



Tecnologías Web: Pruebas de Web

Desarrollo del software

- Compilación
- Empaque
- Pruebas
 - Unitarias
 - Integración
 - **Sistema**
 - **Aceptación**
- Despliegue



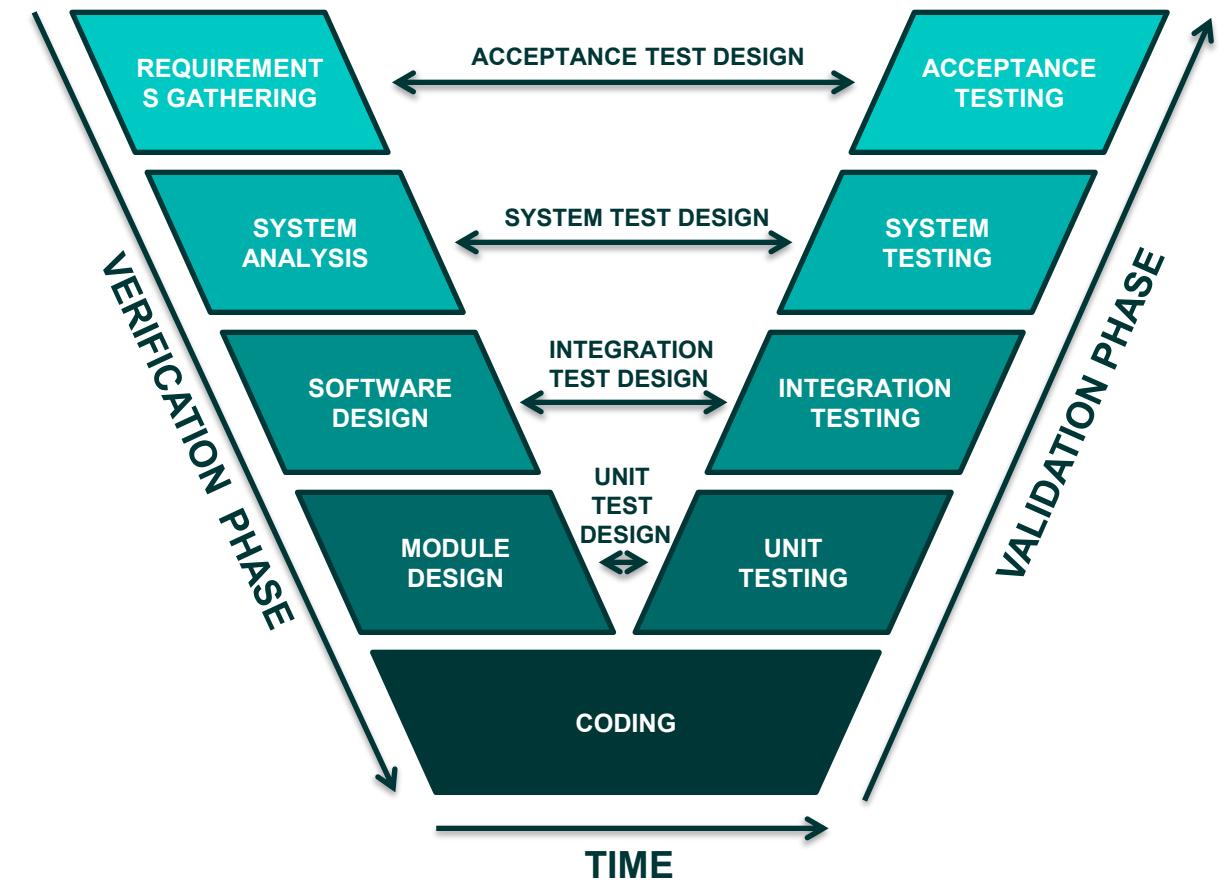
Repaso ISOFT- Spoiler Pruebas

- **¿Qué es una prueba?**
 - **ISO/IEEE 24765:** Actividad en la que un sistema o componente se ejecuta bajo unas condiciones determinadas, los resultados son observados o grabados y se evalúa alguno de sus aspectos.
- Diferentes **tipos de pruebas** dependiendo a que **calidad** se enfoque:
 - Funcional
 - Seguridad
 - Usabilidad
 - Rendimiento...

