XPATH 快速参考

轴、谓词、函数、运算符、节点选择

语法

路径表达式

`/`	根节点（绝对路径起点）
`/bookstore/book`	直接子节点选择
`//book`	选择任意位置的所有 book 节点
`.`	当前上下文节点
`..`	当前节点的父节点
`@lang`	名为 lang 的属性
`node()`	任意类型的节点
`*`	任意元素节点
`@*`	任意属性

基本示例

```
/html/body/div          # absolute path to <div>
//input[@type='text']   # all text inputs
//div[@class='main']/*  # children of div.main
//a/@href               # all href attributes
```

合并路径

```
//book/title | //book/price   # union of two paths
//h1 | //h2 | //h3           # multiple element types
```

轴

轴方向

`child::`	直接子节点（默认轴）
`parent::`	直接父节点
`ancestor::`	到根节点的所有祖先
`ancestor-or-self::`	祖先 + 当前节点
`descendant::`	所有后代节点
`descendant-or-self::`	后代 + 当前节点
`following::`	文档中当前节点之后的所有节点
`following-sibling::`	之后的兄弟节点
`preceding::`	文档中当前节点之前的所有节点
`preceding-sibling::`	之前的兄弟节点
`self::`	仅当前节点
`attribute::`	当前节点的属性
`namespace::`	命名空间节点

轴示例

```
//div/child::p             # <p> children of <div>
//td/parent::tr            # <tr> parent of <td>
//h2/following-sibling::p  # <p> after <h2>
//li/ancestor::ul          # <ul> containing <li>
```

谓词

用谓词过滤

```
//book[1]                  # first book element
//book[last()]             # last book element
//book[position() < 3]    # first two books
//book[@lang='en']         # books with lang="en"
//book[price > 30]         # books with price > 30
```

谓词模式

`[n]`	位置 n 的元素（从 1 开始）
`[last()]`	最后一个元素
`[last()-1]`	倒数第二个元素
`[@attr]`	存在某属性
`[@attr='val']`	属性等于某值
`[element]`	存在某子元素
`[element='text']`	子元素包含指定文本
`[not(@attr)]`	不存在某属性

链式谓词

```
//div[@class='list']//a[1]         # first <a> in div.list
//input[@type='text'][@name='q']   # AND condition
//book[price>30][@lang='en']       # multiple conditions
```

函数

字符串函数

`contains(s, sub)`	s 包含 sub 时为真
`starts-with(s, pre)`	s 以 pre 开头时为真
`string-length(s)`	字符串长度
`normalize-space(s)`	去除首尾空白并折叠内部空白
`concat(a, b, ...)`	连接字符串
`substring(s, pos, len)`	截取子字符串（从 1 开始）
`translate(s, from, to)`	逐字符替换

数字函数

`sum(node-set)`	数值之和
`count(node-set)`	节点数量
`floor(n)`	向下取整
`ceiling(n)`	向上取整
`round(n)`	四舍五入到最近整数
`number(val)`	转换为数字

函数示例

```
//div[contains(@class, 'active')]
//a[starts-with(@href, 'https')]
//p[string-length(text()) > 100]
//ul[count(li) > 5]
```

运算符

比较运算符

`=`	等于
`!=`	不等于
`<`	小于
`<=`	小于等于
`>`	大于
`>=`	大于等于

逻辑与算术运算符

`and`	逻辑与
`or`	逻辑或
`not()`	逻辑非（函数）
`+`	加法
`-`	减法
`*`	乘法
`div`	除法
`mod`	取模
`|`	节点集合并

运算符示例

```
//book[price > 20 and price < 50]
//item[@type='a' or @type='b']
//span[not(contains(@class, 'hidden'))]
```

节点测试

节点类型

`node()`	任意节点（元素、文本、注释、PI）
`text()`	仅文本节点
`comment()`	仅注释节点
`processing-instruction()`	处理指令节点
`*`	任意元素节点
`@*`	任意属性节点
`element-name`	指定名称的元素

节点测试示例

```
//p/text()                # text content of <p>
//div/comment()           # comments inside <div>
//body/node()             # all child nodes of <body>
//div/*                   # all element children of <div>
```

布尔函数

`true()`	布尔真
`false()`	布尔假
`boolean(expr)`	转换为布尔值
`not(expr)`	取反布尔值
`lang(code)`	节点语言匹配时为真

缩写

简写 vs 完整写法

`(无)`	child::（默认轴）
`@`	attribute::
`//`	/descendant-or-self::node()/
`.`	self::node()
`..`	parent::node()
`[n]`	[position()=n]

缩写示例

```
# These pairs are equivalent:
child::div            →  div
attribute::href       →  @href
/descendant-or-self::node()/p  →  //p
self::node()          →  .
parent::node()        →  ..
```

常用缩写模式

```
//div[@id='main']       # div with id="main"
//table//td             # all <td> in any <table>
../sibling              # sibling via parent
.//span                 # span descendants of context
```

常用模式

爬虫 / 测试

```
//input[@name='username']          # form input by name
//button[text()='Submit']          # button by text
//div[contains(@class, 'error')]   # element by partial class
//a[contains(@href, 'login')]      # link by partial href
```

条件选择

```
//div[@class='item'][.//span[@class='price']]
//tr[td[1]='Active']               # row where 1st cell = Active
//*[contains(text(), 'Warning')]   # any element with text
```

XPath in Python (lxml)

```
from lxml import html
tree = html.fromstring(page_content)
links = tree.xpath('//a/@href')
titles = tree.xpath('//h2/text()')
```

XPath in Selenium

```
driver.find_element(By.XPATH, "//input[@id='search']")
driver.find_elements(By.XPATH, "//li[@class='result']")
```