

深入理解大数据-大数据处理与编程实践

大数据开源项目的开发工具与 流程简介

南京大学计算机科学与技术系

主讲人：黄宜华 顾荣

鸣谢：本课程得到Google公司(北京)
中国大学合作部精品课程计划资助

大数据开源项目的开发工具与流程简介

1. 开发环境工具IntelliJ IDEA简介
2. 代码库版本控制工具Git简介
3. 项目组织管理工具Maven简介
4. 项目的Code Style
5. 实验1：基本开发工具的安装使用以及开发流程熟悉
6. 实验指南

1. 开发环境工具IntelliJ IDEA简介

- IntelliJ IDEA是一个相当智能化的Java、Scala开发IDE环境

IntelliJ IDEA is focused on raising your productivity by providing **instant** and **clever code completion**, **on-the-fly code analysis**, easy project navigation and reliable refactorings.

- For Eclipse user:

IntelliJ IDEA没有类似Eclipse的workspace，最顶级的是Project，次级别是Module，一个Project可以有多个Module，每个project需要打开一个窗口（类似Visual Studio）

- 官网

<https://www.jetbrains.com/idea/>

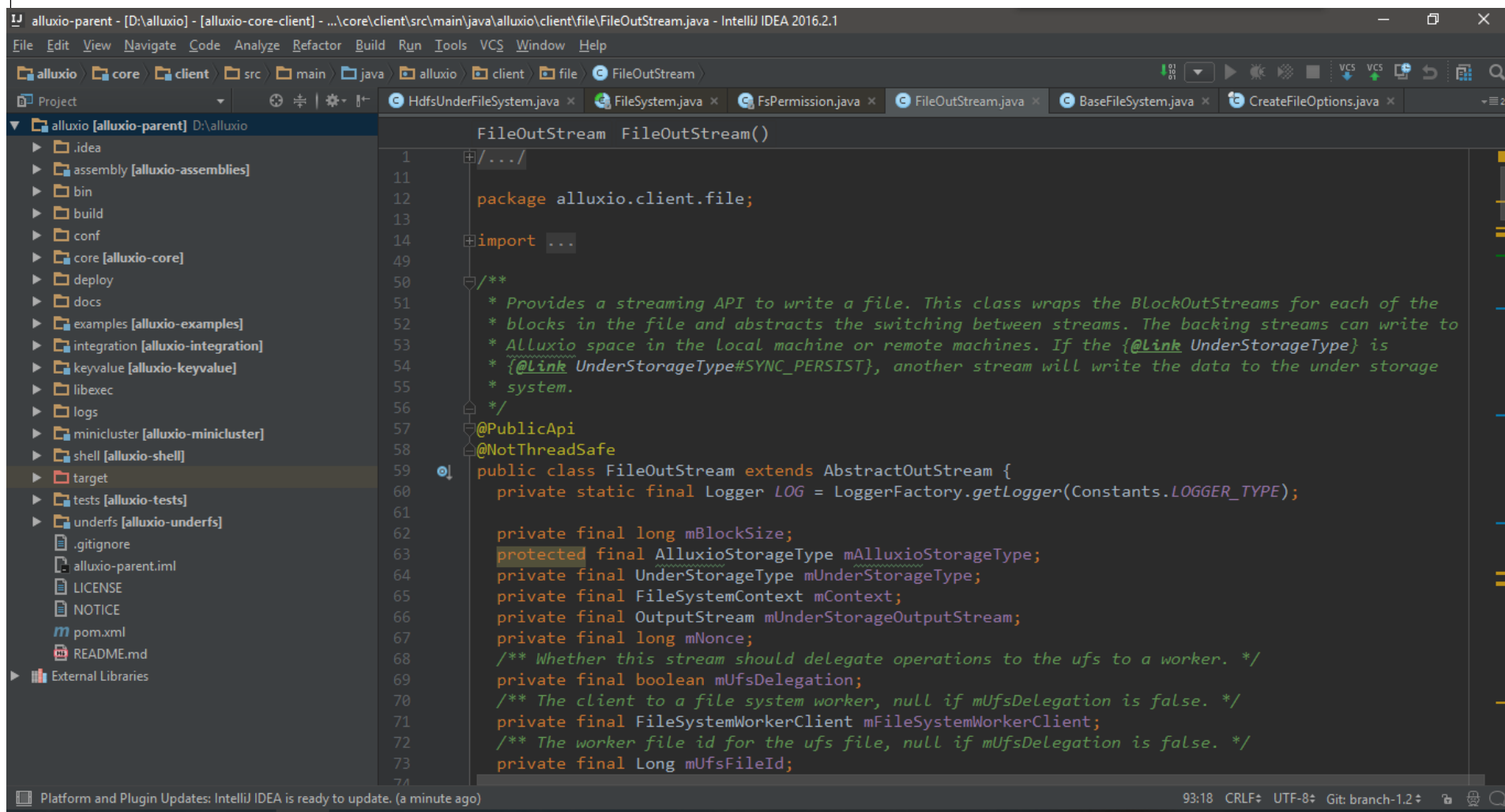
- 参考教程

<http://wiki.jikexueyuan.com/project/intellij-idea-tutorial/> (中文)

<https://www.jetbrains.com/idea/help/meet-intellij-idea.html>

开发环境工具IntelliJ IDEA简介

IntelliJ IDEA 基本界面



2.后续实验中你可能需要使用IDEA

- 1. 添加Lib
 - 通过IDEA的项目设置
 - 通过Maven（插件）
- 2. 生成Jar包
 - 使用IDEA的build功能
 - 使用Maven install/package
- 3. 本地调试Hadoop程序

2.代码库版本控制工具Git简介

Git简介

- Git是一个分布式版本控制工具，适用于管理
 - 大型开源软件源码
 - 个人软件源码
 - 私人文档
 - ...
- 在不同粒度上进行版本控制，能够得到任意一个版本的内容
 - 仓库（repository）
 - 分支（branch）
 - 每一次的改动（commit）

