**代码审查之Upsource**

1.代码审查概述 3

1.1为什么需要Code Review 3

1.2 Code Review的前提 3

1.3Code Review需要做的事情 3

1.3.1完整性检查 3

1.3.2一致性检 3

1.3.3正确性检查 4

1.3.4可修改性检查 4

1.3.5可预测性检查 4

1.3.6健壮性检查 4

1.3.7结构性检查 4

1.3.8可追溯性检查 4

1.3.9可理解性检查 4

1.3.10可验证性检查 5

2.Upsource简介 5

3.Upsource的特点 5

4.Upsource作为代码审查工具的优劣 6

4.1Upsource优势 6

4.2Upsource的不足 6

5.Upsource部署 6

5.1下载方式 6

5.2安转配置 7

5.2.1 windows下安装 7

5.3配置/启动/停止软件操作 7

5.3.1配置，指定监听端口 7

5.3.2启动服务 7

5.3.3停止服务 8

6.Upsource首次登录配置（首次注册是管理员账号） 8

7.Upsource管理员使用说明 10

7.1创建项目 10

7.1.1创建项目的作用 10

7.1.2创建项目步骤 10

7.2编辑项目 11

7.3删除项目 11

7.4重置初始化失败的项目 11

7.5设置用户，组，角色和权限 11

7.5.1设置用户 11

7.5.2设置组 12

7.6设置邮件通知功能 13

7.7恢复数据库 14

7.8服务器迁移 14

8.Upsource用户使用说明 15

8.1浏览存储库 15

8.1.1查看存储库中的更改历史 15

8.1.2在一个项目导航 15

8.2比较修正 18

8.2.1使用内联diff使用修订视图 18

8.2.2使用并排的diff视图 19

8.2.3比较修改和分支 20

9.简单分析代码审查流程 21

9.1 Upsource 的提交特性 21

9.2 代码审查的角色 21

9.3自定义工作流程 22

9.4请求代码评审 22

9.5 Create a code review 22

9.6 Posting comments 22

9.6.1post a review commet 22

9.6.2 post a line commet 23

9.7 Discussion options 24

9.8 Reviewing branches 25

9.9 创建问题跟踪器 26

9.10 Upsource analytics 27

10.如何更好的应用于代码审查 30

10.1. 避免经常性Code Review 30

10.2. Code Review不要太正式，而且要短 30

10.3. 尽可能的让不同的人Reivew你的代码 30

10.4. 保持积极的正面的态度 31

# 1.代码审查概述

## 1.1为什么需要Code Review

这里只有几个原因:

1.你对代码已经做出了改变，现在你希望你的队友检查它，并帮助你确保新代码清晰，可读，不包含任何严重的缺陷。

2.你在你的队友的改变中发现了一个潜在的问题，你想要快速的告诉他们。

3.当一个重要的改变发生时，你想让你的一些队友更新你的项目状态。

4.新队友更快地了解项目的不同部分。

5.促进集体代码所有权。

## 1.2 Code Review的前提

1.Code Review人员需要理解何为Code Review，使用Code Review能做什么。使用的人都不能很好的理解，将造成效率更低下，应付Code Review。

2.工程代码需要正确的编译通过。Code Review针对的不是编译报错的代码。

3.代码运行时需要功能正常使用。Code Review针对的是优化代码，减少BUG，而不是经测试不能正常使用设备或（针对嵌入式开发）的新代码。

4.Review人员需要理解代码。Review人员应对代码实现的功能熟悉，才能更好的针对性分析代码。

## 1.3Code Review需要做的事情

主要检查代码中以下存在的问题：代码的一致性，编码风格，代码的安全问题，代码冗余，是否正确设计满足需求。

### 1.3.1完整性检查

1.代码是否完全实现了设计文档中提出的功能需求

2.代码是否按照设计文档进行了集成和Debug

3.代码是否已创建了需要的数据库，包括正确的初始化数据

4.代码是否存在任何没有定义或没有引用的变量，常数或数据类型

### 1.3.2一致性检

1.代码的逻辑是否符合设计文档

2.代码红使用的格式，符号，结构等风格是否保持一致

### 1.3.3正确性检查

1.代码是否符合指定的标准

2.所有的变量都被正确定义和使用

3.所有的注释都是准确的

4.所有的程序都调用了正确的参数个数

### 1.3.4可修改性检查

1.代码涉及到的常量是否易于修改

2.代码是否包含了交叉说明，以描述程序是如何对常量和常量访问的

3.代码是否只有一个出口和一个入口

### 1.3.5可预测性检查

1.代码使用的开发语言是否具有定义良好的语法和语义

2.是否避免了依赖于开发语言缺省提供的功能

3.代码是否无意陷入死循环

4.代码是否避免了无穷递归

### 1.3.6健壮性检查

1.代码是否采取措施避免运行时错误（数据边界溢出，除数为0，值越界，堆栈溢出等）

### 1.3.7结构性检查

1.程序每个功能是否都作为一个可辨识的代码块存在

2.循环是否只有一个入口

### 1.3.8可追溯性检查

1.代码是否对每个程序进行唯一标识

2.代码是否包括一个修订历史记录，记录中对代码的修改和原因都有记录

3.是否所有的安全功能都有标识

### 1.3.9可理解性检查

1.注释是都足够清晰的描述每个子程序

2.是否使用到不明确或不必要的复杂代码，它们是都被清楚的注释

3.使用一些统一的格式化技巧（缩进/空白等），增强代码的清晰度

4.是否在定义命名规则时采用便于记忆，反映类型等方法

5.代码中的算法是否符合开发文档中描述的数学模型

### 1.3.10可验证性检查

1.代码中的实现技术是否便于测试

# 2.Upsource简介

Upsource是一个代码审查工具，支持基于web UI来进行代码审查，讨论和代码跟踪，省去在本地机器或者通过开会进行代码审查和讨论的各种非必要开销，提高工作效率。

因为其稳定性和功能齐全，Upsource是非常流程的代码审查工具。

# 3.Upsource的特点

3.1 通过web浏览器来进行代码审查

3.2 支持在windows，linux，mac搭建，搭建方便

3.3支持多种开发语言

3.4 支持SVN，GIT，Mercurial等代码版本控制工具

3.5支持浏览器有：Firefox，Chrome，Safari，Edge，Opera

**3.6只支持在代码提交后进行代码审查**

**3.7设置review人员，开发人员，项目组长，系统管理员的权限。**

# 4.Upsource作为代码审查工具的优劣

## 4.1Upsource优势

1.快速导入现有的svn，git等代码工程进行代码审查（通过创建一个工程来维护跟踪对应的代码工程）

2.多个代码审查工程并行进行

3.支持管理员手动增加用户或发送邀请邮件，让用户自行注册

4.支持有review消息，评论代码，有回复等情况时浏览器发消息通知，或者发送邮件通知review人员

5.选中一个代码审查工程后，可选择目录树的形式展开源码，方便查看

6.选择代码版本，进行diff对比修改处

7.私聊@对应的用户

8.分析对应的代码审查工程（每一份源码工程）的审查记录，各开发人员的提交记录个数等

9.查看修改历史，搜索文件

10.动态修改代码审查工程的配置，方便增加属性

11.实时接受提交，动态显示修改，动态通知项目组员的reviewer人员

## 4.2Upsource的不足

1.不支持先代码评审再通过commit提交代码

2.用户可复制源码（暂时还没发现不可以复制代码审查工具上的代码的工具）

# 5.Upsource部署

## 5.1下载方式

1.windows，linux，mac方式选择，直接在官网下载（注意：10人版是免费的，超过10人是按人数收费，可一年买断永久，每年的版本升级用户自己决定是否花钱升级，不升级不影响使用）

2.一键下载即可下载整个相关工具包（前端UI，VCS分析，监视，默认绑定数据库Hub（用于身份验证，权限设置，组管理等服务））

## 5.2安转配置

### 5.2.1 windows下安装

1.禁用反病毒软件，windows防御系统和windows搜索服务（非必须，本人安装忽略这一步正常安装）

#### 5.2.2 linux下安装

未防止资源限制不足导致错误，修改 /etc/security/limits.conf文件，修改如下：

1.memlock unlimited

2.nofile 100000

3.nproc 32768

4.as unlimited

#### 5.2.3其他设置

1.反向代理服务器设置，可选择Nginx， apache http， IIS， Custom等服务器

2.下载设置Hub，可选默认自带，也可另外下载单独的，一般使用 选用自带即可

## 5.3配置/启动/停止软件操作

### 5.3.1配置，指定监听端口

For Windows:

<upsource\_home>\bin\upsource.bat configure –listen-port **11111(自行设定)**

For Linux or Mac OS:

<upsource\_home>/bin/upsource.sh configure –listen-port **11111(自行设定)**

### 5.3.2启动服务

For Windows:

<upsource\_home>\bin\upsource.bat start

For Linux or Mac OS:

<upsource\_home>/bin/upsource.sh start

### 5.3.3停止服务

For Windows:

