Python官方站点：http://www.python.org

一、安装Python3

#!/bin/bash

yum install -y sqlite-devel tk-devel tcl-devel readline-devel zlib-devel gcc gcc-c++ openssl-devel libffi-devel

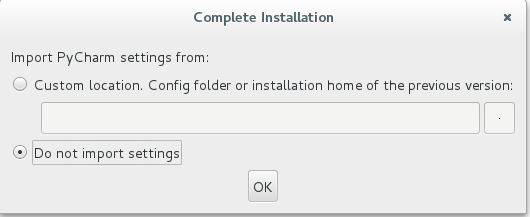
tar xzf Python-3.6.7.tgz

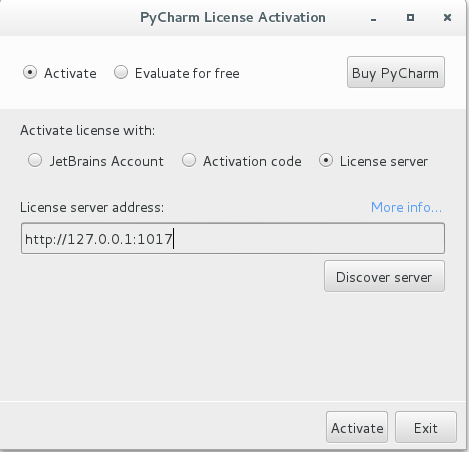
cd Python-3.6.7

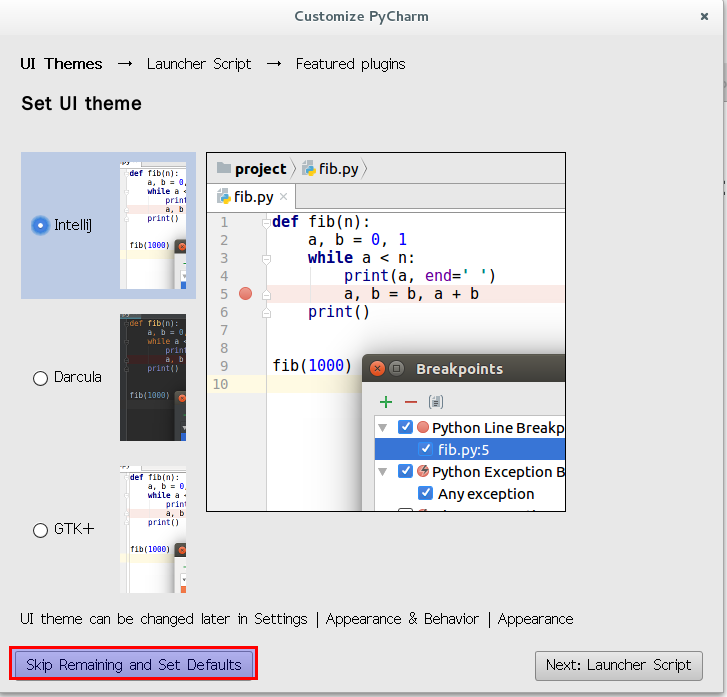
./configure --prefix=/usr/local/

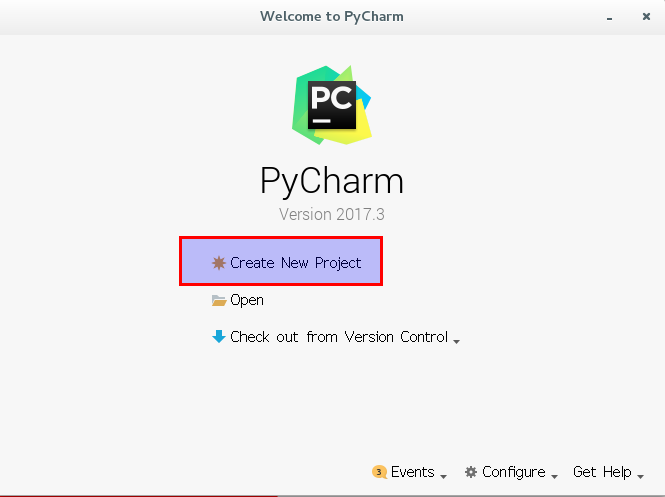
make && make install

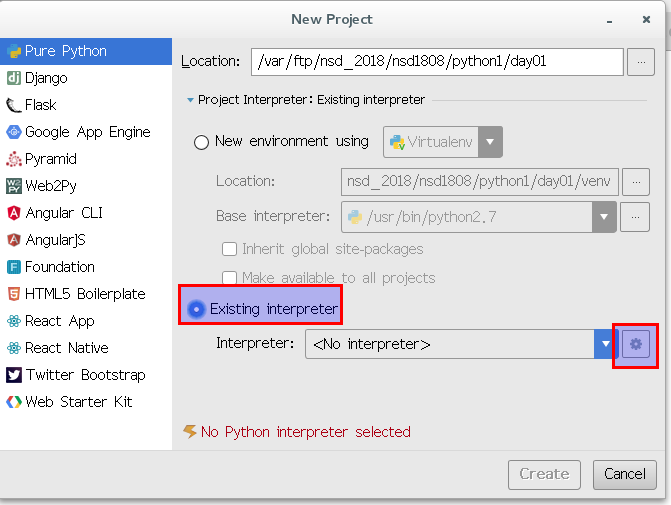
二、Pycharm配置



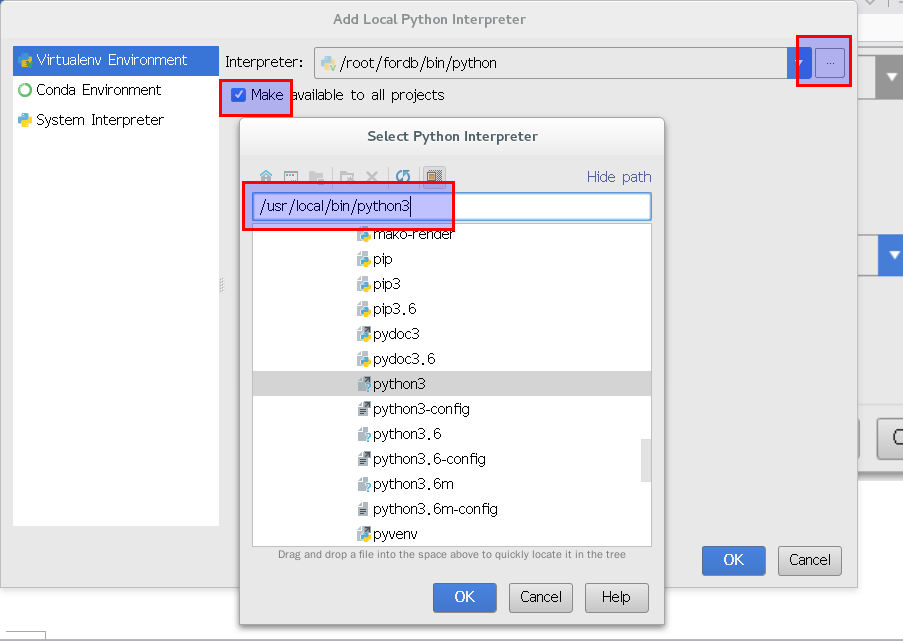






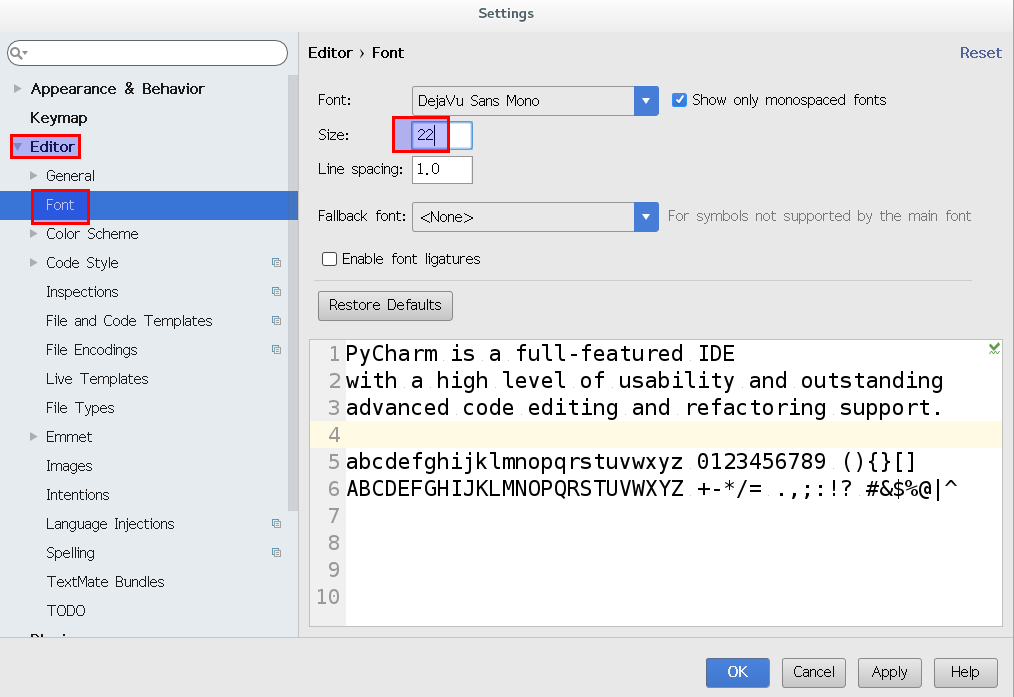


点击齿轮后，选“add local”



接下来点OK和create即可

菜单上点击file -> settings



**使用tab提示：**

p：parameter 参数

m：method 方法

c：class 类

v：variable 变量

f：function 函数

三、git的使用

**1、安装及初始配置**

[root@room9pc01 ~]# git config --global user.name "Mr.Gao"

[root@room9pc01 ~]# git config --global user.email "4..@qq.com"

[root@room9pc01 ~]# git config --global core.editor vim

#设置用户信息、默认编辑器为vim

[root@room9pc01 ~]# git config --list

[root@room9pc01 ~]# cat ~/.gifconfig

**2、git三个区域**

工作区：电脑里的工作目录

暂存区：工作区和版本库的过渡区域

版本库：真正存储各个版本的位置（工作目录下的隐藏目录.git）

**3、git本地操作**

1）初始化使用git管理的项目

[root@node2 ~]# cd devops/

[root@node2 devops]# git init

或者

[root@node2 ~]# git init devops

[root@node2 ~]# ls -a devops/

2）在项目中编写代码

[root@node2 ~]# cd devops/

[root@node2 devops]# vim hi.py

print('Hello World!')

