实验 1 静态白盒测试

四川大学计算机（软件）学院 杨秋辉 yangqiuhui@scu.edu.cn

1. 实验介绍

本实验要求学生使用静态测试工具进行代码静态分析，并能够理解和掌握静态代码分析工具的使用。

在业界，静态代码分析是高级的缺陷查找技术。目前有较多的商用和开源静态代码分析工具，它们针对不同的语言，比如 Java，C/C++，C#等。本实验中，要求学生使用开源工具 FindBugs（SpotBugs），针对 Java 程序进行静态代码分析。

1.1 实验目的

1. 巩固软件代码静态分析的基本概念；

2. 了解并使用静态白盒测试工具 SpotBugs，看工具如何帮助程序员找出程序缺陷，这些缺陷是大多数的编译器不能发现的，也有一些是程序员进行人工代码检查时不容易发现的，甚至有的缺陷是动态软件测试也很难发现的。

1.2 实验工具

本实验使用的工具是 FindBugs（Spot Bugs）。

FindBugs 是由马里兰大学提供的一款开源 Java 静态代码分析工具。FindBugs 通过检查类文件或 JAR 文件，将字节码与一组缺陷模式进行对比从而发现代码缺陷，完成静态代码分析。FindBugs 既提供可视化 UI 界面，同时也可以作为 Eclipse 插件使用。本实验选择的是将 FindBugs 作为 Eclipse 插件来使用。在安装成功

后会在 Eclipse 中增加 FindBugs 选项，用户可以对指定 Java 类或 JAR 文件运行 FindBugs，此时 FindBugs 会遍历指定文件，进行静态代码分析。从版本 3.1.0 开始，FindBugs 已更名为 SpotBugs，当前最新版本为 4.0.

1.3 被测系统

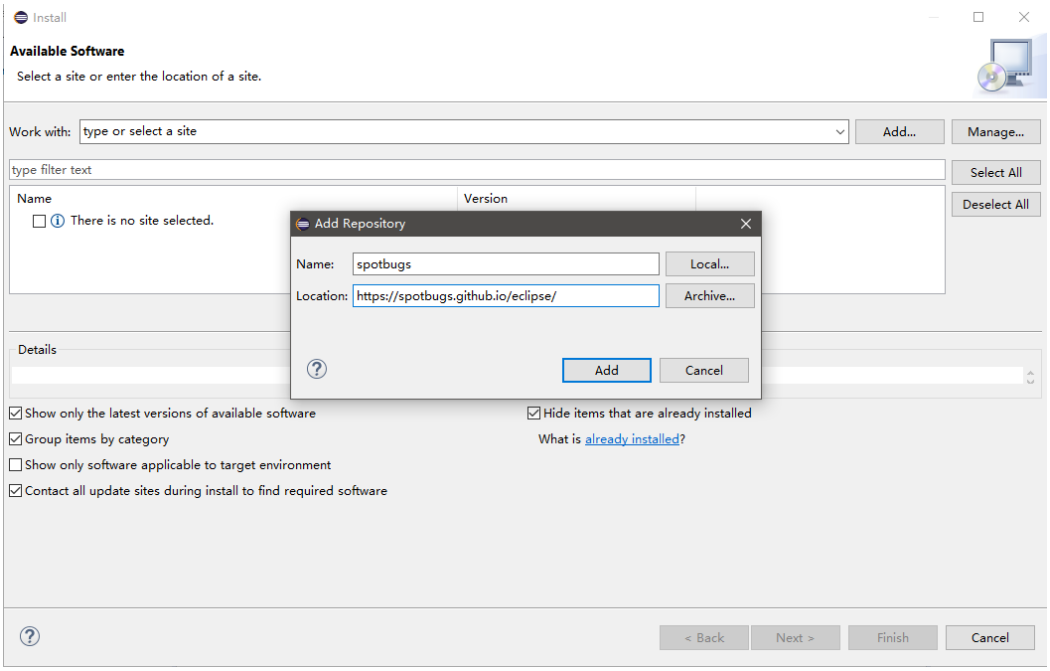
本实验的被测系统是教师指定的几个例子（见下文），以及学生自己的 Java 程序。

2. 实验步骤

2.1 安装 SpotBugs 插件

1） 启动 Eclipse，依次点击 Help->Install new software...->Add

2） 输入仓库名和地址 https://spotbugs.github.io/eclipse/



3） 点击 Add 后，选中 SpotBugs->接受协议->等待安装，重启 Eclipse 后生效。

4） 如果安装成功，Eclipse 中可以看到 SpotBugs 选项（点击 windows->show view->other），见图 1；找到 SpotBugs，打开 Bug Explorer 和 Bug Info 窗口。