Repaso ISOFT- Spoiler Pruebas

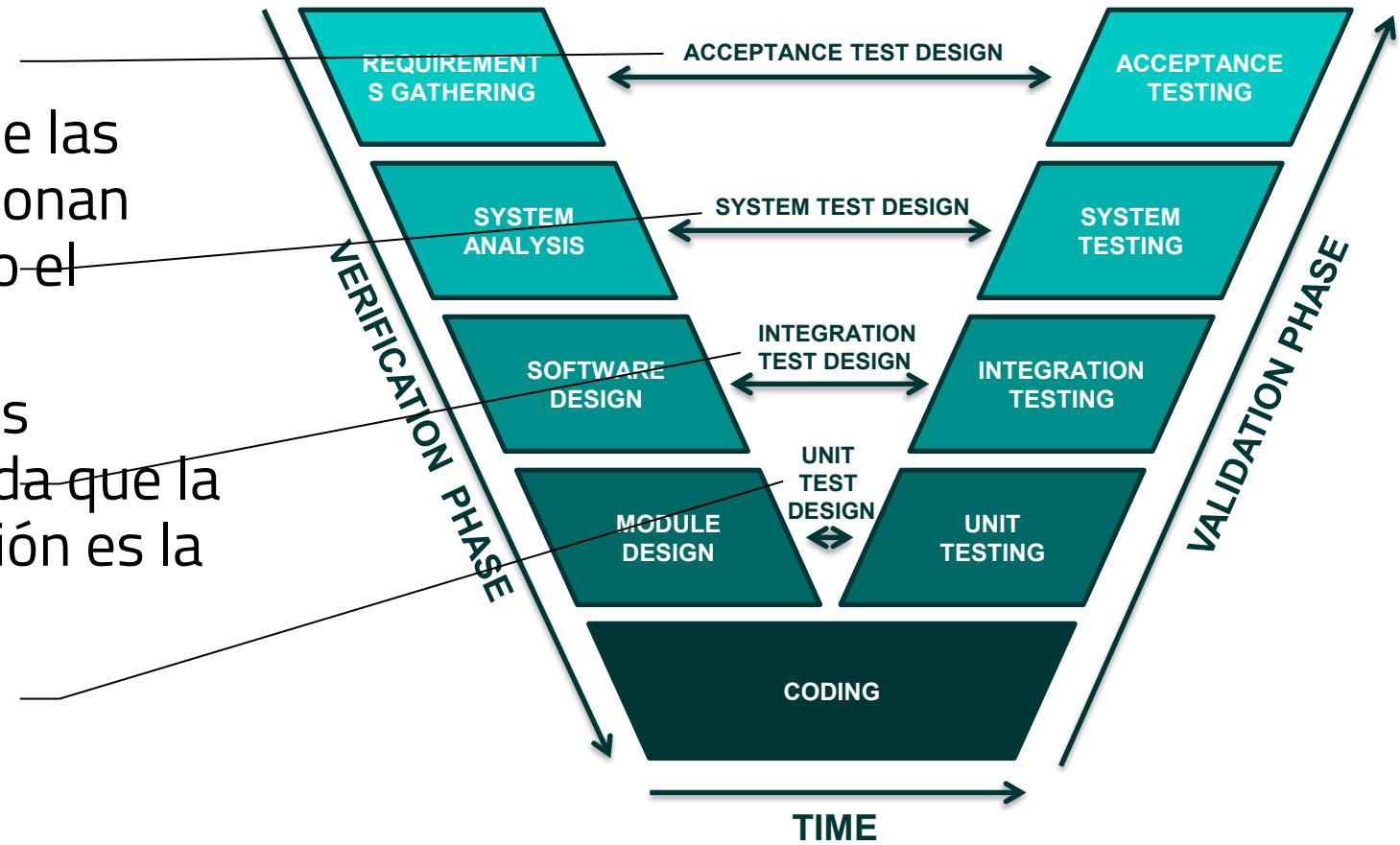
■ Métodos de pruebas:

- **Caja blanca:** garantizan que las operaciones internas funcionan acorde a la especificación o el proceso diseñado
- **Caja negra:** se centra en las entradas y las salidas, valida que la funcionalidad de la aplicación es la deseada, sin entrar en su funcionamiento interno.



