法律声明

- 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

- 课程详情请咨询
  - 微信公众号：小象
  - 新浪微博：ChinaHadoop

分布式爬虫

大纲

- 网站结构分析及案例：马蜂窝
- XPath
- 正则表达式
- 动态网页
- Headless 的浏览器：PhantomJS
- 浏览器的驱动：Selenium

XPath

基本语法

表达式	描述
nodename	选取此节点的所有子节点，tag 或 * 选择任意的tag
/	从根节点选取，选择直接子节点，不包含更小的后代（例如孙、从孙）
//	从匹配选择的当前节点选择文档中的节点，而不考虑它们的位置，包含所有后代
.	选取当前节点
..	选取当前节点的父节点
@	选取属性

@ 属性

在DOM树，以路径的方式查询节点

通过 @ 符号来选取属性

```
<a rel="nofollow" class="external text" href="http://google.ac">google<wbr />.ac</a>
```

rel class href 都是属性，可以通过 "//*[@class='external text']" 来选取对应元素

= 符号要求属性完全匹配，可以用 contains 方法来部分匹配，例如 “//*[contains(@class, ‘external’)]” 可以匹配，而 "//*[@class='external']" 则不能

运算符

and 和 or 运算符：

选择 p 或者 span 或者 h1 标签的元素

```
soup = tree.xpath('//td[@class="editor bbsDetailContainer"]//*[self::p or self::span or self::h1]')
```

选择 class 为 editor 或者 tag 的元素

```
soup = tree.xpath('//td[@class="editor" or @class="tag"]')
```

正则表达式

简介

正则表达式是对字符串操作的一种逻辑公式，就是用事先定义好的一些特定字符、及这些特定字符的组合，组成一个“规则字符串”，这个“规则字符串”用来表达对字符串的一种过滤逻辑

在爬虫的解析中，经常会将正则表达式与 Dom 选择器结合使用。正则表达式适用于字符串特征比较明显的情况，但是同样的正则表达可能在HTML源码里多次出现；而 Dom 选择器可以通过 class 及 id 来精确找到 DOM 块，从而缩小查找的范围

常用规则

\	转移符，例如 \?
^	字符串起始
$	字符串结束
*	匹配前面子表达式0次或多次
+	匹配前面子表达式1次或多次
?	匹配前面子表达式0次或1次
{n,m}	匹配至少n次，最多m次
.	匹配除 \n 之外的单个字符
(pattern)	匹配并获取这个匹配，例如匹配ab(cd)e正则表达式只返回 cd
[xyz]	字符集合，匹配任意集合里的字符
[^xyz]	排除集合里的字符，不能匹配
\d	匹配一个数字，等价 [0-9]

爬虫常用的正则规则

获取标签下的文本

```
'<th[^>]*>(.*?)</th>'
```

查找特定类别的链接，例如/wiki/不包含 Category 目录：

```
'<a href="/wiki/(?!Category:)[^/>]*>(.*?)<'
```

查找商品外链，例如jd的商品外链为 7位数字的 a 标签节点：

```
'/\d{7}.html'
```

查找淘宝的商品信息，' 或者 " 开始及结尾

```
'href=[\"\']{1}(//detail.taobao.com/item.htm[^>\"\'\s]+?)"'
```

贪婪模式及非贪婪模式

? 该字符紧跟在任何一个其他限制符（*,+,?，{n}，{n,}，{n,m}）后面时，匹配模式是非贪婪的。非贪婪模式尽可能少的匹配所搜索的字符串，而默认的贪婪模式则尽可能多的匹配所搜索的字符串

```
<a
href='https://www.google.com/search?q=regular+expression&oq=regular+expression&
aqs=chrome..69i57.3239j0j7&sourceid=chrome&ie=UTF-8'>Google Test 1</a><a
href='https://www.google.com/search?q=regular+expression&oq=regular+expression&
aqs=chrome..69i57.3239j0j7&sourceid=chrome&ie=UTF-8'>Google Test 2</a>
```

