

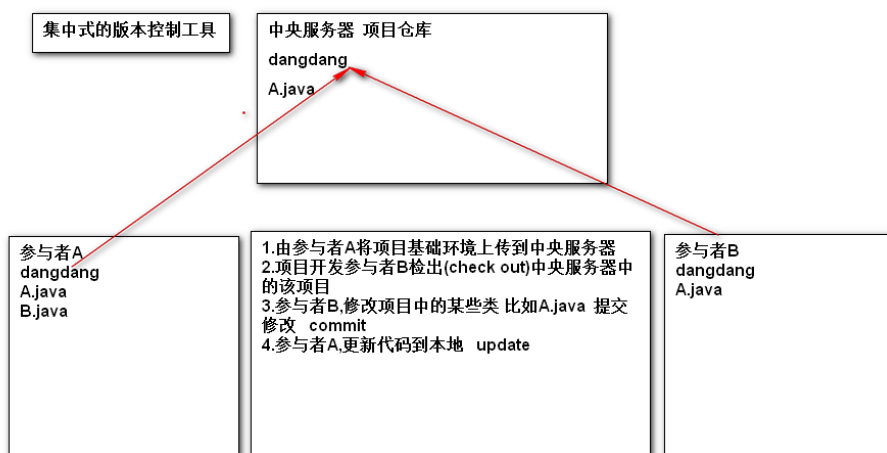
# git

## 1.版本控制解决现有项目中的哪些问题?

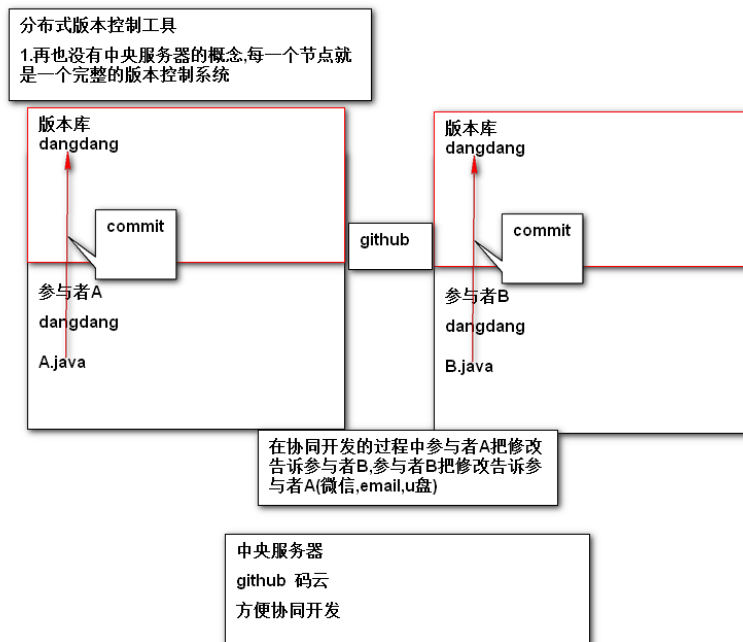
- 1.项目安全性太低
- 2.项目很难进行协同开发
- 3.项目无端报错
- 4.项目版本混乱

## 2.版本控制工具的分类

### 1.集中式版本控制工具      svn



### 2.分布式的版本控制工具      git



### 3.git的概念

git是一个分布式的版本控制及协同开发工具.

### 4.git的环境搭建

1.git安装,安装git bash

2.自报家门

```
1 git config --global user.name "htf"
2 git config --global user.email "xxx@.com"
```

### 5.git的使用

```
1 1.创建版本库 在项目根路径下会创建.git文件夹
2 git init
3 2.git管理一个修改需要几步
4 git add 文件名
5 git commit -m ""
6 3.git的状态
7 git status 红色 工作区已经修改,但是还未提交内容
8 绿色 工作区中的修改,加入到暂存区
9 working tree clean 工作区和版本库是一致的
10 4.版本回退
11 只能做回退
12 git reset --hard HEAD^
13 git reset --hard HEAD~100
```

