模板层Template

在Django中，将前端的内容定义在模板中

模板是纯文本文件，可以生成任何基于文本的文件格式，比如HTML，XML，CSV等。

模板包含所需HTML页面的静态部分，以及一些特殊的模版语法，用于将动态内容插入静态部分

模板层就是如何往HTML文件中填入动态内容的系统

模板语言简介

Django可以配置一个或多个模板引擎（语言），也可以不用引擎。

Django自带一个称为DTL（Django Template Language ）的模板语言

另外一种流行的Jinja2语言(需要提前安装，pip install Jinja2)

配置模板settings.py

TEMPLATES

模板语言

DTL　　'BACKEND': 'django.template.backends.django.DjangoTemplates',

Jinja2　　'BACKEND': 'django.template.backends.jinja2.Jinja2'

模板文件路径　　'DIRS': [os.path.join(BASE_DIR, 'templates')]

是否进入已注册的应用中的templates文件夹查找模板选项　　'APP_DIRS': True,

示例

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [
            '/home/html/example.com',
            '/home/html/default',
        ],
    },
    {
        'BACKEND': 'django.template.backends.jinja2.Jinja2',
        'DIRS': [
            '/home/html/jinja2',
        ],
    },
]
```

模板文件加载顺序

首先去配置的模板目录下面去找模板文件

去INSTALLED_APPS下面的每个应用的templates去找模板文件，前提是应用中必须有templates文件夹

模板变量

{{变量}}

解析顺序

{{ book.btitle }}

1）首先把book当成一个字典，把btitle当成键名，进行取值book['btitle']

2）把book当成一个对象，把btitle当成属性，进行取值book.btitle

3）把book当成一个对象，把btitle当成对象的方法，进行取值book.btitle

如果解析失败，则产生内容时用空字符串填充模板变量

模板标签

{%代码段%}

模版语言中的标签类似Python中的函数

可以输出内容、控制结构，甚至可以访问其他的模板标签

常见标签

for标签

```
{%for item in 列表%}
循环逻辑
{{forloop.counter}}表示当前是第几次循环，从1开始
{%empty%}
列表为空或不存在时执行此逻辑
{%endfor%}
```

if标签

```
{%if ...%}
逻辑1
{%elif ...%}
逻辑2
{%else%}
逻辑3
{%endif%}
```

比较标签

运算符左右两侧不能紧挨变量或常量，必须有空格

==

!=

<

>

<=

>=

布尔运算

and

or

not

block和extends标签　　继承和复写模版

其他内建标签　　http://python.usyiyi.cn/translate/django_182/ref/templates/builtins.html

自定义模板标签

simple_tag

```
from django import template

register = template.Library()

@register.simple_tag
def current_time(format_string):
    return datetime.datetime.now().strftime(format_string)
```

inclusion_tag

模板继承

典型应用：网站的头部、尾部信息。

模版继承允许用户创建一个包含基本“骨架”的父亲模版，它包含站点中的共有元素，并且可以定义能够被子模版覆盖的blocks。

父模板

标签block

用于在父模板中预留区域，留给子模板填充差异性的内容，名字不能相同

```
{%block 名称%}
预留区域，可以编写默认内容，也可以没有默认内容
{%endblock  名称%}
```

子模板

标签extends

继承，写在子模板文件的第一行

{% extends "父模板路径"%}

填充父模板预留的某个区域

```
{%block 名称%}
实际填充内容
{{block.super}}用于获取父模板中block的内容
{%endblock 名称%}
```

模板过滤器

过滤器用于修改变量或标签参数的值

变量|过滤器:参数

常见过滤器

length

返回字符串包含字符的个数，或列表、元组、字典的元素个数

default

如果变量不存在时则返回默认值

data|default:'默认值'

date

用于对日期类型的值进行字符串格式化

- Y表示年，格式为4位，y表示两位的年。
- m表示月，格式为01..02..12等。
- d表示日，格式为01..02等。
- j表示日，格式为1..2等。
- H表示时，24进制，h表示12进制的时。
- i表示分，为0-59。
- s表示秒，为0-59。

```
value|date:"Y年m月j日  H时i分s秒"
```

其他过滤器　　http://python.usyiyi.cn/translate/django_182/ref/templates/builtins.html

自定义过滤器

场景　　例如，求余操作没有存在的现成过滤器，需要自己定义

配置使用过程

1.在应用中创建名为templatetags的python包

2.在该目录下创建filters.py文件或其他名字的文件

```
#导入Library类
from django.template import Library

#创建一个Library类对象
register=Library()

#使用装饰器进行注册
@register.filter
#定义求余函数mod，将value对2求余
def mod(value):
    return value%2 == 0
```

3.模板中使用自定义过滤器

首先使用load标签引入模块　　{%load filters%}

使用自定义过滤器　　{%变量名 | 自定义过滤器名：参数%}

注释

单行　　{#...#}

多行　　在 {% comment %} 和 {% endcomment %}，之间的内容会被忽略，作为注释

自动HTML转义

原因

用户提交的数据都被不应该被盲目的信任，并且被直接插入到网页中

危害示例

模板片段　　Hello, {{ name }}

如果用户提交的name数据包含一个 '' 符号（比如下面这样），会发生什么呢

如：username

这将会导致模板被渲染成这样　　Hello, username

这会导致网页的其余部分被粗体化！

为了避免跨站脚本攻击(Cross Site Scripting)（XSS）

1. 起初，对每个不被信任的值运行escape过滤器，这将把潜在的有害的HTML字符转换成无害的字符串

2，利用Django的自动HTML转义功能

自动转义的字符

```
小于号< 转换为 &lt;

大于号> 转换为 &gt;

单引号' 转换为 &#39;

双引号" 转换为 &quot;

与符号& 转换为 &amp;
```

局部关闭自动转义

对单个变量　　使用safe过滤器来关闭变量上的自动转义

对于模板块　　要控制模板上的自动转义，将模板（或者模板中的特定区域）包裹在autoescape标签中

过滤器参数

参数硬编码不转义

可手动转义

```
自动转义：{{content}}

过滤器safe关闭转义：{{content|safe}}
<hr>
标签autoescape关闭转义：
{%autoescape off%}
{{content}}
{%endautoescape%}
<hr>
模板硬编码不转义：{{data|default:"<b>hello world</b>"}}
<hr>
模板硬编码手动转义：{{data|default:"&lt;b&gt;hello world&lt;/b&gt;"}}
<hr>
```

实例

自动转义：<h1>hello world</h1>

过滤器safe关闭转义：

hello world

标签autoescape关闭转义：

hello world

模板硬编码不转义：hello world

模板硬编码手动转义：hello world

可读性转换

Django在django.contrib.humanize中提供了一系列的模板过滤器，有助于为数据展示添加“人文关怀”。

把django.contrib.humanize添加到INSTALLED_APPS设置中来激活这些过滤器

在模板中使用{% load humanize %}标签