对于上面的字符串，贪婪模式将匹配整个字符串，

```
re.findall('https://www.google.com/search\?q=.*UTF-8', s)
```

对于上面的字符串，而非贪婪模式才是我们想要的，只返回一个链接

```
re.findall('https://www.google.com/search\?q=.*?UTF-8', s)
```

动态网页

动态网页的使用场景

- 单页模式

  单页模式指的是不需要外部跳转的网页，例如个人设置中心经常就是单页

- 页面交互多的场景

  一部分网页上，有很多的用户交互接口，例如去哪儿的机票选择网页，用户可以反复修改查询的参数

- 内容及模块丰富的网页

  有些网页内容很丰富，一次加载完对服务器压力很大，而且这种方式延时也会很差；用户往往也不会查看所有内容

动态网页带来的挑战

对于爬虫：

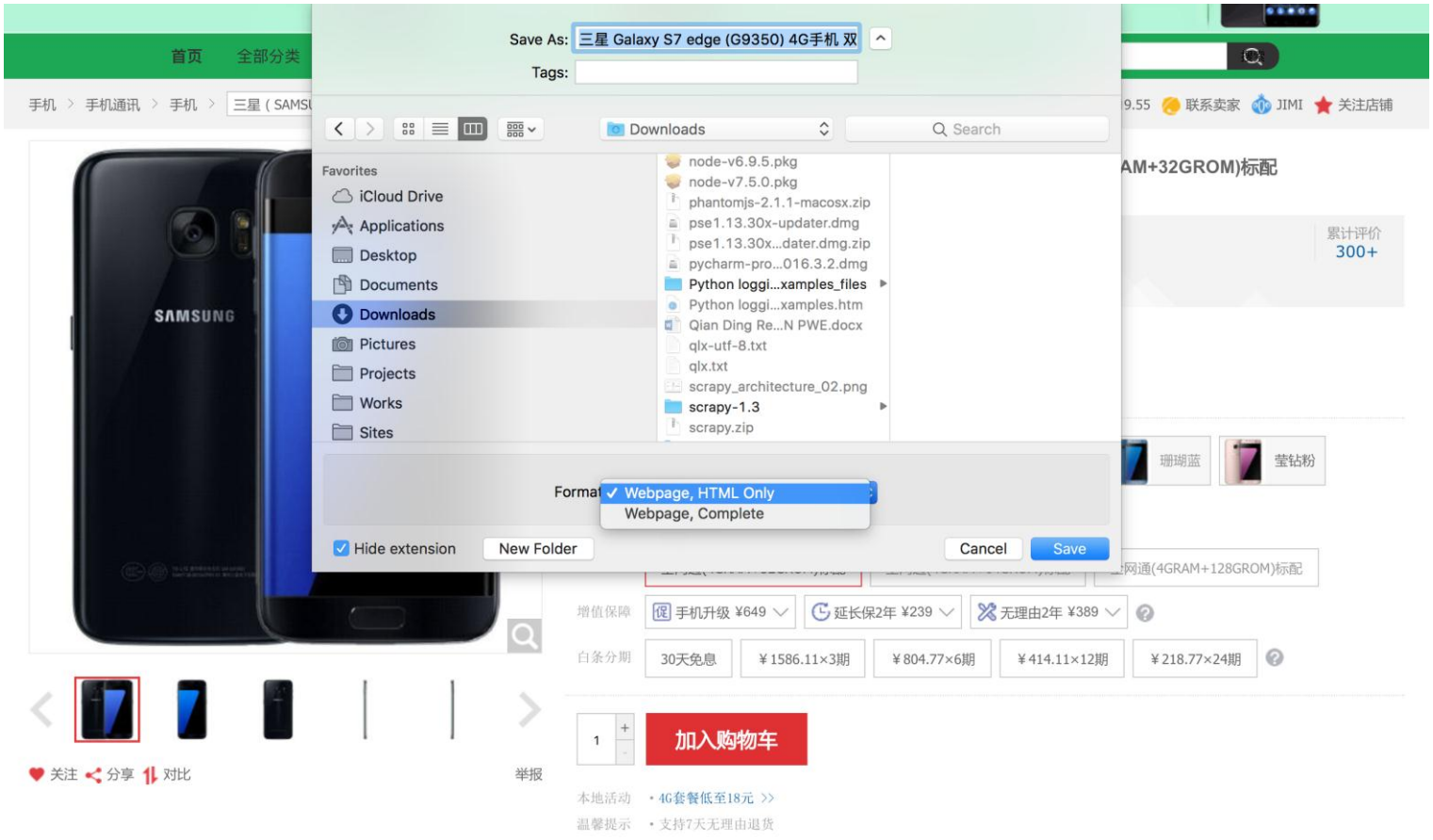
- 简单下载HMTL已经不行了，必须得有一个Web容器来运行HTML的脚本
- 增加了爬取的时间
- 增加了计算机的CPU、内存的资源消耗
- 增加了爬取的不确定性