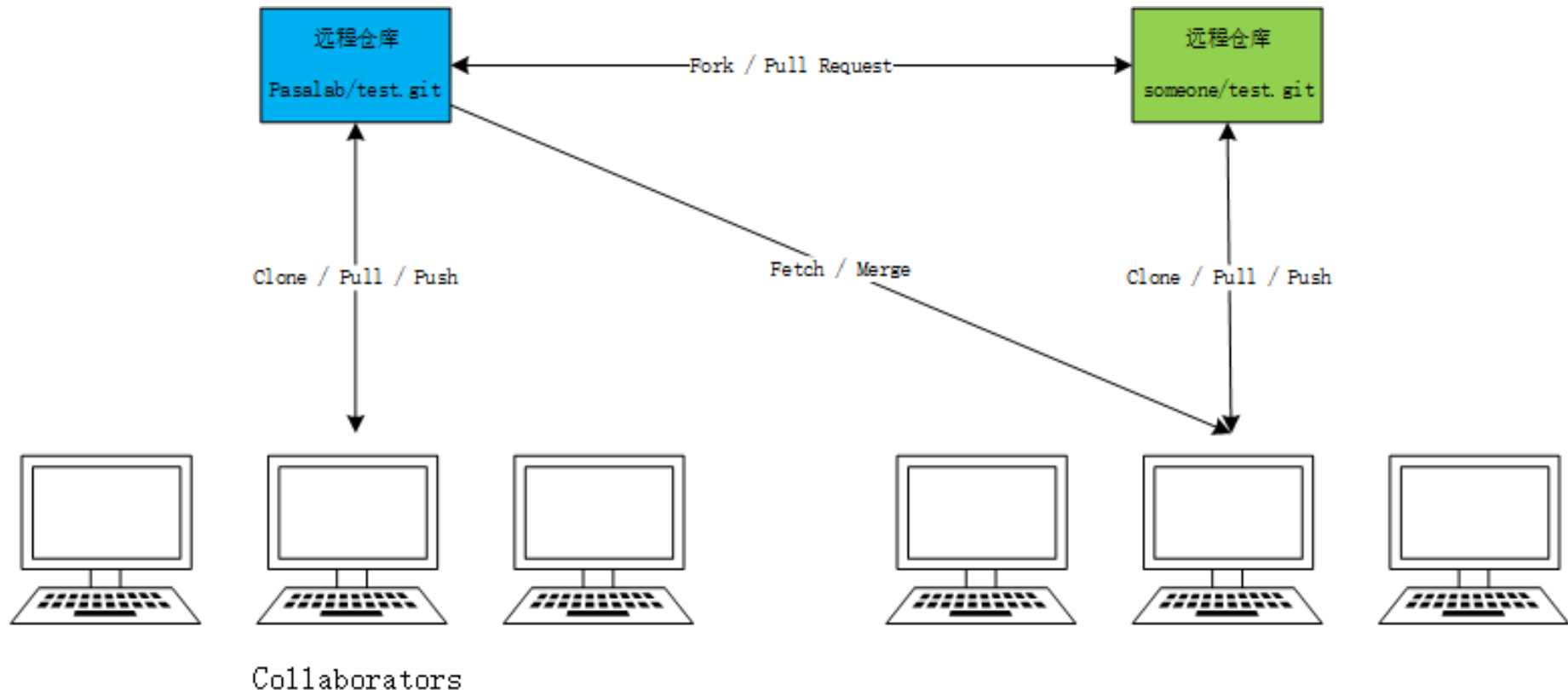
2.代码库版本控制工具Git简介

Git简介

- Git
 - 版本控制（存档点）
 - 分支管理
 - 标签管理
 - 远程仓库
- GitHub
 - 最常用的Git仓库

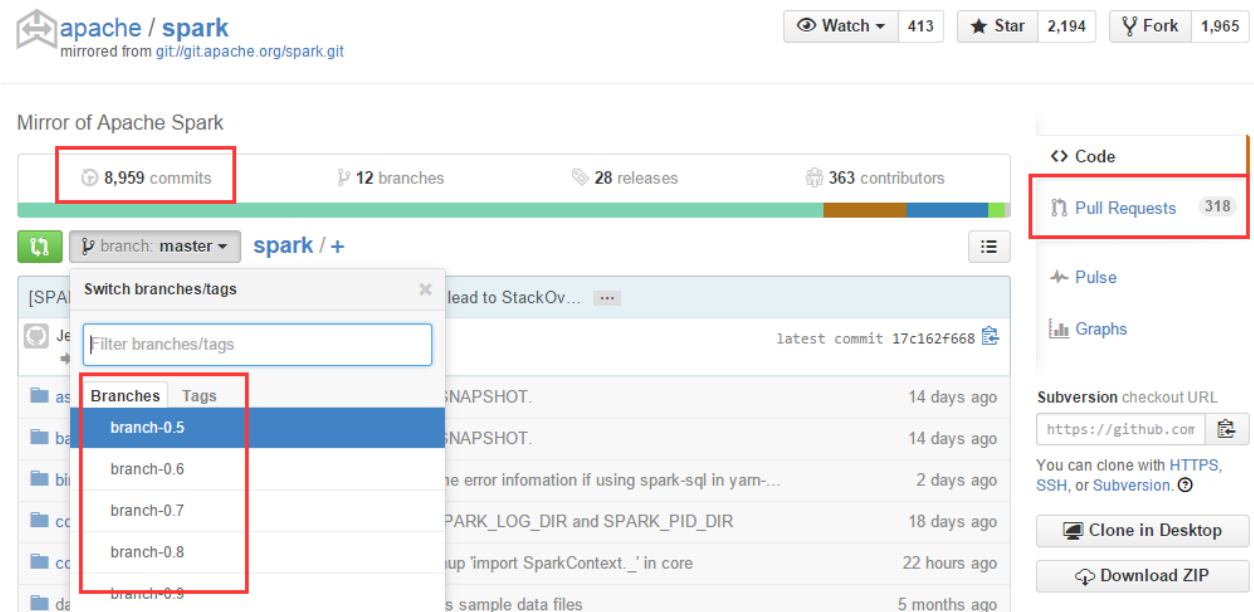
Git项目分布框架

- 项目存储于远程版本库中
- Collaborators拥有直接修改远程仓库内容的权限
- 其他人通过提交PR的方式修改内容



Git项目版本框架

- 一个项目为一个Repository
- 一个Repository可包含多个分支（Branch）
- 通常不同版本对应不同分支
- 一个分支由若干Commit构成
- 一个Commit表示一次内容修改
- 一个Commit可以包含对多个文件内容的修改
- 一个PR可以包含多个Commit



