

**wink测试文档**

新闻分享app测试文档



**指导教师：**

**杜庆峰**

**小组成员：**

**1752026 李航**

**1652977 王星洲**

**1650350 乔宇**

**1753499 潘小逸**

目录

[修订历史 2](#_Toc29339911)

[1. 介绍 3](#_Toc29339912)

[1.1 目的 3](#_Toc29339913)

[1.2 软件团队 3](#_Toc29339914)

[2. 开发具体内容 3](#_Toc29339915)

[2.1 敏捷开发 3](#_Toc29339916)

[2.2 需求分析 4](#_Toc29339917)

[2.3 前端开发过程 4](#_Toc29339918)

[2.4 后端开发过程 5](#_Toc29339919)

[3. 软件过程管理工具 5](#_Toc29339920)

[4. 版本控制工具 7](#_Toc29339921)

# 修订历史

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 编写日期 | SEPG | 版本 | 说明 | 作者 | 评审时间 | 评审人员 | 批准日期 | 签字人员 |
| 2019/11/15 | wink | 1.0.01115\_alpha | 初稿，第一次Scrum例会 | 王星洲 | 2019/11/15 | 李航 | 2019/11/15 | 李航 |
| 2019/11/26 | wink | 1.0.11126\_alpha | 框架完成 | 李航 | 2019/12/03 | 王星洲 | 2019/12/03 | 王星洲 |
| 2019/12/03 | wink | 1.1.01203\_alpha | sprint1冲刺完成 | 乔宇 | 2019/12/03 | 李航 | 2019/12/03 | 李航 |
| 2019/12/10 | wink | 1.121210\_alpha | 需求大变更，代码重构 | 潘小逸 | 2019/12/10 | 李航 | 2019/12/10 | 李航 |
| 2019/12/17 | wink | 1.2.01217\_alpha | sprint2完成 | 王星洲 | 2019/12/18 | 李航 | 2019/12/18 | 李航 |
| 2019/12/23 | wink | 1.4.51223\_alpha | sprint3完成，需求变更，代码重构 | 潘小逸 | 2019/12/24 | 李航 | 2019/12/24 | 李航 |
| 2019/12/27 | wink | 2.0.01227\_release | sprint4完成，集成测试完成，发行版本 | 王星洲 | 2019/12/27 | 李航 | 2019/12/27 | 李航 |
| 2020/01/07 | wink | 2.0.30107\_release | 修改部分bug，最终版本 | 李航 | 2020/01/07 | 潘小逸 | 2020/01/07 | 潘小逸 |

# 1. 介绍

## 1.1 目的

本文档是wink软件的测试文档，旨在说明测试软件的过程中采用的软件工程方法，总结涉及到的软件工程知识。在本文档中，我们会说明随着开发进程，我们所采用的一些测试用工具，代码质量检测工具，部分单元测试等内容。

# 2. 测试工具

## 2.1 JUnit

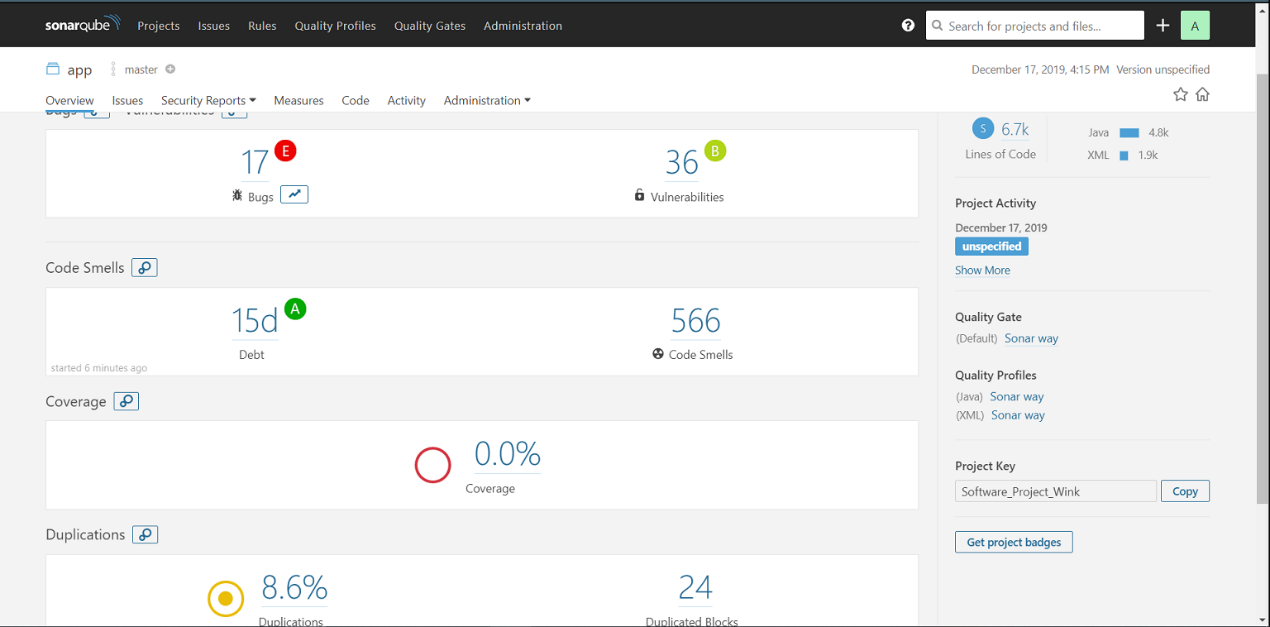
在系统的技术架构中，我们的前端和后端都使用了JUnit作为单元测试工具，不同的是，前端是Android项目，使用了Android-JUnit4，使用gradle安装依赖，而后端是SpringBoot项目，采用了最新的JUnit5。