<upsource\_home>\bin\upsource.bat stop

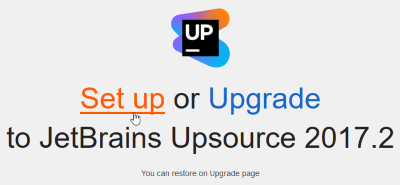
For Linux or Mac OS:

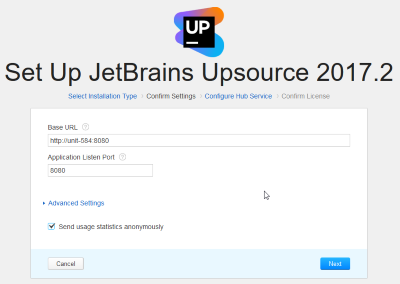
<upsource\_home>/bin/upsource.sh stop

**注意注意：启动软件后，第一个登录注册的用户是管理员账号**

# 6.Upsource首次登录配置（首次注册是管理员账号）

启动软件服务后，启动过程中的log日志保存在 <upsource\_home>\logs\upsource-stdout.log

第一次可以进行特定的设置



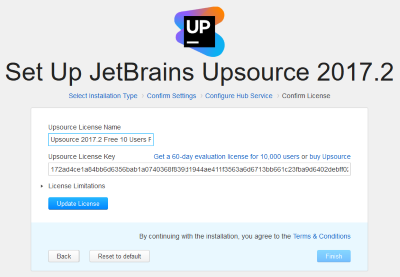
部署机器的ip地址

监听端口

选择Hub，一般使用选择[built-in Hub](https://www.jetbrains.com/help/upsource/hub-user-and-project-management.html)即可。

+

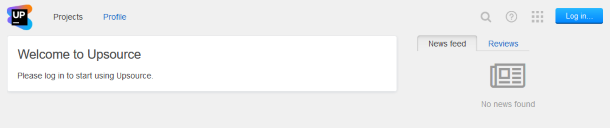
使用10人免费版，直接默认即可。超过10人用需付费，需输入购买的 license



等待若干分钟……



设置启动完成，进入此主界面



# 7.Upsource管理员使用说明

## 7.1创建项目

### 7.1.1创建项目的作用

一个Upsource项目对应于VCS存储库。创建一个新项目使您能够:

1.在VCS存储库中可视化提交历史

2.浏览项目中任何修订的代码

3.查看每个修订的更改

4.创建关于代码和全面的代码审查的讨论

### 7.1.2创建项目步骤

1.使用管理员帐户登录到Upsource（非管理员用户帐户没有资格创建新项目），点击创建一个项目。

2.General / Code intelligence / Integration / Advanced选择一种项目模板

3.以 General 项目创建为例子：

Project name: 项目名称（自行设定）

Project ID: （自动根据Project name设定，不用管）

Code review ID pattern: （一般默认）

VCS: Git / Subversion / Perforce / Mervurial

Repository URL: 源码的URL

4.Branches:

+:trunk

+:branches/\*

5.Check interval:反映Upsource检查仓库中新修订的频率(默认情况下每300秒一次)。

（如果提交到您的存储库并不是经常这样做，请考虑将检查间隔设置为更大的值）

6.Users: 创建用户组:让Upsource为您创建以下用户组:

1.代码的浏览者

2.开发者

3.项目管理员

或稍后自己创建它们。组是用户的集合，其中每个用户具有相同的权限或角色(权限集)

5.点击 Create Project

至此，一个项目创建完成。

## 7.2编辑项目

1.点击屏幕右上方的设置图标设置。

2.管理视图将打开。

3.单击项目名称旁边的Edit Project按钮。显示的编辑项目表单包含创建项目时输入的设置。

4.编辑设置并单击Save changes。

## 7.3删除项目

1.点击屏幕右上方的设置图标设置。

2.管理视图将打开。

3.找到你要删除的项目，然后点击它旁边的“删除项目”按钮。

## 7.4重置初始化失败的项目

1.点击屏幕右上方的设置图标设置。管理视图将打开。

2.一个失败的项目可以通过一个错误消息和旁边显示的重置项目按钮来识别:

3.点击重置项目。

## 7.5设置用户，组，角色和权限

只有管理员账号才有权限增删用户和设置用户的权限。Upsource利用Hub——该服务为所有JetBrains服务和连接到它的应用程序提供了一个单一入口点。对于Hub，用户只需要一个用户帐户来访问所有已连接的服务，管理员就可以从一个位置处理这些服务和用户帐户

### 7.5.1设置用户

两种方式创建用户：1.管理员手动增加用户，并填写对应的信息。2.生成并发送邀请邮件，让用户自己填写注册.

1.管理员手动增加用户账号

1.在管理菜单的访问管理部分中，选择Users用户

2.点击创建用户Create Users按钮

3.在侧栏中，单击Create Users按钮

4.输入下列设置的值:

设置描述：显示名称设置新用户的全名

电子邮件：输入新用户的电子邮件地址。电子邮件地址必须有效

密码：输入新用户帐户的密码

确认新用户帐户的密码

5.点击Create 按钮:

创建指定的设置的用户帐户

新用户的配置文件页面将打开。在这里，可以为用户帐户配置访问权限

2.生成并发送邀请邮件

1.在管理菜单的访问管理部分中，选择Users用户

2.点击Create Users按钮

3.在侧栏中，单击Invite Users按钮

4.在Emails输入框中，输入您想要邀请到中心注册帐户的用户的空间分隔的电子邮件地址列表

5.点击“Invite”按钮

邀请被发送到每个用户的电子邮件地址被邀请。邀请包含登录页面的链接，用户可以在其中设置密码和访问中心

### 7.5.2设置组

设置不同的组，方便管理一个项目，针对不同的组可以拥有不同的项目，拥有不同的权限

#### 7.5.2.1增加组

1.选择帐户配置文件的Groups选项卡

2.单击Add Group按钮

3.从下拉列表中选择所需的组

4.添加用户进组，设置组名，组权限

#### 7.5.2.2配置组权限

访问一个组：

1.点击屏幕右上方的设置图标设置。管理视图将打开

2.点击用户。Hub的用户视图将显示在一个新窗口中

3.在这个新窗口中，单击settings图标设置并从下拉菜单中选择Groups。Groups表单将显示出来

4.从列表中选择组并单击Access选项卡

5.点击Grant Access然后从显示的菜单中选择角色或权限和一个项目

6.通过向组分配一个包含指定权限的角色来实现对组的权限

访问用户:

1.点击屏幕右上方的设置图标设置。管理视图将打开

2.点击用户。用户视图将显示出来

3.从列表中选择用户并单击Access选项卡

4.点击Grant Access然后从显示的菜单中选择角色或权限和一个项目

5.通过将用户包含到具有指定访问权限的组中，可以实现对用户的许可

#### 7.5.3设置角色和权限

角色是一组权限。角色被分配给每个项目的用户或组。用户帐户根据成员的成员身份继承角色，但也可以有直接分配的角色：

1.System admin

2.Project admin

3.Developer

4.Observer

5.Code viewer

查看、编辑或创建角色:

1.点击屏幕右上方的设置图标设置。管理视图将打开

2.Click Roles。角色视图将显示出来

3.单击一个角色，查看它包含的权限(在权限选项卡中)

4.可以通过添加或删除权限来定制角色

## 7.6设置邮件通知功能

Upsource的用户很可能希望被告知在Upsource中发生的不同事件，比如创建和关闭评论、分配审阅人员和创建新评论

配置Upsource以发送电子邮件通知:

1.点击屏幕右上方的设置图标设置。管理视图将打开

2.单击Set up email notifications。您将被重定向到settings

3.按照说明填写表单并配置SMTP连接

每个用户还可以定制他想要通知的事件类型，使用他的个人通知设置，或者完全禁用电子邮件通知

允许用户直接通过电子邮件添加评论，不用在再登录浏览器评论和回复评论

配置Upsource以接收电子邮件:

1.点击屏幕右上方的设置图标设置。管理视图将打开

2.点击设置“Add comment by email添加评论邮件”

3.在添加评论的电子邮件页面，指定您的专用电子邮件地址及其设置，并勾选启用“添加评论邮件”复选框Enable "Add comment by email"

4.点击保存更改Save changes

## 7.7恢复数据库

Upsource允许您轻松备份和恢复所有内容和配置数据，以防止在系统故障或升级到更新的构建时出现数据丢失.

