Selenium United™ Certified Selenium Engineer (SeU-CSE)
Examen de muestra - 10 preguntas

Liberado

Programa de estudios Versión 2018

Selenium United

© SeU – Selenium United

Copyright © 2018 Selenium United (en adelante denominado SeU). Todos los derechos reservados.

Finalidad de este documento

Este documento contiene un examen de prueba completo para SeU Certified Selenium Engineer (CSE) en el idioma español.

Los ejemplos de preguntas, conjuntos de respuestas y justificaciones asociadas en este documento han sido creados por un equipo de expertos en la materia y escritores de preguntas experimentados con el objetivo de ayudar a las personas que están planeando tomar el examen de Certified Selenium Engineer (CSE).

Ninguna de estas preguntas se utilizará en el examen oficial SeU Certified Selenium Engineer (CSE), pero están escritas con el mismo nivel de dificultad que en el examen oficial de certificación.

Instrucciones

Los conjuntos de preguntas y respuestas están organizados de la siguiente manera:

- Objetivos de aprendizaje (OA) / Capítulos
- Pregunta - incluyendo cualquier escenario seguido por el origen de la pregunta
- Grupo de Respuestas

Información general sobre el examen de prueba

- Número de preguntas: 10
- Número de puntos: 1 por pregunta
- Por favor, seleccione sólo una respuesta por pregunta.

Lista de los capítulos

- Capítulo 1 - Automatización de la interfaz de usuario web
- Capítulo 2 - Introducción Selenium
- Capítulo 3 - Automatización de la interfaz de usuario web con Selenium
- Capítulo 4 - Más allá de las simples construcciones de códigos Selenium
- Capítulo 5 - Elaboración de un marco básico

Lista principal de los Objetivos de Aprendizaje (OA) para la certificación SeU CSE:

OA1	Entender la importancia de la cobertura del navegador y distinguir varias opciones para probar la interfaz de usuario de las aplicaciones web. (K2)
OA2	Entender la relación de la interfaz web con el HTML subyacente y JavaScript con DOM Inspection. (K2)
OA3	Recordar la historia de Selenium y varias herramientas en su suite, junto con su propósito. (K1)
OA4	Entender la arquitectura de Selenium en términos de enlaces de lenguaje (language bindings), protocolos de comunicación y controladores (drivers). (K2)
OA5	Reconocer el propósito y la API de la automatización de la interfaz de usuario web a nivel de navegador, página y elemento. (K2)
OA6	Entender y explicar el propósito de las anotaciones, accesorios y afirmaciones de JUnit. (K3)
OA7	Aplicar diferentes estrategias de identificación para los elementos de la interfaz de usuario. (K3)
OA8	Aplicar diferentes estrategias de investigación e interacción para los elementos de la interfaz de usuario. (K3)
OA9	Aplicar varias construcciones de automatización de Selenium más profundas, que establezcan las bases para una automatización más avanzada. (K3)
OA10	Automatice los escenarios de usuario final (end to end user scenarios) utilizando buenas prácticas de codificación y principios orientados a objetos para colocar el código Selenium en diferentes clases y métodos. (K3)
OA11	Solucionar problemas y describir el alcance de las mejoras para la implementación de la automatización que se muestran en ejemplos cortos de código. (K3)

© SeU – Selenium United

Pregunta 1

(La respuesta correcta vale 1 punto)

¿Cuál de las siguientes propiedades de estilo puede utilizarse para determinar si un elemento es visible?

- ☐ (a) `view`
- ☐ (b) `visible`
- ☐ (c) `display`
- ☐ (d) `Hide`

Pregunta 2

(La respuesta correcta vale 1 punto)

¿Cuál de las siguientes afirmaciones es **VERDADERA** sobre Selenium?

- ☐ (a) Selenium Grid se utiliza para desarrollar código de automatización de navegador basado en Selenese.
- ☐ (b) Selenium necesita ejecutables específicos de navegador que soporten el protocolo de cableado JSON estándar.
- ☐ (c) Los controladores de Selenium utilizan el protocolo de cable JSON para comunicarse con los navegadores.
- ☐ (d) Selenium soporta sólo lenguajes de programación orientados a objetos para desarrollar sus enlaces con clientes.

Pregunta 3

(La respuesta correcta vale 1 punto)

¿Cuál de las siguientes afirmaciones es **FALSA**? (Suponga que el controlador (`driver`) es un objeto `WebDriver`)

- ☐ (a) `By.className` soporta clases compuestas.
- ☐ (b) No se puede usar la funcion `ends-with` de Xpath para identificar elementos porque no es soportado por la mayoría de los navegadores.
- ☐ (c) `driver.switchToParentFrame()` cambia el contexto de búsqueda de un marco hijo (child frame) al marco padre (parent frame).
- ☐ (d) `HtmlUnitDriver` no necesita ningún controlador(driver) ejecutable.

