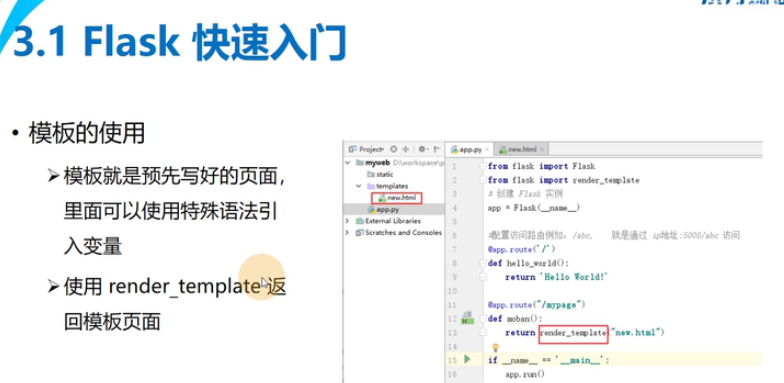
# Flask搭建web/Echarts可视化大屏

# 



1. 新建一个flask项目，也可以新建一个python文件

具体代码：

**from** flask **import** Flask

app = Flask**(**\_\_name\_\_**)**#创建了一个flask实例

@app.route**(**"/"**) #**我们使用 [route()](https://dormousehole.readthedocs.io/en/latest/api.html#flask.Flask.route) 装饰器来告诉 Flask 触发函数 URL 。

**def** hello\_world**():**

**return** "<p>Hello, World!</p>"

函数返回需要在用户浏览器中显示的信息。默认的内容类型是 HTML ，因此字 符串中的 HTML 会被浏览器渲染。

if \_\_name\_\_ == '\_\_main\_\_':

app.run()

这样就启动了一个非常简单的内建的服务器。现在在浏览器中打开 http://127.0.0.1:5000/ ，应该可以看到 Hello World! 字样。

2.我们另外在复制要给路由（）：具体代码如下

@app.route('/abc')

def hell\_word1():

return f"""

<form action="/login">

账号：<input ><br>

密码：<input

>

<input type="submit">

</form>

"""

如果要获取参数，我们要引入包：from flask import request #获取参数，然后修改我们的路由代码：

@app.route('/abc')

def hell\_word1():

id=request.values.get("id") #flask的get传递参数

return f"""

<form >

账号：<input name="name" value="{id}"><br>

密码：<input name="pwd">

</form>

"""

调试时使用：?=10086,传递参数

3.我们再定义一个路由：

新的路由：

@app.route('/login')

def hello\_world2():

name=request.values.get("name")

pwd=request.values.get("pwd")

return f"name={name},pwd={pwd}"

@app.route('/abc')

def hell\_word1():

id=request.values.get("id")

return f"""

<form action="/login">

账号：<input name="name" value="{id}"><br>

密码：<input name="pwd">

<input type="submit"> #要发送一个请求要再定义一个按钮

</form>

"""

如果我们整夜页面都是字符串形式，这是不可能的，问哦们可以使用flask的模块：

1. 首先我们先构建一个页面：在template文件夹里新建一个index.html页面，具体内容如下：

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>my page</title>

</head>

<body>

<h1> 疫情追踪 </h1>

<h3>实时报道</h3>

</body>

</html>

1. 再回到py里面再定义一个路由为tem ，

导入包：from flask import render\_template

@app.route('/tem')

def hell\_world3():

return render\_template("index.html") #这里返回一个模板

这就是flask的快速入门。

4. **如果我们只要局部的变化而不是整个页面的发送，那么我们就要用AJIAX发送异步请求，我们要下载jquery框架。方便编写ajax代码** 基本格式：



1. 修改我们的index.html引入js文件：

<script src="../static/js/jquery.js"></script>

修改我们里面正文代码：

<head>

<meta charset="UTF-8">

<title>my page</title>

<script src="../static/js/jquery.js"></script>

</head>

<body>

<h1> 疫情追踪 </h1>

<h3>实时报道</h3>

<button>点我</button>

<script>

$("button").click(function(){

$.ajax({

url:"/ajax",

type:"post",

data:{"name":"张某","score":89},

success:function(d){

$("h1").html("感染人数"+d);

},error:function() {

alert('发送ajax失败 ');

}

})

})

1. 前台完成后我们写后台，在py里面，再来一个路由：

@app.route('/ajax',methods=["get","post"]) #定义请求方式

def hell\_world4():

name=request.values.get("name")

score=request.values.get("score")

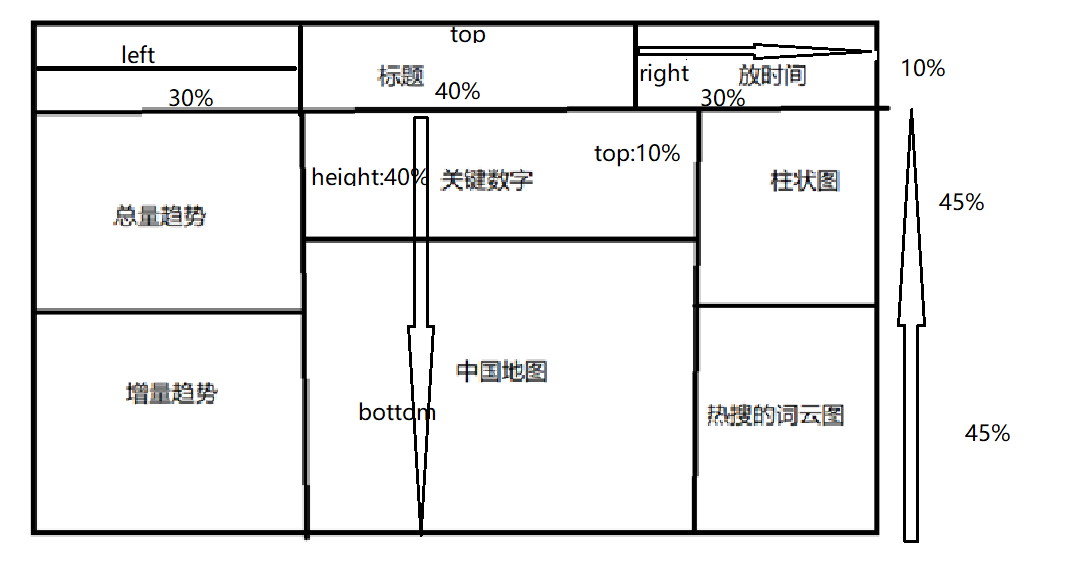
print(f"name:{name},score:{score}")

return '10000'

运行后，发现传递的数据也成功了。



一、规划板块（使用画图程序）



1. 要实现这个布局我们用到h5绝对定位划分版块

#title{

color:#FFFFFF; /\*设置字体\*/

position:absolute; /\*绝对定位\*/

height:10%;

width:40%;

left:30%;

top:0;

/\*居中分布,弹性布局\*/

Display:flex;

Align-items:center;

Justify-content:center;

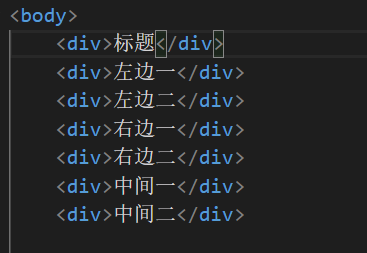
}

1. 使用hbuild 或者vscode，其它网页编辑工具。甚至记事本都可以
2. 在static文件夹里新建一个css文件夹，放样式
3. 在templeat文件夹里新建一个main.html用来运行我们大屏图像

