



CS2001

软件工程

7. 软件过程与团队合作 — 团队与协作

团队



有一致的目标
有各自的分工
相互依赖
相互合作

从全能乐手到交响乐队



全能乐手
“全栈”工程师

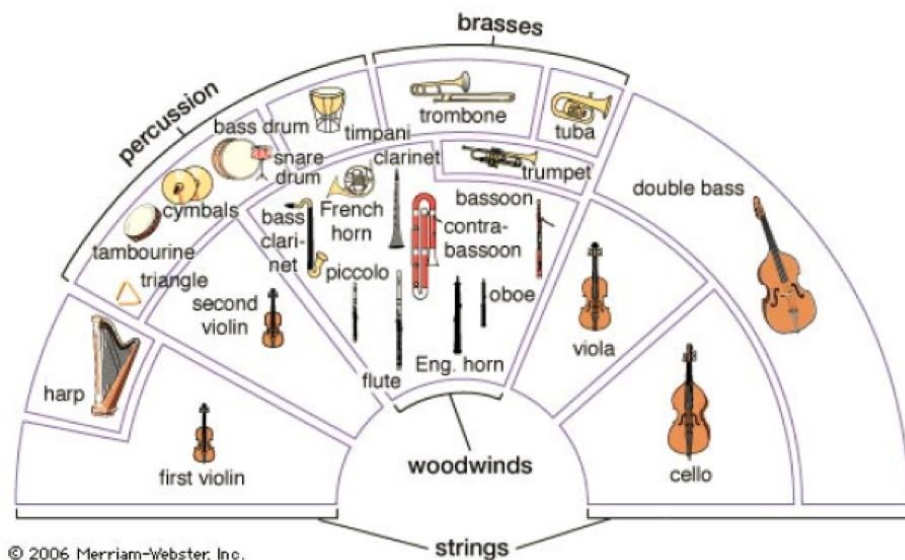


草台班子
松散团队



交响乐队
高素质团队

不同的团队模式



交响乐（计划驱动）

人数和乐器多、门类全
各司其职、分工明确
位置固定、演奏期间没有直接交流
演奏都靠谱，同时看指挥的
按计划演出，很少即兴发挥



爵士乐（敏捷）

人数和乐器较少
没有谱子、没有指挥
强调个性化的表达、鼓励即兴发挥
强有力的互动
对变化的内容有创意的回应

详见《构建之法》5.1-5.2节

有凝聚力的软件开发团队

- 能够建立起团队自己的质量标准
- 成员互相学习、互相帮助
- 知识能够共享
- 鼓励重构和持续改进

团队建设与团队协作

- 成员的选择
- 物理工作环境
- 沟通与交流
- 工作评价与激励机制
- 团队协作机制

软件开发协作

- 代码评审与结对编程
- 问题追踪系统
- 开源软件的社会化协作

代码评审 (Code Review)

- 通过阅读代码并根据经验进行代码质量检查
- 发现代码中的错误和可改进的地方
- 同时也是一种相互学习并熟悉项目代码的过程
- 很多时候是代码正式检入的前置条件
- 评审类型
 - ✓ 自我评审 (Self Review)
 - ✓ 同伴评审 (Peer Review)
 - ✓ 团队评审 (Team Review)

代码评审的准备和前提要求

- 代码必须成功编译并使用最严格的编译警告等级
- 开发者对代码已经进行过测试并覆盖了所有正常执行和错误处理分支
- 开发者提供最新的代码以及变更差异比较

代码评审过程

- 开发者控制流程，讲述代码修改的前因后果
- 评审者可打断叙述并提出意见和问题
- 开发者针对意见和问题进行回答，并对发现的问题建立工作任务
- 双方针对评审结果达成一致
 - ✓ 打回返工：发现严重问题，解决之前不能检入
 - ✓ 有条件通过：发现小问题，问题解决或记录后可以检入，无需再次评审
 - ✓ 通过：无需修改，可直接检入版本管理系统

评审者需要进一步考虑的问题

- 这样修改后，有没有别的功能会受到影响？
- 项目中还有其他地方需要类似的修改吗？
- 有没有留下足够的说明，让将来维护代码时不会出现问题？
- 对于这样的修改，有没有别的团队成员需要告知？
- 导致问题的根本原因是什么？以后如何避免类似的情况再次出现？

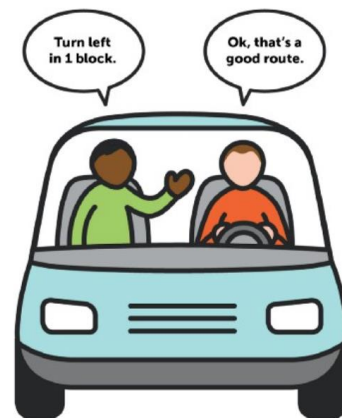
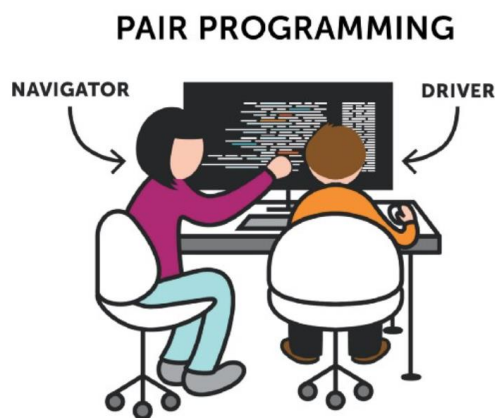
代码评审的检查表 (checklist)

