[Solución]
Práctico 12: Testng

Notaciones de Testng

Ejercicio
Utilizar las notaciones de testng: @beforeTest @beforeClass @beforeMethod @afterMethod @afterClass y @afterTest
Crear 3 métodos de test, que muestren en pantalla un mensaje

```java
@BeforeTest
public void beforeTest() {
    System.out.println("@BeforeTest");
}

@BeforeClass
public void init() {
    System.out.println("@BeforeClass"); }

@BeforeMethod
public void beforeMethodTest() {
    System.out.println("@BeforeMethod ");
}

@Test
public void test1() {
    System.out.println("Ejecutando Test 1");
}

@Test
public void test2() {
    System.out.println("Ejecutando Test 2");
}

@AfterTest
public void afterTestMethod() {
    System.out.println("@AfterTest"); }

@AfterMethod
public void afterMethod() {
    System.out.println("@AfterMethod"); }

@AfterClass
public void afterClass() {
    System.out.println("@AfterClass"); }
```

Spotify con Testng

Crear una clase llamada **testngSpotify**

Ejercicio
Crear un método llamado **setUp**. Debe inicializar el driver el @BeforeMethod

```
WebDriver driver;

@BeforeMethod
public void setup() {
    System.setProperty("webdriver.chrome.driver", "drivers/chromedriver");
    WebDriver driver = new ChromeDriver();
}
```

Ejercicio
Crear un método llamado verifySpotifyTitle
Ingresar a **https://www.spotify.com/uy/**
Verificar que el título sea "**Escuchar es todo - Spotify**"

```
@Test
public void verifySpotifyTitle(){
    String expectedTitle = "Escuchar es todo - Spotify";
    String actualTitle = driver.getTitle();
    Assert.assertEquals(actualTitle, expectedTitle);
}
```

Ejercicio
Crear un método llamado **verifySignupUrl**
Ingresar a **https://www.spotify.com/uy/**
Utilizar xpath con caminos relativos para acceder al botón de Registrar
Validar que la url actual, contenga *signup*

```java
@Test
public void verifySignupUrlTest(){
    driver.findElement(By.xpath("//a[@href='https://www.spotify.com/uy/signup/']")).click();
    String currentUrl = driver.getCurrentUrl();
    Assert.assertTrue(currentUrl.contains("signup"));
}
```

Ejercicio

Crear un método llamado **invalidEmailTest**
Ingresar a spotify y hacer click en Registrar
Completar el email con un email inválido: “test.com”
Validar que se despliegue el error: “La dirección de email que proporcionaste no es válida.”

```java
@Test
public void invalidEmailTest(){
    driver.findElement(By.xpath("//a[@href='https://www.spotify.com/uy/signup/']")).click();
    driver.findElement(By.name("email")).sendKeys( ...charSequences: "test.com");
    driver.findElement(By.name("confirm")).sendKeys( ...charSequences: "test.com");
    WebElement emailErrorMsg = driver.findElement(By.xpath("//span[contains(text(),'Este correo electrónico no es válido.')]
    Assert.assertEquals(emailErrorMsg.getText(), expected: "Este correo electrónico no es válido. Asegúrate de que tenga un
}
```

Ejercicio

Crear un método llamado **validateExistingEmail**
Ingresar a spotify y hacer click en Registrar
Completar el email con uno inválido: **“test@test.com”**
Validar que se despliegue el error: **“Lo sentimos, este correo ya está registrado.”**

```java
@Test
public void validateExistingEmail() throws InterruptedException {
    driver.findElement(By.xpath("//a[@href='https://www.spotify.com/uy/signup/']")).click();

    driver.findElement(By.id("email")).sendKeys( ...charSequences: "test@est.com");
    driver.findElement(By.id("confirm")).sendKeys( ...charSequences: "test@test.com");

    Thread.sleep( millis: 1000);
    WebElement emailErrorMessage = driver.findElement(By.xpath("//*[contains(text(), 'Este correo electrónico ya está conectado a una cuenta.')]"));

    Assert.assertEquals(emailErrorMessage.getText(), expected: "Este correo electrónico ya está conectado a una cuenta. Inicia sesión.");
}
```

Ejercicio

Crear un método llamado **checkEqualEmailsError**
Ingresar a spotify y hacer click en Registrar
Completar el email con uno válido: “**test999@test.com**”
En el campo de confirmar email, colocar otro: “**hola@hola.com**”
Situarse en otro campo y completarlo
Validar que se despliegue el error: “Las direcciones de correo electrónico no coinciden.”

```java
@Test
public void checkEqualEmailsError() throws InterruptedException {
    driver.findElement(By.xpath("//a[@href='https://www.spotify.com/uy/signup/']")).click();

    driver.findElement(By.id("email")).sendKeys( ...charSequences: "selenium@cursos.com");
    driver.findElement(By.id("confirm")).sendKeys( ...charSequences: "hola@hola.com");
    driver.findElement(By.id("password")).sendKeys( ...charSequences: "holamundo123");

    Thread.sleep( millis: 1000);
    WebElement emailErrorMessage = driver.findElement(By.xpath("//*[contains(text(), 'Las direcciones de correo electrónico no coinciden')]"));

    Assert.assertEquals(emailErrorMessage.getText(),  expected: "Las direcciones de correo electrónico no coinciden.");
}
```

