Web 开发

# 环境配置

Python、Virtualenv、Django安装

# 创建Django项目和App应用

创建项目：django-admin startproject myproject

运行项目：python manage.py runserver

创建App：django-admin startapp app\_name

# 认识Django项目组成

## 重要概念

App: 是一个可以做完成某件事情的Web应用程序。一个应用程序通常由一组**models(数据库表)**，**views(视图)**，**templates(模板)**，**tests(测试)** 组成。

Project: 是配置和应用程序的集合。一个项目可以由多个应用程序或一个应用程序组成。

## Django项目组成

**manage.py**：使用**django-admin**命令行工具的快捷方式。

**\_\_init\_\_.py**：这个空文件告诉python这个文件夹是一个python包。

**settings.py**：这个文件包含了所有的项目配置。

**urls.py**：这个文件负责映射我们项目中的路由和路径。

**wsgi.py**：该文件是用于部署的简单网关接口。

**备注**：django自带了一个简单的网络服务器。在开发过程中非常方便，所以我们无需安装任何其他软件即可在本地运行项目。

## App应用组成

migrations/：在这个文件夹里，Django会存储一些文件以跟踪你在models.py文件中创建的变更，用来保持数据库和models.py的同步。

admin.py：这个文件为一个django内置的应用程序Django Admin的配置文件。

apps.py：这是应用程序本身的配置文件。

models.py：这里是我们定义Web应用程序数据实例的地方。models会由Django自动转换为数据库表。

tests.py：这个文件用来写当前应用程序的单元测试。

views.py：这是我们处理Web应用程序请求(request)/响应(resopnse)周期的文件。视图：是接收一个httprequest对象并返回一个httpresponse对象。接收request作为参数，返回一个response作为结果。

# 修改Django项目

## View

接收请求，返回响应

导入模型，读取数据，将这些数据和模型结合，输出视图

## Model

### 创建模型

在models.py文件中创建模型，每一个模型都是一个类，通过Django内部的：模型<->数据库机制，能将将模型转化为数据库

模型基本上是应用程序的数据库布局的表示；

所有的模型都是django.db.models.Model的子类。每个类将被转换为数据库表。每格字段都是由django.db.models.Field（内置的Django核心）实例表示，并将被转换为数据库列。

CharField，DateTimeField等字段都是django.db.models.Field的子类，他们包含在Django核心中准备使用。

CharField，TextField，DateTimeField和ForeignKey字段来定义我们的模型。

**创建模型之间的关系**的一种方法是使用ForeignKey字段。它将在模型之间创建一个链接，并在数据库级别创建适当的关系。ForeignKey字段的位置参数需要一个它将关联的模型。

### 迁移模型

创建当前app的状态文件：python manage.py makemigrations 这会创建一个0001\_initial.py的文件，代表应用程序的当前状态；将model转为迁移文件migration。

查看迁移过程中创建数据的指令：python manage.py sqlmigrate app\_name 0001 这将输出创建app所需的数据库的所有SQL指令。

迁移：python manage.py migrate 这将执行迁移文件（也就是app状态文件），来更新数据库。

### 对模型的API进行实验

进入shell交互式平台：python manage.py shell

创建：1）利用类来创建；

board = Board(name='Django', description='This is a board about Django.') 这将创建一个对象，但是还没保存到数据库，因此需要board.save()来保存。

2）利用Django模型的特殊属性来创建，也称该属性为模型管理器。通过python属性objects来访问模型的属性、方法；

board = Board.objects.create(name='Python', description='General discussion about Python.') 这将在数据库中创建并保存一个对象。

保存：board.save()

访问字段：board.id

更新字段：board.descritption = 'Django discussion board.'

board.save()

查询所有的模型对象：Board.objects.all() 这将返回一个QuerySet对象，它是列表类型的，每一个元素都是一个Board对象（模型对象）。

查询制定的模型对象：Board.objects.get(字段名=字段值) 这将由字段标识，获取单个对象

## Django模板引擎设置

### 创建html文件

这将创建一个基本的html文件模型，包括基本的head，body部分。

可以对这两部分修改，获得所需的视图效果；

Django中的一些基本语法可以实现动态的HTML；

将原始HTML和一些特殊的标签{% for ... in ... %}和{{ variable }}混合在一起。它们是Django模板语言的一部分。上面的例子展示了如何使用for循环遍历对象列表。在HTML模板中，{{ board.name }}显示了版块的名称，生成了动态HTML文档。

备注：要注意在setting中配置html文件的路径，打开**settings.py**，找到TEMPLATES变量，并将DIRS键设置为os.path.join(BASE\_DIR, 'templates')。

### 改进模板

使用表格来改进html模板。

根据具体的视图输出要求修改模板，使得其更加的美观，易于阅读。

## Test

测试是web开发的一个基本部分。

不管项目有多简单或多复杂，或者开发人员多老练。

每个项目都应该进行自动化测试。它将帮助我们提高自信并提供更高质量的产品。

简单的视图测试：通过reverse直接反向解析url，然后发出请求返回响应，并响应状态码。

简单的url测试，检查是否返回了正确的视图：利用了resolve函数，来将将urls.py模块中的URL列表和请求的URL相匹配。

获取详细的测试结果：可在执行测试时，设置更高的级别：python manage.py test –verbosity = 2

备注：Verbosity决定将打印到控制台的通知和调试信息的数量；0不是输出，1是正常输出，2是冗长输出。

## 静态文件设置

静态文件是CSS、javascript、字体、图像，或者我们可以用来组成用户界面的任何其他资源。

实际上，Django并不提供这些文件。除了在开发过程中，让我们的生活更轻松。但是Django提供了一些特性来帮助我们管理静态文件。这些特性在已经在INSTALLED\_APPS配置中列出的**django.contrib.staticfiles**应用程序中可用。

创建静态文件：

在项目根目录下，创建一个名为static的新文件夹，在static文件夹中创建另一个命名为css的文件夹；下载Compiled CSS and JS。在你的计算机中，解压你从网站中下载的压缩包，复制**css/bootstrap.min.css**到你的项目的css文件夹中。

在Django中配置静态文件。打开**settings.py**，滚动到文件底部，然后在STATIC\_URL后添加以下内容：STATICFILES\_DIRS = [os.path.join(BASE\_DIR, 'static'),]。

我们使用模板开头的{% load static %}来加载静态文件。模板标记{% static %}用于构成资源所在的URL。在这种情况下，{% static 'css/bootstrap.min.css' %}将返回/static/css/bootstrap.min.css

通过编辑模板，以便利用Bootstrap CSS更好的显示界面。备注：这需要对html语法以及css语法有所了解才能比较顺畅的修改编辑模板。需要了解一个网页的组成，从而能够对不同的组成进行编辑，以实现较好的显示。

## Django管理理论

当启动一个新项目时，Django已经在INSTALLED\_APPS中配置了Django Admin列表。

Django管理是一个很好的工具。但是我们需要知道什么时候使用它。

它的目的是用作博客和电子商务等内容驱动网站的后台应用程序

网站的访问者不需要使用它

**首先创建管理员账户**：python manage.py createsuperuser

然后进入管理员登录界面：输入用户名和密码登录

进入管理员界面后，可以添加用户（Users）和组（Groups）来管理权限。

**创建管理模块：**

直接在app下的admin.py文件下，添加板块即可