- 代码是否符合需求和规格说明
- 代码设计是否考虑周全
- 代码对于未来的移植和扩展支持如何
- 代码中是否存在无用的部分
- 代码中的错误处理是否完备
- 代码是否符合编码规范和风格
- 代码的可读性与可维护性如何
- 代码的资源使用是否合理、性能如何
- 代码的可测试性如何

详见《构建之法》4.4.3节

结对编程

- 一种敏捷开发实践
- 一对程序员使用一套开发设备一起工作
- 两个角色
 - ✓ 驾驶员 (Driver) 控制键盘输入
 - ✓ 领航员 (Navigator) 发挥领航和提醒作用
- 角色可以互换



不间断的代码评审

- 任何一段代码都被两个人思考过
- 强化了代码归属团队所有的意识
- 工作方式和思考透明化、公开化
- 迫使开发者频繁交流、相互学习
- 互相督促，提高开发效率和质量
- 经常互换角色，保持高效工作

结对编程的好处和坏处

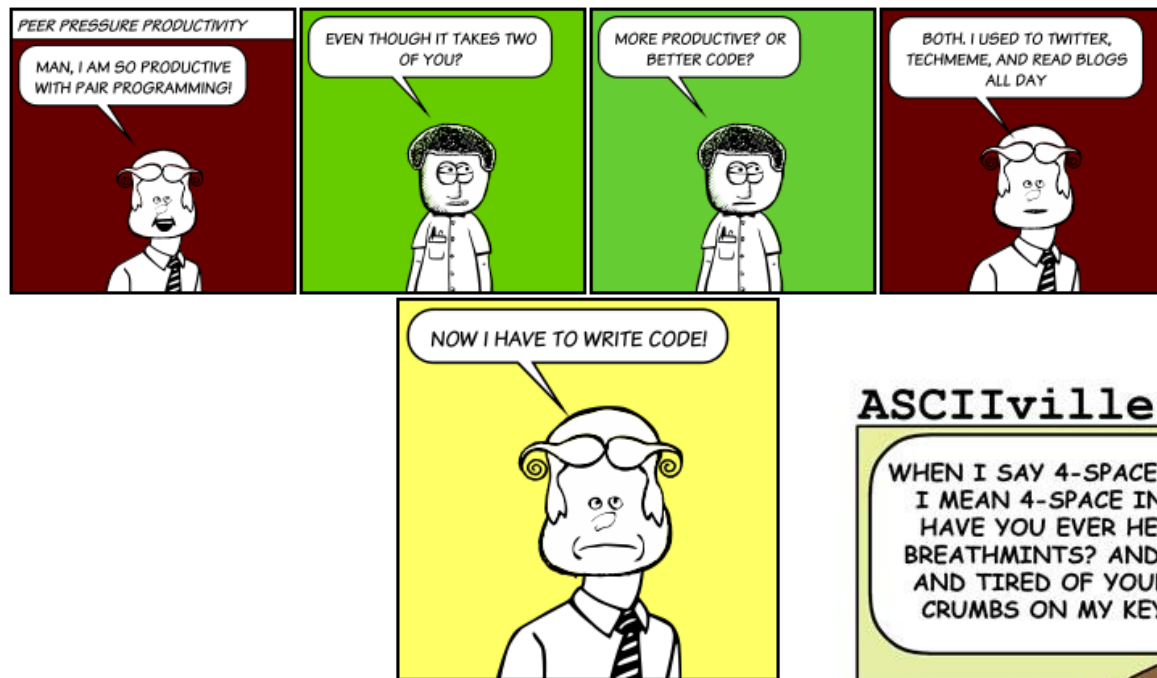
• 好处

- ✓ 提高设计和代码质量
- ✓ 相互激励，相互学习，提高水平
- ✓ 带来更多的信心，提高满足感
- ✓ 有效的交流和知识分享，能更好地应对人员变化

• 坏处

- ✓ 不同的人工作方式不同，可能带来冲突
- ✓ 大量时间可能花在对同伴的培训上
- ✓ 让人感觉到威胁
- ✓ 对个人情绪和自尊的不利影响

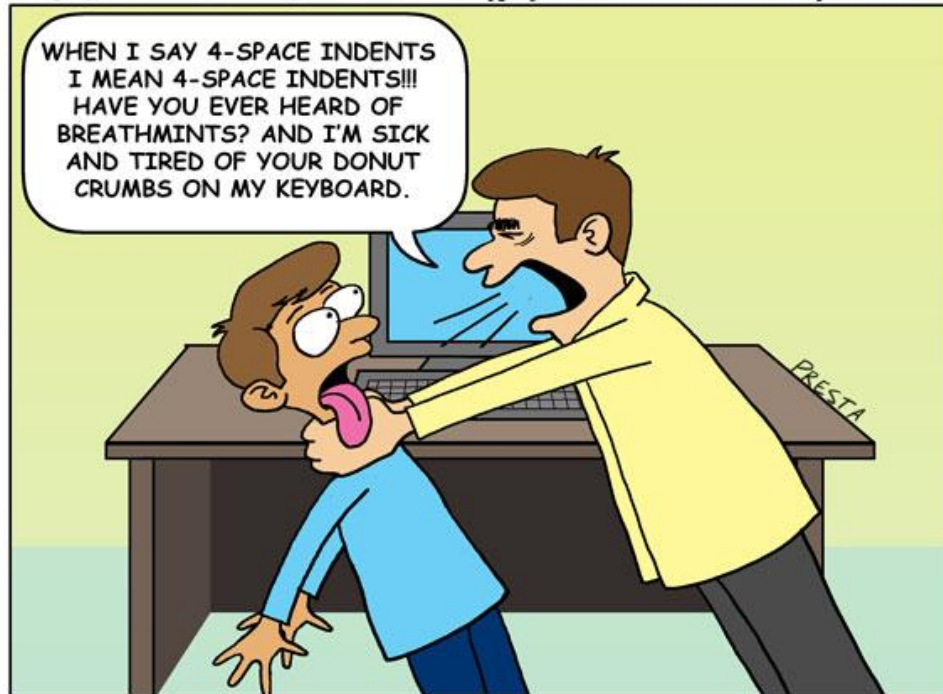
结对编程的好处和坏处



toonlet.com/creator/dalmaer

ASCIIVille

<http://www.asciiville.com>