# 注意：vscode默认的html格式生成

**创建html页面，输入！，点击tab键（enter）即可自动生成**

1. **按照上图的版块进行划分出来**

****

# VScode如何在浏览器中打开html文件

### 2.点击左侧的扩展，打开扩展栏

**3.在扩展栏的搜索栏中输入open in browser，找到open in browser这款插件，点击右下角“安装”字样即可安装。**

**（二）**

**为了美观，我们在main.html写一个style样式，先在 css文件夹里新建一个main.css**

**然后**<link href="../static/css/main.css" rel=”stylesheet>

**此处一定要注明它的样式类型，是样式表。否则的话即使你从style引入了它的样式也不会生效。**

**<style>**

**</style>**

**（三）然后我们在main.css里面设置这7个版块的位置：具体代码如下：**

**先做第一个：**

**#title{**

**position: absolute;**

**width: 40%;**

**height: 10%;**

**top: 0;**

**left: 30%;**

**background: #666666;**

**}**

**然后应用到第一个div,看看效果**

 <div id="title">标题</div>

**最后在css复制7个**

**分别命名：**

#title{

    position: absolute;

    width: 40%;

    height: 10%;

    top: 0;

    left: 30%;

    background: #666666;

}

#time{

    position: absolute;

    width: 30%;

    height: 10%;

    top: 0;

    background: green;

}

#c1{

    position: absolute;

    width: 40%;

    height: 30%;

    top: 10%;

    left: 30%;

    background: #777777;

}

#c2{

    position: absolute;

    width: 40%;

    height: 60%;

    top: 40%;

    left: 30%;

    background: #888888;

}

#l1{

    position: absolute;

    width: 30%;

    height: 45%;

    top: 10%;

    left: 0;

    background: #666666;

}

#l2{

    position: absolute;

    width: 30%;

    height: 45%;

    top: 55%;

    left: 0;

    background: #777777;

}

#r1{

    position: absolute;

    width: 30%;

    height: 45%;

    top: 10%;

    right:0;

    background: #666666;

}

#r2{

    position: absolute;

    width: 30%;

    height: 45%;

    top: 55%;

    right: 0;

    background: #777777;

}

**Main.html**

**<body>**

**<div id="title">全国疫情事实追踪</div>**

**<div id="time">日期时间</div>**

**<div id="c1">中间一</div>**

**<div id="c2">中间二</div>**

**<div id="l1">左边一</div>**

**<div id="l2">左边二</div>**

**<div id="r1">右边一</div>**

**<div id="r2">右边二</div>**

**这样我们的大屏就布局好了。**

**（四）在main.css增加body样式，并设置标题弹性布局，修改字体颜色和大小**

**body{**

**margin: 0;/\*定义无边界\*/**

**background: #333;**

**}**

**#title{**

position: absolute;

width: 40%;

height: 10%;

top: 0;

left: 30%;

/\*background: #666666;\*/

color: white;

font-size: 30px;

**/\*使用弹性布局\*/**

**display: flex;/\*使用弹性布局\*/**

**align-items: center;** **/\*水平居中对齐\*/**

**justify-content: center;** **/\*垂直居中对齐\*/**

**}**

**二、可视化大屏制作**

****

**（一）中间数字修改**

**1.在main.css增加设置数字和文字的样式**

**.num {**

**width: 25%;**

**float: left;**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**color: gold;**

**font-size: 20px;**

**}**

**.txt{**

**width: 25%;**

**float: left;**

**font-family: "幼圆";**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**}**

**2.在main.html中id=c1的放数字的块中继续嵌套四个小块，应用样式class=num,字体是标题1，应用txt 标题2**

**<div id="c1">**

**<div class="num"><h1>1234</h1></div>**

**<div class="num"><h1>1234</h1></div>**

**<div class="num"><h1>1234</h1></div>**

**<div class="num"><h1>1234</h1></div>**

**<div class="txt"><h2>累计确诊</h2></div>**

**<div class="txt"><h2>剩余疑似</h2></div>**

**<div class="txt"><h2>累计治愈</h2></div>**

**<div class="txt"><h2>累计死亡</h2></div>**

**</div>**

**查看效果**

1. **我们修改css,将c1背景去掉，字体颜色设置为白色**

**#c1{**

**position: absolute;**

**width: 40%;**

**height: 30%;**

**top: 10%;**

**left: 30%;**

**color: white;**

**/\* background: #777777;\*/**

**}**

**运行就可以了，以后我们就可以后台发送ajax请求，动态修改里面的数字。**

**（二）时间设置**

**1.修改我们在css中dada样式：**

**#time{**

**position: absolute;**

**/\*width: 30%;\*/**

**height: 10%;**

**top: 5%;**

**right: 2%;**

**font-size: 20px;**

**color: #FFFFFF;**

**/\* background: green; \*/**

**}**

**（三）在main.html中插入javascript发请求**

**1.先导入库：**

**<title>疫情监控</title>**

**<script src="../static/js/jquery.js"></script>**

1. **在body插入script代码**

**<script>**

**function gettime(){**

**$.ajax({**

**url:"/time",**

**timeout:10000,//超时时间设置为10秒**

**success:function(data){**

**$("#time").html(data) #这里设置选择页面中的样式为time元素，在页面显示time内容。**

**},**

**error:function(xhr,type,errorThorwn){**

**}**

**});**

**}**

**setInterval(gettime,1000) //每隔1秒就调用时间函数**

**</script>**

**4.我们定义一个时间路由(与ajax的url一致)，然后定义一个方法，用它调用显示时间函数。**

**#（1）我们再新建一个工具类utils.py放我们编写的工具接口等。修改f1.py文件的root**

**import time**

**def get\_time():**

**time\_str=time.strftime("%Y{}%m{}%d{} %X")**

**return time\_str.format("年","月","日")**

**#做一个测试**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print(get\_time())**

**回到app.py后台，我们import utils使用里面的函数。**

**#（2）定义一个路由**

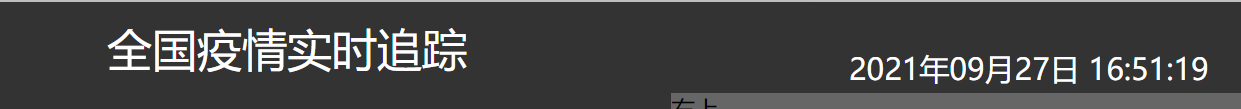
@app.route('/time',methods=["get","post"])  
def get\_time():  
 return utils.get\_time()

