# 地理位置子服务代码测试报告

|  |  |  |
| --- | --- | --- |
| 测试人员 | 测试开始日期 | 测试结束日期 |
| 王凯能 | 2018/12/10 | 2018/10/11 |

地理位置子服务代码测试报告 1

一、 引言 3

1.1 编写目的 3

二、 测试明细 4

## 引言

### 编写目的

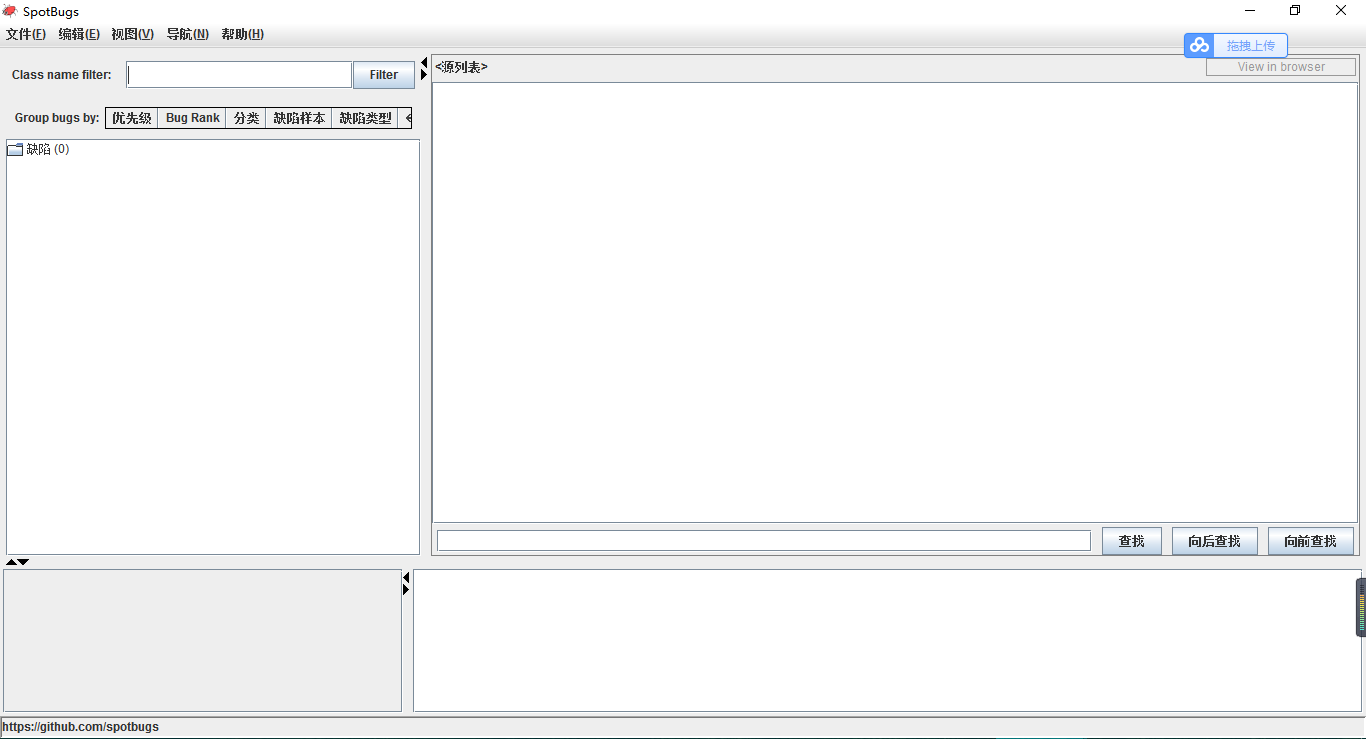
本报告为《地理位置子服务》的代码测试报告，目的在于提升代码的可读性、可维护性，建立团队共同的编码风格，提升业务系统的代码质量。

