# 课程介绍

* 1. Linux源码安装软件的方法;（理解）
* 2. Python环境搭建;（掌握）
* 3. Pycharm远程调试环境搭建;（了解）
* 4. Linux的启动执行顺序；（理解）

1. Linux源码安装
   1. 为什么要学习用源码安装Linux软件

一般来说，当遇到一些特定需求的时候，我们需要使用源码进行软件的安装：

比如软件放出了新的版本，而所用的发行版并没有及时跟进，这时候，想要尝鲜的话，就非得靠自己不可；

默认的Centos的BASE参考是没有python3的二进制文件支持的，所以只能使用源码安装（或者使用EPEL源）

源码包的编译用到了linux系统里的编译器，常见的源码包一般都是用C语言开发的，这也是因为C语言为linux上最标准的程序语言。Linux上的C语言编译器叫做gcc，利用它就可以把C语言变成可执行的二进制文件。所以如果你的机器上没有安装gcc就没有办法去编译源码。你可以使用 yum install -y gcc 来完成安装。

* 1. 源码安装的一般思路
     + 1. **./configure**

在这一步会自动检测你的linux系统与相关的套件是否有编译该源码包时需要的库，因为一旦缺少某个库就不能完成编译。只有检测通过后才会生成一个Makefile文件。

常用的选项有--prefix=PREFIX，用以指定程序的安装位置。更多的选项可通过--help查询。

* + - 1. **make**

Linux使用make命令进行源码工程管理，make会自动调用上一步生成的Makefile文件进行源码编译，一般这步不会出错。

* + - 1. **make install**

将生成的可执行文件和配置文件，拷贝到配置的安装目录下。

* 1. 编译型程序与解释型程序

如果简单点说，所有的程序语言最终都是被解释的，只是被谁解释的不同，才导致了程序分为了编译型程序语言和解释型程序语言。

所有的程序都是被人编写的，可以认为是一种人类能够沟通的语言，很显然，这样的语言，计算机是不可能认识的，通过一个工具将其转化为计算机能认识的，这个工具我们称之为编译器。

* + 1. 编译型语言

被编译器转码后才被计算机执行的代码，称之为编译型语言。

比如C/C++语言

* + 1. 解释型语言

程序被一个叫做解释器的程序来作为中间商和CPU打交道，这样的话，程序不需要关心CPU是谁，只要关心解释器能看懂什么语法，就写对应的语法即可。

比如python、java语言

* 1. 小结

本节内容偏重理论，在后面的python源码安装时，理论结合实际来进行理解。

1. Python环境搭建
   1. 源码安装python3.6
      1. 下载源码并解压

可以使用wget工具下载源码，也可以在windows下载后通过SCP工具传输。

* + 1. 安装编译所需的库

# Compilers and related tools:

yum groupinstall "development tools" # 可以只安装gcc： yum install gcc

# Libraries needed during compilation to enable all features of Python:

yum install zlib-devel bzip2-devel openssl-devel ncurses-devel sqlite-devel readline-devel tk-devel gdbm-devel db4-devel libpcap-devel xz-devel expat-devel

* + 1. 编译配置

./configure --enable-optimizations --prefix=/usr/local/python

make && make install

* 1. 在终端上可以直接使用Python3
     1. python安装后目录结构介绍

查看python安装目录的树状结构

# tree –L 2 /usr/local/python

/usr/local/python/

├── bin

│   ├── 2to3 -> 2to3-3.6

│   ├── 2to3-3.6

│   ├── easy\_install-3.6

│   ├── idle3 -> idle3.6

│   ├── idle3.6

│   ├── pip3

│   ├── pip3.6

│   ├── pydoc3 -> pydoc3.6

│   ├── pydoc3.6

│   ├── python3 -> python3.6

│   └── python3.6

├── include

│   └── python3.6m

├── lib

│   ├── libpython3.6m.a

│   ├── pkgconfig

│   └── python3.6

└── share

└── man

熟悉目录的作用：

|  |  |
| --- | --- |
| **通配符** | **含义** |
| **bin** | 保存了python的可执行文件，包括virtualenv,django-admin等 |
| **include** | python底层开发所需头文件 |
| **lib** | python代码import的模块和包目录 |
| **lib/site-packages** | 第三方库的保存路径 |

