

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

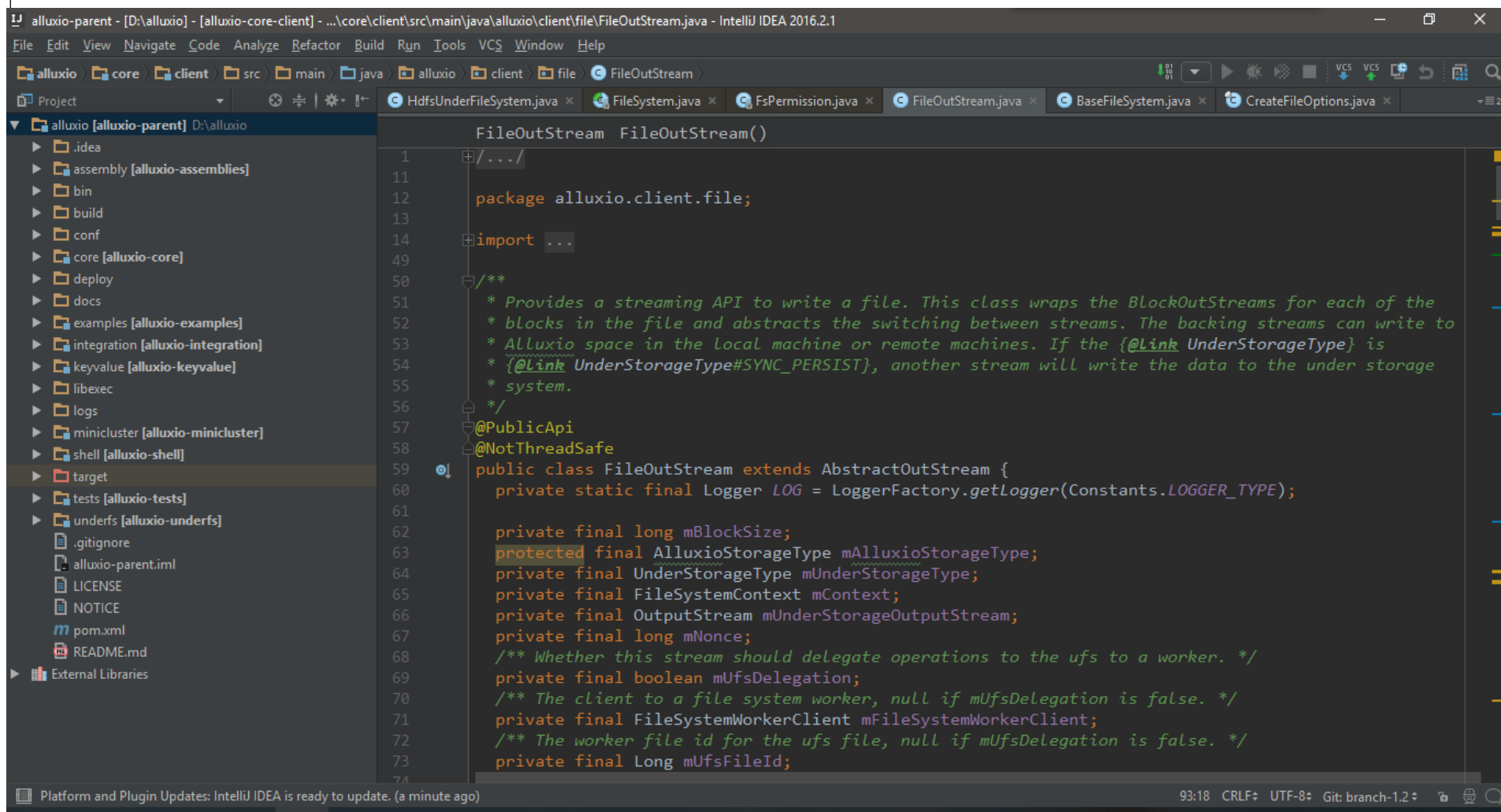
<https://www.jetbrains.com/idea/>

- 参考教程

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

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

IntelliJ IDEA 基本界面



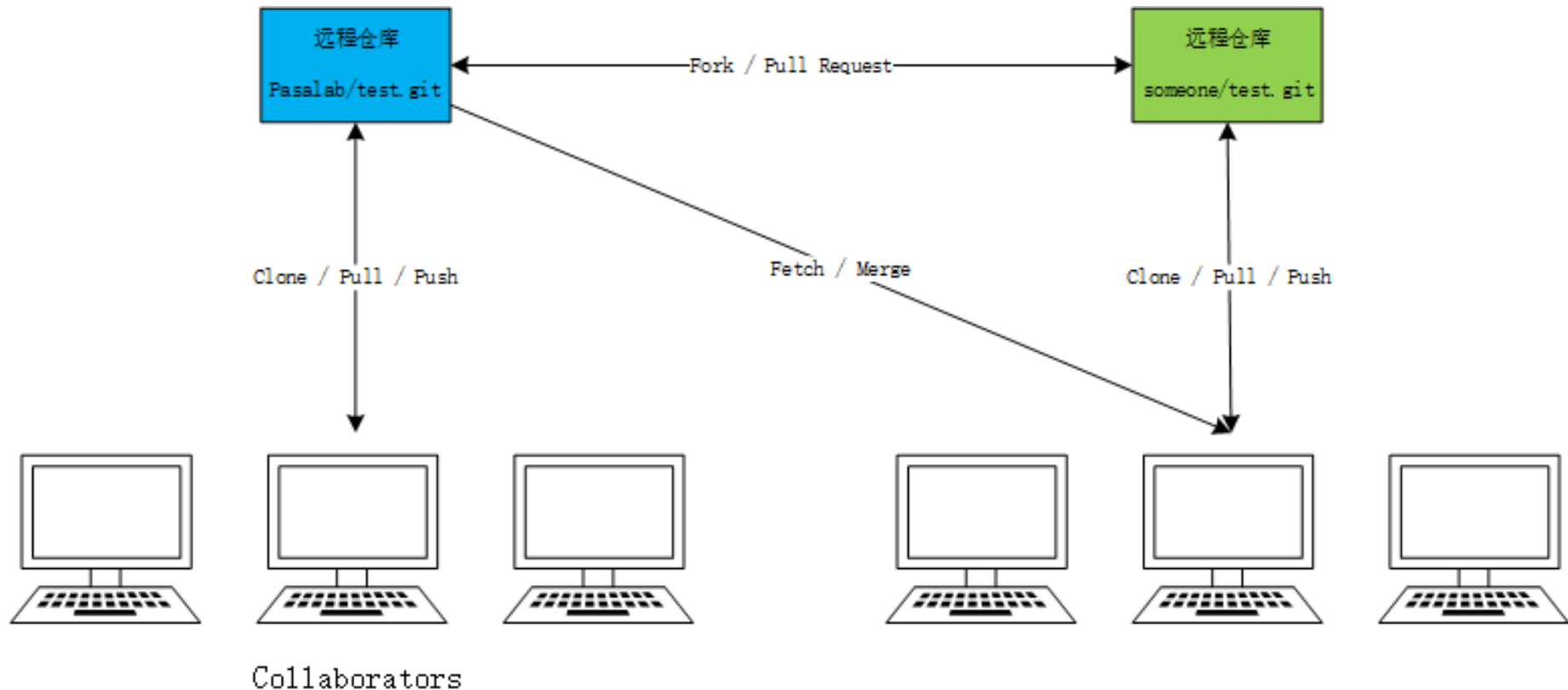
2.代码库版本控制工具Git简介

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）