创建一个备份:

1.作为管理员登录到Upsource

2.点击屏幕右上方的设置图标设置。管理视图将打开

3.点击备份链接Backup now link

4.单击OK创建备份

安排自动备份:

1.在Administration视图中，单击设置备份计划Set up backup schedule。

2.指定备份的时间间隔，并单击Save

## 7.8服务器迁移

1.停止Upsource:< upsource\_home > \ bin \ Upsource.bat

2.将**整个Upsource主目录**移动到新位置。

3.在新服务器上运行以下命令:

< upsource\_home > \ bin \ upsource.bat configure --listen -port 1111 --base -url http://servername:1111/

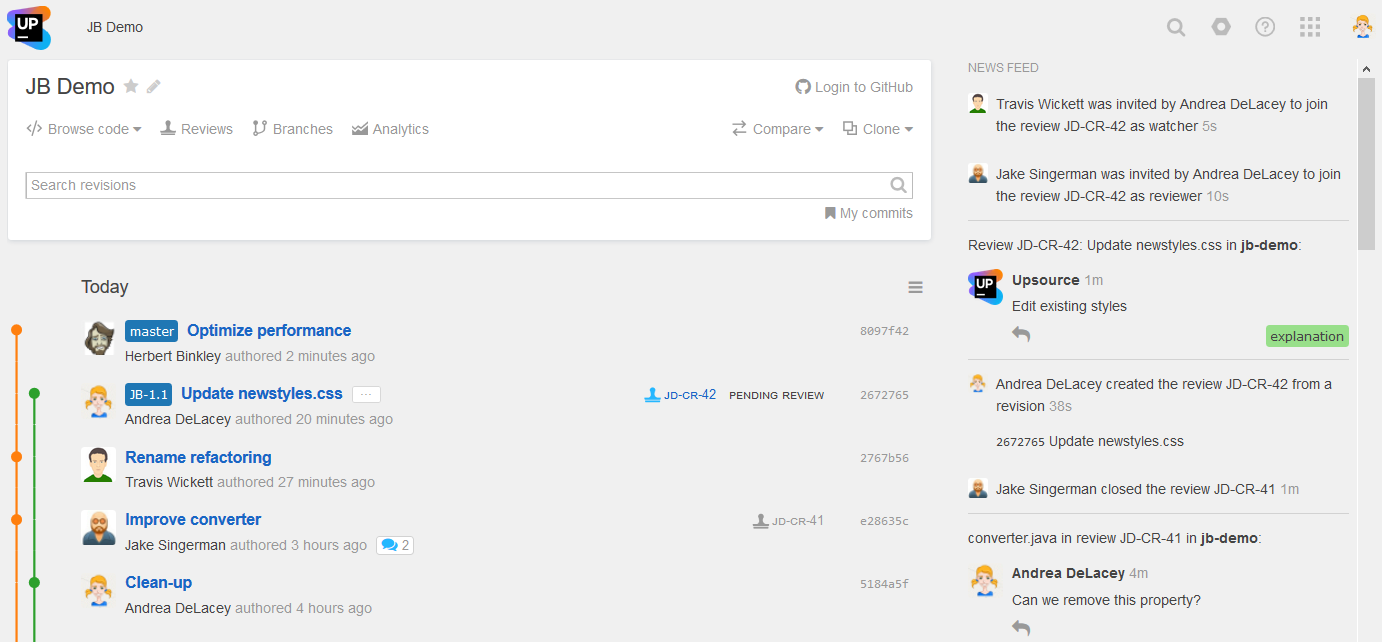
其中1111是要使用的端口号，servername是新服务器的名称

4.启动新服务器 Upsource:< upsource\_home > \ bin \ Upsource.bat

# 8.Upsource用户使用说明

## 8.1浏览存储库

### 8.1.1查看存储库中的更改历史



提交者的用户名

分支的名字

提交的信息

一个修订ID

提交时间戳(当commit发生时)

一个review图标(CR\_review open和closed\_review close)和ID链接到评审一个讨论计数器discussion counter，它将显示讨论线程的数量

在当前页面可以很好的查看关联项目的代码修改记录和讨论记录。

### 8.1.2在一个项目导航

修改条目可以扩展到列出所有受修改影响的文件——只需点击进入字段:

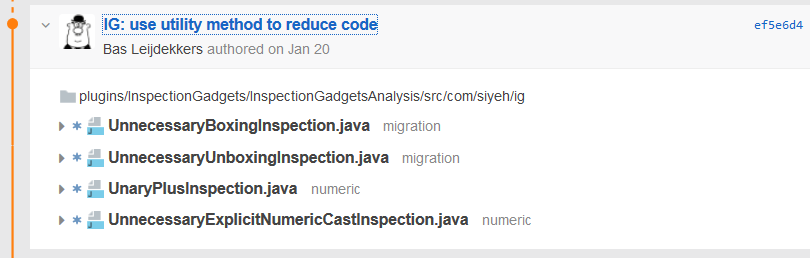
1.modified

2.renamed

3.added

4.removed

5.commented on



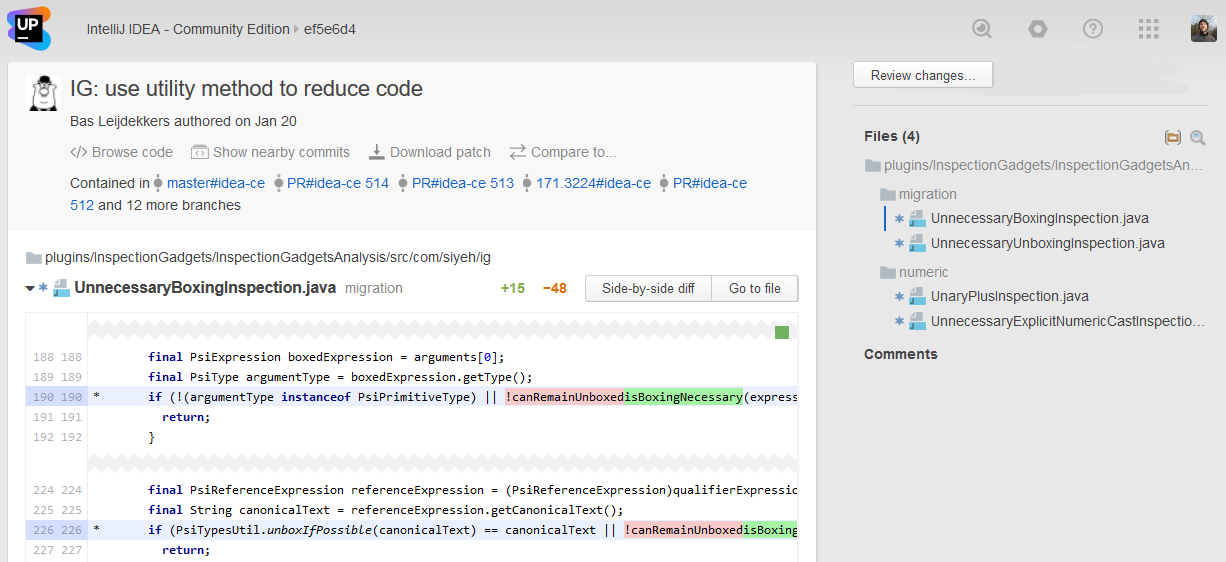
每个列出的文件条目可以依次展开，以显示它们的diff——只需点击文件名称:



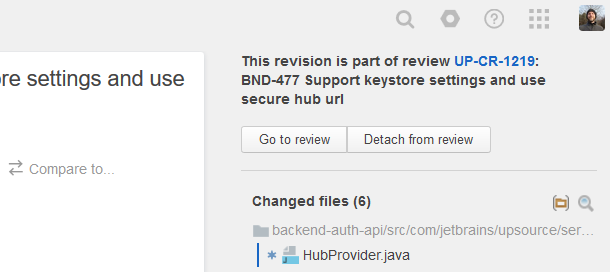
从这个diff视图中，您可以看到一个并行的diff视图或一个文件视图——因为在diff窗格的右上角单击一个相应的按钮:



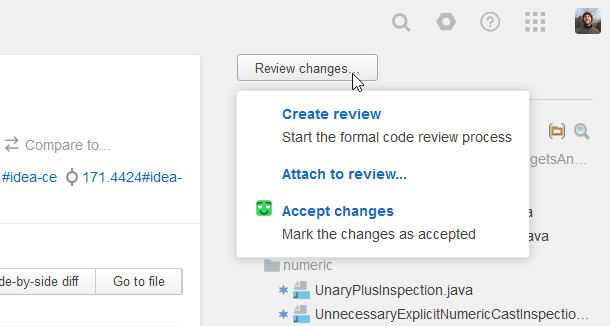
要打开修改，单击它的消息文本或ID。将被重定向到一个修改视图，其中包含修改的所有文件都将显示在其中，每个文件下面都有一个diff:



如果修改版已被审查，你可以去看审查或将修订从它:



如果它还没被审查，可以为它创建一个新的审查:



## 8.2比较修正

### 8.2.1使用内联diff使用修订视图

在修订视图中，可以看到更改文件的列表，以及在修改中显示每个更改文件的修改部分的内联diff视图，所有这些都在一个页面上。

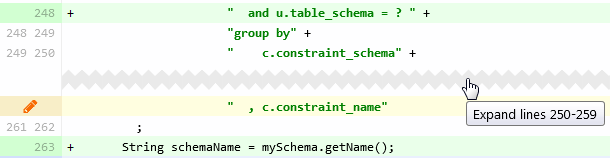
要从项目主页打开修订视图，单击修订ID或提交消息。修改中包含的文件将会在每个文件下面显示一个内联的diff:



在每个diff窗格的顶部，您可以看到添加和删除了多少代码片段:

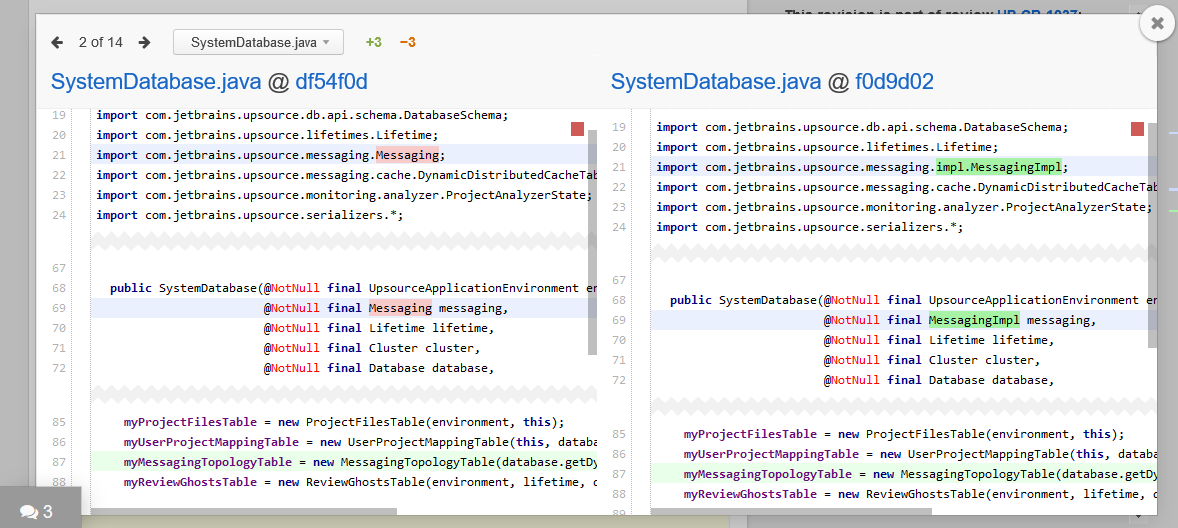


删除的片段用红色突出显示，添加的片段用绿色表示。不变的代码块是默认隐藏的。你可以通过点击剪切线来显示它们:



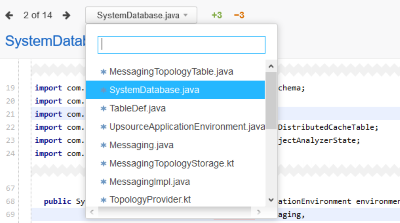
### 8.2.2使用并排的diff视图

并排的diff视图显示了不同版本的文件——旧的和更新的——一起在一个屏幕上。可以查看两个版本的全部内容，并逐行比较不同的内容



删除的片段用红色突出显示，添加的片段用绿色表示。

上面的文件列表允许跳转到变更集中的其他文件，而不会留下这个视图:



如果修订包含注释，可以通过点击左下角的讨论图标来显示它们:



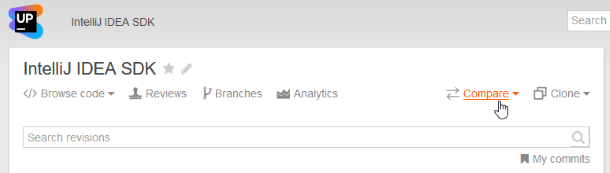
可以通过单击行内diff窗格的右上角的相应按钮，从修订视图中打开并排的diff视图:



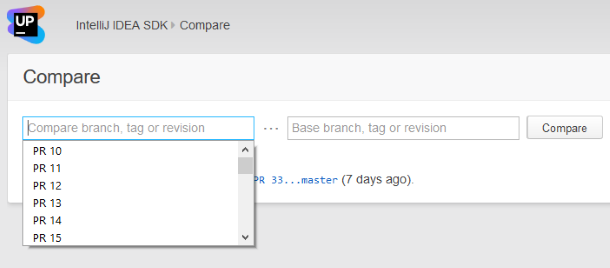
### 8.2.3比较修改和分支

比较页面允许查看指定的修订、分支和标记之间的差异。

1.打开比较页面,单击项目主页顶部的比较:

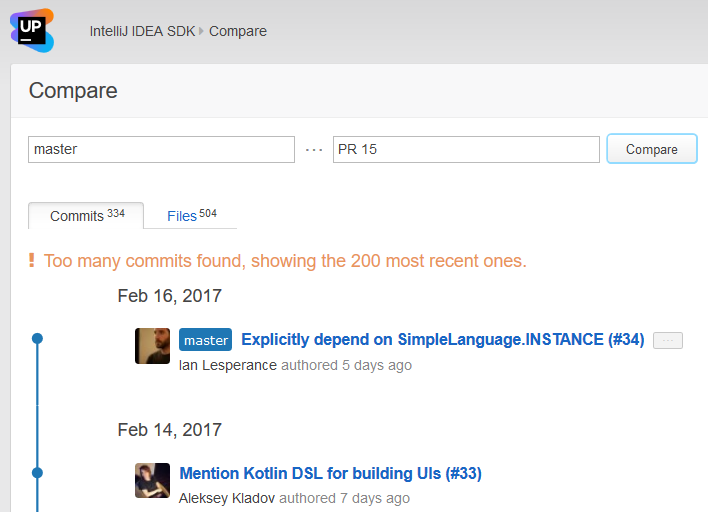


2.输入修改或分支的id，或希望与文本字段进行比较的标记。也可以从下拉列表中选择分支:

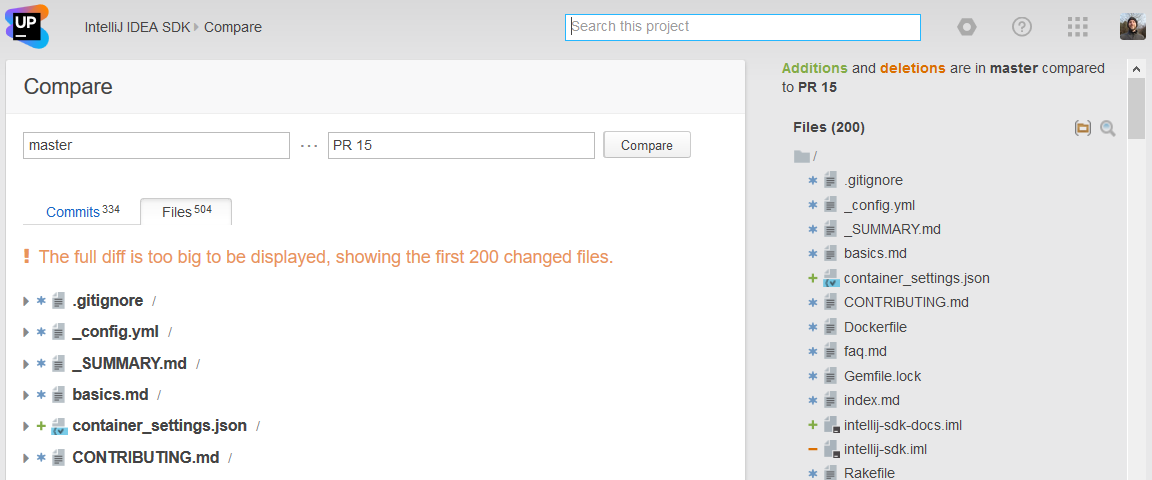


注意:如果指定了一个分支ID或标记，那么将会对指定的分支或标记的顶部修改进行比较。

3.提交选项卡显示在按时间顺序列出的修订之间发生的提交:



4.可以展开或打开每个条目以查看内联的diff，与修订视图相同。文件选项卡列出与这些提交相关的文件:



列表中的文件也标记为：修改、补充、移除。单击文件名以显示该文件的内联diff。

# 9.简单分析代码审查流程

## 9.1 Upsource 的提交特性

Upsource支持提交后的评审，这意味着在开始评审之前，您应该提交对VCS存储库的更改。

**如果公司要求在将代码合并到主分支之前进行审查，可以先提交特性分支，并请求对该分支提交进行审查。**

一旦该提交的代码评审成功通过，再将更改合并到主分支。

## 9.2 代码审查的角色

**Author** ：通常会对别人的修订进行评论，并邀请其他参与者——评论者和观察者来检查和观察他(这个author)所提交的变更。

提交者总是被认为是评审的作者，不管谁实际上创建了评审。Upsource允许任何用户(拥有适当的权限)创建审查。此外，可以在管理员设置的自定义工作流中自动地创建一个评审。

审查可以包括由不同的提交人作出的不同修订。在这种情况下，这些提交者也会成为评论作者。

代码提交者的工作流请求代码评审。

Reviewer：查看和检查作者提交的变更，并留下反馈。他可以讨论具体的代码，或者添加一般性注释;也可以完成审查，接受改变或提高对他们的关注。查看审查队友对审查员工作流程的更改。

