backend_01

> **Note**
>
> 总结

> **Info**
>
> 拓展：一些小技巧、或是可能不重要或者目前看不懂也没关系的内容

> **Question**
>
> 思考：怎么绘世呢

> **Caution**
>
> 小心：你应该注意这些内容

上学期，你亲自完成了自己的第一个HTML网页、完成了婚恋交友网页、仿淘宝网页；也许你还在大作业中尝试为网协编写自己的yuna.ysu.edu.cn，或是自己实现了一些好玩的小游戏，又或者是实现了一个无人机飞行仿真模型并以wasm的形式嵌入到网页.....

那么如何便捷地和别人分享自己的作品呢？你当然可以直接把自己的代码发给别人，不过既然是网页，最自然的当然是通过网址访问了。想象你现在有一台服务器（你的笔记本电脑也可以的），当有人通过网址访问时，服务器就将网页文件发送给他，这样他就能在浏览器上直接看到你的作品了。其中发挥接受请求和发送文件作用的就是**Web服务器软件**了。

使用nginx挂载静态网页

Web服务器软件有很多。比如Apache Http Server、Tomcat..... nginx就是一款开源的web服务器软件，具有模块化、高并发、对静态文件支持好等特点，同时也具有反向代理、负载均衡等功能.....你目前只需要知道**nginx能帮助托管你的静态网页和资源文件，并处理请求提供给他人访问**

nginx配置

安装

首先选择下载安装合适的nginx版本：[nginx: download]

修改配置文件

配置nginx：安装在任何位置，而后更改conf/nginx.conf。你只需要关心以下几行的内容

```
 http {
                // ...
         server {
        listen       8080;        // 监听端口
        server_name  localhost;
        charset utf-8;            // 字符集
        location / {
            root   html/;         // 项目根目录
            index  index.html;    // 入口文件
        }
                // ...
}
```

配置路由

通过location块，nginx可以对请求的url进行解析，实现不同的路径对应不同的资源。这里我们实现了通过不同路径访问不同的一组静态网页。

```
server {
                // ...
        location = /doge/ {
            root   html;
            index  doge.html;
        }

        location = /crp/ {
            root   html;
            index  index.html;
        }
                // ...
    }
```

或者配置多个 HTTP 端口监听

想挂载多个网页在不同的端口也当然是可行的，你只需要在conf/nginx.conf里配置多个server块即可。例如：

```
http {
        // ...
    server {
        listen       8080;
        server_name  localhost;
        charset utf-8;
        location / {
            root   html/crp;
            index  index.html;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }

        server {
        listen       8081;    // 配置一个不同的监听端口
        server_name  localhost;
        charset utf-8;
        location / {
            root   html/doge;
            index  doge.html;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
    }
        // ...
}
```

Info

全自动生成nginx配置，还有反向代理、Docker容器和全自动SSL配置等实用功能：
NGINXConfig | DigitalOcean

nginx，启动！

运行nginx

现在我们可以运行nginx了！
首先在nginx安装目录下打开终端，而后

```
start nginx
```

如果想关闭nginx，就执行

```
.\nginx.exe -s stop
```

> **Caution**
>
> 注意
>
> 在初次启动时，系统可能会提示你对nginx进行防火墙的配置，请记得允许，否则后续可能无法从外部进行访问

访问服务

最后在浏览器输入地址`127.0.0.1:[端口号]`就可以访问到你的网站了，你可以测试自己的路由配置是否正确。如果连接到了同一个局域网，你还可以尝试拿其他设备访问你的网站。

> **Caution**
>
> **注意**
>
> 在更改网页或配置后，你可能需要清除浏览器的缓存才能看到最新更改的内容。在浏览器设置中搜索“缓存”即可找到清除缓存的设置项

> **Info**
>
> **localhost和127.0.0.1和0.0.0.0**
>
> localhost是一个域名，就像ysu.edu.cn一样，只不过它仅仅在你自己的电脑上有效，打开C:\Windows\System32\drivers\etc\hosts，你会发现127.0.0.1和::1都对应localhost。其中127.0.0.1是IPv4的本地回环地址，::1是IPv6的本地回环地址。
>
> "回环"一词表示数据似乎被发送到了一个外部网络地址，但实际上数据只是在本地主机内的不同进程之间传输（而外部无法访问），你可以将localhost简单地理解成一台和正常的没什么区别的服务器，只不过它是你自己的电脑。

相比之下，0.0.0.0并非是一个具体的地址，而是一个地址集合，表示所有不清楚的主机和目的网络。

如果你将服务绑定到127.0.0.1下的端口，那么从本机以外的任何地方是无法访问的；相对的，如果绑定到0.0.0.0，那么同一网络下所有联通的计算机都能访问了（如果防火墙和其他配置允许）。