Copyright 2008. Todd Presta. All rights reserved.



The dark side of pair programming.

IT历史上著名的“结对”协作

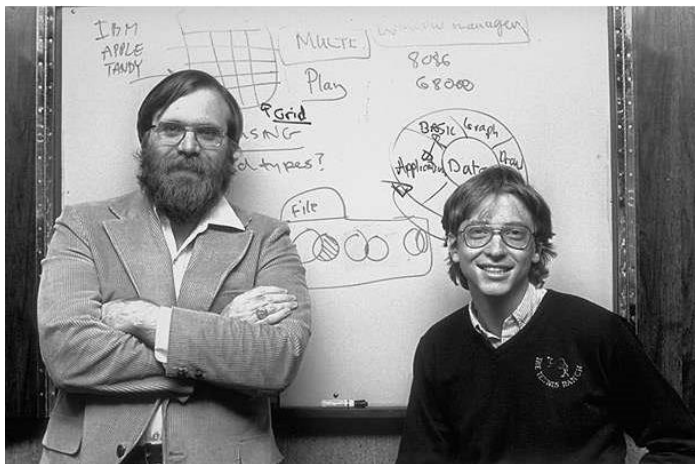


HP (Hewlett, Packard)



Steve Wozniak and Steve Jobs

Apple (Steve Jobs, Steve Wozniak)



Microsoft (Bill Gates, Paul Allen)



Google (Sergei Brin, Lawrence Page)

问题 (Issue)

- 起源于客户服务中反映用户问题的工单 (ticket)
- 现已泛指软件开发中的各种问题
- 面向客户、用户和开发者等不同群体
- 常见的问题类型：Bug、计划中的开发任务、需求变更、评审记录等
- 目的是报告待解决的问题并记录整个处理过程

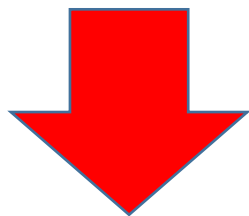
问题追踪系统 (Issue Tracking System)

- 管理软件开发问题生命周期，例如提出、讨论、分配、评价、关闭等
- 常用的问题追踪系统
 - ✓ JIRA (商业软件)
<https://www.atlassian.com/software/jira>
 - ✓ Bugzilla (开源软件)
<http://www.bugzilla.org>
 - ✓ Trac (开源软件)
<https://trac.edgewall.org>