Repaso ISOFT- Spoiler Pruebas

- Métodos de Prueba
 - **Caja blanca:** se centra en las operaciones internas que las unidades funcionan acorde a las especificaciones. o el Probar a añadir un vehículo, comenzar un repostaje
 - **Caja negra:** se centra en las entradas y salidas para comprobar que la función es la deseada. Probar que las APIs de RESTEasy
 - Probar los métodos De validación de DNI Matrículas, ODO...



Pruebas de Sistema y Pruebas de Aceptación

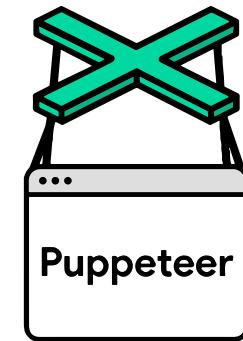
- Prueba de Sistema (E2E)
 - Valida de principio a fin la aplicación, empezando por la interacción del usuario, hasta niveles bajos como la persistencia/inserciones en BBDD.
- Pruebas de Aceptación
 - Valida si el sistema satisface sus criterios de aceptación, permitiendo discernir al cliente cuando “aceptar” este.
- Ambas son pruebas de **caja negra**

Pruebas de sistema

- Manuales



- Automáticas



Pruebas de Sistema automatizadas

DIFERENTES FRAMEWORKS

MISMA IDEA

SIMULAR ITERACCIÓN USUARIO-UI

INTERFACES PROGRAMACION – PROTOCOLOS + ARCHIVOS DE SCRIPT

Selenium

- Framework de pruebas Opensource para aplicaciones web
- Provee de herramientas que permiten:
 - Grabar y reproducir pruebas
 - DSL (en C#, Ruby, Java..) para implementar casos de prueba.
- Estas pruebas emplean agentes remotos que reproducen las acciones del usuario:
 - Selenium<v2.0 → Selenium Remote Control
 - Selenium>v2.0 → **WebDriver**

Selenium: WebDriver

- W3C: interfaz de control remota que permite la introspección y control de agentes de usuario. Provee de un protocolo de lenguaje neutral para instruir el comportamiento de los navegadores
- Para nosotros (y la mayoría de veces) el agente será un **navegador**.



Selenium: WebDriver

- Dispone de la mayoría de acciones que un usuario real podría hacer en la UI:
 - Navegar hacia una URL
 - Redimensionar el navegador
 - Hacer clic
 - Introducir texto
 - Obtener un elemento de la UI
 - Esperar que se cargue o aparezca un elemento
 - Muchas más...

Webdriver: Uso

■ Importar las dependencias:

- Seleniumhq-java
- WebDriver:
 - Firefox-driver
 - Edge-driver
 - Chrome-driver
 - Safari-driver...

```
<dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>4.26.2</version>
</dependency>
<dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-firefox-driver</artifactId>
    <version>4.26.2</version>
</dependency>
```

WebDriver: Instanciación

- Para instanciar un nuevo driver:

```
WebDriver driver = new FirefoxDriver();
```

