

信息可视化课程大作业系统文档

设计需求

2019年末发生在中国武汉的新型冠状病毒疫情肆虐，疫情发生以来，疫情数据在疫情溯源和监测、分析疫情情况、支撑疫情态势研判和疫情防控部署等方面起到了积极作用。当前，全民抗击新冠肺炎疫情进入新阶段，通过疫情数据可视化，能够有效的展示疫情数据，直观的展示疫情形态，让民众更好的把握疫情走向，同时提升民众对疫情的重视，促进各方更好的加入这场战役中。

基本需求

为疫情数据可视化，将给定的国内各省市的感染情况的日志文本，将其不够直观的文字形式的数据，转化为直观、具体且用户友好的3D地图形式，来直观显示疫情的大致分布情况，并且可以查看具体省份的疫情统计情况。

扩充需求

新冠肺炎疫情发生以来，有关疫情传播态势、虚假防治措施等流言和谣言在网上传播，引发不少网民误解误读，个别网民编造传播虚假病例数据等谣言，极易引发社会恐慌情绪，严重影响网络安全秩序和社会稳定，同时也给疫情防控工作带来严重干扰。因此设置一个新闻和相关信息科普网页，帮助民众在防疫时期，更好的识别谣言。

设计介绍

开发框架

- vue.js echart axios
- 三维地图基于 [earth sdk](#), 一个基于Cesium的开源三维地球开发包
- 静态网页bootstrap

功能介绍

功能模块	功能细则
全国数据总计模块	展示累计确诊、治愈、死亡的人数
全国数据可视化地图模块	<ol style="list-style-type: none">1. 展示一个依照省份划分边界的地图2. 在每个省份上表示出省份的名称
省份疫情的具体情况模块	<ol style="list-style-type: none">3. 依照给定疫情数据统计每个省份患者数量，并依照每个省份患者给定标准的多少划分地区疫情的严重程度，以颜色深浅标识出来4. 左下角表示出各个区域颜色所代表的确诊患者区间5. 点击某个省份的板块，显示省份项的数据标签，并显示对应省份的市区
疫情信息普及模块	分为主页的成果展示，预防方式、提供的服务和治疗措施

条形图

- 展示当前地理范围的分区域详情图表，并且有悬浮窗可以显示该地区的确诊、死亡和康复人数。可以在国家和省份之间切换。
- 为条形图的确诊、死亡和康复条设置点击事件，可以点击收起和放出。

```
1 | updateChart () {
2 |     this.$root._dataserver.loadSubArea(this.$root.currentArea).then(data => {
```

```

3     var ydata = [];
4     var confirmed = [];
5     var cured = [];
6     var dead = [];
7     var empty = [];
8     if (!data.subs)
9         return;
10    data.subs.sort((a, b) => {
11        return a.confirmedCount - b.confirmedCount;
12    });
13    var maxValue = 0;
14    data.subs.forEach(element => {
15        maxValue =
16            maxValue > element.confirmedCount
17            ? maxValue
18            : element.confirmedCount +
19                element.curedCount +
20                element.deadCount;
21    });
22    data.subs.forEach(element => {
23        ydata.push(element.name);
24        confirmed.push(element.confirmedCount);
25        cured.push(element.curedCount);
26        dead.push(element.deadCount);
27        empty.push(
28            maxValue -
29            element.confirmedCount -
30            element.curedCount -
31            element.deadCount
32        );
33    });
34
35    var option = {
36        tooltip: {
37            trigger: "axis",
38            axisPointer: {
39                // 坐标轴指示器，坐标轴触发有效
40                type: "line" // 默认为直线，可选为: 'line' | 'shadow'
41            },
42            formatter: function (param) {
43                var tip = param[0].name + "</br>";
44                for (var i = 0; i < param.length; i++) {
45                    if (param[i].seriesName.length > 0) {
46                        tip += param[i].marker + param[i].seriesName + ":" + +
param[i].data + "</br>";
47                    }
48                }
49                if (param.length > 3) {
50                    tip += " " + " ";
51                }
52                return tip;
53            }
54        },
55        legend: {
56            data: ["Confirmed", "Death", "Recovered"],
57            textStyle: {
58                color: "white"
59            }
60        },
61        grid: {
62            left: "2%",
63            right: "6%",
64            bottom: "3%",
65            containLabel: true
66        },
67        xAxis: {
68            type: "value",
69            axisLabel: {
70                textStyle: {
71                    color: "white" //坐标值得具体的颜色
72                }
73            },
74            axisLine: {
75                lineStyle: {
76                    type: "solid",
77                    color: "white" //坐标线的颜色
78                }
79            },
80            splitLine: {
81                show: true,
82                lineStyle: {
83                    type: "dashed"
84                }
85            }
86        },
87        yAxis: {
88            type: "category",

