[Top](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/DEVOPS/DAY04/CASE/01/index.html" \l "page_top_case)

# NSD Devops DAY04

1. [案例1：配置git](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/DEVOPS/DAY04/CASE/01/index.html" \l "case1)
2. [案例2：git本地操作](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/DEVOPS/DAY04/CASE/01/index.html" \l "case2)
3. [案例3：使用自建gitlab服务器](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/DEVOPS/DAY04/CASE/01/index.html" \l "case3)

## 1 案例1：配置git

### 1.1 问题

1. 安装git版本控制软件
2. 设置用户信息，如用户名、email等
3. 设置默认编辑器为vim
4. 查看用户配置

### 1.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：安装git版本控制软件

1. [root@localhost ~]# yum install -y git
2. 已安装:
3. git.x86\_64 0:1.8.3.1-11.el7
4. 作为依赖被安装:
5. perl-Error.noarch 1:0.17020-2.el7 perl-Git.noarch 0:1.8.3.1-11.el7
6. perl-TermReadKey.x86\_64 0:2.30-20.el7
7. 完毕！
8. [root@localhost ~]# git –version        #查看版本
9. git version 1.8.3.1

步骤二：设置用户信息

Git 提供了一个叫做 git config 的工具，专门用来配置或读取相应的工作环境变量。

1. [root@localhost ~]# git config --global user.name "Mr.Zhang"
2. [root@localhost ~]# git config --global user.email zhangzg@tedu.cn

步骤三：设置默认编译器为vim

1. [root@localhost ~]# git config --global core.editor vim

步骤四：查看配置

1. [root@localhost ~]# git config --list
2. user.name=Mr.Zhang
3. user.email=zhangzg@tedu.cn
4. core.editor=vim

## 2 案例2：git本地操作

### 2.1 问题

1. 创建devops目录
2. 为devops创建git仓库
3. 新建文件hello.py，并将文件初始化到仓库中
4. 修改hello.py并将其更新到仓库
5. 从他库中删除hello.py

### 2.2 方案

Git 使用 git init 命令来初始化一个 Git 仓库，Git 的很多命令都需要在 Git 的仓库中运行，所以git init是使用 Git 的第一个命令。添加文件第一步使用git add是将文件添加进暂存区，第二部git commit提交更改，实际上将暂存区的所有内容提交到仓库。

### 2.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：创建目录初始化

此时创建了一个空仓库，可以发现当前目录下有一个隐藏的目录.git，此目录为Git来跟踪管理版本库，建议不要修改内部文件，以免Git仓库遭到破坏。

1. [root@localhost ~]# mkdir devops
2. [root@localhost ~]# cd devops/
3. [root@localhost devops]# git init        #通过命令把目录变成Git可以管理的仓库
4. 初始化空的 Git 版本库于 /root/devops/.git/
5. [root@localhost devops]# git init devops
6. 初始化空的 Git 版本库于 /root/devops/devops/.git/
7. [root@localhost devops]# ls -a
8. . .. devops .git

步骤二：新建文件hello.py，并将文件初始化到仓库中

1)添加指定文件hello.py到暂存区

1. [root@localhost devops]# echo 'print("hello world!")' > hello.py
2. [root@localhost devops]# git add hello.py        #将文件添加到暂存区
3. [root@localhost devops]# git status        #查看状态
4. # 位于分支 master
5. #
6. # 初始提交
7. #
8. # 要提交的变更：
9. # （使用 "git rm --cached <file>..." 撤出暂存区）
10. #
11. #    新文件： hello.py

2)将暂存区文件初始化到仓库中

1. [root@localhost devops]# git commit -m "初始化仓库"     #把暂存区所有内容提交到分支
2. [master 8e6e22a] 初始化仓库
3. 1 file changed, 1 insertion(+)
4. create mode 100644 hello.py
5. [root@localhost devops]# git status
6. # 位于分支 master
7. 无文件要提交，干净的工作区

步骤三：修改hello.py并将其更新到仓库

1. [root@localhost devops]# echo 'print("done.")' >> hello.py
2. [root@localhost devops]# git commit -am "向hello.py添加新行"
3. [master 1ca03d5] 向hello.py添加新行
4. 1 file changed, 1 insertion(+)

步骤四：从库中删除hello.py文件

要从 Git 中移除某个文件，就必须要从已跟踪文件清单中移除

1. [root@localhost devops]# git ls-files         #查看版本库中文件
2. hello.py
3. niha.py
4. [root@localhost devops]# git rm hello.py
5. rm 'hello.py'
6. [root@localhost devops]# git commit -m '删除hello.py'
7. [master a37ff34] 删除hello.py
8. 1 file changed, 2 deletions(-)
9. delete mode 100644 hello.py

## 3 案例3：使用自建gitlab服务器

### 3.1 问题

1. 通过docker搭建gitlab服务器
2. 新建群组devops
3. 新建项目core\_py
4. 新建用户，他/她在devops组中是主程序员
5. 新用户上传版本库到gitlab
6. 熟悉git远程操作方法

