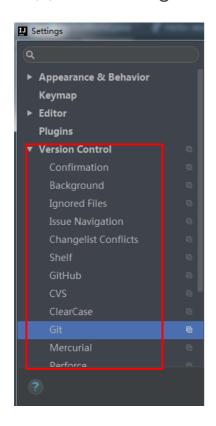
# 一、基本入门

# 1.IntelliJ-IDEA预装的版本控制介绍

我们来看IntelliJ-IDEA的版本控制设置区域

打开File>Settings>Version Control



### 可以看到里面有git插件

这让很多人认为 IntelliJ IDEA 自带了 SVN 或是 Git 等版本控制工具,认为只要安装了 IntelliJ IDEA 就可以完全使用版本控制应有的功能。这完全是一种错误的解读, IntelliJ IDEA 是自带对这些版本控制工具的支持插件,但是该装什么版本控制客户端还是要照样装

上面可以看到里面还自带了github插件,这是因为目前太多人使用

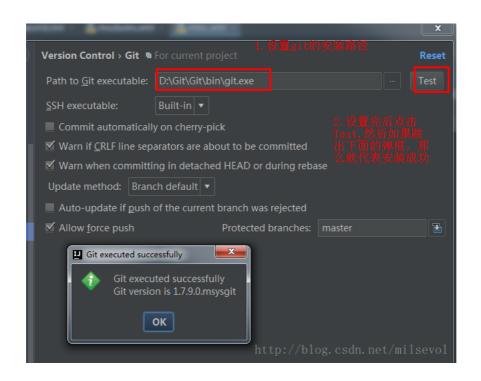
Github 进行协同或是项目版本管理

# 2.git.exe和Github的配置

# 1.配置git.exe

先确保电脑上已经安装了Git和TortoiseGit

接着再配置git客户端



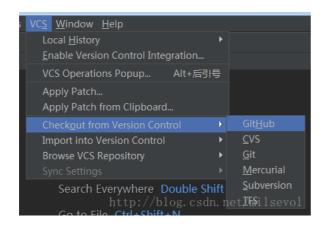
## 2.配置Github

下面是选择账号和密码登录

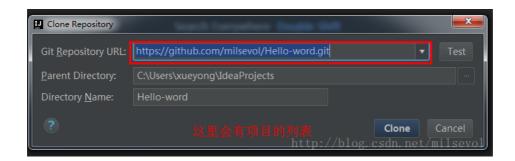


## 3.clone项目从Github上

接着可以将GitHub上的项目checkout下来



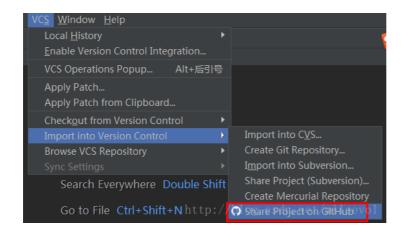
如果GitHub上有多个项目的话,那么可以选择其中的一个



点击clone后, IntelliJ-IDEA就会开始clone项目

## 4.将本地项目上传到Github上

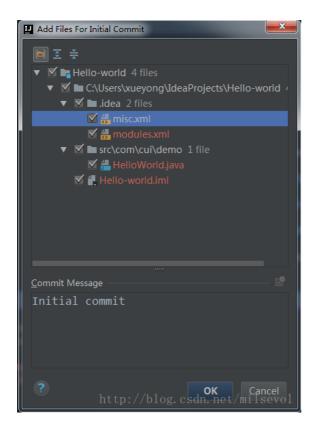
#### 选择Import into Version Control



#### 然后填写相关的信息



这时候会提示你选择上传的文件



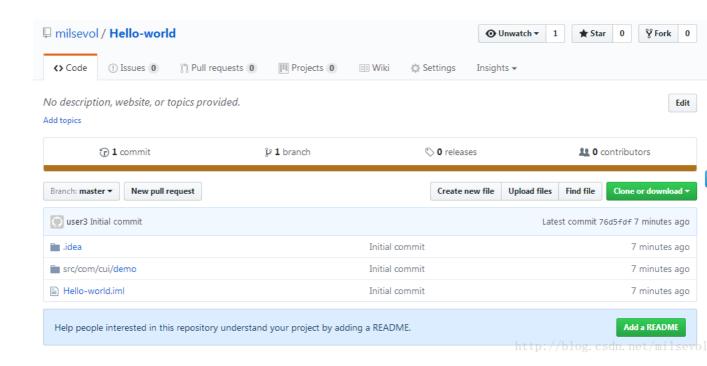
### 仓库创建成功后,会出现如下的提示



## 再看GitHub下,多了一个项目仓库

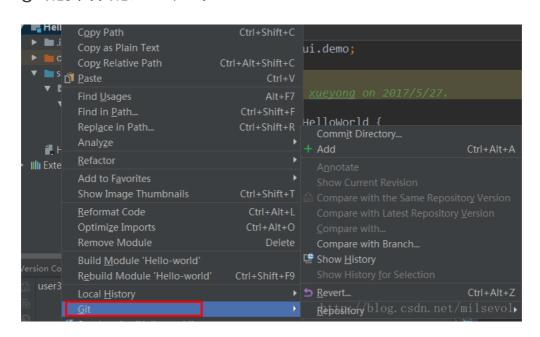


点击进去,项目的相关结构如下



# 二、版本控制的主要操作按钮

### git的操作的入口如下:



也可以在工具栏中进行操作

注意这五个按钮

```
Refactor Build Run Iools VCS Window Help

WHELD WING HelloWorld NAME HelloWorld NAME HelloWorld NAME HelloWorld NAME HELD WIND HELD NAME HELD NAME
```

#### 五个按钮的功能如下:

第一个按钮: Update Project 更新项目。

第二个按钮: Commit changes 提交项目上所有变化文件。

点击这个按钮不会立马提交所有文件,而是先弹出一个被修改文件的一个汇总框,具体操作下面会有图片进行专门介绍。

第三个按钮: Compare with the Same Repository Version 当前文件与服务器上该文件通版本的内容进行比较。如果当前编辑的文件没有修改,则是灰色不可点击。

