# Django安装配置

前言:

Django是一个功能强大的Web框架, 可以帮助你将Python应用程序或网站变为现实. Django包含一个简化的开发服务器, 用于在本地测试你的代码, 但对于与生产相关的任何事情, 都需要更安全, 更强大的Web服务器(Nginx, Apache等). 我们将在虚拟环境中安装Django, 这将允许你的项目和他们的需求分开处理, 本教程将创建两个示例项目.

**安装virtualenv(创建Python虚拟环境)与virtualenvwrapper(管理虚拟环境):**

pip3 install virtualenv virtualenvwrapper

**对virtualenvwrapper进行配置, 并运行一下它的初始化工具:**

echo “export WORKON\_HOME=~/Env” >> ~/.bashrc

echo “source /usr/bin/virtualenvwrapper.sh” >> ~/.bashrc

source ~/.bashrc

*添加到shell初始化脚本中, 以便每次启动终端自动执行.*

*WORKON\_HOME指定的是它将要用来存放各种虚拟环境目录的目录, 这里我设置为~/Env*

**创建Django项目:**

现在我们有了虚拟环境工具, 我们将创建两个虚拟环境, 分别安装Django, 并启动两个项目.

**创建第一个项目:**

通过virtualenvrapper脚本的一些命令轻松地创建一个虚拟环境.

mkvirtualenv firstsite

*这将创建一个虚拟环境, 在其中安装Python和PIP(自动),并自动激活环境. 你的命令行提示符的开始会附加上你的虚拟环境名称. 现在,通过pip安装的任何软件都会安装到虚拟环境中, 而不是安装在全局系统上. 这使我们能够在每个项目的基础上分离我们的软件包.*

使用pip安装Django:

pip3 install Django

安装Django后, 通过以下命令创建我们的第一个示例项目:

cd ~

django-admin startproject firstsite #也可以django-admin.py startproject firstsite. 不过会产生不同的依赖问题, 所以此处跟着做就行

这将创建一个目录名firstsite在你的家目录. 在这里面是一个管理脚本, 用于处理项目的各个方面, 以及用于存储实际项目项目代码的另一个同名目录.

进入第一级目录, 以便我们可以开始设置示例项目的最低要求.

cd ~/firstsite

首先迁移数据库以初始化我们的项目将使用的SQLite数据库.

python3 manage.py migrate

**如果出现” TypeError: argument of type 'PosixPath' is not iterable”错误:**

进入settings.py

vim ~/firstsite/firstsite/settings.py

导入os:

import os

修改DATABASES为:

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': str(BASE\_DIR / "db.sqlite3"),

}

}

**或者出现” django.core.exceptions.ImproperlyConfigured: SQLite 3.9.0 or later is required (found 3.7.17).”错误:**

下载SQLite3最新包:

<https://www.sqlite.org/download.html>

wget <https://www.sqlite.org/2021/sqlite-autoconf-3370100.tar.gz>

编译安装:

tar xf sqlite-autoconf-3370100.tar.gz

cd sqlite-autoconf-3370100

./configure –prefix=/usr/local/ && make && make install

改变原来的SQLite3命令:

首先查看SQLite3在哪个目录中有可执行文件

whereis sqlite3

**结果:** sqlite3: /usr/bin/sqlite3 /usr/local/bin/sqlite3 /usr/share/man/man1/sqlite3.1.gz

确定是否是旧版本

/usr/bin/sqlite3 -version

然后替换旧版本

mv /usr/bin/sqlite3 /usr/bin/sqlite3\_old

ln -s /usr/local/bin/sqlite3 /usr /bin/sqlite3

同时要找到新版本的libsqlite3.so.0.8.6、libsqlite3.so.0然后替换掉旧版本的.

这时候你重新运行命令发现还是报错, 原因是django读取了旧库,独立的sqlite3可执行文件的版本无关紧要(对于此问题), 重要的是Python库的版本.