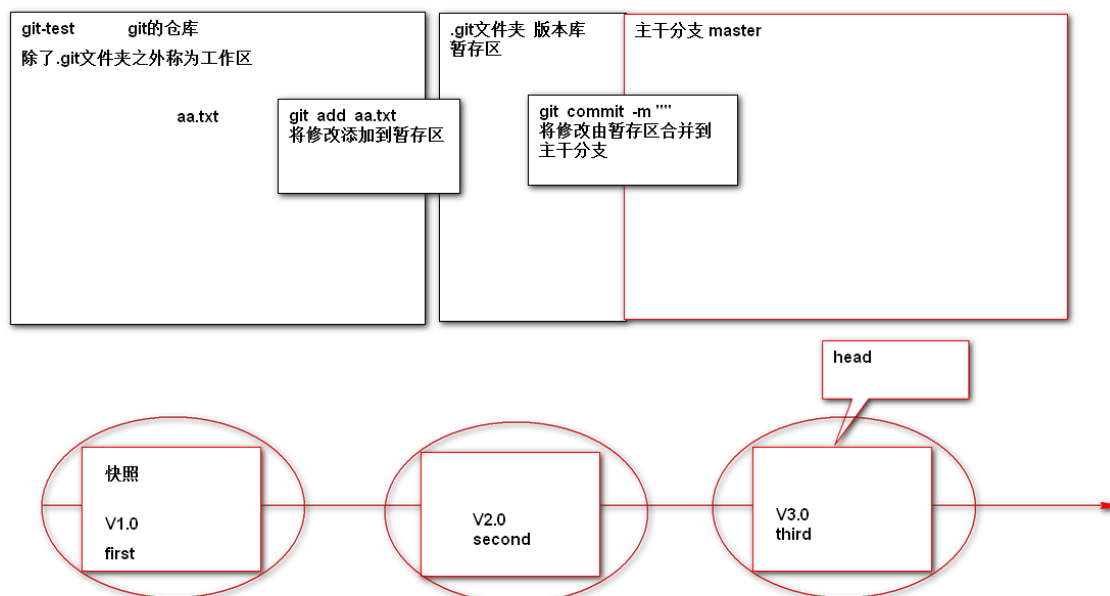
```

14 通用
15  git reset --hard 3628164
16
17 5. 日志
18  git log    打印当前提交之前的日志
19  git log --pretty=oneline
20  git reflog --pretty=oneline  打印所有的日志
21
22 6. 撤销修改
23  git checkout -- readme.txt    撤销工作区中的修改
24  git reset HEAD readme.txt     撤销暂存区中的修改撤销到工作区
25

```

## 6.git仓库的原理

git的相关概念



## 7.git的实战

### 7.1.生成 SSH key并添加到github信任列表

1. 创建 SSH Key : `ssh-keygen -t rsa -C "youremail@example.com"`

2. 添加github的信任列表


### 7.2.通过github创建仓库完成git管理

1. github上创建仓库

# Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

 28899121390 ▾

Repository name

/ 140-git ✓



Great repository names are short and memorable. Need inspiration? How about [effective-octo-broccoli](#).

Description (optional)



Public

Anyone can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

2

Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None ▾

Add a license: None ▾



2.克隆中央服务器上的项目到本地

```
git clone git@github.com:28899121390/git-140.git
```

3.提交一个修改

```
git add
```

```
git commit
```

推送修改到中央服务器

```
git push origin
```

更新修改到本地

```
git pull origin
```

## 7.3.将本地的项目交给git管理并且分享到中央服务器

1.初始化仓库 `git init`

2.配置忽略文件 `.gitignore` 注意放置到项目的根部目录下

```
1 /target/
2 !.mvn/wrapper/maven-wrapper.jar
3
4 ### STS ###
5 .apt_generated
```

```
6 .classpath
7 .factorypath
8 .project
9 .settings
10 .springBeans
11 .sts4-cache
12
13 ### IntelliJ IDEA ###
14 .idea
15 *.iws
16 *.iml
17 *.ipr
18
19 ### NetBeans ###
20 /nbproject/private/
21 /build/
22 /nbbuild/
23 /dist/
24 /nbdist/
25 /.nb-gradle/
26
27
```

3.添加到暂存区 git add

4.添加到主干分支 git commit -m

5.在github上创建项目仓库

## Create a new repository


A repository contains all the files for your project, including the revision history.