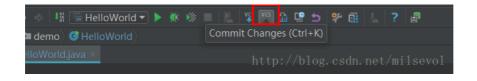
第四个按钮: Show history 显示当前文件的历史记录。

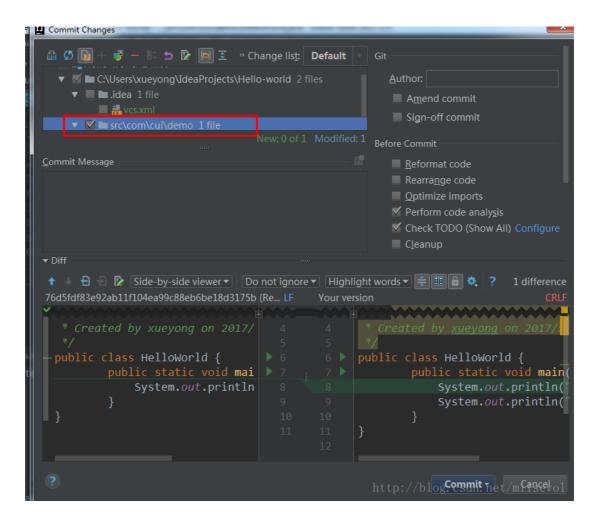
第五个按钮:Revert 还原当前被修改的文件到未被修改的版

本状态下。如果当前编辑的文件没有修改,则是灰色不可点击。

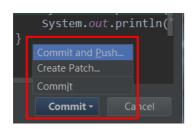
具体作用如下,如下面添加了一行,如果不满意,可以还原

### 但是如果想提交的话,可以选择这个





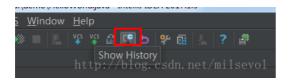
## 填写相关的信息后,可以直接选择commit和push



点击push



#### 提交完毕后,我们可以查看相关的记录



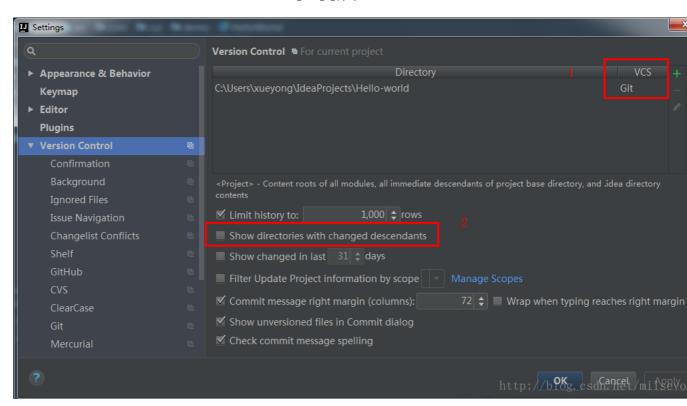
## 可以看到如下的记录



这时候GitHub上项目就会出现相关的信息了

# 三、版本控制的常用设置

## 1.Version Control面板



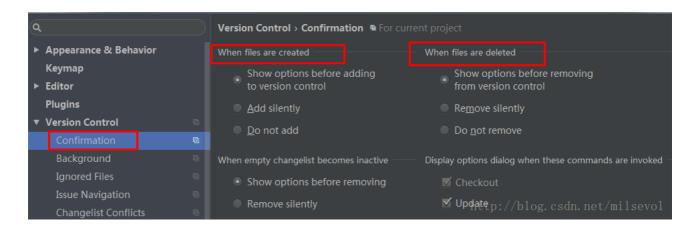
- 1. 代表这个项目的版本控制是GIT
- 2. Show directories with changed descendants 表示子目

录有文件被修改了,则该文件的所有上层目录都显示版本

控制被修改的颜色(建议勾选)

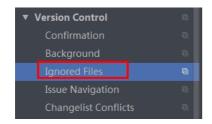
## 2.Confimation确认信息面板

添加新文件和删除文件的提示



# 3.Ignored Files面板

这个是设置不添加到版本控制的文件



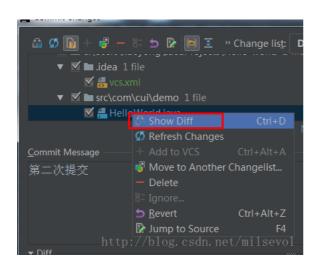
## 4.commit文件的面板介绍

commit文件的时候会出现如下的面板

```
Commit Changes
            🎳 — 🖺 🖢 🖟 📵 🖫 » Change list: Default
  🛍 Ø 📭
      ▼ ☑ 🖿 .idea 1 file
          🗹 🖶 vcs.xml
      ▼ M src\com\cui\demo 1 file
                                                          Sign-off commit
          🗹 📇 HelloWorld.java