对于网站：

- 为了配合搜索引擎的爬取，与搜索相关的信息会采用静态方式
- 与搜索无关的信息，例如商品的价格、评论，仍然会使用动态加载

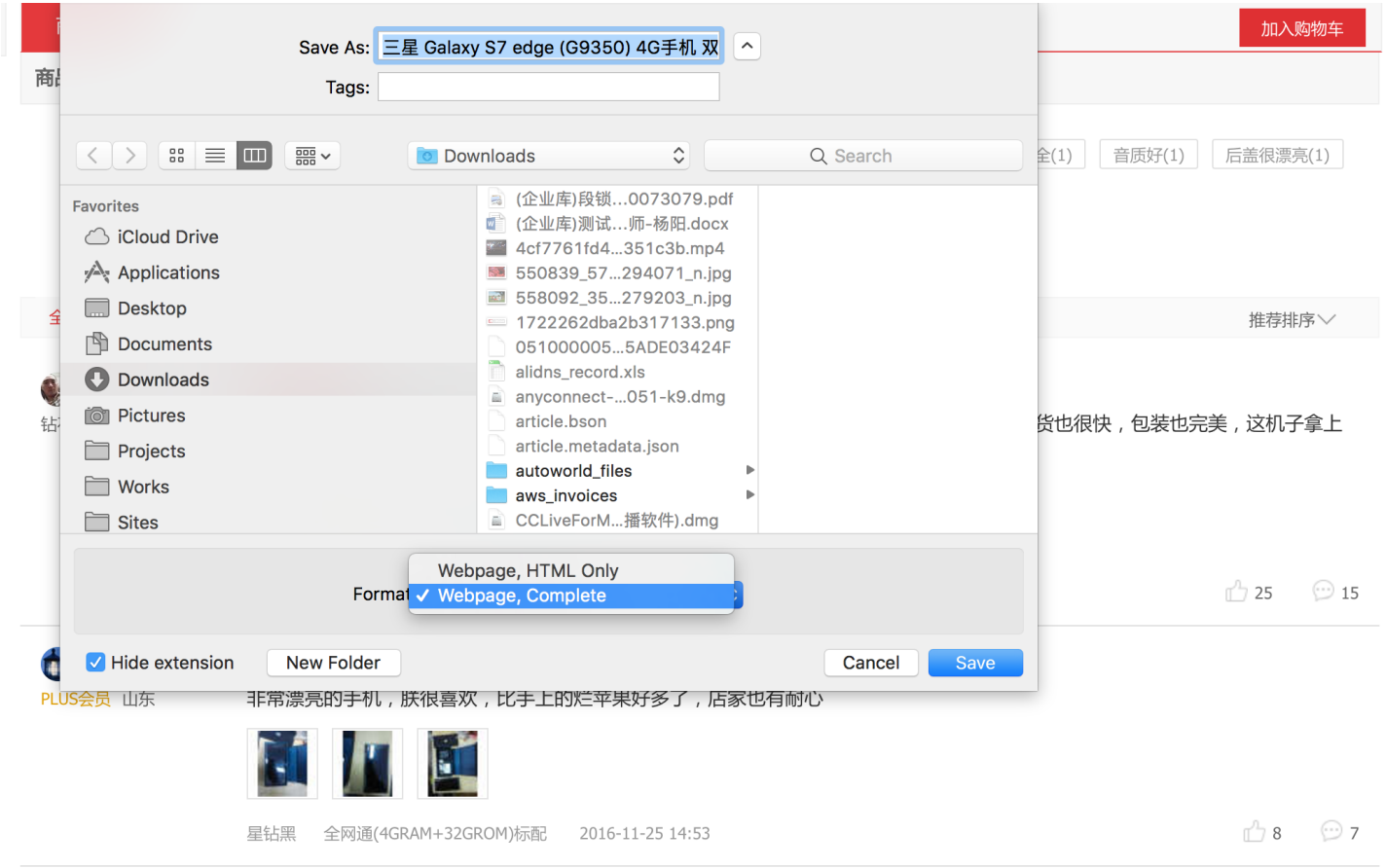
抓取动态网页 – 分析

打开目标网页后，直接右键点击，只保存HTML



分析动态网页 – 分析

把网页滑动到页面的最下面，然后再次保存，这次选择保存完整网页



分析动态网页 – 对比

- 使用 BeyondCompare 或者 SVN 等工具，来对比网页，大致找出动态加载的部分
- 针对要提取的部分，分别查看html only 与 full webpage，找出动态数据的部分
- 记录下它们的 class 或 id，试着用以下代码来提取，如果不能提取，说明是动态的：

```
from lxml import etree

f = open('./s7-full.htm')
c = f.read().decode('gbk')
f.close()

e = etree.HTML(c)
print e.xpath(u'//span[@class="price J-p-10524731933"]')
```

Python Web 引擎

- PyQt PySide：基于QT的python web引擎，需要图形界面的支持，需要安装大量的以来。安装和配置复杂，尤其是安装图形系统，对于服务器来说代价很大
- Selenium：一个自动化的Web测试工具，可以支持包括 Firefox、chrome、PhatomJS、IE 等多种浏览器的连接与测试
- PhantonJs：一个基于Webkit的 Headless 的Web引擎，支持JavaScript。相比 PyQt 等方案，phamtoms 可以部署在没有UI的服务器上

PhantomJS + Selenium

安装

Selenium

pip install selenium

PhantomJS

- PhantomJS 需要先安装 nodejs

```
# yum install nodejs
```

- 为了加速，将NPM的源改为国内的淘宝

```
$ npm install -g cnpm --registry=https://registry.npm.taobao.org
```

- 利用NPM的Package Manager 安装 phantomjs

```
$ npm -g install phantomjs-prebuilt
```

使用 PhantomJS 来加载动态页面

```
# import webdriver from selenium
from selenium import webdriver

# load PhantomJS driver
driver = webdriver.PhantomJS(service_args=['--ignore-ssl-errors=true'])
# set window size, better to fit the whole page in order to
# avoid dynamically loading data
driver.set_window_size(1280, 2400)  # optional
# data page content
driver.get(cur_url)
# use page_source to get html content
content = driver.page_source
```

set_window_size

对于动态网页，有可能存在大量数据是根据视图来动态加载的，PhantomJS 允许客户端设置用来模拟渲染页面的窗口的尺寸，这个尺寸如果设置比较小，我们就不得用 javascript 的 scroll 命令来模拟页面往下滑动的效果以显示更多内容，所以我们可以设置一个相对大的窗口高度来渲染

```
driver.set_window_size(1280, 2400)  # optional
```

