


Python+selenium入门

目 录

-  - Python介绍

-  - 测试环境搭建

-  - WebDriver API

Python 介绍

Python来源

- Python是著名的“龟叔” Guido van Rossum在1989年圣诞节期间，为了打发无聊的圣诞节而编写的一个编程语言，1991 年正式公布。
- 龟叔给Python的定位是“优雅”、“明确”、“简单”，所以Python程序看上去总是简单易懂，初学者学Python，不但入门容易，而且将来深入下去，可以编写那些非常非常复杂的程序。
- Python的官网：www.python.org

Python的优缺点

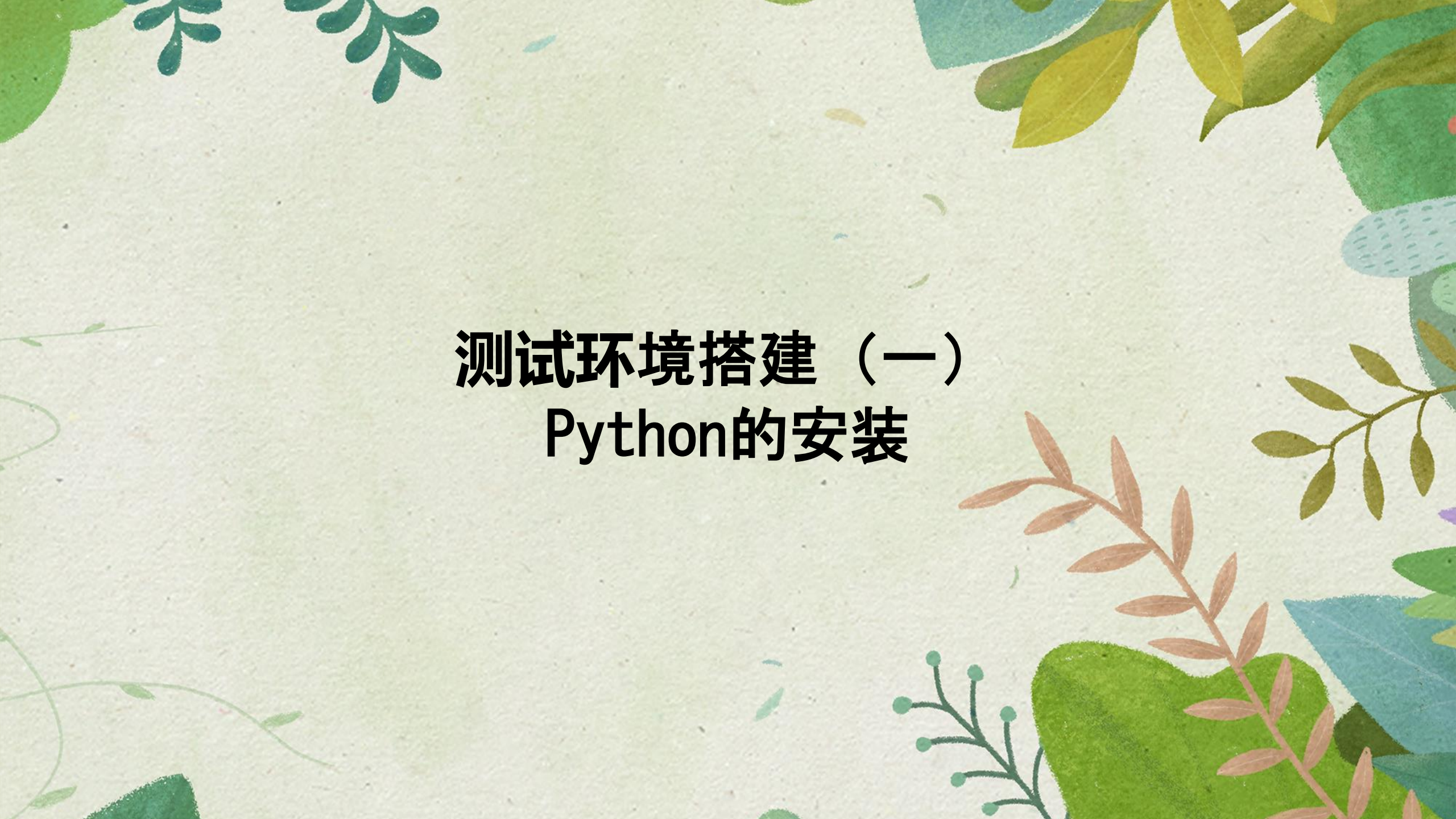
- 可读性强：Python 采用强制缩进的方式使得代码具有极佳的可读性
- 简洁：Python 是由 C 语言开发，但是不再有 C 语言中指针等复杂数据类型，Python 的简洁性让开发难度和代码幅度大幅降低，开发任务大大简化。免费、开源
- 可移植性：由于它的开源本质，Python 已经被移植在许多平台上（经过改动使它能够工作在不同平台上）。
- 解释性：Python 语言写的程序不需要编译成二进制代码，你可以直接从源代码运行程序。在计算机内部，Python 解释器把源代码转换成称为字节码的中间形式，然后再把它翻译成计算机使用的机器语言并运行。
- 可扩展性：你可以把Python嵌入你的C/C++程序，从而向你的程序用户提供脚本功能。
- 丰富的库：丰富的标准库， 多种多样的扩展库。
- 运行速度慢：因此对于性能要求比较高的部分一般使用一 C/C++/JAVA去开发。
- 代码不能加密：它的源码都是以明文形式存放的。
- 线程不能利用多CPU问题，这是Python被人诟病最多的一个缺点。

Python的应用领域

- 科学计算
- 人工智能
- **WEB** 服务端和大型网站后端。 YouTube、gmail 等应用基于 python 开发
- **GUI** 开发（图形用户界面开发）
- 游戏开发
- 移动设备
- 嵌入式设备
- 系统运维
- 大数据
- 云计算

Python的版本

- 目前主要两个版本：Python2 和 Python3 。
- Python2：2000年10月发布，最新版本是 2.7，已经停止更新，不会再有 2.8 以后了。预计 2020 年退出历史舞台。
- Python3：2008 年发布，Python3 有了较大的提升，但是不兼容 Python2。
- 因此建议从python3开始学



测试环境搭建（一） Python的安装

01

下载Python安装包

- 官网: <https://www.Python.org/>

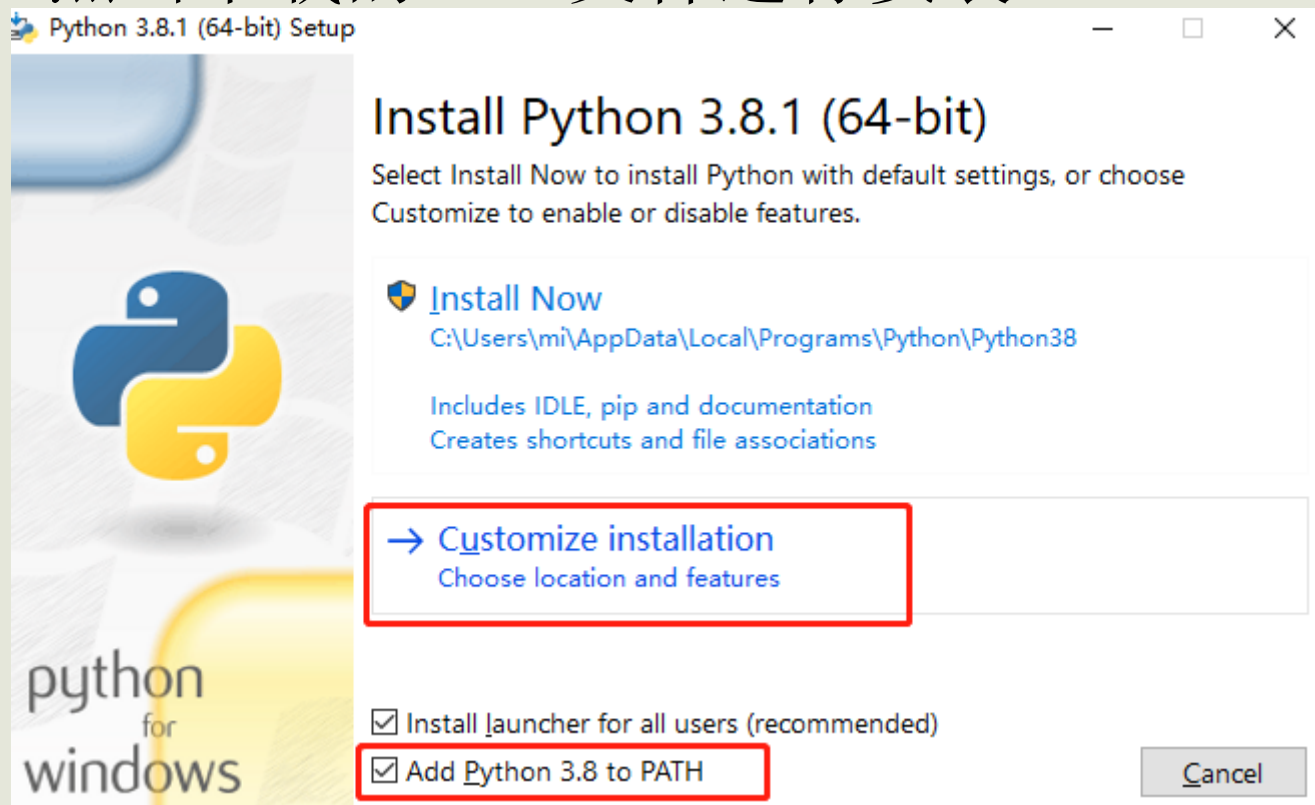


Version	Operating System	Description
Gzipped source tarball	Source release	
XZ compressed source tarball	Source release	
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later
Windows help file	Windows	
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64
Windows x86 embeddable zip file	Windows	
Windows x86 executable installer	Windows	
Windows x86 web-based installer	Windows	