**#（3）修改5000端口路由，然其返回 main.html页面**

**@app.route('/')**

**def hell\_world():**

**return render\_template("main.html")**

**运行py后台，时间每隔1秒就刷新：**

**（四）中间数据的获取**

**1.在utils.py编写获取数据函数，从数据库里面获取（之前爬取的数据）**

**（1）导入包**

**import pymysql #与数据库交互需要用到**

**（2）定义封装查询函数：**

**#2.获取数字函数(封装一个模块)**

**def get\_conn(): #封装连接**

**"""**

**:return: 连游标**

**"""**

**#创建连接**

**conn=pymysql.connect(host="root",**

**user="root",**

**password='root',**

**db="cov",**

**charset='utf8')**

**#创建游标**

**cursor=conn.cursor()#执行完毕返回的结果集默认以元组显示**

**return conn,cursor**

**def close\_conn(conn,cursor): #执行完毕关闭连接和游标**

**cursor.close()**

**conn.close()**

**def query(sql,\*args):#封装查询**

**"""**

**封装通用查询**

**:param sql:**

**:param args:**

**:return: 返回查询到的结果((),())的元组形式**

**"""**

**conn,cursor=get\_conn()**

**cursor.execute(sql,args)**

**res=cursor.fetchall()**

**close\_conn(conn,cursor)**

**return res**

**def test():**

**sql="select \* from details"**

**res=query(sql)**

**return res (或者res[0])**

**#中间4个数据**

**def get\_c1\_data():**

**sql = "select sum(confirm)," \**

**"(select suspect from history order by ds desc limit 1)," \**

**"sum(heal),sum(dead) from details " \**

**"where update\_time=(select update\_time from details order by update\_time desc limit 1) "**

**res = query(sql)**

**return res[0]**

**(3)测试utils.py**

#做一个测试  
if \_\_name\_\_ == '\_\_main\_\_':  
 print(get\_c1\_data())  
 # print(test())  
 #print(get\_time())

**（3）修改app.py**

**#1. 引入包**from flask import jsonify #返回json数据

**#2. 定义新路由，并定义函数**

**就是在utils封装查询数据库函数，然后在app.py注册新的路由，并定义函数**

**@app.route('/c1',methods=["get","post"])**

**def get\_c1\_data():**

**data=utils.get\_c1\_data()**

**#dada 是元组，利用jsonify转换为字典**

**return jsonify({"confirm":data[0],"suspect":data[1],"heal":data[2],"dead":data[3]})**

**（4）回到main.html编写获取数字的ajax代码：**

**在setInterval(gettime,1000) 代码上一行编写：**

**function get\_c1\_data(){**

**$.ajax({**

**url:"/c1",/\*数据发送给c1路由\*/**

**success:function(data){**

**$(".num h1").eq(0).text(data.confirm);**

**$(".num h1").eq(1).text(data.suspect);**

**$(".num h1").eq(2).text(data.heal);**

**$(".num h1").eq(3).text(data.dead)**

**},**

**error:function(xhr,type,errorThrown){**

**}**

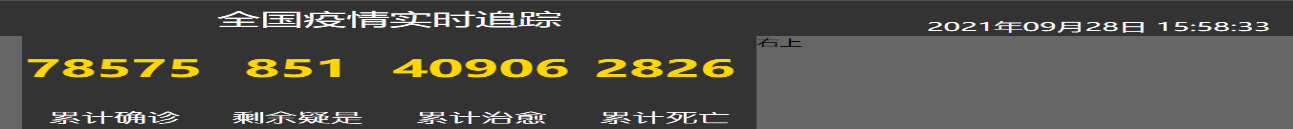
**})**

**}**

**setInterval(gettime,1000) /\*每隔1秒调用gettim函数\*/**

**setInterval(get\_c1\_data,1000) /\*每隔1秒调用get\_c1\_data函数\*/**

**然后，运行后台app.py，就可以发现数据改变了。**

****

**（五）echarts图（是Java写的）的插入，就是左边右边中下图的更新。**

****

**Echarts官网https://echarts.apache.org/zh/index.html**

****

**然后再最下面5分钟上手看看学习：**

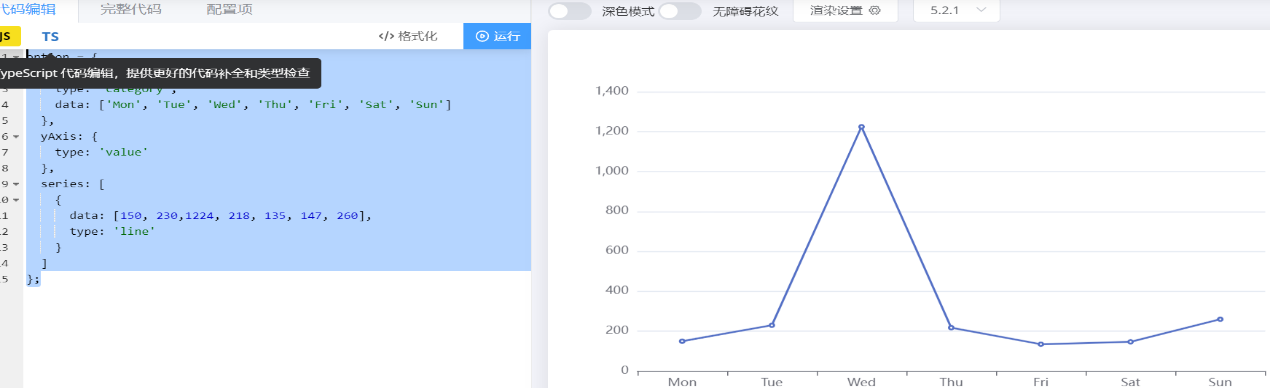
****

****

****

****

**实际上我们可以复制示例中的option就可以了。**

****

1. **做一个测试，熟悉echarts编写过程，打开我们的项目的main.html，复制上面的option**

**#1.导入js包，**

**china.js , 引入此文件，引入中国地图**

**worldcloud.js,** 创建词云图

**echarts.min.js引入此文件，创建echart实例**

**#2.新建一个test.html我们用来做测试引入echarts.min.js**

**<script src="../static/js/echarts.min.js"></script>**

**#3引入固定宽高的容器（div）**

**<body> /\*定义一个容器，指定高和宽否则显示不了图\*/**

**<div id="main" style="width: 600px;height: 800px;"></div>#**

**#4.初始化一个图像对象**

**<script>**

**/\*实例化一个图对象,并初始化\*/**

**Var ectest=echarts.init(document.getElementById('main'))**

**#5,将刚刚示列option复制过来**

**var option= {**

**xAxis: {**

**type: 'category',**

**data: ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun']**

**},**

**yAxis: {**

**type: 'value'**

**},**

**series: [ {**

**data: [150, 230, 1224, 218, 135, 147, 260],**

**type: 'line'**

**}]**

**};**

**#6.使用刚刚使用实例化对象的setOption(option)显示图像\*/**

**ectest.setOption(option)**

**#7.我们打开浏览器看看效果。就是我们看到的案例。我们可以修改option,data值。图跟着变化。**

**2.绘制中国地图：**

****

**#1.首先新建一个ec\_center.js文件，将案例里面的option复制过来。**

**具体代码：详看ec\_center.js。建议每个图都建立自己的js文件。**

**//初始化echarts实例**

**var ec\_center = echarts.init(document.getElementById("c2"),"dark");////dark 主题颜色，默认浅色,大家可以在官方下载主题，在下载-主题下载，除了签名的两种，都必须讲对应的js文件导入进来才行。**

