Ficha de ayuda: Búsqueda de elementos

Selenium WebDriver es un controlador de navegador. Puedes usarlo para implementar código que controle al navegador y para emular las acciones de usuario.

Selenium imita la mayoría de las acciones de usuario en el navegador:

- Hacer clic en un elemento: un botón, un botón de opción, una casilla de verificación, una lista desplegable.
- Rellenar campos de entrada.
- Navegar entre páginas: retroceder y avanzar, o usar una dirección URL.
- Navegar por la página: actualizar la página o desplazarse por ella.

Algunas acciones no pueden ser automatizadas:

- Realizar la autenticación en dos pasos (se requiere un dispositivo o una dirección de correo electrónico por separado para recibir un código de un solo uso).
- Ingresar CAPTCHA.
- Progreso de descarga de archivos.
- Acceder a ciertos sitios web, como Gmail (estos casos se describen en los Términos de servicio).

HTML, elementos y etiquetas

HTML

HTML (HyperText Markup Language, lenguaje de marcado de hipertexto) es un lenguaje de marcado que almacena datos para que tu navegador los lea, procese y muestre de una manera predefinida.

Elementos

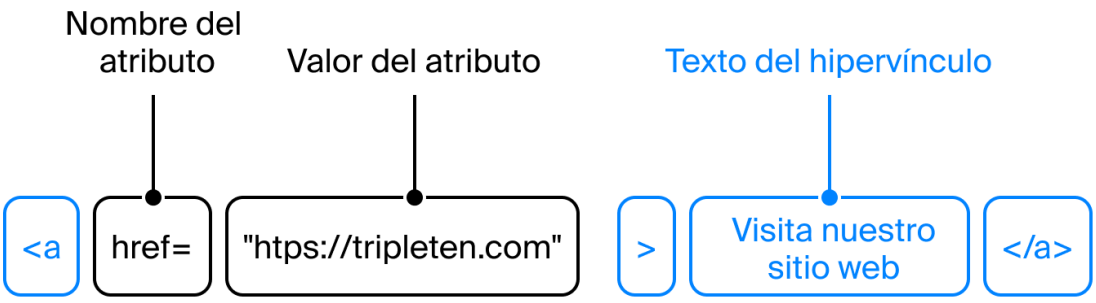
HTML se compone de elementos, y los elementos se componen de etiquetas.

El elemento, `h1` en nuestro ejemplo, le dice al navegador qué hacer con ese elemento. Los elementos `h1` son "encabezados de primer nivel", `h2` es un "encabezado de segundo nivel", y así sucesivamente.

Las etiquetas simplemente le dicen al navegador dónde comienza un elemento y dónde termina.

Atributos

Los elementos HTML pueden tener **atributos** para dar al navegador información adicional sobre un elemento. Un atributo está contenido en la etiqueta de apertura. Por ejemplo, un atributo `href` te permite agregar una dirección web a un fragmento de texto, creando un hipervínculo.



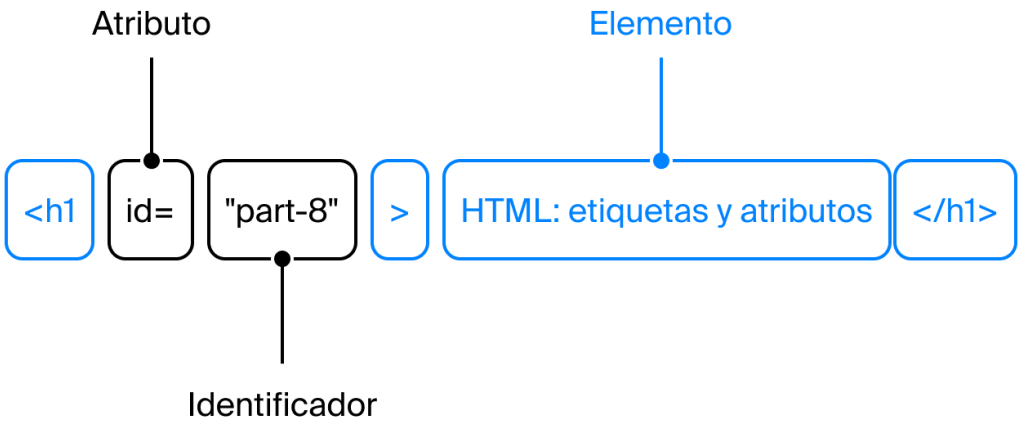
El atributo se agrega solo a la etiqueta de apertura. Si hay varios atributos, se separan con un espacio. No importa el orden, por lo que un mismo elemento se puede escribir

