**Тема 4.2. Использование Selenium WebDriver. Поиск элементов**

**Вопросы**:

1. Введение в Selenium WebDriver.
2. Виды локаторов для поиска веб-элементов на странице.
3. Организация поиска с использованием Selenium WebDriver.

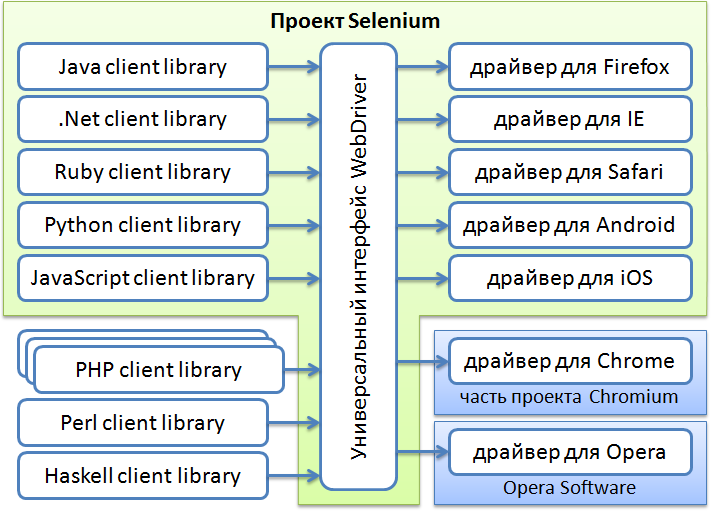
Введение в Selenium WebDriver.

**Selenium WebDriver – это программная библиотека для управления браузерами**. Часто употребляется также более короткое название WebDriver.

Иногда говорят, что это «драйвер браузера», но на самом деле это целое семейство драйверов для различных браузеров, а также набор клиентских библиотек на разных языках, позволяющих работать с этими драйверами.

Это основной продукт, разрабатываемый в рамках проекта Selenium.

WebDriver представляет собой семейство драйверов для различных браузеров плюс набор клиентских библиотек для этих драйверов на разных языках программирования:



В рамках проекта Selenium разрабатываются драйверы для браузеров Firefox, Internet Explorer и Safari, а также драйверы для мобильных браузеров Android и iOS. Драйвер для браузера Google Chrome разрабатывается в рамках проекта Chromium, а драйвер для браузера Opera (включая мобильные версии) разрабатывается компанией Opera Software. Поэтому они формально не являются частью проекта Selenium, распространяются и поддерживаются независимо. Но логически, конечно, можно считать их частью семейства продуктов Selenium.

Аналогичная ситуация и с клиентскими библиотеками – в рамках проекта Selenium разрабатываются библиотеки для языков Java, .Net (C#), Python, Ruby, JavaScript. Все остальные реализации не имеют отношения к проекту Selenium, хотя, возможно, в будущем, какие-то из них могут влиться в этот проект.

Виды локаторов для поиска веб-элементов на странице.

Локатор — это путь к элементу в интерфейсе, с помощью которого автоматизированный тест (автотест) сможет его найти. Локаторы используются в автотестах, имитирующих работу пользователя в интерфейсе, в любых системах, но мы сегодня будем говорить только о web-системах. Для других систем виды локаторов будут другими, но к ним можно применять тот же подход поиска, как и в вебе.

Создание уникального запроса для локатора – вот вторая половина. Если локатор чересчур широк, он будет возвращать ложноположительные значения. При слишком узком подходе он начнет ломаться при любом изменении DOM, и его будет сложно читать другим людям.

WebDriver предоставляет следующие типы локаторов:

|  |  |
| --- | --- |
| **Локатор** | **Описание** |
| Class name | Поиск элемента(ов) на странице по имени класса.  Имеет смысл использовать его для поиска одного элемента, когда класс элемента уникален (в противном случае мы получим первый элемент из списка всех, чьё имя класса соответствует указанному).  Может быть использован для создания списка элементов с одинаковым классом. |
| ID | Поиск одного элемента на странице по его параметру ID. Нежелательно использовать, если ID у элементов динамические. |
| Link text | Поиск элемента(ов) на странице по тексту ссылки, которую содержит в себе элемент. |
| Name | Поиск элемента(ов) на странице по его параметру name. Имя элементов задается не всегда, как и ID, поэтому, если хотите использовать этот локатор в тестах, лучше предупреждать команду разработки заранее и просить сразу добавить соответствующие атрибуты всем элементам. |
| Partial link text | Поиск элемента(ов) на странице по тексту ссылки, которую содержит в себе элемент. Отличается от локатора Link text тем, что принимает не только полную ссылку, но и часть её.  Удобно для поиска списка элементов, например, ведущих на разные страницы одного и того же домена. |
| Tag name | Поиск элемента(ов) на странице по  названию тэга, которым обозначен искомый элемент. Удобен для поиска однородных элементов в структуре документа, например, для нахождения всех заголовков или всех полей ввода. |
| CSS Selector | Поиск элемента(ов) на странице по CSS-селектору: уникальному идентификатору, который может содержать в себе информацию о классе, тэгах, ID, расположении в DOM-структуре страницы и других параметрах элемента.  Удобен для использования, когда необходимо создать уникальный идентификатор элемента, у которого нет имени или ID, но есть неповторяющийся у других элементов набор параметров.  Впрочем, при наличии у элемента имени, ID или уникального тэга использование CSS-селекторов также возможно. |
| XPath | Поиск элемента(ов) с помощью запроса XPath (XML path) — языка разметки в XML-документе.  У CSS-селекторов и XPath есть важное отличие: при использовании XPath  можно производить перемещение как в глубину DOM-иерархии, так и в обратную сторону. При поиске по CSS-селектору можно двигаться только в глубину.  Это значит, что с помощью XPath можно найти родительский элемент по дочернему, чего не позволяют CSS-селекторы. |