**var mydata = []**

**var optionMap = {**

**title: {**

**text: '',**

**subtext: '',**

**x: 'left'**

**},**

**tooltip: {**

**trigger: 'item'**

**},**

**//左侧小导航图标，图例**

**visualMap: {**

**show: true,**

**x: 'left',**

**y: 'bottom',**

**textStyle: {**

**fontSize: 8**

**},**

**//图列**

**splitList: [{start: 1,end: 9},{start: 10,end: 99},{start: 100,end: 999},{start: 1000,end: 9999},{start: 10000}],**

**color: ['#8A3310','#C64918', '#E55B25','#F2AD92', '#F9DCD1']**

**},**

**//配置属性**

**series: [{**

**name: '累积确诊人数',**

**type: 'map',**

**mapType: 'china',**

**roam: false,**

**itemStyle: {**

**normal: {**

**borderWidth: .5,**

**borderColor: '#009fe8',**

**areaColor: '#ffefd5'**

**},**

**emphasis: {**

**borderWidth: .5,**

**borderColor: '#4b0082',**

**areaColor: '#fff'**

**}**

**},**

**label: {**

**normal: {**

**show: true, //省份名称**

**fontSize: 8**

**},**

**emphasis: {**

**show: true,**

**fontSize: 8**

**}**

**},**

**data: mydata //数据**

**}]**

**};**

**//使用制定的配置项和数据显示图表**

**ec\_center.setOption(optionMap);**

**#2. 在main.html导入包china.js**

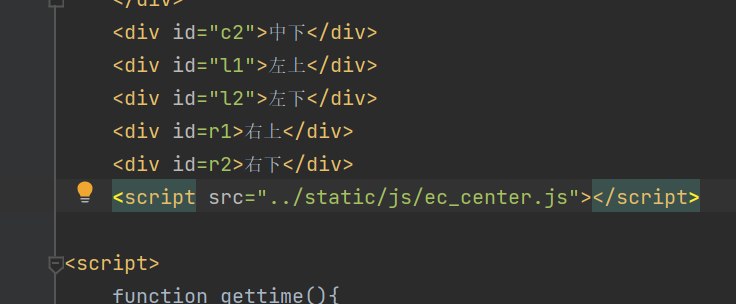
**<script src="../static/js/jquery.js"></script>**

**<script src="../static/js/china.js"></script>**

**<script src="../static/js/echarts.min.js"></script>**

**<script src="../static/js/worldcloud.js"></script>**

**# .在我们body 里面引入我们定义的ec\_center.js包**

****

**运行html.发现图已经上去了。**

****

**#3.编写ajax通过后台数据库，sql查询获取最新数据然后通过转换成前端html能接收的json数据。**

**（1） 首先，将我们刚刚显示的数字和时间的script和</script>中的代码移动到一个新键的controller.js里面去。**

**（2）这样我们在main.html中引入就可以了**

**<script src="../static/js/ec\_center.js"></script>**

**<script src="../static/js/controller.js"></script>**

**以后我们把控制数据显示的功能代码都写入到controller.js里面去。**

**在controller.js添加控制地图疫情数据的ajax代码：**

**function get\_c2\_data(){**

**$.ajax({**

**url:"/c2",**

**/\* 成功将数据回调到页面 \*/**

**success:function(data){**

**/\*获取地图对象数据 option 中的series\*/**

**optionMap.series[0]=data.data**

**/\*将更新数据的optionMap 在地图对象更新 \*/**

**ec\_center.setOption(optionMap);**

**},**

**error:function(xhr,type,errorThrown){**

**}**

**get\_c2\_data() 调用get\_c2\_data**

**（3）编写 util.py中的获取数据库的查询get\_c2\_data**

**def get\_c2\_data():**

**sql = "select province,sum(confirm) from details " \**

**"where update\_time=(select update\_time from details " \**

**"order by update\_time desc limit 1) " \**

**"group by province"**

**res = query(sql)**

**return res**

**#做一个测试**

**if \_\_name\_\_ == '\_\_main\_\_':**

**print(get\_c1\_data())**

**# print(test())**

**#print(get\_time())**

**(4) 编写后台app.py路由和函数**

**@app.route('/c2')**

**def get\_c2\_data():**

**res = []**

**for tup in utils.get\_c2\_data():**

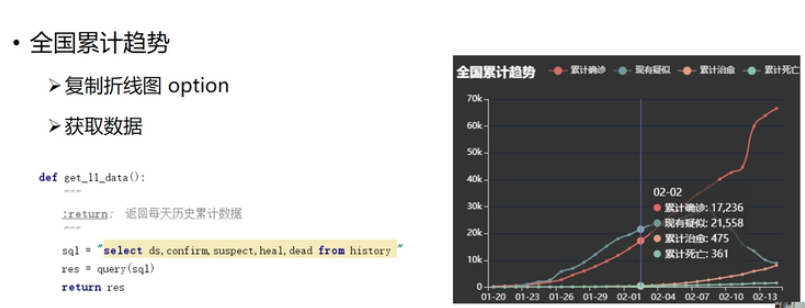
**print(tup)**

**res.append({"name":tup[0],"value":int(tup[1])})**

**return jsonify({"data":res})**

**运行app.py**

**(三)、绘制全国累计趋势图)div=l1**

****

1. **复制折线图option**

**详见：ec\_left1.js，如果要了解里面的配置各个属性可以在官网查看配置项：文档-配置项查看：**

1. **在main.html中，引入我们配置的ec\_left1.js折线图js.然后启动app.py后台服务器，看到图已经加载上来。**
2. **在utils中编写获取数据函数**

**def get\_l1\_data():**

**sql = "select ds,confirm,suspect,heal,dead from history"**

**res = query(sql)**

**return res**

1. **在app.py中定义路由**

**@app.route('/l1')**

**def get\_l1\_data():**

**data = utils.get\_l1\_data()**

**day,confirm,suspect,heal,dead = [],[],[],[],[]**

**for a,b,c,d,e in data[7:]:**

**day.append(a.strftime("%m-%d"))**

**confirm.append(b)**

**suspect.append(c)**

**heal.append(d)**

**dead.append(e)**

**return jsonify({"day":day,"confirm":confirm,"suspect":suspect,"heal":heal,"dead":dead})**

1. **在前台main.html前端编写ajax获取后端数据的请求。（controller.js编写）**

**function get\_l1\_data() {**

**$.ajax({**

**url:"/l1",**

**success: function(data) {**

**option\_left1.xAxis.data = data.day**

**option\_left1.series[0].data = data.confirm**

**option\_left1.series[1].data = data.suspect**

**option\_left1.series[2].data = data.heal**

**option\_left1.series[3].data = data.dead**

**ec\_left1.setOption(option\_left1)**

**},**

**error: function(xhr, type, errorThrown) {**

**}**

**})**

**}**

**//调用函数**

**get\_l1\_data()**

1. **运行后台**

**(四)、绘制l2,因为与l1类似，我们直接复制l1过来**

**1.在utils.py直接复制get\_l1\_data() 函数改名def get\_l2\_data()，修改字段名**

**def get\_l2\_data():**

**sql = "select ds,confirm\_add,suspect\_add from history"**

**res = query(sql)**

**return res**

**2.在main.py后台增加l2路由**

@app.route('/l2')

def get\_l2\_data():

data = utils.get\_l2\_data()

day,confirm\_add,suspect\_add = [],[],[]

for a,b,c in data[7:]:

day.append(a.strftime("%m-%d"))

confirm\_add.append(b)

suspect\_add.append(c)

return jsonify({"day":day,"confirm\_add":confirm\_add,"suspect\_add":suspect\_add})