安装Git

- 这里的安装主要是指Git的客户端
 - For Windows
 - Git for Windows (适合shell 命令使用)
 - GitHub Desktop (良好界面的桌面应用, 仅适合GitHub)
 - For Linux
 - apt-get install git
 - yum install git
- 安装完后 `git --version` 检查版本

使用Git

- 本地从零开始
 - 对于现有项目
 - [可选]在网页上fork repository
 - `git clone https://github.com/github/gitignore`
 - 对于新建项目
 - 在网页上new repository后
 - `git clone https://github.com/someone/test.git`
 - 本地 `git init`
 - `git remote add origin https://github.com/someone/test.git`

使用Git

- 从命令行创建新仓库

```
touch README.md          #创建README.md文件
git init                 #初始化git项目格式，生成.git文件夹
git add README.md        #将README.md添加到缓存
git commit -m "first commit" #记录缓存内容的快照，并提交注释“first commit”
git remote add origin http://website/repo.git #为项目添加一个别名为origin的远端仓库
git push -u origin master #推送你的master分支与数据到名为origin的远端仓库
```

- 从命令行推送已有仓库

```
git remote add origin http://website/repo.git #为项目添加一个别名为origin的远端仓库
git push -u origin master #推送你的master分支与数据到名为origin的远端仓库
```

使用Git

- 对项目进行修改
 - 作为Collaborator
 - (在本地对内容进行了更改)
 - `git commit` #生成一次commit, 包含修改的内容
 - `git push` #将本地的修改提交至远程仓库
 - 非Collaborator
 - 在网页上create a pull request
 - 等待PR被merge

使用Git

- 同步最新的项目至本地
 - 作为Collaborator
 - git pull
 - 非Collaborator
 - git fetch upstream
 - git merge upstream/somebranch
 - 至此，本地代码已更新
 - git push origin somebranch:somebranch
 - 至此，远程仓库代码已更新

创建PR

PasaLab / tachyon
forked from Alluxio/alluxio

Unwatch 19

Star 0

Fork 1,074

Code

Pull requests 0

Projects 0

Wiki

Pulse

Graphs

Memory-centric Storage System for Big Data Analytics <http://tachyon-project.org>

17,896 commits

188 branches

7 releases

165 contributors

Your recently pushed branches:

test-0922 (16 minutes ago)

Compare & pull request

Branch: test-0922

New pull request

Create new file

Upload files

Find file

Clone or download

This branch is 1 commit ahead of Alluxio:master.

Pull request Compare

Yufa Zhou update chinese doc

Latest commit ee4a090 18 minutes ago

assembly	Fix nit in pom.xml	6 days ago
bin	Improve mesos test description	3 days ago
build	Quiet down line ending check	23 days ago
conf	Address Bin's comment	2 days ago

创建PR

Alluxio / alluxio

Watch 389

Unstar 2,607

Fork 1,074

<> Code

Issues 1

Pull requests 13

Projects 0


Wiki

Pulse

Graphs

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

 base fork: Alluxio/alluxio


base: master

...

head fork: PasaLab/tachyon

compare: test-0922

✓ **Able to merge.** These branches can be automatically merged.



[DOCFIX] update chinese doc

Write

Preview

AA B i “ <> ☰ ☷ ☸ ↶ @ 🚩

Leave a comment

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

☒ Allow edits from maintainers. [Learn more](#)

Create pull request

使用Git

- 分支管理
 - 查看本地分支 `git branch`
 - 查看远程分支 `git branch -r`
 - 创建本地分支 `git branch [name]`
 - 切换分支 `git checkout [name]`
 - 删除分支 `git branch -d [name]`
 - 创建远程分支 `git push origin [name]`
 - 删除远程分支 `git push origin :[name]`

使用Git

- 对开源项目作贡献的一般流程
 - fork a repository
 - `git clone https://github.com/PasaLab/tachyon.git`
 - `git checkout master`
 - `git remote add upstream https://github.com/Alluxio/alluxio.git`
 - `git fetch upstream`
 - `git merge upstream/master`
 - `git checkout -b new-branch`
 - (本地修改内容)
 - `git add <the files you modified>`
 - `git commit -m "modify something"`
 - `git push origin new-branch:new-branch`
 - create a pull request

其他

- Git功能远比上述要强大
- Git参考手册 <http://gitref.org/zh/index.html>
- GitHub <https://github.com/>
- <https://help.github.com/articles/create-a-repo/>
- <https://help.github.com/articles/fork-a-repo/>
- <https://help.github.com/articles/using-pull-requests/>
- 不错的Blog
<http://blog.csdn.net/ithomer/article/details/7529022>

3.项目组织管理工具Maven简介

Maven简介

- Maven 是一个项目管理和构建自动化工具
 - 构建项目框架
 - 管理项目结构
 - 自动解析依赖
 - 灵活插拔插件
 - 本地Maven库
- 目前，Hadoop、Spark、Alluxio均由Maven构建

Maven核心概念

- POM(Project Object Model)文件
 - 以xml形式对项目进行描述
 - 基本信息、依赖信息、插件信息等
- Maven依赖项
 - 项目之间的依赖关系
 - 自动解析并从库中获取必要的包
- Maven库
 - 本地库&远程库，包含所有项目所需的依赖包
 - 优先从本地库中寻找
 - <http://mvnrepository.com/> (default Maven 2 repository)
 - <https://maven-repository.com/> (can search other repositories)

Maven安装

- Linux、Windows均可，需要先安装Java
- 下载apache-maven-x.y.z.bin.zip/tar.gz
- <http://maven.apache.org/download.cgi>
- 解压
- 配置环境变量
 - M2_HOME=/path/to/maven
 - M2=\$M2_HOME/bin
 - add \$M2 to \$PATH
- mvn --version 检查版本

