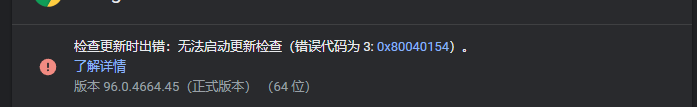
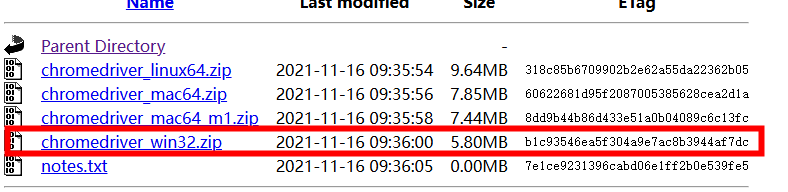
​

 环境：

确定谷歌的版本：

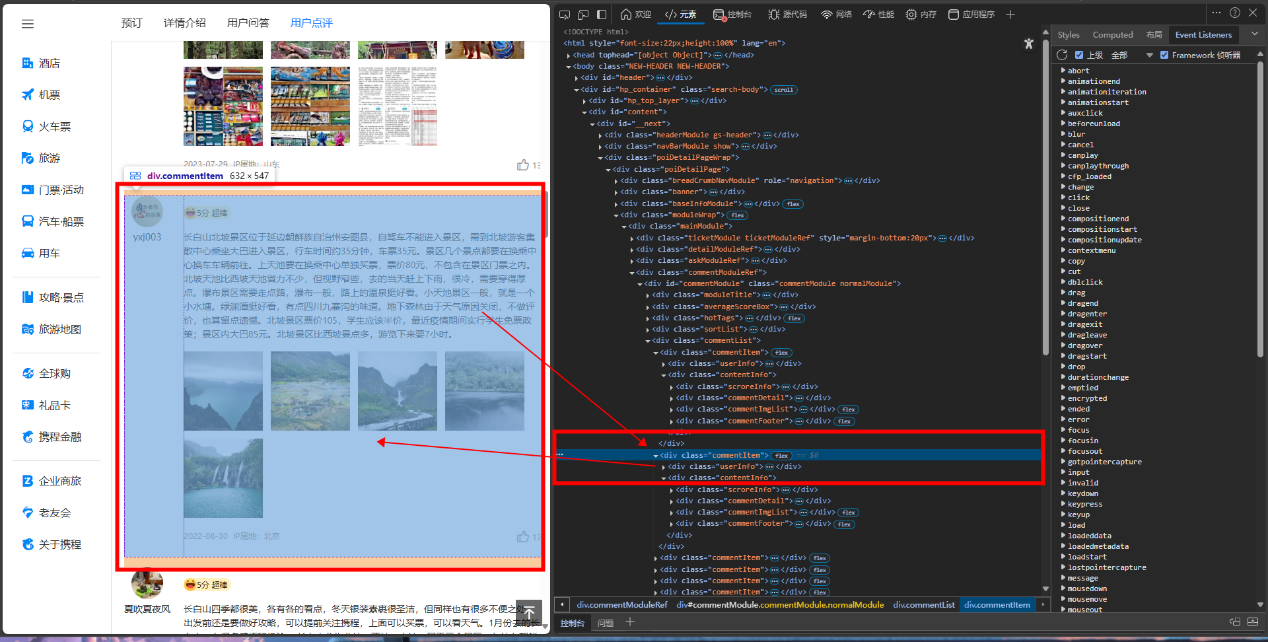
版本 96.0.4664.45（正式版本） （64 位）

​编辑确定chromedriver的版本：（谷歌版本为多少就选择哪个版本的）