Организация поиска с использованием Selenium WebDriver.

Для поиска элементов на странице в Selenium WebDriver используются методы **find\_element\_by**, которые возвращают только первый из всех элементов, которые подходят под условия поиска.

|  |  |
| --- | --- |
| Цель поиска **Используемый метод** | Пример |
| Поиск по ID **find\_element\_by\_id**("user") | |  |  | | --- | --- | | 1 | <div id="user">text</div> | |
| Поиск по имени **find\_element\_by\_name**("username") | |  |  | | --- | --- | | 1 | <input name="username" type="text"> | |
| Поиск по тексту ссылки **find\_element\_by\_link\_text**("Login") | |  |  | | --- | --- | | 1 | <a href="/login">Login</a> | |
| Поиск по частичному тексту ссылки **find\_element\_by\_partial\_link\_text**("Next") | |  |  | | --- | --- | | 1 | <a href="/next">Next page</a> | |
| Поиск используя XPath **find\_element\_by\_xpath**(‘//div[@id="login"]/input’) | |  |  | | --- | --- | | 1  2  3 | <div id="login">      <input type="text">  </div> | |
| Поиск по названию тэга **find\_element\_by\_tag\_name**("body") | |  |  | | --- | --- | | 1  2  3 | <body>      ....  </body> | |
| Поиск по классу элемента **find\_element\_by\_class\_name**("table") | |  |  | | --- | --- | | 1 | <div class="table">text</div> | |
| Поиск по CSS селектору **find\_element\_by\_css\_selector**(‘#login > input [type="text"]’) | |  |  | | --- | --- | | 1  2  3 | <form id="login">      <input type="text">  </form> | |

Также существует способ для поиска элементов с помощью универсального метода **find\_element**() и полей класса **By** из библиотеки **Selenium**.

Поля класса **By**, которые можно использовать для поиска:

* **By.ID** – поиск по уникальному атрибуту id элемента;
* **By.CSS\_SELECTOR** – поиск элементов с помощью правил на основе CSS;
* **By.XPATH**– поиск элементов с помощью языка запросов XPath;
* **By.NAME** – поиск по атрибуту name элемента;
* **By.TAG\_NAME** – поиск по названию тега;
* **By.CLASS\_NAME** – поиск по атрибуту class элемента;
* **By.LINK\_TEXT** – поиск ссылки с указанным текстом. Текст ссылки должен быть точным совпадением;
* **By.PARTIAL\_LINK\_TEXT** – поиск ссылки по частичному совпадению текста.

from selenium import webdriver

from selenium.webdriver.common.by import By

browser = webdriver.Chrome()

browser.get("http://primer.html")

button = browser.find\_element(By.ID, "submit\_button")

Можно столкнуться с ситуацией, когда на странице будет несколько элементов, подходящих под заданные параметры поиска. В этом случае WebDriver вернет только первый элемент, который встретит во время поиска по HTML. Если нужен не первый, а второй или следующие элементы, нужно либо задать более точный селектор для поиска, либо использовать методы **find\_elements\_by.** Методы **find\_elements\_by**, которые в отличие от **find\_element\_by** вернут список всех найденных элементов по заданному условию.

Набор стратегий  такой же, как и в случае с find\_element\_by:

* find\_elements\_by\_css\_selector;
* find\_elements\_by\_xpath;
* find\_elements\_by\_name;
* find\_elements\_by\_tag\_name;
* find\_elements\_by\_class\_name;
* find\_elements\_by\_link\_text;
* find\_elements\_by\_partial\_link\_text.

Также для поиска нескольких элементов мы можем использовать универсальный метод **find\_elements** вместе с атрибутами класса **By**:

from selenium.webdriver.common.by import By

browser.find\_elements(By.CSS\_SELECTOR, "button.submit")