|  |  |
| --- | --- |
|  | 惠州学院  HUIZHOU UNIVERSITY |

GIS应用开发课程报告

钟志伟

姓 名\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2104080109

学 号\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

21地理信息科学2班

专业班级\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2023/12/15

提交日期\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

目录

[**课程所需数据及js库** 3](#_Toc153561024)

[**数据** 3](#_Toc153561025)

[底图数据 3](#_Toc153561026)

[Geojson数据 3](#_Toc153561027)

[渲染所用纹理 3](#_Toc153561028)

[**Js库** 3](#_Toc153561029)

[Leaflet 3](#_Toc153561030)

[Axios 3](#_Toc153561031)

[Bootstrap 3](#_Toc153561032)

[D3 3](#_Toc153561033)

[**引入模块** 3](#_Toc153561034)

[底图引入 3](#_Toc153561035)

[Geojson数据引入 3](#_Toc153561036)

[纹理引入 3](#_Toc153561037)

[渲染模块 3](#_Toc153561038)

[颜色及纹理渲染 3](#_Toc153561039)

[**高亮反馈** 3](#_Toc153561040)

[**搜索模块** 4](#_Toc153561041)

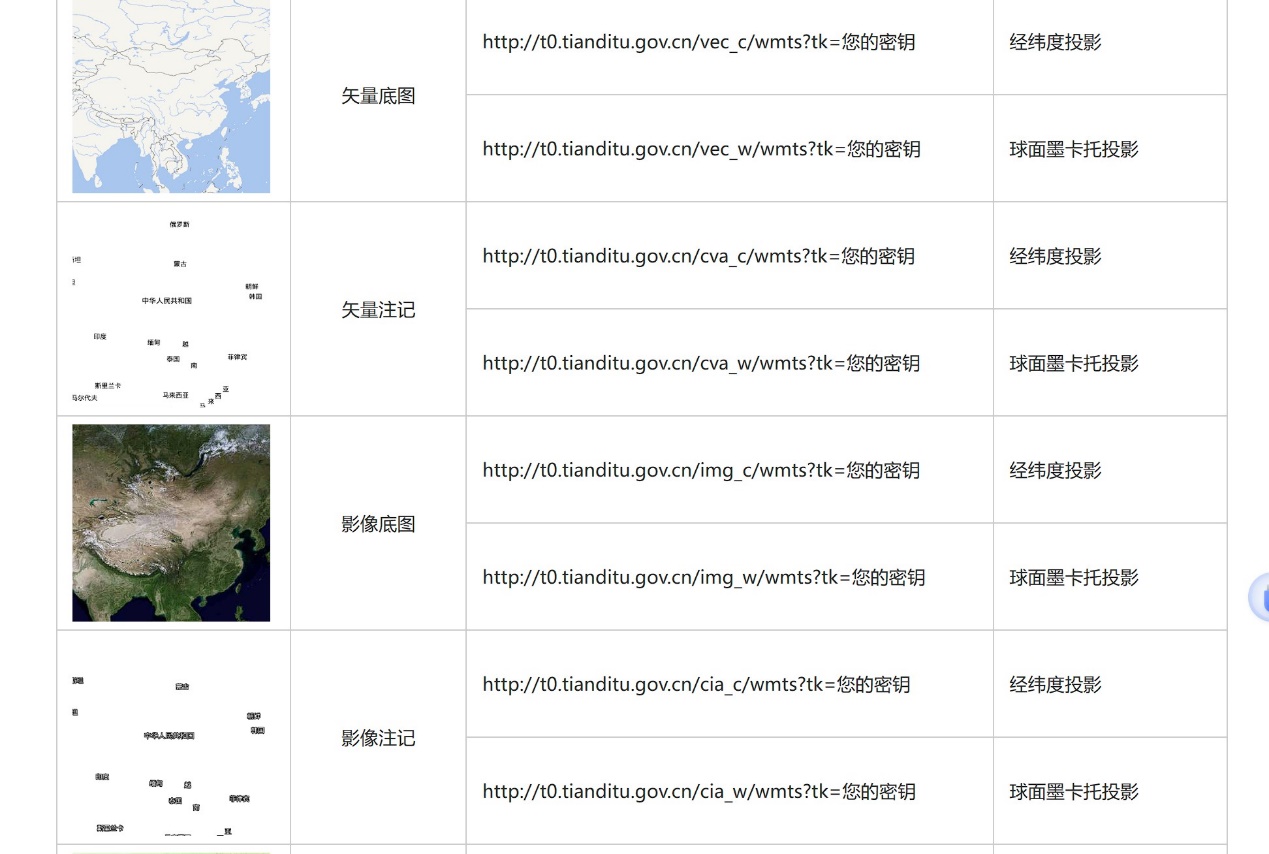
[**信息展示模块** 7](#_Toc153561042)