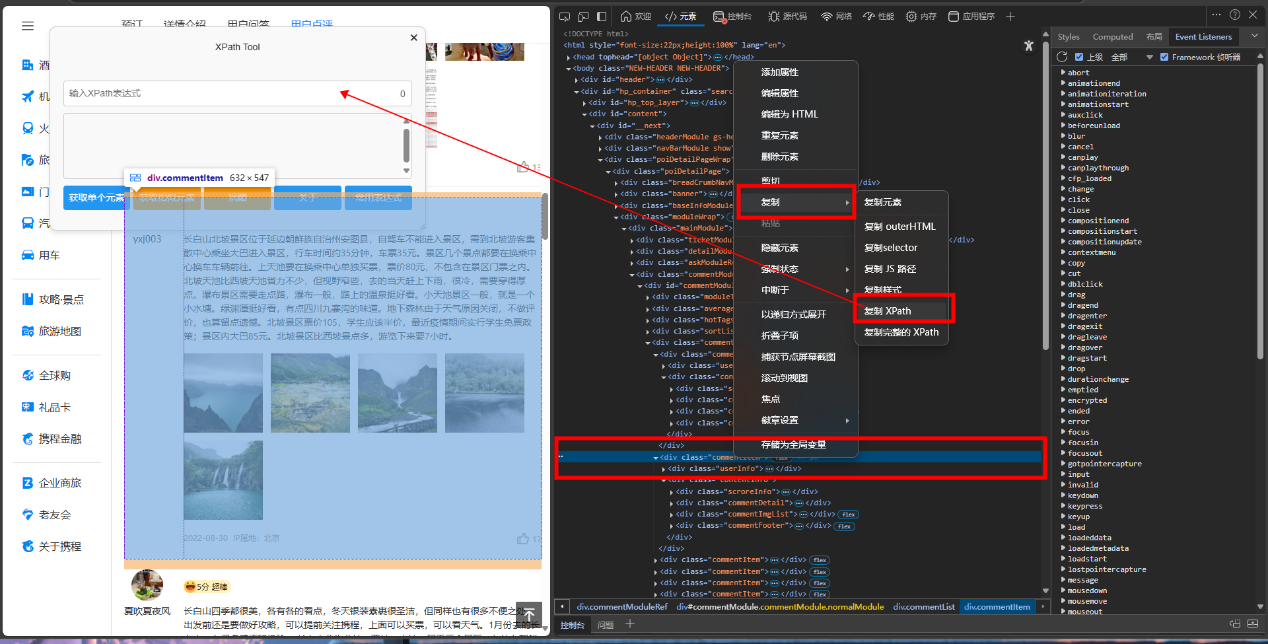
下载链接：[CNPM Binaries Mirror](https://registry.npmmirror.com/binary.html?path=chromedriver/)  
​编辑我的电脑为Windows，直接选取win32（不管是64或者32的都选择32的就可以）