- Para liberarlo:

```
driver.quit();
```

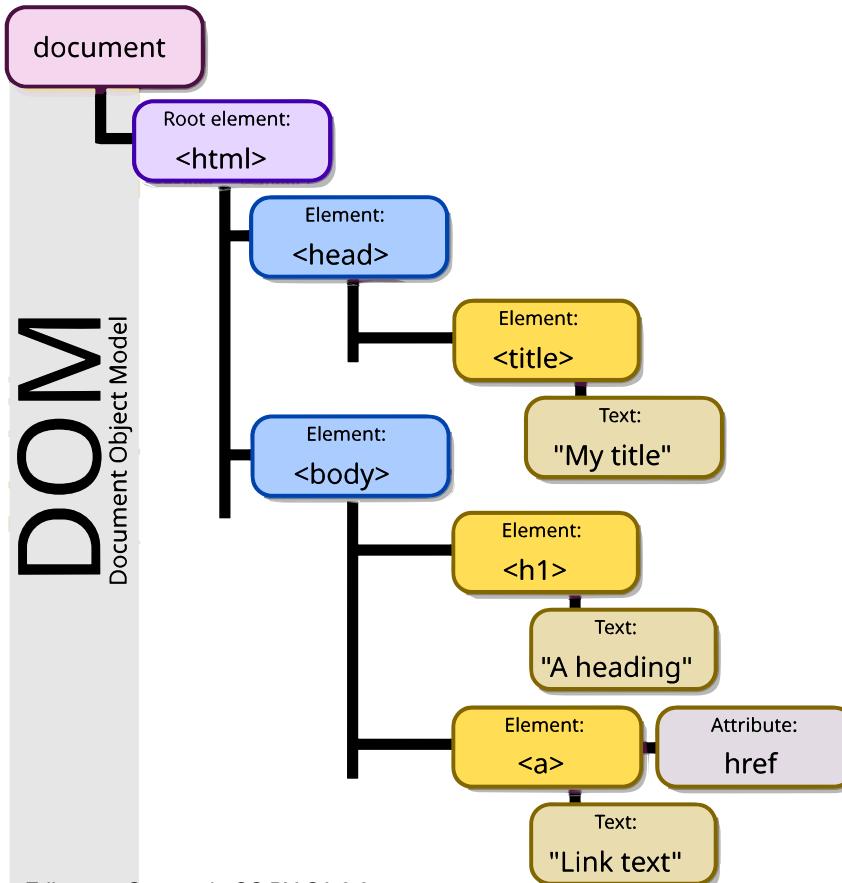
WebDriver: Interacción con elementos

- Navegar: driver.get (URL)
- Obtener un elemento:
`WebElement element= driver.findElement(By)`
- Hacer click: element.click()

Clase By en Selenium

- La clase By se emplea para buscar elementos en el documento, algunos ejemplos disponibles:
 - By.Id (X): busca el/los elemento con el id de campo X, e.g. el textfield “password”
 - By.CssSelector(X): busca los elementos pertenecientes a la clase CSS X, ejemplo “sub-title”
 - By.TagName(X): busca los elementos pertenecientes a la etiqueta X, ejemplo “<tr>” para obtener las filas en una tabla.
 - By.Xpath(X): busca los elementos localizados en el path XML del DOM especificado

