P9-Pruebas de aceptación con Selenium Webdriver

Pruebas de aceptación de aplicaciones Web

Vamos a automatizar pruebas de aceptación de pruebas emergentes funcionales para una aplicación Web. Utilizaremos la herramienta Selenium Webdriver, con la que crearemos nuestros tests utilizando java y Junit. Utilizaremos el driver para el navegador Firefox, desde el que accederemos a nuestra aplicación web. Implementaremos los tests siguiendo el patrón page object, tal y como se ha explicado en clase.

Bitbucket

El trabajo de esta sesión también debes subirlo a Bitbucket. Todo el trabajo de esta práctica, tanto los proyectos maven que vamos a crear, como cualquier otro documento con vuestras notas de trabajo, deberán estar en la carpeta **P9** de vuestro repositorio**.**

Ejercicios

Los ejercicios de esta sesión estarán contenidos en un proyecto Maven (que deberás crear) denominado "practica9", con valores para ArtifactId y GroupId **practica9** y **ppss**, respectivamente. En el campo de texto Package que nos muestra Netbeans pondremos como valor **ppss**.

1. **Ejercicio 1**. Implementa, utilizando el patrón page object y la clase PageFactory, el test de las transparencias de clase para nuestra aplicación bancaria (http://demo.guru99.com/V4). Para utilizar las anotaciones @FindBy no será suficiente con utilizar la librería con el driver de Firefox, por lo tanto como dependencia utilizaremos el artefacto maven "org.seleniumhq.selenium:selenium-java:2.53.0". Como ya hemos visto, tanto el código fuentes de las *page objects* como los tests, deben compartir la misma ubicación lógica, que será "ejercicio1.pageFactory". Las ubicaciones físicas serán las que hemos explicado en clase. A partir de aquí:
   A) Vamos a crear un nuevo test, al que denominaremos "test_Login_Incorrect()" para probar un caso de prueba en el que introducimos un login y contraseña incorrectos. En este caso, la aplicación debe mostrar una ventana de alerta con el mensaje "User or Password is not valid". Para interaccionar con las ventanas de alerta utilizaremos la clase "Alert". En el recuadro siguiente mostramos los métodos disponibles:

```
//Operaciones sobre ventanas de alerta
//cambiamos el foco a la ventana de alerta
Alert alert = driver.switchTo().alert();
//podemos mostrar el mensaje de la ventana
String mensaje = alert.getText();
//podemos pulsar sobre el botón OK (si lo hubiese)
alert.accept();
//podemos pulsar sobre el botón Cancel (si lo hubiese)
alert.dismiss();
//podemos teclear algún texto (si procede)
alert.sendKeys("user");
```

Ventana de alerta

Una ventana de alerta no es más que una pequeño "recuadro" que aparece en la pantalla proporcionándonos algún tipo de información o aviso sobre alguna operación que intentamos realizar, o también puede solicitarnos algún tipo de permiso para realizar dicha operación

Para implementar el test puedes utilizar el usuario y password mostrado en las transparencias, tiene una validez limitada (cuando nos registramos, el usuario y password podemos utilizarlos durante 20 días, transcurrido este plazo, la aplicación borra la cuenta, por lo que tendremos que volvernos a registrar). Podéis crearos vuestra propia cuenta en cualquier momento, y utilizar vuestro propio login y password, en lugar del que hemos propuesto.

2. **Ejercicio 2**. Vamos a modificar y ampliar nuestro código de pruebas. Concretamente queremos implementar nuevos tests en los que: accederemos al banco con nuestro usuario y password, y a continuación añadiremos un nuevo cliente. Para ello crearemos un nuevo paquete "ejercicio2.pageFactory". En dicho paquete vamos a copiar las clases LoginPage y ManagerPage del ejercicio anterior y sobre ellas haremos lo siguiente:
   A) Modifica el método LoginPage.login(), de forma que devuelva directamente una instancia de ManagerPage (utilizando la clase PageFactory).
   B) Modifica la clase ManagerPage para poder acceder desde ella al servicio "newCustomer()", utilizando el patrón Page Object. Al igual que antes, este servicio debería devolver una instancia de un nuevo page object, por ejemplo de tipo NewCustomerPage.
   C) Implementa la nueva clase NewCustomerPage, según el patrón Page Object.
   D) Implementa un test para probar la creación de un cliente nuevo, por ejemplo en la clase TestNewClient.

