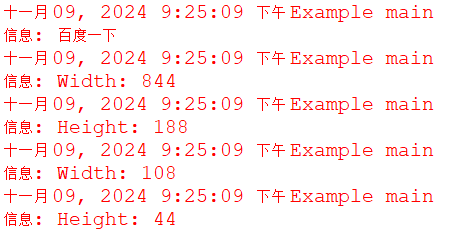
WebElement search = driver.findElement(By.id("su"));

//获取search元素的value属性值并打印

logger.info(search.getAttribute("value"));

//获取search元素的位置坐标值并打印

Point p = search.getLocation();



logger.info("Width: " + p.x);

logger.info("Height: " + p.y);

//获取search元素的大小并分别打印

Dimension size = search.getSize();

int width = size.getWidth();

int height = size.getHeight();

logger.info("Width: " + width);

logger.info("Height: " + height);

//打印页面源代码

logger.info(driver.getPageSource());

熟练掌握不同的定位方法，并根据实际情况灵活应用，是自动化测试工程师的重要技能之一。

**三、Web控件交互进阶**

**1.ActionChains**

import org.openqa.selenium.interactions.Actions;考前先加上这个库名，怕忘

/通过这个类来实现鼠标点击，移动，键盘输入等操作。

WebElement input = driver.findElement(By.id("kw"));//创建输入对象

pause(1)暂停1s

**Actions actions = new Actions(driver);**

actions.sendKeys(input,”abc”).sendKeys(input,Keys.ENTER).build().perform();

perform()方法是 Selenium 中ActionChains类的一个方法，用于执行先前构建的一系列操作，如鼠标或键盘操作。

在运行测试用例时，会按顺序将actions中所有内容按顺序走完一圈。

.sendKeys(input,””)//像所选内容中输入数据

.sendKeys(input,Keys,ENTER)//输入回车

.keyDown(input,Keys.SHIFT)//按住shift

,keyUp(input,Keys.SHIFT)//松开按下的shift

action.keyDown(input,keys.SHIFT).sendKeys(input,”abc”).build().perform();输入了大写abc

//Keys.CONTROL为ctrl键，用来复制粘贴等有奇效

.click(input)//为点击函数.作用就是点击被选中的元素，例如按钮链接等

.clickAndHold(input)//稍微多一点作用就是，点击并按住

.contextClick(input)//右键被选中的元素

actions.pointerAction().click(MouseButton.FROWARD) .build();//点击向前（转发）

actions.pointerAction().click(MouseButton.BACK) .build();//点击向后（后退）

.doubleClick(input)//进行双击

.moveToElement(hoverable).perform;//将光标放到目标元素上

WebElement value1 = driver.findElement(By,id(“value1”));

WebElement value2 = driver.findElement(By,id(“value2”));

actions.dragAndDrop(value1,value2).perform();

把元素1放到元素2的位置

鼠标中键，滚动方法：

WebElement value1 = driver.findElement(By,id(“value1”));

actions.scrollToElement(value1).perform();//滚动到某元素

int delta\_y=value1.getRect().y;

actions.scrollByYOffset(delta\_y);///滚动到该元素的坐标

**2.弹窗的处理**

首先明确一个函数sendKeys(“./文件路径”);可以用于上传文件

Alert alert = driver.switchTo().alert();//获取警告框

alert.accept();//点击警告框的确定

alert.dismiss();//点击警告框的取消

alert.sendKeys(“输入内容”);//向弹窗输入内容

**3.显示等待能力扩展（不重要，正经人都用强制等待）**

WebDriverWait(driver实例，最长等待时间，轮询时间).until(结束条件函数)

WebDriverWait wait = new WebDriverWait(driver,Durationn.ofsecond(10));

WebElement

value =

wait.until(**ExpectedConditions.elementToBeClickable**(By.cssSelector("#success\_btn")));

ExperctedConditions.visibilityOfElementLocated();//判断该元素是否存在可见

ExperctedConditions.urlContains();。判断当且页面URL是否包含指定的字符串

ExperctedConditions.titleIs();//判断当前页面标题是否和目标字符串一致

ExperctedConditions.frameToBeAvailableAndSwitchToit()；//判断指定的frame是否可用

ExperctedConditions.alertIsPresent();//判断是否存在弹窗

**4.frame盒子**

有的元素存在于frame盒子中因此标签可能无法直接精确定位，因此需要转换成frame盒子视角，来进行对标签元素的调用。

driver.switchTo.frame(“frame标签名称”)；//切换到相应的frame句柄

这样写了之后就可以精确定位到盒子标签内的元素了（有的多层标签要分开看（很重要）

driver.switchTo().defaultContent();//切换回html主界面

**5.自动化关键数据的记录**

行为日志（logging模块）

导入logging之后在控制台中即可看到打印对应行为的日志

logging,info(“在运行到此处时希望控制台打印的内容”)；

步骤截图记录（screenShot）//需要专门定义的函数

**private** **void** takeScreenshot(String fileName) {

File screenshotFile = ((TakesScreenshot) driver).getScreenshotAs(OutputType.***FILE***);

**try** {

FileUtils.*copyFile*(screenshotFile, **new** File(fileName));

} **catch** (Exception e) {

e.printStackTrace();

}

}

page source记录

String pageSource = driver.getPageSource();

logger.debug(pageSource);

FileWriter pageSourceFile = new FileWriter("./pageSource.html");

pageSourceFile.write(pageSource);