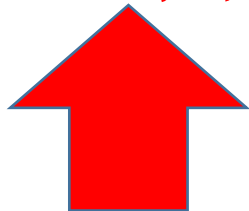
开发流程与issue的发现



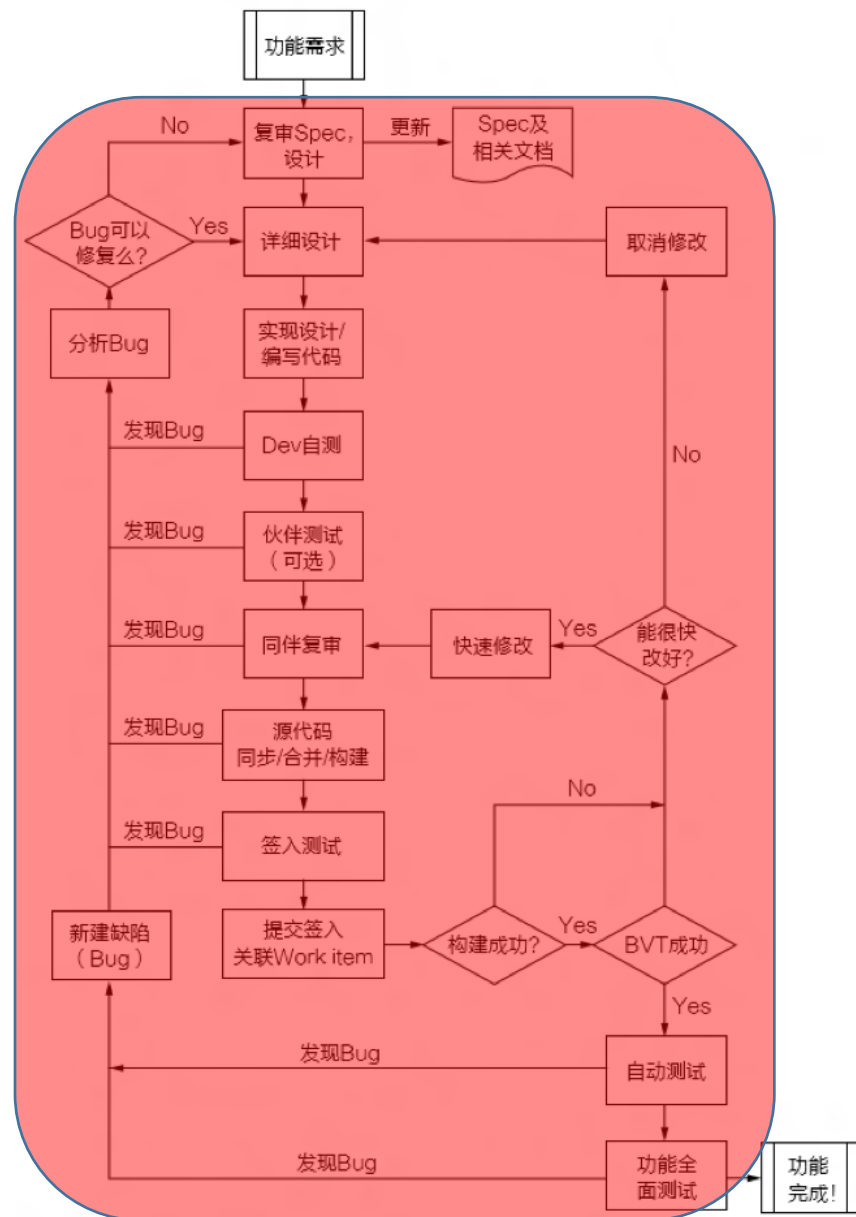
大众反馈（开源用户等）



Bugzilla
问题追踪系统



客户问题反馈



开源项目issue讨论示例-1



o0lwj0o commented on 10 Mar



Hi,

I have found some continuous calls about `StringBuilder.append(..)` in some files. For example, there are eleven continuous calls in the function `toString` from the file `influxdb-java/src/main/java/org/influxdb/dto/BatchPoints.java`.

```
public String toString() {  
    StringBuilder builder = new StringBuilder();  
    builder.append("BatchPoints [database=");  
    builder.append(this.database);  
    builder.append(", retentionPolicy=");  
    builder.append(this.retentionPolicy);  
    builder.append(", consistency=");  
    builder.append(this.consistency);  
    builder.append(", tags=");  
    builder.append(this.tags);  
    builder.append(", points=");  
    builder.append(this.points);  
    builder.append("]");  
    return builder.toString();  
}
```

If it was achieved like `StringBuilder.append(..).append(..).append(..)`, it will promote its performance.



majst01 commented on 10 Mar

Contributor



I'm not sure if this brings any performance benefits, can you try with a test what's the performance gain ?

开源用户提出一个关于代码性能问题的issue

开源项目issue讨论示例-2



fmachado commented on 10 Mar

Contributor



Disclaimer: I'm not a JVM specialist and take this as a naive test as I did just for fun.

My first thought when I read this message was "nah, JIT will eventually optimize the 'hot' code and inline the multi-line calls because *that* BatchPoints.toString() is very short" ([this](#) seems to be a reliable answer on how the hotspot decides how and when to optimize).

Well, I was wrong: :)

With [JMH](#), I got these numbers:

```
# Run complete. Total time: 00:00:12
```

