Selenium(Python)试讲

[自动化测试类型 2](#_Toc486164550)

[什么样的项目适合自动化测试 2](#_Toc486164551)

[Selenium 工具介绍 3](#_Toc486164552)

[Selenium IDE 4](#_Toc486164553)

[Selenium Grid 5](#_Toc486164554)

[Selenium RC 5](#_Toc486164555)

[Selenium 2.0 5](#_Toc486164556)

[环境搭建 6](#_Toc486164557)

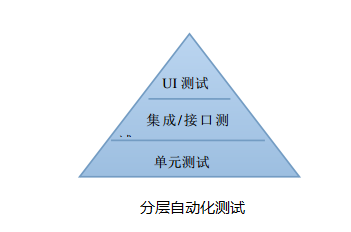
[定位元素 7](#_Toc486164558)

[HTML名词介绍 7](#_Toc486164559)

[八种元素定位方法介绍 8](#_Toc486164560)

## 自动化测试类型

传统的自动化测试更关注产品 UI 层的自动化测试，而分层的自动化测试倡导产品开发的不同阶段（层 次）都需要自动化测试。



**UI 层的自动化测试**：这个大家应该再熟悉不过了，大部分测试人员的大部分工作都是对 UI 层的功能 进行测试。例如，我们不断重复的对一个表单提交，结果查询等功能进行测试，我们可以通过相应的自动 化测试工具来模拟这些操作，从而解放重复的劳动。UI 层的自动化测试工具非常多，比较主流的是 QTP， Robot Framework、watir、Selenium 等。

在自动化测试中最怕的是变化，因为变化的直接结果就是导致测试用例的运行失败，那么就需要对自 动化脚本进行维护；如何控制失败，降低维护成本对自化的成败至关重要。反过来讲，一份永远都运行成 功的自动化测试用例是没有任何价值。

## 什么样的项目适合自动化测试

虽然，在你拿到这本书时已经对要测试的项目做了一些分析和考量，但笔者还是有必要在这里啰 嗦一下不是所有项目都适合实施自动化测试的，以免读者对项目实施自动化过程中感到困难重重，浪 费了大量的人力和时间而没有得到应有的收益。

1、任务测试明确，不会频繁变动

2、每日构建后的测试验证

3、比较频繁的回归测试

4、软件系统界面稳定，变动少

5、需要在多平台上运行的相同测试案例、组合遍历型的测试、大量的重复任务

6、软件维护周期长

7、项目进度压力不太大

8、被测软件系统开发比较规范，能够保证系统的可测试性

9、具备大量的自动化测试平台

10、测试人员具备较强的编程能力

当然，并非以上 10 条都具备的情况下才能开展测试工作。这里就需要读者做综合的权衡。在我 们普遍的自动化测试经验中，一般满足三个条件就可以对项目开展自动化测试：

**软件需求变动不频繁**

测试脚本的稳定性决定了自动化测试的维护成本。如果软件需求变动过于频繁，测试人员需要根 据变动的需求来更新测试用例以及相关的测试脚本，而脚本的维护本身就是一个开发代码的过程，需 要修改、调试，必要的时候还要修改自动化测试的框架，如果所花费的成本高于利用其节省的测试成 本，那么自动化测试便是失败的。

项目中的某些模块相对稳定，而某些模块需求变动性很大。我们便可对相对稳定的模块进行自动 化测试，而变动较大的仍是用手工测试。

**项目周期较长**

由于自动化测试需求的确定、自动化测试框架的设计、脚本的开发与调试均需要时间来完成。这 样的过程本身就是一个测试软件的开发过程。如果项目的周期比较短，没有足够的时间去支持这样一 个过程，那么自动化测试便成为笑谈。

**自动化测试脚本可重复使用**

自动化测试脚本的重复使用要从三个方面来考量，一方面所测试的项目之间是否很大的差异性（如 C/S 系统和 B/S 系统的差异）；所选择的测试工具是否适应这种差异；最后，测试人员是否有能力开发出 适应这种差异的自动化测试框架

