# nsd1906_devweb_day04

## Model

```
# 使用python shell
(nsd1906) [root@room8pc16 mysite]# python manage.py shell
# 导入模型
>>> from polls.models import Question, Choice

# 创建问题，方法一：直接创建实例
>>> q1 = Question(question_text="散伙饭去哪吃？", pub_date='2019-11-25')
>>> q1.save()

# 创建问题，方法二：使用objects管理器。django为每个实体类都创建了一个名为objects的管理，通过它的方法可以实现对模型的增删改查等操作。
# get_or_create是，如果没有相关记录则创建，如果库中已有相关记录则取出。
# get_or_create得到的是元组：(问题实例，True/False)
>>> q2 = Question.objects.get_or_create(question_text="你打算到哪 个城市找工作？",
pub_date="2019-12-1")[0]


# 创建选项，方法一：
>>> c1 = Choice(choice_text='杭州', question=q2)
>>> c1.save()

# 创建选项，方法二：
>>> c2 = Choice.objects.get_or_create(choice_text='深圳', question=q2)[0]

# 创建选项，方法三：通过问题创建选项。问题和选项有一对多关系，一个问题有多个选项。每个问题的实例都有一个管理器，通过这个管理器可以创建选项。选项的class名叫Choice，那么管理器就叫choice_set；如果选项的class叫XuanXiang，那么管理器叫xuanxiang_set。
>>> c3 = q2.choice_set.get_or_create(choice_text='成都')[0]


# 查询
# 取出所有的问题，返回所有问题实例构成的列表
>>> Question.objects.all()
# 取出所有的问题，按发布时间升序排列
>>> Question.objects.order_by('pub_date')
# 取出所有的问题，按发布时间降序排列
>>> Question.objects.order_by('-pub_date')
>>> for q in Question.objects.order_by('-pub_date'):
...   print(q.pub_date, q.question_text)

# 通过get取出满足条件的实例，如果结果不是一项则报错
>>> Question.objects.get(id=1)
<Question: 问题:第一份工作，你期待的工资是多少？>
# 通过filter取出满足条件的实例列表，列表中可以是0到多项
```

```
>>> Question.objects.filter(id=10)
<QuerySet []>
>>> Question.objects.filter(id=1)
<QuerySet [<Question: 问题:第一份工作，你期待的工资是多少？>]>

# 查询条件
# django中查询条件使用双下划线代替句点表示属性或方法
# id=1，实际上是id__exact=1的缩写
>>> Question.objects.filter(id__exact=1)
<QuerySet [<Question: 问题:第一份工作，你期待的工资是多少？>]>
# id>1的写法：
>>> Question.objects.filter(id__gt=1)
# id<2的写法：
>>> Question.objects.filter(id__lt=2)
<QuerySet [<Question: 问题:第一份工作，你期待的工资是多少？>]>
# pub_date.month==12的写法：
>>> Question.objects.filter(pub_date__month=12)
<QuerySet [<Question: 问题:你打算到哪个城市找工作？>]>
# 字符串方法举例，判断字符串是否以某些字符开头
>>> s1 = 'hello world'
>>> s1.startswith('he')
True
>>> Question.objects.filter(question_text__startswith='你')
<QuerySet [<Question: 问题:你期待哪家公司给你发Offer？>, <Question: 问题：你打算到哪个城市找工作？
>]>


# 修改问题，只要将问题实例取出，重新赋值即可
>>> q = Question.objects.get(question_text='你期待哪家公司给你发Offer？')
>>> q.question_text = '你心仪的公司是哪家？'
>>> q.save()

# 删除，将问题实例取出，调用删除方法
>>> q = Question.objects.get(question_text='散伙饭去哪吃？')
>>> q.delete()
```