Owner

28899121390 ▾

/

Repository name

spring-boot-common-mapper 

项目名和本地一致

Great repository names are short and memorable. Need inspiration? How about [reimagined-guacamole](#).

Description (optional)

☒ Public

Anyone can see this repository. You choose who can commit.

☐ Private


You choose who can see and commit to this repository.

☐ Initialize this repository with a README

不能勾

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None ▾

Add a license: None ▾ 

Create repository

### 6.建立本地仓库和中央服务器仓库的连接

git remote -v 查看当前项目是否含有远程仓库地址

git remote remove test 删除远程仓库连接

git remote add origin [git@github.com:28899121390/spring-boot-common-mapper.git](https://github.com/28899121390/spring-boot-common-mapper.git)

### 7.推送本地修改

git push -u origin master

### 7.拉取远程仓库的修改

git pull origin master

## 8.git的分支

### 1.分支的概念

git默认只有一条分支叫做master,是时间线的概念,随着每一次提交,时间线发生延长,所谓的分支就是和master一样的存在,新创建的分支的版本和master所在提交点一致,日后head指针指向那个分支,用户就处于那个分支

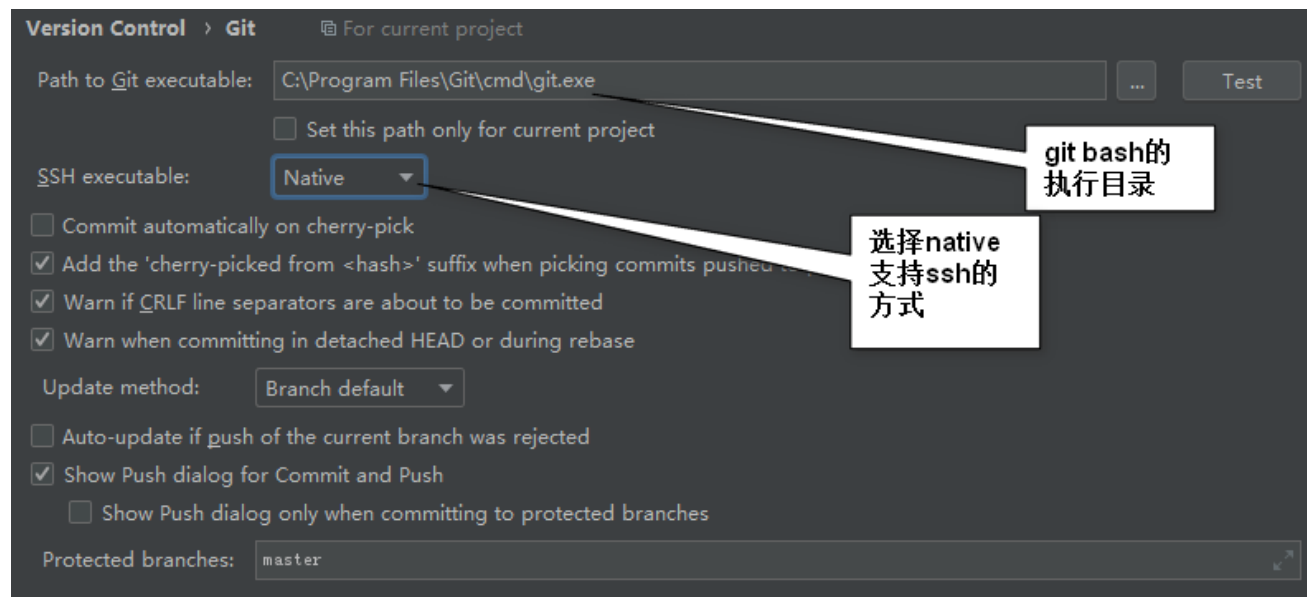
### 2.分支的相关指令

- 1 切换并创建分支
- 2 `git checkout -b dev`
- 3 创建分支
- 4 `git branch dev`
- 5 切换分支
- 6 `git checkout dev`
- 7 查看分支列表
- 8 `git branch`
- 9 合并分支
- 10 `git merge dev`
- 11 删除分支
- 12 `git branch -d dev`

