**XPATH**

XPath는 XML 또는 HTML 문서의 특정 부분을 찾아내는 데 사용되는 언어입니다. 파일 시스템 경로와 유사하게 작동하며, 경로 표기법을 사용하여 원하는 요소를 정확히 찾아낼 수 있습니다.

**기본 표기법**

| 표기법 | 설명 | 예시 |
| --- | --- | --- |
| nodename | 자식 노드 선택 | bookstore/book |
| / | 루트(최상위) 노드부터 시작 | /bookstore |
| // | 문서의 어느 위치에서든 노드 선택 | //book |
| . | 현재 노드 | ./p |
| .. | 부모 노드 | //li/.. |
| @ | 속성(Attribute) | //a/@href |

**조건 및 필터**

대괄호 []를 사용하여 조건을 추가하거나 요소를 필터링할 수 있습니다.

| 표기법 | 설명 | 예시 |
| --- | --- | --- |
| [position()] | 특정 위치의 노드 선택 | //book[2] (두 번째 <book> 노드) |
| [last()] | 마지막 노드 선택 | //book[last()] |
| [@attribute] | 특정 속성을 가진 노드 선택 | //a[@href] |
| [@attribute='value'] | 특정 속성값을 가진 노드 선택 | //a[@class='nav'] |
| [text()='...'] | 특정 텍스트를 가진 노드 선택 | //p[text()='Hello'] |

**1. href 값 추출**

href는 <a href="..."> 태그의 \*\*속성(attribute)\*\*이므로, get\_attribute() 메서드를 사용합니다.

from robocorp.browser import page

# XPath로 요소를 찾고, 그 요소의 href 속성값을 가져옵니다.

locator = page().locator(".//td/a[@href]/p[@class='prname']")

href\_value = locator.get\_attribute("href")

print(f"추출된 href 값: {href\_value}")

**2. prname 값(텍스트) 추출**

prname은 <p> 태그의 class 속성이므로, 이 클래스를 가진 <p> 태그의 **텍스트 값**을 추출해야 합니다. 이를 위해 inner\_text() 또는 text\_content() 메서드를 사용합니다.

from robocorp.browser import page

# XPath로 요소를 찾고, 그 요소의 텍스트 값을 가져옵니다.

locator = page().locator(".//td/a[@href]/p[@class='prname']")

prname\_value = locator.inner\_text()

# 또는 prname\_value = locator.text\_content()

print(f"추출된 prname 값: {prname\_value}")

**3. 참고: 부모/자식 관계**

XPATH에서 .//td/a[@href]/p[@class='prname']는 <a> 태그의 하위 요소인 <p> 태그를 선택하는 문법입니다. 따라서 href 값을 추출하려면 <p>가 아닌 <a> 태그를 직접 가리켜야 합니다.

아래 코드처럼 a[@href]를 로케이터로 지정한 후 get\_attribute()를 호출하면 더 명확합니다.

from robocorp.browser import page

# 부모인 <a> 태그를 로케이터로 지정

a\_tag\_locator = page().locator(".//td/a[@href][./p[@class='prname']]")

# 또는 page().locator("xpath=.//td/a[p[@class='prname']]")

# <a> 태그의 href 속성값 추출

href\_value = a\_tag\_locator.get\_attribute("href")

# <a> 태그의 하위 요소인 <p> 태그의 텍스트 추출

prname\_value = a\_tag\_locator.locator("p").inner\_text()

print(f"href 값: {href\_value}")

print(f"prname 값: {prname\_value}")

**함수**

XPath는 특정 값을 가져오거나 조건을 설정할 때 유용한 다양한 함수를 제공합니다.

| 함수 | 설명 | 예시 |
| --- | --- | --- |
| text() | 노드의 텍스트 값 추출 | //a[text()='Home'] |
| contains(string, string) | 문자열 포함 여부 | //a[contains(@href, 'products')] |
| starts-with(string, string) | 특정 문자열로 시작 여부 | //div[starts-with(@id, 'item')] |
| ends-with(string, string) | 특정 문자열로 끝나는지 여부 | //a[ends-with(@href, '.pdf')] |
| count(nodeset) | 노드의 개수 | count(//div) |

**논리 연산자**

여러 조건을 결합할 때 사용합니다.

| 연산자 | 설명 | 예시 |
| --- | --- | --- |
| and | 두 조건 모두 참 | //a[@href and @class='nav'] |
| or | 둘 중 하나라도 참 | //a[@id='nav1' or @id='nav2'] |