* + 1. 将python命令加入PATH路径

# export PATH=$PATH:/usr/local/python/bin

在普通用户的家目录下修改.bashrc文件，添加：

alias sudo="sudo env PATH=$PATH"

* 1. 修改pip源

建议在国内都将豆瓣源添加为pip的镜像源。

方法：

在家目录下创建一个.pip目录，然后在这个目录下创建pip.conf文件

pip.conf文件内容：

[global]

timeout = 60

index-url = https://pypi.doubanio.com/simple

[list]

format=columns

注意事项：使用https的URL，而不要使用http的方式。

* 1. virtualenv环境搭建
     1. 为什么要搭建python虚拟环境

在工程开发中，往往需要很多第三方库来辅助我们的项目，如果这台机器开发了多个项目，很可能会出现不同的项目需要的第三方库版本的管理混乱。

建议在实际开发中，一个项目就搭建一个python虚拟环境，保证每个项目不会由于系统原因导致版本混乱，为代码调试引入不必要的麻烦。

* + 1. 如何搭建python虚拟环境
       1. 在系统python目录下安装virtualenv

# pip3 install virtualenv

注意：这里一定要安装在系统标准的环境下，这样所有的用户都可以使用virtualenv命令进行后期虚拟环境的创建了。

安装完成后，可以敲入vir然后使用TAB键看是否正确安装。

思考，virtualenv命令安装在哪里了？

* + - 1. 利用virtualenv创建一个虚拟环境项目

使用virtualenv命令最简单的方式就是指定创建虚拟环境需要的python解释器是哪里，然后他就会自动在需要创建的目录下自动创建了。

# virtualenv –p python解释器的路径 创建的虚拟目录名

* + - 1. 虚拟环境项目的目录结构

和系统python的目录结构是一致的，最大的区别是lib下的site-packages目录，他是空的，因为这样的设置，每建立一个虚拟环境，第三方库的目录都是空的，这样每个虚拟环境就隔离开了。

