03 Django模板

模板Template

在Django框架中，模板是可以帮助开发者快速生成呈现给用户页面的工具
模板的设计方式实现了我们MVT中VT的解耦(M：Model， V:View，T:Template)，VT有着N:M的关系，一个V可以调用任意T，一个T可以供任意V使用　　(MVC：Model,View界面，Controller控制器)

模板处理分为两个过程
　加载HTML
　渲染数据　render()
模板主要有两个部分
　HTML静态代码
　模板语言，动态插入的代码段（挖坑，填坑）
模板中的动态代码段除了做基本的静态填充，还可以实现一些基本的运算，转换和逻辑

静态页面：页面数据是本地固定的
动态页面：页面数据来源于后台服务器

模板中的变量：视图传递给模板的数据，遵守标识符规则
　语法：　{{ var }}
　如果变量不存在，则插入空字符串

方法不能有参数。
```
  {{ str }}
    {{ str.upper }}
    {{ str.isdigit }}
  {{ dict.key }}
```
列表，使用索引，不允许负索引
```
  items= ['apples', 'bananas', 'carrots']
  {{ items.2 }}
```

模板中的标签
语法 {%　tag　%}
作用
1. 加载外部传入的变量
2. 在输出中创建文本
3. 控制循环或逻辑

```
if 语句:
  格式：
  if单分支
    {% if　表达式 %}
        语句
    {% endif　%}

  if双分支
```

```
  {%  if 表达式 %}
      语句
  {% else  %}
      语句
  {% endif %}
```

if多分支

```
  {% if 表达式 %}
        语句
  {% elif 表达式 %}
        语句
      {% else  %}
      语句
  {% endif %}
```

判断true或false

```
  {% if today_is_weekend %}
    <p>Welcome to the weekend!</p>
  {% endif %}
```

使用 and or not

```
  {% if athlete_list and coach_list %}
      <p>Both athletes and coaches are available.</p>
  {% endif %}

  {% if not athlete_list %}
      <p>There are no athletes.</p>
  {% endif %}

  {% if athlete_list or coach_list %}
      <p>There are some athletes or some coaches.</p>
  {% endif %}
```

使用 in和not in,

```
  {% if "bc" in "abcdef" %}
    This appears since "bc" is a substring of "abcdef"
  {% endif %}
  {% if user not in users %}
    If users is a list, this will appear if user isn't an element of the list.
  {% endif %}
```

```
for 语句:
  {% for 变量 in 列表 %}
    语句1
  {% empty %}
    语句2
  {% endfor %}
  当列表为空或不存在时,执行empty之后的语句
```

```
{{ forloop.counter }} 表示当前是第几次循环，从1数数
{% for item in todo_list %}
    <p>{{ forloop.counter }}: {{ item }}</p>
{%endfor %}

{{ forloop.counter0}}表示当前是第几次循环，从0数数
{{ forloop.revcounter}}表示当前是第几次循环，倒着数数，到1停
{{ forloop.revcounter0}}表示当前第几次循环，倒着数，到0停
{{ forloop.first }} 是否是第一个  布尔值

  {% for object in objects %}
    {% if forloop.first %}
        <li class="first">
    {% else %}
        <li>
    {% endif %}
    {{ object }}</li>
{% endfor %}

{{ forloop.last }} 是否是最后一个 布尔值
{% for link in links %}
  {{ link }}{% if not forloop.last %} | {% endif %}
{% endfor %}

forloop.parentloop
{% for country in countries %}
  <table>
      {% for city in country.city_list %}
      <tr>
          <td>Country #{{ forloop.parentloop.counter }}</td>
          <td>City #{{ forloop.counter }}</td>
          <td>{{ city }}</td>
      </tr>
      {% endfor %}
  </table>
 {% endfor %}

注释:
  单行注释
  {#  被注释掉的内容  #}

  多行注释
  {% comment %}
    内容
  {% endcomment %}

过滤器:
  {{ var|过滤器 }}
```

作用：在变量显示前修改

add {{ value|add:2 }}
没有减法过滤器，但是加法里可以加负数
{{ value|add:-2 }}
lower
{{ name|lower }}
upper
{{ my_list|first|upper }}
截断：
{{ bio|truncatechars:30 }}
过滤器可以传递参数，参数需要使用引号引起来
比如join： {{ students|join:'=' }}

默认值:default，格式 {{var|default:value}}
如果变量没有被提供或者为False，空，会使用默认值

根据指定格式转换日期为字符串，处理时间的
就是针对date进行的转换
{{ dateVal | date:'y-m-d' }}

HTML转义
将接收到的数据当成普通字符串处理还是当成HTML代码来渲染的一个问题
渲染成html：

```
{{ code|safe }}
```

关闭自动转义

```
{% autoescape off%}
    code
{% endautoescape %}
```

打开自动转义转义

```
{% autoescape on%}
    code
{% endautoescape %}
```

模板继承
block：

```
{% block XXX%}
  code
{% endblock %}
```

extends 继承，写在开头位置

```
{% extends '父模板路径' %}
```

include：加载模板进行渲染

```
{% include '模板文件' %}
```

{{ block.super }} ：获取父模板中block中的内容

在Django项目中使用Jinja2模板引擎

Jinja2是之前我们在Flask框架讲过的一个模板引擎，是模仿Django默认模板引擎基础上开发的，比Django模板引擎性能更好，功能更全.