Maven基本命令

- `mvn archetype:generate -DgroupId=[xxx] -DartifactId=[yyy] -Dpackage=[zzz] -Dversion=[0.1-SNAPSHOT]` 新建Maven项目框架
- `mvn compile` 编译项目源码
- `mvn test` 运行测试
- `mvn package` 将项目打包
- `mvn install` 将项目安装到本地库
- `mvn clean` 清除已生成的项

- 编译举例：
- 打包Spark，依赖Hadoop版本为2.6，且支持native library，略过测试
- `mvn package -Phadoop-2.6 -Pnetlib-lgpl -DskipTests`

Maven基本命令

- 编译举例：
- 打包Spark，依赖Hadoop版本为2.6，且支持native library，略过测试
- mvn package -Phadoop-2.6 -Pnetlib-lgpl -DskipTests

```
<profile>
  <id>hadoop-2.6</id>
  <properties>
    <hadoop.version>2.6.0</hadoop.version>
    <jets3t.version>0.9.3</jets3t.version>
    <zookeeper.version>3.4.6</zookeeper.version>
    <curator.version>2.6.0</curator.version>
  </properties>
</profile>
```

pom.xml的profile

```
<profile>
  <id>netlib-lgpl</id>
  <dependencies>
    <dependency>
      <groupId>com.github.fommil.netlib</groupId>
      <artifactId>all</artifactId>
      <version>${netlib.java.version}</version>
      <type>pom</type>
    </dependency>
  </dependencies>
</profile>
```

mllib/pom.xml的profile

skipTests: 插件maven-surefire-plugin的配置选项

Maven插件命令

- 需要在pom.xml中添加插件的描述

```
<plugin>
```

```
  <groupId>org.apache.maven.plugins</groupId>
```

```
  <artifactId>maven-javadoc-plugin</artifactId>
```

```
  <version>2.9</version>
```

```
</plugin>
```

- eclipse插件，生成eclipse能够识别的项目

```
mvn eclipse:eclipse
```

- javadoc插件，生成Java API Doc

```
mvn javadoc:javadoc
```

- checkstyle插件，对源码进行格式检查

```
mvn checkstyle:checkstyle
```

- [scala-maven-plugin](#) Java-Scala代码混合编译

- ... <http://maven.apache.org/plugins/>

Maven – Hello World

- 命令行生成Maven项目

`mvn archetype:generate -DgroupId=org.pasalab.helloworldproject -DartifactId=helloworld -Dpackage=org.pasalab.helloworld -Dversion=0.1`

- 用IntelliJ IDEA生成Maven项目

File -> New -> Project -> Maven

New Project

GroupId	org.pasalab.helloworldproject
ArtifactId	helloworld
Version	0.1

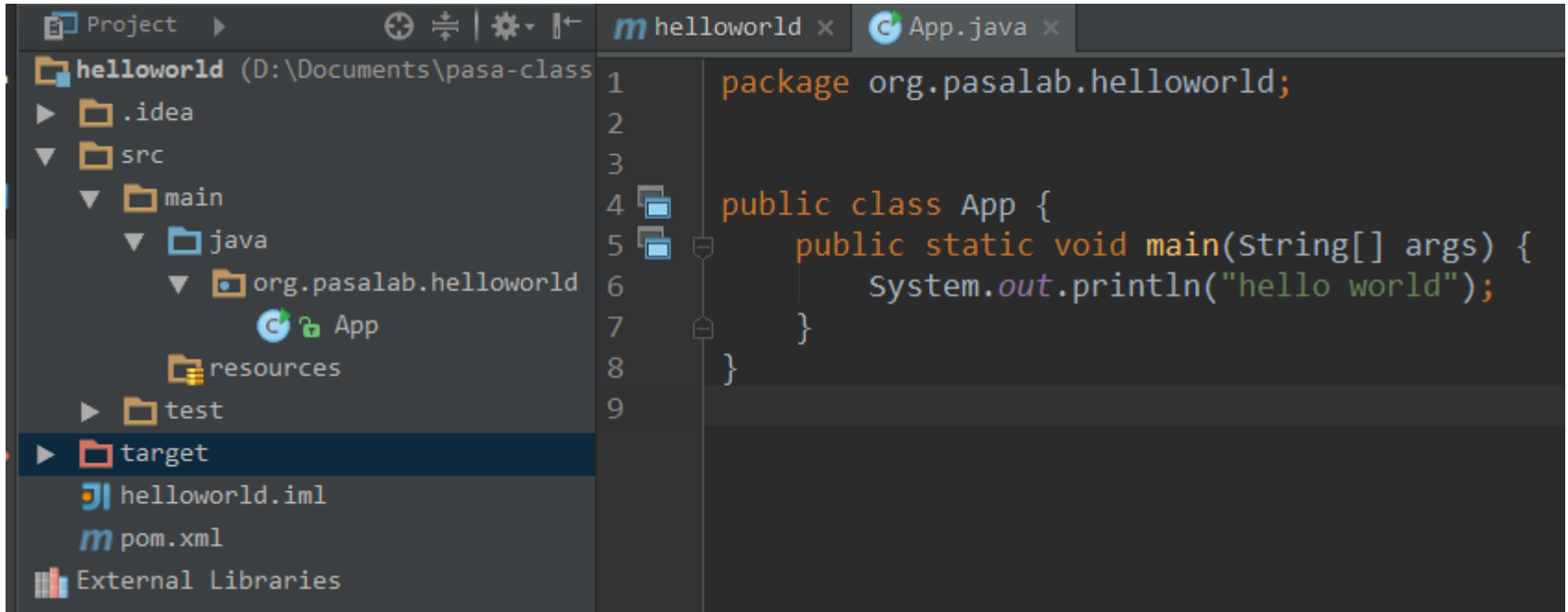
New Project

Project name:	helloworld
Project location:	D:\Documents\pasa-class\helloworld

The screenshot shows the IntelliJ IDEA interface. On the left, the Project Structure view displays the project hierarchy: `helloworld` (D:\Documents\pasa-class\helloworld) with subfolders `.idea`, `src` (containing `main` and `test`), `resources`, and `External Libraries`. The `main` folder is expanded, showing the `java` package. On the right, the `pom.xml` file is open, displaying the following XML content:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema
4         xsi:schemaLocation="http://maven.apache.org
5         <modelVersion>4.0.0</modelVersion>
6
7         <groupId>org.pasalab.helloworldproject</groupId>
8         <artifactId>helloworld</artifactId>
9         <version>0.1</version>
10
11
12 </project>
```

Maven – Hello World



- mvn package
- java -cp target/helloworld-0.1 org.pasalab.helloworld.App