* + - 1. 使用虚拟环境

进入一个虚拟环境

# source 虚拟环境目录/bin/activate

退出一个虚拟环境

# deactivate

* + 1. python虚拟环境的工作原理

进入一个虚拟环境，实际就是执行虚拟环境的activate文件，这个文件实际是一个shell脚本。

通过查看PATH变量值的变化，可以总结：

进入虚拟环境，实际就是把虚拟环境的目录名添加到PATH变量的最开头。这样用户敲python的时候，最先是从虚拟环境里找到对应的解释器了。

* 1. Ipython的安装和使用

演示ipython的使用

* 1. 小结

联想下windows下安装python和linux下安装有什么一致性。

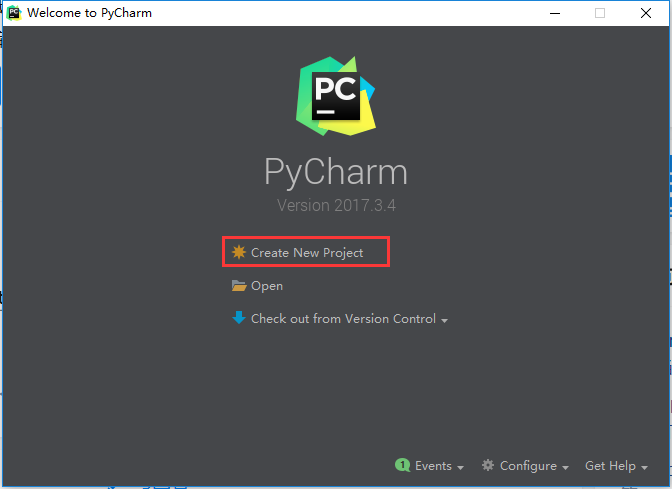
结合windows下python的目录和linux目录做一个对比。

1. Pycharm远程环境的搭建
   1. IDE工具和编辑器的关系

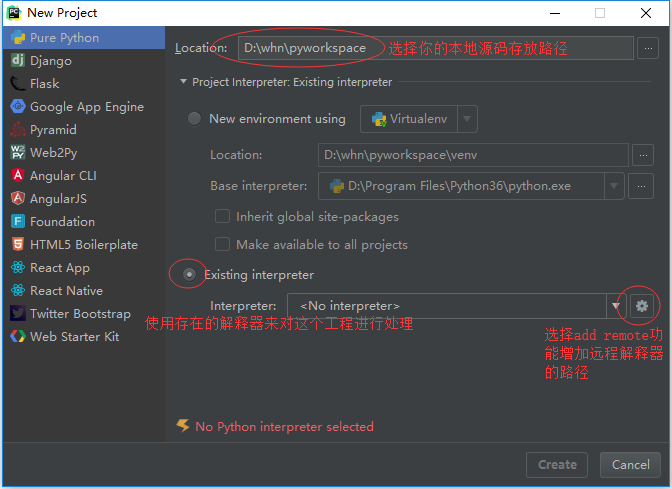
编辑器：只是提供了编写的平台，缺少自动运行，代码提示等高级功能，虽然sublime或者vscode这些工具通过配置也能达到这些功能，但还是有或多或少不尽人意的地方。