© SeU – Selenium United

Pregunta 4

(La respuesta correcta vale 1 punto)

Dado el siguiente fragmento de formulario HTML:

```
<form action='abc'>
  <label>Type of vehicle</label>
  <select id = "vehicle_type">
     <option value = "2-wheeler">2 Wheeler</option>
     <option value = "3-wheeler">3 Wheeler</option>
     <option value = "4-wheeler">4 Wheeler</option>
  </select>
  <input id='B' type='submit'>
</form>
```

¿qué haría el siguiente código Java? (Suponga que el controlador `driver` es un objeto `WebDriver`)

```
driver.findElement(By.id("vehicle_type")).sendKeys("3-wheeler");
```

- ☐ (a) El tipo de vehículo se selecciona como valor de “3 Ruedas” en la lista desplegable.
- ☐ (b) El tipo de vehículo no ha cambiado ya que `sendKeys` no funciona con la etiqueta `select`.
- ☐ (c) El tipo de vehículo no cambia, ya que, en este caso, `sendKeys` espera que el texto sea visible.
- ☐ (d) Selenium lanza `NoSuchElementException`.

© SeU – Selenium United

Pregunta 5

(La respuesta correcta vale 1 punto)

En una aplicación dada, la página de inicio incluye un gran número de etiquetas `<script>` para archivos JavaScript externos de gran tamaño. Por lo tanto, dependiendo de las condiciones de la red, el navegador pasa de 45 a 90 segundos cargando y renderizando (rendering) la página de inicio. ¿Cómo implementaría la sincronización para manejar esta situación?

- ☐ (a) Implementará una espera estática de 90 segundos.
- ☐ (b) Usará una espera implícita de 90 segundos.
- ☐ (c) Implementará un evento sistemático basado en el ciclo `while`.
- ☐ (d) Usará una espera explícita de 90 segundos.

Pregunta 6

(La respuesta correcta vale 1 punto)

¿Cuál de las siguientes afirmaciones es **VERDADERA** cuando una alerta está abierta en un navegador? (Suponga que el controlador `driver` es un objeto `WebDriver`)

I. `driver.findElements` lanza una excepción
II. `driver.close` lanza una excepción
III. `driver.close` cierra la ventana actual.
IV. `driver.findElements` regresa una lista vacía.

- ☐ (a) I, II.
- ☐ (b) I, III.
- ☐ (c) II, III.
- ☐ (d) III, IV.

© SeU – Selenium United

Pregunta 7

(La respuesta correcta vale 1 punto)

¿Cuál sería el resultado del siguiente fragmento de código Java para Selenium? Suponga que:

- `driver` es un objeto `WebDriver`.
- `wait` es un objeto `WebDriverWait`.
- `menu` es un objeto del menú principal de `WebElement`.
- `subMenu` es un objeto del submenú `WebElement` que aparece después de pasar el cursor por el menú principal (parent menu).
- `editName` es un campo de texto (text field) con el id "name" en una página que aparece cuando se hace clic en el submenú arriba mencionado.

```
Actions actionBuilder = new Actions(driver);
actionBuilder.moveToElement(menu).click(subMenu).build();
WebElement editName = wait.until(
     ExpectedConditions.presenceOfElementLocated(By.id("name"))
);
editName.sendKeys("Updated");
```

- ☐ (a) Entra "Updated" en el campo `editName`
- ☐ (b) Lanza `NoSuchElementException`
- ☐ (c) Lanza `TimeoutException`
- ☐ (d) Falla al hacer clic en el `subMenu`

© SeU – Selenium United

Pregunta 8

(La respuesta correcta vale 1 punto)

De las siguientes afirmaciones, ¿cuáles representan la refactorización de código **PREFERIDO** de la siguiente clase básica de `WebAutomator` para colectar (wrapping) Selenium `WebDriver`?

```
public class WebAutomator{
   WebDriver driver;
   public WebAutomator(WebDriver wd){
      driver = wd;
      WebDriverWait waiter = new WebDriverWait(driver, 15);
   }
   public void click(By by) throws Exception{
      WebDriverWait waiter = new WebDriverWait(driver, 15);
      waiter.until(
          ExpectedConditions.elementToBeClickable(by)
       ).click();
   }
}
```

I. La variable `waiter` debe ser declarada como una variable de instancia.
II. La variable `waiter` debe ser declarada como una variable estática `static`.
III. A la variable `waiter` se le debe asignar un valor en constructor.
IV. A la variable `waiter` se le debe asignar un valor en el método `click`.