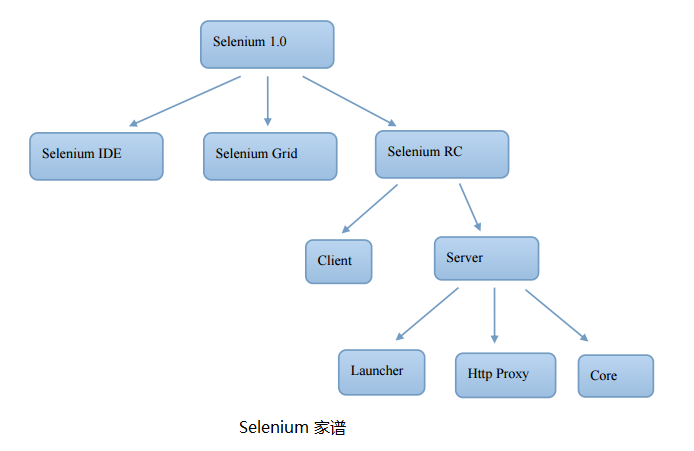
## Selenium 工具介绍

Selenium 自动化测试浏览器，它主要是用于 Web 应用程序的自动化测试，但肯定不只局限于此，同 时支持所有基于 web 的管理任务自动化。

Selenium 的特点：

* 开源，免费
* 多浏览器支持：FireFox、Chrome、IE、Opera
* 多平台支持：linux 、windows、MAC
* 多语言支持：java、Python、Ruby、php、C#、JavaScript
* 对 web 页面有良好的支持
* 简单（API 简单）、灵活（用开发语言驱动）
* 支持分布式测试用例执行

Selenium 经历了两个版本，Selenium 1.0 和 Selenium 2.0 ，Selenium 也不是简单一个工具，而 是由几个工具组成，每个工具都有其特点和应用场景。



### Selenium IDE

Selenium IDE 是嵌入到 Firefox 浏览器中的一个插件，实现简单的浏览器操作的录制与回放功能。那么什么情况下用到它呢？

快速的创建 bug 重现脚本，在测试人员的测试过程中，发现了 bug 之后可以通过 IDE 将重现的步骤录 制下来，以帮助开发人员更容易的重现 bug。

IDE 录制的脚本可以可以转换成多种语言，从而帮助我们快速的开发脚本，关于这个功能后而用到时 再详细介绍。

### Selenium Grid

Selenium Grid 是一种自动化的测试辅助工具，Grid 通过利用现有的计算机基础设施，能加快 Web-app 的功能测试。利用 Grid，可以很方便地同时在多台机器上和异构环境中并行运行多个测试事例。其特点为：

* 并行执行
* 通过一个主机统一控制用例在不同环境、不同浏览器下运行。
* 灵活添加变动测试机

### Selenium RC

Selenium RC 是 Selenium 家族的核心工具，Selenium RC 支持多种不同的语言编写自动化测试脚本， 通过 Selenium RC 的服务器作为代理服务器去访问应用从而达到测试的目的。

Selenium RC 使用分 Client Libraries 和 Selenium Server，Client Libraries 库主要主要用于编 写测试脚本，用来控制 Selenium Server 的库。

Selenium Server 负责控制浏览器行为，总的来说，Selenium Server 主要包括 3 个部分：Launcher、 Http Proxy、Core。其中 Selenium Core 是被 Selenium Server 嵌入到浏览器页面中的。其实 Selenium Core 就是一堆 JS 函数的集合，就是通过这些 JS 函数，我们才可以实现用程序对浏览器进行操作。Launcher 用 于启动浏览器，把 selnium Core 加载到浏览器页面当中，并把浏览器的代理设置为 Selenium Server 的 Http Proxy。

### Selenium 2.0

搞清了 Selenium 1.0 的家族关系，Selenium 2.0 是把 WebDriver 加入到了这个家族中；简单用公 式表示为：**Selenium 2.0 = Selenium 1.0 + WebDriver**

需要强调的是，在 Selenium 2.0 中主推的是 WebDriver ，WebDriver 是 Selenium RC 的替代品，因 为 Selenium 为了向下兼容性，所以 Selenium RC 并没有彻底抛弃，如果你使用 Selenium 开发一个新自 动化测试项目，强列推荐使用 WebDriver 。那么 Selenium RC 与 webdriver 主要有什么区别呢？

Selenium RC 在浏览器中运行 JavaScript 应用，使用浏览器内置的 JavaScript 翻译器来翻译和执行 selenese 命令（selenese 是 Selenium 命令集合）。

WebDriver 通过原生浏览器支持或者浏览器扩展直接控制浏览器。WebDriver 针对各个浏览器而开发，取代了嵌入到被测 Web 应用中的 JavaScript。与浏览器的紧密集成支持创建更高级的测试，避免了 JavaScript 安全模型导致的限制。除了来自浏览器厂商的支持，WebDriver 还利用操作系统级的调用模拟 用户输入。