IDE：将代码提示，编写代码，运行，调试等一条龙的功能都集成在一个软件上，使用方便，唯一的缺点就是要花钱。

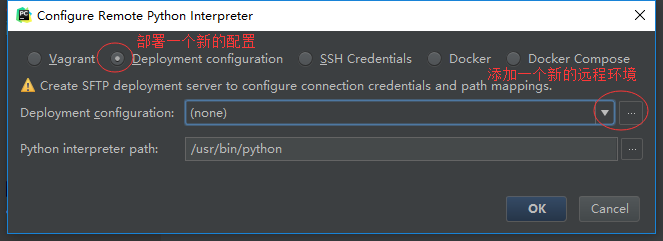
* 1. pycharm配置远程解释器

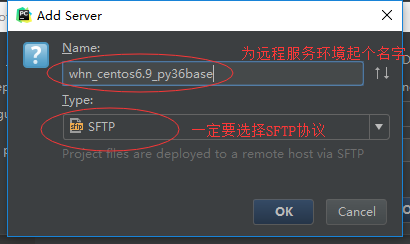


选择创建工程后，进行工程解释器的配置：

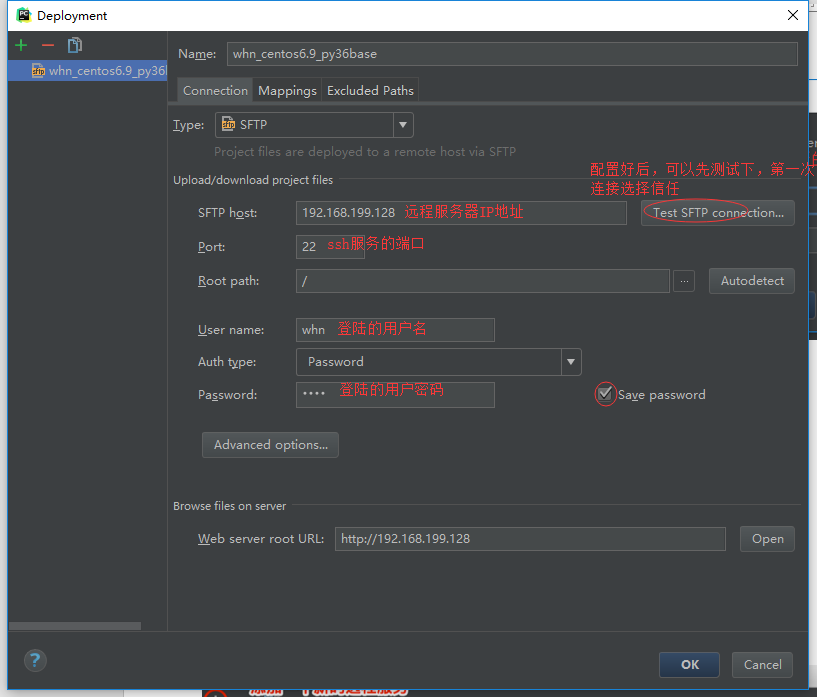


添加一个远程配置选项：

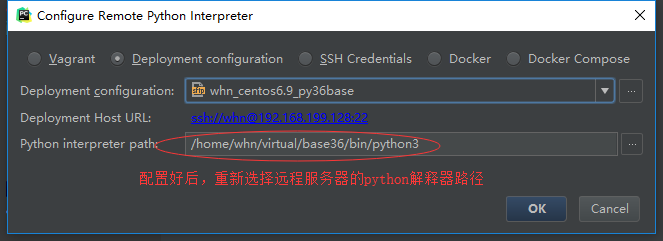




详细配置：

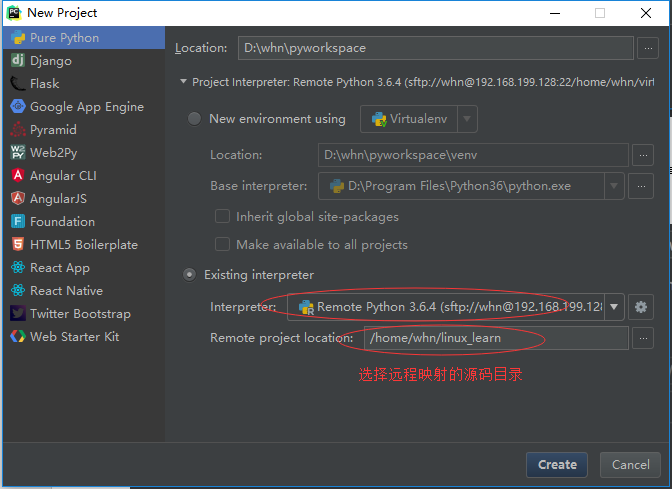


远程服务器的解释器路径配置：



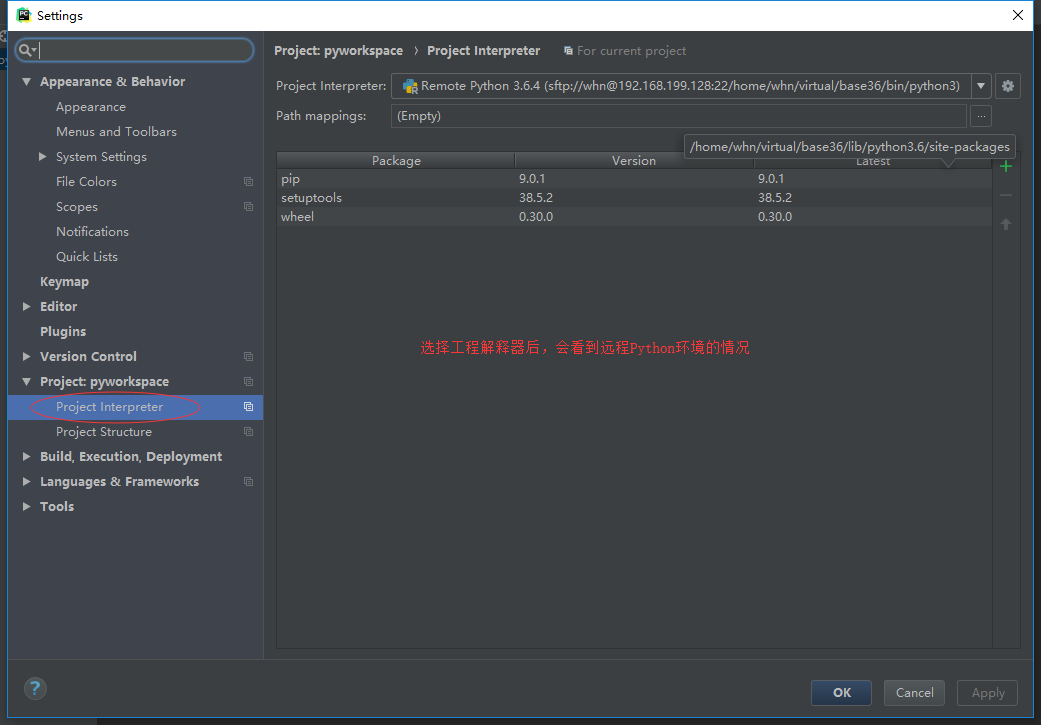
最后，配置项目的解释器信息和远程共享目录。

远程共享目录建议修改为python代码保存在linux系统的目录，这个选项非常关键，配置正确后，后面使用pycharm创建的文件，会自动上传到linux下这个目录。

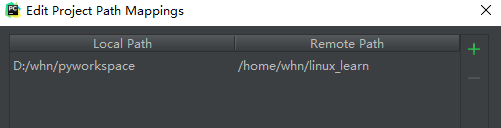


点击create后产生新工程。如果由window防火墙阻止，记得选择允许。

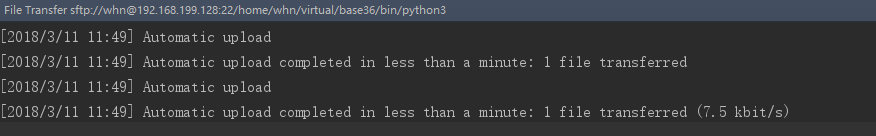
可以在工程创建后，选择File --> settings里看下python解释器是否正确。



若需要在pycharm上调试程序，需要在path mappings里进行目录的映射配置。



在工程中创建文件，会自动上传：



通过这样的配置后，pycharm将自动同步该虚拟环境下的符号表，会自动提示改项目的类、对象、属性等信息，大大提高编码效率。

* 1. 远程执行

远程执行有两种方式：

* + - 1. 直接在Linux下执行，相当于借助pycharm的编辑器强大的功能提高编辑效率。推荐使用这种方式，可以在linux下看执行效果，打印信息最真实。