确定爬取的网站：

url：https://you.ctrip.com/sight/antu643/136032.html?scene=online

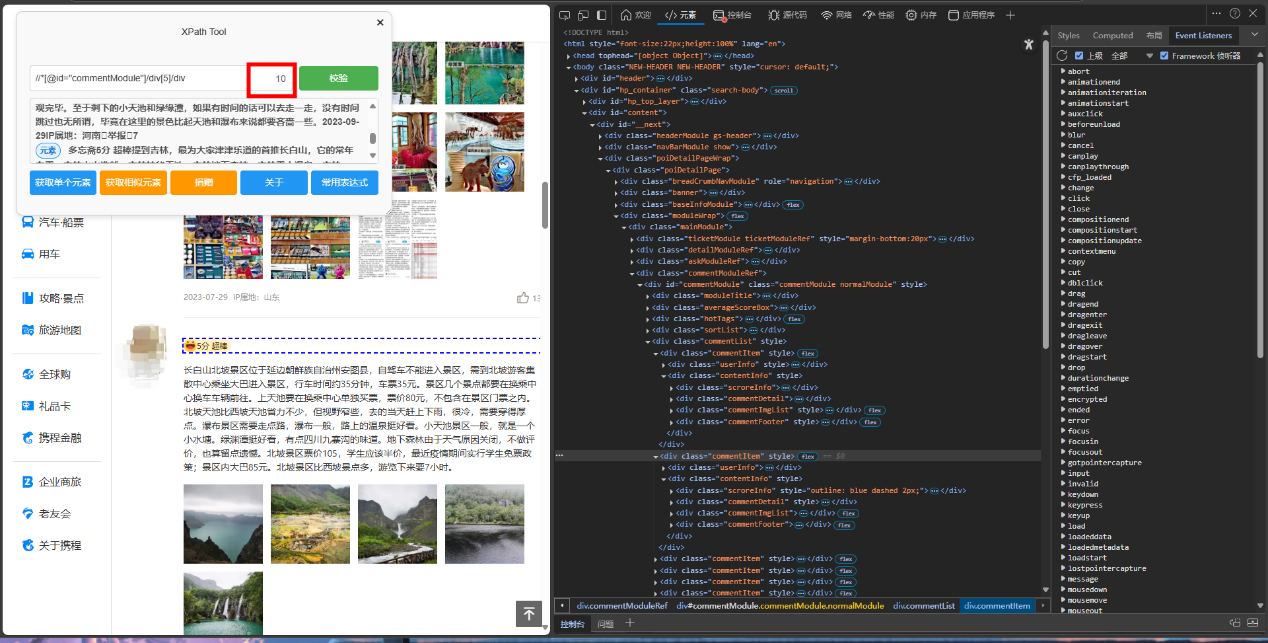
定位爬取的元素：  
​编辑

检验xpath：

​编辑

​编辑

定位本页所有评论：

​编辑

好了直接上代码

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

from selenium.webdriver.common.keys import Keys

import time

import json

import os

import random

# # 获取配置对象 => 什么样的浏览器就选择什么浏览器配置

# option = webdriver.ChromeOptions()

# option.add\_experimental\_option("detach", True)

#

# # 获取driver对象, 并将配置好的option传入进去

# driver = webdriver.Chrome(options=option)

# driver.get('https://you.ctrip.com/sight/antu643/136032.html?scene=online')

# time.sleep(10)

# print("=========第一次尝试=========")

# # 新增爬取文本的操作

# try:

# # 定位元素

# element = driver.find\_element("xpath", '//\*[@id="commentModule"]/div[5]/div[1]')

#

# # 获取文本内容

# text\_content = element.text

#

# # 打印提取的文本

# print("提取到的文本内容：")

# print(text\_content)

#

# except Exception as e:

# print(f"爬取文本时出现错误: {e}")

# print("=========First Over!!!!=========")

# time.sleep(20)

# 新增爬取文本的操作

# print("=========第二次尝试=========")

# try:

# # 定位所有匹配的评论元素（使用 find\_elements 获取列表）

# comment\_elements = driver.find\_elements("xpath", '//\*[@id="commentModule"]/div[5]/div')

#

# # 检查是否找到了评论

# if not comment\_elements:

# print("未找到评论内容！")

# else:

# # 遍历每个评论元素并提取文本

# for idx, element in enumerate(comment\_elements, start=1):

# # 获取评论文本

# text\_content = element.text.strip() # strip() 去除多余空格

# print(f"评论 {idx}:")

# print(text\_content)

# print("-" \* 50) # 分隔线

#

# except Exception as e:

# print(f"爬取文本时出现错误: {e}")

# print("=========Second Over!!!=========")

# time.sleep(20)

# print("=========Try Three=========")

# # 新增爬取文本的操作

# try:

# # 定位所有匹配的评论元素（使用 find\_elements 获取列表）

# comment\_elements = driver.find\_elements("xpath", '//\*[@id="commentModule"]/div[5]/div')

#

# # 检查是否找到了评论

# if not comment\_elements:

# print("未找到评论内容！")

# else:

# comments\_list = [] # 存储所有评论的列表

#

# # 遍历每个评论元素并提取文本

# for idx, element in enumerate(comment\_elements, start=1):

# # 获取评论文本

# text\_content = element.text.strip() # strip() 去除多余空格

# print(f"评论 {idx}:")

# print(text\_content)

# print("-" \* 50) # 分隔线

#

# # 添加到评论列表

# comments\_list.append({

# "id": idx,

# "content": text\_content

# })

#

# # 保存为 JSON 文件（项目根目录）

# json\_file\_path = os.path.join(os.getcwd(), "comments.json")

#

# with open(json\_file\_path, "w", encoding="utf-8") as f:

# json.dump(comments\_list, f, ensure\_ascii=False, indent=4) # 格式化输出

#

# print(f"评论数据已保存至：{json\_file\_path}")

#

# except Exception as e:

# print(f"爬取文本时出现错误: {e}")

# print("========Three Over========")

print("========Four Start========")