de diferentes formas:

```
<a href="http://info.cern.ch" target="_blank">Inicio del primer sitio web</a>
<!-- Lo mismo -->
<a target="_blank" href="http://info.cern.ch">Inicio del primer sitio web</a>
```

ID

Un elemento puede tener un **identificador**. Este es el valor del atributo `id`. El área de desarrollo le da al elemento un nombre único mediante el uso de un `id`, es decir, el valor de los identificadores no se repiten a diferencia de otros elementos.

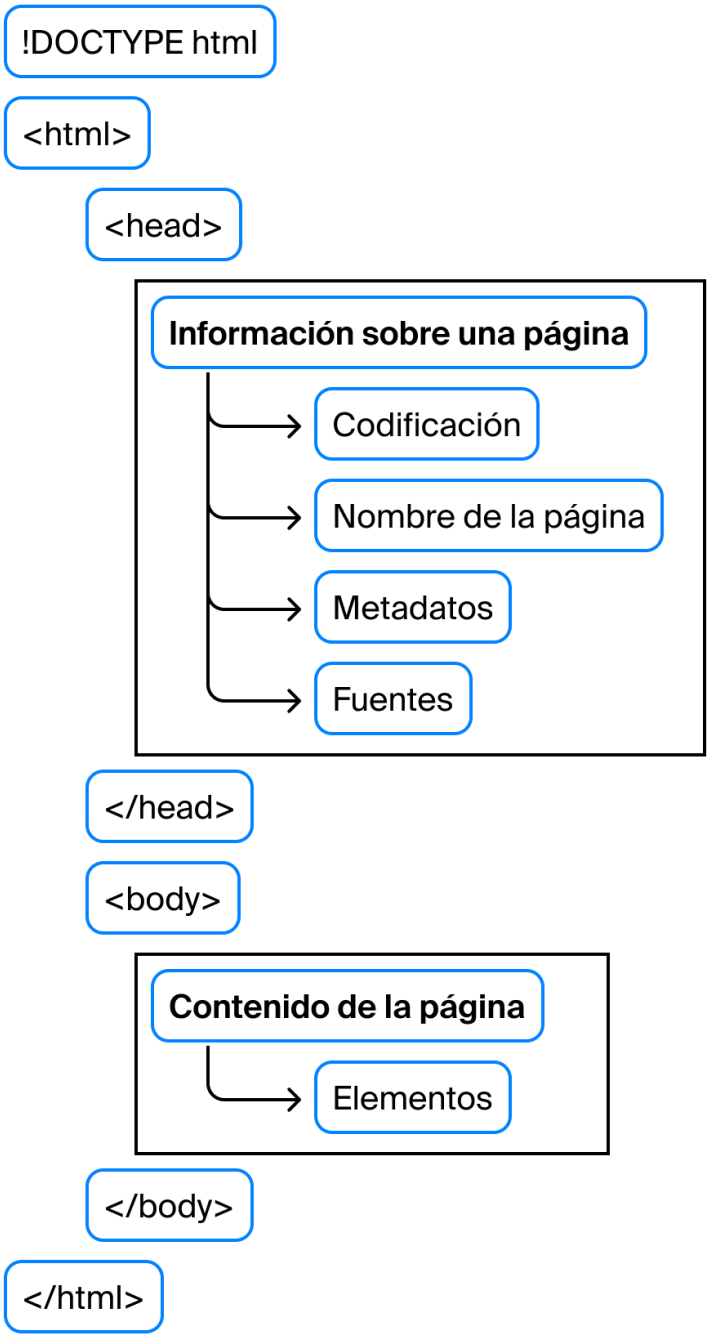


Aquí, `part-8` es el identificador del elemento `HTML: etiquetas y atributos`.