- 如图示在官网选择最新的Python3版本下载
- 如果是32位系统, 选择X86版本
- 如果是64位系统, 选择X86-64版本
- 如果不知道是多少位, 请右击“我的电脑”-选择属性进行查看

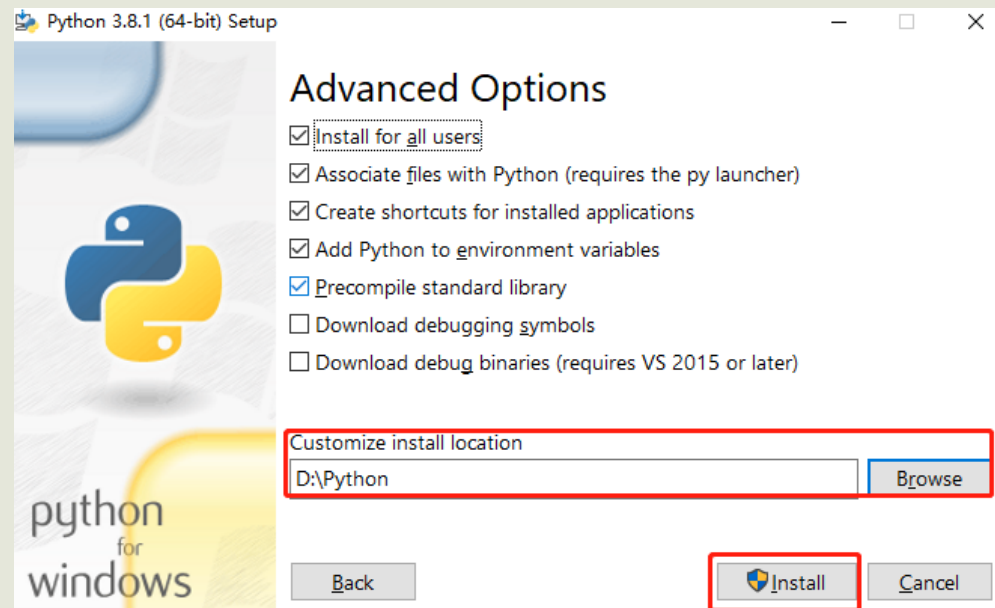
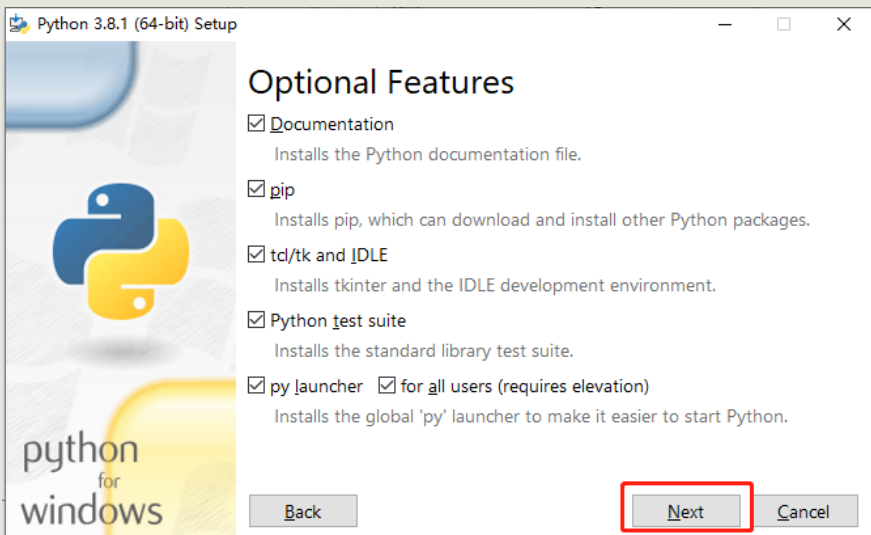
安装Python

- 点击下载的.exe文件进行安装



注意：

- 1、一般采用自定义安装比较好，不然自动安装会把环境安装到个人目录里，有时要找起来很麻烦
- 2、第二个红框前面的勾记录打上，不然需要手动在windows环境变量的路径中添加python安装路径



- 点击next
- 红框位置可以自定义安装路径
- 点击install
- 安装完成点击close



CMD查看是否安装成功

- 打开cmd窗口，输入Python，回车。显示以下内容，表示安装成功

```
C:\Users\mi>python
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 23:11:46) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
```

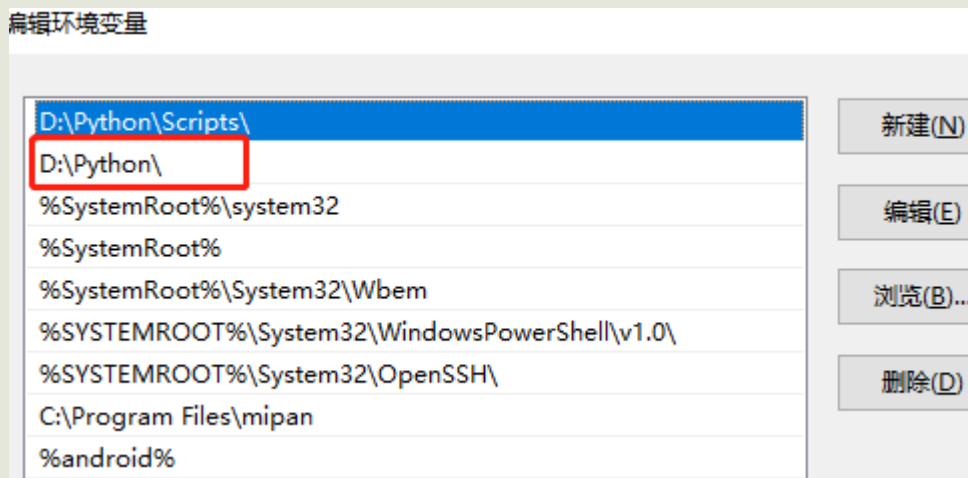
- 出现以下内容，需要手动将python安装路径添加到系统环境变量中


```
C:\Users\Empty>python
'python' 不是内部或外部命令，也不是可运行的程序
或批处理文件。

C:\Users\Empty>_
```


手动配置环境变量

- 右键“我的电脑”-选择属性-高级系统设置-环境变量
- 在系统变量中添加：
变量：Path
变量值：自定义的安装路径





测试环境搭建（二） PyCharm开发环境的使用

01

下载PyCharm安装包

- 官网下载地址：
<http://www.jetbrains.com/pycharm/download/#section=windows>
- 选择社区版本下载，免费