关于hosts文件：Hosts文件 - 维基百科，自由的百科全书 (wikipedia.org)
关于域名和IP地址的对应关系：域名系统 - 维基百科，自由的百科全书 (wikipedia.org)

Info

端口

这里说的是虚拟的逻辑端口（看起来就是一个0到65535的数字），用来区分一个计算机所提供的不同的服务，按照端口号范围和协议的不同有许多不同的分类。一般有许多*约定俗成*的公认端口号定义，比如22是ssh开放的端口、80是http开放的端口、443是https开放的端口。

在开发过程中，我们通常会使用8000或者8080端口来调试本地的web程序（你当然可以选择其他自己喜欢的端口号，不过小心别和其他服务的端口冲突就好）。

那么为什么要用端口号呢？个人理解是便于不同计算机之间的进程通信，比如你访问燕山大学 (ysu.edu.cn)发送请求并获得回复的过程就可以理解为你的电脑上浏览器进程和ysu服务器的web服务进程之间的通信。然而ysu服务器上有许多服务（比如数据库等等），每个进程的ID并不是固定的，你也无法得知ysu服务器上的进程ID，于是就靠端口来提供一个固定的服务接口

在PowerShell使用`netstat -ano`就可以看到当前所有连接和侦听端口的详细信息，包括它们的状态、本地和远程地址、端口号以及与每个连接相关联的进程ID

Cite

任何计算机问题均可通过增加一个间接层来解决

Any problem in computer science can be solved with another layer of indirection.
-- David Wheeler

Info

如何在局域网外部访问到你的网站?

如果你有一个公网IP，那真是太酷了！通过公网IP任何人都可以直接访问到你的服务器。你在服务商（比如阿里云）购买的ECS云服务器应该也会有自带的公网IP。

如果没有公网IP呢（大多数人可能并没有公网IP）QAQ，那么你可以尝试使用内网穿透（试试SakuraFrp吧！https://www.natfrp.com/ ）

Question

想一想

为什么局域网内的设备能访问公网资源，而公网上的设备无法直接访问局域网内部的设备？比如你可以在校外通过连接家里wifi的手机访问ysu.edu.cn或是其他任何网站，但你却不能直接连接到学校的*劳动教育实践平台*，你也不能直接访问邻居家的电脑。

搜索关键词：NAT，内网穿透，VPN

番外

Question

想一想

为什么你平时上网不需要指定端口号呢？你似乎从来不需要像`www.baidu.com:443`这样访问百度

试一试访问`ysu.edu.cn`，之后试试`ysu.edu.cn:443`和`ysu.edu.cn:80`，最后试试`ysu.edu.cn:1145`或者随便一个其他的端口

参考：深入理解什么是端口(port)-今日头条 (toutiao.com)

Cite

程序员和上帝打赌要开发出更大更好连傻瓜都会用的软件，而上帝却总能创造出更大更傻的傻瓜。目前为止，上帝赢了。

Programmers are in a race with the Universe to create bigger and better idiot-proof programs. The Universe is trying to create bigger and better idiots. So far the Universe is winning.

nginx的其它功能

作为web服务器软件，nginx的功能远不止于挂载静态网站。在最终部署Django框架的Web app时我们还需要用到它。

> Info
>
> **nginx的其它功能**
>
> - 反向代理：将客户端请求转发到内网中的多个服务器，并将响应返回给客户端
> - 负载均衡：将流量分发到多个服务器
> - HTTP缓存服务器：降低服务器负载
> - HTTPS终端：nginx支持SSL/TLS协议，可以用作HTTPS终端，对传输的数据进行加密和解密，从而保护数据的安全性
> - ...

缺了点什么？静态&动态

通过nginx，我们算是成功把网页部署到了服务器供其他人访问，不过似乎还缺了点什么...

缺了什么?

> Question
>
> 想一想你的仿淘宝网站和实际的淘宝有什么区别，比如能不能根据访问者的兴趣展示不同的商品？或者是你的小游戏能不能保存每个人的进度...

实际上，这样的静态网站并不区分不同的用户的请求，不同的用户会看到完全相同的内容；而用户在不同时间看到的网页内容也一样。例如一个用户今天访问了某个静态网页，明天他再次访问该网页时，还会看到同样的内容。