# 配置浏览器

option = webdriver.ChromeOptions()

option.add\_experimental\_option("detach", True)

# 添加随机User-Agent

user\_agents = [

"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36",

"Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0",

"Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_15\_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/14.0.3 Safari/605.1.15"

]

option.add\_argument(f"user-agent={random.choice(user\_agents)}")

# 禁用自动化特征检测

option.add\_argument("--disable-blink-features=AutomationControlled")

driver = webdriver.Chrome(options=option)

# 使用已有的date目录

date\_dir = os.path.join(os.getcwd(), "date\_CBSBP")

if not os.path.exists(date\_dir):

raise Exception(f"目录 {date\_dir} 不存在，请先创建该目录")

print(f"将使用已有目录: {date\_dir}")

# 访问目标页面

driver.get('https://you.ctrip.com/sight/antu643/136032.html?scene=online')

time.sleep(10 + random.randint(1, 5)) # 随机初始等待

# 在开始爬取前点击一次指定元素

# try:

# click\_element = WebDriverWait(driver, 15).until(

# EC.element\_to\_be\_clickable((By.XPATH, '//\*[@id="commentModule"]/div[4]/span[2]'))

# )

# exit()

# # click\_element.click()

# # print("已成功点击评论筛选元素")

# # time.sleep(5) # 等待评论加载完成

# # except Exception as e:

# # print(f"点击评论筛选元素失败: {e}")

# # driver.quit()

# # exit()

# 存储所有评论

all\_comments = []

max\_pages = 150 # 最多爬取150页

current\_page = 1

try:

while current\_page <= max\_pages:

print(f"\n========= 正在爬取第 {current\_page} 页 =========")

# 随机滚动页面模拟人工操作

for \_ in range(3):

driver.execute\_script("window.scrollBy(0, 500)")

time.sleep(random.uniform(0.5, 2))

# 提取当前页的评论

try:

comment\_elements = WebDriverWait(driver, 15).until(

EC.presence\_of\_all\_elements\_located((By.XPATH, '//\*[@id="commentModule"]/div[5]/div'))

)

except Exception as e:

print(f"定位评论元素失败: {e}")

comment\_elements = []

if not comment\_elements:

print("未找到评论内容！尝试刷新页面...")

driver.refresh()

time.sleep(10)

continue

else:

comments\_list = [] # 当前页评论列表

for idx, element in enumerate(comment\_elements, start=1):

try:

text\_content = element.text.strip()

print(f"评论 {idx}:")

print(text\_content[:50] + "...") # 只打印前50个字符

print("-" \* 50)

# 添加到当前页评论列表

comments\_list.append({

"page": current\_page,

"id": idx,

"content": text\_content

})

except Exception as e:

print(f"提取评论{idx}时出错: {e}")

# 添加到总评论列表

all\_comments.extend(comments\_list)

# 每页保存到date目录

json\_file\_path = os.path.join(date\_dir, f"comments\_page\_{current\_page}.json")

with open(json\_file\_path, "w", encoding="utf-8") as f:

json.dump(comments\_list, f, ensure\_ascii=False, indent=4)

print(f"第 {current\_page} 页评论已保存至：{json\_file\_path}")