# **课程所需数据及js库**

## **数据**

### 底图数据

影像底图和矢量底图以及矢量注记和影像注记均使用天地图提供的瓦片数据，坐标系均使用4326（WGS1984）



### Geojson数据

在QGIS中使用天地图的瓦片数据作为底图，绘制出惠州学院的道路、绿地、建筑物、运动场、水域面图层，并添加了一些字段，如建筑物添加了名称、楼层字段，最后导出为Geojson格式得到数据，均在文件夹hzu\_geojson中



### 渲染所用纹理

草地纹理和水系纹理均在网络上获取

## **Js库**

### Leaflet

在项目中主要用于渲染数据

### Axios

在项目中主要用于模拟从后端获取数据

### Bootstrap

在项目中主要用于参考一些CSS样式

### D3

在项目中主要用于将纹理渲染至图层

# **引入模块**

## 底图引入

用leaflet中的L.tileLayer方法获取，以影像图层为例

 const YX=L.tileLayer('http://t{s}.tianditu.gov.cn/img\_w/wmts?SERVICE=WMTS&REQUEST=GetTile&VERSION=1.0.0&LAYER=img&STYLE=default&TILEMATRIXSET=w&FORMAT=tiles&TILEMATRIX={z}&TILEROW={y}&TILECOL={x}&tk=d4ded130df268fab61dfde0ae28cc758',

                        { subdomains:'01234567',maxNativeZoom:30})

引入后使用leaflet中的图层控件引入，默认加载影像图层

  const baseLayers = {

                            "影像底图":YX,

                             "矢量底图": SL

                                            };

                        const overlays = {

                             "影像标注": YXZJ,

                             "矢量标注": SLZJ,

                                            };

 const layercontrol= L.control.layers(baseLayers, overlays).addTo(map);

        YX.addTo(map)

## Geojson数据引入

使用axios库模拟后端获取数据，再使用leaflet引入数据，由于本项目主要展示惠州学院全域，就不将引入的图层加入到图层控件中了，不过惠州学院全域的边界加入到了图层控件中，以便于用户查看边界，以下代码以引入道路面图层为例

   axios({

         url:'./hzu\_geojson/道路面.geojson',

         method:'get',

        }).then(result=>{

         const road\_p=L.geoJSON(result.data,{

          style: function (feature) {

             return {color: feature.properties.color="white",

                      fillColor:"#ada6a6",

                      fillOpacity:1};

          }

         }).addTo(map)

         // layercontrol.addOverlay(road\_p,"道路")

        })

## 纹理引入

由于本项目都是使用svg方式渲染,且leaflet渲染在标签元素中只有一个svg标签,使用D3库找到svg标签添加相应的标签和纹理,以草地纹理为例

 const svg=d3.select('svg')

    const defs=svg.append('svg:defs')

    const pattern=defs.append('svg:pattern')

    pattern.attr('id','green').attr('width',1024).attr('height',1024).attr('patternUnits','userSpaceOnUse');

    const image\_green=pattern.append('svg:image');

    image\_green.attr('href','./草地纹理.jpg').attr('width',1024).attr('height',1024).attr('x',0).attr('y',0)

# 渲染模块

## 颜色及纹理渲染

绿地面和水域面图层均使用D3将纹理渲染至图层。道路面图层边框使用白色系，填充使用灰色系渲染,使其有真实道路的效果，建筑物面图层使用黄色系，边框使用灰色系渲染，并给上一些透明度。运动场面图层使用蓝色系渲染，使其有运动的的心理效果，并给上一些透明度。



## **高亮反馈**

触碰图层进行高亮反馈可以让用户获得一种直观的、及时的交互体验。当用户触碰某个图层时，该图层会高亮显示，这种视觉上的反馈使用户能够明确知道自己的操作效果，从而增强了他们的操作信心和满足感。此外，交互动效一直是影响用户体验的关键因素，它可以帮助用户更好地理解和掌握产品的使用方法。通过提供这种即时的高亮反馈，产品不仅显得更加“高大上”，而且还能增强用户的沉浸感和参与度，从而提高整体的用户体验。

实现方法：为每个图层设置一个鼠标进入的事件，用于监听鼠标是否进入图层，当鼠标进入图层时，获取回调函数参数e中的当前图层，绘制出当前进入的图层并进行与原图层有相关性且更强的渲染效果，并且绑定上弹窗等等效果，而当鼠标离开该图层时，移除该图层，即可实现高亮反馈功能，以建筑物图层的高亮反馈为例