Download PyCharm

Windows Mac Linux

Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

Download

Free trial

Community

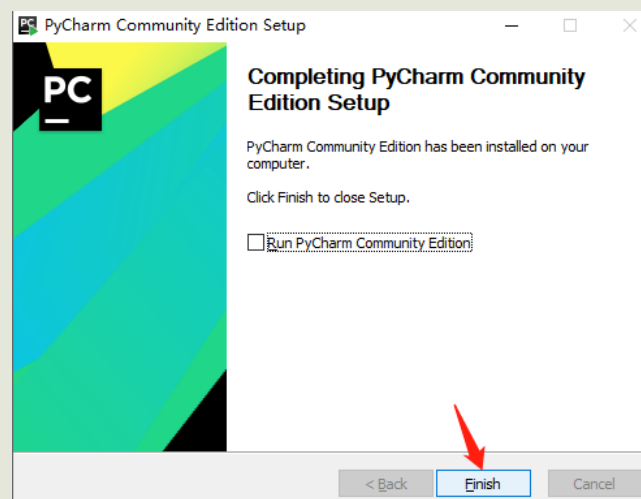
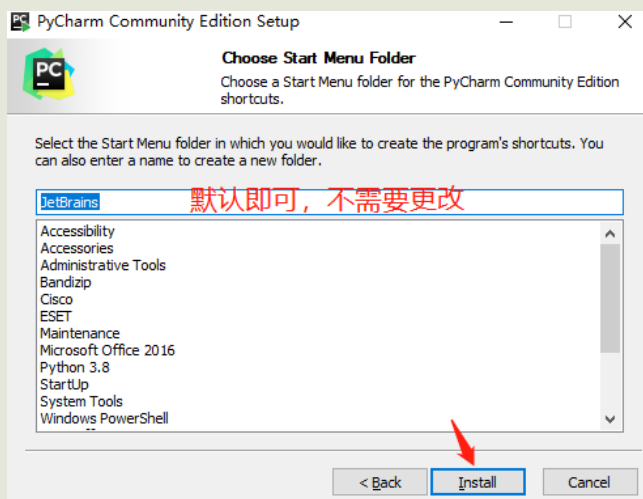
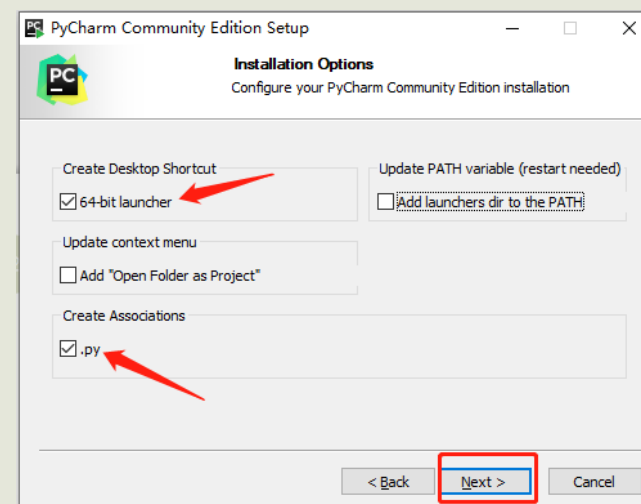
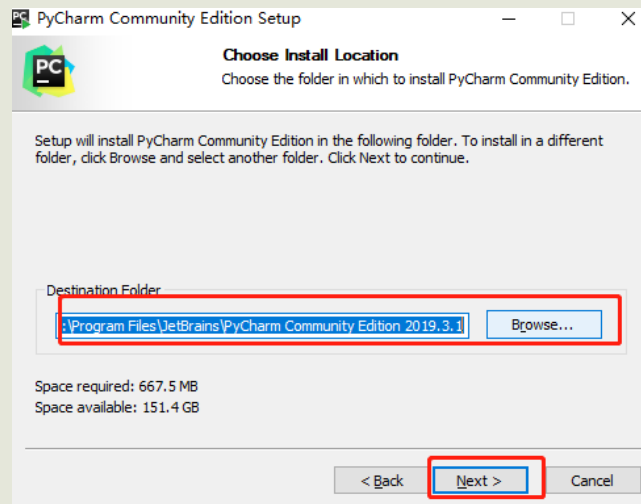
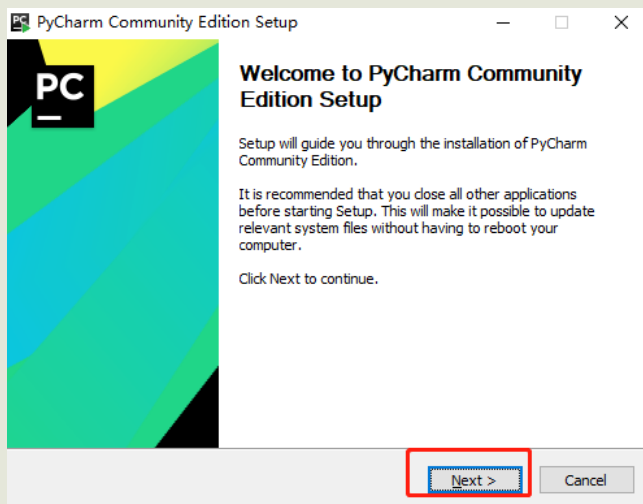
For pure Python development

