

目录

前言	1.1
基本语法	1.2
使用场景和举例	1.3
附录	1.4
参考资料	1.4.1

XPath知识总结

- 最新版本: v0.3
- 更新时间: 20190526

鸣谢

感谢我的老婆陈雪雪的包容理解和悉心照料，才使得我 `crifan` 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

简介

整理出在上层软件领域折腾 `XPath` 所获得的经验和心得，供参考。

源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

Gitbook源码

- [crifan/xpath_summary: XPath知识总结](#)

如何使用此Gitbook源码去生成发布为电子书

详见：[crifan/gitbook_template: demo how to use crifan gitbook template and demo](#)

在线浏览

- [XPath知识总结 book.crifan.com](#)
- [XPath知识总结 crifan.github.io](#)

离线下载阅读

- [XPath知识总结 PDF](#)
- [XPath知识总结 ePUB](#)
- [XPath知识总结 Mobi](#)

版权说明

此电子书教程的全部内容，如无特别说明，均为本人原创和整理。其中部分内容参考自网络，均已备注了出处。如有发现侵犯您版权，请通过邮箱联系我 `admin 艾特 crifan.com`，我会尽快删除。谢谢合作。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2019-07-17 20:10:27

XPath基本语法介绍

常见符号

XPath 中有很多常用符号，下面来解释其具体含义和基本的逻辑：

- 查找和定位元素的位置和层次时用到的：
 - /：表示绝对路径，绝对路径是指从根目录开始
 - //：表示相对路径
 - 典型的用法是表示从任意路径去搜索
 - .：表示当前层
 - ..：表示上一层
- 想要具体查找和定位哪个元素时用到的：
 - *：表示通配符
- 找到了该元素还要判断其属性时用到的：
 - []：属性的判断条件表达式
 - @：表示属性，@属性
 - 比如常用的 @class，@name，@id 等等

常见函数

- contains()：是否包含对应的值
- starts-with()：是否以某种字符串开始
- ends-with()：是否以某种字符串结束
- last()：列表/集合类元素的最后一个值
- text()：一个节点的文本值

逻辑判断

- and：逻辑且
- not：否定
 - 一般情况下会与返回值为 true 或者 false 的函数组合起来使用。
 - 比如上面提到的 contains() 与 starts-with()
 - 当然 not() 还有一种特殊用法就是直接 not

场景和举例

下面就来介绍一下XPath的常见使用场景和具体的示例。

XPath常见使用场景

比如之前自己就遇到过的：

- docbook的xml的处理中的XSLT中使用xpath
- beautifulsoup中的html的提取
- selenium中元素的选择
- scrapy的Selector中提取内容

XPath使用举例

TODO： 1.把selenium相关的帖子中的示例代码，尤其是关于xpath的，整理过来供参考

常见符号和属性判断

@属性 举例

对于判断元素中是否包含某个属性（注意不是去判断属性的值），则语法是： `[@someProperty]`

比如：

```
<span aria-label="Price" class="x-hidden-focus">$799.00</span>
```

和

```
<div class="price-text srv_price">
    <s class="srv_saleprice" aria-label="Full price was $1,699.00">$1,699.00</s>
    <span>&ampnbsp</span>
    <div class="price-disclaimer ">
        <span aria-label="Now $1,299.00" class="x-hidden-focus">$1,299.00</span>
    </div>
</div>
```

只想要判断中存在aria-label属性即可，则可以用：`//span[@aria-label]`

更多例子如下：

- `@class="dropdown-menu"`
 - `cartNumOptionElemList = driver.find_elements_by_xpath('//ul[@class="dropdown-menu"]/li[@role="option"]')`
- `@name="loginfmt"`

- `inputEmailElement = WebDriverWait(driver, gCfg["waitTimeout"]).until(EC.presence_of_element_located((By.XPATH, '//input[@type="email" and @name="loginfmt"]')))`
- `@id="idSIButton9"`
 - `nextElement = driver.find_element_by_xpath('//input[@type="submit" and @id="idSIButton9" and @value="Next"]')`

selenium中特殊列表元素的选择

之前遇到列表选项的html不是 select 的 option , 而是:



The screenshot shows a dropdown menu with the following HTML structure:

```

<ul id="22bc22dd-ef9d-4d3c-8de9-1e7bc704f9f9_menu" role="group" aria-labelledby="22bc22dd-ef9d-4d3c-8de9-1e7bc704f9f9" class="dropdown-menu">
    <li role="option">
        <a data-m="{'aN': 'shoppingCart', 'cN': 'UpdateQuantity', 'ot': 'bhvr', 'v1': '91', 'pid': '8X58XHDX57SX', 'sku': 'F96R', 'itemCount': '1'}" id="22bc22dd-ef9d-4d3c-8de9-1e7bc704f9f9_menuItem_1" class="ember-view x-hidden-focus"> 1
    </a>      </li>
    <li role="option">
        ...
    </li>

