1、安装

# yum install -y git

2、配置作者信息

[root@node1 ~]# git config --global user.name "changzhihong"

[root@node1 ~]# git config --global user.email ["changzhihong@163.com"](mailto:\"changzhihong@163.com\")

//设置用户名和邮箱

[root@node1 ~]# git config --global core.editor vim

//设置默认编辑器为vim

[root@node1 ~]# git config --list

[root@node1 ~]# cat .gitconfig //查看用户配置

使用git

1、初始化

[root@node1 ~]# mkdir devops

[root@node1 ~]# cd devops

[root@node1 devops]# git init

或

[root@node1 ~]# git init devops

//等同于上面的三条命令

[root@node1 devops]# ls -a

2、添加跟踪文件

[root@node1 devops]# echo 'hello world' > hello.txt

[root@node1 devops]# git add .

3、查看暂存区

[root@node1 devops]# git status

[root@node1 devops]# git status -s

4、确认文件到版本库

[root@node1 devops]# git commit -m "add hello.txt"

[root@node1 devops]# git commit -am "add hello.txt" //像hello.txt中添加新行

[root@node1 devops]# git status

5、删除版本库中文件

[root@node1 devops]# git ls-files

[root@node1 devops]# git rm hello.txt

[root@node1 devops]# git commit -m "rm hello.txt"

6、把不想放到版本库中的文件写到.gitignore中

[root@node1 devops]# vim .gitignore

\*.swp

\*.pyc

搭建gitlab服务器，统一管理软件项目

1、启动一台虚拟机，内存至少改为4G

2、在虚拟机上配置docker

[root@node1 docker]# yum install -y docker\_pkgs/\*rpm

[root@node1 docker]# systemctl start docker

[root@node1 docker]# systemctl enable docker

[root@node1 docker]# docker load < images/gitlab\_zh.tar

3、因为，为了方便，gitlab需要占用22端口，将node1的ssh端口改为2222

[root@node1 ~]# vim /etc/ssh/sshd\_config

Port 2222

[root@node1 ~]# systemctl restart sshd

[root@room8pc16 ~]# ssh -p2222 192.168.4.1

[root@node1 ~]# docker run -d -h gitlab --name gitlab -p 80:80 -p 443:443 -p22:22 --restart always -v /srv/gitlab/config:/etc/gitlab -v /srv/gitlab/logs:/var/log/gitlab -v /srv/gitlab/data gitlab\_zh

gitlab用户将代码上传到服务器

1、使用ssh方式上传代码，生成密钥

[root@localhost ~]# ssh-keygen -t rsa -C "zhangzhg@tedu.cn" -b 4096

1. 将公钥~/.ssh/id\_rsa.pub内容复制粘贴到gitlab用户ssh密钥中

3.全局配置（前面做过就不用做了）

#git config --global user.name "常志红"

#git config --global user.email ["changzhihong@163.com"](mailto:\"changzhihong@163.com\")

1. 创建新的版本库

#git clone http://gitlab/devops/core\_py.git

#cd core\_py

#touch README.md

#git add README.md

#git commit -m "add README"

#git push -u origin master

5.将已存在版本库的项目上传

[root@localhost devops]# git remote rename origin old-origin

如果出现以下以下错误，可以忽略

error: 不能重命名配置小节 'remote.origin' 到 'remote.old-origin'

[root@localhost devops]# git remote add origin git@192.168.4.1:devops/core\_py.git

6、向远程服务器推送代码文件

[root@localhost devops]# git push -u origin --all

[root@localhost devops]# git push -u origin --tags

7、以后再有新文件执行以下操作即可

[root@localhost devops]# cp /etc/passwd .

[root@localhost devops]# git add .

[root@localhost devops]# git commit -m "add new file passwd"

[root@localhost devops]# git push

8、其他人下载

[root@room8pc16 demo]# git clone http://192.168.4.1/devops/core\_py.git

如果已经下载，可以进入到目录中，执行

[root@room8pc16 demo]# cd core\_py/

[root@room8pc16 core\_py]# git pull

二．python

1.Python语法结构

1)语句块缩进

python代码块通过缩进对齐表达代码逻辑而不是使用大括号

缩进表达一个语句属于哪个代码块

缩进风格

– 1或2:可能不够,很难确定代码语句属于哪个块

– 8至10:可能太多,如果代码内嵌的层次太多,就会使

得代码很难阅读

– 4个空格:非常流行, 范·罗萨姆支持的风格

缩进相同的一组语句构成一个代码块,称之代码组

首行以关键字开始,以冒号 : 结束,该行之后的一行或多行代码构成代码组

如果代码组只有一行,可以将其直接写在冒号后面,但是这样的写法可读性差,不推荐

1. 注释及续行

注释可以在一行的任何地方开始,解释器会忽略掉该行#之后的所有内容

一行过长的语句可以使用反斜杠\分解成几行

分号 ; 允许你将多个语句写在同一行上

但是些语句不能在这行开始一个新的代码块

1. 输出语句

# print('Hello World!') # 字符串必须放到引号中

# print('Hello', 'World', '!', 123) # 数字不用放到引号中

# print('Hello', 'World', '!', 123, sep='\*\*\*') # 各项间用\*\*\*分开

# print('Hello' + 'World!') # 字符串拼接

# print('Hello', 'World', '!', 123, end='') # 不打印回车

1. 输入语句

#user = input('username: ')

#print(user)

#a = input('number: ') # input读入的数据都是字符类型

# a + 5 # 错误，不能把数字和字符进行加法运算

#print(int(a) + 5) # 将字符转换成数字，再和5进行加法运算

#print(a + str(5)) # 将数字5转换成字符，再和字符串拼接

1. 变量定义

第一个字符只能是大小写字母或下划线

后续字符只能是大小写字母或数字或下划线区分大小写

a = 10 + 5 # 变量赋值，自右向左进行

a = a + 10 # 可以简化为以下形式

a += 10

# b += 10 # 错误，因为等价于b = b + 10，b没有提前赋值

print(5 / 2) # 2.5

print(5 // 2) # 2

print(5 % 2) # 只要余数，模运算，1

print(2 \*\* 3) # 2的3次方，8

20 > 10 > 5 # True，它相当于下面的写法

10 > 10 and 10 > 5

课堂练习

1. 创建名为login.py的程序文件

2. 程序提示用户输入用户名

3. 用户输入用户名后,打印欢迎用户

#a = input(“aa: ”)

#print(‘欢迎用户’，‘a’)

#print(‘欢迎用户’+ ‘a’)

#print(‘欢迎用户’，str(a))

gitub

[root@room9pc01 ~]# mkdir notes

[root@room9pc01 ~]# cd notes/

[root@room9pc01 notes]# git init

[root@room9pc01 notes]# git add 第四阶段笔记

[root@room9pc01 notes]# git commit -m 'notes'

[root@room9pc01 notes]# git remote add origin https://github.com/changzhihong294/notes.git

[root@room9pc01 notes]# git push -u origin master