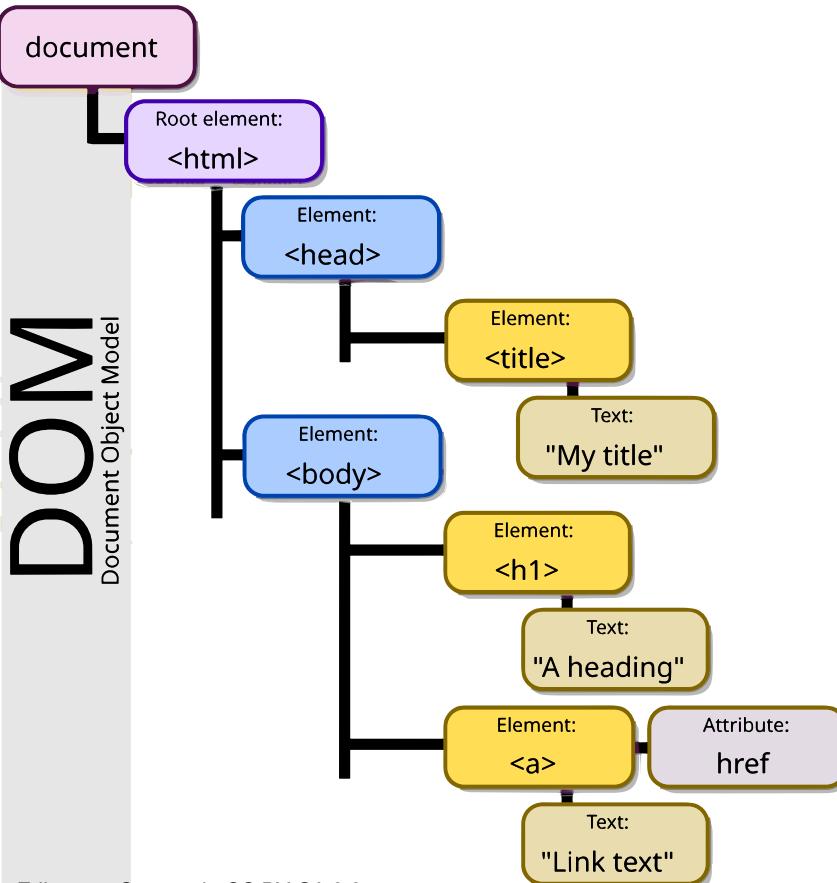
DOM y XPath



By Birger Eriksson - Own work, CC BY-SA 3.0,
<https://commons.wikimedia.org/w/index.php?curid=18034500>

- Pensad en el Xpath como las rutas en Windows/Linux
- Posición del elemento respecto al nodo raiz, el elemento "A heading", estaría:
/html/body/a
- Si tuviésemos varios div dentro de body:
/html/body/div[0] → 1er div
/html/body/div[0] → 2o div
...

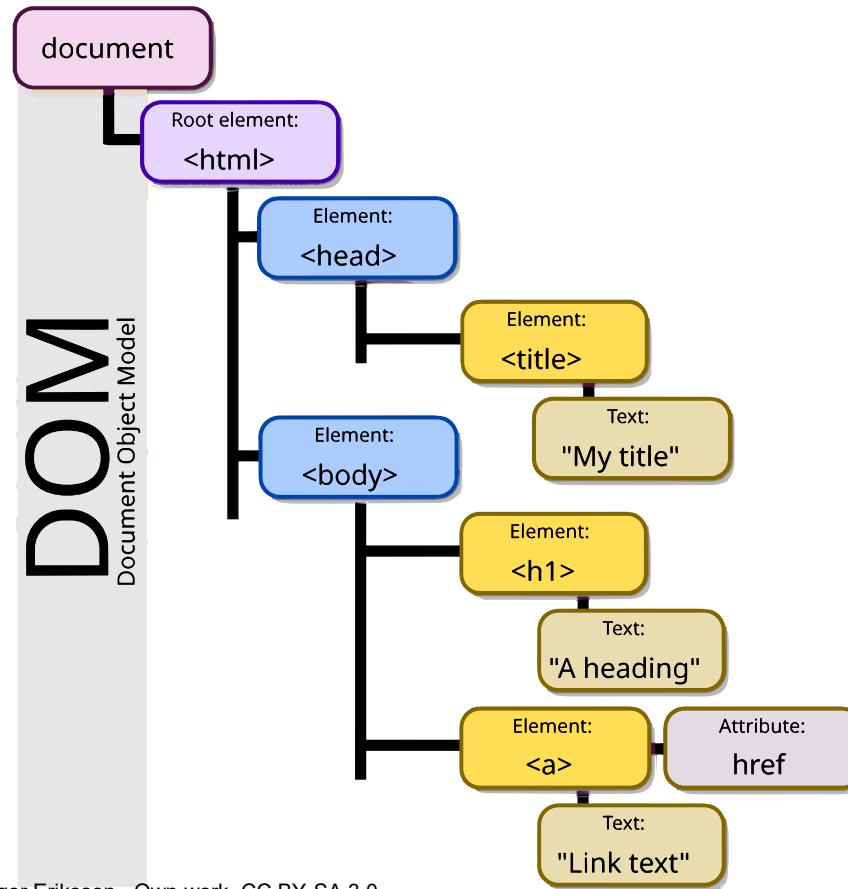
DOM y XPath



By Birger Eriksson - Own work, CC BY-SA 3.0,
<https://commons.wikimedia.org/w/index.php?curid=18034500>