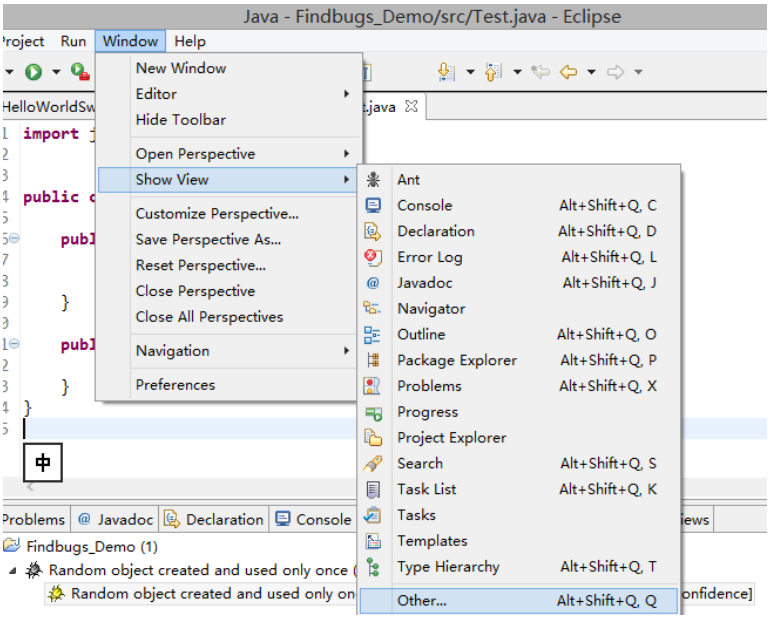


图 1　调出 Spotbugs 窗口步骤 1

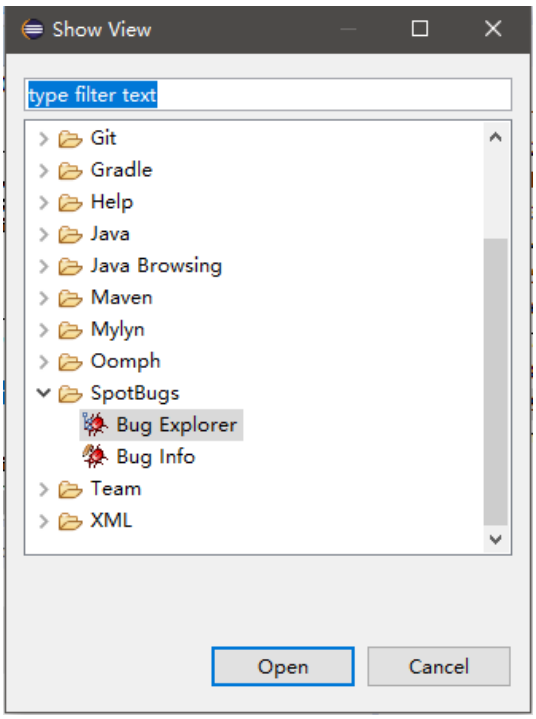


图 2　调出 Spotbugs 窗口步骤 2

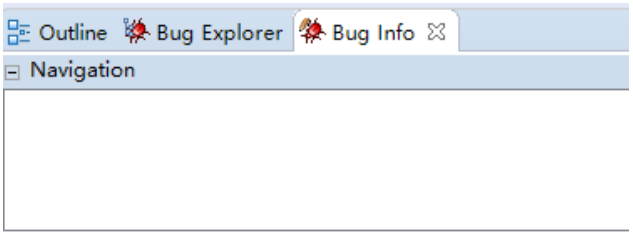


图 3　调出 Spotbugs 窗口步骤 3

到此，SpotBugs 插件安装成功。

2.2 熟悉测试工具

2.2.1 一个简单的测试流程

1) 新建一个项目（项目名可以为：FindbugsTest），然后新建一个类（FindbugsTest），在该类中添加一个方法，如下：

被测程序 1：

```
public static void func1() {
    String str = "check_string";
    if (str != null) {
        if (str != null) {
            System.out.println(str);
        }
    }
}
```

阅读程序，你认为有什么缺陷？

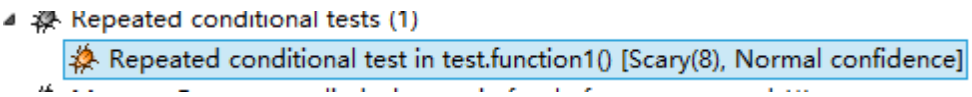
2) 接着，写一个 main 方法，在 main 方法中调用 func1 方法，如下：

```
public static void main(String[] args) {

    func1();
```

3) 然后点击右键->Spot Bugs-> Find Bugs，查看 Bug Explorer 窗口，你会看到 SpotBugs 找到的 bugs 列表。查看列表，如下：

Bug Explorer 窗口：



Bug Info 窗口：

Bug: Repeated conditional test in test.function1()

The code contains a conditional test is performed twice, one right after the other (e.g., x == 0 || x == 0). Perhaps the second occurrence is intended to be something else (e.g., x == 0 || y == 0).