3）提交代码到暂存区

[root@node2 devops]# git add .

4）查看暂存区

[root@node2 devops]# git status

5）撤出暂存区

[root@node2 devops]# git rm --cached hi.py

6）提交文件到版本库

[root@node2 devops]# git add .

[root@node2 devops]# git commit -m "script hi.py" #注释

[root@node2 devops]# git log #查看提交记录

1. 查看版本库中的文件

[root@node2 devops]# git ls-files

1. 误删除后，重新检出文件

[root@node2 devops]# git rm hi.py #删除工作区文件

[root@node2 devops]# git status

[root@node2 devops]# git reset HEAD hi.py

[root@node2 devops]# git checkout -- hi.py

**4、搭建本地gitlab服务器**

1）使用docker搭建gilab服务器

[root@node1 ~]# systemctl start docker

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

[root@node1 ~]# 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

#使用浏览器访问http://192.168.1.1/，首次启动需要一段时间

#第一次访问需要设置8位以上密码。用户名为root

1. gitlab服务器的重要组成部分

群组：对应一个个团队

项目：每个group都会编写多个软件项目，每个软件项目创建一个project

成员：对应一个个用户

#新建用户时不能设置密码。创建完成后，编辑用户才可以设置密码

1. 设置用户成为项目的主程序员

**5、将本地软件上传到gitlab服务器**

1）使用HTTP上传软件项目到gitlab

[root@node1 ~]# cd devops/

[root@node1 devops]# git remote rename origin old-origin

#error: 不能重命名配置小节... 忽略即可

[root@node1 devops]# git remote add origin \

> http://192.168.1.1/devops/core\_py.git

[root@node1 devops]# git push -u origin --all

[root@node1 devops]# git push -u origin --tags

2）使用SSH免密的方式上传

[root@node1 ~]# ssh-keygen -t rsa -C "gjq@qq.com" -b 4096

#将本地~/.ssh/id\_rsa.pub内容拷贝到gitlab用户的SSH密钥；

[root@node1 devops]# git remote remove origin

[root@node1 devops]# git remote add origin \

> git@192.168.1.1:devops/core\_py

[root@node1 devops]# echo "demo project" > README.md

[root@node1 devops]# git add .

[root@node1 devops]# git commit -m "add readme"

[root@node1 devops]# git push

如果在上传时候出现Permission Denied，解决办法：

[root@node1 devops]# ssh-agent

#把输出的环境变量直接运行一下，然后执行以下命令

[root@node1 devops]# ssh-add

[root@node1 devops]# chmod 0600 ~/.ssh/id\_rsa

#若提示id\_rsa权限过高，修改为0600

四、Python起步

**1、语法结构、I/O语句**

1）代码编写风格

代码块缩进：以关键字开始，以冒号结束，之后n行代码构成，必须缩进对齐；

注释及续行：注释使用#(快捷键：选中内容Ctrl+/)、续行使用\

同行多个语句：使用分号;

1. 输出语句print

#字符串必须使用单引/双引，否则认为是变量

>>>print('hello world')

>>>print('hello', 'world')

>>>print('hello', 'world', 10, sep='\*\*\*', end='') #修改分隔符，结束符

>>>print('Jerry\_' + 'Gao') #拼接

hello world

hello world

hello\*\*\*world\*\*\*10Jerry\_Gao

3）输入语句input

>>>num = input("number:") #input读入的数据全部是字符类型

>>>print(int(num) + 10) #int()函数将字符转成数字

>>>print(num + '10')

>>>print(num + str(10))

number:100

110

10010

10010

**2、Python变量、运算符**

1）变量命名规范

-字母、数字、下划线（开头不可以数字）；区分大小写

-变量名建议采用小写字母，多个单词使用\_分隔，如：py\_str

-变量名用名词，函数名用谓词（动+名），如：get\_name

-类名采用驼峰形式，如：FirstName

2）变量赋值

-python变量使用前，必须先赋值(=)

-python的数据有严谨的类型，相同类型的数据进行运算

-python的增量赋值

>>>n += 1 #相当于n=n+1，不支持n++/++n

>>>n \*= 1

1. 运算符

-标准算术运算符

+ - \* / //(求商) %(求余) \*\*(乘方)

divmod(5,2) #同时求商和余数

-比较运算符

< <= > >= == != <>

10 < 20 > 15 #支持连续比较

'ab' > 'b' #根据ASCII码比较，第1个字母开始比较

-逻辑运算符

and not(优先级最高) or