El `id` ayuda a los y las testers de automatización a encontrar un elemento en la página de manera rápida y sin ambigüedades.

El DOM

Todos los documentos HTML tienen una cierta estructura:



En HTML, podemos anidar un elemento dentro de otro. El elemento interior se llama **hijo**. El elemento en el que se anida el hijo se llama **padre**. Es importante tener en cuenta que podemos tener varios niveles de anidamiento de padres e hijos.

Por ejemplo, la etiqueta `head` contiene datos adicionales sobre la página:

- una descripción para motores de búsqueda y redes sociales;
- codificación necesaria para una visualización adecuada;
- idioma de la página.

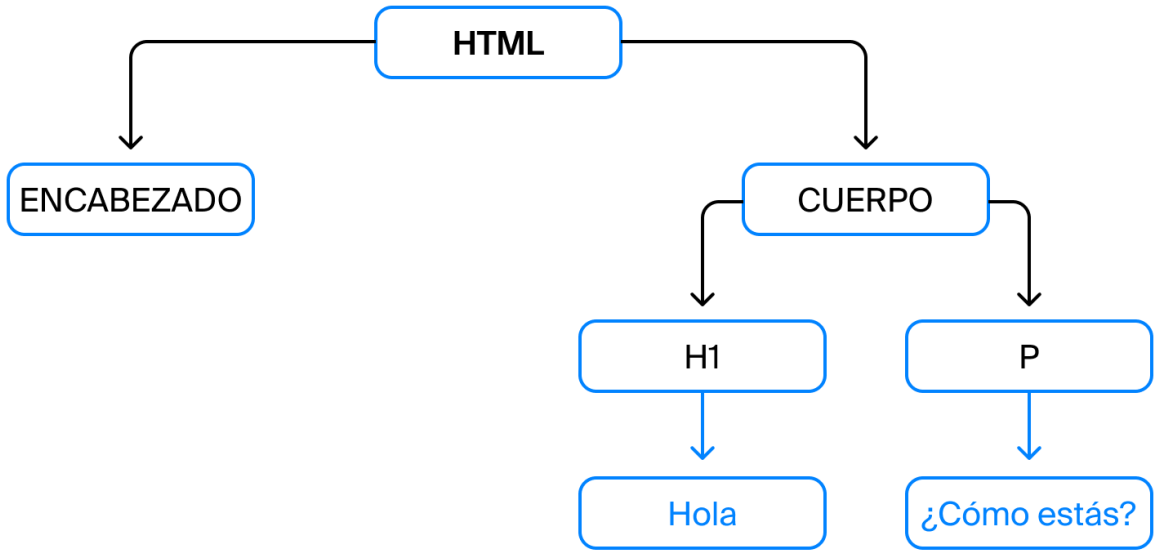
Mientras tanto, dentro de la etiqueta `body` se encuentra el principal contenido visible de la página: encabezados, texto, enlaces, lo que sea.

El DOM

Para trabajar con una página web, el navegador convierte el archivo HTML en un árbol DOM.

El **DOM** (Document Object Model, modelo de objetos del documento) de una página es un modelo formado por diferentes objetos. El árbol DOM es necesario para gestionar la estructura o los estilos de una página.

Así es como se ve un árbol DOM:



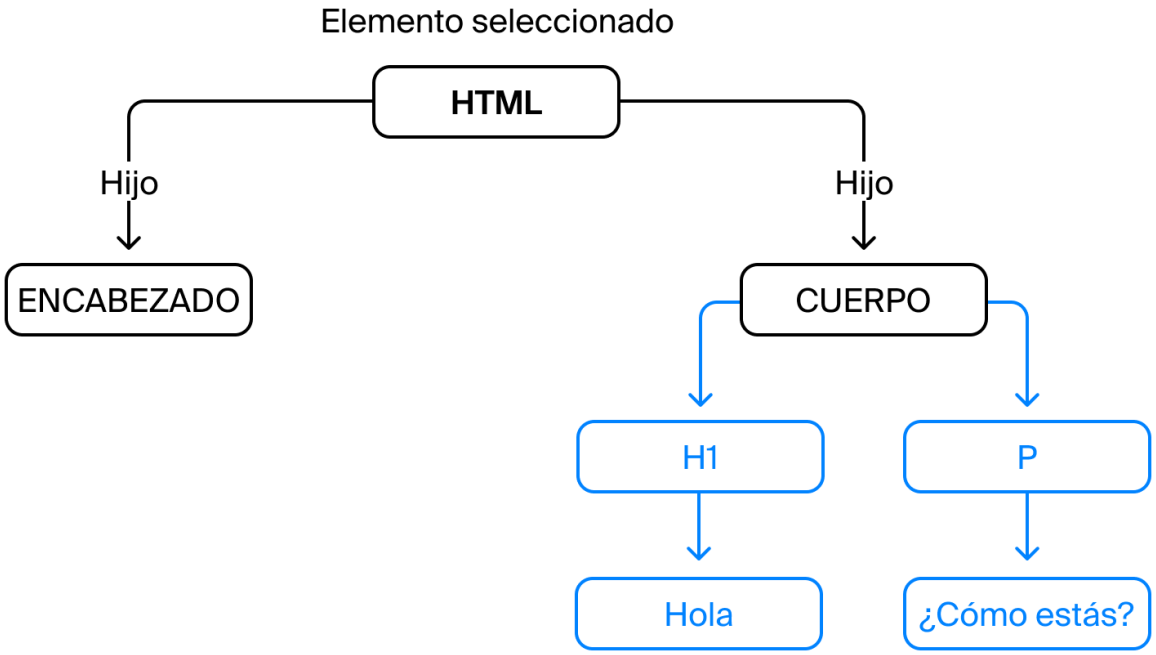

```
<!doctype html>
<html lang="eng">
  <head></head>
  <body>
```

```
    <h1>Hi</h1>
    <p style="color: red">¿Cómo estás?</p>
  </body>
</html>
```

La estructura del árbol DOM es la misma que el marcado del propio documento. Si las etiquetas están anidadas dentro de otras etiquetas, lo mismo sucede en el DOM: esos elementos se convierten en hijos en relación con otros elementos.

Aquí es donde debemos distinguir entre dos términos: hijos y descendientes.

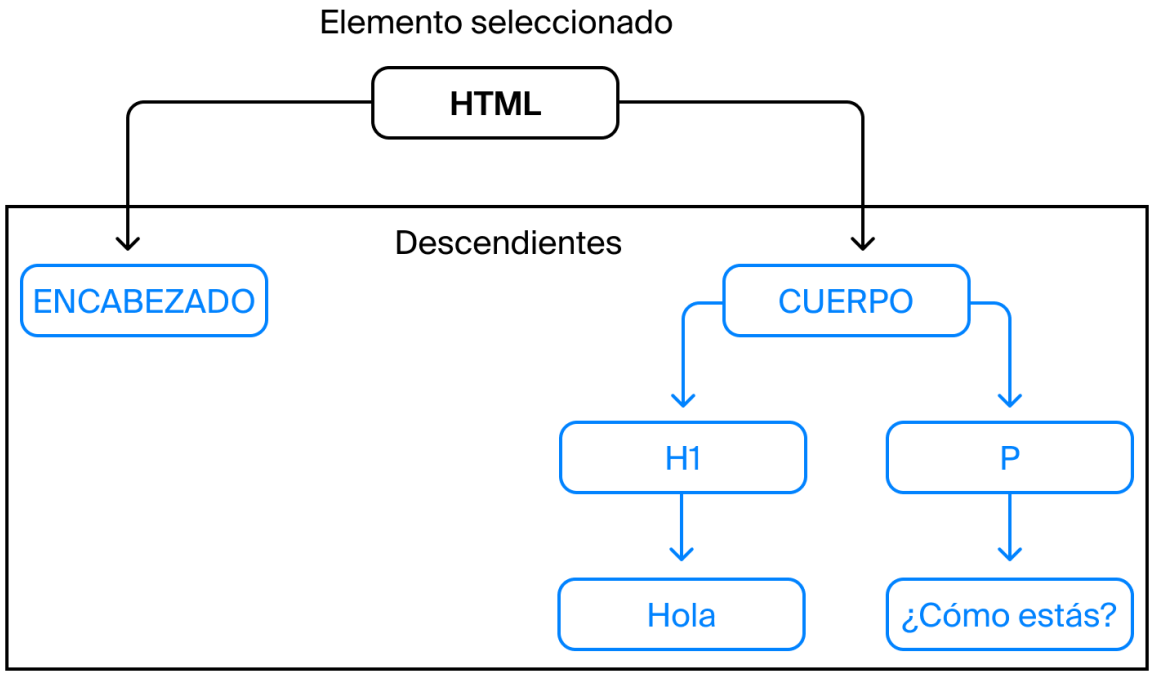
Los **hijos** son los elementos de un árbol DOM un nivel por debajo del elemento seleccionado. Del mismo modo, un elemento de un nivel por encima se denomina **padre**. Por ejemplo, `<head>` y `<body>` son hijos de `<html>`.