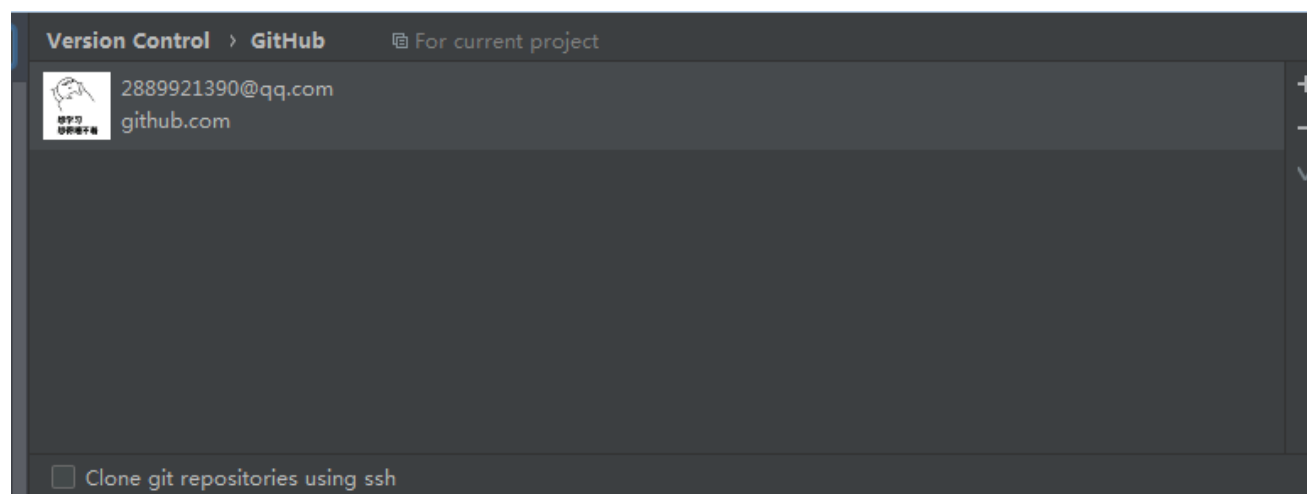
## 9.git集成idea

### 1.idea集成Git

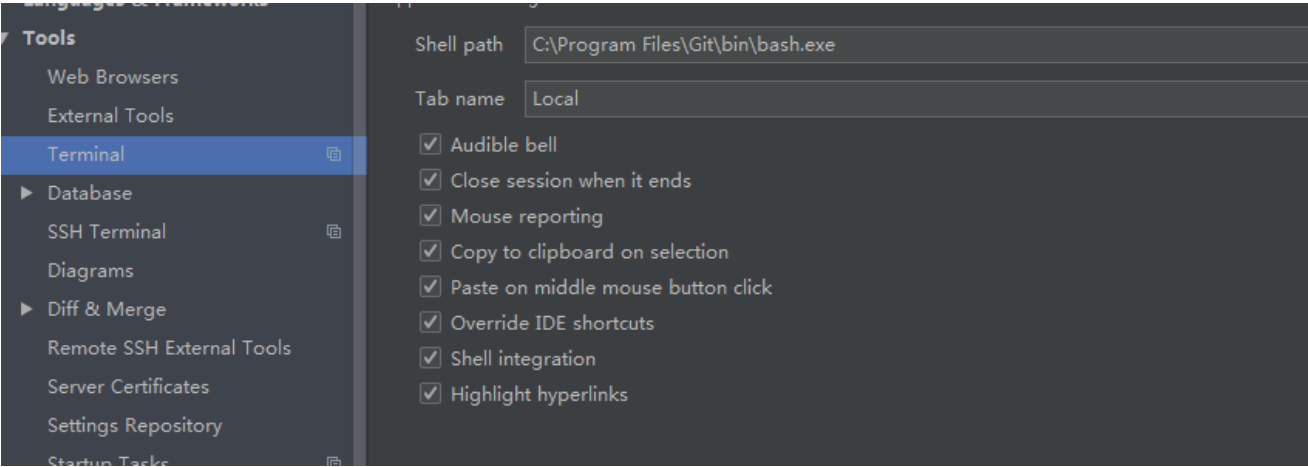
#### 1.idea集成git bash



#### 2.需要填写github的账号密码

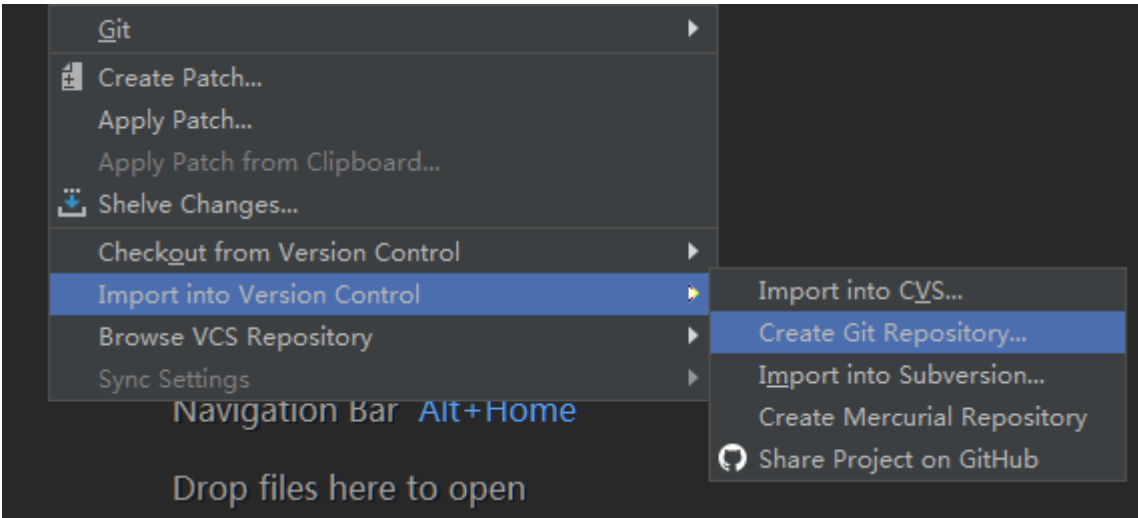