Download

Free, open-source

02

安装PyCharm

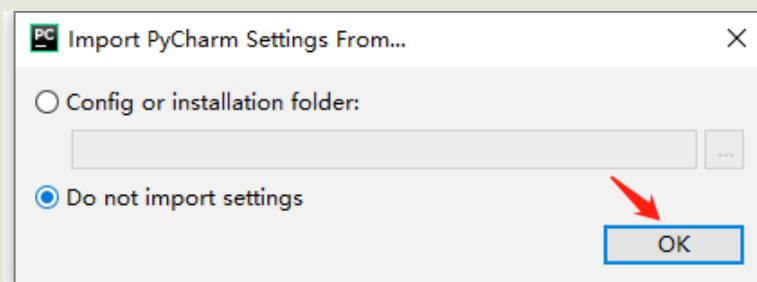


注意：
如图示从上到下，从左到右操作即可。

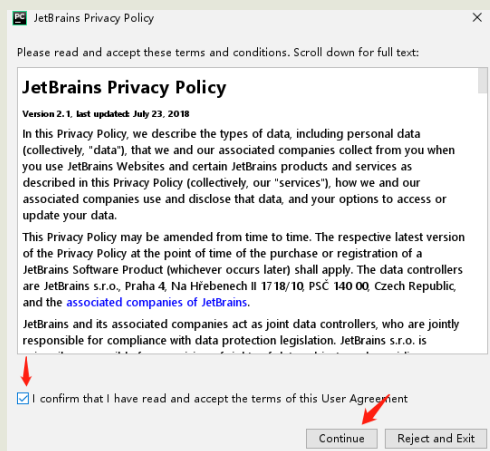
03

PyCharm配置

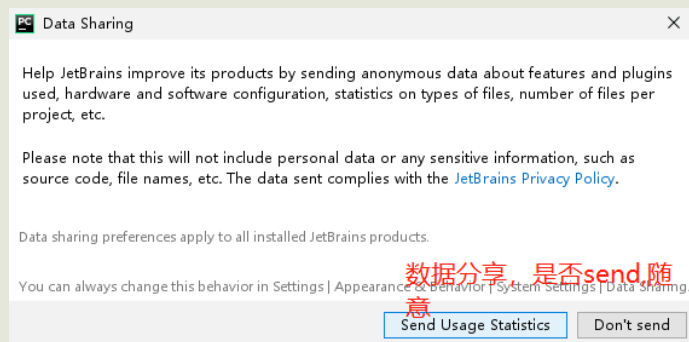
1. 导入pycharm设置，选不导入即可



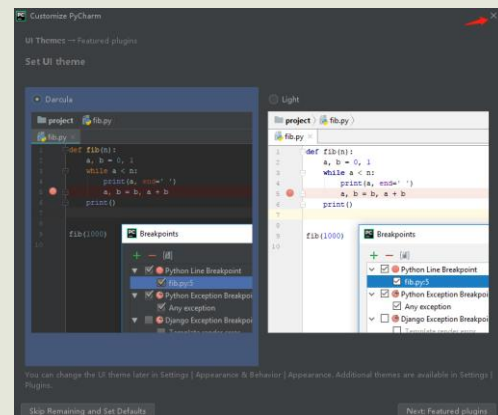
2. 勾选用户协议



3. 数据分享，随意

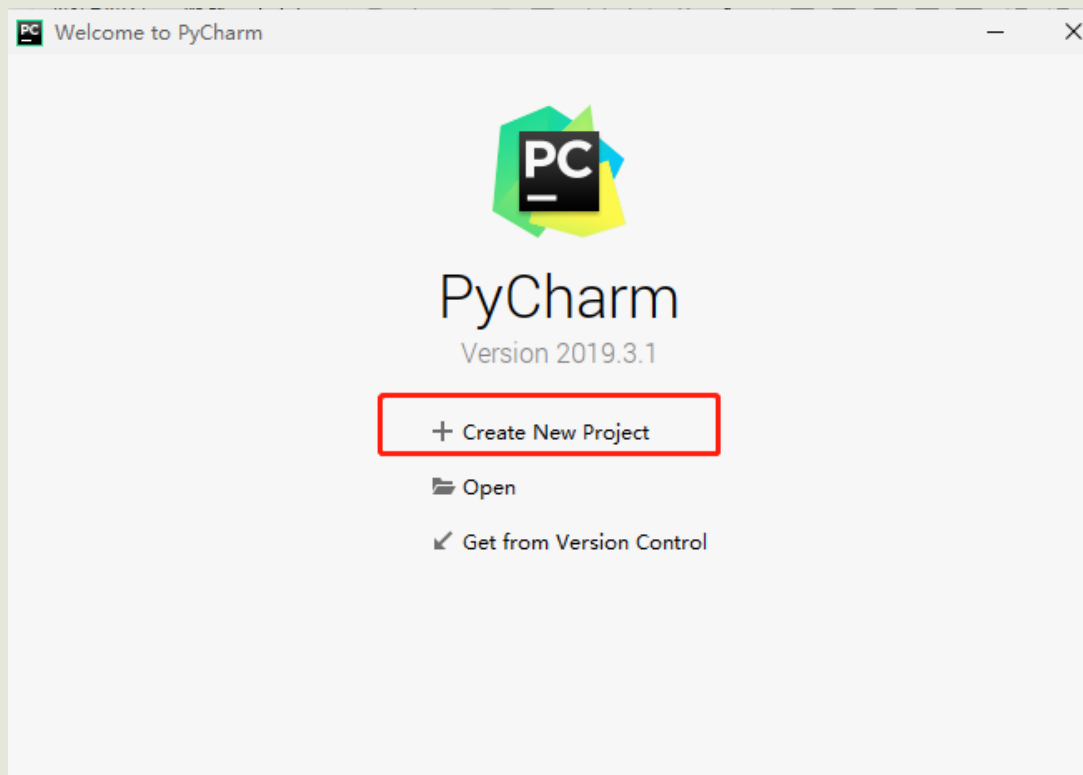


4. 皮肤选择，根据个人喜好选，点击右上角“X”即可

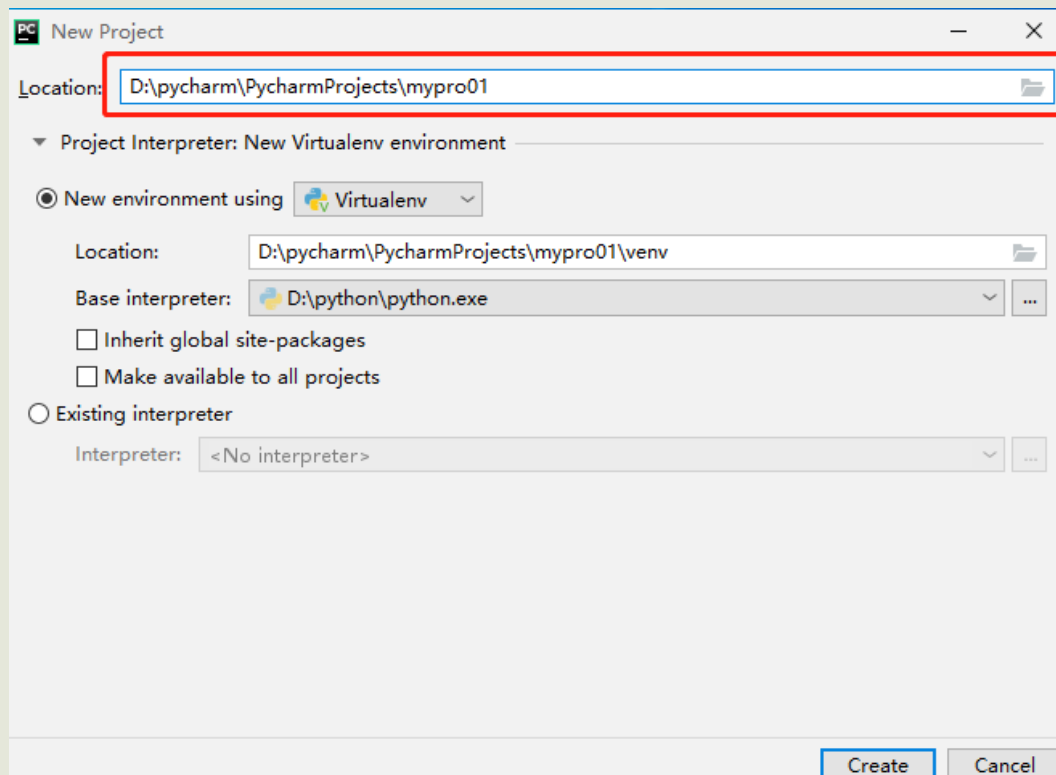


创建项目和初始配置

1. 选择：“Create New Project”



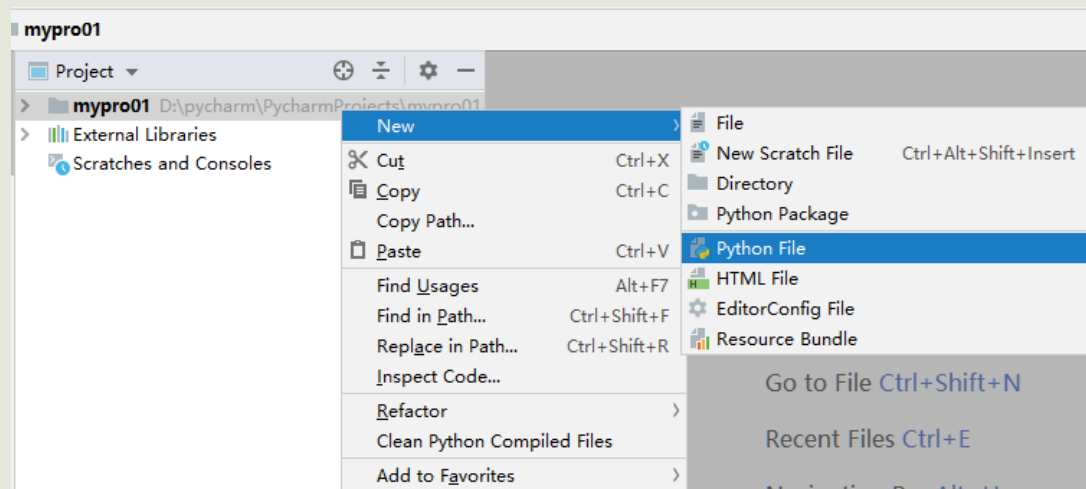
2. 选择路径（尽量不要包含中文），文件名：mypro01 就是我们的项目名称，可以修改。



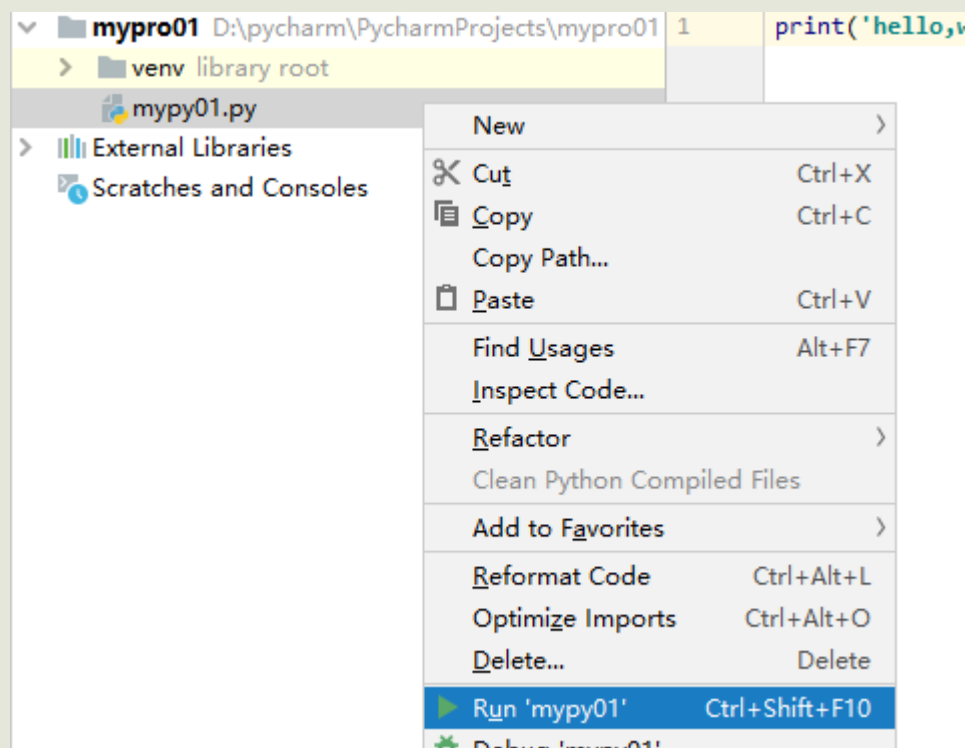
05


Python文件的创建和运行

1. 创建.py文件：右击如图创建.py文件并取名



2. 运行：右击文件名选择run





测试环境搭建（三） selenium的安装

pip命令安装

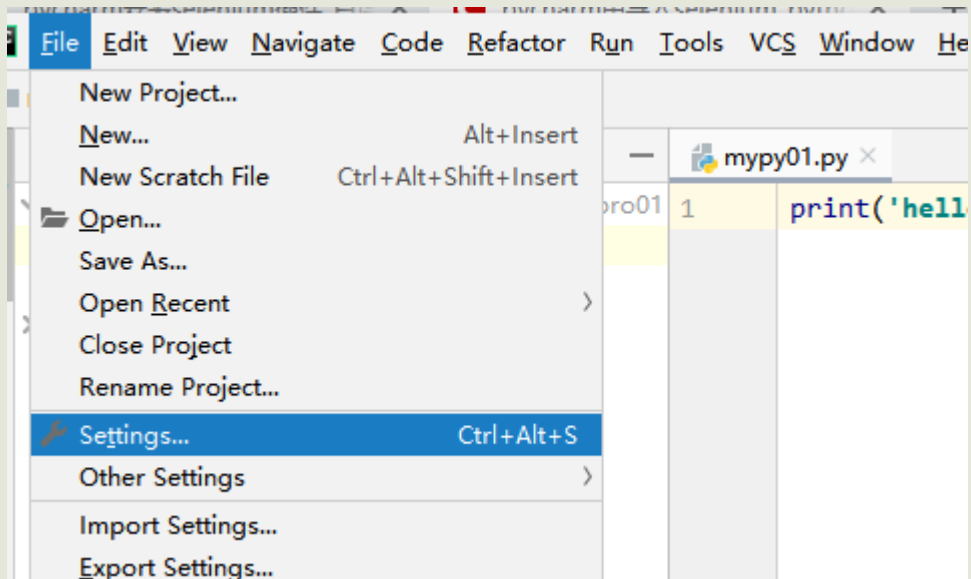
```
C:\Users\lj276>pip install selenium
Collecting selenium
  Downloading https://files.pythonhosted.org/packages/80/d6/4294f0b4bce4de0abf13e17190289f9d0613b0a44e5dd6a7f5ca98459853/selenium-3.141.0-py2.py3-none-any.whl (904kB)
    |██████████████████████████████████████████████████████████████████████████████| 911kB 35kB/s
Collecting urllib3 (from selenium)
  Downloading https://files.pythonhosted.org/packages/b4/40/a9837291310ee1ccc242ceb6ebfd9eb21539649f193a7c8c86ba15b98539/urllib3-1.25.7-py2.py3-none-any.whl (125kB)
    |██████████████████████████████████████████████████████████████████████████████| 133kB 46kB/s
Installing collected packages: urllib3, selenium
Successfully installed selenium-3.141.0 urllib3-1.25.7
```