Selenium 与 WebDriver 原先属于两个不同的项目，WebDriver 的创建者 Simon Stewart 早在 2009 年 八月的一份邮件中解释了项目合并的原因。

*Selenium 与 WebDriver 合并原因： 为何把两个项目合并？部分原因是 WebDriver 解决了 Selenium 存在的缺点（比如，能够绕过 JS 沙 箱。我们有出色的 API），部分原因是 Selenium 解决了 WebDriver 存在的问题（例如支持广泛的浏览 器），部分原因是因为 Selenium 的主要贡献者和我都觉得合并项目是为用户提供最优秀框架的最佳途 径。*

## 环境搭建

Python + Selenium环境搭建需要以下步骤：

* 安装python
* 安装setuptools
* 安装pip
* 安装selenium
* 配置webdriver
* 安装pycharm

详细步骤请参考附件：

## 定位元素

### HTML名词介绍

****

**（1）它们由标签对组成：**

<html></html>

<body></body>

<div></div>

<form></form>

html、div 就是标签的标签名，也就是Tag

**（2）标签各种属性属性：**

<div id="head" class="s\_down">

<from class="well">

<input id="kw" name="wd" class="s\_ipt">

就像人一样也会有各种属性，身份证号（id）、姓名（name）、职业（class）等。

**（3）标签对之间可以有文本数据。**

<a>新闻</a>

<a>hao123</a>

<a>地图</a>

**（4）标签有由层级关系**

<html>

<body>

</body>

</html>

<div>

<from>

<input />

</from>

<div>

对于上面结构，如果把 input 看作是子标签，那么 form 就是它的父标签。

### 八种元素定位方法介绍

webdriver 提供了八种元素定位方法：

|  |  |
| --- | --- |
| **八种元素定位方法** | **Python 语言中对应的定位方法** |
| id | find\_element\_by\_id() |
| name | find\_element\_by\_name() |
| class name | find\_element\_by\_class\_name() |
| tag name | find\_element\_by\_tag\_name() |
| link text | find\_element\_by\_link\_text() |
| partial link text | find\_element\_by\_partial\_link\_text() |
| xpath | find\_element\_by\_xpath() |
| css selector | find\_element\_by\_css\_selector() |

#### id 定位

name 如果把页面上看元素看作一个人的话，如果我们想找一个人如何去找，那么这个人一定有其别于

其它人的“属性”，比如他的身份证号一定和别人不一样，他的名字和别人不一样。那么我们就可以通过

身证号和名字来找到一个人。那么 id 就可以看做是一个人的身份号，当然这个 id 并不像我们现实中的身

份证号有那么强的唯一性，如果在一个页面上发现有两个元素的 id="kw"也是不足为奇的，这个取决前端

代码的规范程度。

对百度首页上的输入框与百度搜索按钮来说，定位方法如下：

find\_element\_by\_id("kw")

find\_element\_by\_id("su")

find\_element\_by\_id()方法用于元素中 id 属性的定位。

#### name 定位

name 的定位与 id 类似，每一个人都会有名字，那么 name 就可作是一个元素的名字。通过 name 定位

输入框：

find\_element\_by\_name("wd")

find\_element\_by\_name()方法用于元素中 name 属性的定位，百度搜索按钮并没有提供 name 属性，那

么我们就不能通过 name 去定位百度搜索按钮。

#### class 定位

class 也是不少元素会有的一个属性，它的定位和 name 以及 id 类似，下面通过 class 去定位百度输入

狂和百度搜索按钮：

find\_element\_by\_class\_name("s\_ipt")

find\_element\_by\_class\_name("bg s\_btn")

find\_element\_by\_class\_name()方法用于元素中 class 属性的定位。

#### tag 定位

tag 定位取的是一个元素的标签名，通过标签名去定位单个元素的唯一性最底，因为在一个页面中有太多的元素标签为<div>和<input>了，所以很难通过标签名去区分不同的元素。