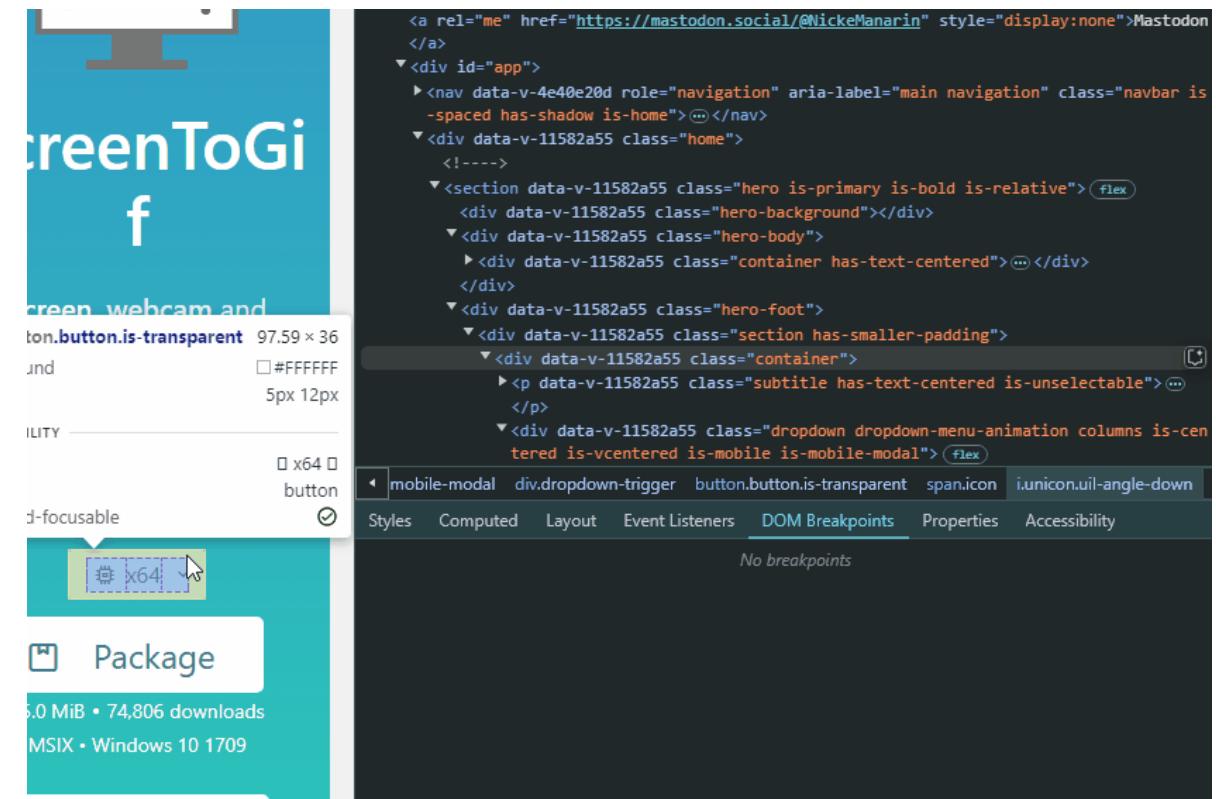
- Nos permite también seleccionar elemento por valores de sus atributos, ejemplo:
Tag `[@attribute=' value']`
- Acceder a los elementos de forma relativa (rutas)

DOM y XPath



By Birger Eriksson - Own work, CC BY-SA 3.0,
<https://commons.wikimedia.org/w/index.php?curid=18034500>

- Podemos sacarlo “a mano” o...



WebDriver: Esperar por un elemento

- **Flaky test:** Una prueba que pasaba en local o remoto, sin ninguna modificación en el código, en ocasiones falla causas:
 - No se ha cargado la página
 - La operación ha tardado más de lo habitual
 - No aparecen los elementos que se buscan
 - Un elemento que se esperaba que fuese clickable no lo es
 - ...