### 3.2 步骤

实现此案例需要按照如下步骤进行。

步骤一：通过docker搭建gitlab服务器

1)从ftp://172.40.50.116/pub/docker/images/处获取gitlab\_zh.tar镜像文件，导入中文版gitlab镜像

1. [root@localhost devops]# docker load < /path/to/gitlab\_zh.tar
2. a94e0d5a7c40: Loading layer [==================================================>] 116.5MB/116.5MB
3. 88888b9b1b5b: Loading layer [==================================================>] 15.87kB/15.87kB
4. 52f389ea437e: Loading layer [==================================================>] 14.85kB/14.85kB
5. 52a7ea2bb533: Loading layer [==================================================>] 5.632kB/5.632kB
6. db584c622b50: Loading layer [==================================================>] 3.072kB/3.072kB
7. 62786ff6a243: Loading layer [==================================================>] 75.85MB/75.85MB
8. 71bc04f4b7c7: Loading layer [==================================================>] 2.048kB/2.048kB
9. 26e083d332d8: Loading layer [==================================================>] 2.048kB/2.048kB
10. 2c02e58e96b8: Loading layer [==================================================>] 2.048kB/2.048kB
11. 589c7a23de2a: Loading layer [==================================================>] 15.87kB/15.87kB
12. 44474d2cdcd1: Loading layer [==================================================>] 1.359GB/1.359GB
13. 41c94e16b901: Loading layer [==================================================>] 16.78MB/16.78MB
14. 04cafa6a1534: Loading layer [==================================================>] 160MB/160MB
15. Loaded image: gitlab\_zh:latest

2)将物理主机ssh端口改为2022后，输入如下命令起动容器：

1. [root@localhost devops]# docker run -d -h gitlab --name gitlab -p 443:443 -p 80:80 -p 22:22 --restart always -v /srv/gitlab/config:/etc/gitlab -v /srv/gitlab/logs:/var/log/gitlab -v /srv/gitlab/data gitlab\_zh:latest
2. b9dc65e0def51a4d09d2a597b2b929490e972a34f3de993439d2f7cc22039b77
3. 此时端口成功启动

步骤二：新建群组

1)在浏览器地址栏中输入启动容器ip地址即可登录GitLab的界面，第一次登录使用的用户名为root，首次登录会强制用户修改密码。密码修改成功后，输入新密码进行登录，如图-6所示：

