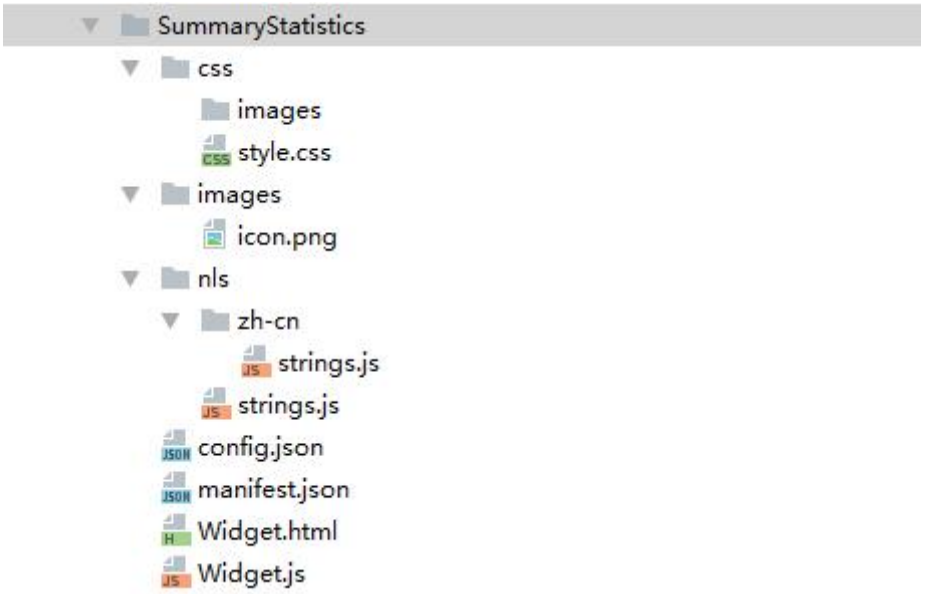
部分总结

前言:以下内容或有不正确之处实则不可避免并概不负责亦望包涵并更正

一．模块化

1.1 模块文件结构



些许说明：

- Widget.js 模块 js 代码
- Widget.html 模块 html 代码
- css/style.css 模块 css 代码
- images/icon.png 是模块默认的图标路径【可配置】
- config.json 模块配置文件，比如配置表明，字段等
- manifest.json 模块相关配置如：hasConfig【是否有 config.json 配置】,inPanel【是否在 Widget 窗口中】等
- nls/zh-cn/strings.js 文件主要配置 Widget.html 中一些固定文本信息

注意：这些文件之所以能做到上面对应效果均基于 hugegis/BaseWidget.js 文件【区分'digit/_WidgetBase',dojo 提供的一个类】并且 Widget.js 文件会继承该文件的一些方法。Widget.js 文件中几乎都引用 hugegis/BaseWidget.js 文件，当然只是几乎，具体如何引用文件参考项目代码

1.2 一些方法

- 头文件
  - dojo/topic
- 广播与订阅

```
1. topic.publish("eventName", params);//广播【params 传递的参数】
2. topic.subscribe("eventName",hamdlerFunction);//订阅【调用的函数】
```

```
1. var url = require.toUrl('hugegis');//js 文件中多数情况下需要用这样的方式获取本地图片地址
```

访问 dojo toolkit 官网　访问 arcigs api 3.2 官网

2.地图交互

2.0.在对地图操作之前

- 头文件
  - esri/layers/GraphicsLayer
  - esri/graphic

```
var layer = new GraphicLayer();
map.addLayer(layer);//Widget 会自动继承 map
```

注意：以下内容默认已经有以上头文件并实例化 layer

2.1 画点

- 头文件
  - esri/geometry/Point
  - esri/symbols/PictureMarkerSymbol

- 代码

```
var pt = new Point(x,y,map.spatialReference);//x,y 坐标及参考系
var pms = new PictureMarkerSymbol(url,36,36);//url：point 的 symbol
var gra = new Graphic(pt,pms);//Graphic 也可能是 graphic
layer.add(gra);
```

2.2 画线

- 头文件
  - esri/geometry/Polyline
  - esri/symbols/SimpleLineSymbol
  - esri/symbols/SimpleFillSymbol

- 代码

```
//坐标数组
var points =[];
points.push([x,y]);//注意数据组成

var polyline = new Polyline(map.spatialReference);
polyline.paths = [points];//数据组成需要重视
var sls = new SimpleLineSymbol(SimpleLineSymbol.STYLE_SOLID, c, 5);
var gra = new Graphic(this.polyline, sls);
layer.add(gra);
```

2.3 画面

- 头文件
  - esri/geometry/Polygon
  - esri/symbols/SimpleFillSymbol
  - esri/symbols/SimpleLineSymbol