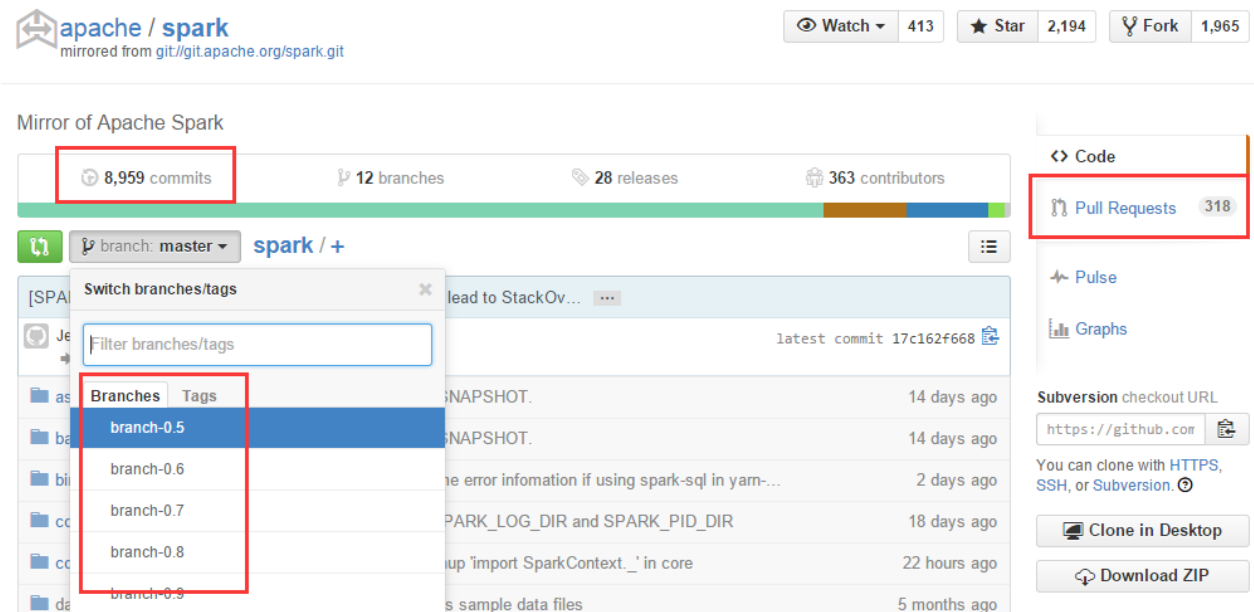
Git项目分布框架

- 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

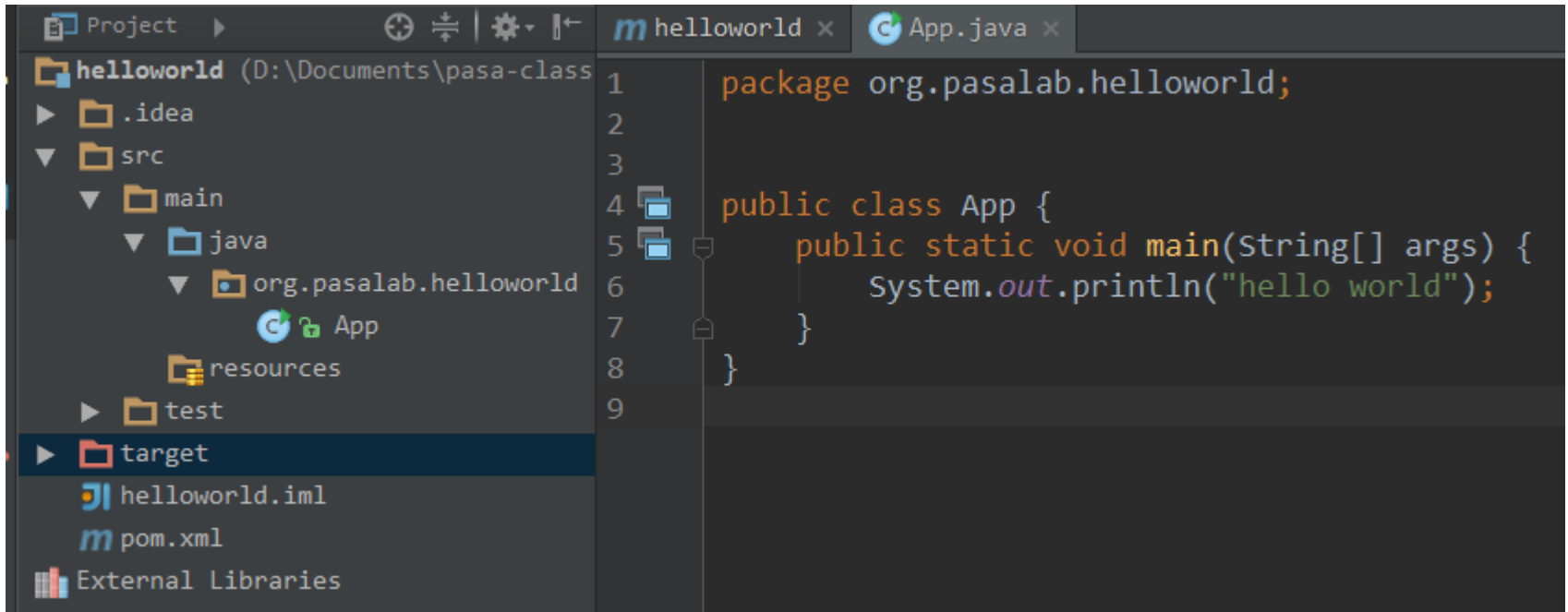
New Project

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:\Document) contains `.idea`, `src` (with `main` and `test` subfolders), `resources`, and `External Libraries`. The `main` folder is expanded, showing `java` and `resources` subfolders. On the right, the `pom.xml` file is open, showing 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     </project>  
12
```


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

Alluxio项目Code Style

- 2空格缩进
- 每行不超过100字符
- import语句根据前缀分组
- 类成员变量以m为前缀
- 静态类变量以s为前缀
- 具体参考

<http://www.alluxio.org/docs/master/en/Contributing-to-Alluxio.html>

Alluxio项目Code Style

项目的Code Style