```
D:\Documents\pasa-class\helloworld
java -cp .\target\helloworld-0.1.jar org.pasalab.helloworld.App
hello world
```

进一步学习Maven

- 详细的官方文档
 - <http://maven.apache.org/index.html>
- 不错的Maven入门教程
 - <http://www.oracle.com/technetwork/cn/community/java/apache-maven-getting-started-1-406235-zhs.html>
- 具体的POM文件分析
 - Hadoop、Spark、Alluxio

4.项目的Code Style

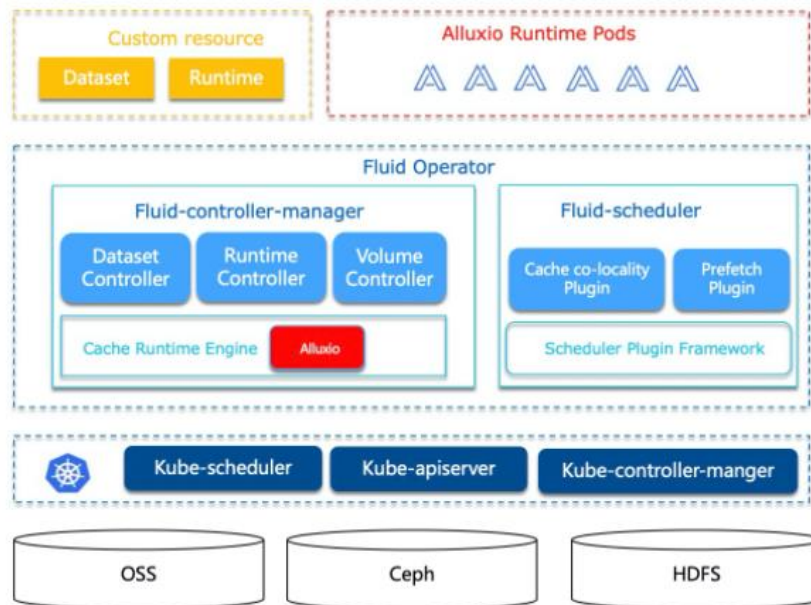
Code Style

- 作用
 - 整个工程项目代码风格统一，便于管理
 - 方便code review
 - 一定程度避免某些低级错误
 - 代码风格本身带有一定语义
- 规则约定
 - Doc上约定
 - Maven脚本定义规则
- 检查
 - 人工检查
 - Maven自动检查

5.实验1：基本开发工具的安装使用以及参与大数据云环境支撑平台Fluid的开发

Fluid简介

- Fluid是一个开源的Kubernetes原生的分布式数据集编排和加速引擎，主要服务于云原生场景下的数据密集型应用，例如大数据应用、AI应用等。通过定义数据集资源的抽象，实现如下功能：



5.实验1：基本开发工具的安装使用以及参与大数据云环境支撑平台Fluid的开发

实验内容与要求

1. 在本机上安装Git；
2. 创建自己的GitHub帐号；
3. 将fluid（<https://github.com/fluid-cloudnative/fluid>）fork到自己的GitHub仓库，clone到本地，建立新的分支完成给定任务；
4. 编译测试成功后提交commit，并push到自己GitHub帐号远程仓库相应分支，最后创建并提交pull request至Fluid的仓库；
5. 及时处理PR页面中他人提出的修改意见，本地修改后push到自己GitHub帐号远程仓库即可（不需要重新创建PR），并等待最终merge。
6. **DDL: 2020.12.26（由于Review阶段的耗时不确定，请尽早开始）**

实验1：基本开发工具的安装使用以及参与大数据云环境支撑平台Fluid的开发

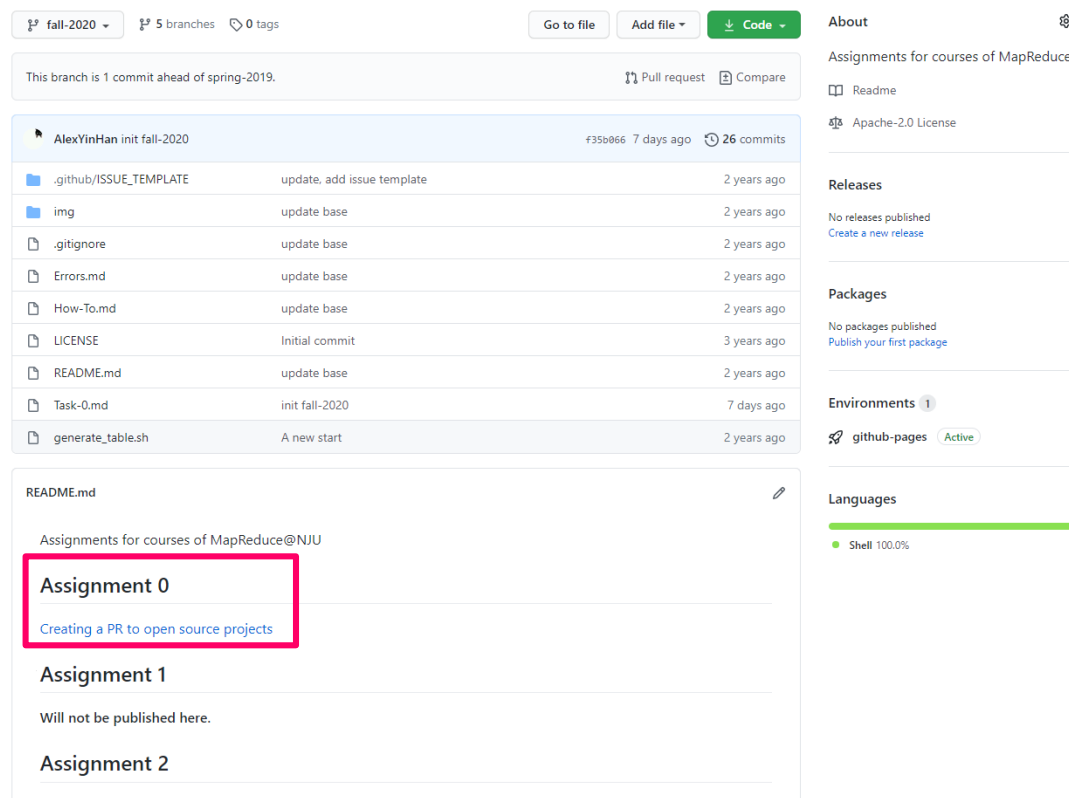
注意事项

- 每人只能完成自己的任务，若完成他人的任务则不计分；
- 仔细阅读任务要求（包括文档及注释规范）；
- 注意pull request标题及描述格式；
- 时常关注pull request页面，并及时处理相应评论

6.实验指南

领取自己的任务

- 进入课程仓库MR-Course-Assignments的fall-2020分支
 - <https://github.com/PasaLab/MR-Course-Assignments/tree/fall-2020>



fall-2020 5 branches 0 tags

This branch is 1 commit ahead of spring-2019. Pull request Compare

File	Commit	Time
github/ISSUE_TEMPLATE	update, add issue template	2 years ago
img	update base	2 years ago
.gitignore	update base	2 years ago
Errors.md	update base	2 years ago
How-To.md	update base	2 years ago
LICENSE	Initial commit	3 years ago
README.md	update base	2 years ago
Task-0.md	init fall-2020	7 days ago
generate_table.sh	A new start	2 years ago

README.md

Assignments for courses of MapReduce@NJU

Assignment 0

[Creating a PR to open source projects](#)

Assignment 1

Will not be published here.

Assignment 2

About

Assignments for courses of MapReduce

Readme

Apache-2.0 License

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Environments 1

github-pages Active

Languages