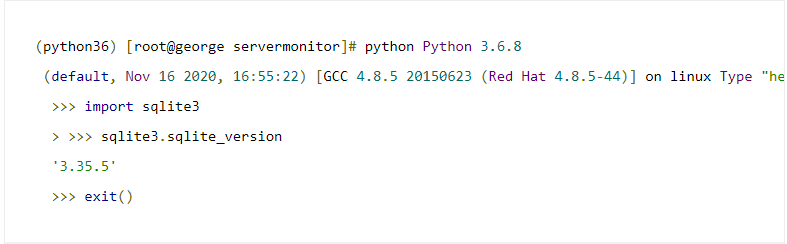
**验证:**



更改库路径:

export LD\_LIBRARY\_PATH=”/usr/local/lib/” # 可以放进自动初始化脚本中, 避免每次手动执行.

**复试:**

****

至此, 执行恢复正常.

现在, 项目目录中应该有一个名为db.sqlite3的数据库文件. 现在, 可以通过键入以下命令创建一个管理用户:

python3 manage.py createsuperuser

这时会提示你提供一些信息: 用户名、电子邮箱地址、密码.

接下来, 使用文本编辑器打开项目的配置文件:

nano firstsite/settings.py

由于我打算将设置Nginx来服务我的网站, 需要配置一个目录来保存我们网站的静态资源. 这将使Nginx能够直接提供这些服务, 这将对性能产生积极影响.我们将告诉Django把它们放到我们项目的基本目录下的静态目录中. 将此行添加到文件的底部以配置此行为:

STATIC\_ROOT = os.path.join(BASE\_DIR, “static/”)

完成后保存并关闭文件. 现在, 收集我们网站的静态元素, 并通过输入以下内容将其放置在该目录中:

python3 manage.py collectstatic

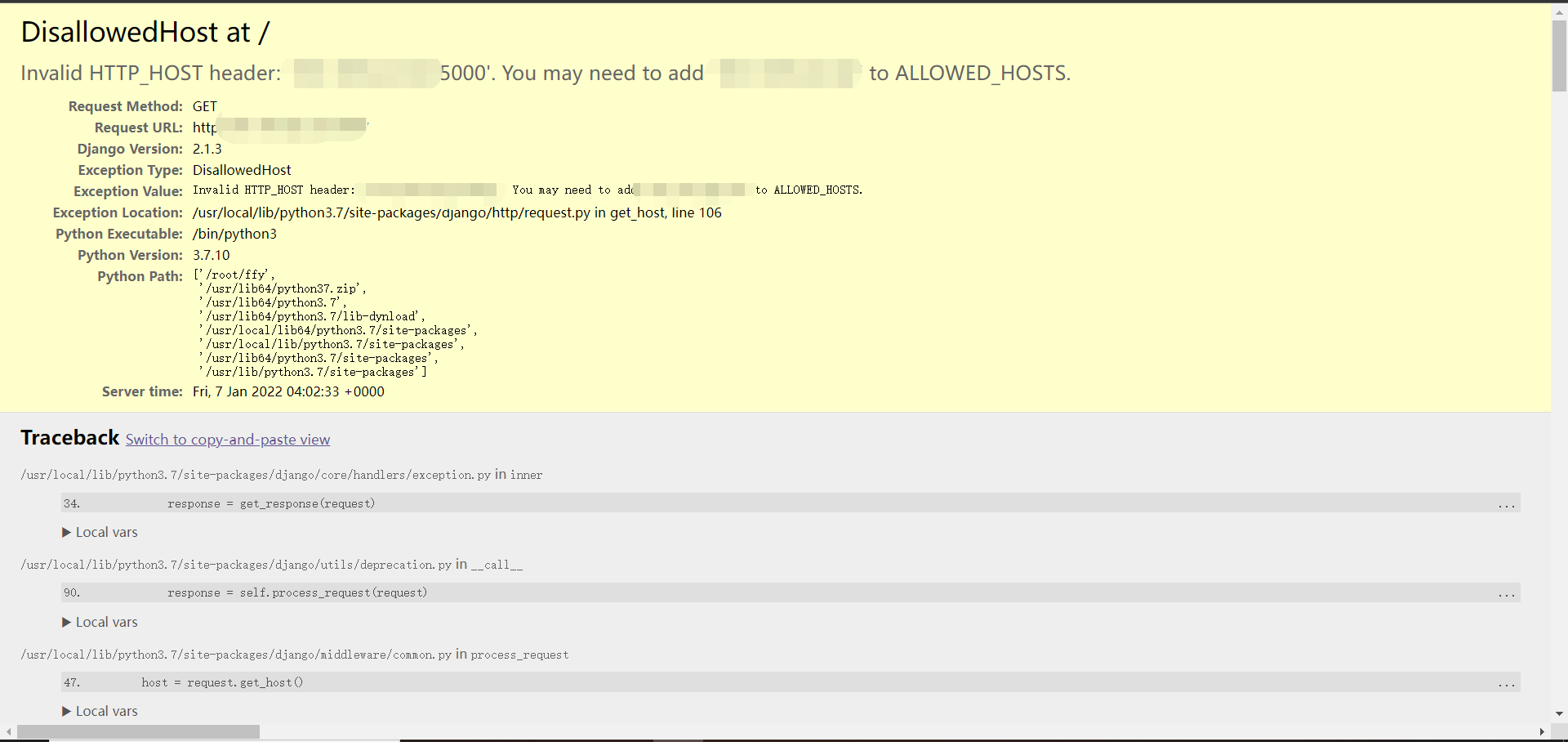
现在项目目录中将有一个名为static的新目录. 接下来可以通过暂时启动开发服务器来测试我们的项目.