```
80 @NotThreadSafe // TODO(jiri): make thread-safe (c.f. ALLUXIO-1624)
81 public final class TieredBlockStore implements BlockStore {
82     private static final Logger LOG = LoggerFactory.getLogger(Constants.LOGGER_TYPE);
83     // TODO(bin): Change maxRetry to be configurable.
84     private static final int MAX_RETRIES = 3;
85
86     private final BlockMetadataManager mMetaManager;
87     private final BlockLockManager mLockManager;
88     private final Allocator mAllocator;
89     private final Evictor mEvictor;
90
91     private final List<BlockStoreEventListener> mBlockStoreEventListeners = new ArrayList<>();
92
93     /** A set of pinned inodes fetched from the master. */
94     private final Set<Long> mPinnedInodes = new HashSet<>();
95
96     /** Lock to guard metadata operations. */
97     private final ReentrantReadWriteLock mMetadataLock = new ReentrantReadWriteLock();
98
99     /** ReadLock provided by {@link #mMetadataReadLock} to guard metadata read operations. */
100    private final Lock mMetadataReadLock = mMetadataLock.readLock();
101
102    /** WriteLock provided by {@link #mMetadataReadLock} to guard metadata write operations. */
103    private final Lock mMetadataWriteLock = mMetadataLock.writeLock();
104
105    /** Association between storage tier aliases and ordinals. */
106    private final StorageTierAssoc mStorageTierAssoc;
107
108    /**
109     * Creates a new instance of {@link TieredBlockStore}.
110     */
111    public TieredBlockStore() {
112        mMetaManager = BlockMetadataManager.createBlockMetadataManager();
113        mLockManager = new BlockLockManager();
114
115        BlockMetadataManagerView initManagerView = new BlockMetadataManagerView(mMetaManager,
116            Collections.<Long>emptySet(), Collections.<Long>emptySet());
117        mAllocator = Allocator.Factory.create(initManagerView);
118        if (mAllocator instanceof BlockStoreEventListener) {
119            registerBlockStoreEventListener((BlockStoreEventListener) mAllocator);
120        }
121    }
122}
```

5.实验1：基本开发工具的安装使用以及参与大数据存储系统Alluxio的开发

实验内容与要求

1. 在本机上安装Maven、Git;
2. 创建自己的Github帐号;
3. 将Alluxio/alluxio (<https://github.com/Alluxio/alluxio>) fork到自己的Github仓库，clone到本地，建立新的分支完成给定任务（**任务待分配**），并在本地使用maven进行编译测试（**需在linux环境下**）；
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Alluxio/alluxio (<https://github.com/Alluxio/alluxio>) （参考<http://www.alluxio.org/docs/master/cn/Contributing-to-Alluxio.html>）；