“SOLUCIÓN” : ESPERAR

WebDriver: Esperar por un elemento

- Selenium provee de la clase WebDriverWait, permite esperar por un determinado estado:

- Implícitamente (similar a hacer un sleep)

```
driver.manage().timeouts().implicitlyWait(duration);
```

- Explícitamente (espera condicional)

- Debemos instanciar un objeto de tipo WebDriverWait:

```
WebDriverWait waiter = new WebDriverWait(driver,duration);
```

- Se hará una espera hasta que se cumpla una condición:

```
waiter.until(CONDICION)
```

DISCUSION: ¿Solucionado?

Espera explícita: Condiciones

- Selenium nos aporta una serie de Condiciones de espera muy útiles, ejemplos:
 - ExpectedConditions.visibilityOfElementLocated(By)
 - ExpectedConditions.number Of ElementsToBeMoreThan(By, Int)
 - ExpectedConditions.elementToBeClickable(By)
 - ExpectedConditions.invisibilityOf/AllElements(WebElement/List<WebElements>)
 -

¡¡Son pruebas!! No olvidarse de los assertos

■ Ejemplos:

- Que el número de elementos es adecuado `assertEquals(x,y,"Mensaje")`
- Comprobar que el elemento tiene el valor adecuado (e.g una etiqueta)
- Comprobar que se muestra un mensaje.
- ...

Pruebas: Preparación y disposición

- Hay parte del código que es común a todas ellas:
 - Instanciar el WebDriver
 - Cargar datos de prueba
 - Guardar algún dato que interese de la ejecución
 - Esperar los distintos recursos.
 - ...
- ¡NO REPETIR ESTE CÓDIGO EN TODAS ELLAS!:

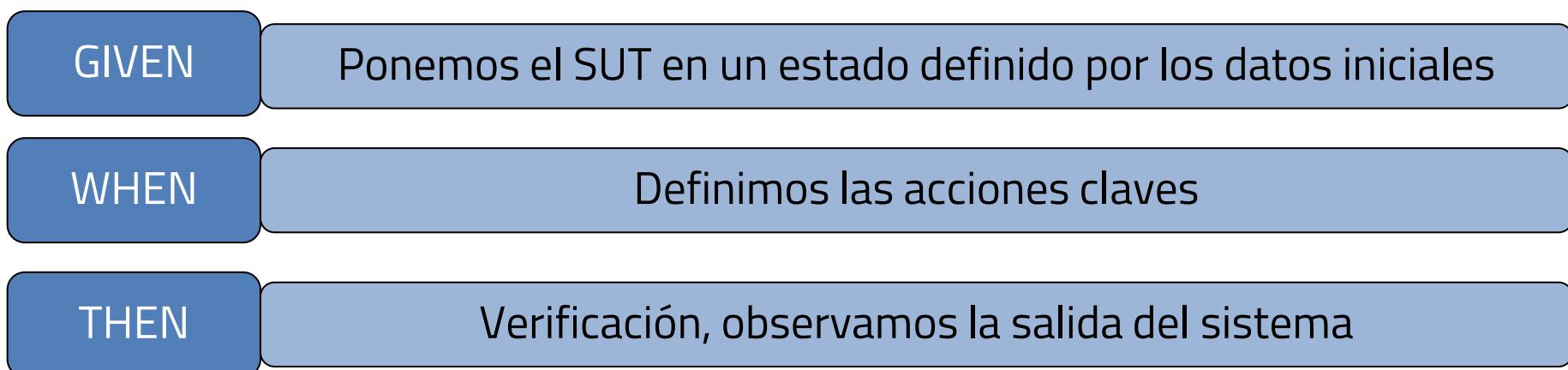
CONFIGURAR EL SET-UP Y TEAR-DOWN DE LA PRUEBA

Test Set-up y Test Tear-down

- Para que un método se ejecute antes de una prueba o clase disponemos de diferentes anotaciones de JUnit:
 - @BeforeEach: el método se ejecuta antes de cada test.
 - @AfterEach: el método se ejecuta después de cada test.
 - @BeforeAll: el método se ejecuta antes de todos los test.
 - @AfterAll: el método se ejecuta después de todos los test
- Habitual: crear métodos setUp(), teardown(), init()... con estas anotaciones y ese código a ejecutar

