java初学selenium

24/11/12 17:03 257 0 0 T 6312 12:37 ~ 21:02

JAVA

第一章 Selenium 环境搭建

1.1.java 环境

想要通过 java 语言来使用 selenium 框架，前提要完成 jdk

详细安装教程见：https://blog.csdn.net/shengmer/article/

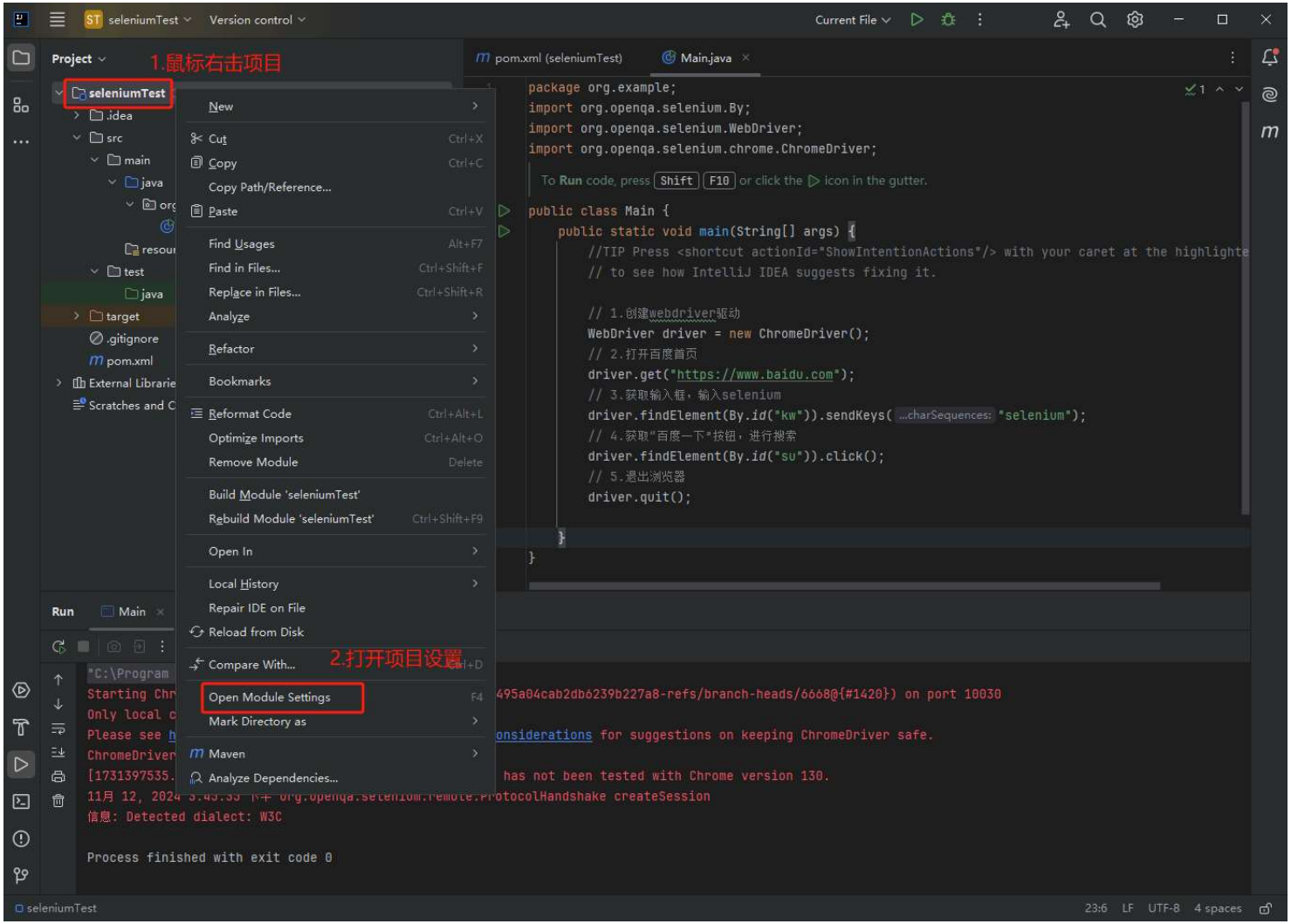
1.2.selenium 环境

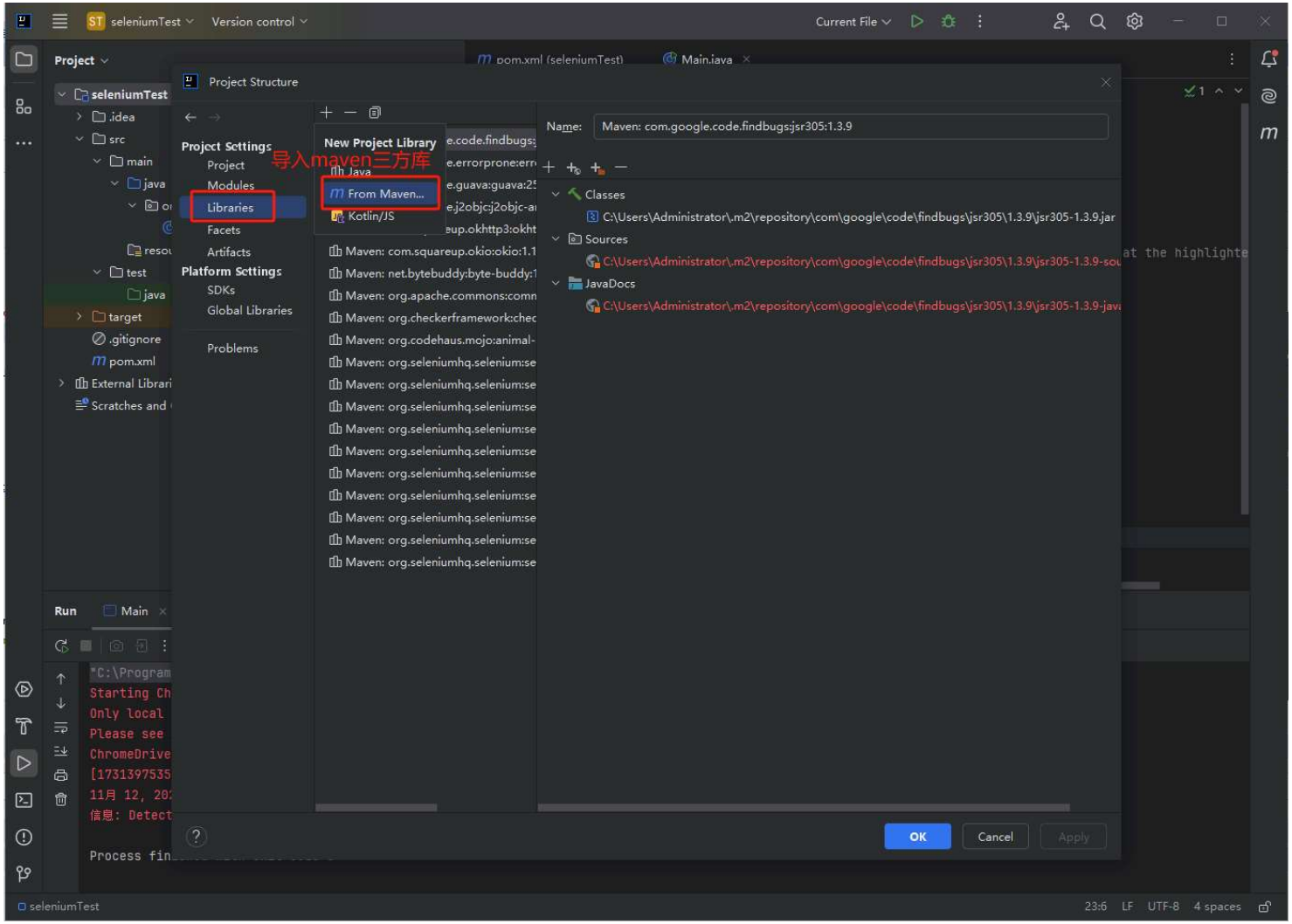
简单 java 工程：

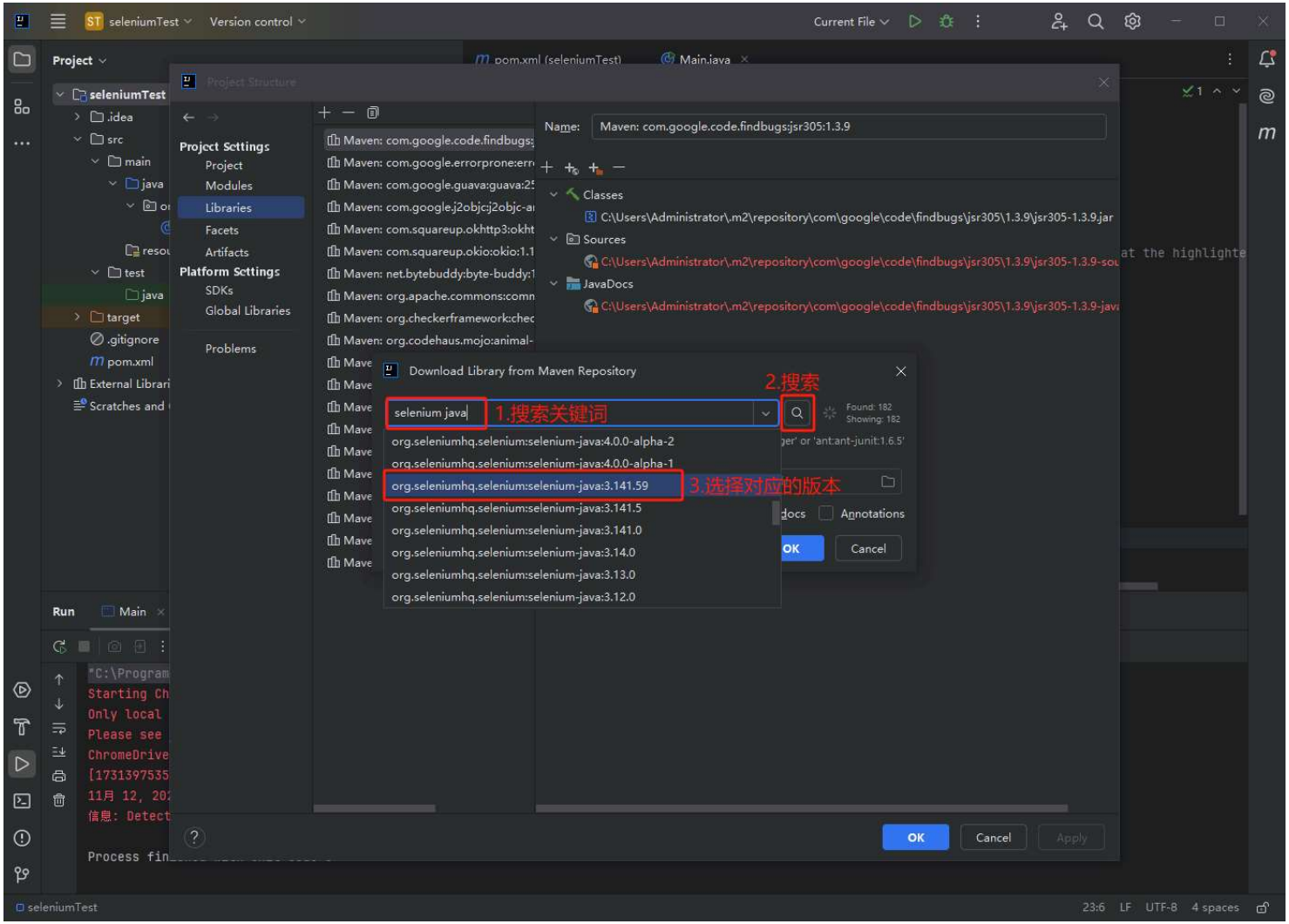
直接导入 selenium的 jar 包就可以了。

jar包下载地址： Selenium各个版本jar下载， 下载对应的版本即可

maven工程：




seleniumTest
Version control
Current File
Project
pom.xml (seleniumTest)
Main.java
Project Structure
Project Settings
Project
Modules
Libraries
Facets
Artifacts
Platform Settings
SDKs
Global Libraries
Problems
导入maven三方库
New Project Library
Java
From Maven...
Kotlin/JS
Name:
Maven: com.google.code.findbugs:jsr305:1.3.9
Classes
C:\Users\Administrator\.m2\repository\com\google\code\findbugs\jsr305\1.3.9\jsr305-1.3.9.jar
Sources
JavaDocs
OK
Cancel
Apply
Run
Main
seleniumTest
23:6
LF
UTF-8
4 spaces



在pom文件中引入对应的依赖即可：

maven仓库：https://mvnrepository.com/

▶ 点击查看代码

1.3.selenium3 对应浏览器驱动下载

当selenium升级到3.0之后，对不同的浏览器驱动进行了规范。如果想使用selenium驱动不同的浏览器，必须单独下载并设置不同的浏览器驱动。
各浏览器下载地址：