Shell 100.0%

6.实验指南

领取自己的任务

- 查看任务内容

Assignment 0 - New Contributors

Purpose

Learning how to participate in the open source projects by creating a pull request in GitHub.com

Step(s)

Please read these guides first, even if you known how to create a PR

[Steps to create this PR](#)

[常见错误](#)

Tasks

Part 1 (Due: xx/x/2020)

NJUID	Task	PR link	Merged	Score
17124051	#1	-	-	0
DG20330036	#2	-	-	0
DZ20330014	#3	-	-	0
DZ20330021	#4	-	-	0
DZ20330029	#5	-	-	0
DZ20330043	#6	-	-	0
MF20330001	#7	-	-	0
MF20330002	#8	-	-	0

6.实验指南

领取自己的任务

- 查看任务内容

Assignment 0 - New Contributors

Purpose

Learning how to participate in the open source projects by creating a pull request in GitHub.com

Step(s)

Please read these guides first, even if you know how to create a PR.

[Steps to create a PR](#)

[常见错误](#)

Tasks

Part 1 (Due: xx/x/2020)

	Task	PR link	Merged	Score
17124	#1	-	-	0
DG20330036	#2	-	-	0
DZ20330014	#3	-	-	0
DZ20330021	#4	-	-	0
DZ20330029	#5	-	-	0
DZ20330043	#6	-	-	0
MF20330001	#7	-	-	0
MF20330003	#8	-	-	0

2.完成任务后，请联系助教并发送你的PR链接，助教会将其填到这里。

3.这里一开始会填入“N”，你提交的commit被merge后，这里才会更新为“Y”，至此本次实验才算完成。

1.任务编号，点击链接跳转到 issue_list.md 中对应条目。

6.实验指南

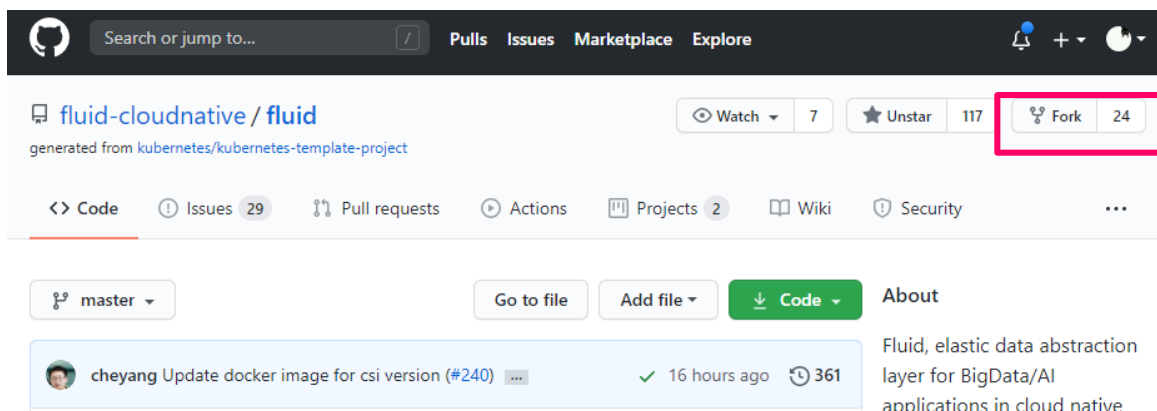
Demo Task

- 任务内容
 - 类别： 删除无用注释
 - 位置： pkg/utils/kubectrl/configmap.go
 - 内容： 删除方法func kubectrl上的注释，该注释与 func SaveConfigMapToFile 重复，是无用注释

6.实验指南

Demo Task

- 环境准备
 - 1. 注册并登录GitHub账号 （下面以用户AlexYinHan为例）
 - 2. 找到Fluid项目的GitHub页面
 - 3. 点击右上角的Fork，将该仓库Fork到自己的GitHub账号下



6.实验指南

Demo Task

● 环境准备

● 4. git clone 自己账号下的项目

- 在本机的某个目录下执行

- `cd <your_path>`

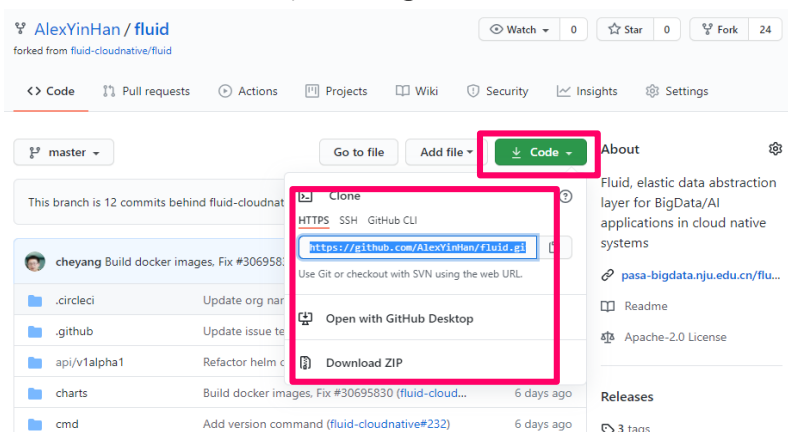
- `git clone https://github.com/AlexYinHan/fluid.git`

- clone成功后，该目录下会出现一个名为fluid的目录

● 5. 本地配置git，命令行下运行以下命令

- `$git config --global user.name <your_name>`

- `$git config --global user.email <your_github_email>`



6.实验指南

Demo Task

● 环境准备

● 5.设置git remote

● 首先查看git remote状态:

- cd fluid

- git remote -v

● 会得到如下结果:

- origin https://github.com/AlexYinHan/fluid.git (fetch)

- origin https://github.com/AlexYinHan/fluid.git (push)

● 为了后续方便, 我们手动创建一个Upstream Remote, 其repo url为Fluid项目的地址:

- git remote add upstream https://github.com/fluid-cloudnative/fluid.git

- 使用git remote -v查看结果

