

# 性能测试设计与执行

软件质量保障与测试课程 Lab6 课程作业（第 9 组）

Tian, Jiahe<sup>\*</sup>    Hu, Xiaoxiao<sup>†</sup>    Huang, Jiani<sup>‡</sup>    Liu, Jiaxing<sup>§</sup>  
Shi, Ruixin<sup>¶</sup>    Wu, Chenning<sup>||</sup>    Zhang, Cenyuan<sup>\*\*</sup>  
Zhang, Yihan<sup>††</sup>    Wang, Chen<sup>‡‡</sup>

2020 年 5 月 14 日

---

<sup>\*</sup>Equal Contribution, Fudan University, 17307130313 ([tianjh17@fudan.edu.cn](mailto:tianjh17@fudan.edu.cn))

<sup>†</sup>Equal Contribution, Fudan University, 17302010077 ([xxhu17@fudan.edu.cn](mailto:xxhu17@fudan.edu.cn))

<sup>‡</sup>Equal Contribution, Fudan University, 17302010063 ([huangjn17@fudan.edu.cn](mailto:huangjn17@fudan.edu.cn))

<sup>§</sup>Equal Contribution, Fudan University, 17302010049 ([jiaxingliu17@fudan.edu.cn](mailto:jiaxingliu17@fudan.edu.cn))

<sup>¶</sup>Equal Contribution, Fudan University, 17302010065 ([rxshi17@fudan.edu.cn](mailto:rxshi17@fudan.edu.cn))

<sup>||</sup>Equal Contribution, Fudan University, 17302010066 ([cnwu17@fudan.edu.cn](mailto:cnwu17@fudan.edu.cn))

<sup>\*\*</sup>Equal Contribution, Fudan University, 17302010068 ([cenyuanzhang17@fudan.edu.cn](mailto:cenyuanzhang17@fudan.edu.cn))

<sup>††</sup>Equal Contribution, Fudan University, 17302010076 ([zhangyihan17@fudan.edu.cn](mailto:zhangyihan17@fudan.edu.cn))

<sup>‡‡</sup>Equal Contribution, Fudan University, 16307110064 ([wangc16@fudan.edu.cn](mailto:wangc16@fudan.edu.cn))

# 性能测试设计与执行

软件质量保障与测试课程 *Lab7* 课程作业

## 摘要

本次作业为软件质量保障与测试课程的 Lab6 课程作业，需要我们以小组为单位完成对出题系统的性能测试。本文档分为两小节。第一小节介绍了本小组进行性能测试采用的策略；第二小节介绍了性能测试的结果及系统性能分析。

## 关键词

系统与软件工程; 系统与软件质量要求和评价; 测试文档

目录	3
----	---

## 目录

摘要	2
关键词	2
1 Sonar 工具静态测试	4
1.1 测试结果完整报告 . . . . .	4
1.2 测试结果核心内容截图 . . . . .	4
1.3 测试结果分析 . . . . .	4
2 p3c 工具静态测试	4
2.1 测试结果完整报告 . . . . .	4
2.2 测试结果核心内容截图 . . . . .	4
2.3 测试结果分析 . . . . .	5
2.3.1 阿里巴巴 JAVA 开发手册 . . . . .	5
2.3.2 扫描效果 . . . . .	5
2.3.3 插件使用 . . . . .	5
3 jshint 工具静态测试	6
3.1 测试结果完整报告 . . . . .	6
3.2 测试结果核心内容截图 . . . . .	6
3.3 测试结果分析 . . . . .	6
3.3.1 配置项 . . . . .	7
3.3.2 扫描效果 . . . . .	8
3.3.3 工具使用 . . . . .	8
4 不同工具之间的对比分析	8
5 测试总结	8
参考文献	9

## 1 Sonar 工具静态测试

### 1.1 测试结果完整报告

(currently left blank)

### 1.2 测试结果核心内容截图

### 1.3 测试结果分析

## 2 p3c 工具静态测试

### 2.1 测试结果完整报告

请通过网址[第 9 组 P3C 报告网站](#)来访问本小组的 P3C 完整报告。

### 2.2 测试结果核心内容截图



如图所示,可分为 Blocker、Critical、Major 三部分。

#### IntelliJ IDEA inspection report:

##### Inspection tree:

- [-] 'InspectionViewTree' project 172 blockers 99 criticals 628 majors
  - [-] Blocker 172 blockers
  - [-] Critical 99 criticals
  - [-] Major 628 majors

