Linux中的开发环境

开机启动

两个地方可以设置
1、/etc/init.d目录下
2、/etc/init.d/rc.local

命令：sysv-rc-conf　设置和查看开机启动服务

```
sudo update-rc.d ttest defaults
#90是优先级，越大优先级越低，越晚执行
```

设置开机启动

使用sysv-rc-conf命令设置运行级别

```
#sysv-rc-conf默认没有安装，首先安装
sudo apt-get install sysv-rc-conf

#执行命令sysv-rc-conf，用空格键选中或取消指定的运行级别。

#取消开机启动可以使用
sudo update-rc.d -f 脚本名  remove
```

防火墙

UFW概述

---UFW或UncomplicatedFirewall是iptables的接口，旨在简化配置防火墙的过程
UFW默认安装在Ubuntu上。如果没有安装，你可以使用sudo__apt-getinstall__ufw

UFW常见操作

```
sudo ufw status                 #查看状态和规则
sudo ufw disable                #禁用
sudo ufw enable                 #启用
sudo ufw reset                  #重置
sudo ufw status numbered        #显示规则编号
```

设置默认策略

```
sudo ufw default deny incoming   #拒绝所有传入连接
sudo ufw default allow outgoing  #允许所有传出连接
```

如果您刚刚开始使用防火墙，则首先要定义的规则是您的默认策略
这些规则控制如何处理未明确匹配任何其他规则的流量

开启或禁用制定连接

```
#允许连接
sudo ufw allow 端口/服务

#允许ssh远程连接
sudo ufw allow ssh  #或者sudo ufw allow 22/tcp

#允许未加密的web访问
sudo ufw allow http  #或sudo ufw allow 80
#允许加密的web访问
sudo ufw allow https  #或sudo ufw allow 443

#允许ftp访问
sudo ufw allow ftp  #或sudo ufw allow 21/tcp

#允许远程mysql访问
sudo ufw allow 3306

#允许特定范围的端口
sudo ufw allow 6000:6007/tcp #允许使用端口6000 - 6007 X11连接

#允许特定ip地址
sudo ufw allow from 15.15.15.51

#允许特定子网
sudo ufw allow from 15.15.15.0/24  #允许所有的ip地址范围从15.15.15.1到15.15.15.254

#拒绝连接
sudo ufw deny http
sudo ufw deny from 15.15.15.51
```

删除规则

```
sudo ufw status numbered #先查看编号
sudo ufw delete 2  #再按编号删除

#按实际规则
sudo ufw delete allow http
sudo ufw delete allow 80
```

Linux中的下载

概述

___wget，wget命令用来从指定的URL下载文件。wget非常稳定，它在带宽很窄的情况下和不稳定网络中有很强的适应性，如果是由于网络的原因下载失败wget会不断的尝试，直到整个文件下载完毕。如果是服务器打断下载过程它会再次联到服务器上从停止的地方继续下载。这对从那些限定了链接时间的服务器上下载大文件非常有用

使用

__wget-c重新启动下载中断的文件，对于我们下载大文件时突然由于网络等原因中断非常有帮助，我们可以继续接着下载而不是重新下载一个文件。需要继续中断的下载时可以使用-c参数

-P 将下载文件存到指定目录

认识curl

___curl命令是一个利用URL规则在命令行下工作的文件传输工具。它支持文件的上传和下载
所以是综合传输工具，但按传统，习惯称curl为下载工具

curl的下载

```
curl -O http://mirrors.sohu.com/python/3.6.4/Python-3.6.4.tar.xz
```

python开发环境配置

概述

___Python2和Python3之间存在着较大的差异，并且，由于各种原因导致了Python2和Python3的长期共存。在实际工作过程中，我们可能会同时用到Python2和Python3，因此，也需要经常在Python2和Python3之间进行来回切换。这就需要对python的版本进行管理，除此之外还需要对不同的软件包进行管理。大部分情况下，对于开源的库我们使用最新版本即可。但是，有时候可能需要对相同的Python版本，在不同的项目中使用不同版本的软件包

解决

___在这里，我们要使用两个工具：pyenv和virtualenv。前者用于管理不同的Python版本，后者用于管理不同的工作环境。有了这两个工具，Python相关的版本问题将不再是问题

认识pyenv

___pyenv是一个Python版本管理工具，它能够进行全局的Python版本切换，也可以为单个项目提供对应的Python版本。使用pyenv以后，可以在服务器上安装多个不同的Python版本，也可以安装不同的Python实现。不同Python版本之间的切换也非常简单。
pyenv官方地址https://github.com/pyenv/pyenv-installer

安装pyenv

详情见：开发环境.md

命令行输入：

```
$ curl -L https://github.com/pyenv/pyenv-installer/raw/master/bin/pyenv-installer | bash
```

pyenv的使用

virtualenv概述

___virtualenv本身是一个独立的项目，用以隔离不同项目的工作环境。例如，项目A和项目B都是使用Python2.7.13，但是，项目A需要使用Flask0.8版本，项目B需要使用Flask0.9版本。我们只要组合pyenv和virtualenv这两个工具，就能够构造Python和第三方库的任意版本组合，拥有了很好的灵活性，也避免了项目之间的相互干扰

注意：

___virtualenv本身是一个独立的工具，用户可以不使用pyenv单独使用virtualenv
但是，如果你使用了pyenv，就需要安装pyenv-virtualenv插件而不是virtualenv软件来使用virtualenv的功能

psm用途

切换pip的源

注意：psm需要安装

virtualenv虚拟环境的创建　详情见：开发环境.md

pycharm和虚拟开发环境结合

详情见：开发环境.md