**？：**

1. 如何使用phpmyadmin连接mysql

**注意事项：**

如果在开发过程中安装了新的组件，记得pip freeze

**建立Django的标准步骤：**

待整理 P184

**数据库的设置**

**前提：mblog为项目名。Mainsite为其中的一个应用**

>修改mainsite/models.py：定义数据模型（即数据表及其字段）。Mainsite为该项目的一个app应用。本例增加了数据表Post

>使模型生效

(VENV) ~myDjango/mblog $ python manage.py makemigrations mainsite

(VENV) ~myDjango/mblog $ python manage.py migrate

>修改mainsite/admin.py将模型在Admin管理页面中注册：将Post纳入管理，

先导入（import）Post类（即Model），然后通过admin.site.register(Post)进行注册

可参考P184

>通过Admin管理界面增加Post数据表的记录

前提：创建Admin管理界面的管理员账号和密码——

Python manage.py createsuperuser

>修改admin.py，创建“表名Admin”类并注册：

创建PostAdmin类并注册，使Admin管理界面能够显示除第一个字段外的其他字段

**读取数据库的内容并展示到页面中**

>修改mainsite/view.py

导入模型： from .models import Post

定义内容输出函数：首页函数homepage和展示具体内容函数showpost

>修改mblog/urls.py

从views.py中导入内容输出函数：from mainsite.views import homepage

在urlpatterns[ ]中，将网址和内容输出函数对应（批注A）

**模板的使用**

**练习**

>创建templates文件夹。与mblog、mainsite同级

>修改mblog/settings.py

设置模板文件夹的位置：修改TMPLATES->DIRS

（补充：如果需要将模板引擎更换成Jinja2，则可以修改‘BACKEND’）

>在url.py中创建网址和views.py中函数的对应关系。（参考批注A）

>在templates文件夹中创建.html文件

>运行程序，以objects.all()在views.py中取得数据

>以render函数（渲染函数）把数据送到指定的模板文件（如index.html）中

>将得到的结果交给HttpResponse输出给浏览器

**深入探讨**

使用模板的标准步骤 P155

引入框架时，建议使用CDN的方式。引入代码的存放位置（Bootstrap和jQuery的存放位置不一样） P157

**模板语言**

forloop.counter P169

forloop.first P188 一般用来取表头

其它{% for %} 循环的参数 P173

{% empty %} {% cycle %} P171

常用的过滤器 P173

Django.contrib.humanize模块的引入 P174

{% now %} 和 {% lorem %} 命令 P176

其它标记和特殊语句：参考官方文档

alert-warning P215、220、227、252、259（使用messages framework框架）

**共享模板的使用**

**练习**

>在template文件夹中新建base.html、header.html、footer.html。

>base.html（主要用来存放HTML框架结构和引入的框架CDN链接）

① include引入header.html和footer.html

② 创建block。此案例创建了3个block：title、headmessage、content

>其它文件，如index.html

① 继承base.html ：{% extends ‘base.html’ %}

② 不再需要<html></html>等标签，因为已经在base.html中定义过了

③ 提供base.html 中定义的区块（block）内容

**深入探讨 P163**

**CSS、JS和静态文件的使用**

**练习**

> Bootstrap的引用

将Bootstrap的CDN链接复制到base.html的</head>前面。这样继承base.html的文件中都可以使用Bootstrap的功能

> 图像文件的使用

新建static文件夹（和template、APP应用文件夹）

在static文件夹下，新建images、css、js文件夹

修改settings.py，指定静态文件要放置的位置

修改.html，加入 {% load staticfiles %}

> HTML内容处理，如在文章内容中加入图片外链 <img src=http://xxxxxx.jgp/>

在.html输出时，需要加safe过滤器：{{ post.body | safe }}

>Markdown语句解析与应用

在虚环境VENV中安装相关组件：

$ pip install django-markdown-deux

$ pip freeze > requirements.txt

修改settings.py，在INSTALLED\_APPS中，加入markdown\_deux

在.html文件中加载MarkDown语句标记：{% load markdown\_deux\_tags %}

在.html输出时，加markdown过滤器：{{ post.body | markdown }}

**深入探讨**

使用Static文件的标准步骤 P161

**数据库**

1. 注意随时makemigrations和migrate
2. 可以通过python manage.py shell来在Python的交互界面中进行数据表的操作
3. 通过 Product.objects.all()函数取得的数据，数据类型是QuerySet
4. 可通过def \_unicode\_(self)来定义实例打印的结果
5. 在models.Model中常用的数据字段格式说明 P117
6. Django ORM常用的函数以及修饰词 P125
7. 关系操作 《Python高效开发实战》 P198
8. 面向对象ORM：抽象类继承、多表继承、代理模型继承 《Python高效开发实战》 P202
9. ForeignKey的设置参数 P183
10. 跨表查询并展示 P188
11. 调整Admin管理网页的外观。包括重写Model类、排序、搜索、展示字段中文名。 P193
12. 安装Mysql数据库 195

**网址的对应与委派**

urlpatterns的Regular Expression语法说明 P145 另可见《Python高效开发实战》 P207

在URL正则中加入要传送的参数名称 P147

验证RE设计URL的正确性：<http://pythex.org/>

Views.py文件中，函数的参数可以加默认值。如def about(request,author=’0’):

——注意事项：传入的参数都是字符串，需要转换才能当做数字使用

手动传参：url(r’^$’, views.homepage, {‘testmode’:’YES’} )

include 其他整组的urlpatterns 设置 P150 另可见《Python高效开发实战》 P208

URLconf的反解功能 P151 另可见《Python高效开发实战》 P209

**网站窗体的应用**

1. HTML窗体中的常用组件 P208
2. getlist方法 P209
3. 关于删除 P216
4. 关于POST P218
5. {% csrf\_token %} P221、252
6. Form类（放在form.py中）和常见的字段格式设置方法 P222
7. {{form.as\_p}}、{{form.as\_table}}、{{form.as\_ul}} P224
8. 使用第三方服务发送电子邮件 P229
9. 模型窗体类ModelForm的应用 P233

> 新建form.py

> 在from.py中加入ModelForm类（而不在是Form类）

> 指定引用的Model和fields

1. 为窗体加上防机器人的验证机制：django-simple-captcha模块 P237

**网站的Session功能**

1. 在开发阶段复制Django网站的方法 P241
2. 检查客户端的浏览器设置是否接受Cookie P242
3. 关于Session P249
4. Username变量放在header.html中 P251
5. Cookie用response，Session用request P254
6. Django的信息显示框架messages framework P257
7. Django auth用户验证。可以不需要再自行设置Session变量 P260

——@login\_required(login\_url=’/login/’) P263

1. 增加auth.User的字段：新建Model，并和user一一对应（OneToOneField） P265
2. 应用auth用户验证存取数据库 P269

——窗体和user相结合 P269、272