##### Problem description:

Select a problem element in tree

## 2.3 测试结果分析

P3C 是阿里巴巴推出的《阿里巴巴 Java 开发规约》扫描插件，目前在 IDEA 和 Eclipse 都有较好的支持。P3C 扫描结果文档给出了项目的 bug，并根据 bug 的严重程度分为三个级别展示。三个级别分别是：Blocker, Critical, Major，严重程度由高到低。

### 2.3.1 阿里巴巴 JAVA 开发手册

根据《阿里巴巴 JAVA 开发手册》版本 1.3.0，我们可以对扫描结果进行分析。该版本对 JAVA 开发的规约含有编程规约、异常日志、单元测试、安全规约等六大类，而本项目的扫描结果重点体现的是编程规约。编程规约在开发手册共分为 9 类：

- 命名规范
- 常量定义
- 代码格式
- OOP 规约
- 集和处理
- 并发处理
- 控制语句
- 注释规约
- 其他

### 2.3.2 扫描效果

根据扫描结果可以发现：同一类型的编程规约由于其严重性不同可以划分为不同级别的 bug。比如注释规约中，抽象方法的 javadoc 注释属于 Major，而枚举字段的注释属于 Critical；命名规范中也有类似的例子。另外，bug 的严重程度分类比较合理。注释规约、命名规范的约定内容严重程度较低，并发处理、控制语句的约定内容严重程度较高。整体上，P3C 扫描插件发现的问题比较基础，它侧重 JAVA 编程细节可能导致的系统失效。

### 2.3.3 插件使用

P3C 插件有以下的优点：

- 基本满足代码规范检测的需求。
- 能够检测出细致和易忽略的问题，可以提高开发过程中对代码细枝末节的注意。

- Quick fix，检测出问题后可以快速查看 bug 位置、解决方案并一键替换。
- 中文提示，解释内容与开发手册一致。

此外 P3C 插件也存在平台限制、功能不成熟、扫描能力有限等问题。

### 3 jshint 工具静态测试

#### 3.1 测试结果完整报告

(currently left blank)

#### 3.2 测试结果核心内容截图

如图所示，共发现 236 个 failures, 1 个 error 和 235 个 warnings。其中每

一项都标明了具体的行数、问题代码和问题原因。

Code	Line	Column	Evidence	Reason
./webapp/assets/js/ajax-utils.js				
W097	1	1	'use strict';	Use the function form of "use strict".
W033	60	11	}}	Missing semicolon.
W033	61	2	}	Missing semicolon.
W117	12	16	var data = _._isNull(param)    _._isUndefined(param) ? null : param;	'_' is not defined.
W117	12	35	var data = _._isNull(param)    _._isUndefined(param) ? null : param;	'_' is not defined.

#### 3.3 测试结果分析

JSHint 是由 Anton Kovalyov 基于 JSLint 的代码实现的开源项目，JSHint 与 JSLint 相比较之下，更友好，也更容易配置，所以发展很快并得到了众多 IDE 和编辑器的支持。JSHint 是一个 JavaScript 语法和风格的检查工具，但检查不出逻辑问题。它可以根据配置参数扫描 JavaScript 代码，分析其中的语法与风格从而给出代码质量报告。

### 3.3.1 配置项

```
{
  "curly": true,
  "eqnull": true,
  "eqeqeq": true,
  "undef": true,
  "browser": true,
  "devel": true,
  "globals": {
    "jQuery": true,
    "$": true,
    "CONTEXT": true,
    "AjaxUtils": true,
    "_": true,
    "Handlebars": true,
    "Dialogs": true,
    "DateTimeUtils": true,
    "QuestionUtils": true,
    "TableFilter": true,
    "moment": true,
    "BootstrapDialog": true,
    "DatePickerUtil": true,
    "SyllabusUtils": true,
    "Navigation": true
  }
}
```

JSHint 工具使用的关键是配置项。如果不设置配置项，那么可能会有很多“假”错误或警告。比如自定义全局变量，不同脚本上下文的符号引用。这些内容既不算错误的语法，也不算差劲的风格，可是 JSHint 依旧把他们认错了。JSHint 有一些常见的配置项：