```

即, ul 的 li 的列表

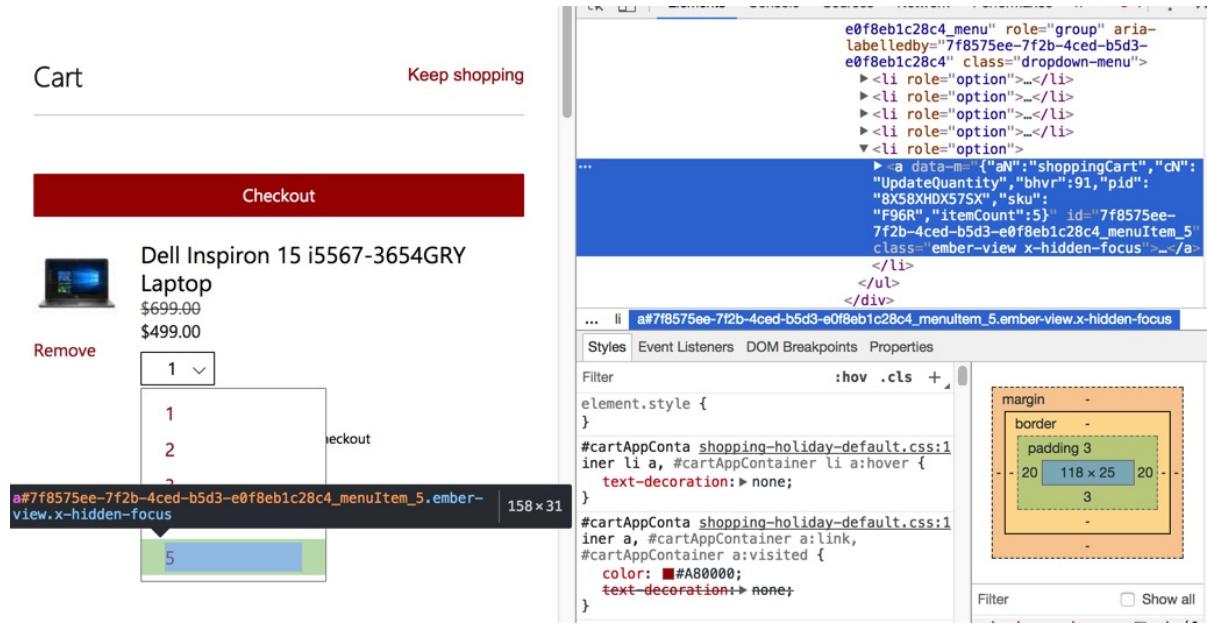
最后用代码:

```

cartNumOptionElemList = driver.find_elements_by_xpath('//ul[@class="dropdown-menu"]/li[@role="option"]')
cartNumOptionCount = len(cartNumOptionElemList)
logging.info("cartNumOptionElemList=%s, cartNumOptionCount=%s", cartNumOptionElemList, cartNumOptionCount)
if cartNumOptionCount < gCfg["msStore"]["onceBuyNum"]:
    logging.error("Current Cart select max number %s < expected select number %s", cartNumOptionCount, gCfg["msStore"]["onceBuyNum"])
    driver.quit()
toSelectIdx = gCfg["msStore"]["onceBuyNum"] - 1
# carNumSelect = Select(cartNumOptionElemList)
# carNumSelect.select_by_index(gCfg["msStore"]["onceBuyNum"])
carNumSelectElem = cartNumOptionElemList[toSelectIdx]
logging.info("carNumSelectElem=%s", carNumSelectElem)
carNumSelectElem.click()
# aLinkElem = carNumSelectElem.find_element_by_link_text(str(gCfg["msStore"]["onceBuyNum"]))
# logging.info("aLinkElem=%s", aLinkElem)
# aLinkElem.click()

```

去实现, 找到列表的元素, 点击后下拉显示所有的选项, 然后点击选中某个选项:



常见函数的使用

text()函数

- 对于HTML代码:

```
<a target="_self" href="/s?rsv_idx=1&wd=111&usm=3&ie=utf-8&sl_lang=en&rsv_srlang=en&rsv_rq=en&rqlang=cn">英文结果</a>
```

定位英文结果即可使用: //a[text()="英文结果"]

另外一个例子:

- `clickHereBtnElement = driver.find_element_by_xpath('//a[text()="click here"]')` 中的
`text()="click"`

last()函数

- `//input[@name="identity"] [last()]`

docbook的xml的处理 -» XSLT中使用xpath

```
<key name="book" match="books/book" use="concat(@title, '|', @author)"/>
```

逻辑判断

not 逻辑非

- `//input[@name="identity" and not (contains(@class, 'A'))]`
- `//input[not(@class)]`

不等于 !=

Selenium的Chrome的driver中用xpath去查找元素， 是可以通过：

```
// a[@id="uhf-shopping-cart" and @aria_label!="0 items in shopping cart"]
```

实现判断， 只匹配到

```
id="uhf-shopping-cart" aria_label="2 items in shopping cart"
```

这类元素， 而不匹配：

```
id="uhf-shopping-cart" aria_label="0 items in shopping cart"
```

即xpath支持：

```
@someProperty != "some not expected text"
```

这种写法的

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2018-01-16 09:41:55

附录

下面列出相关官网的语法教程和参考资料。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2018-01-16 09:28:02

官网教程

- [XPath Tutorial](#)
- [XML Path Language \(XPath\)](#)
- [XPath 1.0 Tutorial @ZVON.org](#)
- [starts-with \[XPath 2.0 Reference @ Zvon.org\]](#)
- [contains \[XPath 2.0 Reference @ Zvon.org\]](#)
- [matches \[XPath 2.0 Reference @ Zvon.org\]](#)
- [ends-with \[XPath 2.0 Reference @ Zvon.org\]](#)
- [Selectors](#)

参考资料

- [【已解决】Xpath中如何判断属性值不等于某个值](#)
- [【已解决】xpath中如何实现或者or的判断](#)
- [【已解决】Selenium如何点击下拉框并选择某个值](#)
- [查找元素 — Selenium-Python中文文档 2 documentation](#)
- [xml - Multiple conditions \(and operator\) in XPath - Stack Overflow](#)
- [Selenium 自动化测试从零实战 - CSDN博客](#)

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2018-01-16 09:27:56