Rank: Scary (8), confidence: Normal
Pattern: RpC_REPEATED_CONDITIONAL_TEST
Type: RpC, Category: CORRECTNESS (Correctness)

4) 分析：很明显，SpotBugs 提示：重复的条件控制语句，即有一个 if 语句是冗余的。去掉一个 if（str!=null）即可。

2.2.2 被测程序

5) 阅读下面的各被测程序段，你认为有哪些缺陷？记录下来。

被测程序 2：

```
private static void Func2(){
    System.out.println("chekc fun");
}
```

被测程序 3：

```
private static String StrAb;
private static String func3(){
    return StrAb;
}
```

被测程序 4：

```
public static void func4() {
    String str = "software testing.";
    System.out.println(str);
    ;
}
```

被测程序 5：

```
public static void func5() {
    String str1 = "123";
    String str2 = "456";
    String str3 = String.format("{0} {1}", str1, str2);
    System.out.println(str3);
}
```

被测程序 6：

```
public static void func6() {
    String str = "function 4";
    System.out.println(str.toString());
}
```

被测程序 7：

```
public static int func7(int seed) {
    return new Random(seed).nextInt();
}
```

被测程序 8：

```
public static void func8() {
    int[] arr = { 1, 2, 3, 4, 5 };
    for (int i = 0; i < 7; i++) {
        System.out.println(arr[i]);
    }
}
```

被测程序 9：

```
public static void func9() {
    String str = "123";
    StringBuffer strb = new StringBuffer("123");
    System.out.println(str.equals(strb));

}
```

主程序：

```
public static void main(String[] args) {

    func1();
    Func2();
    func3();
    func4();
    func5();
    func6();
    func7(10);
    func8();
    func9();
}
```

注：建议把 main 函数和 9 个被测代码函数写在同一个类里面。如果你觉得有更好的方法，也可以按照自己的方式编写代码。

2.2.3 进行静态代码测试

6) 输入以上测试代码（被测程序 2~被测程序 9），用 SpotBugs 分析（点击右键->Spot Bugs-> Find Bugs，查看 Bug Explorer 窗口，会看到 SpotBugs 能找到的 bugs 列表，点击列表，查看 Bug 详情）。

7) 对比 SpotBugs 找到的缺陷和你认为的缺陷是否一致。

2.3 测试结果分析

8) 针对上面的各段代码（被测程序 2~被测程序 9）及其测试结果，分情况写出你的理解：

（1）是缺陷。你认为属于什么缺陷？是否值得修复？若不修复，你认为会有什么潜在的危险？

（2）不是缺陷（你认为是误报）。分析一下：为什么静态测试工具会认为是缺陷？

将测试结果记录在实验报告的“表 1 SpotBugs 静态测试结果分析表”中。

9) 针对找到的缺陷，看哪些是你能够修改的？并对修改后的程序重新进行静态测试（即重新执行测试，看分析后的结果中，哪些缺陷仍然存在）。将你的修改和再次静态测试的结果记录在实验报告中的 “表 1Spotbugs 静态测试结果分析表”中。

10) 被测程序中还有哪些缺陷是静态分析没有找到的？分析一下为什么静态分析不能发现这些缺陷？请填写在实验报告中的“表 2 Spotbugs 静态测试遗漏缺陷表”中。

2.4 添加你的被测代码

11) 你还能想到哪些在 Java 编程中容易出现的缺陷？请写出示例代码（至少 5 个），然后交给 SpotBugs 测试，在实验报告中写出你的被测代码，并记录测试结果。

提示：可以访问链接 https://spotbugs.readthedocs.io/en/latest/bugDescriptions.html，了解 Spotbugs 能检测的 Bug 类型，然后写你的示例代码。

2.5 撰写实验报告

根据给出的实验报告模板“实验 1-静态白盒测试(SpotBugs)-实验报告模板”，撰写实验报告，可以根据需要修改部分章节的标题。

提交时的文件格式为 doc 文件，命名方式：“学号_姓名_静态白盒测试实验报告.doc”。

3. 实验总结

完成本实验之后，学生可以对静态代码测试工具有一个基本理解，知道静态测试工具能够带来什么好处，同时可以看到静态测试在能力上的局限性。更多有关 Spotbugs 的信息请访问 https://spotbugs.readthedocs.io/en/latest/index.html