- 代码

```
var polygon = new Polygon(map.spatialReference);
```

```
2.  polygon.addRings([[-180,-90],[-180,90],[180,90],[180,-90],[-180,-90]]);//数据组成需要重视
3.  var symbol = new SimpleFillSymbol("solid", new SimpleLineSymbol("solid", new dojo.Color([232, 104, 80]), 2), new dojo.Color([232, 104, 80, 0.25]));
4.  var gra = new Graphic(polygon,symbol);
5.  layer.add(gra);
```

2.4 定位

- 点定位
  - esri/geometry/Point

- 代码

```
1.  var pt = new Point(x,y,map.spatialReference);//这里一定要是当前地图的空间参考系
2.  map.centerAt(pt);//或者 map.centerAndZoom(pt,6);
```

- 面定位

```
1.  var extent = polygon.getExtent();//默认已经存在有一个面或者线对象
2.  map.setExtent(extent);//map.setExtent(extent.expand(1.5));
```

2.5 buffer

- 头文件
  - esri/tasks/GeometryService
  - esri/tasks/BufferParameters
  - esri/symbols/SimpleFillSymbol
  - esri/symbols/SimpleLineSymbol
  - esri/geometry/Polygon
  - esri/tasks/AreasAndLengthsParameters

- 代码（第一种：只是处理面的拐角出，变得圆滑）

```
1.  var gs = new GeometryService(gsUrl);//appConfig.geometryService 地图服务地址
2.  var map = this.map;
3.  function doConvexHull(points) {//points 点数组
4.              var symbol = new SimpleFillSymbol("solid", new SimpleLineSymbol("solid", new dojo.Color([232, 104, 80]), 2), new dojo.Color([232, 104, 80, 0.25]));
5.              //convexHull
6.              gs.convexHull(points).then(function (result) {
7.                  var gra = new Graphic(result);
8.                  //buffer
9.                  var params = new BufferParameters();
10.                 params.distances = [parseInt(10)];//bufferDistance 距离
11.               params.bufferSpatialReference = map.spatialReference;
12.                 params.geometries = [gra.geometry];
13.                 gs.buffer(params).then(function (buf) {
14.                     var obj = buf;
15.                     //画面
```

```
                    var arrPoint = obj[0].rings[0].slice(0);
                    var newPolygon = new Polygon(arrPoint);
                newPolygon.setSpatialReference(map.spatialReference);
                    var polygonExtent = newPolygon.getExtent();
                    var graTemp = new Graphic(obj[0]);
                    graTemp.symbol = symbol;
                    layer.add(graTemp);
                    map.setExtent(polygonExtent);
                    //計算面積
            var areasAndLengthsParameters = new AreasAndLengthsParameters();
            areasAndLengthsParameters.polygons = [graTemp.geometry];
            gs.areasAndLengths(areasAndLengthsParameters, function (evt) {
                        var obj = evt;
                    });
                });
            });
        },
```

- 头文件
  - esri/geometry/normalizeUtils
- 代码（第二种：在面的边缘等距离的做 buffer，与第一种有极大区别）

```
var gs = new GeometryService(gsUrl);//appConfig.geometryService 地图服务地址
var map = this.map;
function doConvexHull(geom) {//geom 是一个 polygon 或者是 polyline
                var symbol = new SimpleFillSymbol("solid", new SimpleLineSymbo
l("solid", new dojo.Color([232, 104, 80]), 2), new dojo.Color([232, 104, 80,
0.25]));
    //gs.convexHull 与 normalizeUtils 两种 buffer 的差异
    normalizeUtils.normalizeCentralMeridian([geom]).then(function(result){
                var normalizedGeometry = result[0];
                //buffer
                var params = new BufferParameters();
                params.distances = [parseInt(10)];//距离
                params.bufferSpatialReference = map.spatialReference;
                params.geometries = [normalizedGeometry];
                gs.buffer(params).then(function (buf) {
                    var obj = buf;
                    var arrPoint = obj[0].rings[0].slice(0);
                    //转换成 esri 对象，求取 Extent
                    var newPolygon = new Polygon(arrPoint);
                     newPolygon.setSpatialReference(map.spatialReference);

                    var polygonExtent = newPolygon.getExtent();
                    //绘制
```

```
                    var gra = new Graphic(obj[0]);
                    gra.symbol = symbol;
                    layer.add(gra);
                    map.setExtent(polygonExtent);
                });
            });
        },
```

2.6 画图工具

- 头文件
  - esri/toolbars/draw
  - esri/symbols/PictureMarkerSymbol

- 代码

```
var map = this.map;
var symbol  = new PictureMarkerSymbol(url, 36, 36);
var drawToolBar = new Draw(map);
drawToolBar.setMarkerSymbol(symbol);
on(drawToolBar, 'draw-end', onDrawEnd);//onDrawEnd 画图完毕回调函数
drawToolBar.activate(Draw.POINT);//激活画点，其他如画面，画线可参考 arcgis api, 方式几乎一样

function onDrawEnd(e){
  var geo = e.geometry;
  var gra = new Graphic(geo,symbol);
  layer.add(gra);
  drawToolBar.deactivate();
}
```