通过标签名定位百度首页上的输入框与百度搜索按钮：

find\_element\_by\_tag\_name("input")

find\_element\_by\_tag\_name("input")

find\_element\_by\_tag\_name()方法通过元素的 tag name 来定位元素。通过上面的例子，我们并不能区别

不同的元素，因为在一个页面上标签名相同很难以避免。

#### link 定位

link 定位与前面介绍的几种定位方法有所不同，它专门用来定位本链接。百度输入框上面的几个文本

链接的代码如下：

###############################################################

<a class="mnav" name="tj\_trnews" href="http://news.baidu.com">新闻</a>

<a class="mnav" name="tj\_trhao123" href="http://www.hao123.com">hao123</a>

<a class="mnav" name="tj\_trmap" href="http://map.baidu.com">地图</a>

<a class="mnav" name="tj\_trvideo" href="http://v.baidu.com">视频</a>

<a class="mnav" name="tj\_trtieba" href="http://tieba.baidu.com">贴吧>

###############################################################

通过查看上面的代码，我们发现通过 name 属性定位是个不错的选择。不过我们这里为了要学习 link

定位，通过 link 定位实现如下：

find\_element\_by\_link\_text("新闻")

find\_element\_by\_link\_text("hao123")

find\_element\_by\_link\_text("地图")

find\_element\_by\_link\_text("视频")

find\_element\_by\_link\_text("贴吧")

find\_element\_by\_link\_text()方法通过元素标签对之间的文本信息来定位元素。不过，需要强调的是

Python 对于中文的支持并不好，如查 Python 在执行中文的地方出现在乱码，可以在中文件字符串的前面

加个小“u”可以有效的避免乱码的问题，加 u 的作用是把中文字符串转换中 unicode 编码，如：

find\_element\_by\_link\_text(u"新闻")

#### partial link 定位

parial link 定位是对 link 定们的一个种补充，有些文本连接会比较长，这个时候我们可以取文本链接的有一部分定位，只要这一部分信息可以唯一的标识这个链接。

###############################################################

**<a class="mnav" name="tj\_lang" href="#">一个很长很长的文本链接</a>**

###############################################################

通过 partial link 定位如下：

find\_element\_by\_partial\_link\_text("一个很长的")

find\_element\_by\_partial\_link\_text("文本连接")

find\_element\_by\_link\_text()方法通过元素标签对之间的部分文本信息来定位元素。

前面所介绍的几种定位方法相对来说比较简单，我们理想状态下在一个页面当中每一个元素都会有一

个唯一 id 和 name 属性值，我们通过它的属性值来找到他们，但在实际的项目中并非想象的这般美好。有

时候一个元素并没有 id 或 name 属性，或者会有多个元素的 id 和 name 属性值是一样的，又或者每一次刷

新页面，id 的值都会随机变化。那么在这种情况下我们如何来定位元素呢？

下面介绍 xpath 与 CSS 定位相比上面介绍的方式来说比较难理解，但他们的灵活的定位能力远比上

面的几种方式要强大得多。

#### XPath 定位

XPath 是一种在 XML 文档中定位元素的语言。因为 HTML 可以看做 XML 的一种实现，所以 selenium

用户可是使用这种强大语言在 web 应用中定位元素。

**绝对路径定位：**

XPath 有多种定位策略，最简单和直观的就是写元素的绝对路径。如果仍然把一个元素看做一个人的

话，那么现在有一个人，他没有任何属性特征，那么这个人一定会存在于某个地理位置，如：xx 省 xx 市

xx 区 xx 路 xx 号。那么对于一个元素在一个页面当中也会有这样的一个绝对地址。

参考 baidu.html 前端工具所展示的代码，我们可以用下面的方式来找到百度输入框和搜索按钮。

find\_element\_by\_xpath("/html/body/div/div[2]/div/div/div/from/span/input")

find\_element\_by\_xpath("/html/body/div/div[2]/div/div/div/from/span[2]/input")

find\_element\_by\_xpath()方法用于 XPath 语言定位元素。XPath 的绝对路径主要用标签名的层级关系来

定位元素的绝对路径。最外层为 html 语言，body 文本内，一级一级往下查找，如果一个层级下有多个相

同的标签名，那么就按上下顺序确定是第几个，div[2]表示第二个 div 标签。

**利用元素属性定位：**

除了使用绝对路径的以外，XPath 也可以使用使素的属性值来定位。同样以百度输入框和搜索按钮为

例了：

find\_element\_by\_xpath("//input[@id='kw']")

find\_element\_by\_xpath("//input[@id='su']")

//表示当前页面某个目录下，input 表示定位元素的标签名，[@id='kw'] 表示这个元素的 id 属性值等于

kw。下面通过 name 和 class 属性值来定位。

find\_element\_by\_xpath("//input[@id='wd']")

find\_element\_by\_xpath("//input[@class='s\_ipt']")

find\_element\_by\_xpath("//\*[@class='bg s\_btn']")