Pruebas de Aceptación: Cucumber

- Es un framework orientado a BDD (SSII) que nos permite elaborar pruebas a partir de los criterios de aceptación.
- BASE: estructura GIVEN-WHEN-THEN



Pruebas de Aceptación: Cucumber

- Cucumber proporciona un lenguaje de dominio propio para especificar estos pasos: GHERKIN
- Se especifican en funcionalidades (features) con uno o varios escenarios a probar:

FEATURE: El profesor puede calificar a sus alumnos

SCENARIO: Calificar a un solo alumno

GIVEN Alumno en calificaciones del campus virtual.

WHEN Seleccionamos el alumno, introducimos la nota y guardamos

THEN La calificación aparece en formato numérico y textual

- Se especifican en uno o varios ficheros ".feature"

Pruebas de Aceptación: Cucumber

- Por medio de anotaciones, unimos (“pegamos”) esos pasos del escenario en GHERKIN a métodos-código en Selenium que los materialicen:

```
@Given("Alumno en calificaciones del campus virtual")
public void indexCampus() {
    driver.get(" https://www.campusvirtual.uniovi.es/course/1 ");
    waiter.until(ExpectedConditions.NombreAlumno));
}
```

- Estos se encontraran en uno o varios ficheros .java, habitualmente [Nombre]Steps.java

Set-up/Tear-down en Cucumber

- Cucumber provee de los denominados Hooks: bloques de código en determinados puntos del ciclo de vida. Hook → Anotación, ejemplos:

@Before
SCEN 1
@After
@Before
SCEN 2
@After

@Before(order=1)
@Before(order=2)
SCENARIO 1
@After(order=1)
@After(order=2)



Pruebas de Aceptación: Cucumber

- Finalmente instanciamos y configuramos el runner de Cucumber y ejecutamos (desde el IDE o empleando el comando mvn test)

```
@Suite
@IncludeEngines("cucumber")
@SelectPackages("es.tew")
@ConfigurationParameter(key = GLUE_PROPERTY_NAME, value = "es.tew.testcases")
@ConfigurationParameter(key = PLUGIN_PROPERTY_NAME, value = "pretty,
html:target/cucumber-report/cucumber.html")
public class NavigationTests {

}
```

Reporte de Pruebas

- Genera un reporte HTML con la duración, cuantos test han pasado así como otra información del entorno y plataforma.

1 PASSED

100 % passed 1 executed	13 hours ago last run	0,08 seconds duration
 Windows 11	 OpenJDK 64-Bit Server VM 22.0.1+8-FR	 cucumber-jvm 7.20.1

Search with text or @tags
You can search with plain text or [Cucumber Tag Expressions](#) to filter the output

classpath:es/tew/testcases/navigation.feature

Feature: Navegar utilizando la barra de navegación superior

Scenario: Mostrar el listado de alumno

- Given la web inicial de la aplicación
- When Se selecciona la opción de listado en la barra superior
- Then La vista muestra una tabla con las columnas idUser, nombre, apellidos e email.

Referencias

- Documentación de Selenium - <https://www.selenium.dev/documentation/>
- Documentación Cucumber - <https://cucumber.io/docs/cucumber/>
- Documentación Junit 5 : <https://junit.org/junit5/docs/current/user-guide/>
- ISO/IEC/IEEE. (2022). ISO/IEC/IEEE International Standard - Software and systems engineering --Software testing --Part 1: General concepts. *ISO/IEC/IEEE 29119-1:2022(E)*, 1–60. <https://doi.org/10.1109/IEEESTD.2022.9698145>
- ISO/IEC/IEEE. (2017). 24765-2017 - ISO/IEC/IEEE International Standard - Systems and software engineering--Vocabulary. ieee, 2017(IEEE), 1–49. <https://ieeexplore.ieee.org/document/7907158>



Tecnologías Web: Pruebas de Web