Django项目

概述

Django采用MVC架构设计的开源的WEB快速开发框架。

优点：

- MVC设计模式
- 大而全的重框架，自带ORM、Template、Form、Auth核心组件，便于快速开发
- 简洁的url设计
- 实用的管理后台Admin
- 周边第三方插件丰富

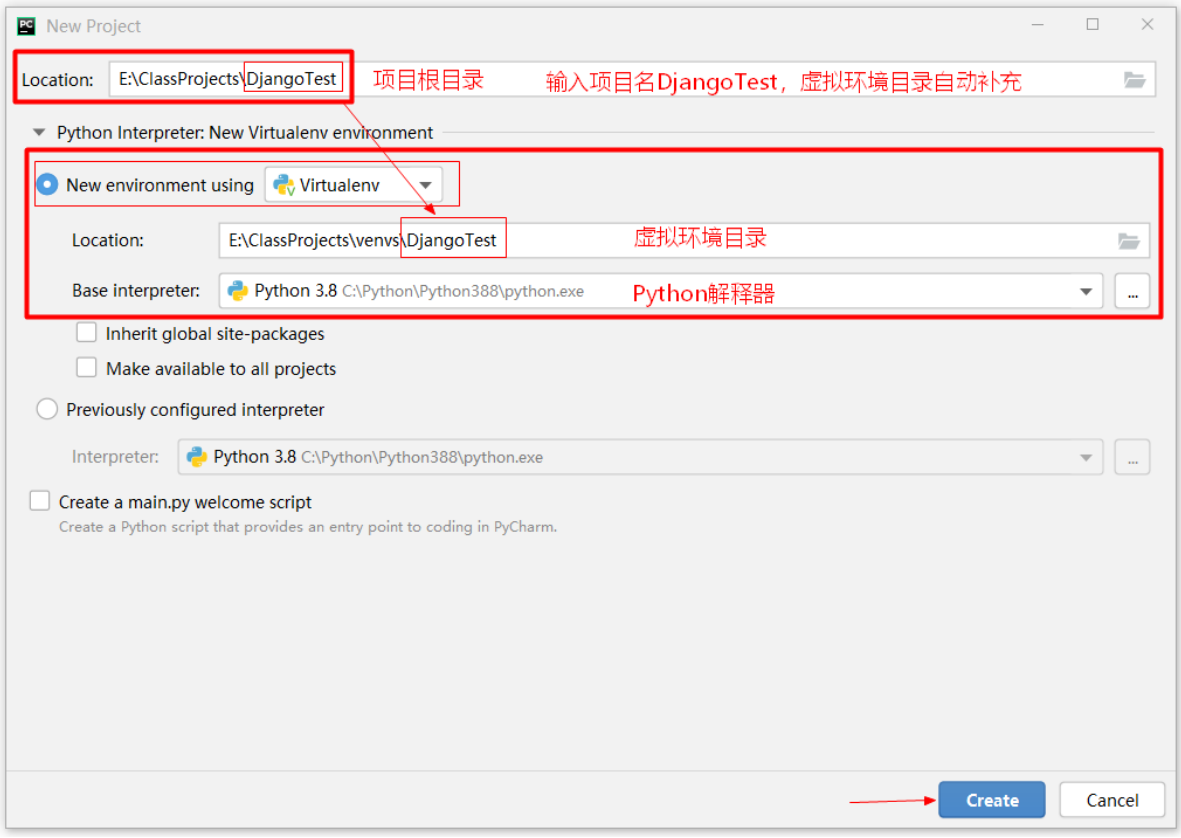
缺点：框架重、同步阻塞

Django的设计目标就是一款大而全，便于企业快速开发项目的框架，因此企业广泛采用。

项目构建

启动Pycham，点击`Create New Project`。选择建立新的虚拟环境（每个项目独立开发，所以需要独立的虚拟环境），选择基于开发的解释器版本。

本次开发依然采用Pycharm社区版，看似缺少了一些功能，但实际开发足够了。



安装Django

Django的安装参考 https://www.djangoproject.com/download/

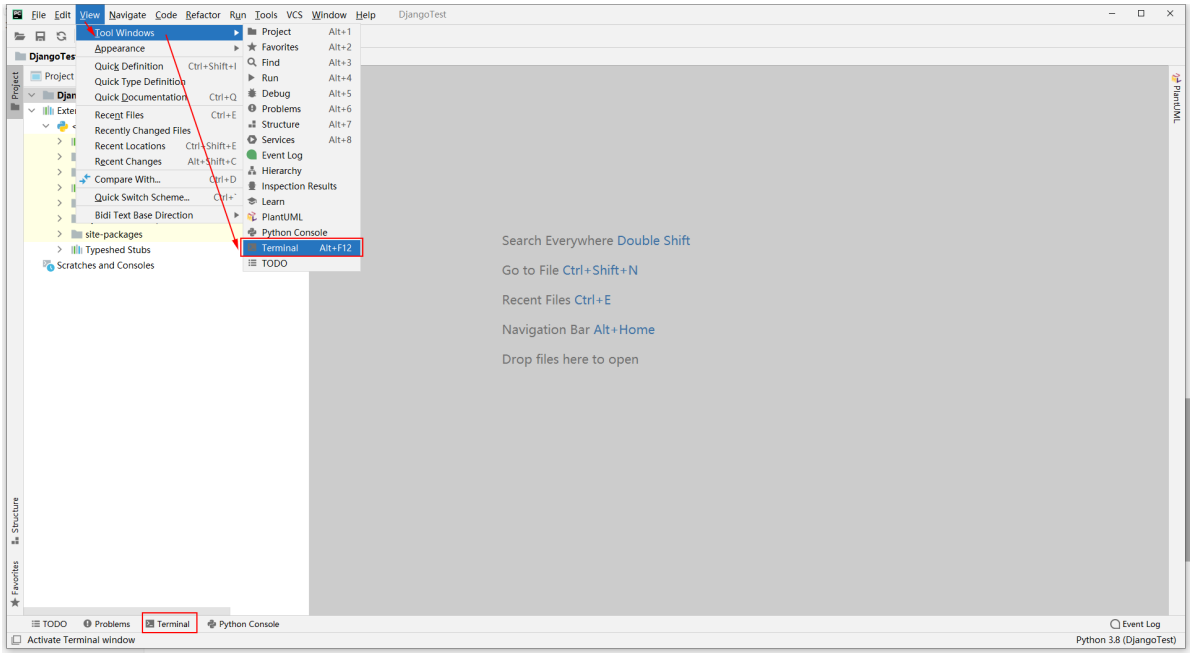
Python版本依赖，参考 https://docs.djangoproject.com/en/3.2/faq/install/#what-python-version-can-i-use-with-django

Django version	Python版本
1.8	2.7, 3.2 (until the end of 2016), 3.3, 3.4, 3.5
1.9, 1.10	2.7, 3.4, 3.5
1.11(LTS)	2.7, 3.4, 3.5, 3.6, 3.7 (added in 1.11.17)