如果不想指定标签名也可以用星号（\*）代替。当然，使用 XPath 不仅仅只局限在 id、name 和 class 这

三个属性值，元素的任意属性值都可以使用，只要它能唯一的标识一个元素。

find\_element\_by\_xpath("//input[@maxlength='100']")

find\_element\_by\_xpath("//input[@autocomplete='off']")

find\_element\_by\_xpath("//input[@type='submit']")

**层级与属性结合：**

如果一个元素本身并没有可以唯一标识这个元素的属性值，我们可以找其上一级元素，如果它的上级

有可以唯一标识属性的值，也可以拿来使用。参考 baidu.html 文本。

如果一个元素本身并没有可以唯一标识这个元素的属性值，我们可以找其上一级元素，如果它的上级 有可以唯一标识属性的值，也可以拿来使用。



假如百度输入框本身没有可利用的属性值，我们可以查找它的上一级属性。比如，“小明”刚出生的

时候没有名字，没上户口（没身份证号），那么亲朋好友来找“小明”可以先到小明的爸爸，因为他爸爸

是有很多属性特征的，找到了小明的爸爸，抱在怀里的一定就是小明了。通过 XPath 描述如下：

find\_element\_by\_xpath("//span[@class='bg s\_ipt\_wr']/input")

find\_element\_by\_xpath("//span[@class='bg s\_btn\_wr']/input")

span[@class='bg s\_ipt\_wr'] 通过 class 属性定位到是父元素，后面/input 也就表示父元素下面标签名为

input 的子元素。如果父元素没有可利用的属性值，那么可以继续向上查找“爷爷”元素。

find\_element\_by\_xpath("//form[@id='form']/span/input")

find\_element\_by\_xpath("//form[@id='form']/span[2]/input")

我们可以通过这种方法一级一级的向上打找，直到找到最外层的<html>标签，那么就是一个绝对路径的写法了。

**使用逻辑运算符**

如果一个属性不能唯一的区分一个元素，我们还可以使用逻辑运算符连接多个属性来区别于其它属性。###############################################################

<input id="kw" class="su" name="ie">

<input id="kw" class="aa" name="ie">

<input id="bb" class="su" name="ie">

###############################################################

如上面的三行元素，假如我们现在要定位第一行元素，如果使用 id 将会与第二行元素重名，如果使用

class 将会与第三行元素的重名。那么如果同时使用 id 和 class 就会唯一的标识这个元素。那么这个时候就

可以通过逻辑运算符号连接。

find\_element\_by\_xpath("//input[@id='kw' and @class='su']/span/input")

当然，我们也可以用 and 连接更多的属性来唯一的标识一个元素。

我们在本书的第一章中介绍的 Firebug 前端调试工具和 FirePath 插件可以方便的辅助 XPath 语法。

打开 FireFox 浏览器的 FireBug 插件，点击插件左上角的鼠标箭头，再点击页面上需要定位的元素，在元

素行上右键弹出快捷菜单，选择“复制 XPath”，将会获得当前元素的 XPath 语法。

#### CSS 定位

CSS(Cascading Style Sheets)是一种语言，它被用来描述 HTML 和 XML 文档的表现。CSS 使用选择器

来为页面元素绑定属性。这些选择器可以被 selenium 用作另外的定位策略。

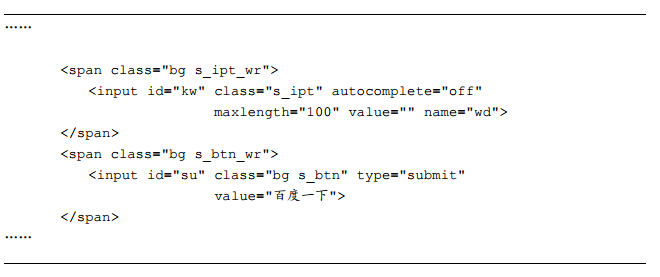
CSS 可以比较灵活选择控件的任意属性，一般情况下定位速度要比 XPath 快，但对于初学者来说比较

