Docker Debian 使用学习

进入

```
docker start ics-vm
```

```
ssh -p 20022 Manfred@127.0.0.1
```

退出

```
exit
```

```
docker stop ics-vm
```

拷贝文件

拷入docker：

```
scp -P 20022 name.suffix Manfred@127.0.0.1:/home/Manfred
```

拷出

```
scp -P 20022 Manfred@127.0.0.1:/home/Manfred/name.suffix .
```

安装/卸载软件

```
sudo apt-get install xxx
```

```
sudo apt-get --purge remove python-notify
```

Git使用与Github

1. 初始化：

```
git init        # 在本地创建
git remote add origin git@github.com:yourName/yourRepo.git

git clone https://github.com/ManfredZhang/Programs.git  # 从Github克隆
```

2. 查看状态与操作记录

```
git status
git log
```

3. Add(ready to be committed) & Commiting

```
git add hello.c
git add '*.c'
git add .

git commit -m "what have I done"
git commit --allow-empty
# The --allow-empty option is necessary, because usually the change is
already committed by development tracing system. Without this option, git
will reject no-change commits. If the commit succeeds, you can see a log
labeled with your student ID and name by
```

4. 将本地内容推送到Github

```
git push -u origin master
git push
```

5. 从Github上更新本地内容

```
git pull origin master
git pull
```

6. 查看更改出的不同处

```
git diff
git diff HEAD
git diff --staged  #查看已add未commit的不同处
```

7. Reset：从已add未commit的状态（staged）取消，但保留本地更改

```
git reset staged_file.c
```

8. Checkout：回到某个commit之前的状态

```
git checkout -- octocat.txt    # Go ahead and get rid of all the changes
since the last commit for octocat.txt
```

9. 创建-查看-进入-(修改后)合并-删除分支

```
git branch clean_up             # 分支名为clean_up
git branch                                # 查看所有分支
git checkout clean-up   # 进入clean_up分支

git checkout master
git merge clean_up              # 合并到master分支

git branch -d clean_up  # 删除clean_up分支
```