

? java - WebDriver: verifique si existe un elemento

Esta pregunta ya tiene una respuesta aquí:

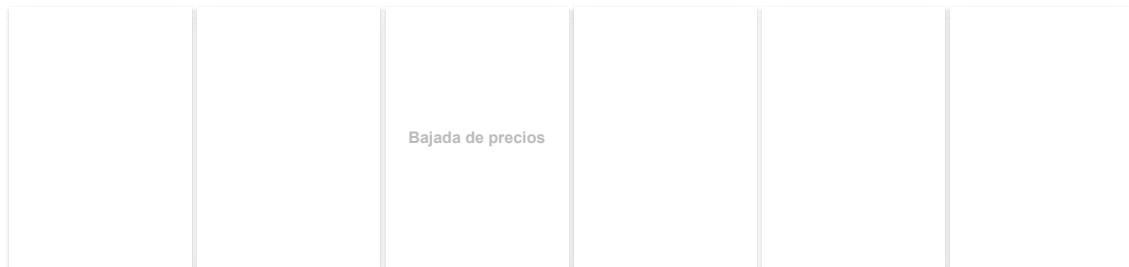
- Selenium WebDriver - Prueba si el elemento está presente (79f0e2) 19 respuestas

¿Cómo verificar si existe un elemento con el controlador web?

¿Usar una captura de prueba es la única forma posible?

```
boolean present;  
try {  
    driver.findElement(By.id("logoutLink"));  
    present = true;  
} catch (NoSuchElementException e) {  
    present = false;  
}
```

i Answers



✓ Alternativamente, podrías hacer:

```
driver.findElements( By.id("...") ).size() != 0
```

Lo que ahorra la desagradable prueba / captura

✓ Estoy de acuerdo con la respuesta de Mike, pero hay una espera implícita de 3 segundos si no se encuentran elementos que puedan activarse / desactivarse, lo cual es útil si se realiza mucho esta acción:

```
driver.manage().timeouts().implicitlyWait(0, TimeUnit.MILLISECONDS);  
boolean exists = driver.findElements( By.id("...") ).size() != 0  
driver.manage().timeouts().implicitlyWait(3, TimeUnit.SECONDS);
```

Poner eso en un método de utilidad debería mejorar el rendimiento si está ejecutando muchas pruebas

✓ Como decía el comentario, esto está en C # no en Java, pero la idea es la misma. Investigué este problema de forma exhaustiva y, en última instancia, el problema es que FindElement siempre devuelve una excepción cuando el elemento no existe. No hay una opción sobrecargada que te permita obtener un valor nulo o cualquier otra cosa. He aquí por qué prefiero esta solución sobre otras.

1. Devolver una lista de elementos y luego verificar si el tamaño de la lista es 0 funciona pero pierde la funcionalidad de esa manera. No puede hacer un .click () en una colección de enlaces, incluso si el tamaño de la colección es 1.
2. Podría afirmar que el elemento existe, pero a menudo eso detiene su prueba. En algunos casos, tengo un enlace adicional para hacer clic dependiendo de cómo llegué a esa página y quiero hacer clic si existe o seguir adelante de lo contrario.
3. Solo es lento si no configura el **controlador de tiempo de espera**. **Administrar ()**. **Tiempos de espera ()**. **ImplicitlyWait (TimeSpan.FromSeconds (0))**;
4. En realidad es muy simple y elegante una vez que se crea el método. Al usar **FindElementSafe** en lugar de **FindElement** , no "veo" el feo bloque try / catch y puedo usar un método **Exists** simple. Eso se vería así:

```

IWebElement myLink = driver.FindElementSafe(By.Id("myId"));
if (myLink.Exists)
{
    myLink.Click();
}

```

Así es como extiende IWebElement & IWebDriver

IWebDriver.FindElementSafe

```

/// <summary>
/// Same as FindElement only returns null when not found instead of an exception.
/// </summary>
/// <param name="driver">current browser instance</param>
/// <param name="by">The search string for finding element</param>
/// <returns>Returns element or null if not found</returns>
public static IWebElement FindElementSafe(this IWebDriver driver, By by)
{
    try
    {
        return driver.FindElement(by);
    }
    catch (NoSuchElementException)
    {
        return null;
    }
}

```

IWebElement.Exists

```

/// <summary>
/// Requires finding element by FindElementSafe(By).
/// Returns T/F depending on if element is defined or null.
/// </summary>
/// <param name="element">Current element</param>
/// <returns>Returns T/F depending on if element is defined or null.</returns>
public static bool Exists(this IWebElement element)
{
    if (element == null)
    { return false; }
    return true;
}

```

Puede usar el polimorfismo para modificar la instancia de la clase IWebDriver de FindElement, pero esa es una mala idea desde el punto de vista del mantenimiento.



puedes hacer una afirmación

mira el ejemplo

```

driver.asserts().assertElementFound("Page was not loaded",
By.xpath("//div[@id='actionsContainer']"), Constants.LOOKUP_TIMEOUT);

```

puedes usar esto esto es nativo:

```

public static void waitForElementToAppear(Driver driver, By selector, long timeOutInSeconds, String timeOutMessage) {
    try {
        WebDriverWait wait = new WebDriverWait(driver, timeOutInSeconds);
        wait.until(ExpectedConditions.visibilityOfElementLocated(selector));
    } catch (TimeoutException e) {
        throw new IllegalStateException(timeOutMessage);
    }
}

```



Esto funciona para mí todo el tiempo:

```

if(!driver.findElements(By.xpath("//*[@id='submit']")).isEmpty()){
    //THEN CLICK ON THE SUBMIT BUTTON
}else{
    //DO SOMETHING ELSE AS SUBMIT BUTTON IS NOT THERE
}

