Práctico 13
Waits y Factories

Factory

Ejercicio
Crear una clase llamada **DocusignFactory** y una clase llamada **DocusignFactoryTest**.
DocusignFactoryTest debe acceder al sitio "https://go.docusign.com/o/trial/" y debe tener un método de test que valide que el título de Docusign sea "DocuSign Free Trial"

DocusignFactory debe invocar 3 veces a la clase DocusignFactoryTest creando nuevas instancias de la misma.

Ejercicio
Crear una clase llamada **ShopifyTest** y una clase llamada **ShopifyFactory**.
ShopifyTest debe tener una variable de clase privada de tipo int
También, debe contar con un constructor que reciba un valor por parámetro, y setee esa variable con ese valor.
La clase **ShopifyTest** debe tener un constructor que reciba y setee ese parámetro. El constructor sin parámetros, debe setear el parámetro en 0.
El @beforeMethod debe acceder a **"https://es.shopify.com/"**
Agregar un método de test llamado testButtons que obtenga todos los botones de la página, e imprima en pantalla el texto del botón que coincide con el parámetro.

La clase **ShopifyFactory**, debe tener un método **@Factory** que instancie la clase **ShopifyTest** de forma tal que cada instancia imprima un botón diferente.

Thread Sleep

Ejercicio
Crear un método que acceda a Salesforce y espere 5 segundos antes de hacer click en “Forgot password”.

Waits Implícitos

Ejercicio
Crear un método llamado **implicitWaitTest** que acceda a Salesforce y espera hasta 10 segundos antes de tirar una excepción.
Hacer click en Forgot Account

Waits Explicitos

Ejercicio
Acceder a **https://www.spotify.com/uy/signup/**
Completar el email y la confirmación del email con test@test.com
Esperar que aparezca el mensaje de error utilizando un explicit wait
Validar que el mensaje de error sea “**Este correo electrónico ya está conectado a una cuenta.”**

Parametrización y Runners

Ejercicio

Dentro de la carpeta clase13, crear un archivo testng.xml.

Realizar una clase llamada testParametrizables y un método llamado pruebaConParametros que reciba por parámetro un tipo de tagName. En base a si el parámetro es **h1, h2 y h3**

mostrar en pantalla lo que se va a imprimir, y obtener por su tagname todos los elementos. Luego imprimirlos en pantalla. En caso de que no se encuentren elementos, se debe mostrar un mensaje indicando que no se encontraron seleccionados.

Dependent tests

Ejercicio
Crear una clase llamada **dependentGroups** que importe utilice la notación *dependsOnGroups* y groups en el cual el método **testOne**, **testTwo** y un método **testThree.** El método 2 y 3 pertenecen a un mismo grupo, y el método 1, depende de la ejecución del grupo 1

Ejercicio
Crear una clase llamada **DependentTest**.
Dividir el test en métodos dependientes uno de otros, de forma tal que el último test, valide los mensajes de error desplegado en pantalla al clickear el botón de registrarse. Se debe completar solo los dos primeros campos del formulario de registro y usar la notación: **dependentOnMethods**. Agregar asserts y waits