**Watcher** ：不参与审查过程。每当有重要的变化时，他就不断更新项目的状态。请参阅由队友介绍的监视更改，以了解观察者的工作流程。

## 9.3自定义工作流程

1.管理员账号登录

2.点击设置图标的settings

3.选中项目点击Edit project

4.点击右边的Custom workflows

可以针对代码审查流程设置自动化的流程，比如创建审查请求，增加自动审查的修改，自动分配评估参与者，关闭审查请求。

## 9.4请求代码评审

1.对svn存储库进行更改。

2.创建一个代码评审。

3.邀请参与者(评论者和可选的观察者)。

4.讨论问题，如果有的话，通过与审稿人交换意见。

5.解决问题，提交修复程序，并将新的修订添加到现有的评审中。(如果公司要求是主干只能是通过代码审查的，请先创建分支，再修改代码，提交代码，通过评审后，最后合并到主干。)

6.一旦评审者准你的修改，你就会关闭审查。

总的来说，开发和代码评审可以是分开进行的，各自对svn存储库的工程代码开发后，提交到svn，Upsource自动关联修改并显示，然后就可以开始代码审查。

## 9.5 Create a code review

1.Upsource并在列表中定位项目:

2.项目名称以打开它。项目主页视图显示按时间顺序列出的修订:

3.滚动列表，或者使用搜索字段找到感兴趣的版本

4.光移动到修订条目——将出现一个Review changes链接:

5.评审更改，然后从弹出菜单中选择Create Review

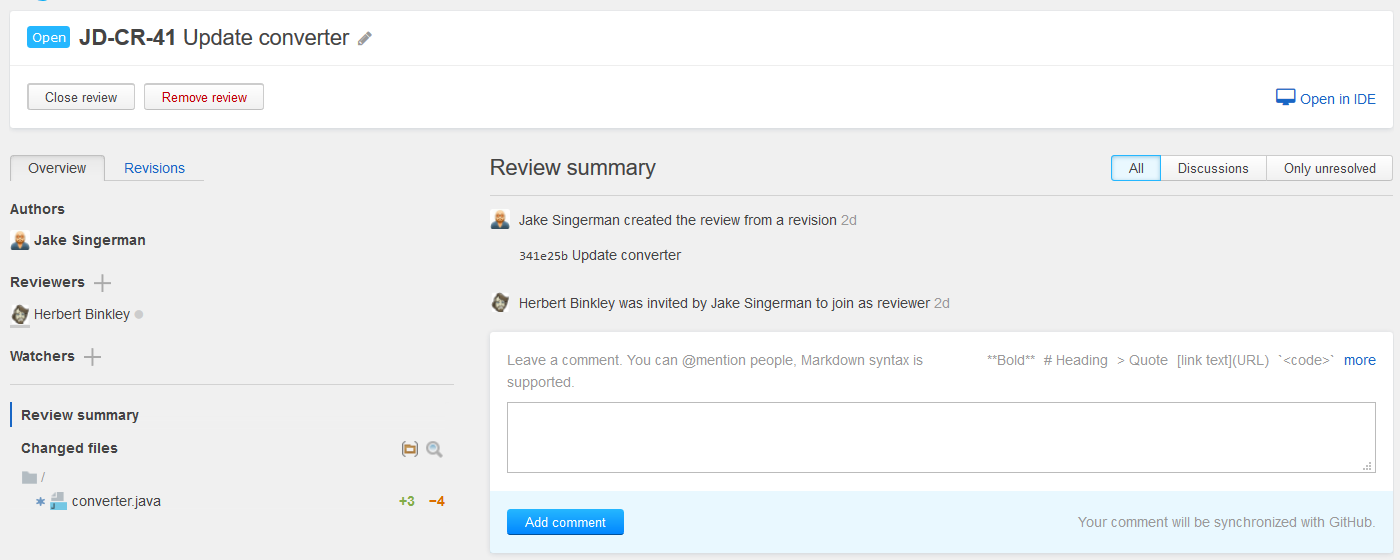
6.新的评审被创建，打开，它的ID显示在左上角(在我们的例子中是jd - cr - 41)。

## 9.6 Posting comments

### 9.6.1post a review commet

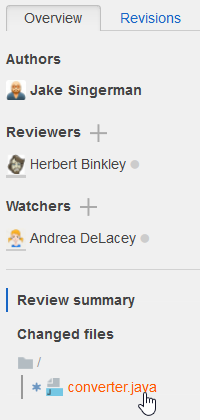
1.review页面上，选择Review summary(这是默认视图)。

2.本框中输入消息，并单击Add comment:

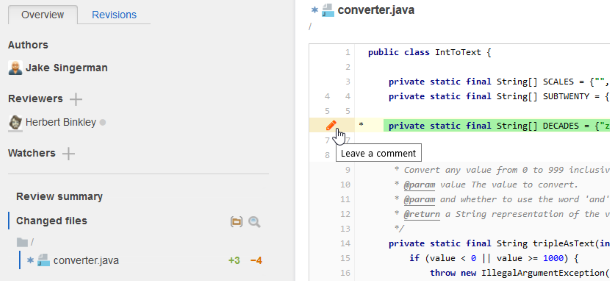


### 9.6.2 post a line commet

1.本框中输入消息，并单击Add comment:



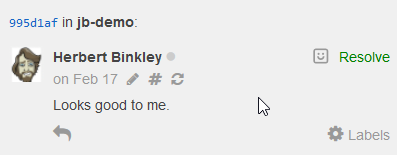
2.左侧的edit图标，选择要评论的行:



3.本框中输入消息，并单击Add comment:



## 9.7 Discussion options



1.编辑你的评论。

2.跟随一个永久链接到讨论。

3.注释与GitHub(GitHub项目)同步。

4.择并添加对评论的反应。

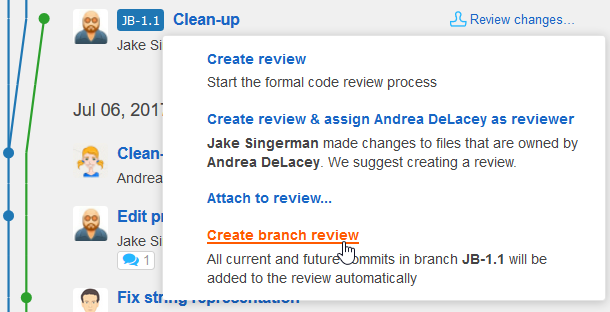
5.以在讨论线程的逻辑结束时解决讨论。

6.回复评论

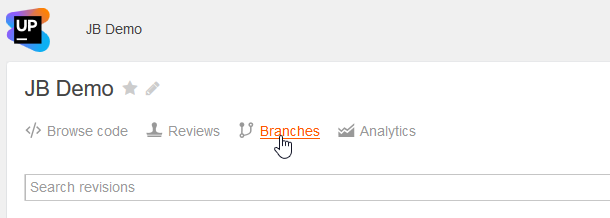
7.论添加一个标签。详情请参阅讨论标签。

## 9.8 Reviewing branches

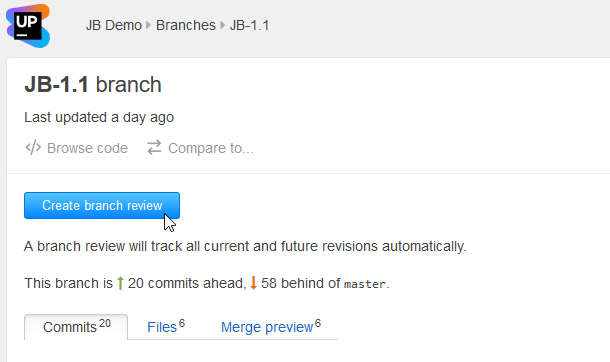
1. 按照项目主页上的图形来定位分支，然后单击评审更改并选择Create branch Review:



2.通过点击链接到分支页面:

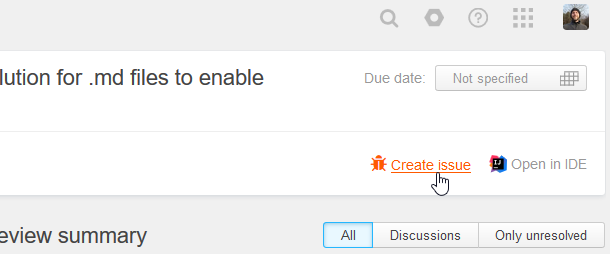


3.点击需要的分支，点击一个蓝色方框内的名字，最后点击创建分支回顾:

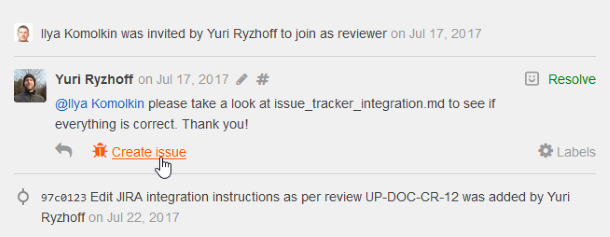


## 9.9 创建问题跟踪器

1.代码评审页面:



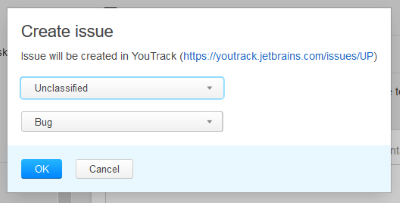
2.任何讨论评论，都是评论的一部分:



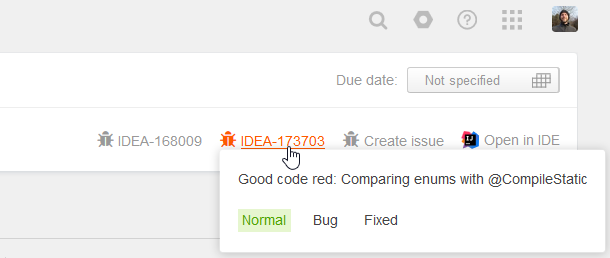
3.修改:



4.单击Create issue后，您将被要求指定问题的强制值:问题类型和优先级:



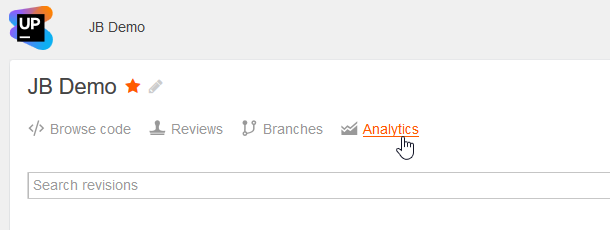
5.从审查中产生的问题，以及其摘要中提到的问题，将在审查页上显示:



## 9.10 Upsource analytics

Upsource收集和分析存储库统计数据，并在易于查看的图形和图表中提供给我们看。

点击项目主页顶部的分析:



1. Commit activity提交活动(随时间而提交活动)

2. Project treemap项目目录树(基于大小和其他标准的项目文件的可视化)

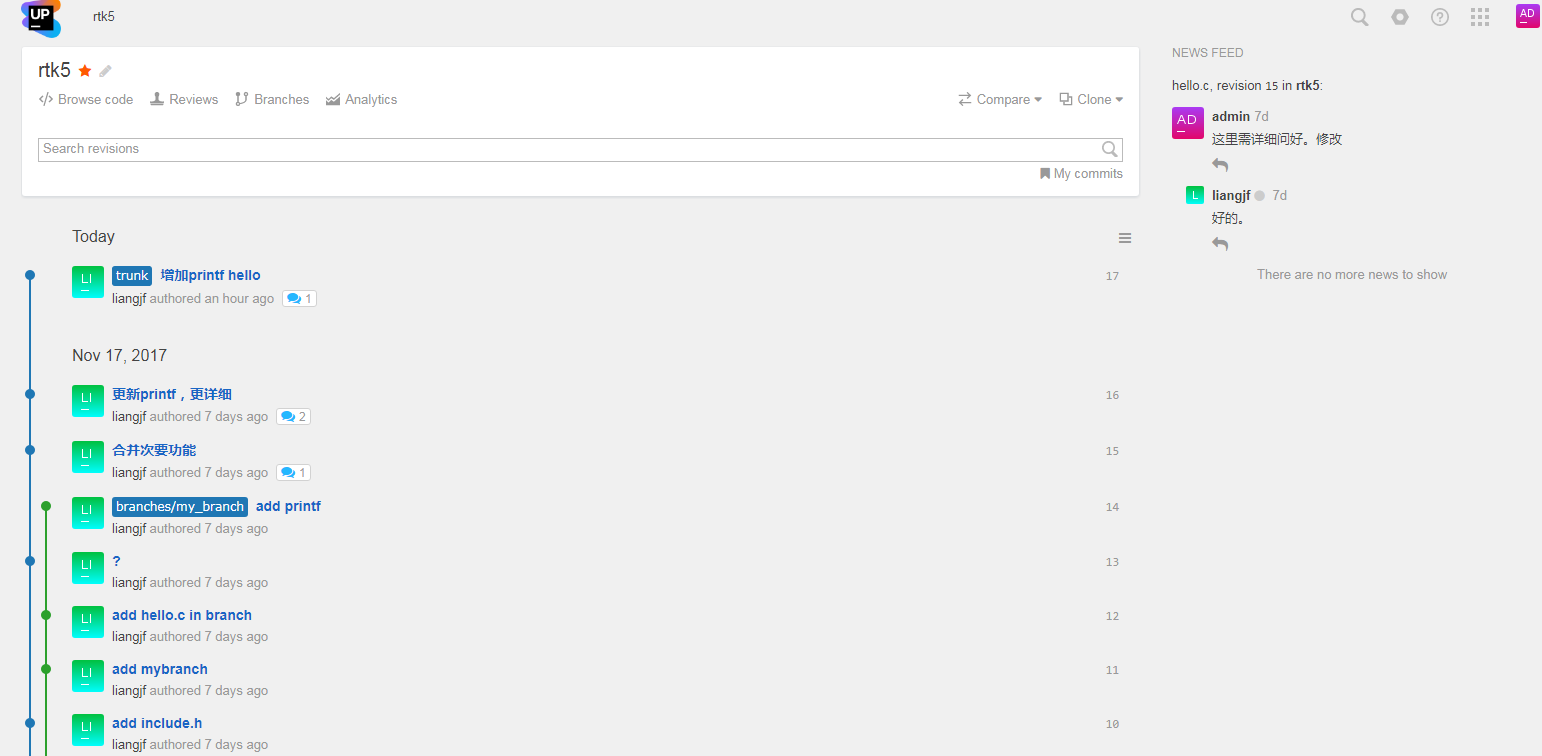
3. File history文件历史(观察项目如何随时间演变)

4. Code review代码审查(审查覆盖率和统计)

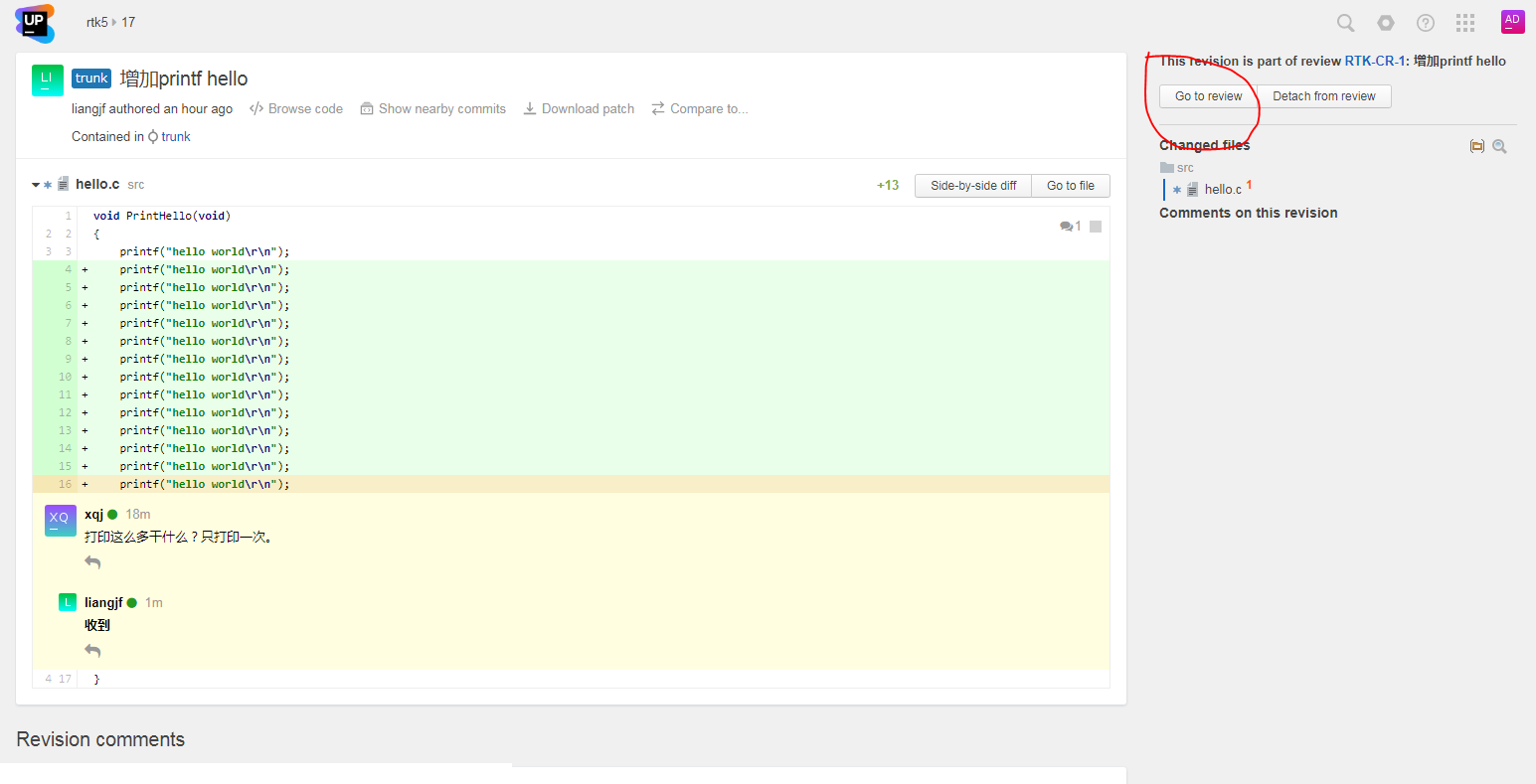
5. Reviewers graph评审人关系图(审查参与者之间的关系可视化)

