Selenium

полное руководство

Software-Testing.Ru

Поиск элементов в DOM

Стратегии поиска

Команды поиска

```
WebElement element =
    driver.findElement(<локатор>)
```

```
List<WebElement> elements =
    driver.findElements(<локатор>)
```

Команда поиска

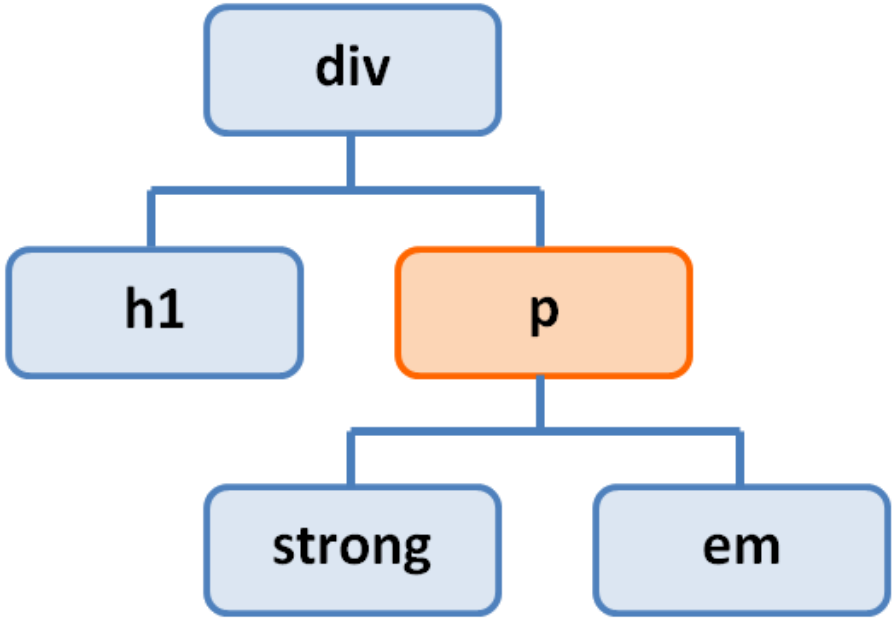
```
driver.findElement(By.name("password"))

driver.FindElement(By.name("password"))

driver.find_element_by_name("password")

@driver.find_element(:name, 'password')

driver.findElement(By.name("password"))
```

Стратегии поиска

- By.id
- By.tagName
- By.className
- By.cssSelector
- By.name
- By.linkText
- By.partialLinkText
- By.xpath

Локаторы

на основе CSS



Команда поиска

```
driver.findElement(By.cssSelector("ul#menu li.active"))

driver.FindElement(By.CssSelector("ul#menu li.active"))

driver.find_element_by_css_selector("ul#menu li.active")

@driver.find_element(:css, 'ul#menu li.active')

driver.findElement(By.css("ul#menu li.active"))
```

Структура CSS-селектора

ul#menu li.active

- серия «прыжков» по DOM
- критерий на основе тега и атрибутов
- специальные атрибуты id и class

Специальные атрибуты

- атрибут **id**
  driver.find_element_by_css_selector("#username")
  driver.find_element_by_id("username")
- атрибут **class**
  driver.find_element_by_css_selector(".error")
  driver.find_element_by_class_name("error")

Обычные атрибуты

- атрибут **name**

  driver.find_element_by_css_selector("[name=password]")

  driver.find_element_by_name("password")
- "[placeholder=search]"
- "[type=button]"

Проверка значения атрибута

- "[checked]" – наличие атрибута
- "[name = email]" – совпадение значения
- "[title *= Name]" – содержит текст
- "[src ^= http]" – начинается с текста
- "[src $= .pdf]" – заканчивается текстом

Комбинация условий

- "label" – по тегу
- ".error" – по классу
- "label.error" – по тегу и классу
- "label.error.fatal" – по тегу и двум классам
- "label.error[for=email]" – по тегу, классу и атрибуту