Benchmark	Mode	Cnt	Score	Error	Units
MyBenchmark.testAppendChained	ss	100	0.011	± 0.001	s/op
MyBenchmark.testAppendMultipleLines	ss	100	0.027	± 0.003	s/op
MyBenchmark.testSimpleStringConcat	ss	100	0.011	± 0.001	s/op

on a 8-core machine

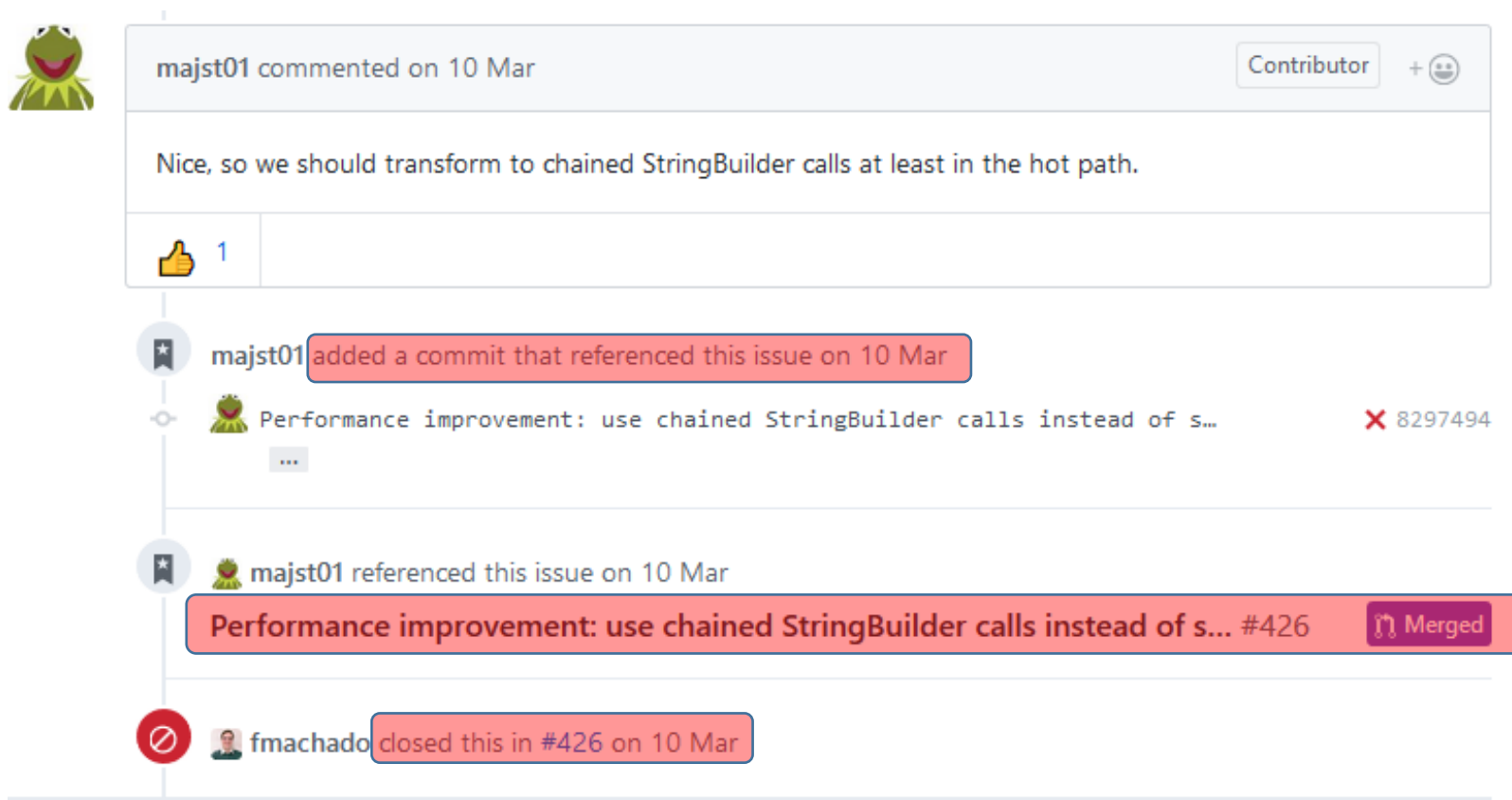
```
$ cat /proc/cpuinfo | grep "model name"
model name      : Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz
...
```

running

```
$ java -version
java version "1.8.0_111"
Java(TM) SE Runtime Environment (build 1.8.0_111-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.111-b14, mixed mode)
```

一个项目开发人员对问题进行了验证和评估

开源项目issue讨论示例-3



The screenshot displays a GitHub issue thread. At the top, a comment by user **majst01** (profile picture of a green frog) is dated 10 Mar. The comment text is "Nice, so we should transform to chained StringBuilder calls at least in the hot path." and has 1 thumbs-up. Below the comment, a timeline shows three events: 1. **majst01** added a commit that referenced this issue on 10 Mar. 2. **majst01** referenced this issue on 10 Mar, with a link to a pull request titled "Performance improvement: use chained StringBuilder calls instead of s..." (numbered #426) which is marked as "Merged". 3. **fmachado** closed this issue in #426 on 10 Mar.

