Xpath cheatsheet

Xpath test bed

Test queries in the Xpath test bed:
Xpath test bed (whitebeam.org)

Browser console

```
$x('//div')
```

Works in Firefox and Ch

Selectors

Descendant selectors

`h1`	`//h1`
`div p`	`//div//p`
`ul > li`	`//ul/li`
`ul > li > a`	`//ul/li/a`
`div > *`	`//div/*`
`:root`	`/`
`:root > body`	`/body`

Attribute selectors

`#id`
`.class`
`input[type="submit"]`
`a#abc[for="xyz"]`
`a[rel]`
`a[href^='/']`
`a[href$='pdf']`
`a[href*='://']`
`a[rel~='help']`

Order selectors

`ul > li:first-child`	`//ul/li[1]`
`ul > li:nth-child(2)`	`//ul/li[2]`
`ul > li:last-child`	`//ul/li[last()]`
`li#id:first-child`	`//li[@id="id"][1]`
`a:first-child`	`//a[1]`

Siblings

`?`
`h1 ~ ul`
`h1 + ul`
`h1 ~ #id`

`a:last-child`	`//a[last()]`

Other things

`h1:not([id])`	`//h1[not(@id)]`
Text match	`//button[text()="Submit"]`
Text match (substring)	`//button[contains(text(),"Go")]`
Arithmetic	`//product[@price > 2.50]`
Has children	`//ul[*]`
Has children (specific)	`//ul[li]`
Or logic	`//a[@name or @href]`
Union (joins results)	`//a | //div`

jQuery

```
$('ul > li').parent(
```

? sec

?)

Class check

```
//div[contains(conca
```

Xpath doesn't have the

?

Expressions

Steps and axes

`//`	`ul`	`/`	`a[@id='link']`
Axis	Step	Axis	Step

Prefixes

Prefix
`//`
`./`
`/`

Axes

Axis	Example	What
`/`	`//ul/li/a`	Child
`//`	`//[@id="list"]//a`	Descendant

Separate your steps with `/`. Use two (`//`) if you don't want to select direct children.

Steps

```
//div
//div[@name='box']
//[@id='link']
```

A step may have an ele
things:

```
//a/text()      #=> "
//a/@href       #=> "
//a/*           #=> A
```

Predicates

Predicates

```
//div[true()]
//div[@class="head"]
//div[@class="head"][@id="top"]
```

Restricts a nodeset only if some condition is true. They can be chained.

Using nodes

```
# Use them inside functions
//ul[count(li) > 2]
//ul[count(li[@class='hide']) > 0]
```

```
# This returns `<ul>` that has a `<li>` child
//ul[li]
```

You can use nodes inside predicates.

Chaining order

```
a[1][@href='/']
a[@href='/'][1]
```

Order is significant, these two are different.

Operators

```
# Comparison
//a[@id = "xyz"]
//a[@id != "xyz"]
//a[@price > 25]
```

```
# Logic (and/or)
//div[@id="head" and
//div[(x and y) or r
```

log

Indexing

```
//a[1]
//a[last()]
//ol/li[2]
//ol/li[position()=2
//ol/li[position()>1
```

Use [] with a number, c

Nesting predicates

```
//section[//h1[@id='
```

This returns <section>

Functions

Node functions

```
name()                  # //[starts-with(name(), 'h')]
text()                  # //button[text()="Submit"]
                        # //button/text()
lang(str)
namespace-uri()
```

```
count()                 # //table[count(tr)=1]
position()              # //ol/li[position()=2]
```

Boolean functions

```
not(expr)
```

String functions

```
contains()
starts-with()
ends-with()
```

Type conversion

```
string()
number()
boolean()
```

```
concat(x,y)
substring(str, start
```

Axes

Using axes

```
//ul/li                       # ul > li
//ul/child::li                # ul > li (same)
//ul/following-sibling::li    # ul ~ li
//ul/descendant-or-self::li   # ul li
//ul/ancestor-or-self::li     # $('ul').closest('li')
```

Steps of an expression are separated by `/`, usually used to pick child nodes. That's not alway
specify a different "axis" with `::`.

`//`	`ul`	`/child::`	`li`
Axis	Step	Axis	Ste

Child axis

```
# both the same
//ul/li/a
//child::ul/child::l
```

`child::` is the default a

```
# both the same
# this works because
//ul[li]
//ul[child::li]
```

```
# both the same
//ul[count(li) > 2]
//ul[count(child::li
```

Descendant-or-self axis

```
# both the same
//div//h4
//div/descendant-or-self::h4
```

`//` is short for the `descendant-or-self::` axis.

```
# both the same
//ul//[last()]
//ul/descendant-or-self::[last()]
```

Other axes

Axis
`ancestor`
`ancestor-or-self`
`attribute`
`child`
`descendant`

Unions

```
//a | //span
```

Use `|` to join two expressions.

Axis
`parent`
`following`
`following-sibling`
`preceding`
`preceding-sibling`
There are other axes yo

More examples

Examples

```
//*                 # all elements
count(//*)          # count all elements
(//h1)[1]/text()    # text of the first h1 heading
//li[span]          # find a <li> with an <span> inside it
                    # ...expands to //li[child::span]
//ul/li/..          # use .. to select a parent
```

Find a parent

```
//section[h1[@id='se
```

Finds a <section> that

```
//section[//h1[@id='
```

Finds a <section> that child)

Closest

```
./ancestor-or-self::[@class="box"]
```

Works like jQuery's `$().closest('.box')`.

Attributes

```
//item[@price > 2*@c
```

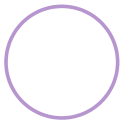
Finds <item> and check

References

- Xpath test bed (whitebeam.org)

1 Comment for this cheatsheet. Write yours!

Search 367+ cheatsheets



Over 367 curated cheatsheets, by developers for developers.

Devhints home

Other HTML cheatsheets

- Input tag cheatsheet
- HTML meta tags cheatsheet
- Layout thrashing cheatsheet
- Appcache cheatsheet
- Applinks cheatsheet
- HTML emails cheatsheet

Top cheatsheets

- Elixir cheatsheet
- ES2015+ cheatsheet
- React.js cheatsheet
- Vim cheatsheet
- Vim scripting cheatsheet
- Capybara cheatsheet