难以学习使用，下面我们就详细的介绍 CSS 的语法与使用。

CSS 选择器的常见语法：

|  |  |  |
| --- | --- | --- |
| **选择器** | **例子** | **描述** |
| .class | .intro | class 选择器，选择 class="intro"的所有元素 |
| #id | #firstname | id 选择器，选择所有 id="firstname"所有元素 |
| \* | \* | 选择所有元素 |
| element | p | 元素所有<p>元素 |
| element > element | div > input | 选择父元素为 <div> 元素的所有 <input> 元素 |
| element + element | div + input | 选择紧接在 <div> 元素之后的所有 <p> 元素。 |
| [attribute=value] | [target=\_blank] | 选择 target="\_blank" 的所有元素。 |

下面同样以百度输入框和搜索按钮为例介绍 CSS 定位的用法。



**通过 class 属性定位：**

find\_element\_by\_css\_selector(".s\_ipt")

find\_element\_by\_css\_selector(".bg s\_btn")

find\_element\_by\_css\_selector()方法用于 CSS 语言定位元素，点号（.）表示通过 class 属性来定位元素。

**通过 id 属性定位：**

find\_element\_by\_css\_selector("#kw")

find\_element\_by\_css\_selector("#su")

井号（#）表示通过 id 属性来定位元素。

**通过标签名定位：**

find\_element\_by\_css\_selector("input")

在 CSS 语言中用标签名定位元素不需要任何符号标识，直接使用标签名即可，但我们前面已经了解到

标签名重复的概率非常大，所以通过这种方式很难唯一的标识一个元素。

**通过父子关系定位：**

find\_element\_by\_css\_selector("span>input")

上面的写法表示有父亲元素，它的标签名叫 span，查找它的所有标签名叫 input 的子元素。

**通过属性定位：**

find\_element\_by\_css\_selector("input[autocomplete='off']")

fnd\_element\_by\_css\_selector("input[maxlength='100']")

find\_element\_by\_css\_selector("input[type='submit']")

在 CSS 当中也可以使用元素的任意属性，只要这些属性可以唯一的标识这个元素。

**组合定位：**

我们当然可以把上面的定位策略组合起来使用，这样就大大加强了元素的唯一性。

find\_element\_by\_css\_selector("span.bg s\_ipt\_wr>input.s\_ipt")

find\_element\_by\_css\_selector("span.bg s\_btn\_wr>input#su")

有一个父元素，它的标签名叫 span，它有一个 class 属性值叫 bg s\_ipt\_wr，它有一个子元素，标签名

叫 input，并且这个子元素的 class 属性值叫 s\_ipt。好吧！我们要找的就是具有这么多特征的一个子元素。

我们通过可以使用 Firebug 工具帮助我们生成 CSS 语法。通过 Firebug 定位元素，在元素上右键点击

选择“复制 CSS”。

#### XPath 与 CSS 的类似功能的简单对比

|  |  |  |
| --- | --- | --- |
| **定位方法** | **Xpath** | **CSS** |
| 标签 | //div | div |
| By id | //div[@id='eleid'] | div#eleid |
| By class | //div[@class='eleclass'] | div#eleid |
| By 属性 | //div[@title='Move mouse here'] | div[title=Move mouse here] div[title^=Move] div[title$=here] div[title\*=mouse] |
| 定位子元素 | //div[@id='eleid']/\* //div/h1 | div#eleid>\* div#eleid >h1 |

通过前面的学习 XPath 和 CSS 都提供了非常强大和灵活的定位方法。相比较 CSS 语法更加简洁，但

真正的理解和使用学难度要更大一点。按照作都的经验这两种定位方式我们只要掌握一种就可解决大部分

定位问题了，至于读者的选择就看个人喜好了。

#### 用 By 定位元素

有时需要使用定位方法，在具体通过哪种定位方式（id 或 name）根据实际场景而定位，By 就可以设

置定位策略。

find\_element(By.ID,"kw")

find\_element(By.NAME,"wd")

find\_element(By.CLASS\_NAME,"s\_ipt")

find\_element(By.TAG\_NAME,"input")

find\_element(By.LINK\_TEXT,u"新闻")

find\_element(By.PARTIAL\_LINK\_TEXT,u"新")

find\_element(By.XPATH,"//\*[@class='bg s\_btn']")

find\_element(By.CSS\_SELECTOR,"span.bg s\_btn\_wr>input#su")

find\_element()方法只用于定位元素。它需要两个参数，第一个参数是定位方式，这个由 By 提供；另

第二个参数是定位的值。在使用 By 时需要将 By 类导入。

from selenium.webdriver.common.by import By