这里我将举一个使用JUnit的例子，在我们的开发文档中，我们提到过新闻的获取并没有成功，这时我们采用一个JUnit测试，来测试我们的请求是否发送过，发送过几次。



## 2.2 SonarCube

我们在前端采用了SonarCube作为代码质量检测工具，使用了SonarCube7.7版本，因为7.8以上的版本逐渐放弃了对JDK1.8的支持，以下是我们使用SonarCube对代码质量进行检测的一个页面：



我们可以参照这个页面进行更清晰的bug定位，和对不规范的代码进行修改。

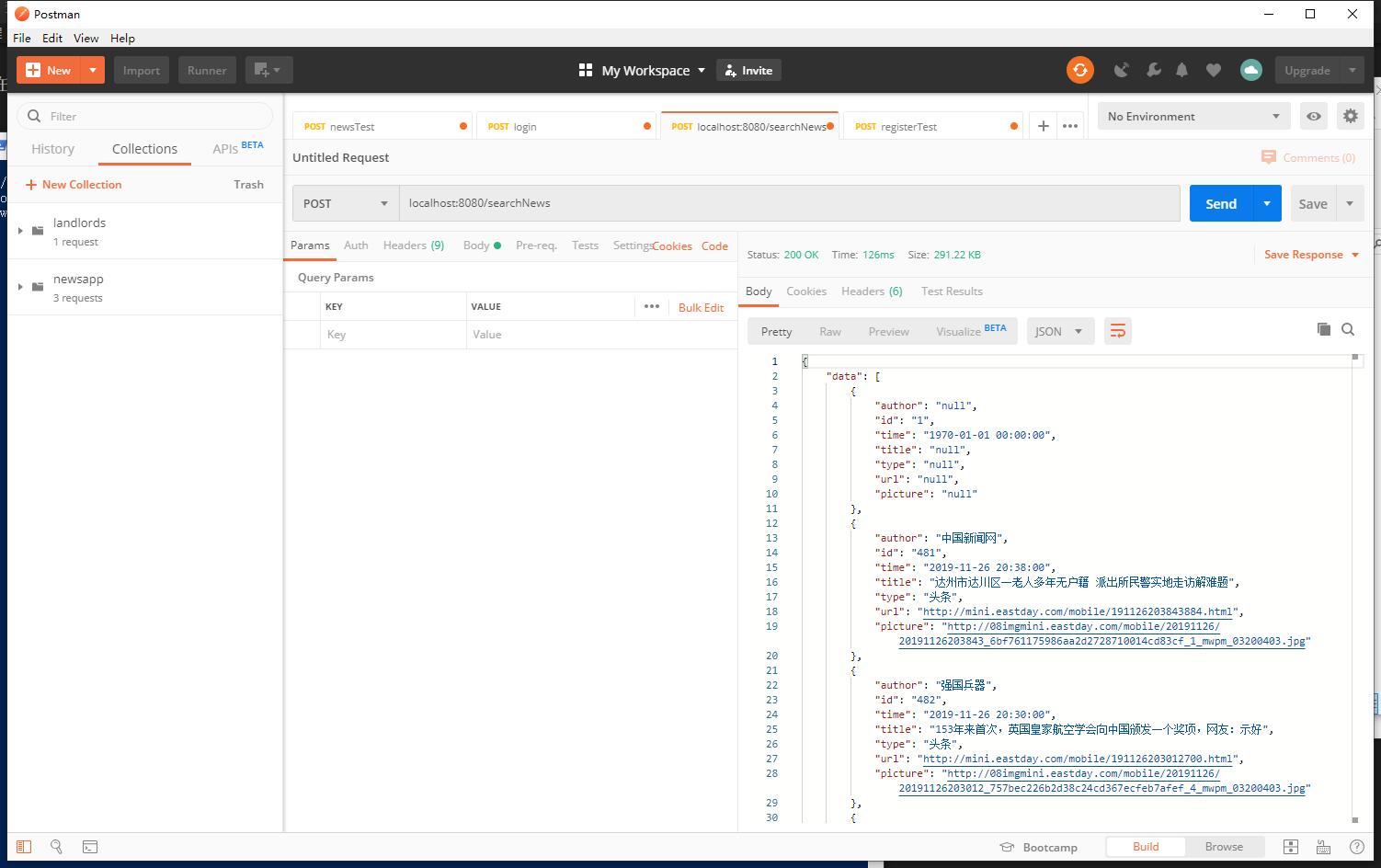
## 2.3 leakcanary

在操作过程中，我们的程序经常因为各种原因崩溃，我们曾经冥思苦想找不到思路，会不会是内存发生泄露了呢？为了验证这个猜想，我们使用了LeakCanary，它可以在程序执行过程中检测某个Activity是否发生了内存泄露，并将定位该Activity：



## 2.4 postman

在编码过程中，为了保障后端接口功能的正确性与可靠性，我们在发布前对每一个接口使用Postman进行测试，对同一个接口使用不同的请求参数多次进行测试，保障了后端接口功能的正确性，便于前端的请求访问。



## 2.5 Performanc Runner

我们在项目开发的过程中使用了轻量级的Web压力测试工具PerformanceRunner，编写压测脚本模拟用户登录，多次浏览新闻，添加评论等等的操作。由于免费版的规模限制，我们只能请求30VU以内的访问数量，在该规模的压力测试下表现良好。