本报告的代码测试使用SpotBugs进行代码静态检查。

## 测试明细

1. 启动SpotBugs工具

我们的代码测试，统一使用图形化SpotBugs工具进行检测，可以通过命令行直接启动，启动后界面如下图：

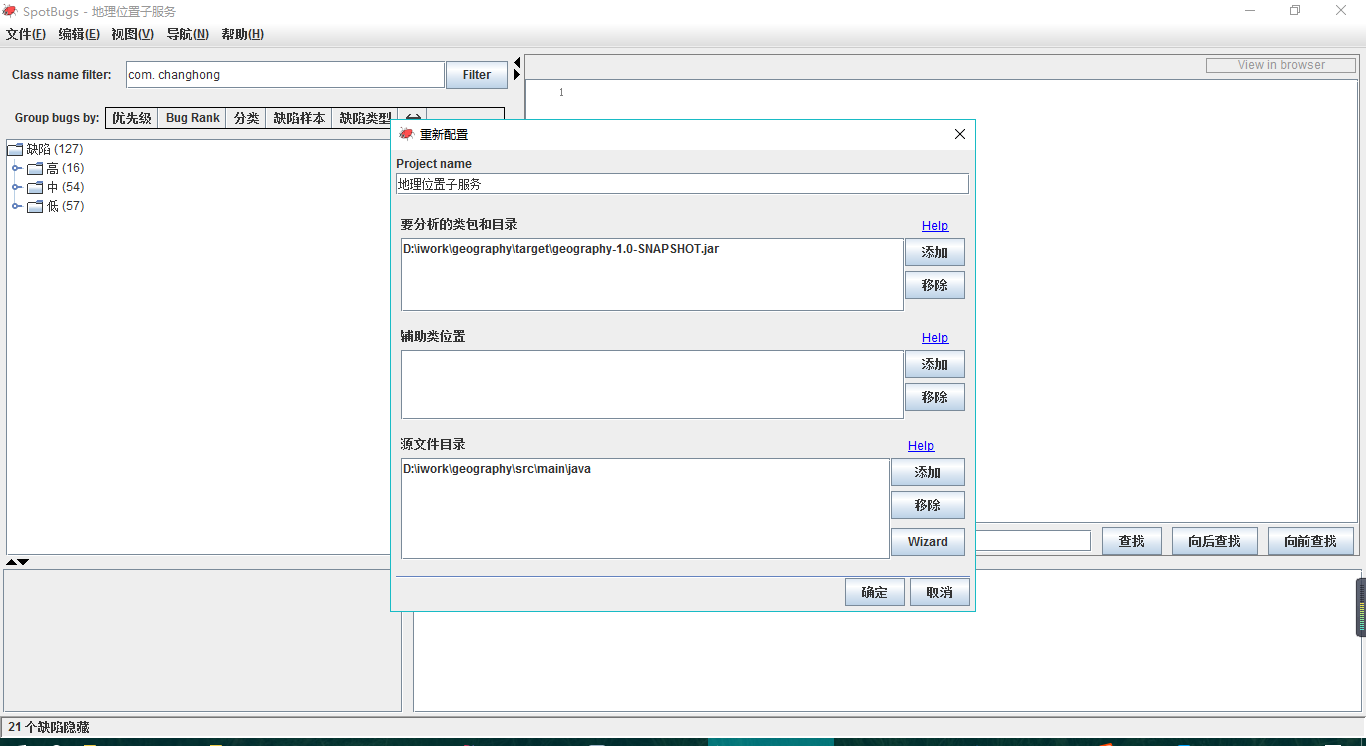