Los **descendientes** son todos los elementos que están debajo del elemento seleccionado, o sea, sus hijos, los hijos de sus hijos, etc.

Por el contrario, los **ancestros** son todos los elementos que se encuentran por encima del elemento seleccionado.

Por ejemplo, todos los elementos de la página son descendientes de `html`.



El árbol DOM consta de **nodos**. Aparecerá un nodo en cada uno de los siguientes casos:

- una etiqueta de apertura;
- texto;
- un comentario HTML.

Todos los elementos son nodos, pero no todos los nodos son elementos. Por ejemplo, un comentario HTML es un nodo, pero no es un elemento. Vamos a comparar:

- `<img>` y `<a></a>` son elementos.
- El texto `<a>dentro del enlace</a>` es un nodo de texto.

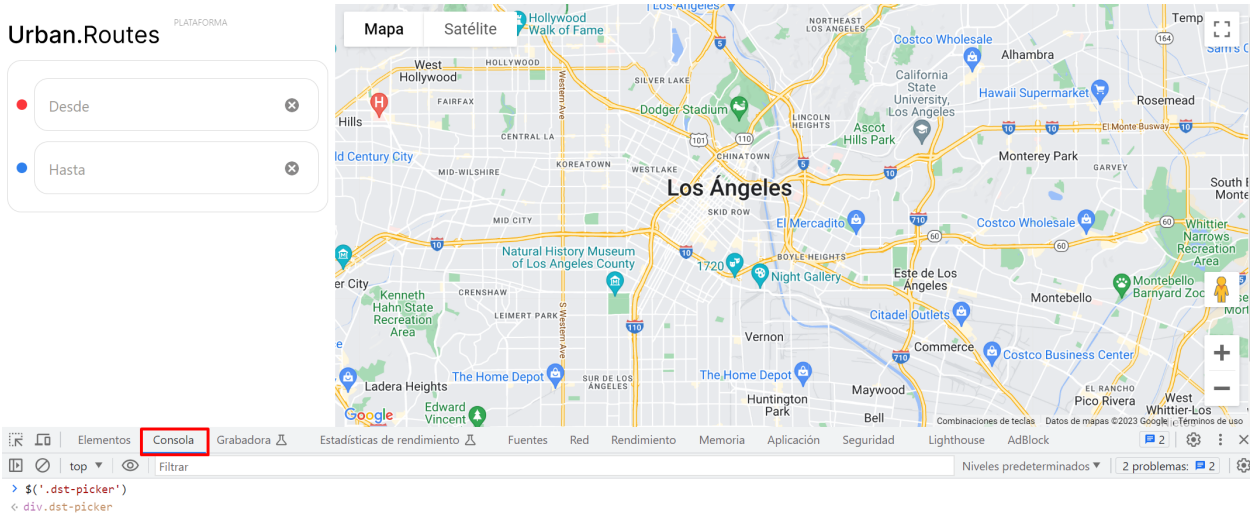
Selectores CSS

Los selectores (y los localizadores en general) son un tipo especial de solicitud de texto que ayuda a encontrar un elemento en una página web.

Primero, debemos enumerar los atributos en los que se basarán nuestros selectores CSS.