Built-in DOM selector

Selenium 实现了一系列的类似于 xpath 选择器的方法，使得我们可以直接调用 driver. find_element() 来进行元素的选择，但是这些都是基于Python的实现，执行效率非常低，大约是基于C 的 正则表达式或 lxml 的10倍的时间，因此**不建议使用built-in的选择器，而是采用 lxml 或者 re 对 driver.page_source （html文本）进行操作**

find_element(self, by='id', value=None)
find_element_by_class_name(self, name)
find_element_by_id(self, id_)
find_element_by_css_selector(self, css_selector)

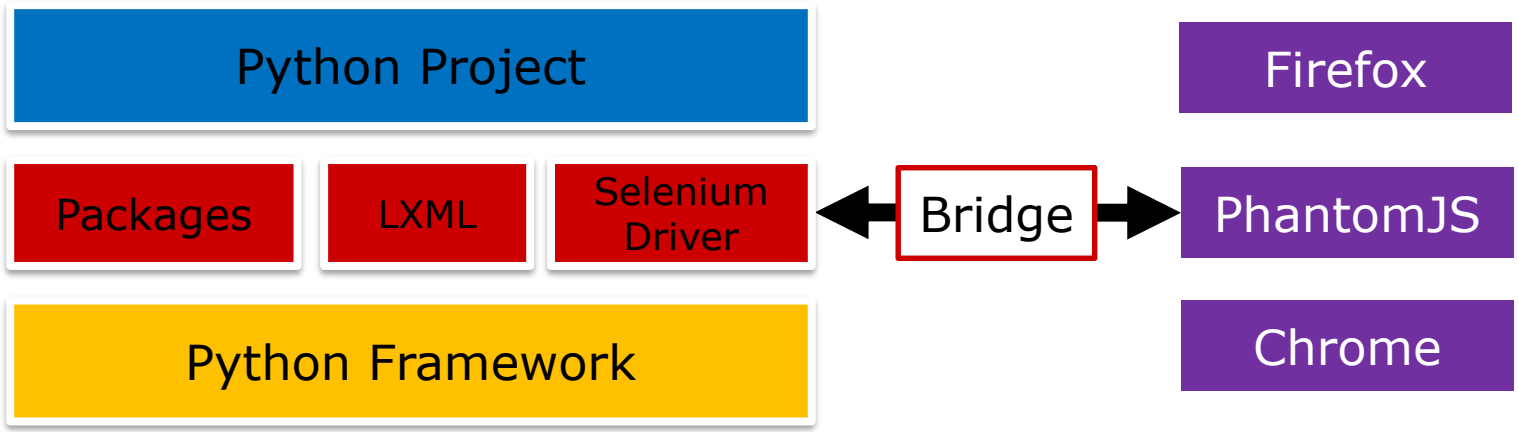
Useful Methods & Properties

Selenium 通过浏览器的驱动，支持大量的HTML及Javascript的操作，常用的可以包括：

- page_source: 获取当前的 html 文本
- title：HTML 的 title
- current_url：当前网页的URL
- get_cookie() & get_cookies()：获取当前的cookie
- delete_cookie() & delete_all_cookies()：删除所有的cookie
- add_cookie()：添加一段cookie
- set_page_load_timeout()：设置网页超时
- execute_script()：同步执行一段javascript命令
- execute_async_script()：异步执行javascript命令

Close and Clear

Selenium 通过内嵌的浏览器 driver 与浏览器进程通信，因此在退出的时候必须调用driver.close() 及 driver.quit() 来退出 PhantomJS，否则 PhantomJS 会一直运行在后台并占用系统资源。