```
$ git remote -v
origin https://github.com/AlexYinHan/fluid.git (fetch)
origin https://github.com/AlexYinHan/fluid.git (push)
upstream https://github.com/fluid-cloudnative/fluid.git (fetch)
upstream https://github.com/fluid-cloudnative/fluid.git (push)
```

6.实验指南

Demo Task

- 本地修改
 - 1.创建新的本地分支
 - 根据自己的任务内容，设置一个简洁明确的分支名字
 - `git checkout -b remove_redundant_comment`

6.实验指南

Demo Task

- 本地修改
 - 2.完成任务要求

```
05
06 /**
07  *
08  * save the key of configMap into a file
09  */
10 func kubect1(args []string) ([]byte, error) {
11     binary, err := exec.LookPath(kubect1Cmd[0])
12     if err != nil {
```

```
96 /**
97  *
98  *
99  */
00 func kubect1(args []string) ([]byte, error) {
```

6.实验指南


Demo Task

- 本地修改
 - 3.完成后，查看代码仓库的状态
 - git status

```
LS@LAPTOP-IFJV20S7 MINGW64 ~/Desktop/pasa/助教/fluid (remove_redundant_comment)
$ git status
On branch remove_redundant_comment
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   go.mod
    modified:   pkg/utils/kubect1/configmap.go

no changes added to commit (use "git add" and/or "git commit -a")
```



不相关的修改，请不要add

6.实验指南

Demo Task

- 本地修改
 - 4.commit本地修改
 - 添加修改的文件
 - `git add pkg/utils/kubectrl/configmap.go`
 - 生成commit, commit-message应当足够能描述该commit修改的内容, 同时应当足够简洁
 - `git commit -m <commit-message>`
 - Notes: 请确保你只add了自己改动的文件, commit前请确保你已经Review过自己的全部改动 (可通过git status确认)

6.实验指南

Demo Task

- 本地修改
 - 4.5. 查看commit
 - git log 命令可以查看最近的commit

```
commit 83222ef6b9ee7f4e52dfc27f5cdf47133be3e282
Author: AlexYinHan <411629507@qq.com>
Date: Sat Oct 17 19:05:39 2020 +0800

    remove redundant comment on 'kubectl' function

commit b5b2ed3025f7e7bf750ab19e821fce4ef0a3f3cd (upstream/master, origin/master,
origin/HEAD, master)
Author: cheyang <cheyang@163.com>
Date: Sat Oct 17 00:17:26 2020 +0800