- El nombre de la clase asignada a un elemento: `<div class="example-class-name"></div>`. `example-class-name` es el nombre de clase.
- El ID de un elemento: `<div id="example-id"></div>`. `example-id` es el ID.
- Nombres de etiqueta del elemento: `<span></span>`. `span` es el nombre de etiqueta.
- Atributos del elemento: `<input type="text">`. `type="text"` es el atributo.
- Selectores de consulta CSS: `<div><span></span></div>`. `div span` es el selector de consulta. Indicando que un nodo `div` tiene como sucesor a un nodo `span`.

Para encontrar un selector, puedes utilizar la consola JavaScript. Está disponible en Chrome y en la mayoría de los navegadores modernos dentro de las DevTools en la pestaña “Consola” (Console). Por ejemplo, para encontrar un elemento con una clase llamada `dst-picker`, puedes ingresar `$('.dst-picker')` en la consola. Fíjate en el `.`, que es como indicamos que el texto que sigue es un nombre de clase.



Atributo	Ejemplo de atributo	Cómo encontrar un elemento	Ejemplo de búsqueda
El nombre de una clase	`<div class="example-class-name"></div>`	Prefijar el nombre de clase con un `.`	`$('.example-class-name')`
El ID de un elemento	`<div id="example-id"></div>`	Prefijar el nombre de ID con `#`	`$('#example-id')`

Atributo	Ejemplo de atributo	Cómo encontrar un elemento	Ejemplo de búsqueda
Nombre de la etiqueta del elemento	`<span></span>`	Escribir el nombre de la etiqueta	`$('span')`
Atributo del elemento	`<input type="text">`	Escribir el atributo entre corchetes	`$('[type="text"]')`

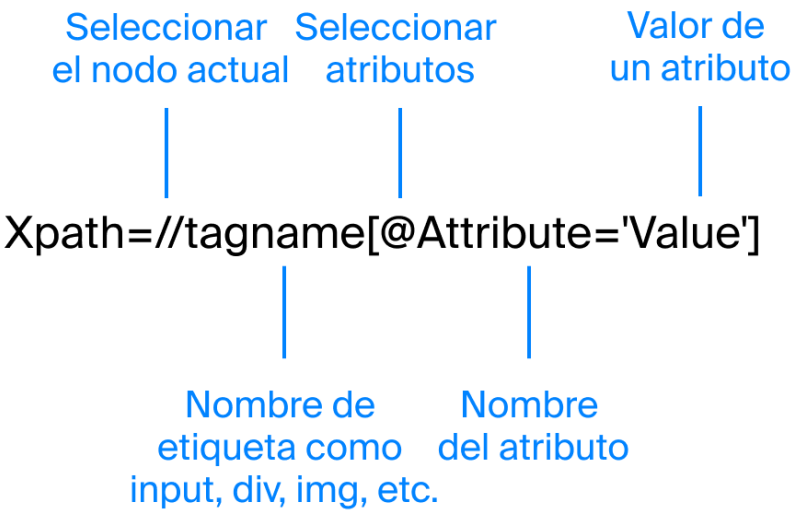
Localizadores XPath

Al igual que los selectores CSS, los localizadores XPath son una forma de localizar un elemento en una página web. Si bien los selectores CSS tienden a ser más rápidos y fáciles de aprender, XPath tiene una serie de características únicas, por lo que es útil comprender ambos.

XPath significa lenguaje de ruta XML (XML Path Language). Aunque lo usaremos para trabajar con archivos HTML, originalmente se creó para trabajar con documentos XML, como sugiere el nombre.
Por eso, funciona describiendo la localización de los elementos en relación con el árbol del documento.

Usaremos la consola DevTools nuevamente para probar nuestras consultas XPath con el comando integrado `$x('your_xpath')`.

XPath emplea una sintaxis similar a una ruta para seleccionar elementos en una página web. La ruta comienza con una barra inclinada `/` y es seguida por los nombres de los elementos que quieres seleccionar. Una consulta XPath está estructurada de la siguiente forma:



El contenido dentro de los corchetes `[]` se conoce como un predicado. Los predicados se utilizan para encontrar un nodo específico o un nodo que contiene un valor específico. También podemos usar el índice del elemento como predicado.