说明：在通过pip安装python的第三方库时，如果只输入包名，则默认安装当前库中最新的版本，如果不想安装最新版本的包，则可以在包名后面加版本号。

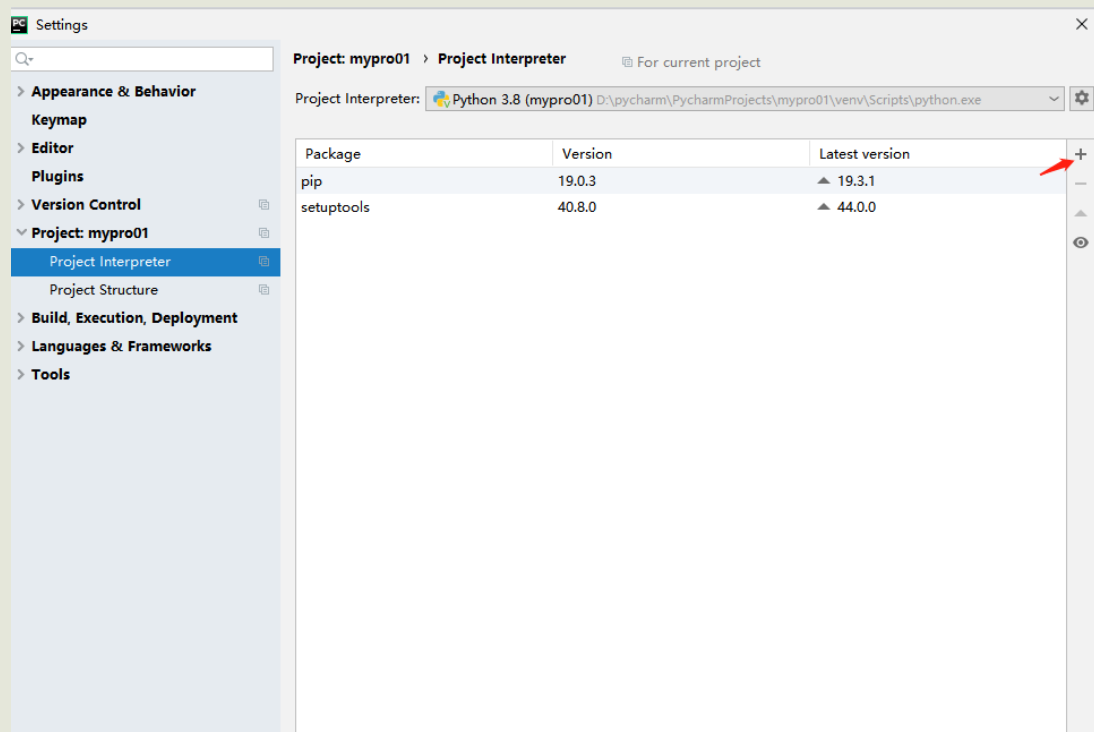
02

可视化安装

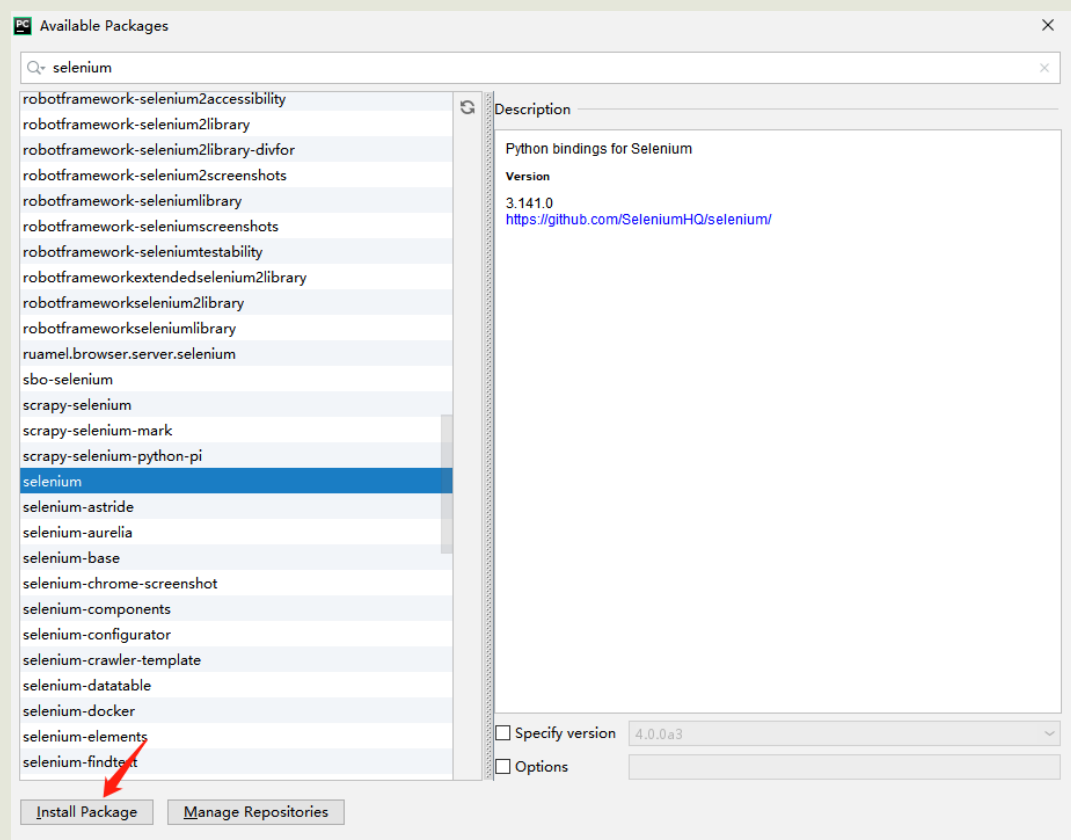
1、通过菜单栏File >> Settings 进入配置界面



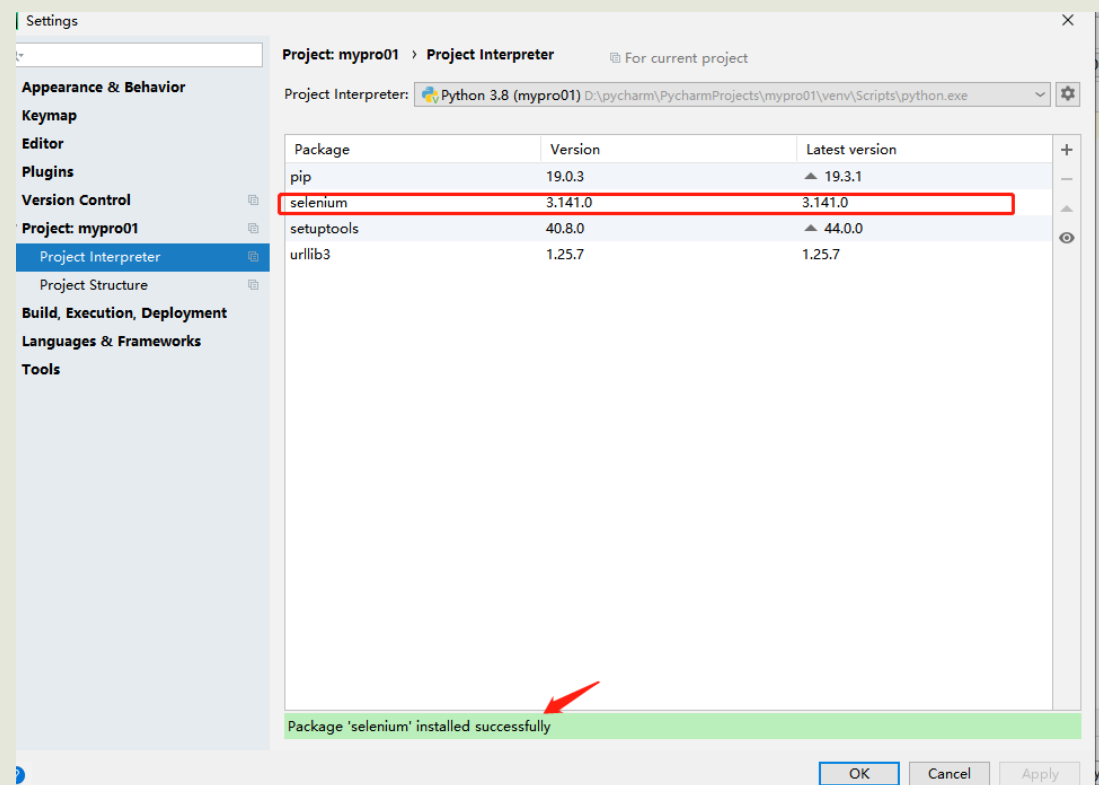
2、如图所示在interpreter页面点击“+”