      2. 利用pycharm的run功能进行调试，pycharm会再次连接远程服务器，将打印信息传输到pycharm的控制台上。懒人方法，可以用，但由于目前pycharm的版本问题，每次调试都会建立一个进程在服务器上，无形给服务器增大了压力。

1. Linux系统启动过程
   1. Linux程序启动过程

抛开硬件的过程，Linux下启动第一个程序是init程序，然后通过配置文件开始进行后续程序的执行，下面以Centos6.9版本为例，他的启动标准是SystemV，虽然古老，但仍然被广泛使用：

/etc/inittab ---> /ect/rc.d/rc.sysinit ---> /etc/rc.d/rcX.d/下的服务启动脚本 ---> /etc/rc.d/rc.local。

* 1. shell配置启动过程

为了在系统运行时，或者用户登陆时，执行一些特定事务，Linux系统提供了下面一系列文件，下面列举如下：

|  |  |
| --- | --- |
| **文件名** | **功能描述** |
| **/etc/profile** | 此文件为系统的每个用户设置环境信息,**当用户第一次登录时**,该文件被执行.并从 /etc/profile.d 目录的配置文件中搜集shell的设置. |
| **/etc/bashrc** | 为每一个运行**bash shell**的用户执行此文件. |
| **~/.bash\_profile** | 每个用户都可使用该文件输入专用于自己使用的shell信息,当用户登录时,该文件仅仅执行一次. |
| **~/.bashrc** | 该文件包含专用于用户的bash shell的bash信息,**当登录时以及每次打开新的shell时,该该文件被读取**. |
| **~/.bash\_logout** | 当每次退出系统(退出bash shell)时,执行该文件. |