**下面以一个简单的小项目为例来简单说明整个Code Review的流程**

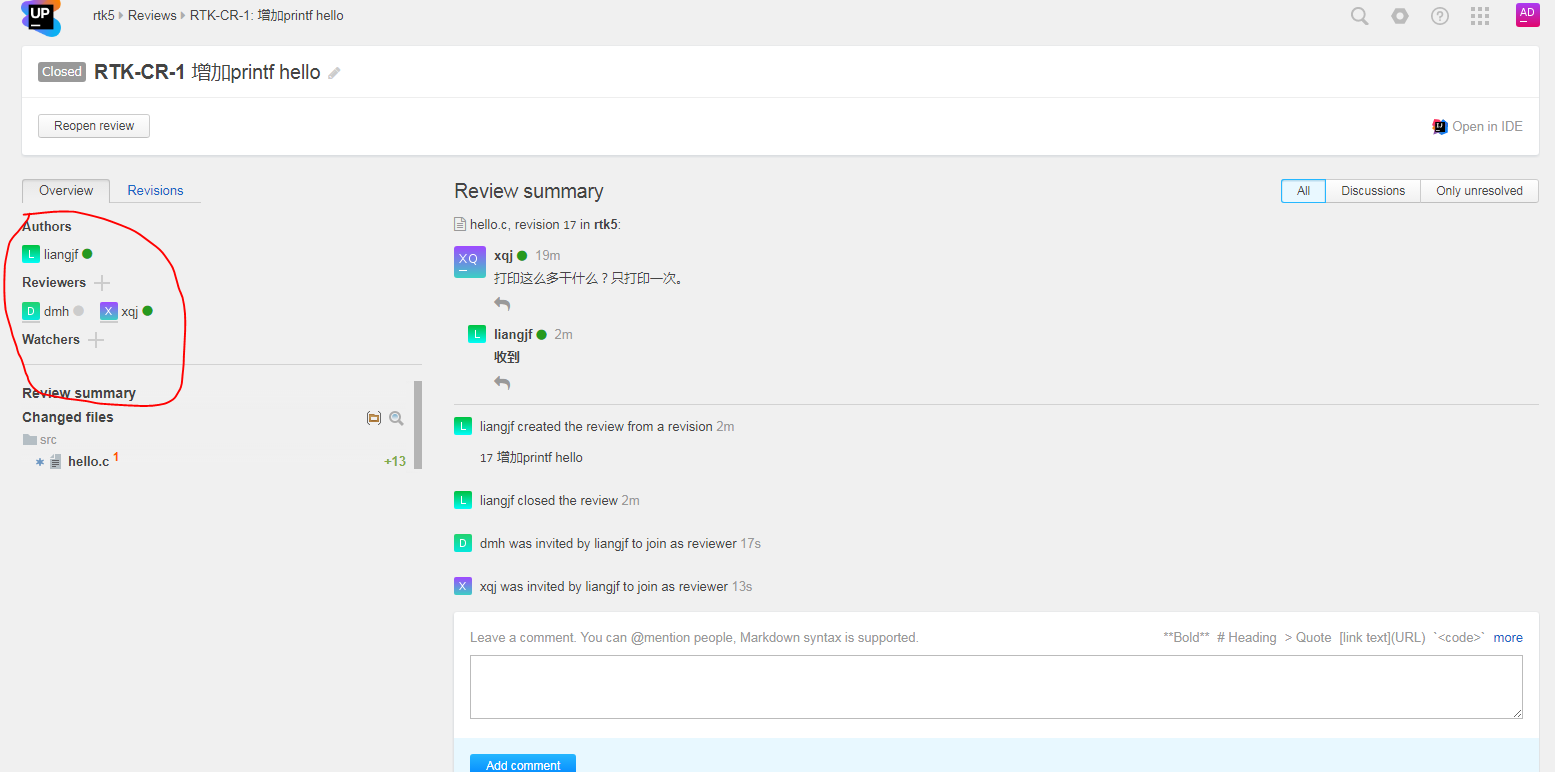
1.项目组某人提交了代码



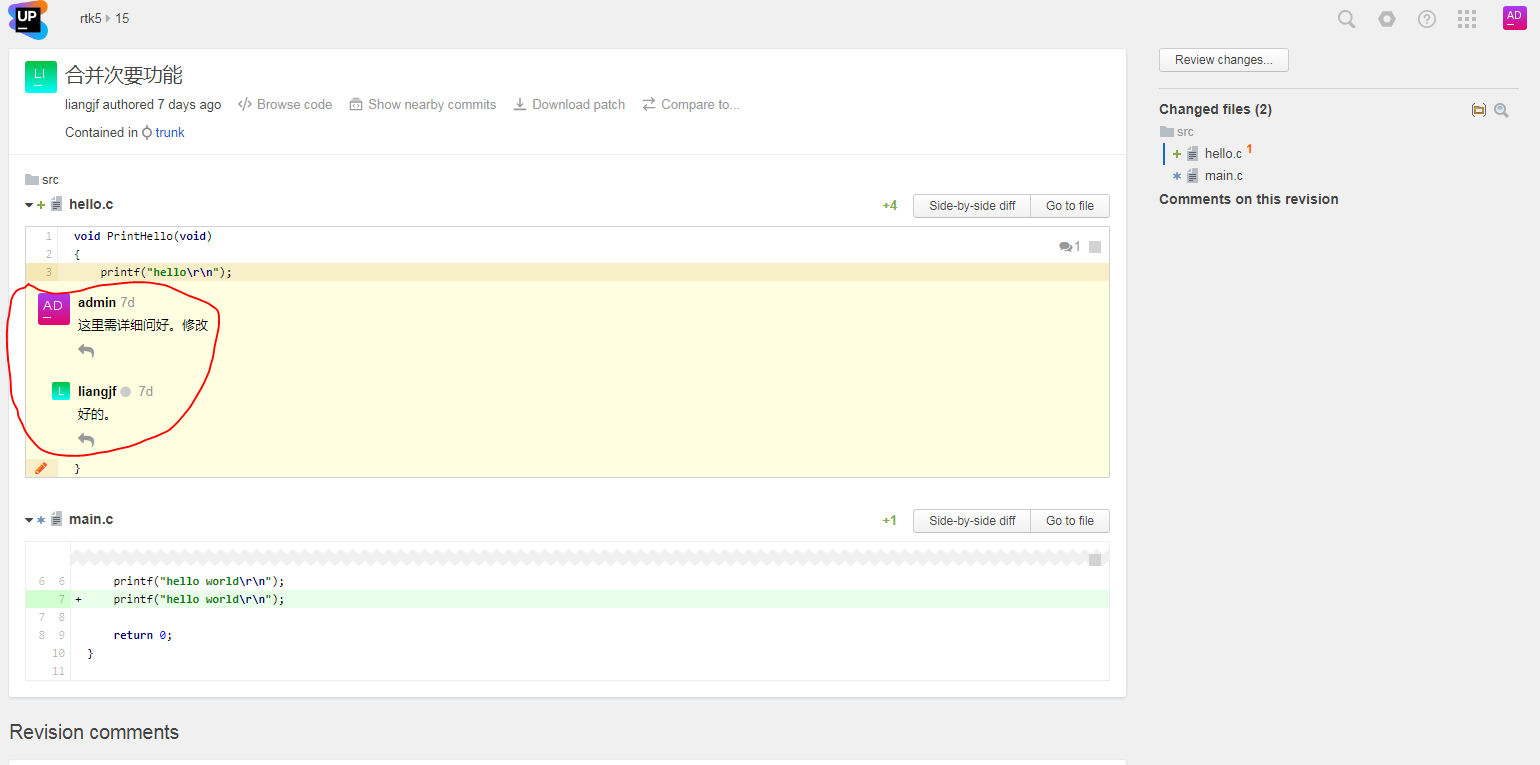
2.代码提交者发起一次新的review



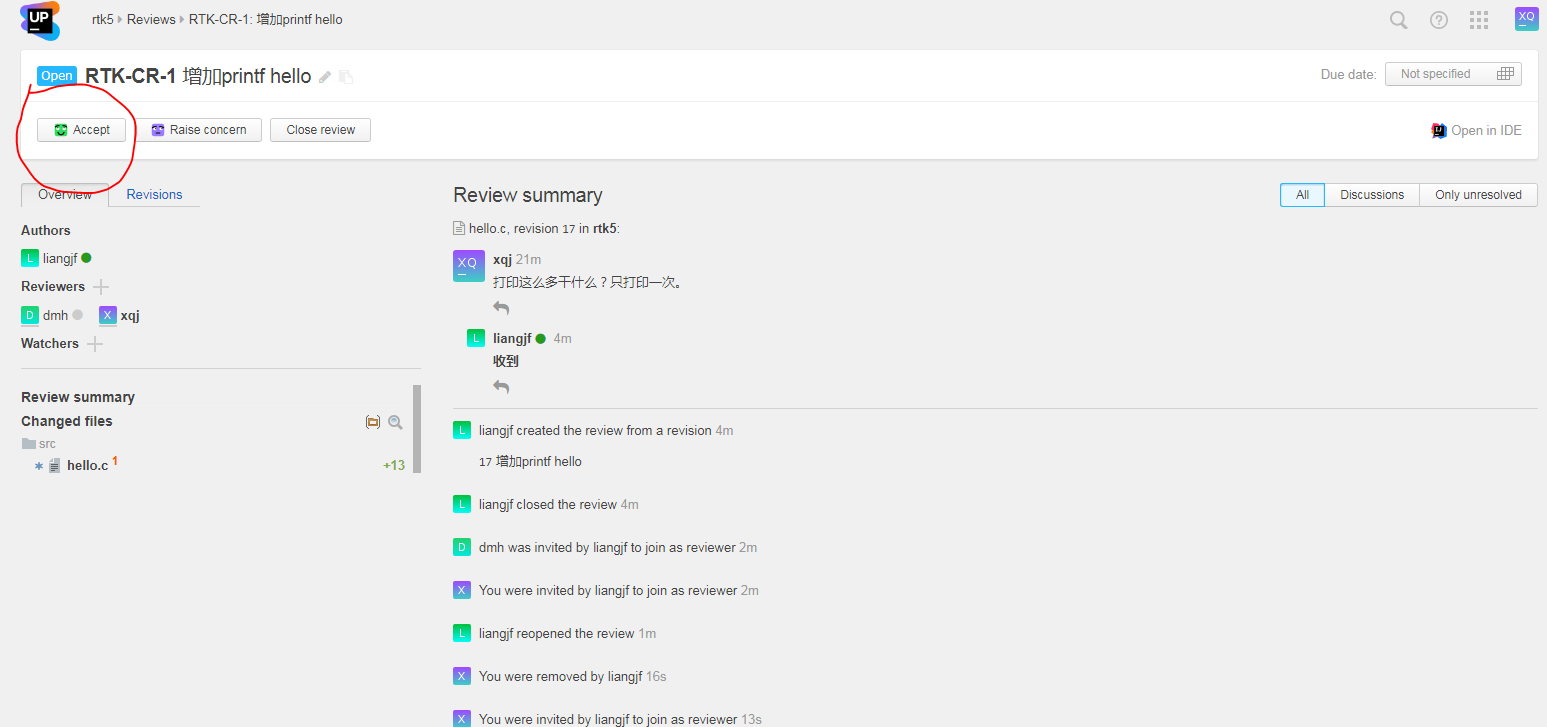
3.添加代码审查成员

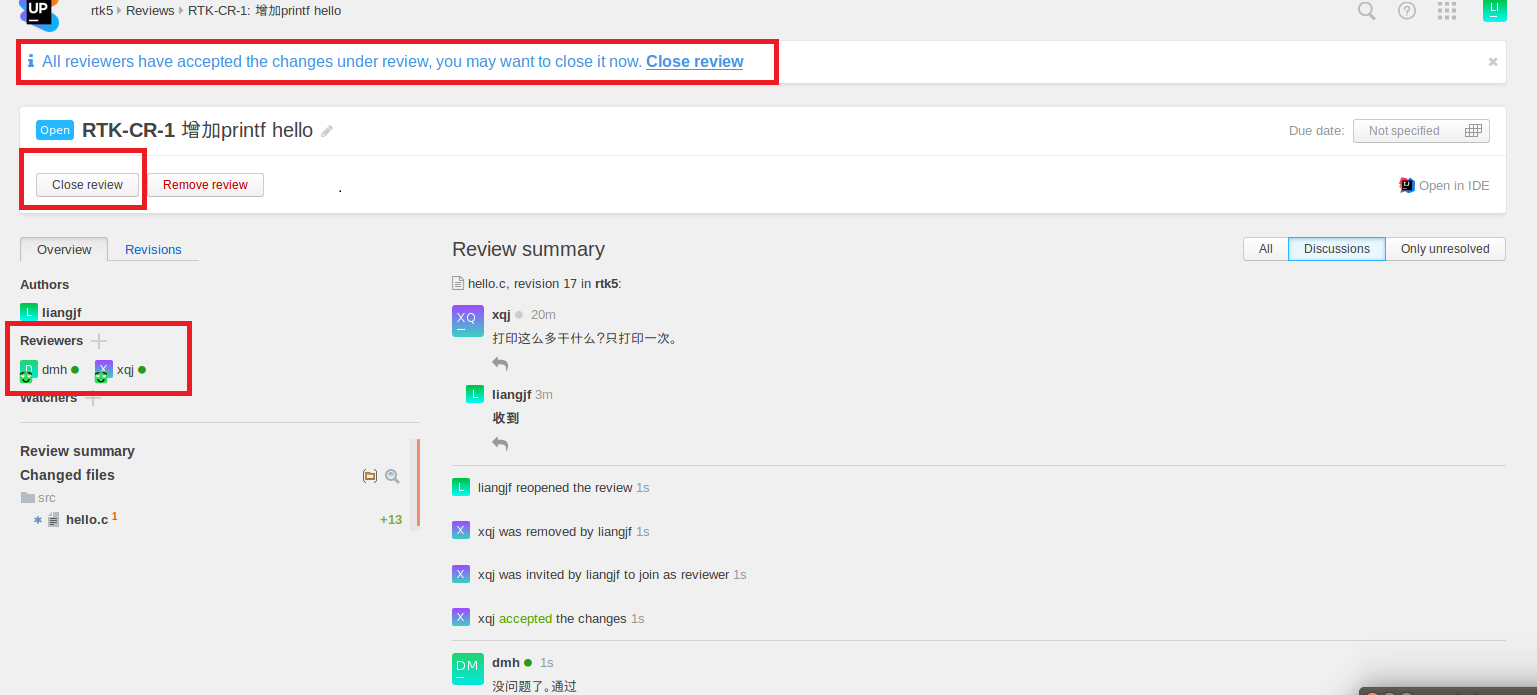


4.成员点开修改的代码浏览，并且对代码作评审，可以详细到对每一行代码作评审



5.代码审查成员对代码审查后，提交者会做出修改。修改到最终所有代码审查成员Accept此次review申请。



6.所有成员Accept此次review申请，代码提交者或项目管理员会关闭此次review申请

至此，一个简单的代码审查的流程过了一遍，整个过程就像一次面对面的讨论，很方便。

# 10.如何更好的应用于代码审查

## 10.1. 避免经常性Code Review

（1）要Review的代码越多，那么要重构，重写的代码就会越多。而越不被程序作者接受的建议也会越多，唾沫口水战也会越多。

（2）程序员代码写得时候越长，程序员就会在代码中加入越来越多的个人的东西。

（3）越接近软件发布的最终期限，代码也就不能改得太多。