**3.在controller.js，编写获取后台l1数据的ajax代码(复制get\_l1\_data )get\_l2\_data：**

function get\_l2\_data() {

$.ajax({

url:"/l2",

success: function(data) {

option\_left2.xAxis.data = data.day

option\_left2.series[0].data = data.confirm\_add

option\_left2.series[1].data = data.suspect\_add

ec\_left2.setOption(option\_left2)

},

error: function(xhr, type, errorThrown) {

}

})

}

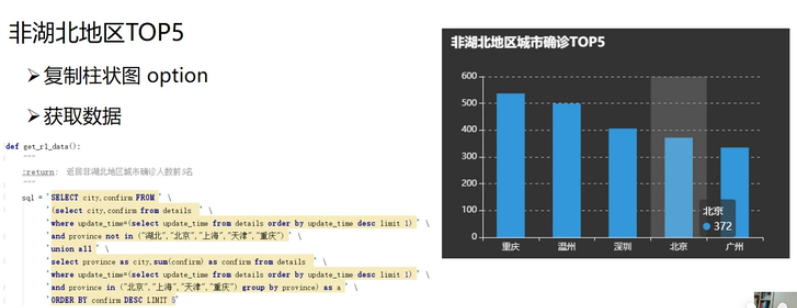
//调用函数

get\_l2\_data()

效果图：



1. **、右1柱状图。**



1. **复制柱状图option**

**ec\_right1.js**

1. **获取数据**

**def get\_r1\_data():**

**sql = 'select city,confirm from ' \**

**'(select city,confirm from details ' \**

**'where update\_time=(select update\_time from details order by update\_time desc limit 1) ' \**

**'and province not in ("湖北","北京","上海","天津","重庆") ' \**

**'union all ' \**

**'select province as city,sum(confirm) as confirm from details ' \**

**'where update\_time=(select update\_time from details order by update\_time desc limit 1) ' \**

**'and province in ("北京","上海","天津","重庆") group by province) as a ' \**

**'order by confirm desc limit 5'**

**res = query(sql)**

**return res**

1. **在main.py中定义路由，调用数据库查询函数get\_r1\_data。**

**@app.route('/r1')**

**def get\_r1\_data():**

**data = utils.get\_r1\_data()**

**city = []**

**confirm = []**

**for k,v in data:**

**city.append(k)**

**confirm.append(int(v))**

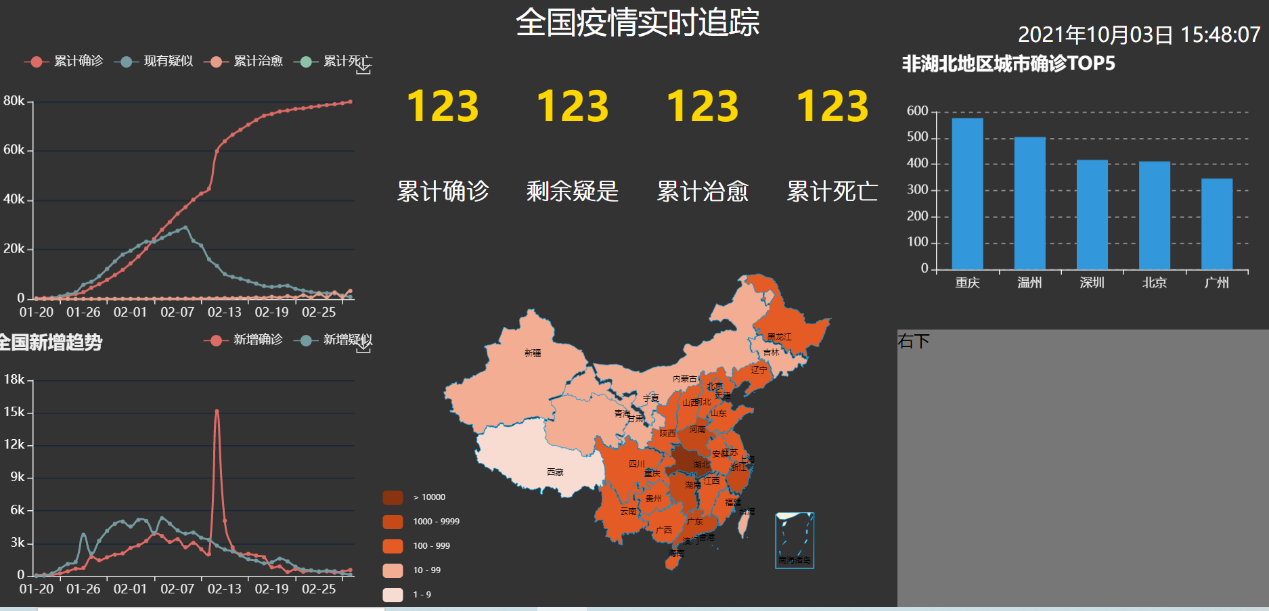
**return jsonify({"city": city,"confirm": confirm})**

1. **在controller.js，编写获取后台r1数据的ajax代码(复制get\_r1\_data )get\_r2\_data**
2. function get\_r1\_data() {  
    $.ajax({  
    url:"/r1",  
    success: function(data) {  
    option\_right1.xAxis.data = data.city  
    option\_right1.series[0].data = data.confirm  
    ec\_right1.setOption(option\_right1)  
    },  
    error: function(xhr, type, errorThrown) {  
     
    }  
    })  
   }

**//调用函数：**

**get\_r1\_data()**

**效果如下：**

****

1. **右2词云图：**
2. **需要在main.html引入包：worldcloud.js(其实在官网也可以下载：在下载专区-扩展下载-字符云：就是我们所要下载的素材，点击进入github（code-downlaod.zip）我们下载就可以使用了。)**

**<script src="../static/js/worldcloud.js"></script>**

1. **编写ec\_right2.js.设置option具体代码如下：参看ec\_right2.js**
2. **在utils.py编写获取数据的查询。 get\_r2\_data():**

**#返回最近的20条热搜，降序**

**def get\_r2\_data():**

**sql = "select content from hotsearch order by id desc limit 20"**

**res = query(sql)**

**return res**

1. **在后台app.py配置r2路由并编写调用get\_r2\_data数据函数：**

**#1.导入结巴库，导入字符串库，数字库**

**import string**

**import decimal #十进制计算**

**from jieba.analyse import extract\_tags**

**说明：**jieba.analyse.extract\_tags()提取关键字的原理是使用TF-IDF算法

**@app.route('/r2')  
def get\_r2\_data():  
 data = utils.get\_r2\_data() #格式如：(('安徽全省退出高风险地区51112',), ('大邱医院改造集装箱收治患者55592',))  
 d= []  
 for i in data:  
 k = i[0].rstrip(string.digits)#从右边截取数字出来，得到文字  
 v=i[0][len(k):] #获取热搜数字  
 ks=extract\_tags(k) #使用结巴读取关键字  
 for j in ks:  
 if not j.isdigit(): #如果不是数字就是关键次  
 d.append({"name": j, "value": v})  
 return jsonify({"kws":d})**