//鼠标进入建筑物时触发

        building.on('mouseover',function(e){

        // console.log(e.layer.feature)

         const change\_layer=L.geoJSON(e.layer.feature,{

            style:function(feature){

               //console.log(feature)

               return{

               fillColor:'#eec694',

               fillOpacity:1,

               stroke:'false',

               color:'#898080',

               weight:7

               }

            }

         }).bindPopup(function (layer) {

            return `<div><p>名称:${layer.feature.properties.name}</p><p>楼层：${layer.feature.properties.floot}</p></div>`;

        }).addTo(map)

          //鼠标退出建筑物时触发

          change\_layer.on('mouseout',function(e){

         change\_layer.removeFrom(map)

        })

这种实现方法占用的总内存较小，每个时间段最多绘制一个图层，且简单易懂，

小bug是当地图缩放较小时，可能会难以判定鼠标离开，导致图层一直以高亮显示。

# **搜索模块**

搜索模块可以让用户获得方便快捷的搜索体验。用户可以通过输入关键字或者地址来快速定位到所需的地理信息，这种搜索功能可以帮助用户节省寻找信息的时间

实现方法：添加一个文本框和按钮还有联想框，进行一些CSS样式设置，并且给文本框添加一个监听函数，当输入文本时调用search()函数；给搜索按钮添加一个监听函数，当用户点击按钮时调用search\_result()函数，以下为html标签和CSS样式

<div class="search">

      <input oninput="search()" type="text" id="searchInput" placeholder="请输入关键字">

      <button onclick="search\_result()">搜索</button>

      <div id="result"></div>

    </div>

.search{

        position: absolute;

        top: 10px;

        left:50px;

        z-index: 1000;

        border: 2px solid #ccc;

    }

    #searchInput{

        width: 200px;

        transition: width 0.3s ease;

    }

    #searchInput:focus {

        width: 250px;

      }

      #searchInput:active {

        width: 400px;

      }

    #result{

        height: auto;

        background-color: white;

    }

    p {

        cursor: pointer;

    }

    .highlight {

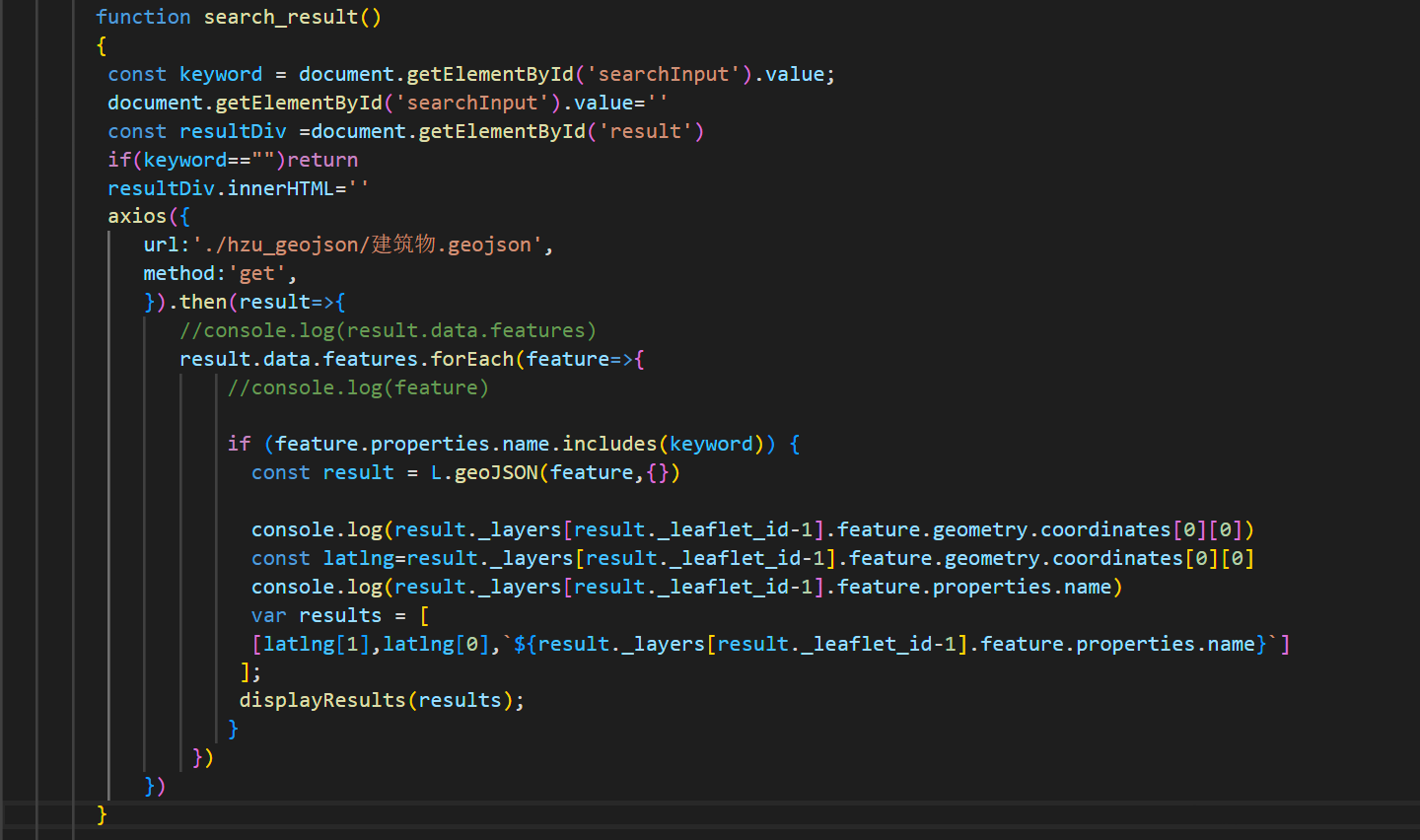
        background-color: yellow;