当然，如果你对网页进行了修改（比如更正错别字），用户再次访问时就可以看到修改后的新内容。但在进行下一次修改之前，它的内容都不会再改变。

它看起来像是这样的：

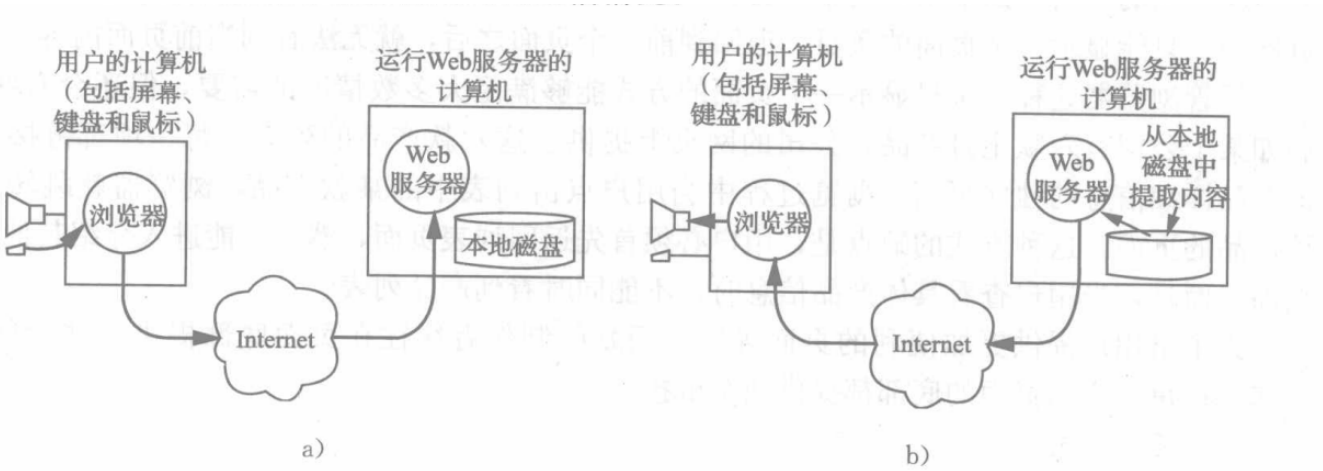


图 25-1　a）用户请求一个 URL，浏览器会通知 Web 服务器取得需要内容；
b）Web 服务器从本地磁盘上找出需要的内容，并返回给浏览器

不过这当然不能满足我们的需求，以电商网站为例，我们需要：

- 根据不同用户兴趣展示不同商品
- 用户能搜索商品
- 用户需要记录自己收藏的商品

另外，想象一下你现在有数以万计的商品信息（包括图片、价格、客户评价），为每个商品独立创建一个页面也是不现实的。而商品价格也可能随时发生变化……

我们需要什么？

- 基于*模板*动态生成不同内容的网页
- 和数据库交互，处理巨量数据
- 和用户交互，提供反馈
- ...

总之，我们需要对不同用户的不同请求进行程序处理，而后返回不同的网页，它看起来像是这样的：

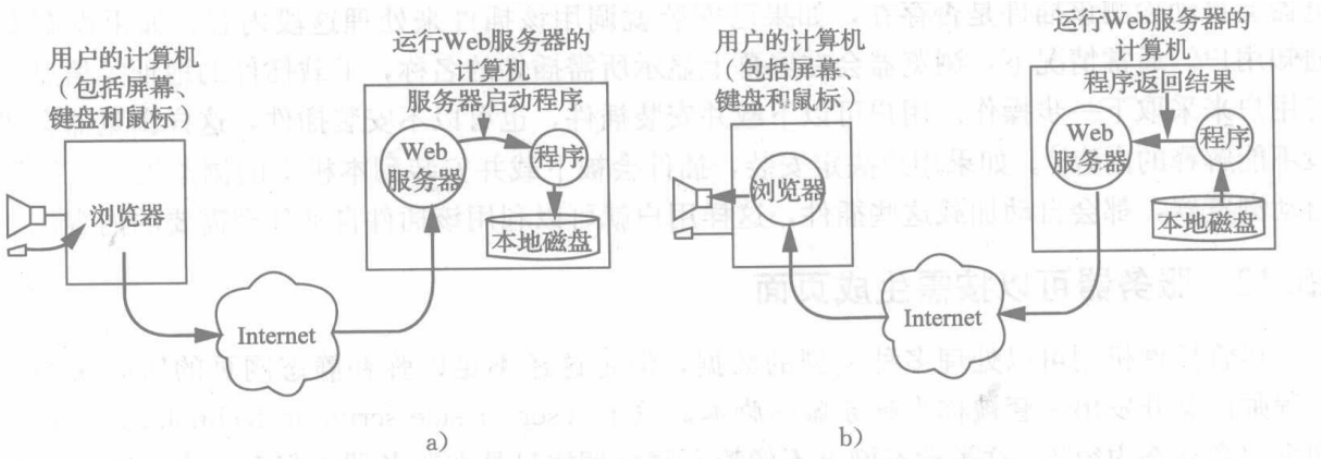


图 25-3 a）浏览器请求的 URL 对应着一段程序，因此服务器会启动这段程序；
b）Web 服务器最后将程序运行的结果返回给浏览器

这里的程序就是我们实现动态内容所需要的

> **Info**
>
> 能不能不借助后端框架自己实现?