☐ (a) I, III.

☐ (b) II, IV.

☐ (c) I, IV.

☐ (d) II, III.

© SeU – Selenium United

Pregunta 9

(La respuesta correcta vale 1 punto)

¿Cuál de las siguientes estrategias de localización se utiliza para identificar el elemento padre a partir del elemento hijo?

- ☐ (a) Nombre de la clase.
- ☐ (b) Selector de CSS.
- ☐ (c) XPath.
- ☐ (d) Hallazgo de elemento anidado (Nested Element finding).

Pregunta 10

(La respuesta correcta vale 1 punto)

John y Carter quieren implementar una estrategia de instalación fija de prueba en una clase de prueba de JUnit5. ¿Cuáles de las siguientes opciones sobre las estrategias de instalación fija de prueba son **CORRECTAS** si el inicio de sesión es un requisito previo y el cierre de sesión es una actividad de limpieza?

I. El método `BeforeAll` inicia el navegador; el método `AfterAll` abandona el navegador.
II. El método `BeforeAll` inicia el navegador y realiza el inicio de sesión; el método `AfterAll` realiza el cierre de sesión y cierra el navegador.
III. El constructor de la clase lanza el navegador; el método `BeforeAll` realiza el inicio de sesión; el método `AfterAll` realiza el cierre de sesión y abandona el navegador.
IV. `BeforeEach` inicie el navegador y realice el inicio de sesión; `AfterEach` realice el cierre de sesión y el cierre de sesión del navegador.

- ☐ (a) I, III.
- ☐ (b) II, III.
- ☐ (c) II, IV.
- ☐ (d) I, IV.

© SeU – Selenium United

Clave de respuestas

Pregunta 1: La respuesta C es correcta

- Los elementos web tienen propiedades múltiples que determinan su comportamiento. La propiedad que determina la visibilidad de un elemento es su propiedad 'display'.

Pregunta 2: La respuesta B es correcta

- La Opción A está incorrecta porque el propósito principal de Selenium Grid es permitir la ejecución paralela de la automatización del navegador. Selenese es el DSL soportado por Selenium IDE.
- La opción B es correcta.
- La opción C es incorrecta porque los ejecutables del controlador (driver) utilizan la API nativa específica del navegador para comunicarse con los navegadores.
- La opción D es incorrecta porque Selenium WebDriver no impone ninguna restricción al tipo de lenguaje de programación que se va a utilizar.

Pregunta 3: La respuesta A es correcta

- La opción A es falsa porque `By.className` no soporta clases compuestas. Otras opciones presentan afirmaciones correctas.

Pregunta 4: La respuesta C es correcta

- Si se implementa una lista desplegable en HTML usando su etiqueta `<select>` estándar, se puede seleccionar una de sus opciones usando el método `sendKeys` de `WebElement`. Para ello, se debe utilizar el texto visible correspondiente a la opción de destino.

Pregunta 5: La respuesta D es correcta

- Las esperas explícitas son el mecanismo de espera sugerido.

Pregunta 6: La respuesta B es correcta

- En este caso el método `driver.findElements` lanzaría una excepción, sin embargo `driver.close` cerraría la ventana como se esperaba.

Pregunta 7: La respuesta C es correcta

- Aquí falta la llamada `perform()` para la acción compuesta, por lo que no se ejecuta la acción compuesta. Debido a esto, la página esperada no se cargaría. Esto haría que la espera explícita no fuera capaz de localizar el elemento durante toda la duración de la espera y lanzar una `TimeoutException`

© SeU – Selenium United

Pregunta 8: La respuesta A es correcta

- Dado el diseño de esta clase, la variable `waiter` sería necesaria en múltiples métodos cuando se añadan más métodos. Por lo tanto, debe convertirse en una variable de instancia e iniciarse una vez en el constructor.

Pregunta 9: La respuesta C es correcta

- Sólo una estrategia de localización en Selenium permite retroceder en el DOM fuera de las opciones proporcionadas - XPath (usando .. o /ancestor)

Pregunta 10: La respuesta C es correcta

- Los dispositivos de fijación de Junit se utilizan normalmente en los pares correspondientes para la configuración y limpieza de estado correctas. En este caso, las declaraciones II y IV muestran la elección correcta de las combinaciones de elementos de fijación de Junit.

© SeU – Selenium United