                                       New: 1 Modified: 1 Before Commit
                                                          Reformat code
                                                          Rearrange code
 第二次提交
                                                          Optimize imports
                                                          Cleanup
  🕇 🤚 🖟 Side-by-side viewer 🔻 Do not ignore 🔻 Highlight words 🔻 🖫 🔒 👯 📍 1 difference
           public static void mai > 7
                                                           public static void main
                System.out.println
                                                                 System.out.println(
                                                                 System.out.println(
                                                      http://blog.csdn.net/mflsevol
```

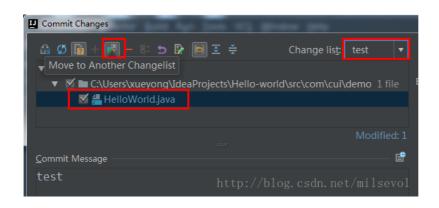
### 1.show Diff

show Diff可以比较本地跟服务器文件的不同

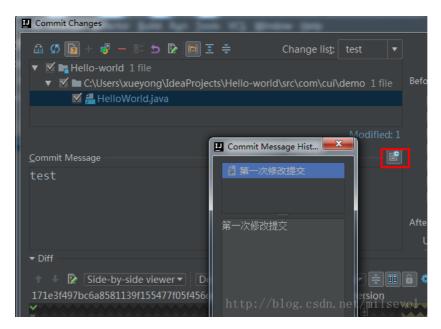


## 2. Move to Another ChangeList

这个选项可以将改动的文件分门别类的放置在一个文件夹中,等修改完毕,可以一起提交(一般用在分模块开发上,也就是开发完一个模块时,可以将这个模块上改动的代码设置到一个文件夹中,然后再一起祈祷)



