GIS开发岗机试题

1. 环境说明

|  |  |
| --- | --- |
| 环境名称 | 版本 |
| Jdk | 1.8.0\_291 |
| Apache Maven | 3.6.3 |
| GDAL | 3.5.0 |
| Geotools | 26-SNAPSHOT |

1. 第三方库安装(GDAL)
2. 安装GDAL

安装文件为：gdal-305-1928-x64-core.msi，双击运行，一直下一步安装成功。安装器默认路径是：C:\Program Files\GDAL，且无法自定义修改。

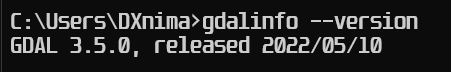
1. 设置环境变量

|  |  |
| --- | --- |
| path | C:\Program Files\GDAL\ |
| GDAL\_DATA | C:\Program Files\GDAL\gdal-data |
| GDAL\_DRIVER\_PATH | C:\Program Files\GDAL\gdalplugins |
| PROJ\_LIB | C:\Program Files\GDAL\projlib |

重启电脑生效！

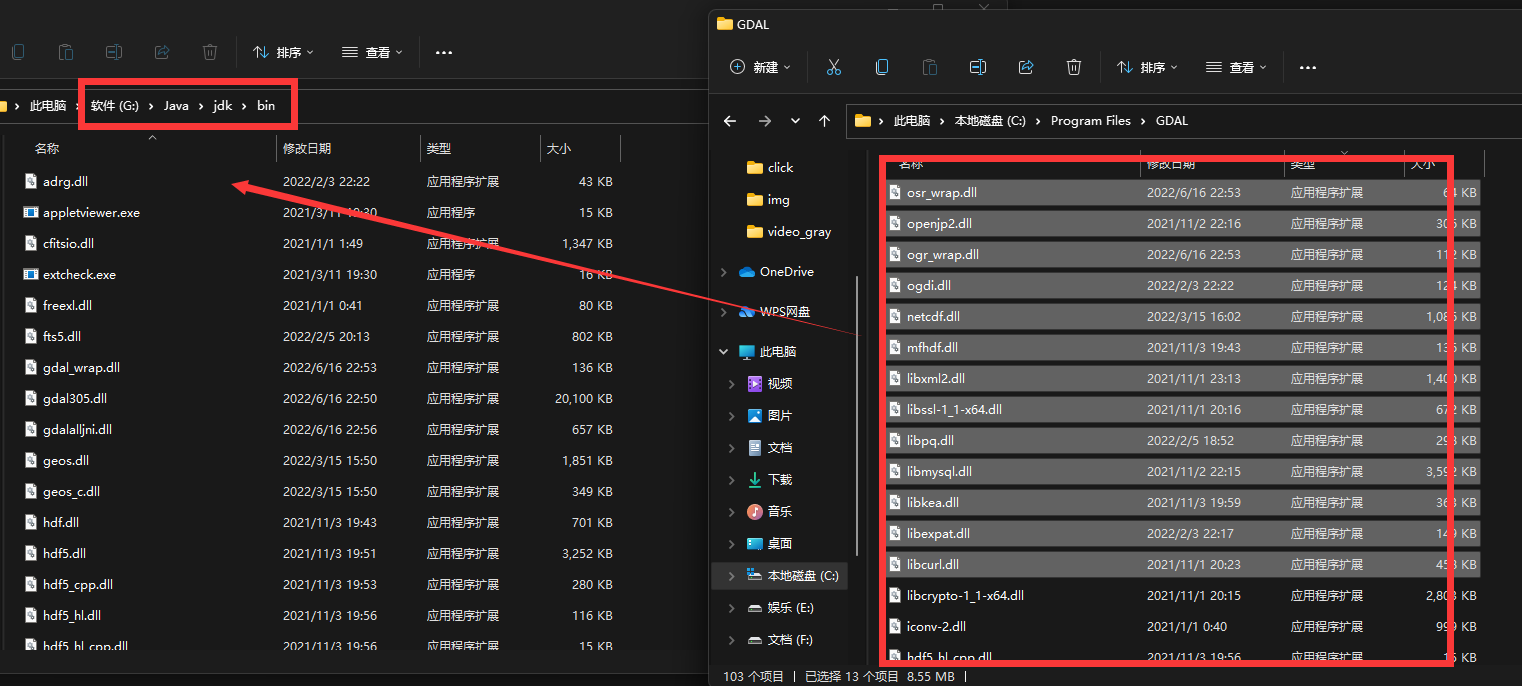
1. 检测GDAL是否安装成功

在cmd终端输入命令:gdalinfo –version

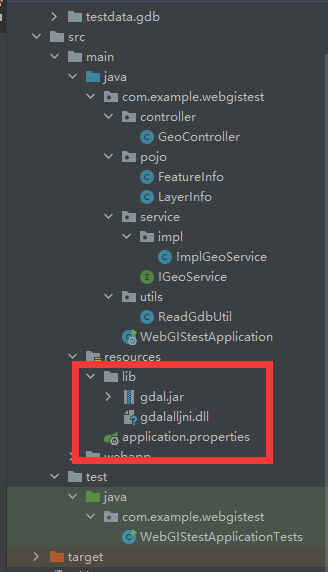


1. Java集成GDAL

把C:\Program Files\GDAL\下所有的.dll文件全部复制到jdk安装路径/bin/下；如图。



把C:\Program Files\GDAL\java\下的gdal.jar和C:\Program Files\GDAL\下的gdalalljni.dll复制到src/main/resources/lib/下；如图



1. Maven项目中的pom.xml引入gdal.jar包

|  |
| --- |
| <dependency>  <groupId>org.gdal</groupId>  <artifactId>gdal</artifactId>  <version>3.5.0</version>  <scope>system</scope>  <systemPath>${project.basedir}/src/main/resources/lib/gdal.jar</systemPath> </dependency> |

1. GDAL相关题目

GIS开发岗机试题