1. **在前端编写获取后端数据的ajax.js（在controller.js编写）.然后引入在main.html即可**

**//获取右2后台数据的ajax请求。**

**function get\_r2\_data() {**

**$.ajax({**

**url:"/r2",**

**success: function(data) {**

**option\_right2.series[0].data = data.kws**

**ec\_right2.setOption(option\_right2)**

**},**

**error: function(xhr, type, errorThrown) {**

**}**

**})**

**}**

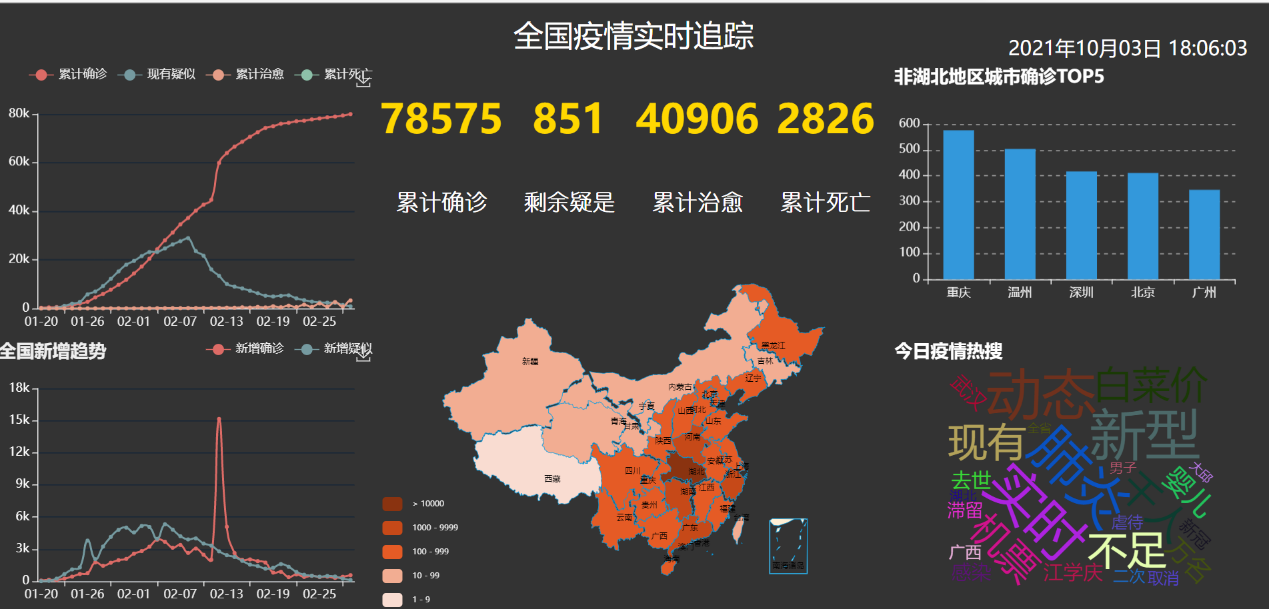
**//调用函数**

**get\_r2\_data()**

1. **在main.html引入包：**

**<script src="../static/js/ec\_right2.js"></script>**

1. **运行后端app.py.即可。**

****

**清除div中的数字，设置各个模块函数调用的时间间隔，形成动态刷新**

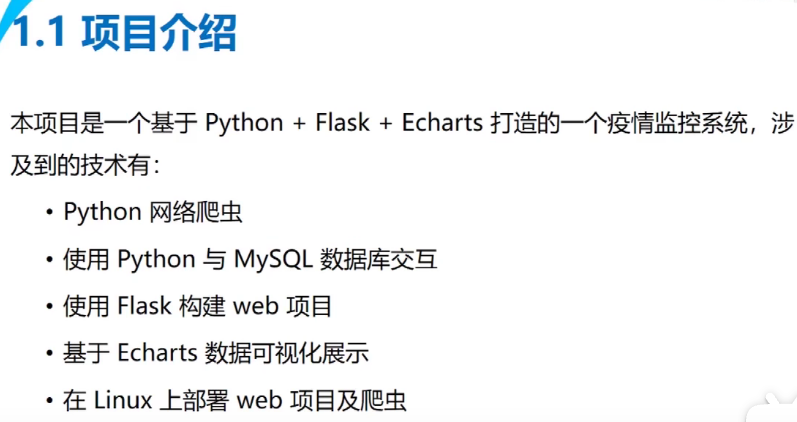
1. 项目部署

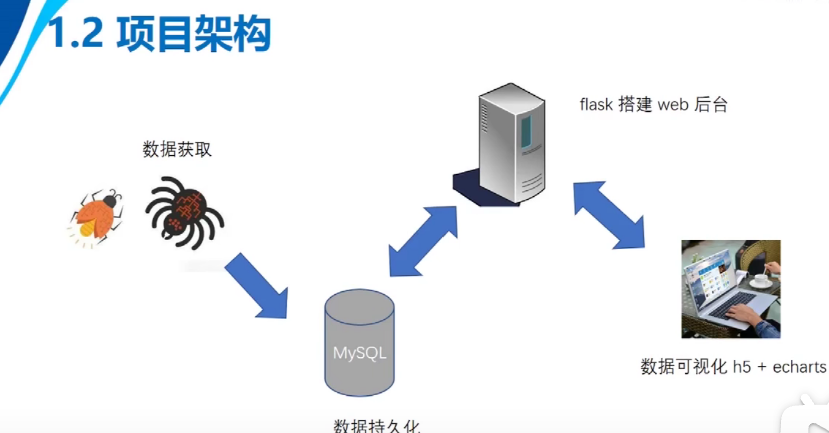


安装：

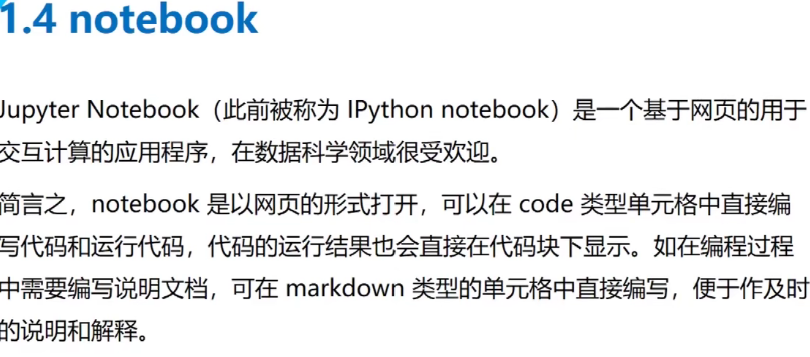


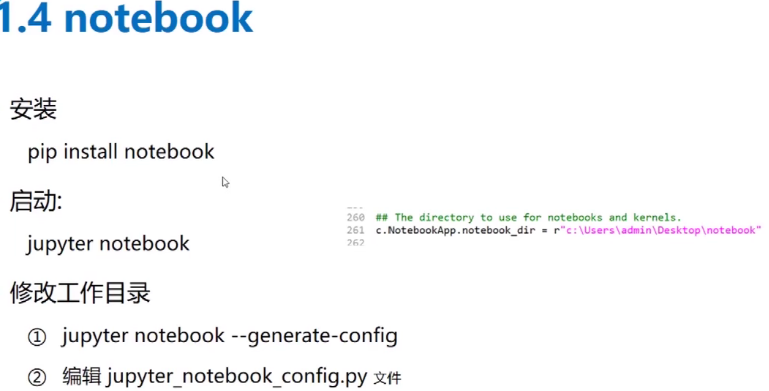




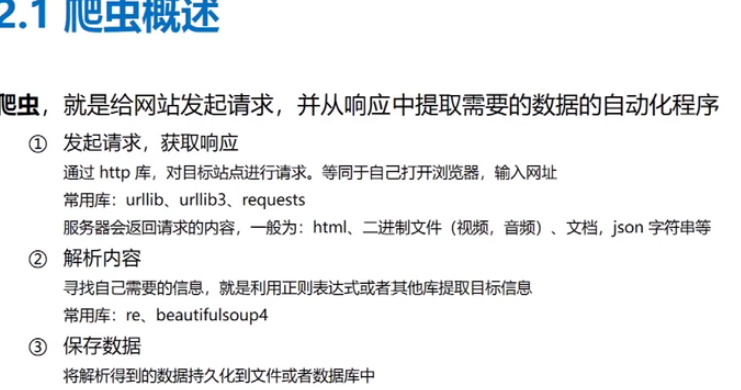


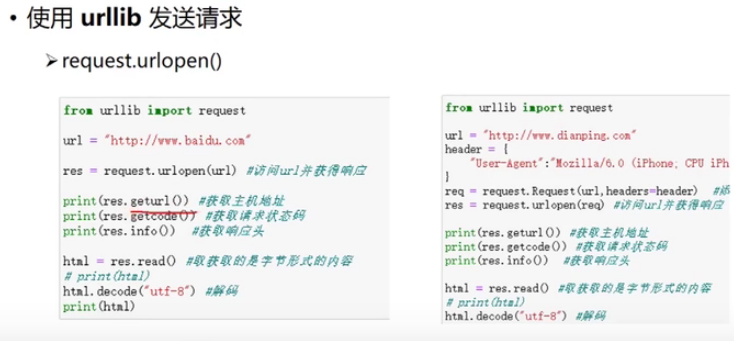


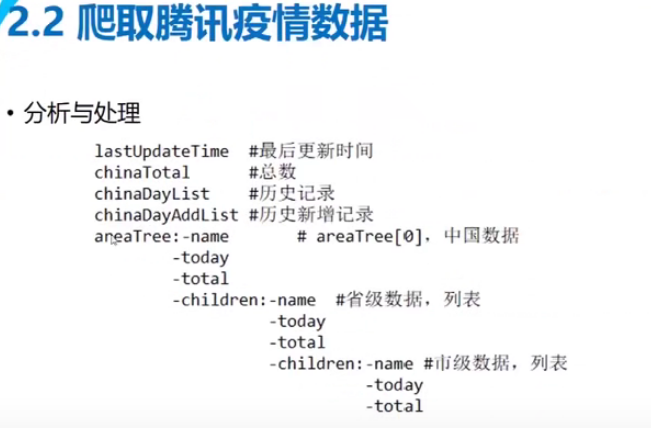
















1. **爬取腾讯疫情数网站网址：**

url1 = “<https://news.qq.com/zt2020/page/feiyan.htm#/>”

url2=””

1. 打开腾讯疫情网站后台，我们可以看到，点击“控制台”可以看到“全局数据”- areaTree- children，里面34个省份的疫情数据 。直接利用腾讯数据，写入数据库就可以了。
2. 比较通用的方法，就是运行网络，然后查看js文件就可以了（点击筛选器下面的js）。

Get开头的都是获取数据的，我们点击在预览查看信息。

url= <https://view.inews.qq.com/g2/getOnsInfo?name=disease_h5>

我们也可以复制预览里面的值在记事本打开。可以查看里面的信息

具体代码：

import sys

import time

import pymysql

import json

import traceback

import requests

from selenium.webdriver import Chrome,ChromeOptions

def get\_tencent\_data():

url1 = "https://view.inews.qq.com/g2/getOnsInfo?name=disease\_h5"

url2 = "https://view.inews.qq.com/g2/getOnsInfo?name=disease\_other"

headers = {

"user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.122 Safari/537.36"

}

r1 = requests.get(url1, headers)

r2 = requests.get(url2, headers)

res1 = json.loads(r1.text)

res2 = json.loads(r2.text)

data\_all1 = json.loads(res1["data"])

data\_all2 = json.loads(res2["data"])

history = {}

for i in data\_all2["chinaDayList"]:

ds = "2020." + i["date"]

tup = time.strptime(ds, "%Y.%m.%d") # 匹配时间

ds = time.strftime("%Y-%m-%d", tup) # 改变时间格式

confirm = i["confirm"]

suspect = i["suspect"]

heal = i["heal"]

dead = i["dead"]

history[ds] = {"confirm": confirm, "suspect": suspect, "heal": heal, "dead": dead}

for i in data\_all2["chinaDayAddList"]:

ds = "2020." + i["date"]

tup = time.strptime(ds, "%Y.%m.%d") # 匹配时间