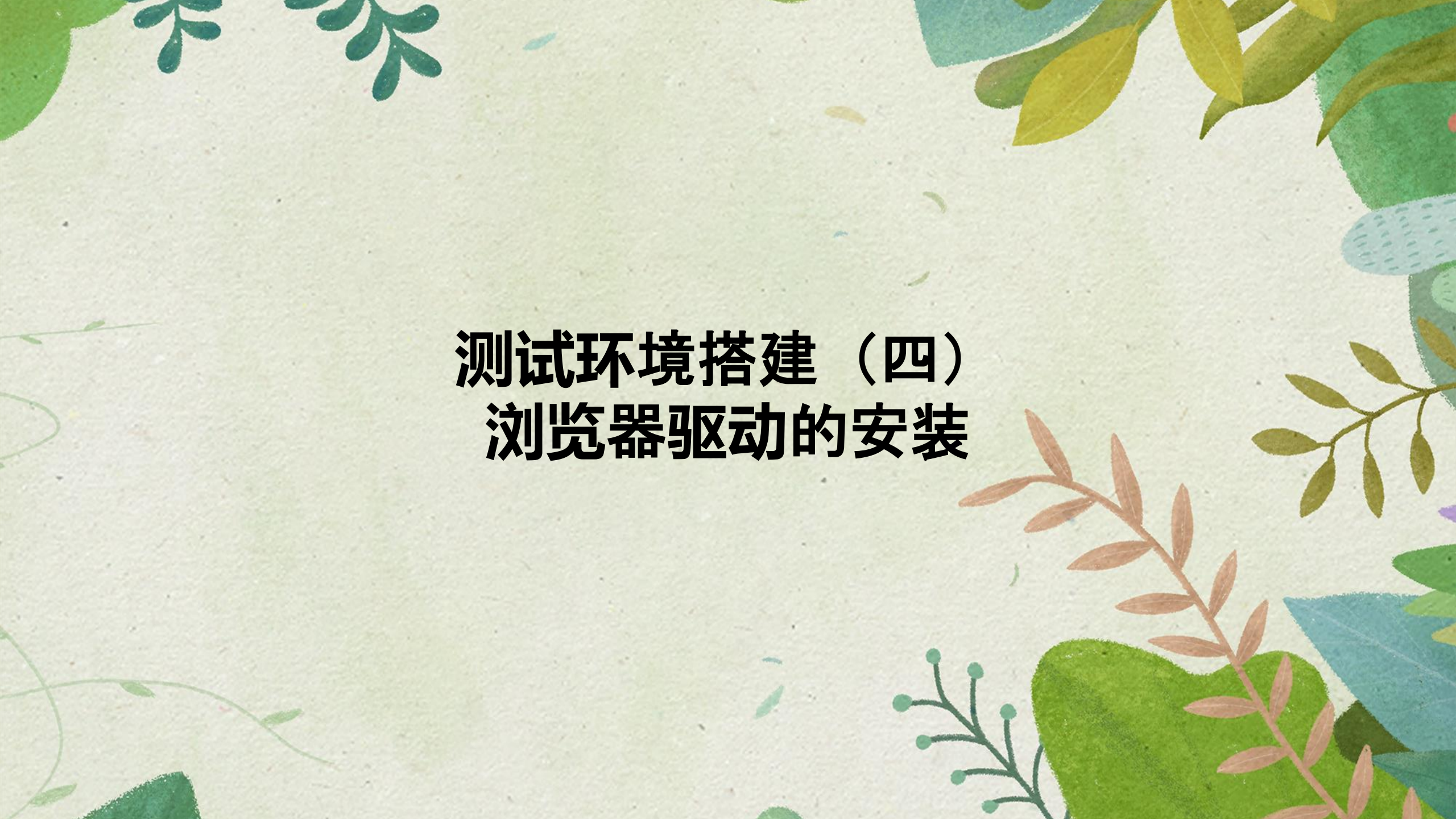


3、点击加号之后，在搜索界面搜索selenium，点击install，提示成功



4、安装成功之后，回到interpreter界面可以看到安装成功

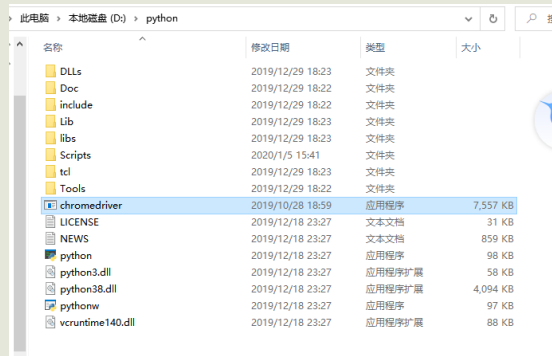




测试环境搭建（四） 浏览器驱动的安装

WebDriver下载和安装

- WebDriver支持Firefox(FirefoxDriver)、IE(InternetExplorerDriver)、和Chrome(ChromeDriver)等浏览器。除此之外，它还支持Android和iPhone的移动应用测试。
- Chrome浏览器驱动下载地址：
<http://chromedriver.storage.googleapis.com/index.html>
- 注意选择和浏览器版本相对应的webdriver版本
- 下载之后将解压好的exe文件放在python同一级目录下即可



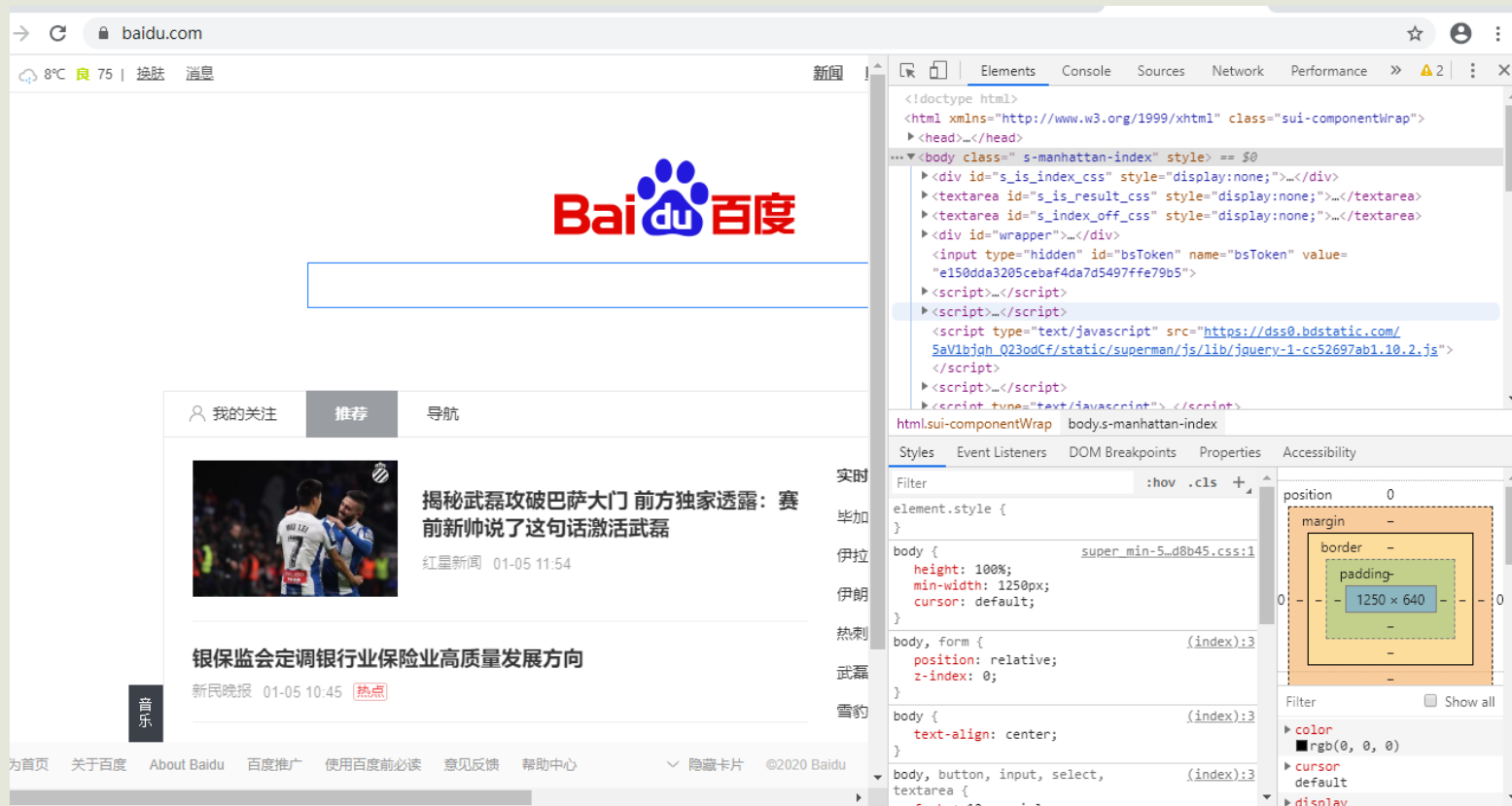


WebDriver API (一)

元素定位

01

开发者工具



说明：我们可以通过Chrome浏览器的开发者工具帮助我们定位页面元素。我们可以看到页面上的元素是由一行行代码组成，有层级，每个元素有不同的标签名和属性值，webdriver就是通过这些信息找到不同的元素。

打开开发者工具：

- 1、下拉菜单中-更多工具-开发者工具
- 2、快捷键：Ctrl+Shift+I
- 3、按F12

八种定位方式

定位方式	python中对应的方法
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()
by xpath	find_element_by_xpath()
css selector	find_element_by_css_selector()

HTML结构特征

- 它们由标签对组成

```
<html></html>
```

```
<body></body>
```

```
<div></div>
```

html、div等就是标签的标签名

- 标签有各种属性（id、name、class name、tag name定位）

```
<body class="s-manhattan-index" style>
  <div id="s_is_index_css" style="display:none;">...</div>
  <textarea id="s_is_result_css" style="display:none;">...</textarea>
  <textarea id="s_index_off_css" style="display:none;">...</textarea>
  <div id="wrapper">...</div>
```

- 标签对之间可以有文本属性
（link text、partial link text定位）

```
<a href="http://news.baidu.com" target="_blank" class="mnav">新闻</a>
<a href="https://www.hao123.com" target="_blank" class="mnav">hao123</a>
<a href="http://map.baidu.com" target="_blank" class="mnav">地图</a>
<a href="http://v.baidu.com" target="_blank" class="mnav">视频</a>
<a href="http://tieba.baidu.com" target="_blank" class="mnav">贴吧</a>
<a href="http://xueshu.baidu.com" target="_blank" class="mnav">学术</a>
```