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。

实验1：基本开发工具的安装使用以及参与 大数据存储系统Alluxio的开发

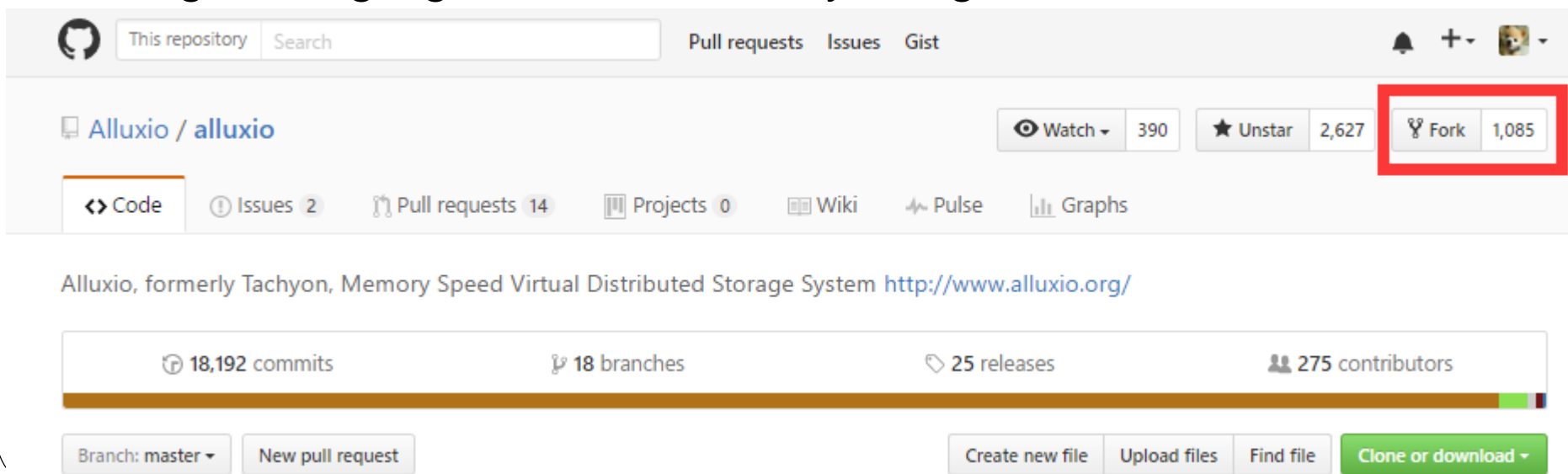
注意事项

- 每人只能完成自己的任务，若完成他人的任务则不计分；
- 确保修改后在本地进行编译测试并通过再提交；
- 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
- 在本学期课程后面的作业项目中，尽量使用Maven搭建管理整个工程，会有相应加分

6.实验指南

相关软件安装以及fork Alluxio仓库

- 在本机安装Maven、Git;
- 创建自己的Github账号（假设账号名为someone）;
- 将Alluxio项目（<https://github.com/Alluxio/alluxio>）fork到自己Github远程仓库
- 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"



The screenshot shows the GitHub repository page for Alluxio. The repository name is "Alluxio / alluxio". The page includes navigation links for Pull requests, Issues, and Gist. The repository statistics show 390 Watchers, 2,627 Stars, and 1,085 Forks. The "Fork" button is highlighted with a red box. Below the repository name, there are tabs for Code, Issues (2), Pull requests (14), Projects (0), Wiki, Pulse, and Graphs. The description of the repository is "Alluxio, formerly Tachyon, Memory Speed Virtual Distributed Storage System" with a link to the website. At the bottom, there are statistics for 18,192 commits, 18 branches, 25 releases, and 275 contributors. The bottom navigation bar includes a dropdown for the current branch (master), a button for "New pull request", and buttons for "Create new file", "Upload files", "Find file", and "Clone or download".

Clone代码到本地、添加远程源并fetch最新代码

- `git clone https://github.com/someone/alluxio.git`
- 进入项目根目录 `cd alluxio`
- `git checkout master`
- `git remote add upstream https://github.com/Alluxio/alluxio.git`
- `git fetch upstream`
- `git merge upstream/master`

The screenshot shows the GitHub repository page for `saltylin / alluxio`, which is a fork of `Alluxio/alluxio`. The repository has 18,192 commits, 18 branches, 25 releases, and 275 contributors. The current branch is `master`. The `Clone or download` button is clicked, showing a dropdown menu with the following options:

- Clone with HTTPS (selected)
- Use Git or checkout with SVN using the web URL
- Use SSH

The HTTPS URL `https://github.com/saltylin/alluxio.git` is highlighted in a red box. Below the URL are two buttons: `Open in Desktop` and `Download ZIP`. The dropdown menu also shows the commit history for the `master` branch, including a commit by `haoyuan` titled `Update Architecture.md`.

