**项目报告**

**一、项目地址**

GitHub链接：<https://github.com/callmeahh/VersionControlSystem>

commit id：

**二、设计文档**

1.实现功能（命令行交互）

‐‐help：获取指令功能列表。

‐‐init：初始化本地仓库。

commit ：提交commit到本地仓库。

追加参数：

‐m：同时输入commit的描述信息。

branch：

追加参数：

‐a, ‐‐all：获取所有分支名。

‐c, ‐‐current：获取当前分支名。

‐n, ‐‐new：创建分支，需追加分支名。

‐s, ‐‐switch：切换分支，需追加分支名。

switch：切换分支，需追加分支名。

rename：重命名分支，需追加待命名的分支名和重命名的分支名。

reset：

追加参数：

‐c, ‐‐commit：本地仓库回滚到指定commit，需追加commit的key的前几位。

‐t：本地仓库回滚指定次数，需追加回滚的次数。

追加参数：

‐HARD：

追加参数：

‐c, ‐‐commit：工作区和本地仓库回滚到指定commit，需追加commit的key的前几位。

‐t：工作区和本地仓库回滚指定次数，需追加回滚的次数。

log：获取历史记录。

追加参数：

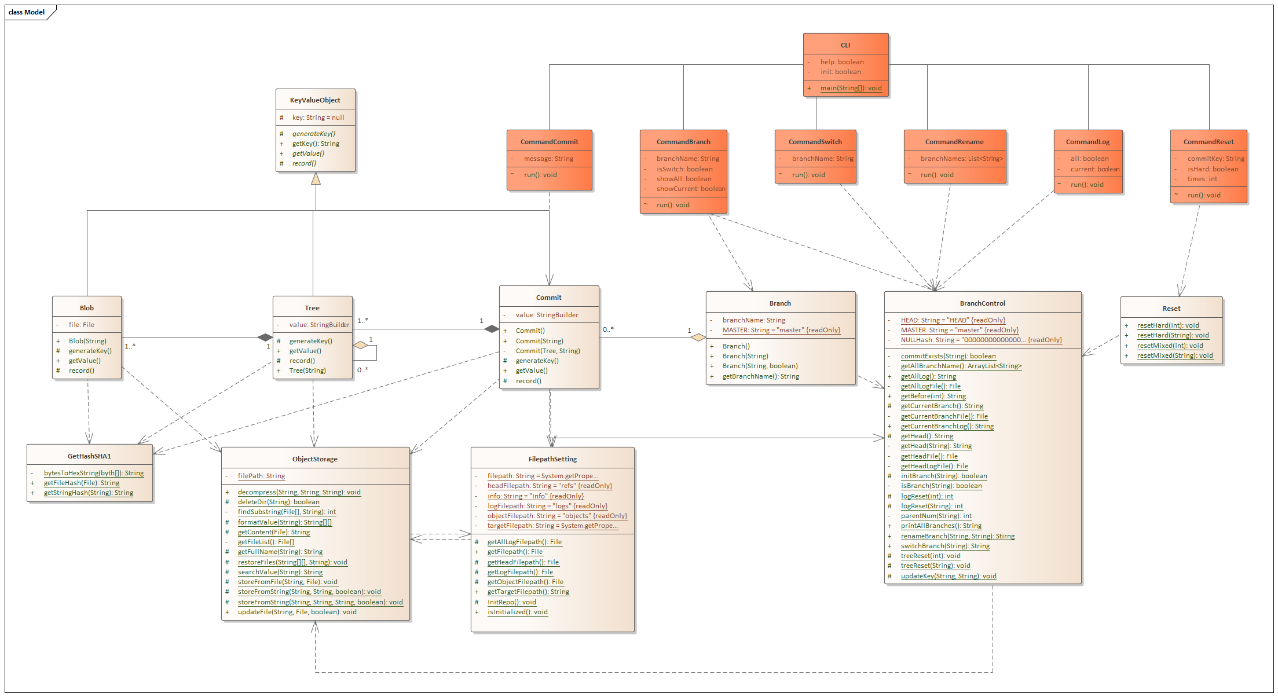
‐a, ‐‐all

：获取所有历史记录。

‐c, ‐‐current

：获取当前分支历史记录。

2.类的设计



项目类图

GetHashSHA1:

getFileHash(File)：将字符串通过SHA-1方法哈希。

getStringHash(String)：将字符串通过SHA-1方法哈希。

bytesToHexString(byte[])：将数组类型转为字符串。

FilepathSetting:

filepath：本地仓库文件存放路径。

objectFilepath

headFilepath

logFilepath

info

targetFilepath

setFilepath(String)：设置本地仓库文件保存路径。

getFilepath()：获取本地仓库文件夹，存放logs，objects，refs文件夹和HEAD，Info文件，HEAD文件记录当前分支，Info文件记录工作区路径。

getObjectFilepath()：获取objects文件夹，存放blob，tree，commit文件。

getHeadFilepath()：获取refs文件夹，存放各分支文件，记录各分支当前commit的key。

getAllLogFilepath()：获取logs文件夹，存放refs文件夹和HEAD文件，HEAD文件记录所有历史记录。

getLogFilepath()：获取logs.refs文件夹，存放各分支文件，记录各分支历史记录。

InitRepo()：初始化目标文件夹，生成objects文件夹，refs文件夹，logs文件夹，logs.refs文件夹。

getTargetFilepath()：获取工作区文件路径。

setTargetFilepath(String)：设置工作区文件路径。

ObjectStorage:

filePath：object类文件存放路径。

storeFromFile(String, File)：将object类文件存入objects文件夹。

storeFromString(String, String, boolean)：将字符串类型存入objects文件夹。

storeFromString(String, String, String, boolean)：将字符串类型存入指定文件夹。

updateFile(String, File, boolean)：更新文件内容。

getFileList()：获取objects文件夹中所有文件的数组。

findSubstring(File[], String)：由key前几位匹配objects文件夹中文件数组中的文件。

getFullName(String)：由key前几位匹配objects文件夹中文件数组中的文件，返回完整key。

searchValue(String)：由key前几位匹配objects文件夹中文件数组中的文件，返回value。

getContent(File)：获取文件中的内容。

deleteDir(String)：删除指定文件夹中所有内容。

formatValue(String)：将value格式化成二维数组。

restoreFiles(String[][], String)：将二位数组中的记录的值还原成文件。

KeyValueObject：

key

record()：将对象记录到文件。

generateKey()：生成key。

getValue()：获取value。

getKey()：获取key。

Blob（继承KeyValueObject）:

file

Blob(String)：通过传入的文件路径初始化Blob对象。

record()：方法重写。

generateKey()：方法重写。

getValue()：方法重写。

Tree（继承KeyValueObject）:

value

Tree(String)：通过传入的文件夹路径初始化Blob对象。

record()：方法重写。

generateKey()：方法重写。

getValue()：方法重写。

Commit（继承KeyValueObject）:

value

Commit()：创建默认commit对象。

Commit(String)：传入message创建commit对象。

Commit(Tree, String)：通过tree对象和commit注释创建commit对象。

record()：方法重写。

generateKey()：方法重写。

getValue()：方法重写。

FileOperation：

HEAD

NULLHash

getAllBranchName()：遍历refs.分支，获取分支名列表。

getCurrentBranchFile()：获取HEAD文件。

getCurrentBranch()：通过HEAD文件，获取当前分支名。

getHeadFile()：获取refs.分支的当前分支文件。

getHeadLogFile()：获取logs.refs.分支的当前分支文件。

getAllLogFile()：获取logs.HEAD文件。

getHead()：通过refs.分支的当前分支文件，获取当前分支commit的key。

getHead(String)：通过refs.分支的指定分支文件，获取指定分支commit的key。

updateKey(String, String)：由key和message更新logs.refs.分支的当前分支文件，logs.HEAD文件，refs.分支的当前分支文件。