```

```

89         data: ydata,
90         axisLine: {
91             lineStyle: {
92                 type: "solid",
93                 color: "white" //坐标线的颜色
94             }
95         },
96         axisLabel: {
97             textStyle: {
98                 color: "white" //坐标值得具体的颜色
99             }
100     },
101     series: [
102     {
103         name: "Confirmed",
104         type: "bar",
105         stack: "总量",
106         data: confirmed,
107         itemStyle: {
108             normal: {
109                 color: "#f56262"
110             }
111         }
112     },
113     {
114         name: "Death",
115         type: "bar",
116         stack: "总量",
117         data: dead,
118         itemStyle: {
119             normal: {
120                 color: "#6c757d"
121             }
122         }
123     },
124     {
125         name: "Recovered",
126         type: "bar",
127         stack: "总量",
128         data: cured,
129         itemStyle: {
130             normal: {
131                 color: "green"
132             }
133         }
134     },
135     {
136         name: "",
137         type: "bar",
138         stack: "总量",
139         data: empty,
140         itemStyle: {
141             normal: {
142                 color: "rgba(88, 255, 255, 0)"
143             }
144         }
145     }
146 ],
147 );
148
149
150
151
152
153
154     this._eChart.setOption(option, true);
155 });
156 }

```

三维地图着色图

统计各省和市区的确诊人数进行相应区域的颜色映射，并且有相应的图例。

```

1 setColor() {
2     if (this._dataSource) {
3         let self = this;
4         this.$root._dataserver
5             .loadSubArea(this.areaName, this.$root.currentTime)
6             .then(data => {
7                 var colormap = {};
8                 self._areaKV = {};
9                 var length = data.subs.length;
10                for (var i = 0; i < length; i++) {
11                    var province = data.subs[i];

```

```

12     colorMap[province.name] = self.getAreaColor(
13         province.confirmedCount
14     );
15     self._areaKV[province.name] = province;
16 }
17 var entities = self._dataSource.entities.values;
18 for (var i = 0; i < entities.length; i++) {
19     var entity = entities[i];
20     entity.polygon.material = colorMap[
21         entity.properties["Name"]._value
22     ]
23     ? colorMap[entity.properties["Name"]._value]
24     : self.colorLegend[5];
25     entity.polygon.outline = false;
26     entity.polygon.outlineWidth = 3;
27 }
28 });
29 }
30 },

```

全国城市热力图

- 准备热力图数据格式：由于热力图使用场景一般为地图，所以，数据源需要提供经纬度作为位置信息。
- 在地图上填充数据：基于canvas绘制热力图，设置热力图每个数据点的半径大小，根据离散点缓冲区的叠加来确定热力分布密度。

```

1 function createH337Data({
2     rectangle,
3     valueRange,
4     radius,
5     width,
6     height,
7     cityCoords,
8     createCityCoordsIteratorFunc,
9     cityInfos,
10    createCityInfosIteratorFunc,
11    // updateDate,
12 }) {
13     var h337DataV = [];
14
15     var rw = (rectangle[2] - rectangle[0]);
16     var rh = (rectangle[3] - rectangle[1]);
17     var vd = (valueRange[1] - valueRange[0]);
18     const r = 0.5 * radius / rw * width;
19
20     const cii = createCityInfosIteratorFunc(cityInfos);
21     for (let ci of cii) {
22         const cityName = ci.name;
23         const cc = findCityCoords(cityName, createCityCoordsIteratorFunc(cityCoords));
24         const v = (ci.confirmedCount - valueRange[0]) / vd;
25
26         // console.log('rv: ' + v);
27         h337DataV.push({
28             x: (cc[1] - rectangle[0]) / rw * width | 0,
29             y: height - (cc[2] - rectangle[1]) / rh * height | 0,
30             value: v === 0 ? 0.001 : v,
31             radius: r < 1.0 ? 1.0 : r,
32             cityName: cityName,
33         });
34     }
35     return h337DataV;
36 }