Hay varios símbolos que puedes utilizar en una consulta XPath.

Símbolo	Propósito
elementName	Selecciona todos los nodos con el nombre "elementName".
/	Selecciona desde el nodo raíz (no se recomienda su uso).
//	Selecciona nodos en el documento del nodo actual que coincidan con la selección sin importar dónde se encuentren.
.	Selecciona el nodo actual.
..	Selecciona el padre del nodo actual.
@	Selecciona los atributos.
*	Relaciona cualquier nodo del elemento.
@*	Relaciona cualquier nodo del atributo.

Para seleccionar todos los enlaces en una página web, puedes usar la siguiente expresión XPath: `//a`. El comando completo es `$x('//a')`. Esta expresión selecciona todos los elementos `a` (hipervínculo) de la página. La doble barra inclinada al comienzo de la expresión significa que la búsqueda debe comenzar en la raíz del documento y buscar todos los elementos descendientes de los elementos `a`.

Así es como se localizan diversos elementos:

1. Para un atributo, necesitamos usar el símbolo `@`. Por ejemplo, para localizar elementos con un tipo de atributo de texto, puedes usar la siguiente expresión: `//input[@type="text"]`.
2. Para todos los elementos con un atributo de clase igual a "input-container" (contenedor de entrada), usa: `//*[@class="input-container"]`.
3. Para todos los elementos img que están dentro de un elemento div con un atributo de clase "logo", usa: `//div[@class="logo"]//img`.
4. Para un elemento de entrada con un atributo ID de "from" que está dentro de un elemento div con un atributo de clase de "input-container", usa: `//div[@class="input-container"]//input[@id="from"]`.
5. Para un enlace con URL, usa: `//a[contains(@href, "https://www.google.com/")]`.

Métodos XPath

Además de la sintaxis básica, XPath proporciona varios métodos que puedes utilizar para refinar tus selecciones. Son los siguientes:

1. `contains()`: selecciona elementos que contienen un string específico. Por ejemplo, para seleccionar todos los elementos `label` que contienen la palabra "From", puedes utilizar la siguiente expresión: `//label[contains(text(), "From")]`
2. `starts-with()`: selecciona elementos que comienzan con un string específico. Por ejemplo, para seleccionar todos los elementos label que comienzan con la letra "F", puedes usar la siguiente expresión: `//label[starts-with(text(), "F")]`

Consejos para escribir XPath

Al escribir expresiones XPath, hay algunos consejos a tener en cuenta:

- Especifica tus expresiones para evitar seleccionar elementos no deseados. Por ejemplo, si quieres seleccionar un botón con una etiqueta de texto específica, usa la función `text()` para seleccionar el contenido de texto del elemento.
- Utiliza atributos como clase e ID para seleccionar elementos siempre que sea posible, ya que es menos probable que cambien estos en lugar de la estructura del

árbol HTML. Por ejemplo, `//div[@id="menu"]//a[@class="link"]` es generalmente más confiable que `//div[2]//a[3]`.

- Prueba tus expresiones XPath en la consola de desarrollo del navegador antes de usarlas en tus pruebas automatizadas.

Mejores prácticas para los localizadores

1. Utiliza atributos únicos.

   Al escribir localizadores, es importante usar atributos únicos. Esto garantiza que tus pruebas apunten al elemento correcto en la página. Puedes emplear atributos como ID, nombres, clases o atributos de datos para identificar elementos.

   Los localizadores XPath pueden ser delicados. A menudo se basan en una ruta absoluta, por lo que son propensos a fallar cuando se realizan cambios en el diseño de la página. Por este motivo, es mejor evitar el uso de localizadores XPath absolutos.

2. Utiliza nombres descriptivos.

   Debes usar nombres descriptivos para tus localizadores. Esto hace que sea más fácil entender a qué elementos apuntan tus pruebas.

3. Mantén actualizados tus localizadores.

   A medida que cambia el diseño de una página, es posible que necesites actualizar tus localizadores. Revisa y actualiza tus localizadores con regularidad para asegurarte de que sigan apuntando a los elementos correctos.