> jinja2宣称比django默认模板引擎快10-20倍。Django也支持jinja2

1.安装jinja2模块

```
pip install jinja2
```

2.在settings.py所在目录中创建jinja2_env.py文件，并写入以下内容

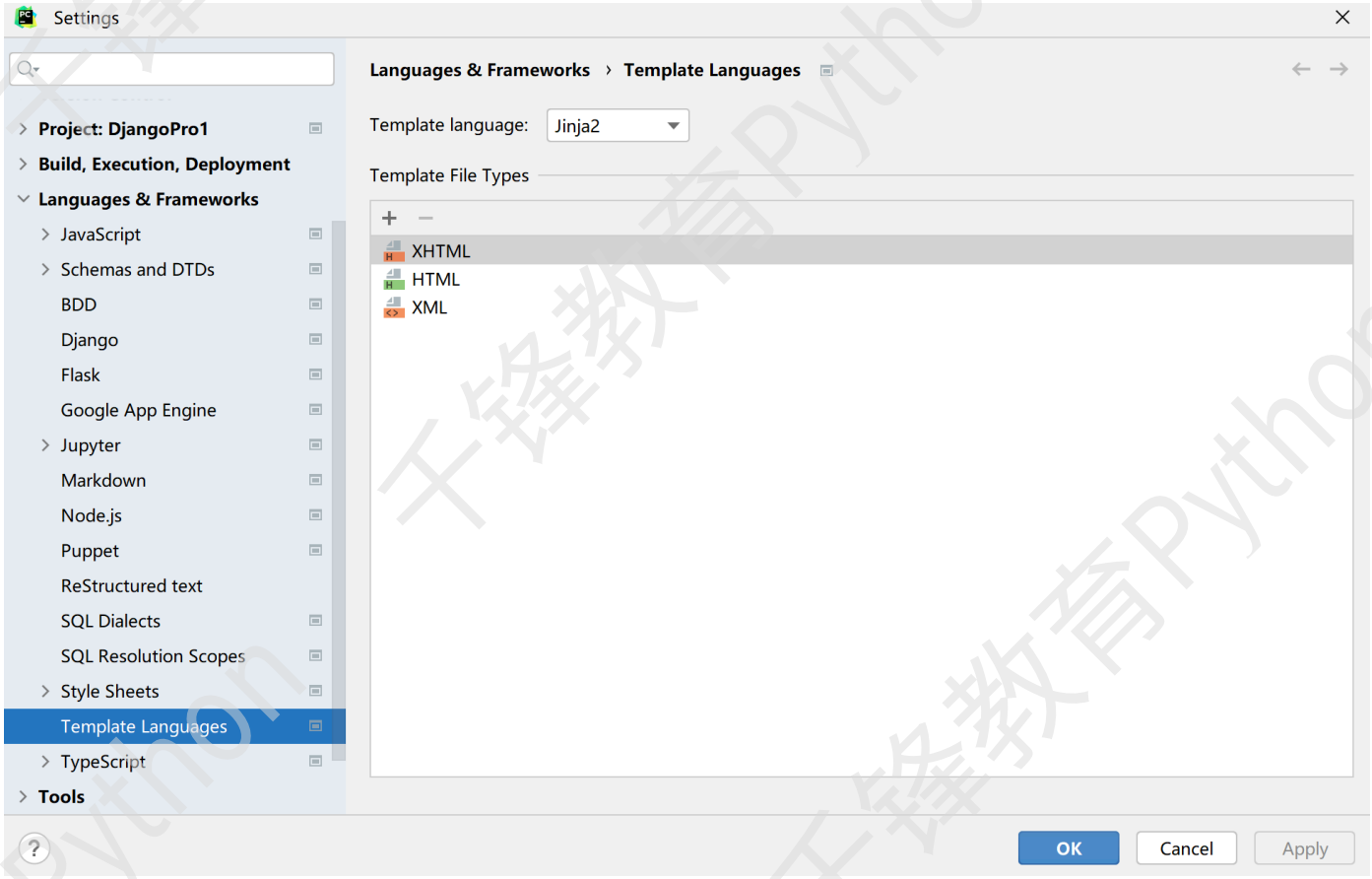
```
from django.templatetags.static import static
from django.urls import reverse
from jinja2 import Environment

def environment(**options):
    env = Environment(**options)
    env.globals.update({
        'static': static,
        'url': reverse,
    })
    return env
```

3.在settings.py文件中配置Jinja2模板引擎

```
TEMPLATES = [
    {
        # 添加Jinja2模板
        'BACKEND': 'django.template.backends.jinja2.Jinja2',
        'DIRS': [BASE_DIR / 'templates'],
        'APP_DIRS': True,
        'OPTIONS': {
            # 这里要添加environment,并指定到jinja2_env文件中的environment
            'environment': 'DjangoPro2.jinja2_env.environment',
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
```

```
    },
    # 保留原来有的Django模板
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [BASE_DIR / 'templates'],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

4.在修改Pycharm中settings的模板语言为Jinja2



5.在HTML模板中要使用Jinja2语法

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
```

```
</head>
<body>
    <h2>Jinja2模板语法</h2>
    <hr>

    <p>{{ name }}</p>
    {% for n in name %}
        <div>
            <b>{{ n }} - {{ loop.index }}</b>
        </div>
    {% endfor %}



</body>
</html>
```