- 标签有层级关系（xpath、css定位）

```
<head>...</head>
<body class="s-manhattan-index" style>
  <div id="s_is_index_css" style="display:none;">...</div>
  <textarea id="s_is_result_css" style="display:none;">...</textarea>
  <textarea id="s_index_off_css" style="display:none;">...</textarea>
  <div id="wrapper">
    <div class="s-skin-container s-isindex-wrap"></div>
    <div id="head" class>
```


- HTML规定id属性在HTML文档中必须是唯一的，类似于身份证号，具有很强的唯一性，WebDriver提供的id定位方式就是通过元素的id属性来查找元素。
- 通过id定位百度输入框：

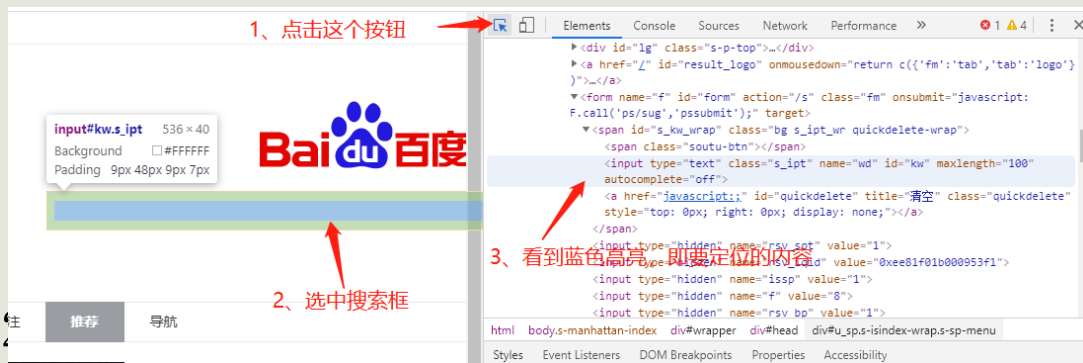
1、打开百度首页，并打开开发者工具按图所示操作定位搜索框



2、id定位方法如下：

`find_element_by_id(“kw”)`

- HTML规定name来指定元素的名称，因此它更像是人名。name的属性值，在当前页面中可以不唯一。
- 通过name定位百度输入框：
 - 1、打开百度首页，并打开开发者工具按图所示操作定位搜索框



find_element_by_name(“wd”)

class定位

- HTML规定class来指定元素的类名，用法与id、name类似。
- 通过class属性定位百度输入框：

1、打开百度首页，并打开开发者工具按图所示操作定位搜索框



2、class定位方法如下：

`find_element_by_class_name(“s_ipst”)`

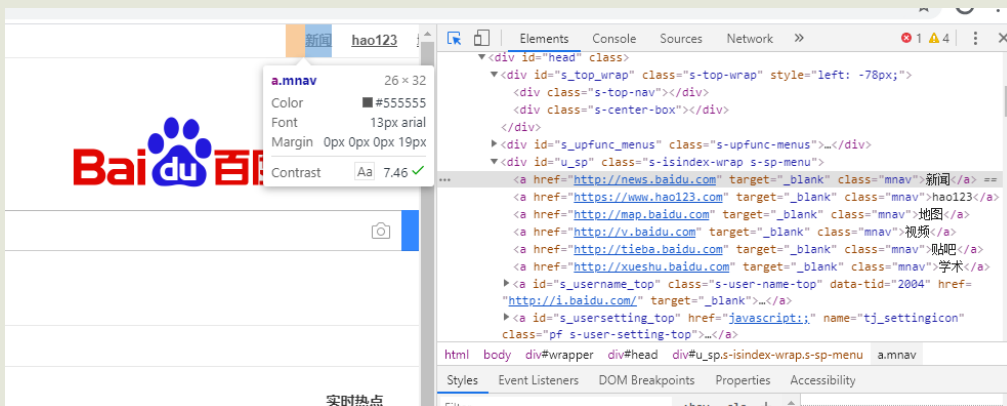
tag定位

- HTML的本质就是通过tag来定义实现不同的功能，每一个元素本质也是一个tag。因为tag往往用来定义一类功能，所以通过tag识别某个元素的概率很低。例如我们打开任意一个页面，就会发现大量的<div>、<input>、<a>等tag，所以很难通过tag name去区分不同的元素，这里仅介绍一下用法。
- tag定位方法如下：
`find_element_by_tag_name(“input”)`

link定位

- link定位与前面介绍的几种定位方式有所不同，它专门用于定位文本链接。
- 通过link定位百度首页的文本链接：

1、打开百度首页，并打开开发者工具按同意的操作定位搜索框



2、link定位方法如下：

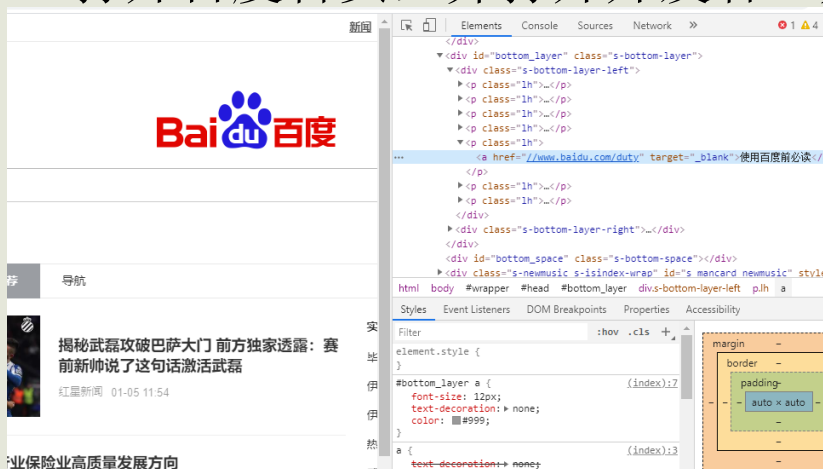
`find_element_by_link_text(“新闻”)`

`find_element_by_link_text(“hao123”)`

partial link定位

- partial link定位是对link定位的一种补充。有些文本链接过长，可以取文本的一部分定位，这要这一部分信息可以唯一地表示这个链接。
- 通过partial link定位百度首页的文本链接：

1、打开百度首页，并打开开发者工具按同意的操作定位搜索框



2、partial link定位方法如下：

`find_element_by_partial_link_text("必读")`

XPath定位--绝对路径

- XPath是XML文档中定位元素的语言，因为HTML可以看作是XML的一种实现，所以selenium可以使用这种强大的语言在web应用中定位元素。
- 绝对路径定位：把元素看成一个人，假设这个人没有任何属性特征（姓名、身份证号、手机号），但是这个人在某个地理位置，比如湖北省武汉市江夏区金融港路XX号。对于页面元素，也有这样的一个绝对地址，我们可以根据这个地址来定位。
- XPath主要用标签名的层级关系来定位元素的绝对路径，最外层是html，在body内一级一级往下找，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，例如：div[2]表示当前层级下的第二个div标签
- XPath绝对路径定位比较繁琐，且一旦元素发生变化，可能就会失效，程序在运行的时候检索也会比较慢，不推荐使用
- 使用方法如下：
`find_element_by_xpath("/html/body/div/div/div/div/div/form/span[1]/input")`

XPath定位--相对路径

- 相对路径是指由这个文件所在的路劲引起的跟其他文件（或文件夹）的路径关系，相比绝对路径，相对路径更容易定位到对应的位置且写法简单，也不用担心元素的位置变化而调整定位。
- 标签+属性匹配

有一个标签+某一个属性的组合，其中@代表匹配属性名称，相当于匹配所有input标签并匹配属性id=kw的元素（以id=kw为例，其他的是一样的意思）。

```
#标签+属性--定位到input标签中id=kw元素
driver.find_element_by_xpath("//input[@id='kw']").send_keys("淘宝")
#标签+属性--定位到input标签中的name=wd元素
driver.find_element_by_xpath("//input[@name='wd']").send_keys("淘宝")
#标签+属性--定位到input标签中的class=s_ipt元素
driver.find_element_by_xpath("//input[@class='s_ipt']").send_keys("淘宝")
```

- 标签和多个属性

由一个标签+多个属性的组合，相当于匹配了所有的input标签并匹配了属性id=kw且/或属性name=wd的元素。

#标签+多个属性

```
driver.find_element_by_xpath("//input[@id='kw' and @name='wd' ]").send_keys("淘宝")  
driver.find_element_by_xpath("//input[@id='kw' or @name='wd' ]").send_keys("淘宝")
```