Ejercicio

Crear un método llamado **checkErrorMessages**
Ingresar a spotify y hacer click en Registrar
Validar que los mensajes de error sean desplegados en los campos obligatorios
Agregar variables estáticas para cada mensaje

```java
private static final String EMAIL_ERROR = "Es necesario que introduzcas tu correo electrónico.";
private static final String CONFIRMATION_ERROR = "Es necesario que confirmes tu correo electrónico.";
private static final String PASSWORD_ERROR = "Debes introducir una contraseña.";
private static final String PROFILE_ERROR = "Introduce un nombre para tu perfil.";
private static final String DAY_ERROR = "Indica un día del mes válido.";
private static final String MONTH_ERROR = "Selecciona tu mes de nacimiento.";
private static final String YEAR_ERROR = "Indica un año válido.";
private static final String SEX_ERROR = "Selecciona tu sexo.";
private static final String CAPTCHA_ERROR = "Confirma que no eres un robot.";

@Test
public void checkErrorMessages() throws InterruptedException {
    driver.findElement(By.xpath("//a[@href='https://www.spotify.com/uy/signup/']")).click();
    Thread.sleep( millis: 2000);
    driver.findElement(By.xpath("//*[@type='submit']")).click();

    WebElement email_message = driver.findElement(By.xpath("//*[contains(text(), 'Es necesario que introduzcas tu correo electr
    WebElement confirmation_message = driver.findElement(By.xpath("//*[contains(text(), 'Es necesario que confirmes tu correo e
    WebElement password_message = driver.findElement(By.xpath("//*[contains(text(), 'Debes introducir una contraseña.')]"));
    WebElement profile_message = driver.findElement(By.xpath("//*[contains(text(), 'Introduce un nombre para tu perfil.')]"));
    WebElement day_message = driver.findElement(By.xpath("//*[contains(text(), 'Indica un día del mes válido.')]"));
    WebElement month_message = driver.findElement(By.xpath("//*[contains(text(), 'Selecciona tu mes de nacimiento.')]"));
    WebElement year_message = driver.findElement(By.xpath("//*[contains(text(), 'Indica un año válido.')]"));
    WebElement sex_message = driver.findElement(By.xpath("//*[contains(text(), 'Selecciona tu sexo.')]"));
    WebElement captcha_message = driver.findElement(By.xpath("//*[contains(text(), 'Confirma que no eres un robot.')]"));

    Assert.assertEquals(email_message.getText(), EMAIL_ERROR);
    Assert.assertEquals(confirmation_message.getText(), CONFIRMATION_ERROR);
    Assert.assertEquals(password_message.getText(), PASSWORD_ERROR);
    Assert.assertEquals(profile_message.getText(), PROFILE_ERROR);
    Assert.assertEquals(day_message.getText(), DAY_ERROR);
    Assert.assertEquals(month_message.getText(), MONTH_ERROR);
    Assert.assertEquals(year_message.getText(), YEAR_ERROR);
    Assert.assertEquals(sex_message.getText(), SEX_ERROR);
    Assert.assertEquals(captcha_message.getText(), CAPTCHA_ERROR);
}
```

Ejercicio con priority

Agregar priority descendiente a los tests.

```java
@Test (priority = 0)
public void primerTest(){
    System.out.println("Test 0");
}


@Test (priority = 2)
public void segundoTest(){
    System.out.println("Test 2");
}

@Test (priority = 1)
public void tercerTest(){
    System.out.println("Test 1");

}

@Test (priority = 3)
public void cuartoTest(){
    System.out.println("Test 3");
}
```

Salesforce con Testng

Ejercicio
Crear una clase llamada **testngSalesforce**
Crear una variable final estática que acceda a "https://login.salesforce.com/"

```java
public static final String SALEFORCE_URL = "https://login.salesforce.com/";
```

Ejercicio
Crear un método llamado **validateSalesforceLogoTest**
El test debe mostrar el tagName del id logo en pantalla y su atributo "alt'
El orden de prioridad de este test, debe ser 1

```java
@Test (priority = 1)
public void validateSalesforceLogo() {
    WebElement e = driver.findElement(By.id("logo"));
    System.out.println(e.getTagName());
    System.out.println(e.getAttribute( s: "alt"));
}
```

Ejercicio
Crear un método llamado **RememberMeIsSelected**
Ingresar al sitio: https://login.salesforce.com/?locale=eu
Hacer click en el botón de Remember me
Validar que el checkbox está seleccionado
El orden de prioridad de este test, debe ser 4

```java
@Test (priority = 4)
public void RememberMeIsSelected() {
    WebElement rememberMe = driver.findElement(By.name("rememberUn"));
    rememberMe.click();
    Assert.assertTrue(rememberMe.isSelected());
}
```