1. 选择项目编译后Jar包及源文件目录

SpotBugs工具启动后，我们选择“项目-》新建”来启动一个新的代码测试，如下图所示。

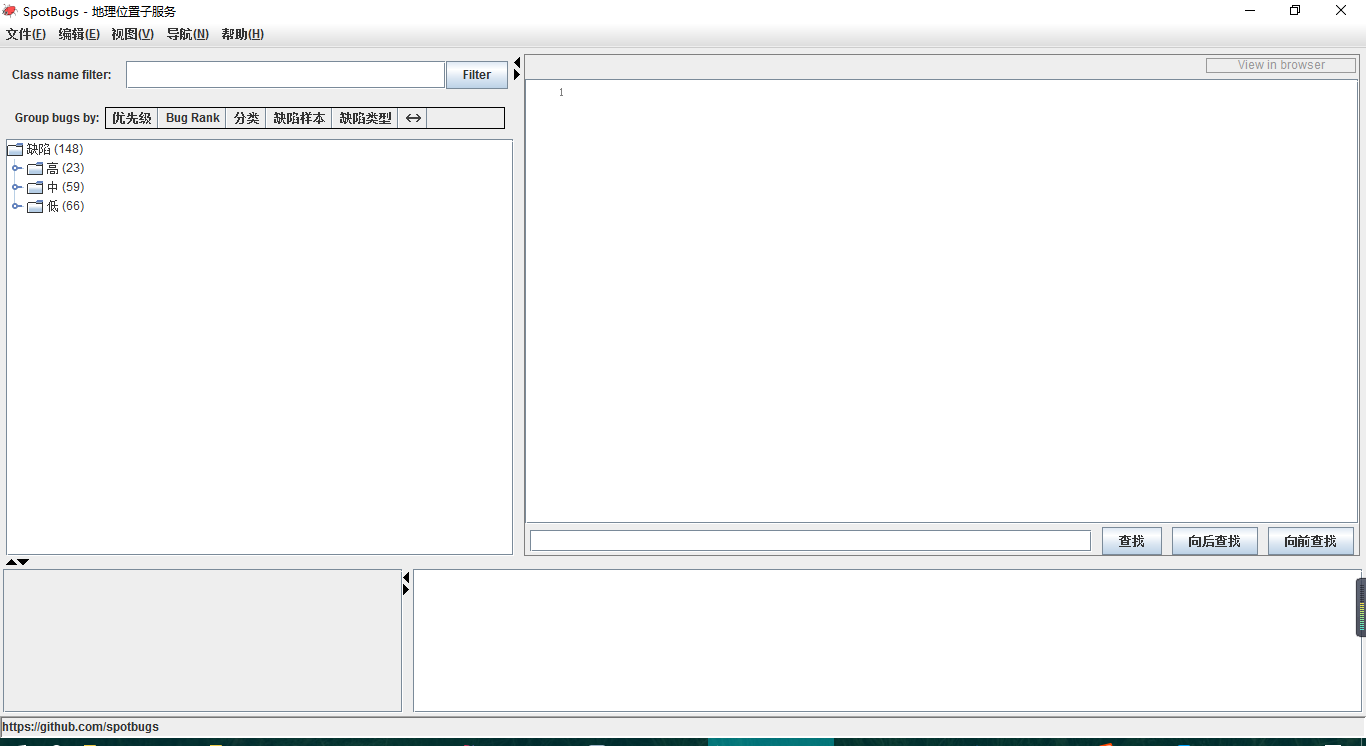
其中，我们需要填入项目名称，选择要检查的类Jar包或者包含class文件的目录，以及相应的源文件根目录。