3.替换idea的终端,替换成git的终端



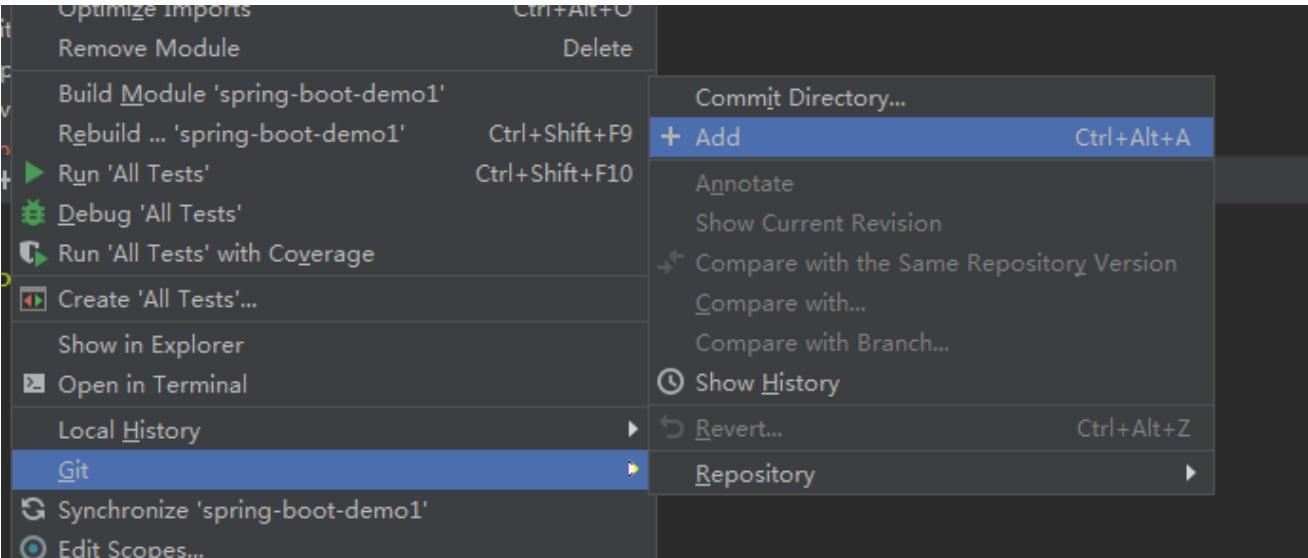
2.idea中git的实战以及相关的快捷键

1.创建git仓库



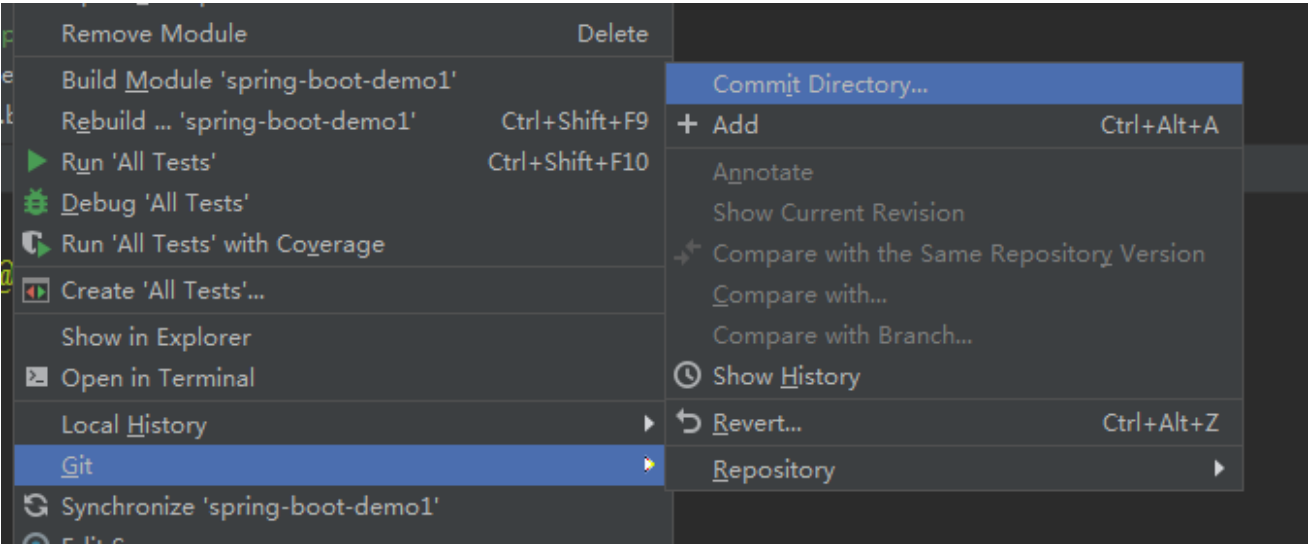
2.添加忽略文件 .gitingore 注意:文件必须放置在项目的根目录下