ds = time.strftime("%Y-%m-%d", tup) # 改变时间格式

confirm = i["confirm"]

suspect = i["suspect"]

heal = i["heal"]

dead = i["dead"]

history[ds].update({"confirm\_add": confirm, "suspect\_add": suspect, "heal\_add": heal, "dead\_add": dead})

details = []

update\_time = data\_all1["lastUpdateTime"]

data\_country = data\_all1["areaTree"]

data\_province = data\_country[0]["children"]

for pro\_infos in data\_province:

province = pro\_infos["name"]

for city\_infos in pro\_infos["children"]:

city = city\_infos["name"]

confirm = city\_infos["total"]["confirm"]

confirm\_add = city\_infos["today"]["confirm"]

heal = city\_infos["total"]["heal"]

dead = city\_infos["total"]["dead"]

details.append([update\_time, province, city, confirm, confirm\_add, heal, dead])

return history, details

def get\_conn():

# 建立连接

conn = pymysql.connect(host="###", user="###", password="###", db="cov", charset="utf8")

# c创建游标

cursor = conn.cursor()

return conn, cursor

def close\_conn(conn, cursor):

if cursor:

cursor.close()

if conn:

conn.close()

#定义更新细节函数

def update\_details():

cursor = None

conn = None

try:

li = get\_tencent\_data()[1]#1代表最新数据

conn,cursor = get\_conn()

sql = "insert into details(update\_time,province,city,confirm,confirm\_add,heal,dead) values(%s,%s,%s,%s,%s,%s,%s)"

sql\_query = 'select %s=(select update\_time from details order by id desc limit 1)'

#对比当前最大时间戳

cursor.execute(sql\_query,li[0][0])

if not cursor.fetchone()[0]:

print(f"{time.asctime()}开始更新数据")

for item in li:

cursor.execute(sql,item)

conn.commit()

print(f"{time.asctime()}更新到最新数据")

else:

print(f"{time.asctime()}已是最新数据！")

except:

traceback.print\_exc()

finally:

close\_conn(conn,cursor)

#插入历史数据

def insert\_history():

cursor = None

conn = None

try:

dic = get\_tencent\_data()[0]#0代表历史数据字典

print(f"{time.asctime()}开始插入历史数据")

conn,cursor = get\_conn()

sql = "insert into history values (%s,%s,%s,%s,%s,%s,%s,%s,%s)"

for k,v in dic.items():

cursor.execute(sql,[k, v.get("confirm"),v.get("confirm\_add"),v.get("suspect"),

v.get("suspect\_add"),v.get("heal"),v.get("heal\_add"),

v.get("dead"),v.get("dead\_add")])

conn.commit()

print(f"{time.asctime()}插入历史数据完毕")

except:

traceback.print\_exc()

finally:

close\_conn(conn,cursor)