此外，如果我们的代码依赖一些第三方的jar包，我们也可以通过添加辅助类的形式进行增加。



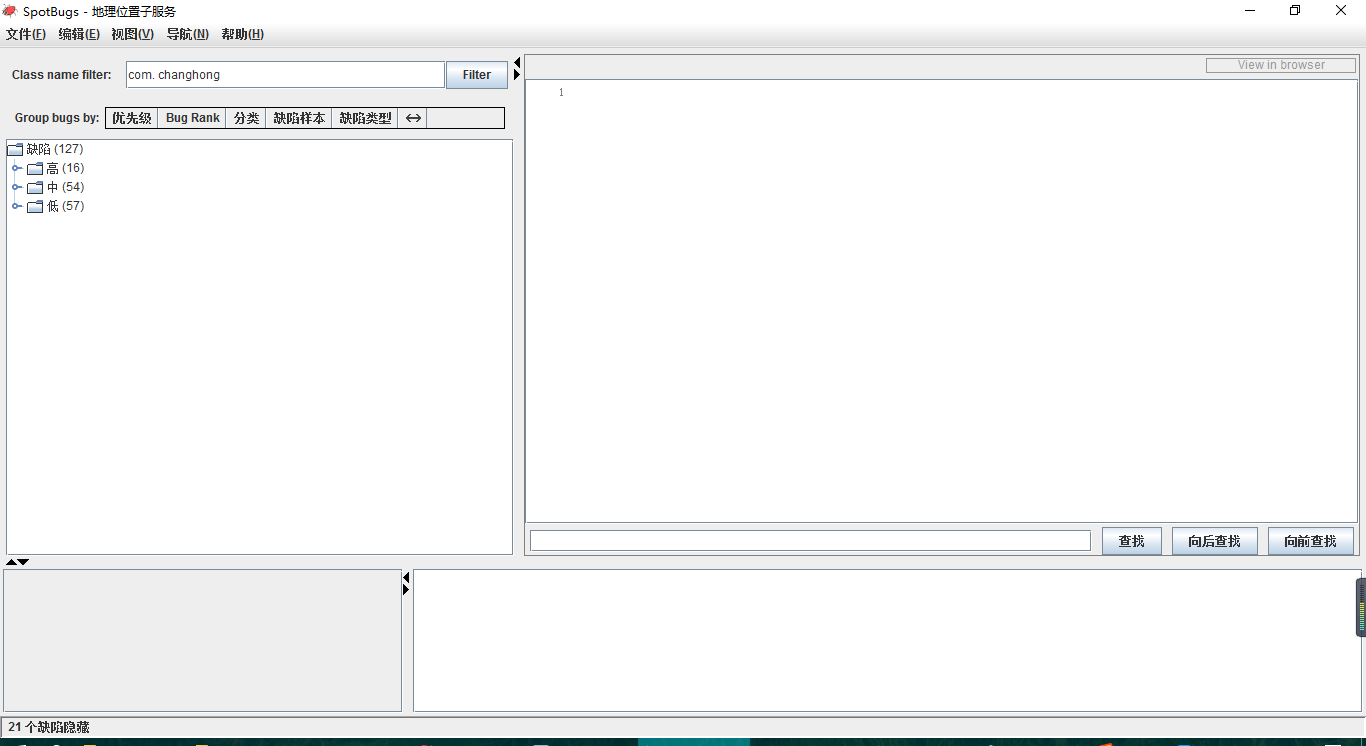
1. 分析代码，并查看分析结果

项目配置完毕后，我们选择“分析”按钮，SpotBugs就会自动分析我们的代码：



从上图可以发现，地理位置子服务共发现148项缺陷，其中高风险缺陷23项，中等风险缺陷59项，低风险缺陷66项目，除第三方依赖包代码的缺陷外，我们至少需要解决我们代码中全部的高风险和中风险级别代码缺陷。

下图是过滤第三方后的：



从上图可以发现，地理位置子服务共发现127项缺陷，其中高风险缺陷16项，中等风险缺陷54项，低风险缺陷57项目，我们需要解决我们代码中全部的高风险和中风险级别代码缺陷。