3.添加修该到暂存区 idea默认执行add



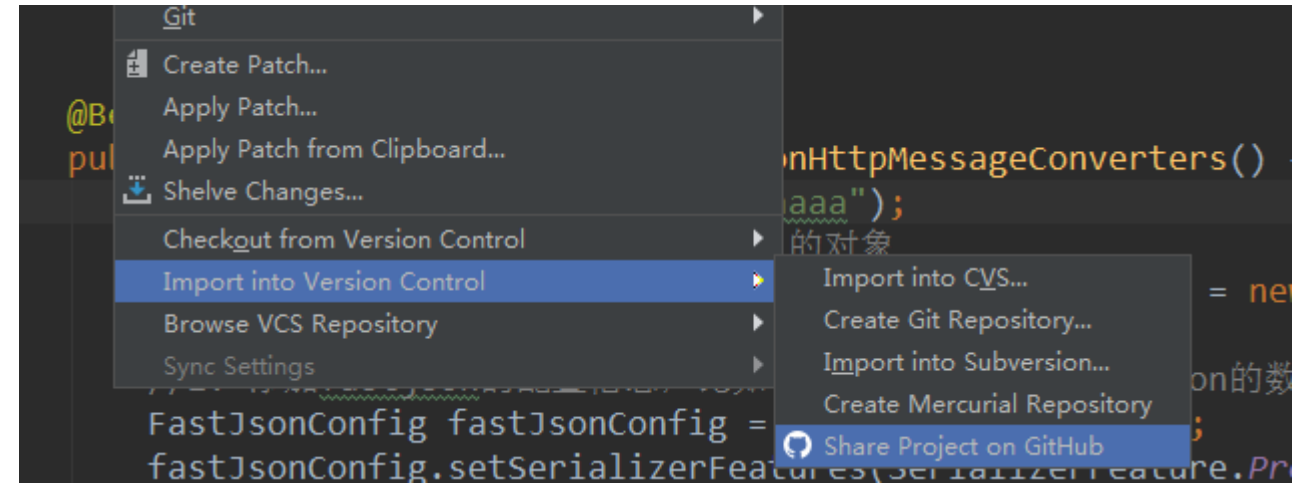


4.添加到主干分支



- 1    ctrl+k    提交
- 2    ctrl+shift+k    提交并推送
- 3    ctrl+t    更新
- 4    alt+9    查看git控制台