Branch:

MASTER

branchName

Branch()：新建默认分支。

Branch(String)：以名称创建分支（不切换分支）。

Branch(String, boolean)：以名称创建分支（切换分支）。

isBranch(String)：判断b1是否为当前分支名。

initBranch(String)：初始化分支（新建分支时）。

switchBranch(String)：切换分支。

renameBranch(String, String)：重命名分支。

printAllBranches()：打印所有分支。

getBranchName()：获取分支名。

Reset:

parentNum(String)：获取当前分支commit的parent个数。

getBefore(int)：获取当前分支commit指定次数的parent。

commitExists(String)：判断commit是否在logs.refs.分支的当前分支文件的内容中。

logReset(int)：回滚n次，修改本地仓库。

logReset(String)：回滚到指定commit，修改本地仓库。

treeReset(int)：回滚n次，修改工作区。

treeReset(String)：回滚到指定commit，修改工作区。

resetHard(int)：指定次数的working tree和repository回滚。

resetHard(String)：指定commit的working tree和repository回滚。

resetMixed(int)：指定次数的repository回滚。

resetMixed(String)：指定commit的repository回滚。

Log：

getCurrentBranchLog()：获取logs.refs.分支的当前分支文件的内容。

getAllLog()：获取logs.HEAD文件的内容。

3.总体测试

commander.parse("--help"); //查看帮助

commander.parse("--init"); //初始化仓库

commander.parse("branch","-n","testing"); //在空分支上新建分支，失败

commander.parse("commit"); //提交commit

commander.parse("commit","-m","commit1"); //（修改文件）提交带注释的commit

commander.parse("branch","-n","testing"); //创建testing分支

commander.parse("branch","-n","testing"); //重复创建testing分支，失败

commander.parse("branch","-n","testing2","-s"); //创建testing2分支并切换

commander.parse("commit","-m","commit2"); //（修改文件）提交带注释的commit

commander.parse("switch","testing"); //切换至testing分支

commander.parse("switch","testing0"); //切换至不存在的分支，失败

commander.parse("commit","-m","commit3"); //（修改文件）提交带注释的commit

commander.parse("reset","-HARD","-t","1"); //回滚一次commit

commander.parse("reset","-HARD","-t","5"); //回滚次数超过当前分支commit次数，失败

commander.parse("reset","-HARD","-c","commitID"); //回滚到指定commit

commander.parse("reset","-HARD","-c","notExist"); //回滚到不存在commit，失败

commander.parse("reset","-t","1"); //回滚一次commit

commander.parse("reset","-c","commitID"); //回滚到指定commit

commander.parse("branch","-a"); //查看所有分支

commander.parse("branch","-c"); //查看当前分支

commander.parse("rename","testing","testing2"); //重命名至已存在分支，失败

commander.parse("rename","testing0","testing2"); //重命名的分支不存在，失败

commander.parse("rename","testing","testing3"); //重命名分支，成功

commander.parse("log","-a"); //查看全部日志

commander.parse("log","-c"); //查看当前分支日志

**三、小组分工**

敖海航：实现本地仓库基本结构，实现生成blob、commit功能，实现分支创建、切换、查看功能，实现命令行交互，做ppt汇报。

刘海云：实现生成tree功能，实现回滚功能，实现分支重命名功能，画项目类图，整理代码注释，汇总项目文档，撰写项目报告。

赵晓浩：

**四、感想与收获**

在知识技能上，主要熟悉了java语言编写程序的方法，掌握了如何写出结构清晰、耦合度低的类以及方法，对文件的遍历和生成等有了更深层次的学习，对于bug的处理也有了更多方法；其次熟悉了git底层实现的原理，对于git指令有了更多了解，并熟悉了利用github进行项目工作的方法。

在团队协作上，明白了团队分工合作的高效性，体会到成员交流带来的思维创新性，并认识到github这样的版本控制系统带给小组项目的便利性。