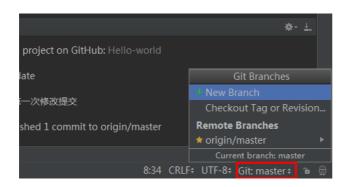
## 3.查看提交信息



# 四、项目分支设置的入口

如果使用Git版本控制的话,可以看到右下角上有相关的控

#### 制入口



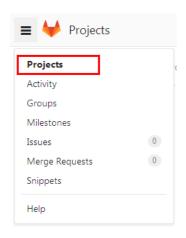
# 五、Gitlab的使用

Gitlab跟Github类似,都是代码托管的网站,最大的不同是Gitlab创建的项目可以免费私有的,不必像Github那样收费,而且Gitlab还可以搭建自己的私服。所以开源项目一般都是放置在Github,个人私有项目可以放置在公网的Gitlab上,而公司私有的项目可以放置在自己搭建的Gitlab上。

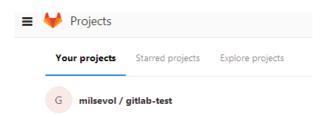
## 1.创建项目

## 2.对项目进行操作

项目创建完后,可以点击左上角进行查看



## 点击Project,可以看到项目的列表



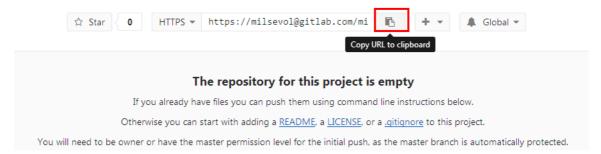
### 可以为项目添加一个README.md



接着可以将指定的项目clone下来,下面是通过用户名和密码的方式clone下来

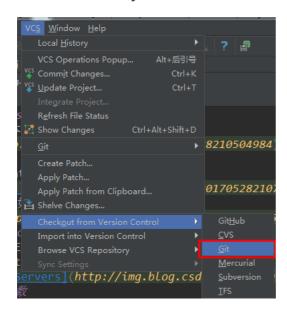


#### gitlab-test •

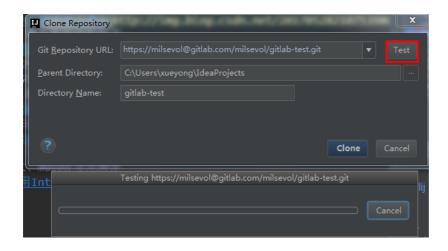


#### 在

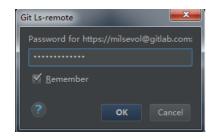
## 接着在IntellIj-IDEA上使用Git将其Clone下来



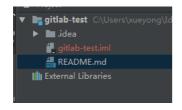
如上图, check out项目的时候因为不是Github, 所以不再选择Github, 而是选择普通的Git来将项目clone下来