Django，启动!

Why Django?

- 基于Python，易上手
- 文档丰富
- 提供ORM模型，不会SQL也能和数据库交互
- 自带管理后台
- ...

安装Django

python

首先确保你安装了Python，并且正确添加到了Path

而后在终端运行

```
pip install django
```

试试conda?

如果你已经安装了conda，那么就在虚拟环境内运行

```
pip install django
```

> **Caution**
>
> 不建议使用`conda install django`，它下载的版本比较旧

> **Info**
>
> **Conda**
>
> Python的版本比较多，并且它的库也非常广泛，同时库和库之间存在很多依赖关系，所以在库的安装和版本的管理上很麻烦。
>
> 而conda就是一个python环境的管理工具，支持创建多个不同的虚拟环境，每个环境就像你电脑上直接安装的Python一样，而且他们之间互相独立，可以按需激活使用某个环境。
>
> 相比于anaconda，推荐安装更轻量的miniconda
> Miniconda — Anaconda documentation

检查Django

在终端输入

```
python -m django --version
```

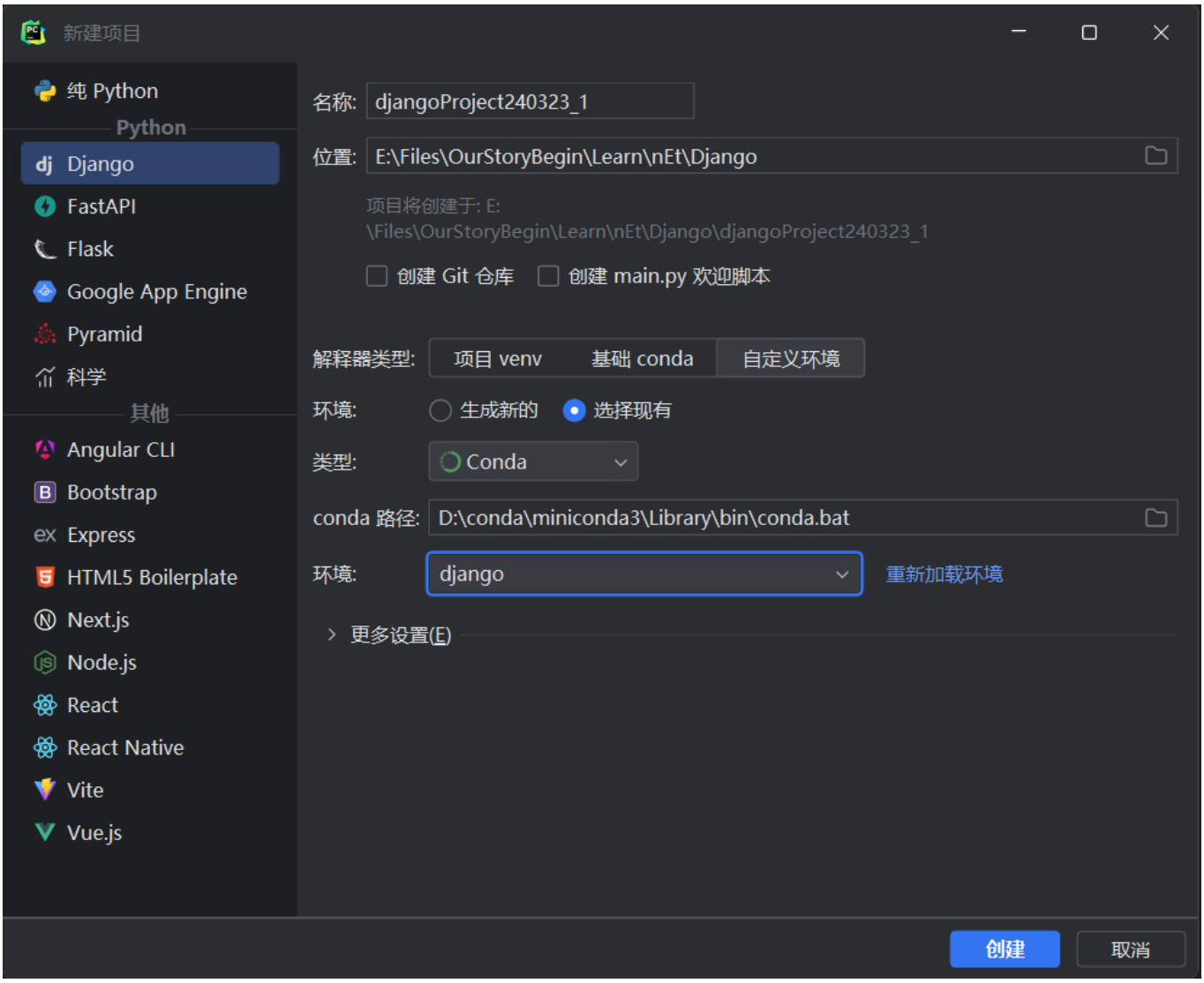
能看到输出的版本号，那么就安装成功了

如果你使用conda，记得先激活相应的虚拟环境

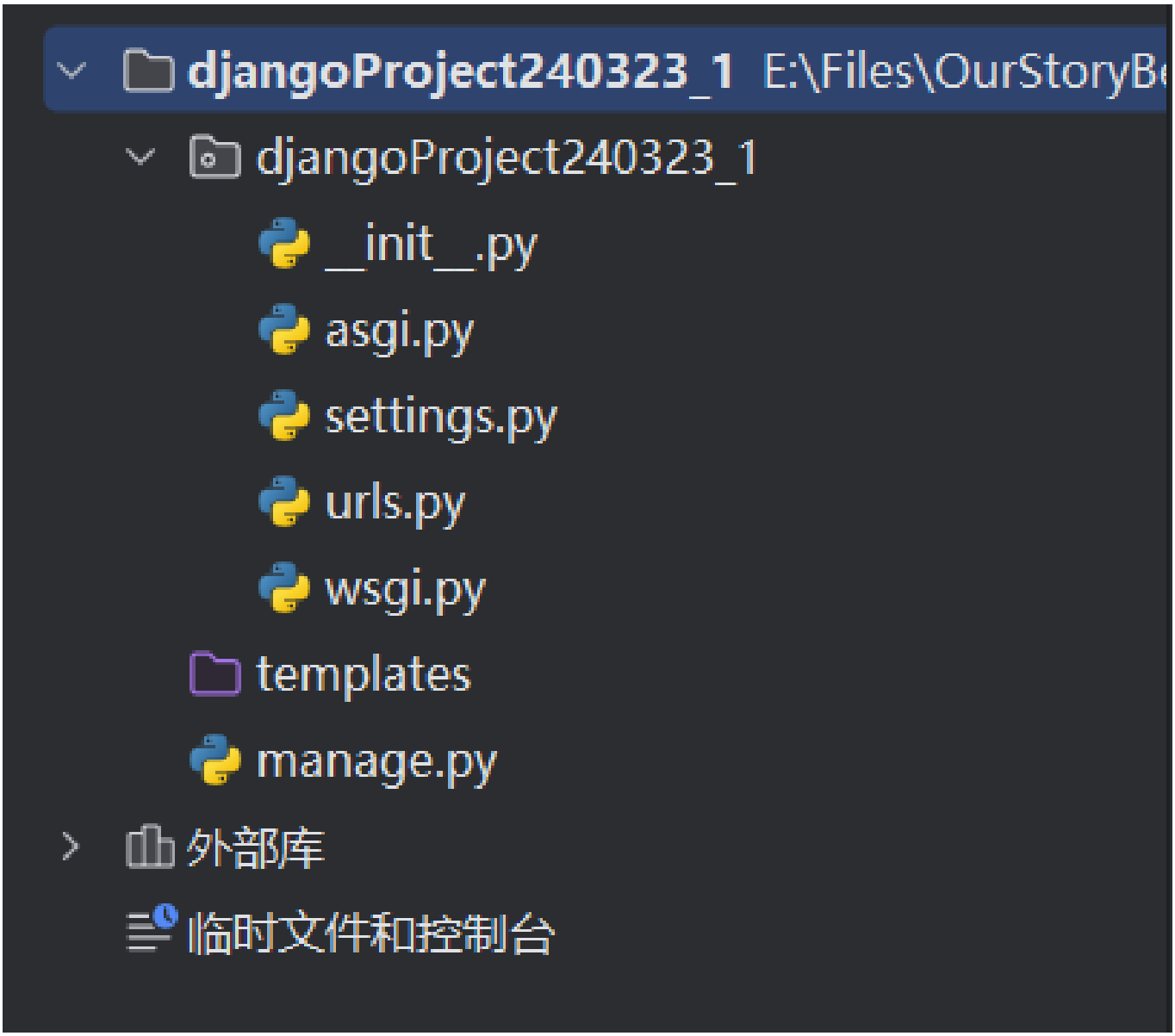
新建项目

在PyCharm Professional中新建项目

左侧选择Django，而后在右侧选择Python环境



PyCharm应该会自动生成相应的项目文件



如果没有的话，就自己在终端新建Django项目（你可能没有在PyCharm选择新建Django项目）

```
django-admin startproject [项目名]
```