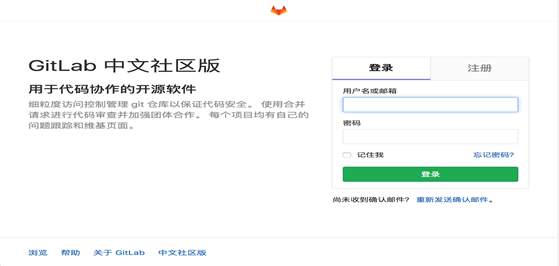


图-1

2)进入网站后点击菜单栏-工具图标打开管理区域，创建群组，使用群组管理项目和成员，如图-7、图-8所示：

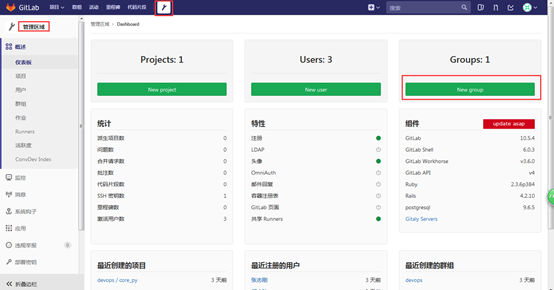


图-2

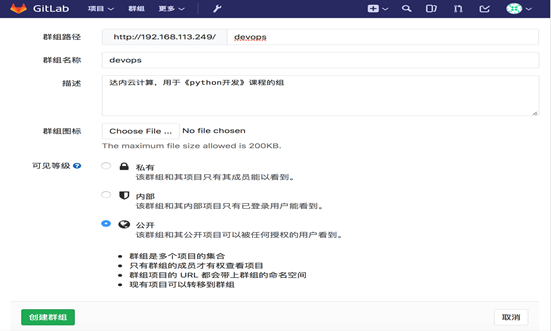


图-3

步骤三：在Gitlab主页中新建一个项目

显示如图-9所示：

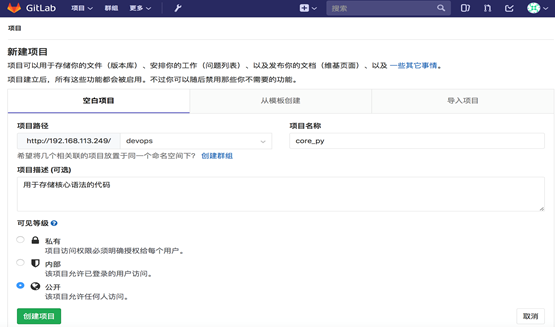


图-4

步骤四：在Gitlab主页中新建一个用户

1)在Gitlab主页中新建一个用户，如图-10所示：

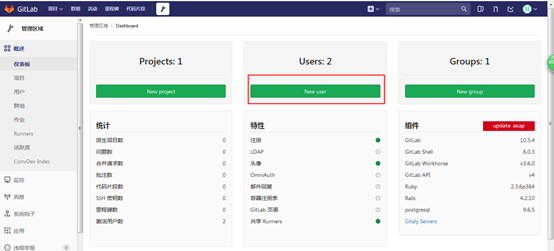


图-5

2)设置账号基本信息，其他均可为默认，如图-11所示：

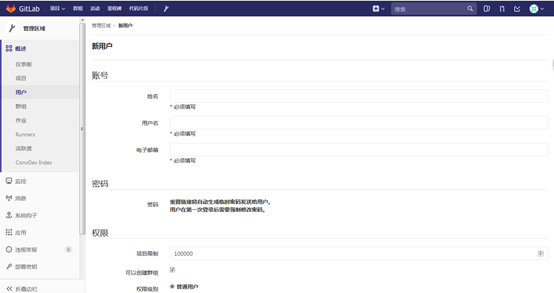


图-6

3)创建用户后，再次编辑可设置密码，如图-12所示：



图-7

4)root用户将新用户加入组中，点击devops进入群组，设置管理权限，在群组中添加成员并设置新成员为“主程序员”，如图-13、图-14、图-15所示：

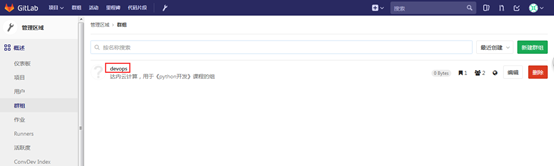


图-8

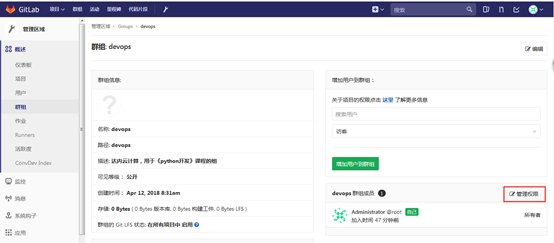


图-9

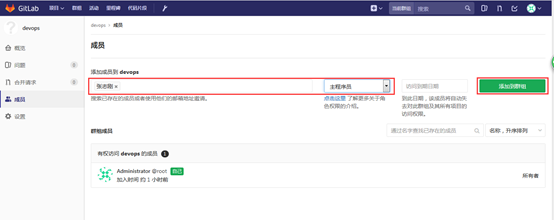


图-10

5)在终端中输入ssh-keygen命令，一路enter，可以生成缺省的rsa方式的sshkey，将/root/.ssh/id\_rsa.pub中生成的rsa公钥内容拷到gitlab中，如图-16所示，进入设置页面对ssh进行配置：

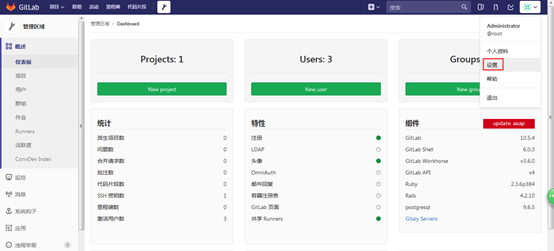


图-11

1. SSH是一种安全协议，在你的电脑与GitLab服务器进行通信时，我们使用SSH密钥（SSH Keys）认证的方式来保证通信安全。你可以在网络上搜索到关于SSH密钥的更多介绍；下面我们重点讲解如何创建 SSH密钥，并将密钥中的公钥添加到GitLab，以便我们通过SSH协议来访问Git仓库。
2. 显示如图-17所示：

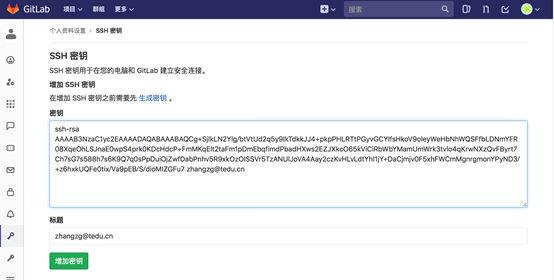


图-12

步骤五：简单远程操作及新用户上传版本库到gitlab

1)克隆远程库代码到本地

1. # git clone git@192.168.113.249:devops/core\_py.git

2)创建一个文件

1. # cd /root/whsir/whsir
2. # echo "Hello" > hello.py

3)将文件添加到仓库

1. # git add hello.py

4)提交文件到仓库，输出信息如图-18所示：

1. # git commit -m "hello文件"

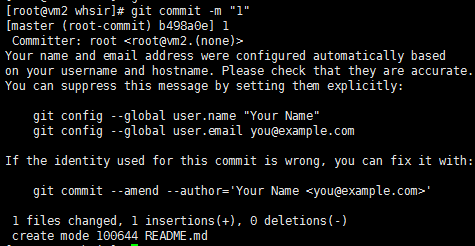


图-13

5)关联远程库

1. # git remote add origin git@192.168.113.249:root/whsir.git

6)最后推送到gitlab上，输出信息如图-19所示：

1. # git push origin master

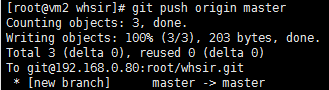


图-14