用Java、Geotools（不限）等工具编写一个Web页面，根据提供的样例数据实现分行政区划统计崩岸治理工程，统计结果采用列表显示，界面如图所示。

分行政区划统计崩岸治理工程

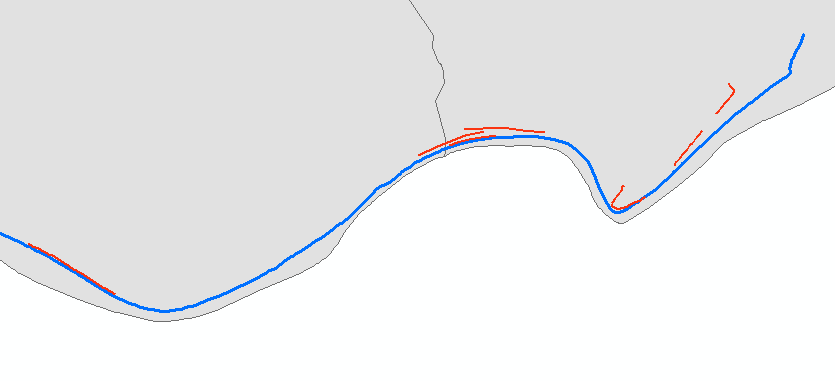
|  |  |  |
| --- | --- | --- |
| 行政区划名称 | 治理河岸长度（km） | 治理工程量（km） |
|  |  |  |
|  |  |  |

tips：

1、治理工程量：历次治理工程长度求和。

2、治理河岸长度：历次治理工程映射到河岸上的长度。

3、样例数据：蓝色为河岸，红色为历次崩岸治理工程，灰色为湖北和安徽区划边界。



1. 设计思路
2. GDAL读取gdb数据

涉及知识：gdb格式数据读取、GeoJSON格式数据结构。

1. 治理工程量

涉及知识：几何关系检测、叠加分析。

a.获取行政区划图层layer；

b.获取治理工程图层每一个线要素zlgcFeature；

c.线要素zlgcFeature与行政区划layer进行相交分析；

d.如果与行政区域相交，再判断是否与行政区域边缘触摸；

e.如果不接触，直接计算该线要素的长度；

f.如果接触，使用线面叠加分析得到在行政区域内的线要素，最后计算该线要素长度；

g.e和f计算的长度累加求和得到最终该图层治理工程量；

h.循环上面步骤计算其他行政区域；

1. 治理河岸长度

涉及知识：领域分析、点到线最短距离、几何关系检测。

说明：此次判断线是否在行政区划内并裁剪步骤和2中的a、b、c、d、e、f思路类似，主要说明治理河岸长度计算。

a.获取河岸图层线要素haFeatures；

b.获取治理工程图层每一个线要素zlgcFeatures；

c.提取线要素zlgcFeatures起点start，中点center，终点end；

d.计算中点center到河岸线要素haFeatures上最近的点haPoint；

e.根据起点start和终点end在河岸线要素haFeatures上裁剪线段haLine；

f. 分析haPoint是否在haLine上；

g.如果在haLine上，计算haLine线长度；

h.如果不在haLine上，则根据haPoint分割haLine线为两端分别为haLine1，haLine2；

i.根据起点start和中点center在线要素haLine1上裁剪线段haLine1丿；计算haLine1丿的长度；根据中点center和终点end在线要素haLine2上裁剪线段haLine2丿；计算haLine2丿的长度；