启动服务器

在终端执行下面的命令

```
python manage.py runserver
```

可以看到项目在127.0.0.1:8000运行，点击即可访问

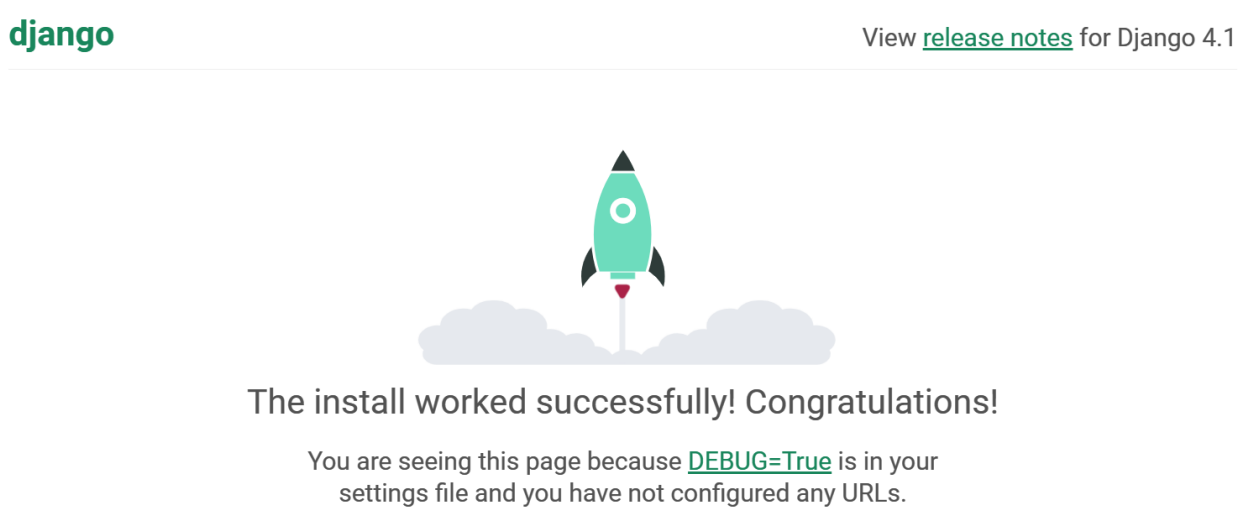
```
System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly unt
Run 'python manage.py migrate' to apply them.
March 23, 2024 - 11:41:36
Django version 4.1, using settings 'djangoProject240323_1.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

你也可以指定端口号开启服务

```
python manage.py runserver 8001
```

看到这个就说明你的Django应用成功运行了！



Question

试一试

runserver默认将服务绑定到了127.0.0.1上，试试在同一网络下用手机能否访问？

然后改成以`python manage.py runserver 0.0.0.0:8000`重新运行，再试试能否访问？如果能访问，上面显示了什么？

> **Info**
>
> 关于这里启动的服务器
>
> 这里runserver启动的是Django自带的测试服务器，它只适合在本地开发和调试网页使用，性能较差，也不够安全可靠，不适合*生产环境*。实际部署时，我们会使用nginx+uWSGI+Django

Django概述

项目文件

- 最外层的根目录只是项目的容器， 根目录名称对 Django 没有影响。