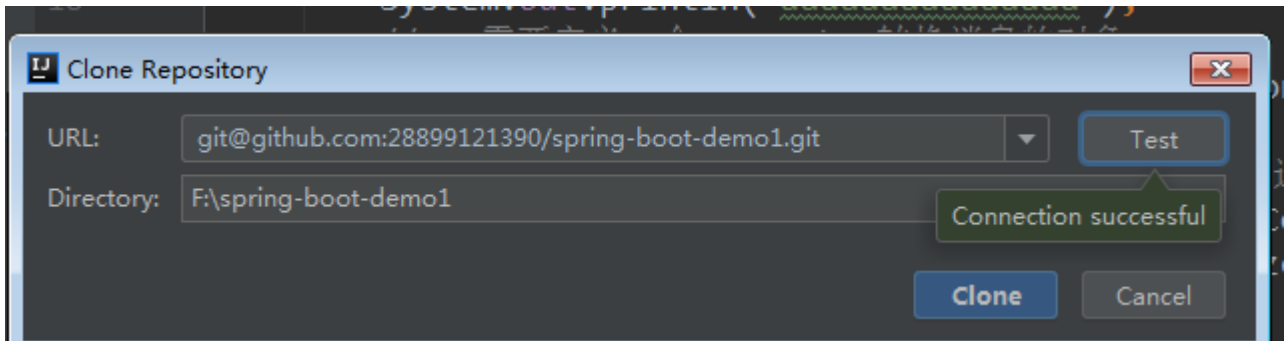
5.分享到github



6.协同开发

3.如何用idea克隆并打开项目

1.指定url以及路径



2.打开项目

#### 4.git的冲突

1.当两个开发者同时操作一个类的时候会出先冲突 代码没有交集

选择遵守你的,或者他的或者两者都要.

2.代码有交集

和相关参与者进行交流.

3.遇到冲突先备份