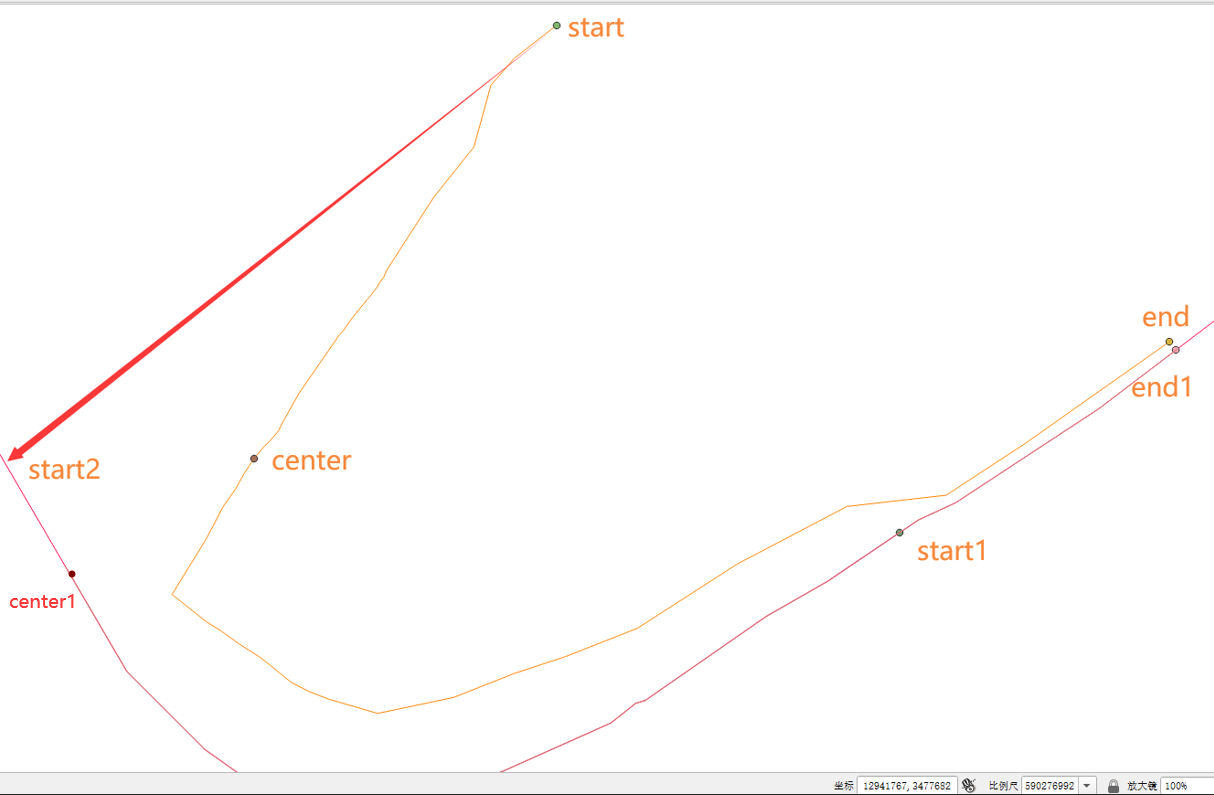
k. j和i计算的长度累加求和得到最终该图层治理工程量；

l.重复上面步骤计算其他区域长度

说明：f,g,h,i步骤主要出现最近点不在理想范围内的情况采取的一种措施。

如图，“崩岸11”的始、中、终点分别是start、center、end，在河岸上的最近点分别是start1、center1、end1；该方式得到的线段是start1-end1计算长度显然不对；理论上应该是start2-end1线段为实际情况。

所以引用中点来判断center1是否在映射的start1-end1河岸线内，如果不在需要以center1进行裁剪，进而转成线段start2-center1和线段center1-end1来计算长度。



1. 样例数据显示

涉及知识：GeoJSON、OpenLayers、Spring Boot

数据显示将读取的gdb数据整理为GeoJSON格式，返回给前端；前端接收GeoJSON数据，使用OpenLayer显示。

最后在浏览器输入：<http://localhost:8080> 显示Web页面，最终效果如图：