开发人员接受了这个问题，进行了代码修改和合并，
最后关闭了这个issue

问题追踪系统的主要功能

- 问题（Issue）及处理流程管理
 - ✓ 指定 Issue 的优先级
 - ✓ 指定 Issue 所在的阶段
 - ✓ 分配负责 Issue 的处理人员
 - ✓ 制定日程
 - ✓ 监控进度，提供统计
- 问题交流与沟通
 - ✓ 问题评论与讨论
 - ✓ 邮件通知
- 代码管理
 - ✓ 问题与代码提交（commit）相关联
 - ✓ 问题解决后的补丁（patch）管理

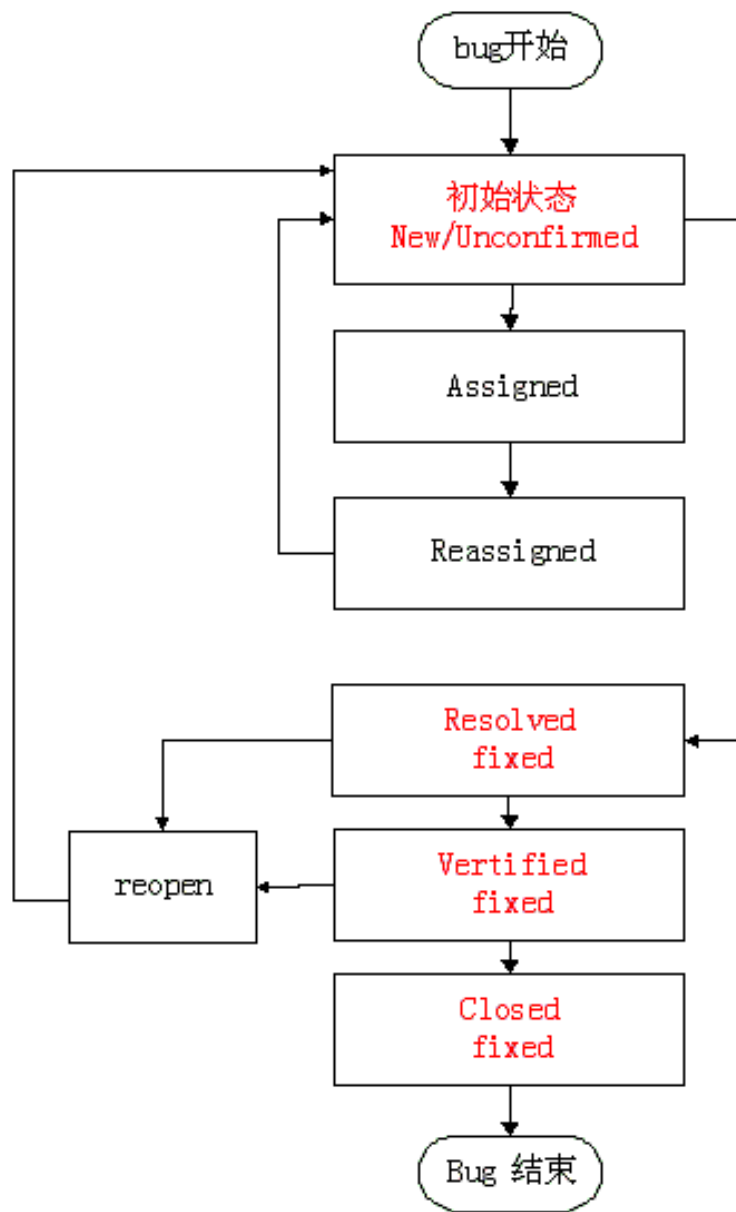
Bug/Issue的生命周期和状态

状态分类 (Status)

待确认的 (Unconfirmed)
新提交的 (New)
已分配的 (Assigned)
问题未解决的 (Reopened)
待返测的 (Resolved)
待归档的 (Verified)
已归档的 (Closed)

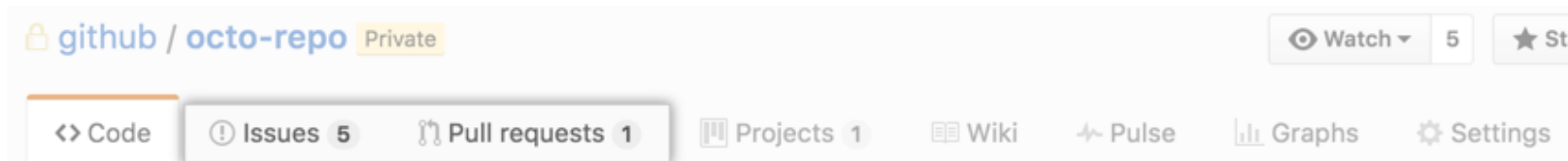
处理意见 (Resolution)

已修改的 (Fixed)
不是问题 (Invalid)
无法修改 (Wontfix)
以后版本解决 (Later)
保留 (Remind)
重复 (Duplicate)
无法重现 (Worksforme)