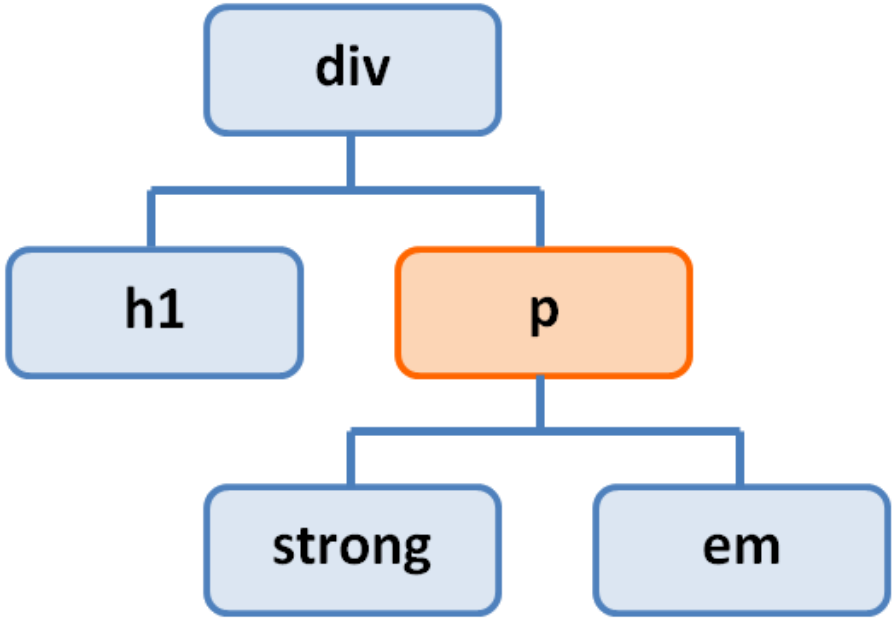
Отрицание условий

- "label:not(.error)" – сообщения не об ошибках
- "input:not([type=text])" – нетекстовые поля ввода
- "a:not([href^= http])" – локальные ссылки

Движение по дереву

- "div#main p" – p где-то внутри блока div#main
- "div#main > p" – p **непосредственно** внутри div#main
- "div#main li:first-child"
- "div#main li:last-child"
- "div#main li:nth-child(1)"
- "div#header > div:nth-of-type(1)"

Локаторы
на основе XPath



Структура XPath-запроса

```
//ul[@id='menu']/li[contains(@class, 'active')]
```

- серия «прыжков» по DOM
- критерий на основе тега и атрибутов

Проверка значения атрибута

CSS-селекторы

- "[checked]"
- "[name = email]"
- "[title *= Name]"
- "[src ^= http]"
- "[src $= .pdf]"

XPath-запросы

- "//*[@checked]"
- "//*[@name='email']“
- "//*[contains(@title, 'Name')]"
- "//*[starts-with(@src, 'http')]"

Комбинация условий

CSS-селекторы

- "label"
- ".error"
- "label.error"
- "label.error.fatal“
- "label.error[for=email]"

XPath-запросы

- "//label"
- "//*[contains(@class, 'error')]"
- "//label[contains(@class, 'error')]"
- "//label[contains(@class, 'error') and contains(@class, 'fatal')]"
- "//label[contains(@class, 'error') and contains(@class, 'fatal') and @for='email']"

Движение по дереву

CSS-селектор

- "div#main p"
- "div#main > p"
- "div#main li:first-child"
- "div#main li:last-child"
- "div#main > div:nth-of-type(2)"