python3 manage.py runserver 0.0.0.0:5000

这将在端口5000上启动开发服务器. 在浏览器中访问服务器的域名或IP地址, 然后接在后面输入:5000

<http://server_domain_or_IP:5000>

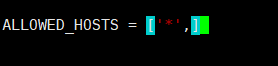
**此时如果无法访问:**



**则进入settings.py**

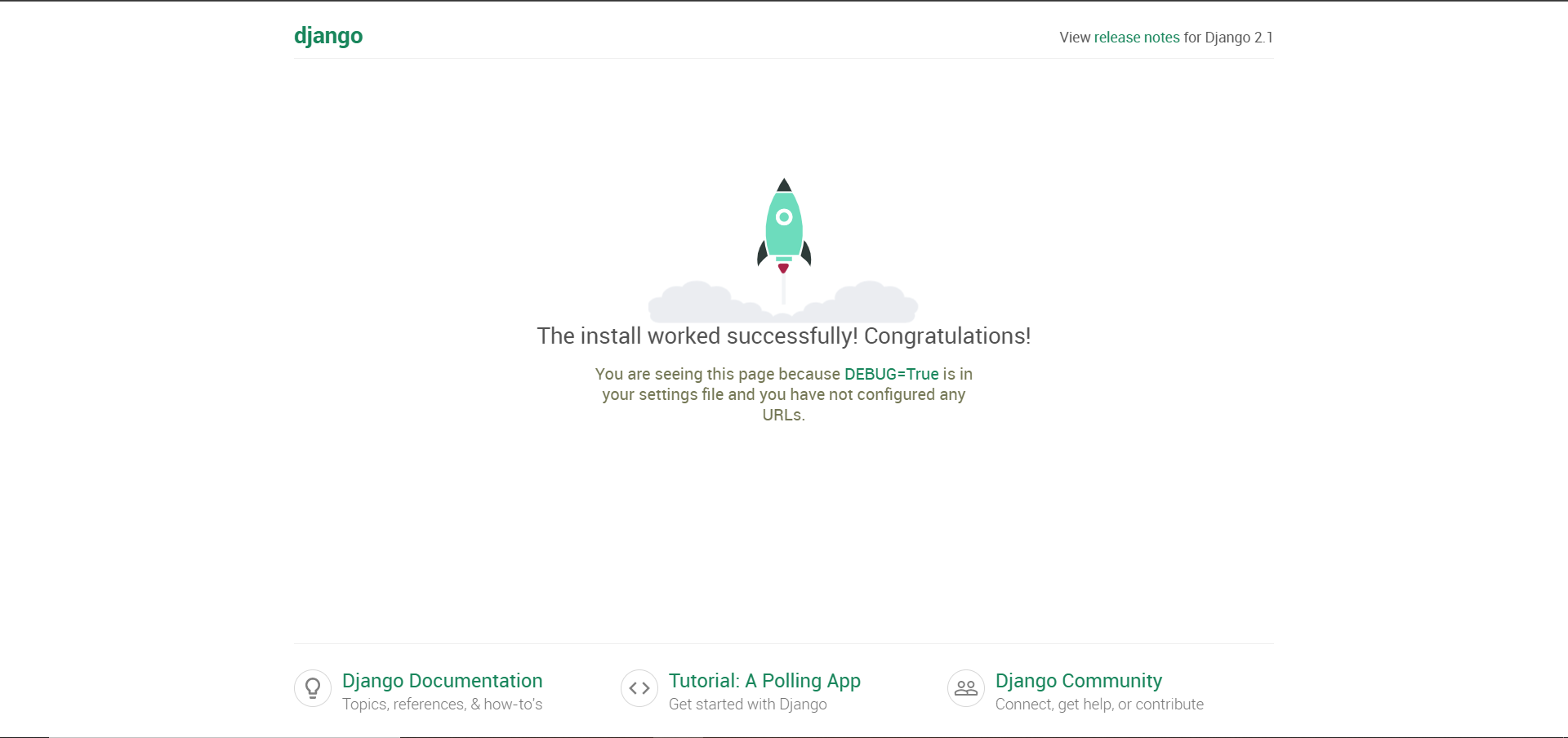
vim ~/firstsite/firstsite/settings.py

**修改ALLOWED\_HOSTS = []为 ALLOWED\_HOSTS = [‘\*’,]:**



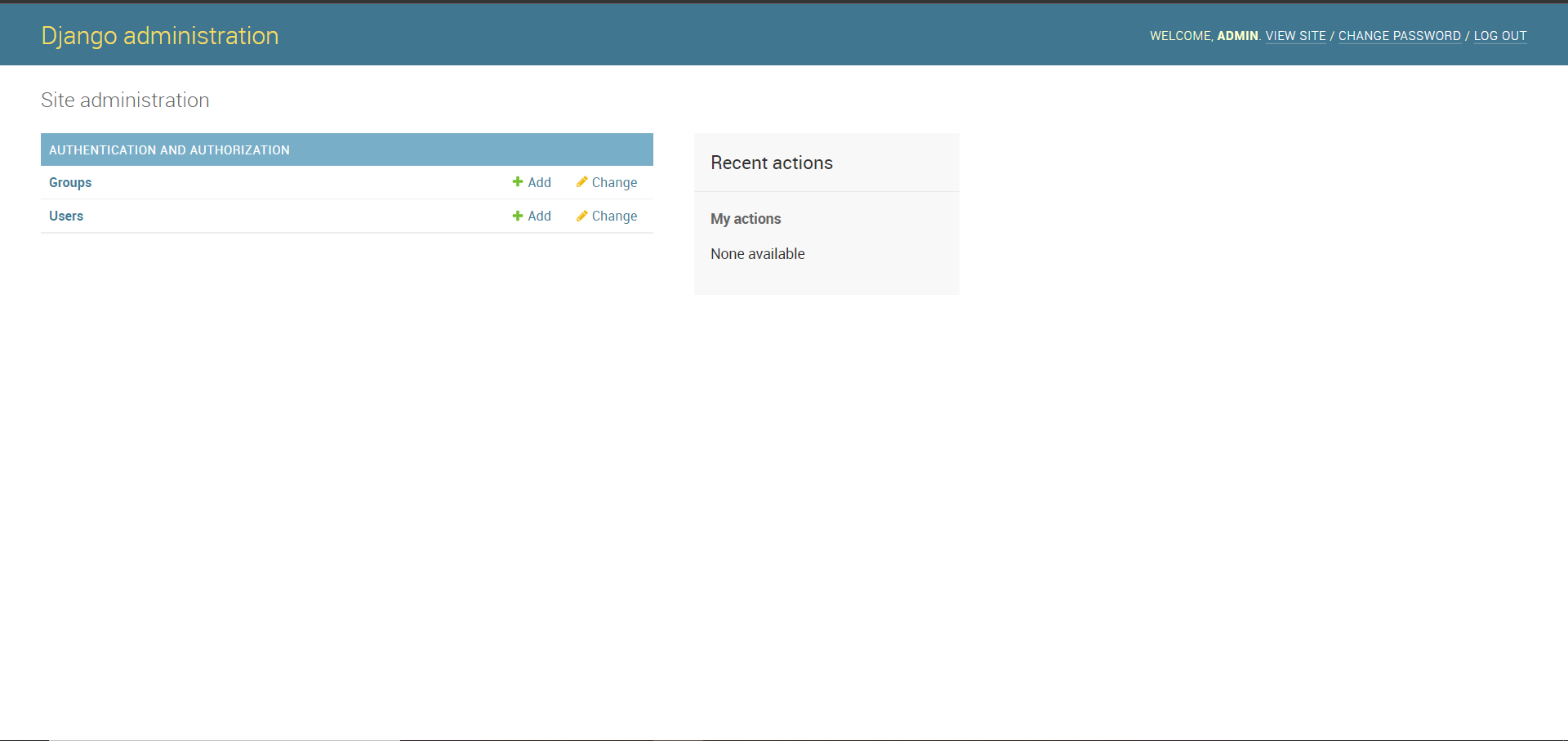
保存退出.