## 10.2. Code Review不要太正式，而且要短

有时，更建议不用Code Review， 而是面对面的交流，这样可以更好的无障碍谈论代码，而且越正式，就会有一种紧迫感，长期会造成一种想逃避的心理。

长期正式的**Code Review**可能会造成下面的结果：

（1）只有在Checklist上存在的东西才会被Review。

（2）Code Reviews 变成了一种礼节性的东西，其实没真正用心审查。

## 10.3. 尽可能的让不同的人Reivew你的代码

如果可能的话，不要总是只找一个人来Review你的代码，不同的人有不同的思考方式，有不同的见解，所以，不同的人可以全面的从各个方面评论你的代码。

下面是几个优点：

（1）从不同的方向评审代码总是好的。

（2）会有更多的人帮你在日后维护你的代码。

（3）这也是一个增加团队凝聚力的方法。

## 10.4. 保持积极的正面的态度

程序员最大的问题就是“自负”，尤其当我们Reivew别人的代码的时候，过度抨击别人的代码，质疑别人的能力。靠这种手段来表现自己的程序员，其实是就是传说中所说的“半瓶水”。

无论是代码作者，还是评审者，都需要一种积极向上的正面的态度，作者需要能够虚心接受别人的建议，因为别人的建议是为了让你做得更好；评审者也需要以一种积极的正面的态度向作者提意见，因为那是和你在一个战壕里的战友。