IMPORTANTE !!

Cuando creamos un nuevo cliente, la aplicación le asigna de forma automática un identificador. Este identificador se utilizará posteriormente en las operaciones para editar un cliente, o borrar un cliente, por lo tanto, deberías almacenar dicho valor (en la clase que contiene los tests) para poder acceder a él en futuros tests que necesiten trabajar con un cliente concreto. También deberías imprimir dicho valor por pantalla por si el código de test contiene errores y necesitas deshacer manualmente la operación de creación del nuevo cliente y poder repetir la ejecución del test.

Los datos de prueba serán:

- como usuario y password registrados, los mismos de antes (login: mngr34733, y password: AbEvydU), o bien el login y password que te hayas creado. Una vez introducimos los datos pulsaremos sobre el botón "Login"
- los datos del nuevo cliente pueden ser:
  - Nombre del cliente: tu login del correo electrónico de la ua
  - Género: "m" (o "f")
  - Fecha de nacimiento: tu fecha de nacimiento en formato "dd/mm/aaaa", podemos seleccionarla desplegando el calendario
  - Dirección: "Calle x",
  - Ciudad: "Alicante", (para el campo State puedes poner el mismo)
  - Pin: "123456" (puedes poner cualquier otra secuencia de 6 dígitos)
  - Número de móvil: "99999999"

- e-mail: tu email institucional
- password: “123456”, (o cualquier otra secuencia de dígitos)

- Una vez introducidos los datos pulsaremos sobre el botón “Submit”

El resultado del test será “pass” si hemos completado con éxito la operación. Para ello podemos comprobar que en la página aparece el texto (o una parte de él) “Customer Registered Successfully!!!”.

Observaciones a tener en cuenta:

- Recuerda que nuestros tests NO deben contener código Webdriver.
- Recuerda que debes verificar, cada vez que cambiemos de página, que estamos en la página correcta, para ello puedes utilizar el título de la página o por ejemplo el login del administrador (en el caso de la página resultante de hacer login).
- El campo e-mail del cliente es único para cualquier cliente. Si intentamos dar de alta un cliente con un e-mail ya registrado, la aplicación nos mostrará un mensaje indicando que no se puede repetir el valor de dicho campo.

E) Teniendo en cuenta que la aplicación asigna un identificador a cada nuevo cliente creado, para poder repetir la ejecución del test que acabamos de implementar deberíamos “eliminar” el cliente que acabamos de crear después de ejecutar con éxito el test. Modifica el test anterior, de forma que se borre la información del nuevo cliente creado. Para ello tendrás que crear una nueva *page object*, que puedes llamar “DeleteCustomerPage” y que representa la página de nuestra aplicación para borrar un cliente. Una vez implementada la nueva page object, modifica el test para borrar el nuevo cliente creado (tendrás que utilizar el identificador del cliente para poder borrarlo), y además aceptar los mensajes de dos alertas que nos aparecerán para confirmar que queremos borrar dicha información.

F) Implementa un segundo test para probar el comportamiento de nuestra aplicación cuando intentamos crear un nuevo cliente duplicado (con un e-mail que ya existe). En este caso, después de introducir los datos del cliente y pulsar el botón “Submit” nos debe aparecer un mensaje de alerta con el mensaje "Email Address Already Exist !!”. Para garantizar que el cliente que introducimos como prueba ya existe, debes añadirlo previamente, y a continuación, repetir la operación para asegurarnos de que efectivamente se trata de un cliente repetido. Debemos comprobar que efectivamente la aplicación nos muestra el mensaje anterior y que no nos deja insertar dos veces el mismo cliente. Al igual que ocurría con el primer test, deberías borrar los datos del cliente creado al final del mismo para poder repetir con las mismas condiciones la ejecución de este segundo test.

A continuación mostramos una captura de pantalla de los archivos del proyecto Maven para esta práctica: