NOTICIAS

Noticias

31
MAR
2016

SELENIUM WEBDRIVER Y BUSCAR ELEMENTOS EN LA PÁGINA WEB

Creado por Roman Rodriguez /
Comentarios 0

La esencia de Selenium WebDriver es interactuar con los elementos de una página web.

Para eso, primero es necesario indicarle a selenium que busque el elemento en la página en la que esté posicionado (ver post anterior para navegación en Selenium WebDriver) e indicarle cómo buscar dicho elemento.
Nunca se podrá interactuar con un elemento sin identificarlo previamente.

ARGENTESTING 2017

El **23** y **24** de **Octubre** se llevó a cabo en la **UTN Sede Medrano**, la **2da Edición** del **Argentesting**, donde hubo Charlas y Talleres vinculados con el Software Testing. Las presentaciones de las Charlas y Talleres ya están publicadas, y muy pronto estaremos publicando las grabaciones de Charlas por el canal de Youtube. +info

Escribe algo...

Offline

La forma de indicarle a selenium cómo encontrar un elemento es a través del método (instrucción) find_element.

```
elemento = browser.find_element(:id, "valor-del
```

Vamos a descomponer la instrucción para entender que significa cada parte de la misma:

mada ***elemento*** vamos a ue devuelve la instrucción o objeto es de la clase ***Selenium::WebDriver::Element***

:id indica que debe buscar un elemento cuyo atributo id tenga el valor ***"valor-del-id"***

Pero ¿qué significa elemento, atributo, valor?

Para entender un poco de que estamos hablando, vamos a entender un poco lo que es un documento html.

AMIGOS

- Federico Toledo
- SoftQaNetwork
- Software, calidad y test
- Testing funcional

FAVORITOS

- HASTQB
- James Bach's Blog
- Lisa Crispin's Blog
- Practitest Blog

Haciendo click en dicha opción vamos a ver el documento html. Html es un lenguaje de marcado de hipertexto. En palabras sencillas (muy sencillas pero que nos sirven para el caso) se trata de un archivo donde toda la información de la página está definida con etiquetas. Los navegadores interpretan este documentos y "dibujan" en la pantalla el significado del mismo donde cada elemento se dibuja según los atributos/características definidas en esas etiquetas. Un documento html tiene muchísimos tags o etiquetas que le indican al navegador de que elementos, con que atributos y en que orden está compuesta la página web.

Las etiquetas (tags) está determinadas por < y > con una de apertura y una de cierre (por lo general).
Existen diversas etiquetas para distintos tipos de elementos donde hay una <etiqueta> de apertura y una </etiqueta> de cierre. Observar que la segunda posee una barra antes del nombre de la etiqueta y eso indica que se trata de la etiqueta de cierre. Por decir algunas:

<html></html>

<head></head>

<body><body>

<form></form>

<button></button>

Offline

de scripting, explicando las características principales del lenguaje de programación utilizado (Ruby), permitiendo confeccionar scripts de automatización robustos y altamente mantenibles, dos de los aspectos más importantes en la automatización de pruebas. **¿Quieres conocer más detalles?** clic AQUI

Como se puede ver en el último ejemplo, las etiquetas pueden ir anidadas siempre manteniendo un orden correcto de apertura y cierre.

En fin, lo importante aquí es que esas etiquetas, como absolutamente todas las etiquetas que pueden existir en un documento html pueden tener atributos (o no).
Los atributos son datos que pueden existir dentro de una etiqueta como por ejemplo un identificador "id", el atributo "name", el atributo "class" y muchos otros mas.

Por ejemplo, el siguiente campo de texto tiene muchos ***atributos:***

<input type ="text" class ="js-site-search-focus js-site-search-field is-clearable chromeless-input" data-hotkey ="s" name ="q" placeholder ="Search" aria-label ="Search this repository" data-global-scope-placeholder ="Search GitHub" data-repo-scope-placeholder ="Search" tabindex ="1" autocapitalize ="off">

Offline

```
browser.find_element(:name, "q")
```

Por atributo class:

```
browser.find_element(:class, "js-site-search-fo
```

Si tuviese un atributo id también se podría buscar aunque éste no es el caso.