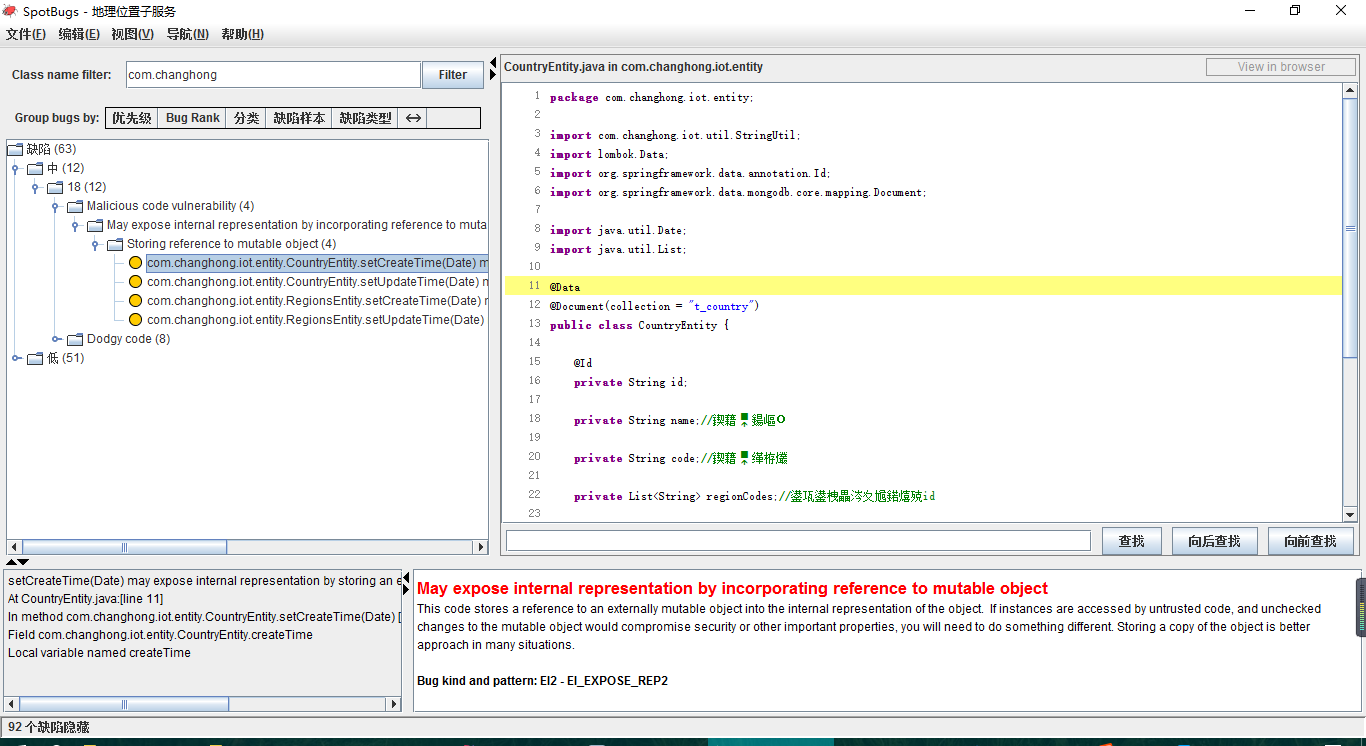
1. 修改SpotBugs的各项缺陷

我们根据SpotBugs的提示，修改所有除第三方依赖包代码的缺陷，解决我们自身代码中全部的高风险和中风险级别代码缺陷。缺陷修改完毕后，代码功能测试正常。

1. 使用SpotBugs重新分析修改后的版本

再次启动SpotBugs，并按照步骤1~3进行代码静态测试。

此处需要截图证明。



上图中中风险级别还剩12个，其中的风险不是由代码引起，绝大部门是由lombok插件引起。还有一些没必要的更改的风险。

1. 结论

我们使用代码静态分析工具SpotBugs发现了大量的代码缺陷，并根据SpotBugs的提示对这些缺陷进行了修复，目前主要解决的是SpotBugs提示的全部中、高风险等级的缺陷，共计修复高风险缺陷16个，中风险缺陷42个。