# 检查是否已达到最大页数

if current\_page >= max\_pages:

print(f"已达到最大爬取页数 {max\_pages}，停止爬取。")

break

try:

# 随机等待时间

wait\_time = 10 + random.randint(-5, 5)

print(f"等待{wait\_time}秒后继续下一页...")

time.sleep(wait\_time)

# 尝试翻页

for attempt in range(3): # 最多尝试3次

try:

# 定位页码输入框

page\_input = WebDriverWait(driver, 15).until(

EC.presence\_of\_element\_located(

(By.XPATH, '//\*[@id="commentModule"]/div[6]/ul/li[10]/div/input'))

)

# 定位跳转按钮

page\_button = WebDriverWait(driver, 15).until(

EC.element\_to\_be\_clickable(

(By.XPATH, '//\*[@id="commentModule"]/div[6]/ul/li[10]/div/span/button'))

)

# 输入下一页页码

for \_ in range(3): # 确保输入成功

page\_input.clear()

page\_input.send\_keys(str(current\_page + 1))

time.sleep(0.5)

# 随机点击延迟

time.sleep(random.uniform(0.5, 2))

# 点击跳转按钮

driver.execute\_script("arguments[0].click();", page\_button)

# 等待页面加载

time.sleep(5 + random.randint(1, 3))

# 验证是否翻页成功

try:

WebDriverWait(driver, 10).until(

EC.presence\_of\_element\_located((By.XPATH, f'//\*[@id="commentModule"]/div[5]/div'))

)

current\_page += 1

break # 翻页成功，退出重试循环

except:

if attempt == 2:

raise Exception("翻页后未加载新内容")

continue

except Exception as e:

if attempt == 2: # 最后一次尝试

raise

print(f"翻页尝试 {attempt + 1} 失败: {e}")

time.sleep(5)

driver.refresh()

time.sleep(5)

except Exception as e:

print(f"翻页最终失败: {e}")

break

finally:

# 保存所有评论到date目录

all\_comments\_path = os.path.join(date\_dir, "all\_comments.json")

with open(all\_comments\_path, "w", encoding="utf-8") as f:

json.dump(all\_comments, f, ensure\_ascii=False, indent=4)

print(f"\n所有评论已保存至：{all\_comments\_path}")

print(f"共爬取 {len(all\_comments)} 条评论。")

# 关闭浏览器

driver.quit()

print("Four Over!!!")

print("Four Over!!!")

代码前部分的注释为测试阶段，读者可以注释掉后面的去一个一个测试，最终爬取的程序就是上传程序没有注释代码  
由于上面代码后面评论页数不足150页会重复爬取最后一页数据直到爬取满150页，所以下面是更新后的代码：