- Firefox浏览器驱动：geckodriver
- Chrome浏览器驱动：chromedriver taobao备用地址
- IE浏览器驱动：IEDriverServer
- Edge浏览器驱动：MicrosoftWebDriver
- Opera浏览器驱动：operadriver
- PhantomJS浏览器驱动：phantomjs

注：部分浏览器驱动地址需要梯子。

设置浏览器驱动

设置浏览器的地址非常简单。 我们可以手动创建一个存放浏览器驱动的目录，如： C:\driver , 将下载的浏览器驱动文件（例如：chromedriver、geckodriver）丢到该目录下。
我的电脑–>属性–>系统设置–>高级–>环境变量–>系统变量–>Path，将“C:\driver”目录添加到Path的值中。

验证浏览器驱动

验证不同的浏览器驱动是否正常使用。

▼ 点击查看代码

```
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.edge.EdgeDriver;
```

```
import org.openqa.selenium.ie.InternetExplorerDriver;
import org.openqa.selenium.opera.OperaDriver;
import org.openqa.selenium.phantomjs.PhantomJSDriver;


WebDriver driver = new ChromeDriver();     //Chrome浏览器

WebDriver driver = new FirefoxDriver();    //Firefox浏览器

WebDriver driver = new EdgeDriver();       //Edge浏览器

WebDriver driver = new InternetExplorerDriver();  // Internet Explorer浏览器

WebDriver driver = new OperaDriver();      //Opera浏览器

WebDriver driver = new PhantomJSDriver();   //PhantomJS
```

第二章 Selenium 简单示例

- 打开百度进行搜索：

▼ 点击查看代码

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

/**
 * @Description: 通过selenium操作浏览器打开百度进行搜索
 * selenium版本: 3.141.59;  通过maven管理jar包
 * 开发工具: IDEA
 * jdk: 1.8
 * 浏览器: chrome 75+
 * @Author: zxb
 * @Date: 2024/11/22
 */
public class BaiduSearch {
```

```
    public static void main(String[] args) {
        // 1.创建webdriver驱动
        WebDriver driver = new ChromeDriver();
        // 2.打开百度首页
        driver.get("https://www.baidu.com");
        // 3.获取输入框，输入selenium
        driver.findElement(By.id("kw")).sendKeys("selenium");
        // 4.获取"百度一下"按钮，进行搜索
        driver.findElement(By.id("su")).click();
        // 5.退出浏览器
        driver.quit();
    }
}
```

第三章 八大元素定位

- 为什么要进行元素定位?

我们必须告诉 selenium 怎么去定位元素，用来模拟用户的动作，或者查看元素的属性和状态，以便于我们可以执行检查。例如：我们要搜索一个产品，首先要找到搜索框与搜索按钮，接着通过键盘输入要查询的关键字，最后鼠标单击搜索按钮，提交搜索请求。

正如上述的人工操作步骤一样，我们也希望 selenium 能模拟这样的动作，然而，selenium 并不能理解类似在搜索框中输入关键字或者点击搜索按钮这样的图形化的操作。所以需要我们程序化的告诉 selenium 如何定位搜索框和搜索按钮，从而模拟键盘和鼠标的操作。

3.1.定位方式

selenium 提供了8种的定位方式：

- id
- name

- class name
- tag name
- link text
- partial link text
- xpath
- css selector

这8种定位方式在java selenium 中对应的方法为：

方法	描述	参数	示例
findElement(By.id())	通过元素的 id 属性值来定位元素	对应的id属性值	findElement(By.id("kw"))
findElement(By.name())	通过元素的 name 属性值来定位元素	对应的name值	findElement(By.name("user"))
findElement(By.className())	通过元素的 class 名来定位元素	对应的class类名	findElement(By.className("passworld"))
findElement(By.tagName())	通过元素的 tag 标签名来定位元素	对应的标签名	findElement(By.tagName("input"))
findElement(By.linkText())	通过元素标签对之间的文本信息来定位元素	文本内容	findElement(By.linkText("登录"))
findElement(By.partialLinkText())	通过元素标签对之间的部分文本信息来定位元素	部分文本内容	findElement(By.partialLinkText("百度"))
findElement(By.xpath())	通过xpath语法来定位元素	xpath表达式	findElement(By.xpath("//input[@id='kw']"))
findElement(By.cssSelector())	通过css选择器来定位元素	css元素选择器	findElement(By.cssSelector("#kw"))

同时这8种方法都对应有着返回复数元素的方法，分别在调用的方法findElements(By.id()) 加上一个s：

- findElements(By.id())
- findElements(By.name())
- findElements(By.className())
- findElements(By.tagName())
- findElements(By.linkText())
- findElements(By.partialLinkText())
- findElements(By.xpath())
- findElements(By.cssSelector())

3.2.定位方式的用法

假如我们有一个Web页面，通过前端工具（如，Firebug）查看到一个元素的属性是这样的。

▼ 点击查看代码