2.0	3.4, 3.5, 3.6, 3.7
2.1	3.5, 3.6, 3.7
2.2(LTS)	3.5, 3.6, 3.7, 3.8(2.2.8), 3.9 (added in 2.2.17)
3.0	3.6, 3.7, 3.8, 3.9 (added in 3.0.11)
3.1	3.6, 3.7, 3.8, 3.9 (added in 3.1.3)
3.2(LTS)	3.6, 3.7, 3.8, 3.9

长期支持版本是企业的选择，由于最新版是3.2是LTS，本次采用它。

Django3.x兼容2.x版本。相比于2.x版本

- 支持MariaDB 10.1+
- 开始支持ASGI，官方建议，但目前可以研究，暂时不要切换
- 对内建模块django.contrib.admin、django.contrib.auth一些改动
- Model中一些增强



菜单`view/Tool Windows/Terminal`打开命令行，其运行当前工作路径正好是项目根目录，而且`(DjangoTest) E:\ClassProjects\DjangoTest>`中`(DjangoTest)`说明使用了虚拟环境。就在这里安装Django。

注意：本文如若未特殊声明，所有的命令操作都在项目根目录下

```
$ pip install django
Installing collected packages: sqlparse, pytz, asgiref, django
Successfully installed asgiref-3.4.1 django-3.2.6 pytz-2021.1 sqlparse-0.4.1
```