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

from selenium.webdriver.common.keys import Keys

import time

import json

import os

import random

# # 获取配置对象 => 什么样的浏览器就选择什么浏览器配置

# option = webdriver.ChromeOptions()

# option.add\_experimental\_option("detach", True)

#

# # 获取driver对象, 并将配置好的option传入进去

# driver = webdriver.Chrome(options=option)

# driver.get('https://you.ctrip.com/sight/antu643/136032.html?scene=online')

# time.sleep(10)

# print("=========第一次尝试=========")

# # 新增爬取文本的操作

# try:

# # 定位元素

# element = driver.find\_element("xpath", '//\*[@id="commentModule"]/div[5]/div[1]')

#

# # 获取文本内容

# text\_content = element.text

#

# # 打印提取的文本

# print("提取到的文本内容：")

# print(text\_content)

#

# except Exception as e:

# print(f"爬取文本时出现错误: {e}")

# print("=========First Over!!!!=========")

# time.sleep(20)

# 新增爬取文本的操作

# print("=========第二次尝试=========")

# try:

# # 定位所有匹配的评论元素（使用 find\_elements 获取列表）

# comment\_elements = driver.find\_elements("xpath", '//\*[@id="commentModule"]/div[5]/div')

#

# # 检查是否找到了评论

# if not comment\_elements:

# print("未找到评论内容！")

# else:

# # 遍历每个评论元素并提取文本

# for idx, element in enumerate(comment\_elements, start=1):

# # 获取评论文本

# text\_content = element.text.strip() # strip() 去除多余空格

# print(f"评论 {idx}:")

# print(text\_content)

# print("-" \* 50) # 分隔线

#

# except Exception as e:

# print(f"爬取文本时出现错误: {e}")

# print("=========Second Over!!!=========")

# time.sleep(20)

# print("=========Try Three=========")

# # 新增爬取文本的操作

# try:

# # 定位所有匹配的评论元素（使用 find\_elements 获取列表）

# comment\_elements = driver.find\_elements("xpath", '//\*[@id="commentModule"]/div[5]/div')

#

# # 检查是否找到了评论

# if not comment\_elements:

# print("未找到评论内容！")

# else:

# comments\_list = [] # 存储所有评论的列表

#

# # 遍历每个评论元素并提取文本

# for idx, element in enumerate(comment\_elements, start=1):

# # 获取评论文本

# text\_content = element.text.strip() # strip() 去除多余空格

# print(f"评论 {idx}:")

# print(text\_content)

# print("-" \* 50) # 分隔线

#

# # 添加到评论列表

# comments\_list.append({

# "id": idx,

# "content": text\_content

# })

#

# # 保存为 JSON 文件（项目根目录）

# json\_file\_path = os.path.join(os.getcwd(), "comments.json")

#

# with open(json\_file\_path, "w", encoding="utf-8") as f:

# json.dump(comments\_list, f, ensure\_ascii=False, indent=4) # 格式化输出

#

# print(f"评论数据已保存至：{json\_file\_path}")

#

# except Exception as e:

# print(f"爬取文本时出现错误: {e}")

# print("========Three Over========")

print("========Four Start========")

# 配置浏览器

option = webdriver.ChromeOptions()

option.add\_experimental\_option("detach", True)

# 添加随机User-Agent

user\_agents = [

"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36",

"Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0",

"Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_15\_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/14.0.3 Safari/605.1.15"

]

option.add\_argument(f"user-agent={random.choice(user\_agents)}")

# 禁用自动化特征检测

option.add\_argument("--disable-blink-features=AutomationControlled")

driver = webdriver.Chrome(options=option)

# 使用已有的date目录

date\_dir = os.path.join(os.getcwd(), "date\_CBSXP")

if not os.path.exists(date\_dir):

raise Exception(f"目录 {date\_dir} 不存在，请先创建该目录")

print(f"将使用已有目录: {date\_dir}")

# 访问目标页面

driver.get('https://you.ctrip.com/sight/fusong2479/136039.html?scene=online')

time.sleep(10 + random.randint(1, 5)) # 随机初始等待

# 获取最大页数

def get\_max\_pages():

try:

max\_page\_element = WebDriverWait(driver, 15).until(

EC.presence\_of\_element\_located((By.XPATH, '//\*[@id="commentModule"]/div[6]/ul/li[8]/a'))

)

max\_pages = int(max\_page\_element.text)

print(f"检测到最大页数: {max\_pages}")

return max\_pages

except Exception as e:

print(f"获取最大页数失败: {e}, 使用默认值150")

return 150 # 如果获取失败，使用默认值

# 存储所有评论

all\_comments = []

max\_pages = get\_max\_pages() # 动态获取最大页数

current\_page = 1

try:

while current\_page <= max\_pages:

print(f"\n========= 正在爬取第 {current\_page} 页(共{max\_pages}页) =========")

