# Django简明介绍

1. 学习方法  
    学习Django的重点：

目录结构规范

urls路由方式

settings配置

ORM操作

模板渲染和其它。

AJAX介绍

# 模板

在实际工作中我们很显然不会吧html硬编码到视图中。我们希望可以单独的编写html代码然后通过django的调用显示到前端。这就需要用到模板了。模板是一个文本，用于分离文档的表现形式和内容。模板定义了占位符以及各种用于规范文档该如何显示的各部分基本逻辑（模板标签）。模板通常用于产生HTML，但是Django的模板也能产生任何基于文本格式的文档。本文将介绍模板的基本用法。

1. 特点：Django的模板代码和普通的HTML代码看上去没有太大差别，只是添加了Django特定的模板标记，这些标记允许开发者为Django模板添加页面逻辑。
2. 使用模板系统
3. 渲染模板
   1. 一旦拥有一个Template对象，可以通过给一个context来给它传递数据。
   2. context是一个变量及赋予的值的集合，模板使用它来得到变量的值，或者对于块标签求值，这个context由django.template模块的Context类表示。

# 模板的语法

* 1. 语法：

用{{}}包围的是变量，如{{person\_name}}；

用{%%}包围的是块标签，如{%if ordered\_warranty%}；

过滤器是改变变量显示的方式，过滤器使用管道字符“|”；

变量名必须以字母(A-Z或a-z)开始，可以包含数字，下划线和小数点，变量名大小写敏感；

* 1. Django中的关键类
     1. Template()类：创建模板。
     2. Context()类：用来传递数据，用来为模板提供变量的值或块标签的值，初始化Context对象会传递一个字典给Context()。Context对象是一个stack，你可以push()和pop()。
     3. Template对象中的render()方法：用来填充模板。
  2. Context变量查找
     1. Django模板系统中处理复杂数据结构的关键是使用(.)字符。
     2. 小数点查找优先级：字典查找，如foo["bar"]；属性查找，如foo.bar； 方法调用，如foo.bar()；列表的索引查找，如foo[bar]。
  3. 关于方法调用
     1. ...........
     2. ...........
  4. 模板标签基础
     1. 一般标签的格式：{% tag %} ...标签 内容 ... {% endtag %}
     2. if/else：{% if %}标签,使用{% endif %}来关闭。
     3. for：{% for %}标签允许你按顺序遍历一个序列中的各个元素。{% for %}标签内置了一个forloop模板变量，这个变量含有一些属性可以提供给你一些关于循环的信息。
     4. ifequal/ifnotequal：
     5. 注释：{#\*\*\*\*#}
  5. 过滤器：
     1. 语法格式：**语法 {{obj|filter\_\_name:param}}**
     2. 种类：default、length、filesizeformat、date、slice、truncatechars、safe；
     3. Default：如果一个变量是false或者为空，使用给定的默认值。否则，使用变量的值；
     4. Length：返回值的长度，仅对str和list起作用；
     5. Filesizeformat：格式化为一个“人类可读的”格式；
     6. Date：
     7. slice：切片
     8. Truncatechars&truncatewords：截断字符串，截断部分用省略号表示；
     9. Safe：告诉是否语法标签进行自动转义；
  6. 自定义标签和过滤器：
     1. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# AJAX

AJAX是一种在无需重新加载整个网页的情况下，能够更新部分网页的技术， 是一种用于创建快速动态网页的技术。

原理：AJAX通过在后台与服务器进行少量数据交换，可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。

# 数据库操作

1. Django中创建数据库/表
2. 语法：
   1. 新建对象：
      1. \*.objects.create()
      2. \*.objects.get\_or\_create()
      3. 直接赋值&save()
   2. 获取对象：
      1. \*.objects.all()
      2. \*.objects.all()[:10] // 切片操作
      3. \*objects.get(\*\*\*)
      4. 带有条件的选取：
         1. \*.objects.filter(\*\*\*)
         2. \*.exclude.filter(\*\*\*)
         3. \*.objects.filter.exclude(\*\*\*)
3. SQLite3操作：
   1. .tables
   2. .schema 表名：列出属性名。

# 补充

* 1. 异步任务队列机制：Celery
     1. Web应用。当用户触发一个动作需要较长时间来执行完成时，可以把它作为任务交给celery异步执行，执行完再返回给用户。这点和你在前端使用ajax实现异步加载有异曲同工之妙。
     2. Celery是基于Python开发的一个分布式任务队列框架，支持使用任务队列的方式在分布的机器/进程/线程上执行任务调度。
  2. threading模块：多线程解决异步执行问题
     1. Python主要是通过thread和 threading这两个模块来实现多线程处理，可以方便地支持创建线程、互斥锁、信号量、同步等特性。

参考文献：

1. [Django官方手册](https://docs.pythontab.com/django/django1.5/" \l "model)
2. [快速介绍Django简明教程](https://www.cnblogs.com/feixuelove1009/p/5823135.html)
3. [关于Django模板语法的介绍](https://blog.csdn.net/lvze0321/article/details/53308082)
4. [Django框架模板使用规则和举例](https://blog.csdn.net/zhangxinrun/article/details/8095118)
5. [关于使用class为模板赋值](https://blog.csdn.net/djshichaoren/article/details/78423554)
6. [模板语法使用](https://www.cnblogs.com/haiyan123/p/7725568.html)
7. [Django2.0模板语法](https://www.cnblogs.com/love9527/p/9077863.html)
8. [Ajax教程](http://www.runoob.com/ajax/ajax-xmlhttprequest-send.html)
9. <https://blog.csdn.net/u013694670/article/details/54288543>
10. [Django结合AJAX更新模板](https://www.jianshu.com/p/8ba6e716d223)
11. [创建数据库](https://blog.csdn.net/qy20115549/article/details/78439693)
12. [异步任务队列Celery在Django中的使用](https://www.cnblogs.com/znicy/p/5626040.html)
13. [Django利用多线程增加异步任务](https://blog.csdn.net/orangleliu/article/details/41961295)