重新访问, 你应该看到一个如下所示的页面:



然后将/admin添加到浏览器地址栏中的URL末尾, 接着会转到管理员登录页面:

使用你通过createsuperuser命令选择的管理登录凭证, 登录到服务器. 你将有权访问管理接口:



登陆后出现此页面就成功了

测试完后, 现在可以在你的终端键入Ctrl-C停止开发服务器.

现在我们可以继续我们的第二个项目了.

**创建第二个项目:**

第二个项目将以与第一个完全相同的方式创建. 接下来将缩短这部分的解释.

返回到你的主目录并为你的新项目创建第二个虚拟环境. 激活后在这个新环境中安装Django:

cd ~

mkvirtualenv secondsite

pip3 install django

新环境将被创建并留下以前的虚拟环境. 这个Django实例与其它配置完全分开. 这使你可以独立管理它们并根据需要进行自定义.

创建第二个项目并进入项目目录:

django-admin startproject secondsite

cd ~/secondsite

初始化数据库并创建一个管理用户:

python3 manage.py migrate

## 如出现报错按前面的方法解决即可.

python3 manage.py createsuperuser

打开设置文件:

nano secondsite/settings.py

添加静态文件的位置, 就像在之前的项目中一样:

STATIC\_ROOT = os.path.join(BASE\_DIR, “static/”)

保存并关闭文件. 现在, 通过键入以下内容将静态元素收集到该目录中:

python3 manage.py collectstatic

最后, 启动开发服务器来测试网站:

python3 manage.py runserver 0.0.0.0:8000

测试网站:

<http://server_domain_or_IP:8000>

然后登录到管理网站:

<http://server_domain_or_IP:8000/admin>

如果确认所有内容都按预期工作, 请在终端中键入CTRL-C以停止开发服务器.

**下面提供一些virtualenvwrapper命令:**

退出(停止)虚拟环境:

deactivate

重新激活环境:

workon 环境名

删除虚拟环境:

rmvirtualenv 环境名

PS: 你创建Django项目需要自行使用rm 命令删除

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**本章至此结束**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_