# 等待元素出现

## 隐式等待

用WebDriver 对象的implicitly\_wait方法。 这个方法有一个参数，指明等待最长时间。

driver.implicitly\_wait(**10**)

## 显式等待

Selenium里面还有一种称之为显式等待的， 可以为一个操作专门指定等待时间

**from** selenium.webdriver.support.ui **import** WebDriverWait  
**from** selenium.webdriver.support **import** expected\_conditions **as** EC  
**from** selenium.webdriver.common.by **import** By  
ele = WebDriverWait(driver, **60**).until(EC.presence\_of\_element\_located((By.ID,**'1'**)))

比较长，我们可以顺着读一遍，很像自然语言.

大家不用强记，需要使用的时候，直接拷贝，修改红色部分就可以了。

# Frame切换

## frame介绍

例子，网页中包含网页：

<https://mail.qq.com/>

iframe 就是一个特殊的html 元素， 它在原来的html 范围内，开辟了一个新的HTML。

如果一个html文档结构比作一个国家的话， iframe就像 在一个国家里面建立一个 附属国家。 里面是一个全新的html文档。就像国中之国

W3C对frame的介绍

iframe:

<http://www.w3school.com.cn/tags/tag_iframe.asp>

frameset:

<http://www.w3school.com.cn/tags/tag_frameset.asp>

两者的区别是：

iframe 用于在网页内显示网页。和frameset不同的是，

frameset内置好几个子html

而iframe 只内置一个子html

## webdriver切换frame的方法

driver.switch\_to.frame(**frame\_reference**)

这个 frame\_reference 可以是

* frame 元素的 name 属性值 或者ID属性值：contentFrame（浏览器里面看一下）
* 索引值 （从0开始）：0 可能有人问，我咋知道索引？谁先出现就是第一个
* frame 所对应的WebElement ：driver.find\_element\_by\_tag\_name("iframe")

建议优先使用第一种，如果frame元素没有 name 属性，就用下面的两种方法。

比如元素有ID， 就可以根据 find element by id 找到webelement对象，然后， 就是第三种了。

从里面的frame切换回主HTML

driver.switch\_to.default\_content()

嵌套frame中如何逐层切换

一层一层的切换

driver.switch\_to.frame(**'layer2'**)  
driver.switch\_to.frame(**'layer3'**)

返回上一层frame

driver.switch\_to.parent\_frame()

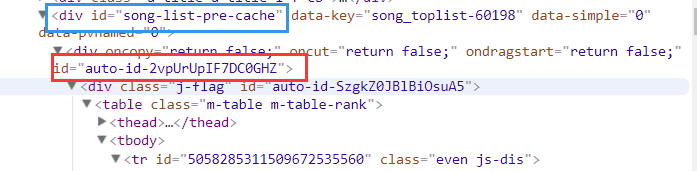
## 注意动态ID不能使用

前面我们找排行榜歌曲的时候，

f12显示，

寻找ID是找的这个蓝框中的， 为什么不找红框子中的呢？

比我们上面的 div 更贴近我们要找的table元素， 而且它也有id啊。？



很多前端开发框架会自动生成一些元素， 有动态生成的id 会变，

这个是 会变的id 要注意，不能用来选择

怎么识别是动态的呢？ 名字往往是一串随机字符，

而不会变的id值通常名字是有意义的单词。

# CSS选择器

## css选择器是什么

css 选择器 是浏览器用来选择元素的， 我们selenium 也要选择元素，也可以使用css 的选择器语法来选择 web 元素。 而且这种方法效率还非常的高。

## CSS选择器基本用法

根据 tag 名 选择

p {color: red;}

表示选择所有的 p 元素 ， 运行一下都变成红色了

如果改为

p {color: blue;} ， 运行一下

这里面 选择器就根据元素标签名。。。。起到了选择的作用

------

根据 id ，修改一下

#food {color: blue;} 表示选择ID为 food的 元素 ，

如果改为

#food2 {color: blue;} ， 运行一下

这里面 选择器就根据 id 。。。。起到了选择的作用

------

根据class 选择，修改一下

.special {color: red;}

有的元素 有两个class 值，

<**span** class=**"vegetable good"**>黄瓜</**span**>

.good {color: red;}

.vegetable {color: blue;} 表示选择所有的 class

为 vegetable 的元素 ，

注意

<**span** class=**"vegetable good"**>黄瓜</**span**>

表示这个元素 有两个class 值， 而不是 一个 **vegetable good** 整体的属性

这里面 选择器就根据 class。。。。起到了选择的作用

可以

.vegetable.good

----

当然也可以组合写 span.vegetable {color: blue;}

这样p.vegetable 就不会被选中

<**p** class=**"vegetable"**>青菜</**p**>

以上介绍的只是css选择器的冰山一角，更多内容移步到w3c有更多资料

后续课程中也会陆续讲到css的高级使用技巧

## 使用css方法选择元素

eles = driver.find\_elements\_by\_css\_selector(**'#choose\_car option'**)