```

- 颜色映射：根据画布上每个像素点累计得到的灰度值，可以从彩色映射色带中得到对应位置的颜色。

```

1 function createHeatMap(earth) {
2     var cfg = {
3         "position": [
4             (70.2938 + 150.1160) * 0.5 * Math.PI / 180.0,
5             (11.7733 + 49.9371) * 0.5 * Math.PI / 180.0,
6             0
7         ],
8         // 在地球上的实际尺寸，单位是米
9         "width": 1352299.693558889,
10        "height": 7628130.204938413,
11        // 调色板参数，第一个代码值域，范围在0-1之间，第二个为css颜色字符串
12        "gradient": [
13            [

```

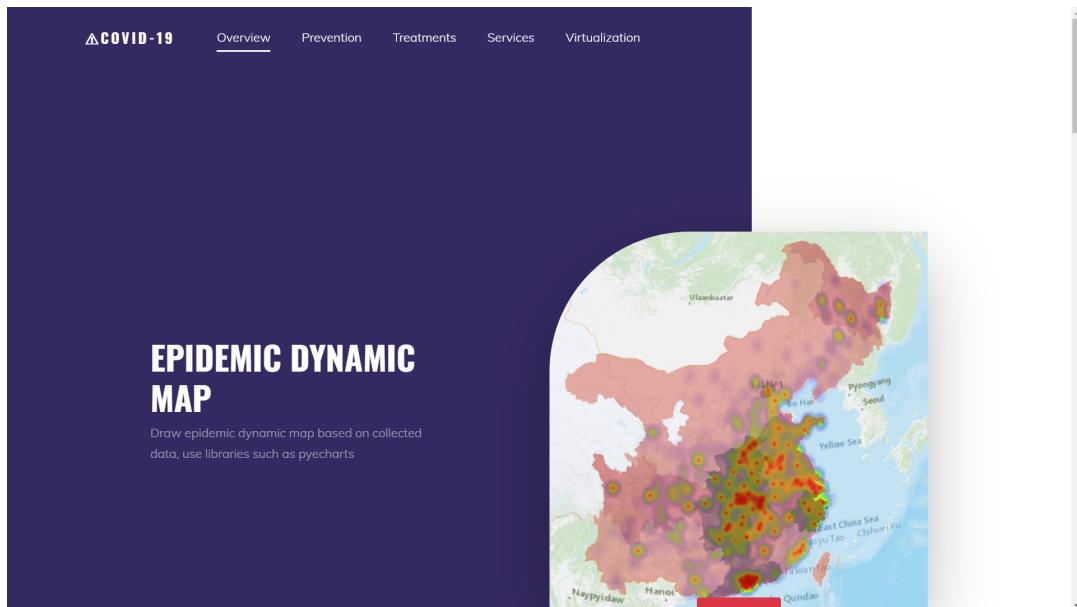
```

14     0,
15     "rgba(0, 255, 0, 0.0)"
16   ],
17   [
18     0.05,
19     "rgba(0, 255, 0, 0.3)"
20   ],
21   [
22     0.5,
23     "rgba(255, 255, 0, 0.5)"
24   ],
25   [
26     1,
27     "red"
28   ]
29 ],
30   "maxValue": 1,
31   "dataWidth": 512,
32   "dataHeight": 256,
33 }
34
35 var h = new XE.Obj.HeatMap(earth);
36 h.xbsjFromJSON(cfg);
37
38 var p = Cesium.createXbsjGroundPrimitiveFromRectangle(70.2938, 11.7733, 150.1160, 49.9371)
39 earth.czm.scene.groundPrimitives.add(p)
40 earth.czm.scene.groundPrimitives.remove(h._customGroundRectangle._groundPrimitive);
41 h._customGroundRectangle._groundPrimitive = p;
42 h._customGroundRectangle._groundPrimitive.appearance.material.uniforms.image =
43 h._customGroundRectangle._texture;
44 return h;
}

```

案例展示

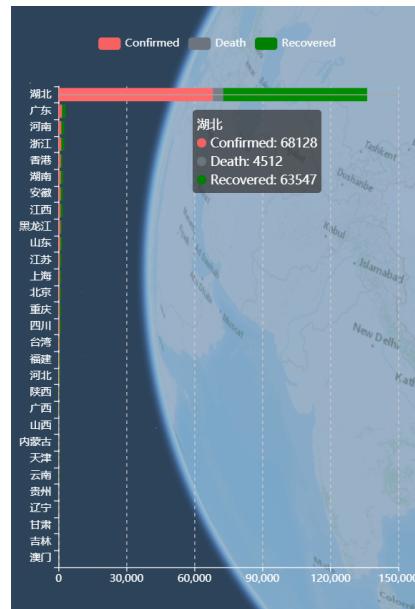
- 主页是疫情的相关信息展示，包括成果展示、预防方式、提供的服务和治疗措施以及可视化界面等。



