-主

git clone [git@github.com:TopShares/Python.git](mailto:git@github.com:TopShares/Python.git)

**nginx**: 反向代理服务器，处理静态资源，负载均衡等。   
**gunicorn**: [**Python**](http://lib.csdn.net/base/python) WSGI HTTP Server for UNIX。   
**virtualenv**: 创建独立的 python 环境。   
**supervisor**: 基于[**Linux**](http://lib.csdn.net/base/linux)[**操作系统**](http://lib.csdn.net/base/operatingsystem)的一款服务器管理工具，用以监控服务器的运行，发现问题能立即自动预警及自动重启等功能。、

yum upgrade

yum update

yum install postgresql postgresql-contrib

yum install supervisor

yum install install libpq-dev python-dev

yum install ngnix

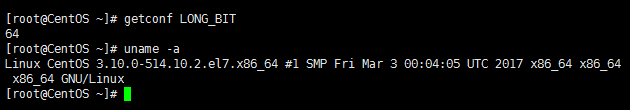
yum install libpq-dev python-dev

先装

yum install sqlite-devel

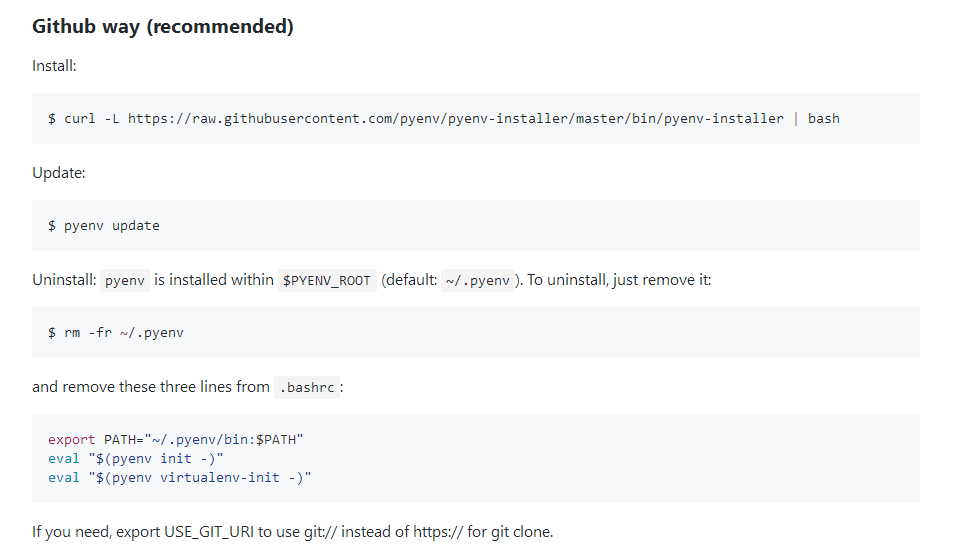
再装python

服务器版本：

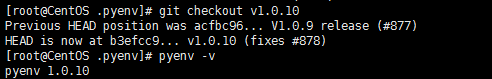
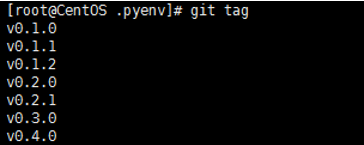


安装python版本控制

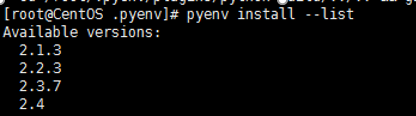
$ curl -L https://raw.githubusercontent.com/pyenv/pyenv-installer/master/bin/pyenv-installer | bash

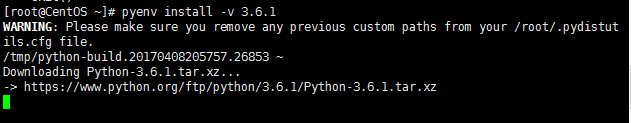


将pyenv切到v1.0.10



安装Python3.6.1





获取失败

手动安装

准备必要的库文件

yum install -y gcc zlib-devel openssl-devel readline\*

# wget <https://www.python.org/ftp/python/3.6.1/Python-3.6..tgz>

# tar -xzvf Python-3.6.1.tgz -C  /tmp

# cd  /tmp/Python-3.6.1/

把Python3.6安装到 /usr/local/python3 目录

# ./configure --prefix=**/usr/local/python3**

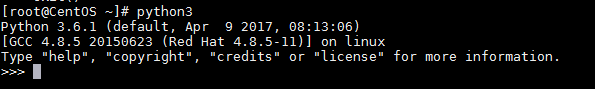
# make && make install

创建软连接

ln -s /usr/local/python3/bin/pip3 /usr/bin/pip3

ln -s /usr/local/python3/bin/python3.6 /usr/bin/python3

ln -s /usr/local/python3/bin/pyvenv /usr/bin/pyvenv



## **虚拟环境配置**

使用virtualenv创建虚拟环境, Ubuntun和Mac安装程序基本一致

#安装virtualenv

$ pip install virtualenv

#创建虚拟环境

$ virtualenv -p /usr/local/python3/bin/python3.6 ENV3.6

Running virtualenv with interpreter /usr/local/bin/python3.4

Using base prefix '/Library/Frameworks/Python.framework/Versions/3.4'

New python executable in ENV3.4/bin/python3.4

Also creating executable in ENV3.4/bin/python

Installing setuptools, pip...done.