脚手架构建

Django安装完，提供了一个命令`django-admin`，它实际是虚拟环境路径中，`Lib/site-packages/django/bin`下的`django-admin.py`。

```
django-admin

Type 'django-admin help <subcommand>' for help on a specific subcommand.

Available subcommands:

[django]
    makemigrations
    migrate
    runserver
    startapp
    startproject

$ django-admin help startproject
```

上面这些都是最常用的命令，这里只需要使用`startproject`来构建一个Django项目目录结构和基础文件。

```
$ django-admin startproject salary .
```

构建一个项目叫做blog，注意最后有个**点**，表示在当前目录即项目根目录创建。

```
项目根目录
├─ manage.py
└─ salary
    ├─  settings.py
    ├─  urls.py
    ├─  wsgi.py
    └─  __init__.py
```

重要文件说明

- manage.py：本项目管理的命令行工具。应用创建、数据库迁移等都使用它完成
- **salary/settings.py**：本项目的全局核心配置文件
  - 应用、数据库配置
  - 模板、静态文件
  - 中间件、日志
  - 第三方插件配置
- **blog/urls.py**：URL路径映射配置。项目初始，只配置了/admin的路由。
- blog/wsgi：定义WSGI接口信息。部署用，一般无需改动。

MySQL数据库驱动

Django支持MySQL 5.5+

Django官方推荐使用本地驱动mysqlclient 1.3.7 +

```
$ pip install mysqlclient
```

创建应用

创建应用`employee`

```
$ python manage.py startapp employee
```

创建应用后，项目根目录下产生一个employee目录，有如下文件：

- admin.py：应用后台管理声明文件
- **models.py**：模型层Model类定义
- **views.py**：定义URL响应函数或类
- migrations包：数据迁移文件生成目录
- apps.py：应用的信息定义文件

配置

salary/settings.py是全局配置文件

注册应用

注册应用，可以做迁移migrate、做后台管理Admin等，一般建议注册。

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'employee',
]

```

数据库配置

用数据库，需要修改默认的数据库配置。

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'test',
        'USER': 'wayne',
        'PASSWORD': 'wayne',
        'HOST': '192.168.142.140',
        'PORT': '3306',
    }
}
```

配置项	说明
HOST	数据库主机。缺省是空字符串，代表localhost。如果是'/'开头表示使用Unix Socket连接
POST	端口
USER	用户名
PASSWORD	密码
NAME	库名
OPTIONS	选项，字典类型，参考MySQL文档

数据库引擎ENGINE
内建的引擎有

- 'django.db.backends.postgresql'
- 'django.db.backends.mysql'
- 'django.db.backends.sqlite3'
- 'django.db.backends.oracle'

本地化和时区

```
LANGUAGE_CODE = 'zh-Hans' #'en-us' 中文简体

USE_TZ = True
TIME_ZONE = 'Asia/Shanghai' #'UTC'
```

日志

Django的日志配置在settings.py中。

```
LOGGING = {
    'version': 1,
    'disable_existing_loggers': False,
    'handlers': {
        'console': {
            'class': 'logging.StreamHandler',
        },
    },
    'loggers': {
        'django.db.backends': {
```

```
            'handlers': ['console'],
            'level': 'DEBUG',
        },
    },
}
```

配置后，就可以在控制台看到执行的SQL语句。

注意，settings.py中必须**DEBUG=True**，同时loggers的**level是DEBUG**，否则从控制台看不到SQL语句。

Django内建loggers可以参考https://docs.djangoproject.com/en/3.2/topics/logging/#django-db-backends

迁移

迁移：指的是把Django中定义的Model类和属性，转换成数据库中表和字段的过程。

迁移一般需要两个过程：

1. 制作迁移文件
2. 迁移

Django内部也有应用，它们也需要表。这些表的迁移文件已经生成了，只需要迁移。

```
$ python manage.py migrate
```

迁移后，数据库中产生下面这些表：auth_group, auth_group_permissions, auth_permission, auth_user, auth_user_groups, auth_user_user_permissions, django_admin_log, django_content_type, django_migrations, django_session。

运行

```
$ python manage.py runserver
```

访问http://127.0.0.1:8000/即可。

这就是一个Django项目构建基本流程，之后项目构建方式、配置都大同小异。