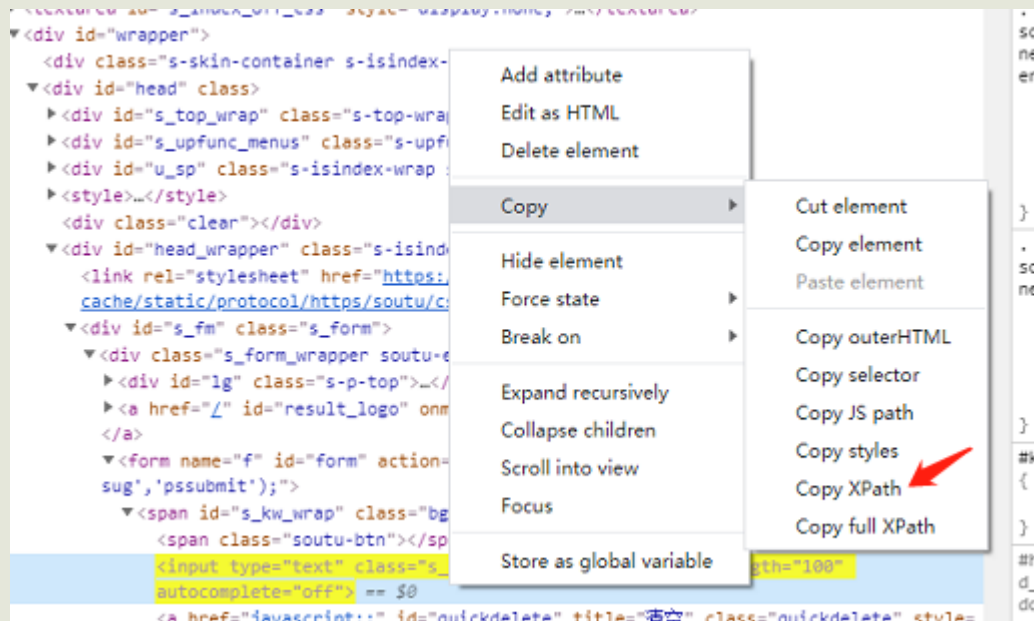
- 父子定位

相当于先定位到了父节点，再找到对应的子节点，常用于当前元素不易定位而父节点较易定位的情况。

#父子定位

```
#driver.find_element_by_xpath("//span[@class='bg s_ipt_wr quickdelete-wrap']/input").send_keys("淘宝")  
driver.find_element_by_xpath("//form[@id='form']/span[1]/input").send_keys("淘宝")
```

- 右键复制xpath



#右键复制xpath

```
driver.find_element_by_xpath("//input[@class='s_ipt']").send_keys("淘宝")
```


CSS定位

- css是一种语言，被用来描述HTML和XML文档的样式，css使用选择器来为页面元素绑定属性，这些属性可以被selenium用作另外的定位策略。
- 一般情况下比xpath定位速度快，但对于初学者学习起来稍微有难度，下面介绍一下css选择器常用的一些定位。
- 根据id定位--用“#”表示

#根据id定位

```
driver.find_element_by_css_selector("#kw").send_keys("淘宝")
```

- 根据class定位--用“.”表示

#根据class定位

```
driver.find_element_by_css_selector(".s_ippt").send_keys("淘宝")
```

- 根据元素属性--不需要加 “@”

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标识这个元素，对于属性值来说，可加引号也可以不加，但注意和整个字符串的引号进行区分。

#根据属性定位

```
driver.find_element_by_css_selector("[name=wd]").send_keys("淘宝")
driver.find_element_by_css_selector("[autocomplete='off']").send_keys("淘宝")
```

- 根据父子关系定位

表示有父元素span, 查找它的所有标签名叫input的子元素，一般为了确保结果的唯一性会加上属性值

#根据父子关系定位

```
driver.find_element_by_css_selector("span>input").send_keys("淘宝")
```

#根据父子关系定位+属性组合

```
driver.find_element_by_css_selector("span.s_btn_wr>input#su").click()
```

定位方法总结

- 策略：选择简单、稳定的定位方法；
- 首选id，如果没有再用其他定位方法；
- `css_selector`执行速度快，推荐使用，对于新手较难；
- 定位超链接的时候，可以考虑`link_text`或`partial_link_text`，但是文本经常发生改变，所以不推荐使用；
- `xpath`功能最强，但是执行速度慢，因为需要查找整个DOM，所以尽量少用，实在没办法时才用`xpath`。



WebDriver API (二)

控制浏览器

1

控制浏览器窗口大小

- 有时候我们希望能以某种浏览器尺寸打开，让访问的页面在这种尺寸下运行，WebDriver提供了set_window_size()方法来设置浏览器的大小，需要传入参数宽和高。在PC端执行自动化测试脚本大多数的情况下是希望浏览器在全屏显示下执行，可以使用maximize_window()，不需要参数。

#全屏显示

```
print("设置浏览器全屏显示")
```

```
driver.maximize_window()
```

#最小化

```
print("设置浏览器最小化")
```

```
driver.minimize_window()
```

#以某种尺寸打开

```
print("设置浏览器宽480，高800显示")
```

```
driver.set_window_size(480, 800)
```

控制浏览器前进后退+模拟浏览器刷新

- 在使用浏览器浏览网页时，浏览器提供了后退和前进按钮，可以方便地在浏览过的网页之间切换，WebDriver也提供了对应的back()和forward()方法来模拟后退和前进按钮。
- 使用refresh()方法模拟浏览器刷新

#返回到百度首页

```
print("back to :", first_url)
```

```
driver.back()
```

```
time.sleep(2)
```

#前进到新闻首页

```
print("foeward to :", second_url)
```

```
driver.forward()
```

```
time.sleep(2)
```

#刷新浏览器

```
driver.refresh()
```



WebDriver API (三)

简单元素操作

1

最常用的元素操作方法

- `clear()` #清除文本
- `send_keys(*value)` #模拟按键输入
- `click()` #单击元素



WebDriver API（四） 多表单切换

多表单切换

- 在Web应用中经常会遇到frame/iframe表单嵌套页面的应用，WebDriver只能在一个页面上对元素识别与定位，对于frame/iframe表单内嵌页面上的元素无法直接定位，这是需要通过switch_to_frame()方法将当前定位的主体切换为frame/iframe表单的内嵌页面中
- 怎么确认是否有frame
 1. 目标元素往上搜索，查找是否有iframe/frame标签
 2. ctrl+f中直接搜索iframe/frame
 3. 整个页面有框框分割布局就要考虑存在iframe/frame

- 定位iframe标签

driver.switch_to.frame() 默认可以直接取表单的id或name属性

#定位iframe标签

```
driver.switch_to.frame("x-URS-iframe1578822624680.8792")
```

如果iframe没有可用的id或者name属性，可以用下面的方式进行定位：

```
a=driver.find_element_by_xpath("//*[@id='loginDiv']/iframe")
```



#切换到对应的iframe，才能在内嵌页面进行元素定位

```
driver.switch_to.frame(a)
```

如果表单完成了在当前表单上的操作，则可以通过driver.switch_to.parent_frame() 方法跳到父层，除此之外，在进入多级表单的情况下，还可以通过driver.switch_to.default_content() 跳回最外层的页面。



WebDriver API（五）

鼠标事件+键盘事件简单介绍

鼠标事件

- 通过前面的例子了解到，可以使用`click()`来模拟鼠标的单击操作，在`WebDriver`中，提供了更丰富的鼠标交互方式，例如：鼠标右击、双击、悬停、鼠标拖动等功能。将这些关于鼠标操作的方法封装在`ActionChains`类中。

方法	鼠标的操作
<code>perform()</code>	执行所有 <code>ActionChains</code> 中存储的行为
<code>context_click()</code>	右击
<code>double_click()</code>	双击
<code>move_to_element()</code>	鼠标悬停
<code>drag_and_drop(source, target)</code>	鼠标拖放

2

鼠标右击操作（其他操作原理一样）

- `from selenium.webdriver import ActionChains`
导入提供鼠标操作的ActionChains类
- `ActionChains(driver)`
调用ActionChains()类，将浏览器驱动driver作为参数传入
- `context_click(right_click)`
`context_click()`方法用于模拟鼠标右键操作，在调用时需要指定元素定位
- `perform()`
执行所有ActionChains()中储存的行为，可以理解成是对整个操作的提交动作

示例:

```
import selenium      #导入selenium
from selenium import webdriver    #导入webdriver
import time          #导入time
from selenium.webdriver import ActionChains #导入提供鼠标操作的ActionChains类

#调用Chrome浏览器
driver = webdriver.Chrome()

#获取百度首页的url
driver.get("https://baidu.com")

#定位到要右击的元素
right_click=driver.find_element_by_link_text("新闻")
#对定位到的元素执行鼠标右键操作
ActionChains(driver).context_click(right_click).perform()

#定位到要悬停的元素
# above=driver.find_element_by_name("tj_briicon")
# 执行鼠标悬停操作
# ActionChains(driver).move_to_element(above).perform()

#定位到要双击的元素
driver.find_element_by_id("kw").send_keys("abc")
double_click= driver.find_element_by_id("kw")
#执行鼠标双击操作
ActionChains(driver).double_click(double_click).perform()
```



键盘事件

- Keys() 类中提供了键盘上几乎所有按键的方法。前面了解到send_keys() 方法可以用来模拟键盘输入，除此之外，我们还可以用它来输入键盘上的按键，甚至是组合键，如Ctrl+A、Ctrl+c等。

方法	键盘的操作
send_keys(Keys.BACK_SPACE)	删除键(BackSpace)
send_keys(Keys.SPACE)	空格键(Space)
send_keys(Keys.TAB)	制表键(Tab)
send_keys(Keys.ESCAPE)	回退键(Esc)
send_keys(Keys.ENTER)	回车键(Enter)
send_keys(Keys.CONTROL,'a')	全选(Ctrl+A)
send_keys(Keys.CONTROL,'c')	复制(Ctrl+C)
send_keys(Keys.CONTROL,'x')	剪切(Ctrl+X)
send_keys(Keys.CONTROL,'v')	粘贴(Ctrl+V)
send_keys(Keys.F1)	键盘F1

谢 谢 观 看