1. 课程总结
   1. 重点
      * 1. 计算机编程语言是如何被计算机执行的，编译的过程，解释器的原理；
        2. 如何使用不同版本的python解释器进行工作；
        3. python虚拟环境工具virtualenv的使用；
        4. pip的配置和使用；
        5. ipython的使用；
   2. 难点
      * 1. python源码编译前，为什么需要安装依赖的库？
        2. 编译、链接的意义和作用
        3. pycharm下的远程调试环境的搭建
        4. shell脚本的启动顺序
   3. 学习方法
      * 1. 使用VMware镜像功能，恢复系统初始化状态后，练习环境搭建；
        2. 使用virtualenv切换不同环境后，查看PATH变量情况；
        3. 利用pycharm的自动提示功能，配置不同的环境；
2. 课后练习
   * + 1. 利用源码安装python环境在centos完整版，在新的Linux环境下将今天的内容重新做一遍。
       2. 在linux编写python代码，完成功能：

列出1000以内的完数，请结合搜索引擎的资料确定完数的概念。

列出2000以内的完数。

* + - 1. 用python解决下面这个问题：

一位法官在审理一起盗窃案时，对涉及到的四名嫌疑犯A、B、C、D进行了审问。四人分别供述如下：

A：“罪犯在B、C、D三人之中。”

B：“我没有作案，是C偷的。”

C：“在A和D中间有一个是罪犯。”

D：“B说的是事实”

经过充分的调查，证实四人中只有两人说了真话，并且罪犯只有一个。请确定真正的罪犯。

1. 面试题
   * + 1. 常见的Linux发行版本有很多，下面不是Linux发行版本的是（ ）。

A、RedHat B、Ubuntu C、X window D、SuSE

* + - 1. 下面不是对Linux操作系统特点描述的是（ ）。

A、良好的可移植性 B、单用户 C、多用户 D、多任务

* + - 1. 下面（ ）命令不能用来显示文件内容。

A、more B、less C、gzip D、cat

* + - 1. 查看创建目录命令mkdir的帮助，可以使用命令（ ）。

A、mkdir --h B、help mkdir C、mkdir -p D、man mkdir

* + - 1. 假设目录中存在若干文件file\_num.txt(num表示文件编号，1~10)，使用以下( )通配符可以挑选出其中编号为1，3，5的文件。

A、file\_\*.txt B、file\_?.txt C、file\_[135].txt D、file\_[^135].txt

* + - 1. 文件权限r-x-wxr—对应的八进制表示形式为（ ）

A、0536 B、0654 C、0656 D、0534

* + - 1. 在Centos系统中，下面关于shell环境变量配置文件的描述，正确的是（ ）

1. 用户登录系统时，bash首先执行/etc/profile配置文件和/etc/profile.d/目录下的配置文件，这些配置文件对所有用户都有效。
2. 用户登录系统时，bash首先执行.bash\_profile文件和.bashrc文件，这些配置文件对所有用户都有效。
3. 用户主目录下的.bashrc设置为每次登录时执行，而.bash\_profile则为每次打开新的终端时执行
4. 执行用户主目录下的环境变量配置文件时，不可以重复设置用户登录时配置文件中已经设置的选项
5. 课外阅读
   1. 扩展阅读
      * 1. Centos7添加开机启动脚本和服务（<https://my.oschina.net/bobchow/blog/807147>）
        2. systemd常用命令（<http://www.ruanyifeng.com/blog/2016/03/systemd-tutorial-commands.html>）