Además de estos atributos, selenium ofrece otras características a indicar en el criterio de búsqueda (lo que está entre paréntesis del método find_element). El orden que se presenta a continuación va desde lo mas rápido en encontrar un elemento (más eficiente) a lo que mas tarda en encontrar un elemento (menos eficiente)

```
:id
:class_name o :class
:name
:link_text
:partial_link_text
:css
:xpath
```

Los tres primeros (id, class, name) son atributos que se ven en el código html. El criterio link_text significa el texto que tiene un link. En el caso de partial_link_text significa que también puede buscar por una porción del texto de un link. El texto de un link es lo que se ve efectivamente como un link en una página web

En el caso de css y xpath, se trata de rutas de acceso a los elementos dentro de la jerarquia de los mismos en el documento. En el caso de css ver ésta referencia y en el caso de xpath ver esta r

Offline

¿Cuál es la diferencia entre find_element y find_elements?

La diferencia está en que find_element buscará un solo elemento según el criterio de búsqueda indicado. Si en el documento html existen más de un elemento que coincide con dicha búsqueda, Selenium WebDriver nos devolverá el primero que encuentra en el documento.
En el caso de find_elements, Selenium WebDriver nos dará una lista de todos los elementos que coinciden con el criterio de búsqueda dado, generalmente se utilizan los criterios class y tag_name en el find_elements.

Esa lista que devuelve find_elements se trata de un tipo de dato llamado Array en Ruby (en otros lenguajes se maneja el mismo concepto).

En el **próximo post** vamos a conocer las operaciones que se pueden realizar con un elemento.

Publicaciones relacionadas

SELENIUM WEBDRIVER – PRIMEROS COMANDOS
SELENIUM WEBDRIVER Y RUBY, PASOS PARA LA INSTALACIÓN
SELENIUM WEBDRIVER, BREVE INTRODUCCIÓN

SOBRE EL AUTOR

Roman Rodriguez

Co fundador - Instructor Selenium WebDriver & Cucumber

Deja un comentario

Tu dirección de correo electrónico no será publicada. Los campos obligatorios están marcados con *

* Nombre

* Correo electrónico

Web

2EGLkz6Cjq9w *** Copy This Password ***

*** Type Or Paste Password Here ***

Comentario

Offline

```
title=""> <abbr title=""> <acronym title=""> <b>
<blockquote cite=""> <cite> <code> <del datetime=""> <em>
<i> <q cite=""> <s> <strike> <strong>
```

PUBLICAR COMENTARIO

SOBRE NOSOTROS

Gustavo Terrera y Roman Rodriguez, apasionados por el Software Testing, a través de este sitio web y sus respectivos canales, difunden y promueven las prácticas del testing con el objeto de hacer crecer la comunidad de software testers hispanoparlantes.

BUSCAR

Escribe algo...

ENERO 2018

L	M	X	J	V	S	D
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

« Dic

ETIQUETAS

AGILE
AGILE TESTING
ANDROID
ARGENTESTING
AUTOMATION
AUTOMATIZACIÓN
AUTOMATIZACIÓN DE PRUEBAS
BSTRIKER BUG
BUGS CALIDAD
CAPACITACIÓN
CUCUMBER
CURSO DEBATE
ENCUESTA

ENTRADAS RECIENTES

- Argentesting 2da Edición – Testeando el estadio aumentado
- Consideraciones para diseñar una estrategias de automatización de pruebas
- Asegurando y controlando la calidad del WordPress 4.9.1.
- Argentesting 2da Edición – Lo que aprendí de Rapid Software Testing con Michael

Offline

ISTQB AL JIRA

JOBS MOBILE

MOBILE TESTING

MÓVILES QA

REQUERIMIENTOS

RUBY SCRIPTING

SCRUM

SELENIUM

SELENIUM WEBDRIVER

TESTING

TESTLINK

TRABAJO

TUTORIAL UTEST

WEBSERVICES

YOUTUBE

tester – Grabación

ARCHIVOS

Archivos

Elegir mes

Offline