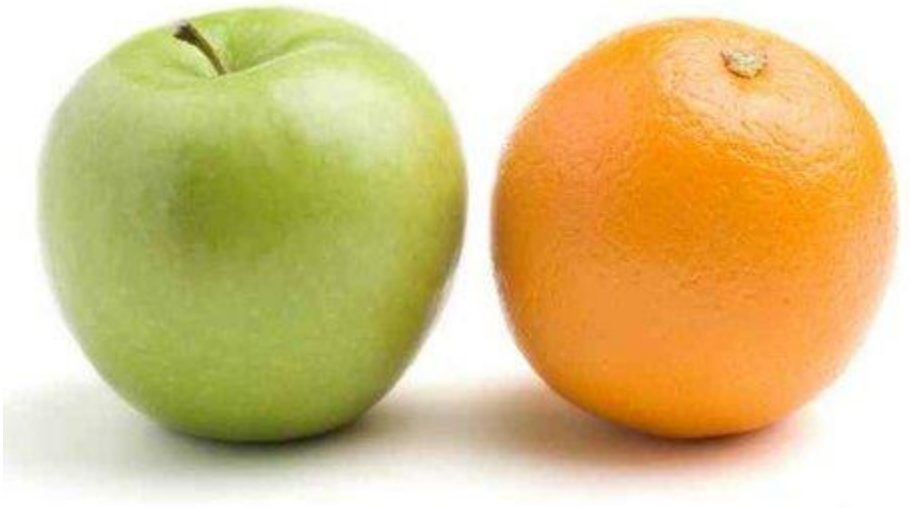
XPath-запрос

- "//div[@id='main']//p"
- "//div[@id='main']/p"
- "//div[@id='main']/div[2]"

XPath мощнее чем CSS?

- Движение в любом направлении
  //input[@id='search']/../input[@type='button']
- Поиск по тексту
  //a[contains(., 'Edit')]
- Подзапросы
  //form[.//input[@name='password']]

Сравнение локаторов



Частные случаи CSS-селекторов

- By.tagName("div")
- By.id("main")
- By.className("error")
- By.cssSelector("div")
- By.cssSelector("#main")
- By.cssSelector(".error")

Смотря как сравнивать...

- Мощность языка – XPath
- Краткость и понятность – CSS
- Поддержка в браузерах – CSS
- Скорость – CSS (с минимальным преимуществом)

Поиск внутри элемента

Контекст поиска

- `input = driver.find_element_by_name("password")`

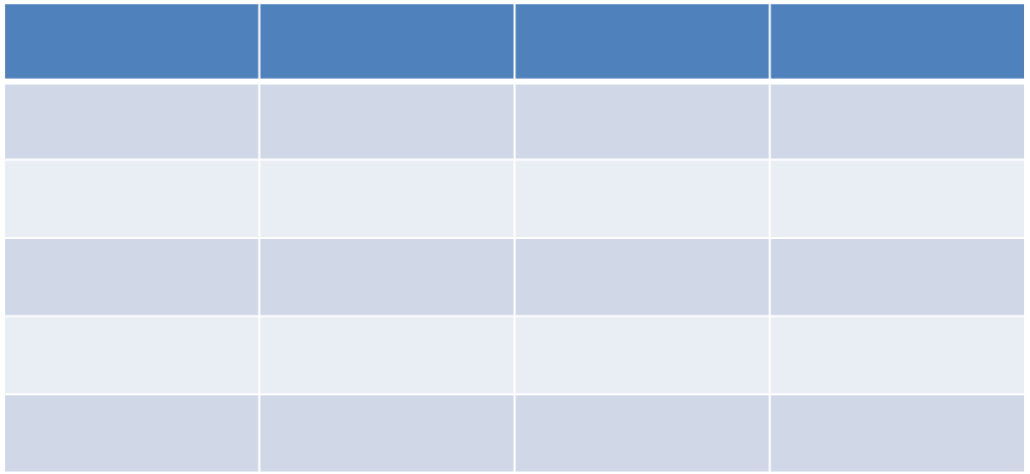
- ```
  form = driver.find_element_by_id("login-modal")
  input = form.find_element_by_name("password")
  ```

- ```
  input = driver.find_element_by_css_selector(
              "#login-modal [name=password]")
  ```

Относительные запросы в XPath

```
form = driver.find_element_by_id("login-modal")
input = form.find_element_by_xpath(
    ".//input[@name='password']"
)
```