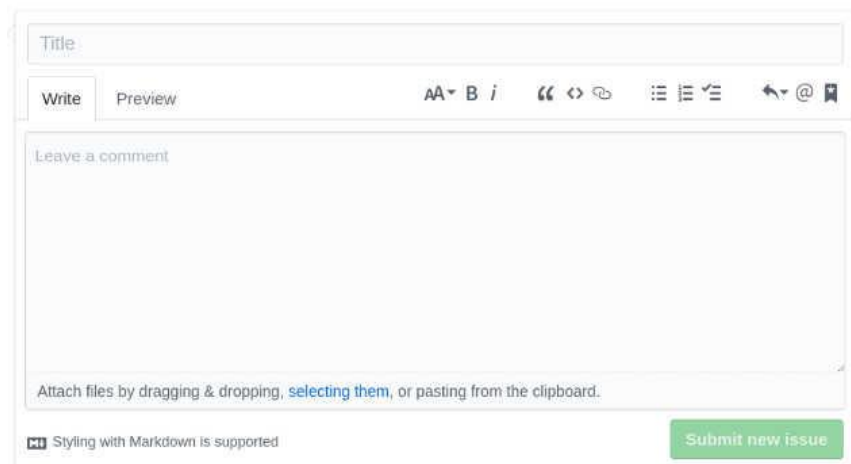


GitHub上的Issue管理

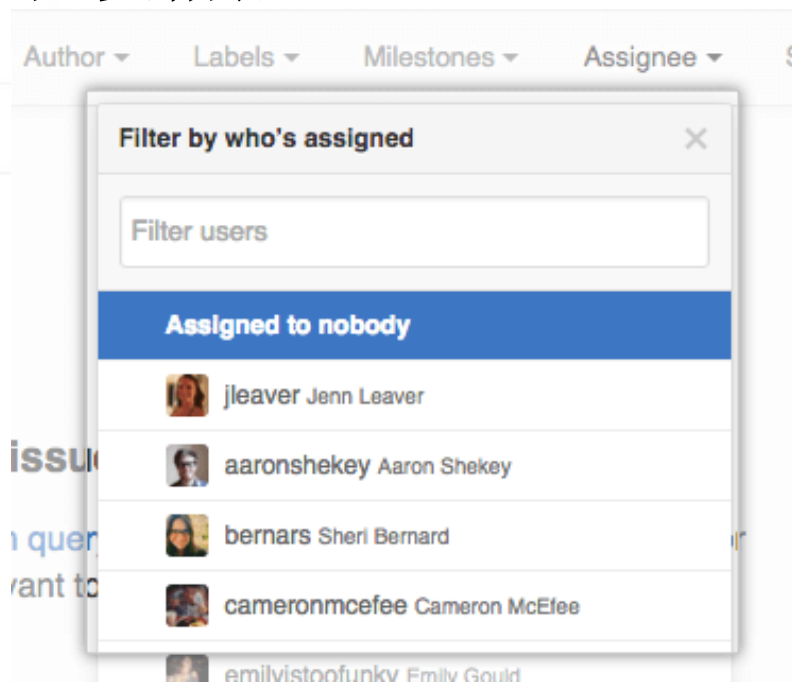
GitHub仓库上的Issue面板



新建Issue







人员指派







GitHub上的Issue管理



Issue标签

 **Open modal is shifting body content to the left** js css confirmed
Opened by [mat0r](#) 3 months ago  62 comments

 **Navbar issues** js css confirmed
Opened by [Nugrata](#) a month ago  36 comments

 **Add support of extra styling class on collapse event** js feature css
Opened by [ziogaschr](#) 22 days ago  24 comments

 **Redundant responsive utility styles** css
Opened by [AlexYursha](#) a month ago  24 comments

 **Select tag not properly styled on stock android browser** css confirmed
Opened by [ADmiad](#) a month ago  19 comments

Issue讨论交流


Please review the [guidelines for contributing](#) to this repository.




This is an example of a new issue

Write

Preview

 [Parsed as Markdown](#)

 [Edit in fullscreen](#)

Text fields on GitHub are parsed with Markdown. This means we can @mention people like @githubstudent or @octocat to reference them into the issue. You can also cross-reference other issues and pull requests with the number of those issues and pull requests like #14020

You can also create tasks list to monitor the progress of smaller tasks as well:

- [] #23
- [] #142
- [] #140

Attach images by dragging & dropping, [selecting them](#), or pasting from the clipboard.

Commit与Issue关联

Fixed #9196 - malformed HTML in doc

[Browse code](#)

Stray <h3> was being closed by an </h2>. Updated to valid HTML. Fixes #9196

 3.0.0-wip



bwhitty authored a day ago

1 parent [f816a18](#)


commit [a4638259a5f7358436ab24ef68e3589e30f18f0b](#)

Bugzilla

Bug 305134 - Remove FeedView from Firefox 1.5 - Mozilla Firefox 1.0+ (Build 2005082813) - Gecko 1.8b4

Fichier Edition Affichage Aller à Marque-pages Outils ?

https://bugzilla.mozilla.org/show_bug.cgi?id=305134

 **Bugzilla**
Bugzilla Version 2.19.1+

Bugzilla Bug 305134 **Description : Remove FeedView from Firefox 1.5** Last modified: 2005-08-28 01:41 PDT
[Search page](#) [Enter new bug](#)