```
<html>
  <head>
  <body link="#0000cc">
    <a id="result_logo" href="/" onmousedown="return c({'fm':'tab','tab':'logo'})">
    <form id="form" class="fm" name="f" action="/s">
      <span class="soutu-btn">按钮</span>
        <input id="kw" class="s_ipt" name="wd" value="" maxlength="255" autocomplete="off">
```

我们的目的是要定位input标签的输入框。

- 通过id定位:

```
driver.findElement(By.id("kw"))
```

- 通过name定位:

```
driver.findElement(By.name("wd"))
```

- 通过class name定位:

```
driver.findElement(By.className("s_ipt"))
```

- 通过tag name定位:

```
driver.findElement(By.tagName("input"))
```

- 通过xpath定位，xpath定位有N种写法，这里列几个常用写法:

▼ 点击查看代码

```
driver.findElement(By.xpath("//*[@id='kw']")) // id定位
driver.findElement(By.xpath("//*[@name='wd']")) // 属性值定位
driver.findElement(By.xpath("//span[text()='按钮']")) // 文本定位
driver.findElement(By.xpath("//input[@class='s_ipt']")) // class属性定位
driver.findElement(By.xpath("/html/body/form/span/input")) // 绝对路径定位
driver.findElement(By.xpath("//span[@class='soutu-btn']/input")) // 相对路径定位
driver.findElement(By.xpath("//form[@id='form']/span/input"))
driver.findElement(By.xpath("//input[@id='kw' and @name='wd']")) // 多组合属性定位
driver.findElement(By.xpath("//span[contains(text(),'按钮')]")) // 是否包含文本
```

- 通过css定位，css定位有N种写法，这里列几个常用写法:

▼ 点击查看代码

```
driver.findElement(By.cssSelector("#kw") // id定位
driver.findElement(By.cssSelector("[name=wd]") // name属性值定位
driver.findElement(By.cssSelector(".s_ipt") // class地位
driver.findElement(By.cssSelector("html > body > form > span > input") // css层级定位
driver.findElement(By.cssSelector("span.soutu-btn> input#kw")
driver.findElement(By.cssSelector("form#form > span > input")
```

- 接下来，我们的页面上有一组文本链接。

▼ 点击查看代码

```
<a class="mnav" href="http://news.baidu.com" name="tj_trnews">新闻</a>
<a class="mnav" href="http://www.hao123.com" name="tj_trhao123">hao123</a>
```

- 通过linkText定位:

▼ 点击查看代码

```
driver.findElement(By.linkText("新闻")
driver.findElement(By.linkText("hao123")
```

- 通过 partialLinkText 定位:

▼ 点击查看代码

```
driver.findElement(By.partialLinkText("新")
driver.findElement(By.partialLinkText("hao")
driver.findElement(By.partialLinkText("123")
```

参考网址：https://www.cnblogs.com/tester-ggf/p/12602211.html

__EOF__

标签: Java

推荐该文 关注博主 收藏本文 分享微信

测试周小白
粉丝 - 1 关注 - 7

+加关注

0 0

« 上一篇： Java程序基础结构

» 下一篇： 面试题

posted @ 2024-11-12 17:03 测试周小白 阅读(257) 评论(0) 编辑 收藏 举报

编辑推荐：
· 硬盘空间消失之谜：Linux 服务器存储排查与优化全过程
· JavaScript是按顺序执行的吗？聊聊JavaScript中的变量提升
· [杂谈]后台日志该怎么打印
· Pascal 架构 GPU 在 vllm下的模型推理优化
· .NET Core 堆结构(Heap)底层原理浅谈

阅读排行：
· 丢人，被自己出的校招题给麻痹了。
· C#/.NET/.NET Core技术前沿周刊 | 第 17 期（2024年12.09-12.15）
· WinForm 通用权限框架，简单实用支持二次开发
· 如何为在线客服系统的 Web Api 后台主程序添加 Bootstrap 启动页面
· 硬盘空间消失之谜：Linux 服务器存储排查与优化全过程