    }

search()函数实现：当用户输入字符串时，先检测联想框中是否有HTML标签内容，如果有，将其删除，然后将字符串中的获取过来，与建筑物中的name字段一一比对，如果字符串中有name中的字符串，即将其添加至联想框，以下是search函数的实现



search\_result()函数实现：将文本框中的字符串获取过来，清空文本框中的字符串以提升用户体验，如果值为空直接返回函数，然后将字符串与建筑物图层中的name字段一一比对，找到了相关的图层即将其坐标和想要展示的字段放入一个数组中，执行displayResult(result)函数



displayResult(result)函数实现：这个函数是为了为搜索目标添加一个点标记并跳转至搜索位置，在添加点标记前，先判断一下是否有之前添加的点标记，如果有就移除，再进行点标记和视图跳转即可



关于联想框的高亮显示，添加一个监听函数即可实现



# **信息展示模块**

信息展示模块可以让用户直观的获取当前想要获取的图层的信息，这种可视化的操作方式可以帮助用户更好地理解和掌握地理信息，提高信息的吸收效率，目前信息展示模块只有经纬度的展示

实现方法：添加一个信息展示模块并作相关样式调整

//html标签

<div class="latlng\_view"><p></p></div>

//CSS样式

 .latlng\_view{

        position: absolute;

        top: 20px;

        right:10px;

        transform: translate(-50%, -50%);

        width: 200px;

        height: 30px;

        z-index: 1000;

        border: 1px solid #0a0909;

        background-color: rgb(254, 254, 254);

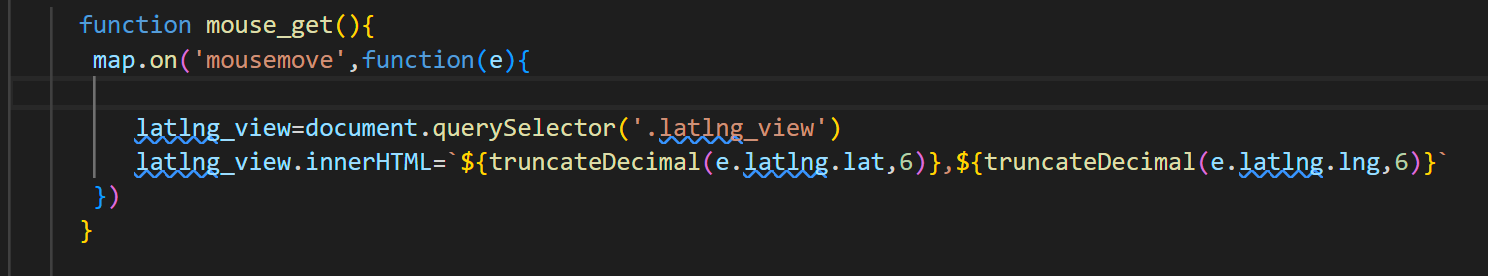
        border-radius: 5px;

        padding: 0 10px;

        font-size: 14px;

    }

调用mouse\_get()函数：给map添加一个鼠标移动事件，将信息展示模块的标签获取过来，将经纬度经过truncateDecimal()函数处理展示到信息展示模块即可



truncateDecimal(num,decimaPlaces)函数实现:

num:要处理的数字

decimaPlaces:需要保留多少为小数

使用Math进行简单处理