## CSS后代选择器用法

语法 如下

<s1> <s2>

表示 选择s1 元素 里面 所有的s2元素， S2可以是S1的直接子节点，也可以不是

其中<s1> <s2>可以是我们前面学过的css 根据tag名、id 、 class 描述的任何语法

比如：

#choose\_car option

就是 选择 id为 choosecar 的元素的内部的 标签为option 的子元素。

到底是 不是直接子元素，不重要，

只要是 属于choose\_car 的内部的option元素就可以了。

# 附录

**元素等待：**

*# coding:utf8***from** selenium **import** webdriver  
  
  
driver = webdriver.Chrome(**r"d:\tools\webdrivers\chromedriver.exe"**)  
driver.implicitly\_wait(3)  
  
driver.get(**'http://www.baidu.com'**)  
  
element\_keyword = driver.find\_element\_by\_id(**"kw"**)  
element\_keyword.send\_keys(**'松勤'**)  
  
element\_search\_button = driver.find\_element\_by\_id(**"su"**)  
element\_search\_button.click()  
  
  
*# ================================***from** selenium.webdriver.support.ui **import** WebDriverWait  
**from** selenium.webdriver.support **import** expected\_conditions **as** EC  
**from** selenium.webdriver.common.by **import** By  
  
ele = WebDriverWait(driver,60). until(EC.presence\_of\_element\_located((By.ID,**'1'**)))  
*#显示等待只正对这一步操作，非全局  
  
  
# ele = driver.find\_element\_by\_id('1')*print(ele.text)  
  
**if** ele.text.startswith(**'松勤网 - 松勤软件测试\_软件测试在线培训'**):  
 print(**'pass'**)  
**else**:  
 print(**'fail'**)  
  
*# ================================*driver.quit()

**frame切换**

*# coding=utf-8***from** selenium **import** webdriver  
  
driver = webdriver.Chrome()  
  
driver.get(**'file:///C:/Users/Administrator/Dropbox/python\_autotest/autoUI\_selenium/iframe/nested.html'**)  
  
driver.switch\_to.frame(**'layer2'**)  
  
driver.find\_element\_by\_tag\_name(**'input'**).send\_keys(**'内1层'**)  
  
driver.switch\_to.frame(**'layer3'**)  
driver.find\_element\_by\_tag\_name(**'input'**).send\_keys(**'内2层'**)  
  
*#回到主HTML  
# driver.switch\_to.default\_content()#回到最外面的HTML  
  
#回到上层*driver.switch\_to.parent\_frame()  
ele=driver.find\_element\_by\_tag\_name(**'input'**)  
ele.clear()  
ele.send\_keys(**'内1层2次'**)  
  
  
  
*#回到主HTML  
# driver.switch\_to.default\_content()  
# driver.find\_element\_by\_id('button-layer1').click()  
# driver.find\_element\_by\_tag\_name('input').send\_keys('最外层')*input(**'press any key to quit...'**)  
driver.quit()

作业代码分享

**from** selenium **import** webdriver  
  
driver = webdriver.Chrome(**r"d:\tools\webdrivers\chromedriver.exe"**)  
  
*#访问百度音乐，找出排名上升的歌曲*driver.get(**'http://music.taihe.com/top/new'**)  
  
*#确定目标元素  
# ele=driver.find\_element\_by\_id('songListWrapper')  
# lis=ele.find\_elements\_by\_tag\_name('li')*lis=driver.find\_elements\_by\_css\_selector(**'#songListWrapper li'**)  
*# print(len(lis))  
# iss=ele.find\_elements\_by\_tag\_name('i')  
# print(len(iss))***for** li **in** lis:  
 i=li.find\_element\_by\_tag\_name(**'i'**)  
 **if** i.get\_attribute(**'class'**)==**'up'**:  
 *#如果是我们就把歌曲信息打印出来* songtitle=li.find\_element\_by\_class\_name(**'song-title '**).text  
 singer=li.find\_element\_by\_class\_name(**'singer'**).text  
  
 *#并且打印出歌曲名和歌手* print(**f'{songtitle}:{singer}'**)  
  
  
driver.quit()