最后输入密码,再将其clone下来



### clone下来后,项目结构如下:

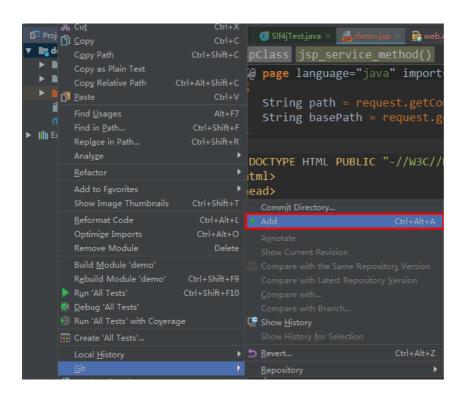


这时候就可以进行正常的add、commit、push操作了

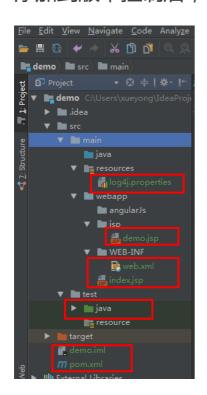
# 3.将项目发布到gitlab上

先在本地创建一个项目

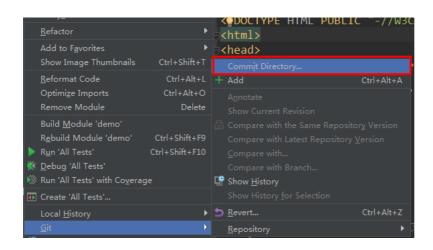
先将项目添加到版本控制中



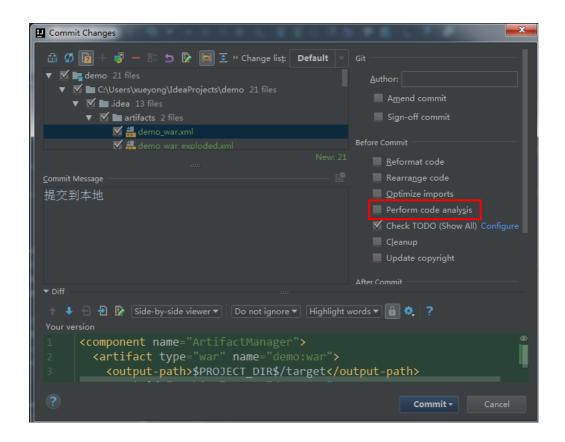
### 添加到版本控制后,我们可以看到文件都变成绿色了



接着往本地提交项目



#### 填写提交的相关信息

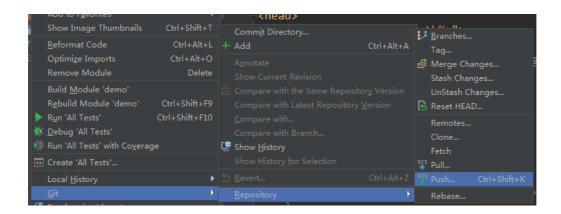


注意perform code analysis 就不要勾选了,这个选项会

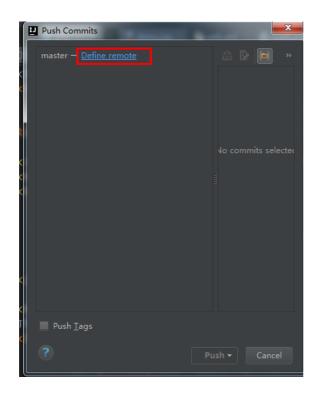
自动检查代码,会很慢

接着将项目push到服务器中

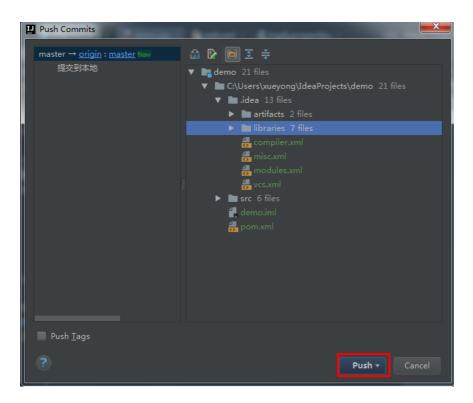
下面点击push



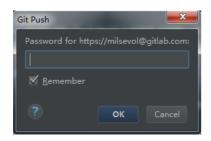
### 接下来需要定义远程的服务



此时需要再弹出框中设置url,我们先在gitlab新建一个project,这样的目的是为了获得相关的url填写url后,再点击push



## 此时还需要填写gitlab上的密码



push成功后,会出现这样的提示,这是成功的提示:



再查看gitlab上的项目信息,可以看到提交的代码,具体如下图