Bug#: 305134 **alias:** **Hardware:** All
Product: Firefox **OS:** All
Component: RSS Discovery and Preview **Version:** unspecified
Status: RESOLVED **Priority:** -
Resolution: FIXED **Severity:** normal
Assigned To: Ben Goodger (use ben at mozilla dot org for email) [<bugs@bengoodger.com>](mailto:bugs@bengoodger.com) **Target Milestone:** -
QA Contact: nobody@mozilla.org
URL:
Summary: Remove FeedView from Firefox 1.5
Status Whiteboard:
Keywords: fixed1.8

Reporter: [Ben Goodger \(use ben dot org for email\)](#) [<bugs@bengoodger.com>](mailto:bugs@bengoodger.com)
Add CC:
CC: alex@spamcop.net
axel@pike.org
bugs.mano@sent.com
bugtrap@psychoticwol
bugzilla@dougweb.org
☐ Remove selected C

Flags: (Help!)
mtschrep: blocking1.8b4rc
bugs: blocking1.8b4
blocking1.9a1
blocking-aviary1.0.7
blocking-aviary2.0
testcase

Attachment	Type	Created	Size	Flags	Actions
------------	------	---------	------	-------	---------

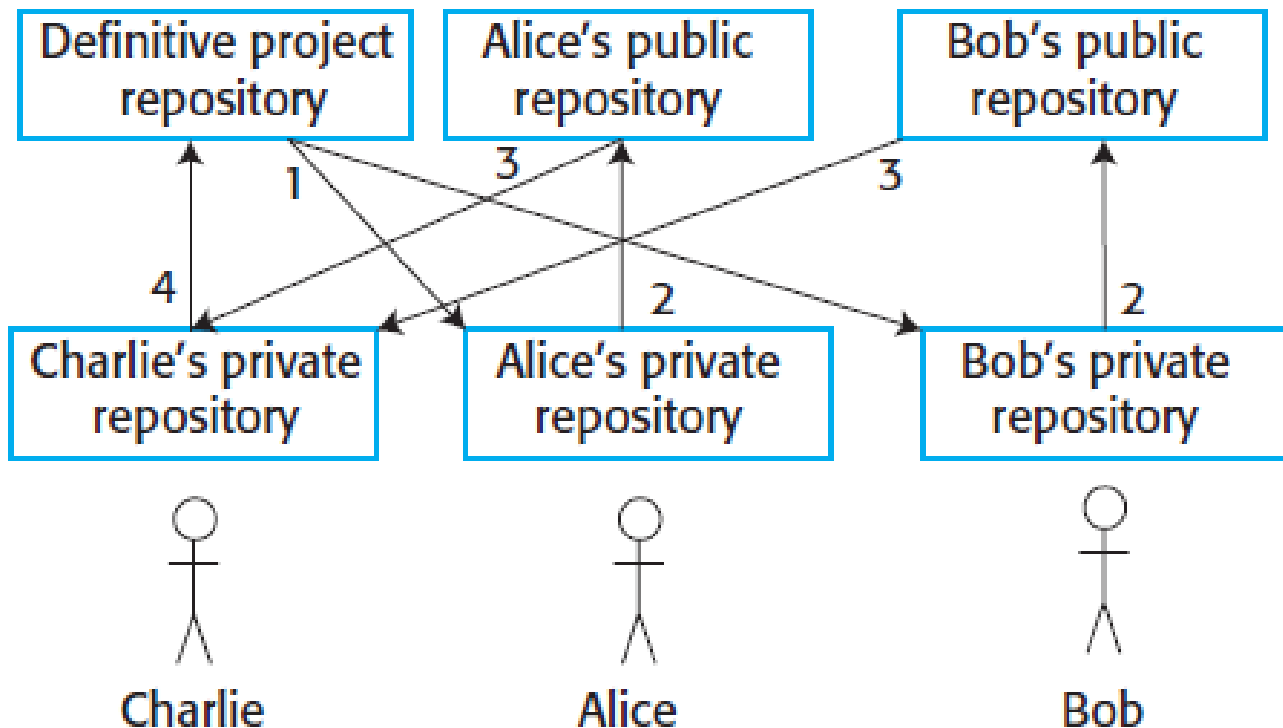
Terminé bugzilla.mozilla.org AdBlock

开源软件的社会化协作 (Pull Request)

- Github等开源社区中流行的开发模式
- 开发者在本地对项目源代码进行修改后请求项目合并
- 开发者可以在社区中进行评论交流
 - ✓ 修复了一个bug, 可以合并一下吗?
 - ✓ 试着开发了一个新功能, 可以合并一下吗?
- 其他开发者可以根据修改描述和代码差异等进一步进行评论和交流
- 项目管理者通过评审和测试确认后决定是否接受合并请求

Pull Request开发模式

1. 开发参与者克隆项目库
2. 开发参与者推送本地变更到个人公共库
3. 集成管理员将变更拉到本地库中进行测试
4. 集成管理员确定接受修改后更新最终项目库



集成管理者 开发参与者 开发参与者

阅读建议

- 《软件工程》 22.3
- 《构建之法》 4.4-4.6、
5.1-5.2
- 《代码大全》 第21章

快速阅读后整理问题
在QQ群中提出并讨论

CS2001

软件工程

End

7. 软件过程与团队合作
— 团队与协作