#更新历史数据

def update\_history():

cursor = None

conn = None

try:

dic = get\_tencent\_data()[0]#0代表历史数据字典

print(f"{time.asctime()}开始更新历史数据")

conn,cursor = get\_conn()

sql = "insert into history values (%s,%s,%s,%s,%s,%s,%s,%s,%s)"

sql\_query = "select confirm from history where ds=%s"

for k,v in dic.items():

if not cursor.execute(sql\_query,k):

cursor.execute(sql,[k, v.get("confirm"),v.get("confirm\_add"),v.get("suspect"),

v.get("suspect\_add"),v.get("heal"),v.get("heal\_add"),

v.get("dead"),v.get("dead\_add")])

conn.commit()

print(f"{time.asctime()}历史数据更新完毕")

except:

traceback.print\_exc()

finally:

close\_conn(conn,cursor)

#爬取百度热搜数据

def get\_baidu\_hot():

option = ChromeOptions()

option.add\_argument("--headless")#隐藏游览器

option.add\_argument("--no--sandbox")

browser = Chrome(options = option,executable\_path="chromedriver-dev.exe")

url = "https://voice.baidu.com/act/virussearch/virussearch?from=osari\_map&tab=0&infomore=1"

browser.get(url)

#print(brower.page\_source)

but = browser.find\_element\_by\_css\_selector('#ptab-0 > div > div.VirusHot\_1-5-3\_32AY4F.VirusHot\_1-5-3\_2RnRvg > section > div')

#点击加载更多

but.click()

time.sleep(1)

#爬虫与反爬，模拟人等待1秒

c = browser.find\_elements\_by\_xpath('//\*[@id="ptab-0"]/div/div[2]/section/a/div/span[2]')

context = [i.text for i in c]

browser.close()

return context

def update\_hotsearch():

cursor = None

conn = None

try:

context = get\_baidu\_hot()

print(f"{time.asctime()}开始更新数据")

conn,cursor = get\_conn()

sql = "insert into hotsearch(dt,content) values(%s,%s)"

ts = time.strftime("%Y-%m-%d %X")

for i in context:

cursor.execute(sql,(ts,i))

conn.commit()

print(f"{time.asctime()}数据更新完毕")

except:

traceback.print\_exc()

finally:

close\_conn(conn,cursor)

if \_\_name\_\_ == "\_\_main\_\_":

l = len(sys.argv)

if l == 1:

s = """

请输入参数

参数说明，

up\_his 更新历史记录表

up\_hot 更新实时热搜

up\_det 更新详细表

"""

print(s)

else:

order = sys.argv[1]

if order == "up\_his":

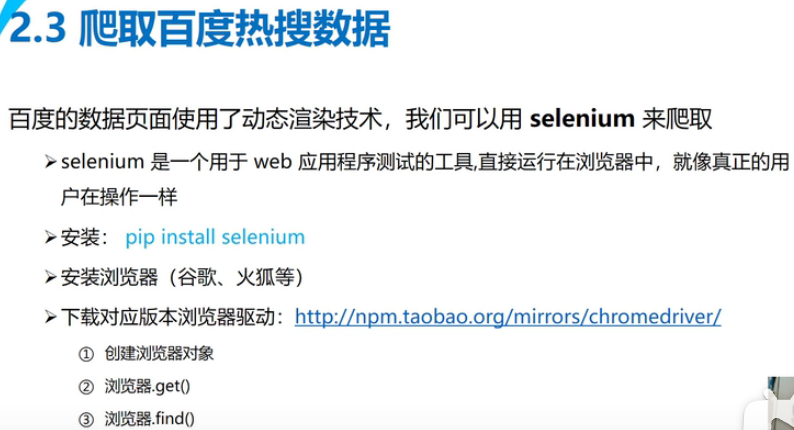
update\_history()

elif order == "up\_det":

update\_details()

elif order == "up\_hot":

update\_hotsearch()



**百度使用的动态渲染技术，我们如果用平常的手段去爬取是爬取不到任何东西的，是空白的，下面演示一下：**

1. **打开百度页面：**[**https://www.baidu.com/**](https://www.baidu.com/)**，点击右侧的热搜。**

**我们发现：出现，简单标签和权重就是热度。**

**1. selenium安装**

**安装selenium**

**在cmd环境下:**

**首先查看python是否安装**

**C:/Users/xxx>python --version //python -V**

**Python 2.7.12**

**通过pip安装selenium**

**$pip install -U selenium //提示安装pip**

**$python -m pip install --upgrade pip**

**$pip --version //查看版本**

**$pip install -U selenium //安装**

**$pip show selenium //是否安装成功**

**安装火狐,并在全局环境变量System Variables的Path中加入Firefox所在的安装路径，例如 ;D:\firfox**

**下载火狐需要的geckodriver 地址: https://github.com/mozilla/geckodriver/releases**

**解压后将geckdriver.exe 放到Python的目录下**

**相应的，Chrome浏览器需要安装chromedriver，下载地址 http://chromedriver.storage.googleapis.com/index.html ，将chromedriver.exe放到Chrome的安装路径下，例如 C:\Program Files (x86)\Google\Chrome\Application ，然后把改路径加入到环境变量Path中。我的电脑->属性->高级系统设置->环境变量--System变量Path增加 ;C:\Program Files (x86)\Google\Chrome\Application**

1. **导入包：from selenium.webdriver import Chrome,ChromeOptions**
2. **实例化对象：**

**browser=Chrome()#1.创建谷歌对象**

1. **打开网址**

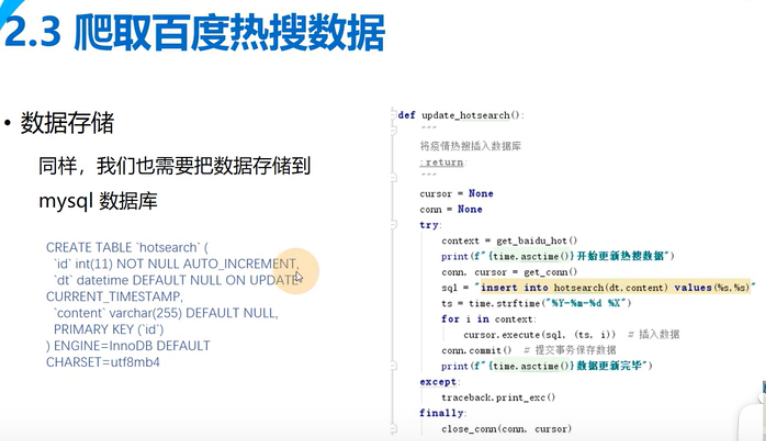
**browser.get(url)#2.打开地址**

1. **xpath:**

**#1.在谷歌浏览器打开网页：https://top.baidu.com/board?tab=realtime**

**#2.选择第一条，然后查看元素，复制-xpath,ctrl+f,粘贴连接，//\*[@id="sanRoot"]/main/div[2]/div/div[2]/div/div[2]/a/div[1]**

**这里的div[1]只是第一条。我们只要div就可以看到前30条了.**

****

**https://view.inews.qq.com/g2/getOnsInfo?name=disease\_h5&callback=jQuery3510017513060143395043\_1634094282090&\_=1634094282091**