Несколько элементов



Пример: чтение таблицы

```
table = driver.find_element_by_id("users")
rows = table.find_elements_by_tag_name("tr")
for row in rows:
    name = row.find_element_by_xpath("./td[1]").text
    email = row.find_element_by_xpath("./td[2]").text
```

Пример: чтение таблицы

```
table = driver.find_element_by_id("users")
rows = table.find_elements_by_tag_name("tr")
for row in rows:
    cells = row.find_elements_by_tag_name("td")
    name = cells[0].text
    email = cells[1].text
```

Пример: проверка наличия

```
boolean isElementPresent(WebDriver driver, By locator) {
    try {
        driver.findElement(locator);
        return true;
    } catch (NoSuchElementException ex) {
        return false;
    }
}
```

Пример: проверка наличия

```
boolean isElementPresent(WebDriver driver, By locator) {
    return driver.findElements(locator).size() > 0;
}
```

JavaScript

ищет элементы

Не найден

нужный элемент

Если элемент не найден, то...

- findElement

  выбрасывает исключение NoSuchElementException

- findElements

  возвращает пустой список

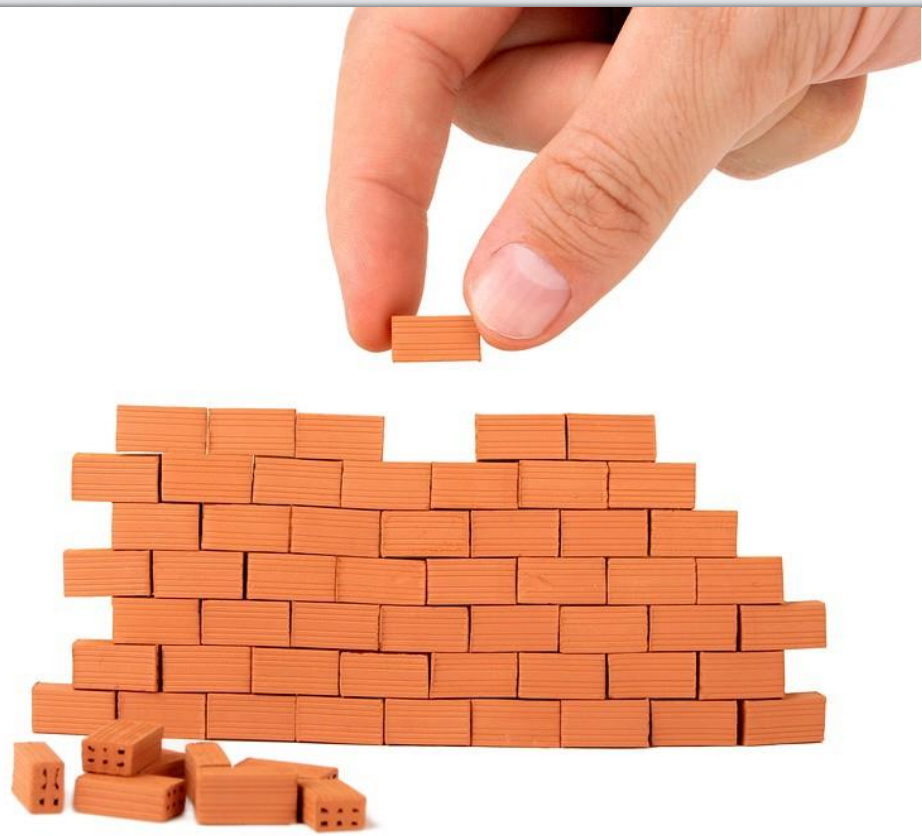
Ожидание

появления элемента

Если элемент не найден...

- Не та страница открыта
- Неправильный локатор
- Элемент находится внутри фрейма
- Нужно немного подождать

Построение локаторов



Устойчивость к изменениям вёрстки

- Максимально точные критерии выбора
- Как можно меньше порядковых номеров
- Привязка к ближайшему уникальному элементу
- Минимум прыжков по DOM

Selenium

полное руководство

Software-Testing.Ru