Close and Clear

1. *send_signal is recommended*

   driver.service.process.send_signal(signal.SIGTERM)
2. *driver.close() this is not guaranteed to close PhantomJS*
3. *to assure it's closed, run below command in terminal*

   pgrep phantomjs | xargs kill

提取动态数据

1. 加载的过程中，根据网络环境的优劣，会存在一些延时，因此要多次尝试提取，提取不到不意味着数据不存在或者网络出错
2. 动态页面的元素，所使用的 id 或 class 经常会不止一个，例如京东一件商品的“好评率”，class 包括了 rate 和 percent-con 两种，因此需要对两种情况都进行尝试。更通用的情况，如果一个元素不能找到而 selenium 并没有报网络错误，那么有可能这个元素的class或id有了新的定义，我们需要将找不到的页面及元素信息记录在日志里，使得后续可以分析，找出新的定义并对这一类页面重新提取信息

提取动态数据

```
# try several times to extract data
while number_tried < constants['MAX_PAGE_TRIED']:
    try:
        price_element = driver.find_element_by_class_name('J-p-%s' % (item_id))
        price = re.findall('\d.*', price_element.get_attribute('innerHTML'))[0]
        item_name_element = driver.find_element_by_class_name('sku-name')
        item_name = item_name_element.text

        # item rating has 2 possible bound class, percent-con and rate
        # try to get rating values with both classes
        perc = re.findall('class="percent-con"', driver.page_source)
        if len(perc) > 0:
            element = driver.find_element_by_class_name('percent-con')
            rating = element.text
        else:
            element = driver.find_element_by_class_name('rate')
            rating = element.find_element_by_tag_name('strong').text
        break
    except selenium.common.exceptions.NoSuchElementException, msgs:
        number_tried += 1
        print msgs[1]
    except Exception, msgs:
        print msgs[1]
```

\ and \\

网页内，href 后面的链接可以有这样三种：

- **href="http://career.taobao.com"**

  http:// 是完整URL，直接跳转（经常外链会是绝对路径，比如引用到了WiKi、百科的一篇文章

- **href="\\detail.taobao.com\iuslkjsd"**

  // 是协议相关的绝对路径，如果现在是 https://xxx 则需要在 //etail.taobao.com 前面加上 https: 前面的协议也可以是 file:// ftp:// 所以这样会比较灵活

- **href="\i\8277375.html"**

  / 是网站的相对路径，需要在前面指明当前url的协议及domain

PhantomJS 配置

--ignore-ssl-errors=[true|false] ignores SSL errors, such as expired or self-signed certificate errors (default is false). Also accepted: [yes|no].

--load-images=[true|false] load all inlined images (default is true). Also accepted: [yes|no].

--disk-cache=[true|false] enables disk cache (at desktop services cache storage location, default is false). Also accepted: [yes|no].

--cookies-file=/path/to/cookies.txt specifies the file name to store the persistent Cookies.