**Axis (축)**

기준 노드를 중심으로 다른 노드를 탐색할 때 사용합니다.

| Axis | 설명 | 예시 |
| --- | --- | --- |
| ancestor | 부모, 부모의 부모 등 상위 노드 | //span/ancestor::div |
| child | 바로 아래 자식 노드 | //div/child::a |
| descendant | 모든 하위 노드 | //div/descendant::p |
| following-sibling | 동일한 부모를 가진 다음 형제 노드 | //p[1]/following-sibling::p |
| preceding-sibling | 동일한 부모를 가진 이전 형제 노드 | //p[2]/preceding-sibling::p |

대부분의 경우 //와 []를 조합하고 @를 사용해 속성을 찾는 것이 가장 흔한 방법입니다.

**CSS**

CSS도 XPath와 비슷하게 웹 요소를 선택하는 데 사용됩니다. 하지만 CSS 셀렉터는 XPath보다 문법이 더 간결하고 직관적인 특징이 있습니다.

**기본 문법**

| 문법 | 설명 | 예시 |
| --- | --- | --- |
| element | 태그 이름으로 요소 선택 | p (모든 <p> 태그) |
| .class | class 이름으로 요소 선택 | .header (모든 class="header"인 요소) |
| #id | id 값으로 요소 선택 | #main-menu (모든 id="main-menu"인 요소) |
| element.class | 태그와 클래스를 결합 | p.intro (class="intro"인 모든 <p> 태그) |
| \* | 모든 요소 선택 | \* (모든 태그) |

**결합자 (Combinators)**

여러 셀렉터를 조합하여 특정 관계에 있는 요소를 선택합니다.

| 문법 | 설명 | 예시 |
| --- | --- | --- |
| (공백) | **자손(descendant)** 셀렉터: 하위의 모든 요소 선택 | div p (<div> 안에 있는 모든 <p> 태그) |
| > | **자식(child)** 셀렉터: 바로 아래의 자식 요소만 선택 | ul > li (<ul> 바로 아래의 <li> 태그) |
| + | **인접 형제(adjacent sibling)** 셀렉터: 바로 다음 형제 요소 선택 | h1 + p (<h1> 바로 다음에 오는 <p> 태그) |
| ~ | **일반 형제(general sibling)** 셀렉터: 다음에 오는 모든 형제 요소 선택 | h1 ~ p (<h1> 뒤에 오는 모든 <p> 태그) |

**속성 셀렉터 (Attribute Selectors)**

요소의 속성을 기준으로 선택합니다.

| 문법 | 설명 | 예시 |
| --- | --- | --- |
| [attribute] | 특정 속성이 존재하는 요소 | a[href] (href 속성이 있는 모든 <a> 태그) |
| [attribute="value"] | 속성값이 정확히 일치하는 요소 | input[type="text"] |
| [attribute\*="value"] | 속성값에 특정 문자열이 포함된 요소 | a[href\*="google"] (href에 'google'이 포함된 링크) |
| [attribute^="value"] | 속성값이 특정 문자열로 시작하는 요소 | a[href^="http://"] |
| [attribute$="value"] | 속성값이 특정 문자열로 끝나는 요소 | a[href$=".pdf"] |

**가상 클래스 (Pseudo-classes)**

요소의 특정 상태를 기반으로 선택합니다.

| 문법 | 설명 | 예시 |
| --- | --- | --- |
| :first-child | 부모의 첫 번째 자식 | p:first-child |
| :last-child | 부모의 마지막 자식 | li:last-child |
| :nth-child(n) | n번째 자식 | li:nth-child(3) (세 번째 <li> 태그) |
| :hover | 마우스가 올라간 상태 | a:hover |
| :enabled, :disabled | 활성화/비활성화된 요소 | input:disabled |
| :has-text('text') | 특정 텍스트를 포함하는 요소 (Robocorp/Playwright 확장 문법) | div:has-text('Submit') |

**XPath vs CSS 셀렉터**

* **CSS 셀렉터의 장점**: 구문이 간결하고 읽기 쉽습니다. 대부분의 경우 XPath보다 성능이 빠릅니다.
* **CSS 셀렉터의 단점**: 부모 노드를 선택하는 기능이 없습니다. 형제 관계나 복잡한 조건에 대한 표현이 XPath보다 제한적입니다.

일반적으로는 CSS 셀렉터가 더 직관적이고 효율적이므로, 먼저 CSS 셀렉터로 해결을 시도하고 복잡한 상황에서만 XPath를 사용하는 것이 좋습니다.