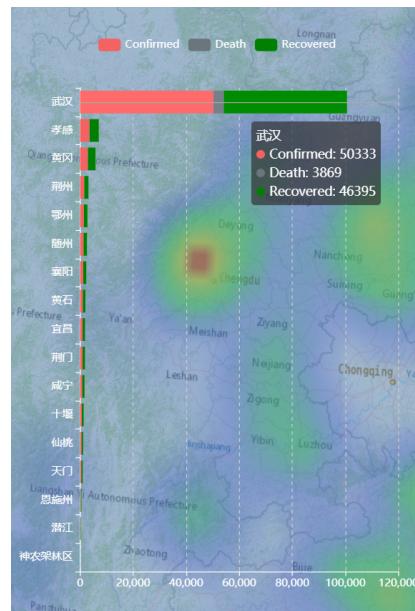
- 可视化界面顶栏左侧Back按钮可以返回主页，中间是我们的网页名字和当前地图显示区域，右侧是数据总览，数据来源于丁香园，可以展现到2020年3月19日22日为止的疫情数据。



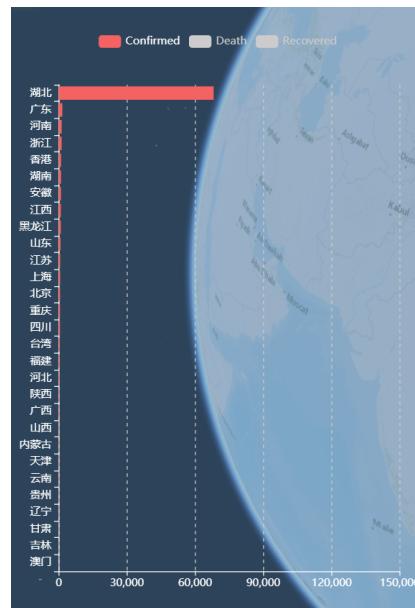
- 左侧是当前地理范围的分区域详情图表，并且有悬浮窗可以显示该省的确诊、死亡和康复人数。



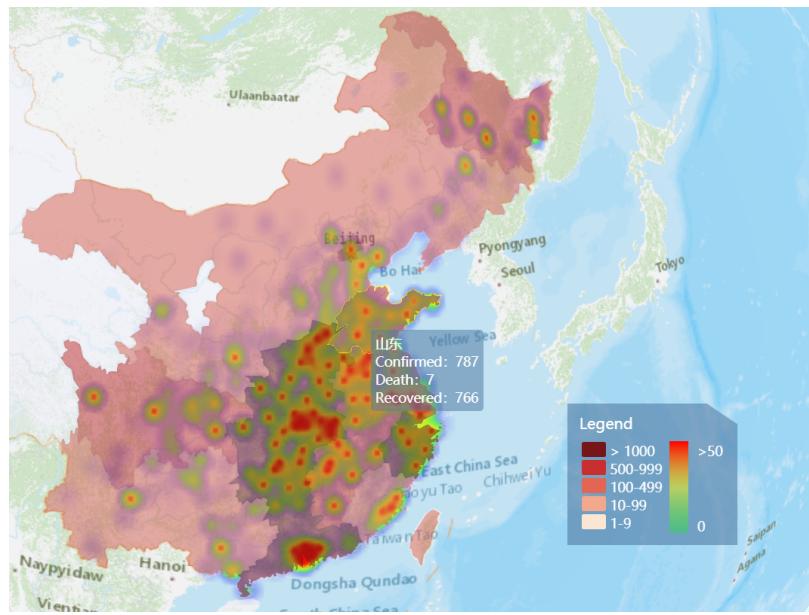
- 点击相应省份的条形图，可以显示该省份对应市区的确诊、死亡和康复人数，并且中央的地图旋转和缩放到该省。



- 条形图的确诊、死亡和康复条可以点击收起和放出，能够相对直观的展现这三项数据之间的差异并进行比较。



- 中间为三维地图可以放大缩小和旋转，省区和市区的面着色颜色深浅可以反映该区域的确诊人数，并且有相应的图例。



- 此外还有全国城市热力图和相应的图例。