#激活虚拟环境

$ source /root/ENV3.6/bin/activate

退出虚拟环境

$ deactivate

#查看当前环境下的安装包

$ pip list

pip (1.5.6)

setuptools (3.6)

### **Django Gunicorn 安装**

安装最新版的Django版本

#安装最新版本的Django

$ pip install django

pip install django

pip install gunicorn

# pip install psycopg2

# 项目与APP

## **项目创建**

现在正式开始吧, 我们创建一个名为my\_blog的Django项目

**创建项目的指令如下:**

$ django-admin.py startproject my\_blog

现在来看一下整个项目的文件结构

$ tree my\_blog #打印树形文件结构

my\_blog

├── manage.py

└── my\_blog

├── \_\_init\_\_.py

├── settings.py

├── urls.py

└── wsgi.py

1 directory, 5 files

## **建立Django app**

　在Django中的app我认为就是一个功能模块, 与其他的web框架可能有很大的区别, 将不能功能放在不同的app中, 方便代码的复用

建立一个article app

$ python manage.py startapp article

现在让我们重新看一下整个项目的结构

── article

│ ├── \_\_init\_\_.py

│ ├── admin.py

│ ├── migrations

│ │ └── \_\_init\_\_.py

│ ├── models.py

│ ├── tests.py

│ └── views.py

├── db.sqlite3

├── manage.py

├── my\_blog

├── \_\_init\_\_.py

├── \_\_pycache\_\_

│ ├── \_\_init\_\_.cpython-34.pyc

│ ├── settings.cpython-34.pyc

│ ├── urls.cpython-34.pyc

│ └── wsgi.cpython-34.pyc

├── settings.py

├── urls.py

└── wsgi.py

并在my\_blog/my\_blog/setting.py下添加新建app

INSTALLED\_APPS = (

...

'article', #这里填写的是app的名称

)

## **运行程序**

$ python manage.py runserver #启动Django中的开发服务器

#如果运行上面命令出现以下提示

You have unapplied migrations; your app may not work properly until they are applied.

Run 'python manage.py migrate' to apply them.

#请先使用下面命令

python manage.py migrate

#输出如下信息

Operations to perform:

Apply all migrations: contenttypes, sessions, admin, auth

Running migrations:

Applying contenttypes.0001\_initial... OK

Applying auth.0001\_initial... OK

Applying admin.0001\_initial... OK

Applying sessions.0001\_initial... OK

运行成功后,会显示如下信息

#重新运行启动Django中的开发服务器

$ python manage.py runserver

#运行成功显示如下信息

System check identified no issues (0 silenced).

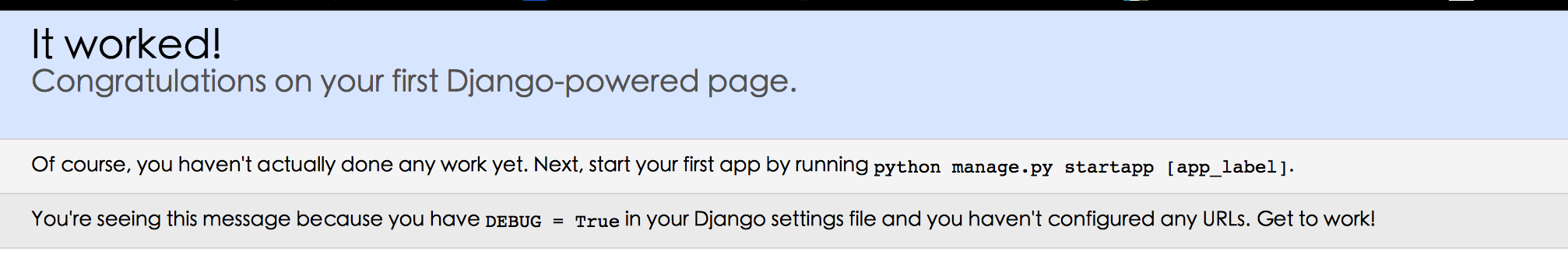
December 21, 2014 - 08:56:00

Django version 1.7.1, using settings 'my\_blog.settings'

Starting development server at http://127.0.0.1:8000/

Quit the server with CONTROL-C.

现在可以启动浏览器, 输入http://127.0.0.1:8000/, 当出现



说明你成功走出了第一步!

命令梳理:

python manage.py [options] #Django Command python manange.py -h帮助文档

django-admin.py startproject my\_blog #创建项目

python manage.py startapp article #创建app