```



Extadí la implementación de Selenium WebDriver, en mi caso HtmlUnitDriver para exponer un método

```

public boolean isElementPresent(By by){}

```

Me gusta esto:

1. compruebe si la página se carga dentro de un período de tiempo de espera.
2. Una vez que se carga la página, reduzco el tiempo de espera implícitamente del WebDriver a algunos milisegundos, en mi caso 100 mills, probablemente también debería funcionar con 0 mills.
3. llame a findElements (By), WebDriver aunque no encuentre el elemento esperará solo la cantidad de tiempo desde arriba.
4. aumentar el tiempo de espera implícitamente para la carga futura de la página

Aquí está mi código:

```
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.htmlunit.HtmlUnitDriver;
import org.openqa.selenium.support.ui.ExpectedCondition;
import org.openqa.selenium.support.ui.WebDriverWait;

public class CustomHtmlUnitDriver extends HtmlUnitDriver {

    public static final long DEFAULT_TIMEOUT_SECONDS = 30;
    private long timeout = DEFAULT_TIMEOUT_SECONDS;

    public long getTimeout() {
        return timeout;
    }
    public void setTimeout(long timeout) {
        this.timeout = timeout;
    }

    public boolean isElementPresent(By by) {
        boolean isPresent = true;
        waitForLoad();
        //search for elements and check if list is empty
        if (this.findElements(by).isEmpty()) {
            isPresent = false;
        }
        //rise back implicitly wait time
        this.manage().timeouts().implicitlyWait(timeout, TimeUnit.SECONDS);
        return isPresent;
    }

    public void waitForLoad() {
        ExpectedCondition<Boolean> pageLoadCondition = new ExpectedCondition<Boolean>() {
            public Boolean apply(WebDriver wd) {
                //this will tel if page is loaded
                return "complete".equals(((JavascriptExecutor) wd).executeScript("return document.readyState"));
            }
        };
        WebDriverWait wait = new WebDriverWait(this, timeout);
        //wait for page complete
        wait.until(pageLoadCondition);
        //lower implicitly wait time
        this.manage().timeouts().implicitlyWait(100, TimeUnit.MILLISECONDS);
    }
}
```

Uso:

```
CustomHtmlUnitDriver wd = new CustomHtmlUnitDriver();
wd.get("http://example.org");
if (wd.isElementPresent(By.id("Accept"))) {
    wd.findElement(By.id("Accept")).click();
} else {
    System.out.println("Accept button not found on page");
}
```

☒ Escribe el siguiente método usando Java:

```
protected boolean isElementPresent(By by){
    try{
        driver.findElement(by);
        return true;
    }
    catch(NoSuchElementException e){
        return false;
    }
}
```

Llame al método anterior durante la aserción.

- ☒

```
String link = driver.findElement(By.linkText(linkText)).getAttribute("href")
```


Esto le dará el enlace al que apunta el elemento.
- ☒

Con la versión 2.21.0 de selenium-java.jar puede hacerlo;

```
driver.findElement(By.id("...")).isDisplayed()
```
- ☒

Según tengo entendido, esta es la forma predeterminada de usar el controlador web.





JH-S9100 Wireless Bluetooth TWS Auriculares
duales Touch Control Sweatproof con caja de...


17,74 €

Anuncio [GeekBuying.com](#)

Saber más

Links

- ¿Cómo puedo verificar si existe un archivo en Java? (1bb861)
- Selenium WebDriver ¿Cómo resolver la excepción de referencia de elemento obsoleto? (f6ad75)



Formación Online en Autómatas Programables. Recomendado por SIEMENS Industria,
Curso para programar autómatas Siemens y automatizar aplicaciones industriales.

Anuncio [SEAS Estudios Abiertos](#)

Visitar sitio

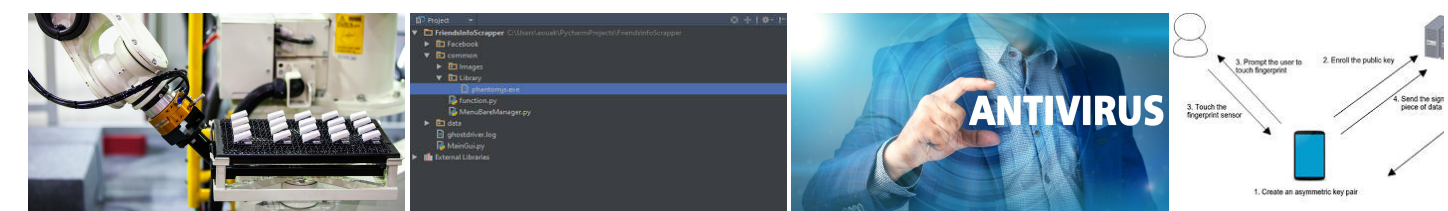
Tags

- java

testing

webdriver

selenium-webdriver



Curso de Autómatas

¿Puede Selenium Webdriver abrir las ventanas del navegador silenciosamente...

Antivirus Gratis 2018

javascript - Autentica huellas dactilares Co en el servidor - andro

Anuncio

SEAS Estudios Abiertos

i-harness.com

Anuncio

mejorantivirus10.com

i-harness.com



Pass Your JAVA Exams

¿Cuál es la diferencia entre varchar y varchar2 en Oracle? - sqldatatypes |...

c# - ¿Cuál es la diferencia entre el patrón de diseño MVC, MVP y MVVM en...

html - WebForms UnobtrusiveValidation requiere un...

Anuncio

prepaway.com

i-harness.com

i-harness.com

i-harness.com

Twittear

Búsqueda personalizada de Google