Ejercicio
Método **FooterIsValid**
Validar que el footer tenga **"All rights reserved"**
El orden de prioridad de este test, debe ser 2

```java
@Test (priority = 2)
public void FooterIsValid() {
    driver.get("https://login.salesforce.com/?locale=eu");
    WebElement footer = driver.findElement(By.id("footer"));
    Assert.assertTrue(footer.getText().contains("All rights reserved"));
}
```

Ejercicio
Ignorar uno de los 3 tests a elección **(enabled = false)**

@Test (enabled = false)

Ejercicio
Crear una clase llamada **testng.xml**
Esta clase debe permitir correr todos los tests de la clase **testngSalesforce**

```xml
<suite name="Test suites">

    <test name="Ejemplos de Testng">
        <classes>
            <class name="clase4.testngSpotify"/>
        </classes>
    </test>
</suite>
```

Ejercicio
Crear dos grupos: successTests y failed tests

```
public class groupTests {

    public String URL = "https://www.spotify.com/uy";
    public WebDriver driver;

    @Test(groups = { "successTests", "failTests" })

    @BeforeTest
    public void setup(){
        GetProperties properties = new GetProperties();
        String chromeDriverUrl = properties.getString(
        System.setProperty("webdriver.chrome.driver", c
        driver = new ChromeDriver();
    }

    @Test(groups = { "successTests" })
    public void successTest1(){

    }

    @Test(groups = { "successTests" })
    public void successTest2(){

     }

    @Test(groups = { "successTests" })
    public void successTest3(){
     }


    @Test(priority = 1, groups = { "failTests" })
    public void failTest1(){

    }

    @Test(groups = { "failTests" })
    public void failTest2(){
     }
}
```

Agregar ambos grupos al el testng.xml

```
<suite name="Test suites" verbose="1" >
    <parameter name="specificTag" value="h2"/>
    <test name="All the Tests">

        <groups>
            <run>
                <include name = "successTests" />
                <include name = "failTests" />
            </run>
        </groups>

        <classes>
            <class name = "clase4_groups.groupTests" />
        </classes>

    </test>
</suite>
```

Ejercicio

Agregar 2 notaciones testng a un test: priority=1 y group = “failTests”

```
@Test (priority = 1, groups = "failTests")
```

********** Ejercicio GIT **********

Ejercicio

Crear un método llamado método **LoginFailureTest**

En el sitio de salesforce: "**https://login.salesforce.com/?locale=eu**"

Validar que se encuentre el logo en el sitio (utilizar un WebElement)

Completar el username con “test@test.com”

Completar el campo Password con “123466”

Hacer click en Login

Imprimir en pantalla el mensaje de error

El orden de prioridad de este test, debe ser 3

```
@Test  (priority = 3)
public void LoginFailureTest() {
    driver.get("https://login.salesforce.com/?locale=eu");
    WebElement userName = driver.findElement(By.id("username"));
    userName.sendKeys( ...charSequences: "test@test.com");

    WebElement password = driver.findElement(By.id("password"));
    password.sendKeys( ...charSequences: "holamundo");

    driver.findElement(By.id("Login")).click();

    WebElement elementError = driver.findElement(By.id("error"));

    System.out.println( elementError.getText());

    String expectedError = "Your access to salesforce.com has been disabled by " +
            "your System Administrator. Please contact your Administrator for" +
            " more information.";
    Assert.assertEquals(expectedError, elementError.getText());
}
```

Parte 4: Parámetros

Ejercicio
Agregar en el testng.xml, un parámetro de tipo String llamado specificTag y que su valor sea h2

```
<parameter name="specificTag" value="h2"/>
```

Crear una clase llamada spotifyTestWithParameters
Agregar el @setup y un método llamado **spotifyTags**. Este método deberá imprimir todas los estilos H del sitio, dependiendo del parámetro recibido. Si no hay ningún parámetro, se deberá mostrar los h1

```
@Test
@Parameters({"specificTag"})
public void spotifyRegistrationForm(@Optional("h1") String tagName){
    driver.get(URL);

    List<WebElement> elements = driver.findElements(By.tagName(tagName));

    if (tagName.equalsIgnoreCase( anotherString: "h1")){
        System.out.println("Se mostraran los h1 ");
    } else if (tagName.equalsIgnoreCase( anotherString: "h2")){
        System.out.println("Se mostraran los h2 ");
    } else if (tagName.equalsIgnoreCase( anotherString: "h3")){
        System.out.println("Se mostraran los h3 ");
    }

    if (elements.size() > 0) {
        System.out.println("No se encontraron elementos");
    } else {
        for (WebElement e: elements ) {
            System.out.println(e.getText());
        }
    }
}
```