# 随机滚动页面模拟人工操作

for \_ in range(3):

driver.execute\_script("window.scrollBy(0, 500)")

time.sleep(random.uniform(0.5, 2))

# 提取当前页的评论

try:

comment\_elements = WebDriverWait(driver, 15).until(

EC.presence\_of\_all\_elements\_located((By.XPATH, '//\*[@id="commentModule"]/div[5]/div'))

)

except Exception as e:

print(f"定位评论元素失败: {e}")

comment\_elements = []

if not comment\_elements:

print("未找到评论内容！尝试刷新页面...")

driver.refresh()

time.sleep(10)

continue

else:

comments\_list = [] # 当前页评论列表

for idx, element in enumerate(comment\_elements, start=1):

try:

text\_content = element.text.strip()

print(f"评论 {idx}:")

print(text\_content[:50] + "...") # 只打印前50个字符

print("-" \* 50)

# 添加到当前页评论列表

comments\_list.append({

"page": current\_page,

"id": idx,

"content": text\_content

})

except Exception as e:

print(f"提取评论{idx}时出错: {e}")

# 添加到总评论列表

all\_comments.extend(comments\_list)

# 每页保存到date目录

json\_file\_path = os.path.join(date\_dir, f"comments\_page\_{current\_page}.json")

with open(json\_file\_path, "w", encoding="utf-8") as f:

json.dump(comments\_list, f, ensure\_ascii=False, indent=4)

print(f"第 {current\_page} 页评论已保存至：{json\_file\_path}")

# 检查是否已达到最大页数

if current\_page >= max\_pages:

print(f"已达到最大爬取页数 {max\_pages}，停止爬取。")

break

try:

# 随机等待时间

wait\_time = 10 + random.randint(-5, 5)

print(f"等待{wait\_time}秒后继续下一页...")

time.sleep(wait\_time)

# 尝试翻页

for attempt in range(3): # 最多尝试3次

try:

# 定位页码输入框

page\_input = WebDriverWait(driver, 15).until(

EC.presence\_of\_element\_located(

(By.XPATH, '//\*[@id="commentModule"]/div[6]/ul/li[10]/div/input'))

)

# 定位跳转按钮

page\_button = WebDriverWait(driver, 15).until(

EC.element\_to\_be\_clickable(

(By.XPATH, '//\*[@id="commentModule"]/div[6]/ul/li[10]/div/span/button'))

)

# 输入下一页页码

for \_ in range(3): # 确保输入成功

page\_input.clear()

page\_input.send\_keys(str(current\_page + 1))

time.sleep(0.5)

# 随机点击延迟

time.sleep(random.uniform(0.5, 2))

# 点击跳转按钮

driver.execute\_script("arguments[0].click();", page\_button)

# 等待页面加载

time.sleep(5 + random.randint(1, 3))

# 验证是否翻页成功

try:

WebDriverWait(driver, 10).until(

EC.presence\_of\_element\_located((By.XPATH, f'//\*[@id="commentModule"]/div[5]/div'))

)

current\_page += 1

break # 翻页成功，退出重试循环

except:

if attempt == 2:

raise Exception("翻页后未加载新内容")

continue

except Exception as e:

if attempt == 2: # 最后一次尝试

raise

print(f"翻页尝试 {attempt + 1} 失败: {e}")

time.sleep(5)

driver.refresh()

time.sleep(5)

except Exception as e:

print(f"翻页最终失败: {e}")

break

finally:

# 保存所有评论到date目录

all\_comments\_path = os.path.join(date\_dir, "all\_comments.json")

with open(all\_comments\_path, "w", encoding="utf-8") as f:

json.dump(all\_comments, f, ensure\_ascii=False, indent=4)

print(f"\n所有评论已保存至：{all\_comments\_path}")

print(f"共爬取 {len(all\_comments)} 条评论。")

# 关闭浏览器

driver.quit()

print("Four Over!!!")

print("Four Over!!!")

最后让我们感谢工程师！！！！

​