新建分支并完成指定任务

- `git checkout -b <my-branch-name>`
- 修改相应文件
- 编译测试（在linux环境） `mvn clean install`
- 查看当前Git修改状态 `git status`
- `git add <修改的文件>` 或者 `git add .`
- `git commit -m "the description for the commit（简单描述你的修改）"`
- `git push origin <my-branch-name>:<my-branch-name>`

创建Pull Request

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Pull Request

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

创建Pull Request

- 创建PR时，PR的标题通常以[ALLUXIO-xxxx]为前缀（如图），如果是针对docs进行的修改则为[DOCFIX]前缀
- 若该任务为某一jira任务，则在description框中的第一行添加该jira的链接

[ALLUXIO-2262] Mark nested class Name in PropertyKey with annotation "@ThreadSafe"
#4033 opened 2 days ago by schemolyas

Conversation 13

Commits 1

Files changed 1



peisun1115 commented 7 days ago • edited



<https://alluxio.atlassian.net/browse/ALLUXIO-2236>

Use linked list (similar as LRU cache) to store the resources to reduce lock contention.

Contribution license agreement

- 创建PR后，会有个 alluxio-bot 提示需要`sign a contribution license agreement`，这个作用是确保我们写的代码能够被开源社区接受和使用，按照提示填写信息（**需填写完整**），之后会收到相关邮件，按提示进行操作即可



alluxio-bot commented 13 hours ago



Hi [redacted], thanks for your contribution!

In order for us to evaluate and accept your PR, we ask that you [sign a contribution license agreement](#). It's all electronic and will take just minutes.

处理PR页面中别人提出的评论及修改意见

- 切换到该PR对应的分支 `git checkout <my-branch-name>`
 - 进行相应修改
 - 编译测试 `mvn clean install`
 - 查看当前Git修改状态 `git status`
 - `git add <修改的文件>` 或者 `git add .`
 - `git commit -m "the description for the commit"`
 - `git push origin <my-branch-name>:<my-branch-name>`
- 注意，push到自己Github远程仓库就行了，不需要再次创建PR，相应的PR会检测到这次更改

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：MapReduce数据处理课程群

群 号：474649813

作业示例

[ALLUXIO-2302] Improve javadoc in Abstract Client (getAddress) Created: 09/Oct/16 Updated: 10/Oct/16

Status:	Open
Project:	Alluxio
Component/s:	Core - Client
Affects Version/s:	1.3.0
Fix Version/s:	None

Type:	Improvement	Priority:	Major
Reporter:	Calvin Jia	Assignee:	Yufa Zhou
Resolution:	Unresolved	Votes:	0
Labels:	NewContributor		
Remaining Estimate:	Not Specified		
Time Spent:	Not Specified		
Original Estimate:	Not Specified		

Last Update By:	Yufa Zhou
-----------------	-----------

Description

Remove the unnecessary description in the getAddress method. Only the @returns clause is necessary.

Thanks !