- `manage.py`: 一个管理 Django 项目的命令行工具。
- 里面一层的 `[项目名]/` 目录包含你的项目，它是一个纯 Python 包。它的名字就是当你引用它内部任何东西时需要用到的 Python 包名。 (比如 `[项目名].urls`).
- `[项目名]/settings.py`： Django 项目的配置文件。
- `[项目名]urls.py`： Django 项目的 URL 声明，就像你网站的"目录"。阅读 URL调度器 文档来获取更多关于 URL 的内容。

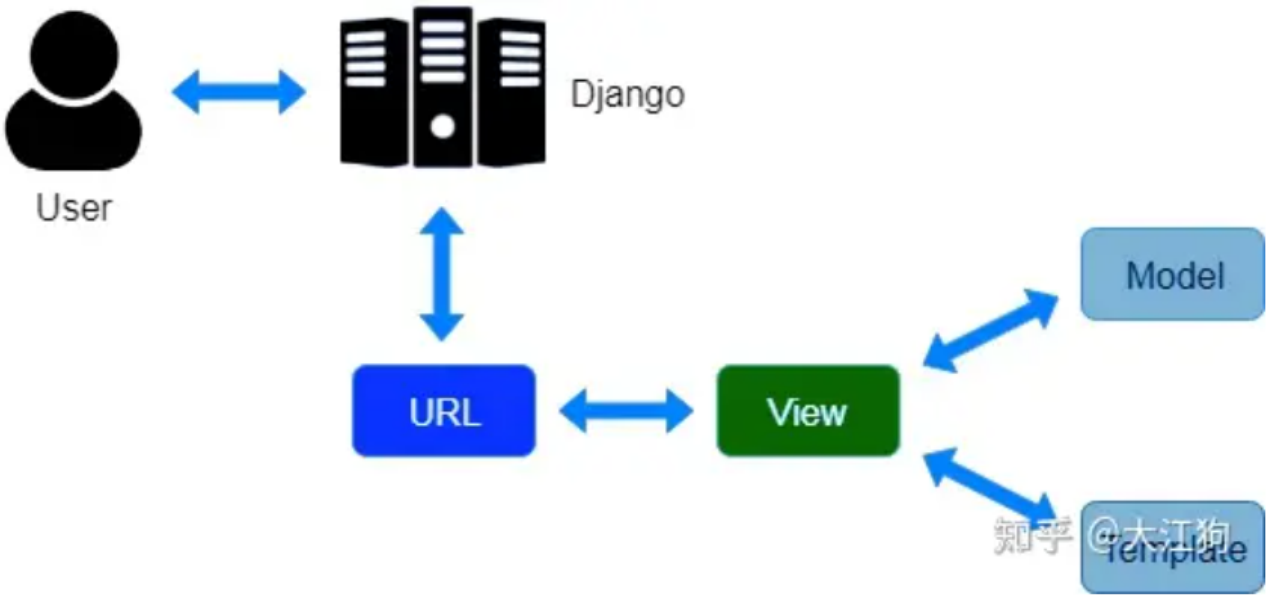
Django的设计模式

Django是一个遵循MVT模式，即Model-View-Template（模型-视图-模板）的框架。

你可以暂时简单理解为：模型是一类数据，模板是一个待动态填充内容的网页，用户发起请求，框架就根据模型和模板生成指定的视图返回给用户

以淘宝网站举例，我现在要访问某个商品的具体信息页，点击链接后会向淘宝服务器发送请求，收到请求后，后端框架就会根据数据库中的商品信息（模型）和商品详情页的模板生成这个商品的具体信息页面（视图）

这三者分别对应了单个app目录下的models.py, views.py和templates文件夹



创建第一个app

app和项目有什么关系，为什么还要创建app

app是一个专门做某件事的网络应用程序——比如博客系统，或者公共记录的数据库，或者小型的投票程序。项目则是一个网站使用的配置和app的集合。项目可以包含很多个app。app可以被很多个项目使用。这也体现了Django模块化的特点。

创建app

请确定你现在处于 `manage.py` 所在的目录下，然后运行这行命令来创建一个应用：