- “strict”: true 严格模式
- “asi”: true 允许省略分号
- “bitwise”: true 禁止使用位运算符，比如经常把 && 写错 & 规避此错误
- “eqeqeq”: true 禁止使用 == 和 != 强制使用 === 和 !==
- “undef”: true 禁止使用不在全局变量列表中的未定义变量
- “curly”: true 循环或者条件语句必须使用花括号包住
- “devel”: true 定义用于调试的全局变量：console,alert
- “jquery”: true 定义全局暴露的 jQuery 库（可以去掉）
- “browser”: true 暴露浏览器属性的全局变量如 window document
- “globals”: {“\$”:true,“require”:true,“\_”:true}

JSHint 的配置项一般是放入项目根目录下的 .jshintrc 文件中，JSHint 工具在扫描的时候就会运用这些配置项。配置项的作用一方面是明确项目

的编程规范，约束开发人员的行为。另一方面是避免某些规范，减轻开发人员的负担（比如允许省略分号等）。

### 3.3.2 扫描效果

不同的配置项扫描结果是不尽相同的。项目最初未定义配置项中的全局变量，这导致了非常多的未定义错误或警告。这显然不是我们需要的扫描结果。完成配置项后，扫描结果报告逐渐明晰，其中报告了出现了缺少分号、使用 `==` 和 `!=` 错误、缺少 `{或}` 错误和未定义变量等错误。正如上面所言，JSHint 可以分析出 JavaScript 代码的语法和风格，但是它无法识别出项目的逻辑错误，比如冗余代码、死循环、无效代码等问题。它的分析功能是非常基础和有限的。

### 3.3.3 工具使用

JSHint 工具不可否认具有一定的优点：

- 有了很多参数可以配置
- 支持配置文件，方便使用
- 支持了一些常用类库（如 jquery 等）
- 支持了基本的 ES6

但与其他功能强大的 Web 项目静态扫描工具比较而言，它具有不支持自定义规则、无法根据错误定位到对应的规则和不提供快捷的修正方式等缺点。不支持自定义规则自然让它只能检测出很基础的预定义规则，无法根据错误定位到对应的规则使得扫描结果不易阅览，不提供快捷的修正方式（不能跳转到指定代码位置和不能提供修正方案）自然无法让开发人员方便的处理扫描结果。

## 4 不同工具之间的对比分析

## 5 测试总结



## 参考文献

International Organization for Standardization. 2014. *Systems and Software Engineering — Systems and Software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*. International Organization for Standardization. Vol. 2014. <https://www.iso.org/standard/64764.html>.

中国国家标准化管理委员会. 2016. *GB/T 25000.51-2016《系统与软件工程系统与软件质量要求和评价 (SQuaRE) 第 51 部分: 就绪可用软件产品 (RUSP) 的质量要求和测试细则》*. 系统与软件工程系统与软件质量要求和评价 (SQuaRE). Vol. 51. 中国国家标准化管理委员会. <http://openstd.samr.gov.cn>.

———. 2017a. *GB/T 25000.12-2017《系统与软件工程系统与软件质量要求和评价 (SQuaRE) 第 12 部分: 数据质量模型》*. 系统与软件工程系统与软件质量要求和评价 (SQuaRE). Vol. 12. 中国国家标准化管理委员会. <http://openstd.samr.gov.cn>.

———. 2017b. *GB/T 25000.24-2017《系统与软件工程系统与软件质量要求和评价 (SQuaRE) 第 24 部分: 数据质量测量》*. 系统与软件工程系统与软件质量要求和评价 (SQuaRE). Vol. 24. 中国国家标准化管理委员会. <http://openstd.samr.gov.cn>.

———. 2018. *GB/T 25000.40-201《系统与软件工程系统与软件质量要求和评价 (SQuaRE) 第 40 部分: 评价过程》*. 系统与软件工程系统与软件质量要求和评价 (SQuaRE). Vol. 40. 中国国家标准化管理委员会. <http://openstd.samr.gov.cn>.

———. 2019. *GB/T 25000.23-2019《系统与软件工程系统与软件质量要求和评价 (SQuaRE) 第 23 部分: 系统与软件产品质量测量》*. 系统与软件工程系统与软件质量要求和评价 (SQuaRE). Vol. 23. 中国国家标准化管理委员会. <http://openstd.samr.gov.cn>.