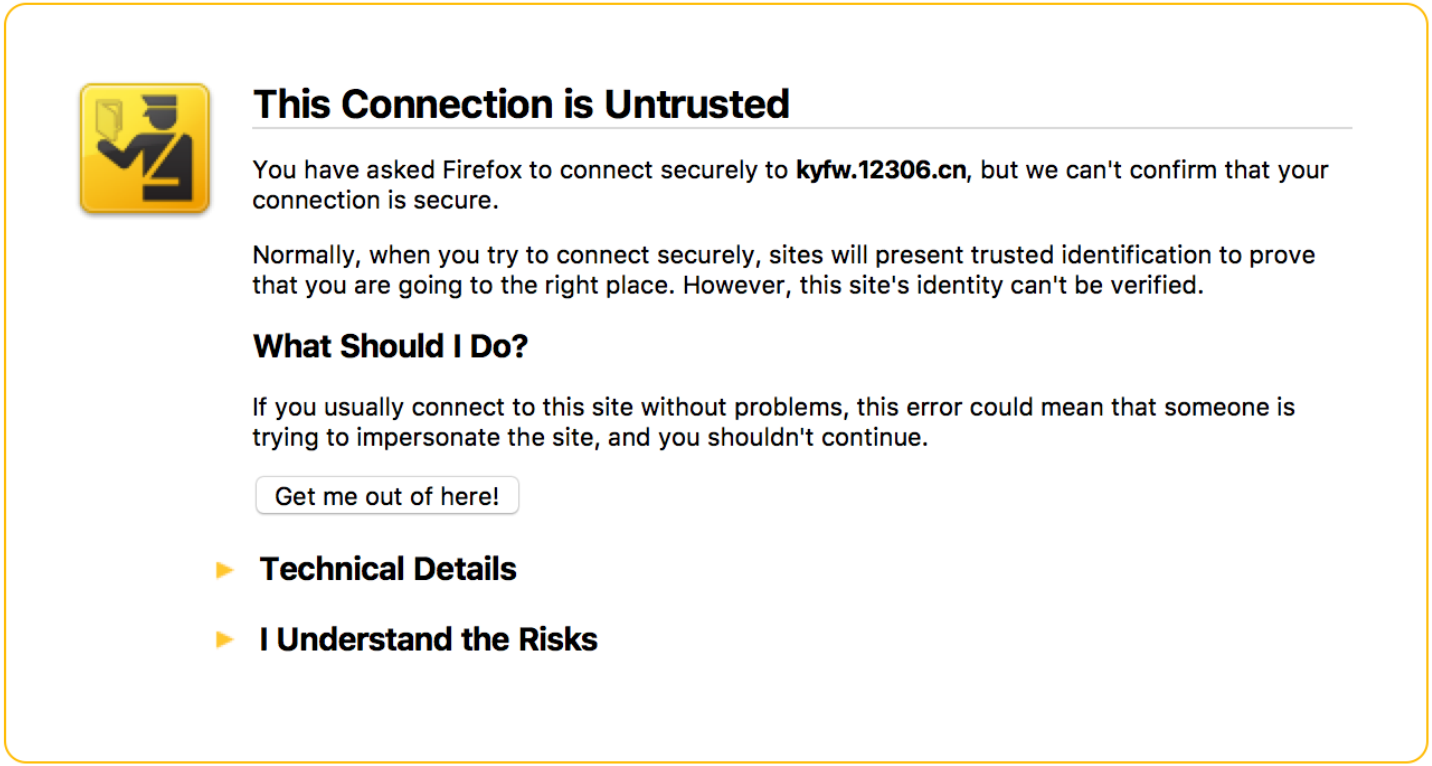
--debug=[true|false] prints additional warning and debug message, default is false. Also accepted: [yes|no].

--config specifies JSON-formatted configuration file (see below).

重要的配置-ignore-ssl-errors

--ignore-ssl-errors=[true|false]

一些证书没有获得CA授权（多是自己制作的证书），浏览器会报出证书不受信任，这种情况需要用户交互操作（点击继续或者新人），使用这个命令后，能自动忽略此类错误



重要的配置- load-images

--load-images=[true|false]

网页上一般都存在大量的图片，这些图片对我们第一次执行抓取是没有用的，在这种情况下，选择 --load-images=false 可以不下载这些图片，加快下载速度

重要的配置- config

--config=/path/to/config.json

```
{
    /* Same as: --ignore-ssl-errors=true */
    "ignoreSslErrors": true,
    /* Same as: --max-disk-cache-size=1000 */
    "maxDiskCacheSize": 1000,
    /* Same as: --output-encoding=utf8 */
    "outputEncoding": "utf8”
    /* etc. */
}
```

There are some keys that do not translate directly:

- --disk-cache => diskCacheEnabled
- --load-images => autoLoadImages
- --local-storage-path => offlineStoragePath
- --local-storage-quota => offlineStorageDefaultQuota
- --local-to-remote-url-access => localToRemoteUrlAccessEnabled
- --web-security => webSecurityEnabled
- --debug => printDebugMessages

疑问

- 问题答疑： http://www.xxwenda.com/
  - 可邀请老师或者其他人回答问题

联系我们

小象学院：互联网新技术在线教育领航者

- 微信公众号：小象
- 新浪微博：ChinaHadoop