```
py manage.py startapp [app名]
```

> **Info**
>
> 此时我们创建的app会和项目文件夹处在同级目录，如果在项目文件夹内创建，app就成为了项目的一个子模块。你可能需要根据实际情况按需决定app和项目之间的关系

编写一个视图

打开 `[app名]/views.py`，把下面这些代码输入进去：

```
from django.http import HttpResponse
```

```
def index(request):
    return HttpResponse("Hello, world. You're at the [app名] index.")
```

将url映射到视图

这是 Django 中最简单的视图。如果想看见效果，我们需要将一个 URL 映射到它——这就是我们需要 URLconf 的原因了。

要在app目录中创建一个 URL 配置，你需要自己创建一个名为 `urls.py` 的文件。

在创建的`urls.py`中，输入如下代码：

```
from django.urls import path

from . import views

urlpatterns = [
    path("", views.index, name="index"),
]
```

下一步是要在根 URLconf 文件中指定我们创建的 `hw01.urls` 模块。在 `[项目名]/urls.py` 文件的 `urlpatterns` 列表里插入一个 `include()`， 如下：

```
from django.contrib import admin
from django.urls import include, path

urlpatterns = [
    path("[app名]/", include("[app名].urls")),
    path("admin/", admin.site.urls),
]
```

这里include的作用就是引用其他的urls文件，将匹配后的剩余部分交给它继续进行匹配。

现在我们尝试访问`127.0.0.1:8000/[app名]`，可以看到路由配置是成功的！

Hello, world. You're at the hw index.

Info

关于path的参数：编写你的第一个 Django 应用，第 1 部分 | Django 文档 | Django (djangoproject.com)

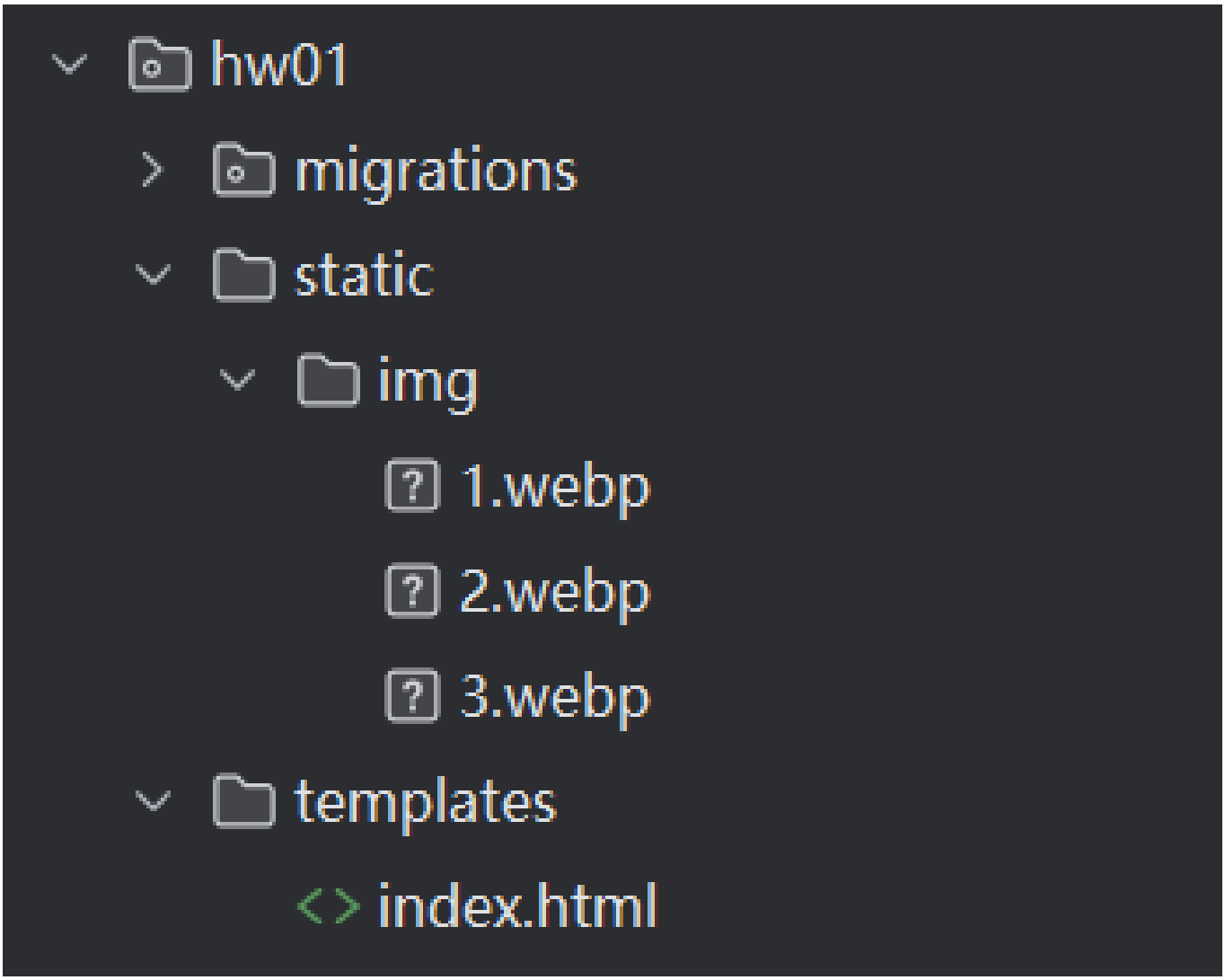
这部分的参考：编写你的第一个 Django 应用，第 1 部分 | Django 文档 | Django (djangoproject.com)

使用html模板和静态资源

他们放在哪里?

一般静态资源指的是图片、css、js和其他插件，会放在static文件夹下的对应文件夹（img、CSS、js、plugins）中；html文件会放在templates文件夹下

app内的这些目录需要你自己创建，目录看起来像是这样：



如何使用静态资源 - 模板语法

首先你需要在项目的settings.py中注册app，否则app下的templates文件夹是无效的

```
# ...
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    '[app名].apps.[app名]Config',  # 会自动补全
]
# ...
```

而后，你需要在模板html文件开头中添加下面一行来加载静态资源：

```
{% load static %}
```

之后，你需要将所有对图片的引用地址改成模板的形式，比如：

```
<img src="{% static 'img/1.webp' %}" width="200"id="#拉布拉多"/>
```

类似的，css文件和js文件也需要这样加载

预览：接下来你将学到什么

目前，我们知道了为什么需要后端框架，以及如何创建一个简单的Django项目。之后我们还将学习后端程序如何与数据库进行交互，如何在后端对数据进行处理等内容；此外我们也会关注一些安全问题，比如如何防止SQL注入、如何阻止CSRF攻击等等。

在完成app的开发后，我们还将学习如何部署项目到服务器（如果有时间TT）

> **Info**
>
> **学习Django**
>
> - Django 文档 | Django 文档 | Django (djangoproject.com)
> - Django Web 框架 (python) - 学习 Web 开发 | MDN (mozilla.org)
> - https://www.bilibili.com/video/BV1NL41157ph （版本较旧，仅供参考）

作业：

创建一个Django项目来展示你上学期制作的网页（或是其他的静态网页）