2.07 Selenium Locators

Guiding principles

Good Locators are:

- Unique
- Descriptive
- Unlikely to change

Be sure to:

1. Start with ID and Class
2. Use CSS selectors (or XPath) when you need to traverse
3. Talk with a developer on your team when the app is hard to automate
    - Tell them what you're trying to automate
    - Work with them to get more semantic markup added to the page

ID

```
driver.find_element(By.ID, "username")
```

Class

```
driver.find_element(By.CLASS_NAME, "dues")
```

CSS Selectors

```
driver.find_element(By.CSS_SELECTOR, "#example")
```

Approach	Locator	Description
ID	`#example`	`#` denotes an ID
Class	`.example`	. denotes a Class

Classes	`.flash.success`	use . in front of each class for multiple
Direct child	`div > a`	finds the element in the next child
Child/subschild	`div a`	finds the element in a child or child's child
Next sibling	`input.username + input`	finds the next adjacent element
Attribute values	`form input[name='username']`	a great alternative to id and class matches
Attribute values	`input[name='continue'][type='button']`	can chain multiple attribute filters together
Location	`li:nth-child(4)`	finds the 4th element only if it is an li
Location	`li:nth-of-type(4)`	finds the 4th li in a list
Location	`*:nth-child(4)`	finds the 4th element regardless of type
Sub-string	`a[id^='beginning_']`	finds a match that starts with (prefix)
Sub-string	`a[id$='_end']`	finds a match that ends with (suffix)
Sub-string	`a[id*='gooey_center']`	finds a match that contains (substring)

Inner text	`a:contains('Log Out')`	an alternative to substring matching

NOTE: Older browser (e.g., Internet Explorer 8) don't support CSS Pseudo-classes, so some of these locator approaches won't work (e.g., Location matches and Inner text matches).

For more info see one of the following resources:

- CSS Selector Game
- CSS & XPath Examples by Sauce Labs
- The difference between nth-child and nth-of-type
- CSS vs. XPath Selenium benchmarks
- CSS Selectors Reference
- XPath Syntax Reference