LS@LAPTOP-IFJV20S7 MINGW64 ~/Desktop/pasa/助教/fluid (remove_redundant_comment)
```

6.实验指南

Demo Task

- 本地修改

- 5. Rebase && Push

- Rebase与Merge有着相似的功能，在Fluid项目里使用Rebase作为分支合并的主要方式
 - 在完成全部改动，并将改动全部commit后，就可以将改动push到自己的远程仓库
 - 由于同时可能有多个人同时并行地进行该项目的修改，因此为了防止合并冲突，强烈建议在Push前进行Rebase操作。
 - Rebase操作会将寻找当前分支和另一个分支的最近公共祖先Commit，然后首先作用另一个分支的新Commits，接着作用当前分支的新Commits。简单来说，是一条分叉链表的合并操作，最终将会得到一条无分叉的commit链。
 - 在进行Rebase操作时，如果发生合并冲突，可以直接在本地分支上进行解决。

6.实验指南

Demo Task

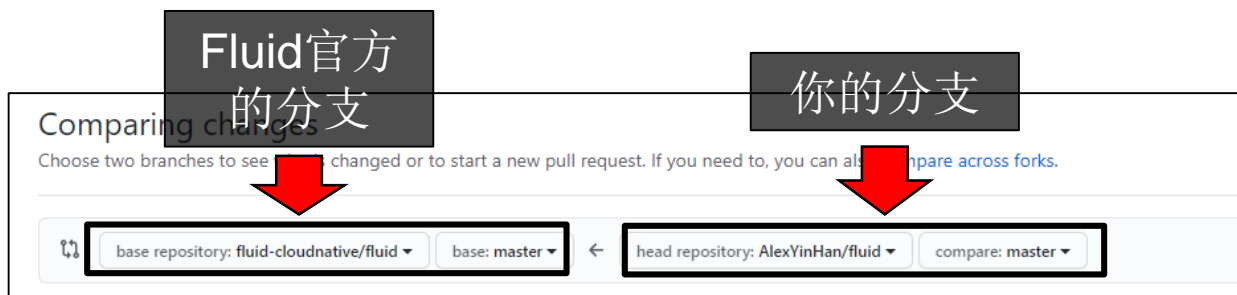
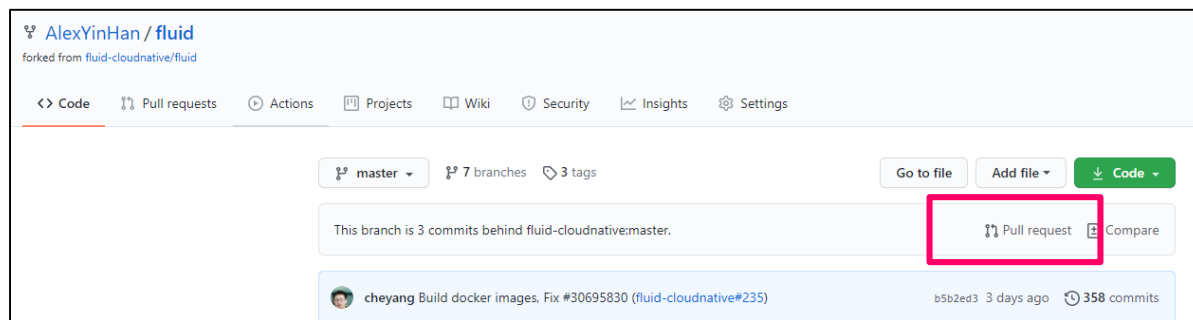
- 本地修改
 - 5. Rebase && Push
 - 首先，同步upstream仓库：
 - `git fetch upstream`
 - 接着，以upstream的master分支为目标进行`git rebase`
 - `git rebase upstream/master`
 - 若无冲突发生，即可进行接下来的Push操作：
 - `git push origin`
 - `git push --set-upstream origin remove_redundant_comment`
 - 上述命令将会提交到你自己的远程仓库，也就是名为origin的remote，这也是你拥有写权限的仓库。如果直接提交到`upstream` remote将会因为权限问题被拒绝

6.实验指南

Demo Task

● Pull Request

- 点击Pull Request，在新页面中选择自己改动后的分支(e.g. remove_redundant_comment)
- **此时，你可以最后一次Review自己提交的commit，以及所做的改动**



6.实验指南

Demo Task

- Pull Request

- 按照Pull Request模板进行填写，PR标题简述改动内容，PR中按照模板提示进行填写。
- 全部填写完成后，提交这个PR。



AlexYinHan commented now



I . Describe what this PR does

Remove redundant comment on `kubect1` function. (pkg/utils/kubectl/configmap.go)

II. Does this pull request fix one issue?

III. List the added test cases (unit test/integration test) if any, please explain if no tests are needed.

IV. Describe how to verify it

V. Special notes for reviews

6.实验指南

Demo Task

- PR Review
 - 提交PR后，请持续关注自己PR的状态。
 - Reviewer会对你的改动进行审核，讨论或提问，要求修改等
 - 请在相应的位置及时进行回复、或修改，如果你认为Reviewer给出的意见存在问题，也可在相应位置回复和Reviewer进行讨论。

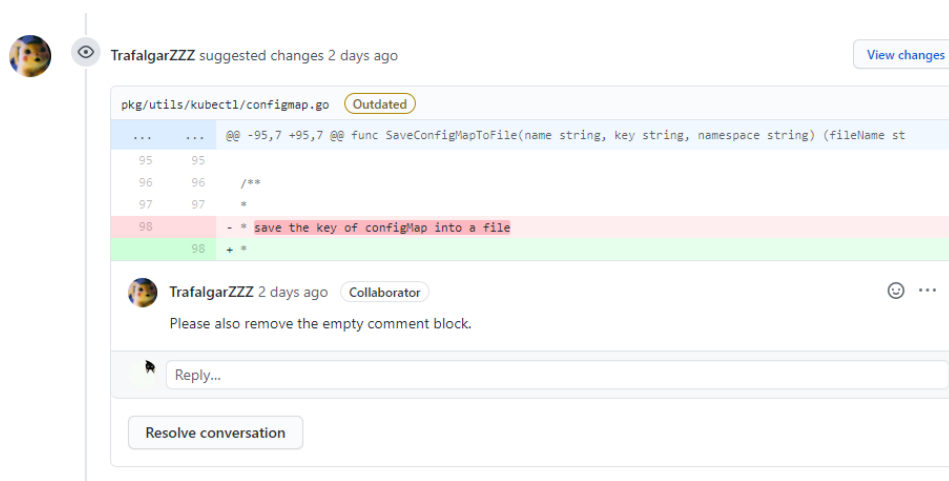
6.实验指南

Demo Task

- PR Review

- 修改

- 如果Reviewer认为你的改动存在问题，将会在相应的地方给出修改建议，并要求重新修改(Request Change):
- 这意味着你需要重复2到4步
- 注意：完成修改后直接commit->push即可，此时你的PR页面已经自动更新，不需要再次创建Pull Request



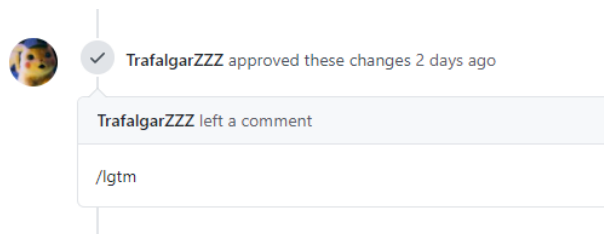
6.实验指南

Demo Task

- PR Review

- 接收

- 如果Reviewer认为你的改动没问题，那么通常会在该PR上留言：



- "lgtn"为 “Looks good to me”的简称，代表该PR可以被合并，
 - 在确定看到自己的PR状态从"Open"变为"Merged"后，恭喜你，正式完成了对于开源项目的贡献

课程讨论群

作业分配与线上答疑在此群，请务必加入！



群名称:2020研究生MapReduce大...

群 号:719869825

Thanks!