# 修改index首页

```
# 修改index函数，取出所有的问题，发给模板
# polls/views.py
from .models import Question

# Create your views here.
def index(request):
    # 用户发起的请求将会作为第一个参数传给函数
    # 所以函数至少要定义一个参数来接收用户的请求
    questions = Question.objects.order_by('-pub_date')
    # render负责找寻模板文件发送给用户
    return render(request, 'index.html', {'questions': questions})
```

```
# 修改模板
# templates/index.html
# 模板文件中，变量用{{ var }}，模板语法用{% %}。在花括号外面的部分，是html语法
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>投票首页</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="{% static 'css/bootstrap.min.css' %}">
</head>
<body>
<div class="container">
    <div id="linux-carousel" class="carousel slide">
        <ol class="carousel-indicators">
            <li class="active" data-target="#linux-carousel" data-slide-to="0"></li>
            <li data-target="#linux-carousel" data-slide-to="1"></li>
            <li data-target="#linux-carousel" data-slide-to="2"></li>
        </ol>
        <div class="carousel-inner">
            <div class="item active">
                <a href="http://www.sogou.com" target="_blank">
                    <img src="{% static 'imgs/first.jpg' %}">
                </a>
            </div>
            <div class="item">
                <img src="{% static 'imgs/second.jpg' %}">
            </div>
            <div class="item">
                <img src="{% static 'imgs/third.jpg' %}">
            </div>
        </div>
        <a href="#linux-carousel" data-slide="prev" class="carousel-control left">
            <span class="glyphicon glyphicon-chevron-left"></span>
        </a>
        <a href="#linux-carousel" data-slide="next" class="carousel-control right">
            <span class="glyphicon glyphicon-chevron-right"></span>
        </a>
    </div>
    <div>
        <h1 class="text-center text-warning">投票首页</h1>
        <div class="h4">
            <ol>
                {% for question in questions %}
                    <li>
                        <a href="{% url 'detail' question.id %}" target="_blank">
                            {{ question.question_text }}
                        </a>
                        {{ question.pub_date }}
                    </li>
                {% endfor %}
            </ol>
```

```
        </div>
    </div>
    <div class="h4 text-center">
        达内云计算 <a href="#">nsd1906</a>
    </div>
</div>

<script src="{% static 'js/jquery.min.js' %}"></script>
<script src="{% static 'js/bootstrap.min.js' %}"></script>
<script type="text/javascript">
    $('#linux-carousel').carousel({
        interval : 3000
    });
</script>
</body>
</html>
```

# 模板继承

- 为了使得各个页面拥有一致的风格，可以采用模板继承的方法
- 创建一个基础模板，把共性内容写到基础模板中。把个性内容，使用block占位
- 具体的每个web页面，都基于基础模板创建，把个性内容写到相应的block中

```
# 基于index.html实现模板继承
(nsd1906) [root@room8pc16 mysite]# cp templates/index.html templates/basic.html
# templates/basic.html  # 把个性内容用block替换，共性内容保留
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>{% block title %}{% endblock %}</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="{% static 'css/bootstrap.min.css' %}">
</head>
<body>
<div class="container">
    <div id="linux-carousel" class="carousel slide">
        <ol class="carousel-indicators">
            <li class="active" data-target="#linux-carousel" data-slide-to="0"></li>
            <li data-target="#linux-carousel" data-slide-to="1"></li>
            <li data-target="#linux-carousel" data-slide-to="2"></li>
        </ol>
        <div class="carousel-inner">
            <div class="item active">
                <a href="http://www.sogou.com" target="_blank">
                    <img src="{% static 'imgs/first.jpg' %}">
                </a>
            </div>
            <div class="item">
                <img src="{% static 'imgs/second.jpg' %}">
            </div>
```

```
            <div class="item">
                <img src="{% static 'imgs/third.jpg' %}">
            </div>
        </div>
        <a href="#linux-carousel" data-slide="prev" class="carousel-control left">
            <span class="glyphicon glyphicon-chevron-left"></span>
        </a>
        <a href="#linux-carousel" data-slide="next" class="carousel-control right">
            <span class="glyphicon glyphicon-chevron-right"></span>
        </a>
    </div>
    <div>
        {% block content %}{% endblock %}
    </div>
    <div class="h4 text-center">
        达内云计算 <a href="#">nsd1906</a>
    </div>
</div>

<script src="{% static 'js/jquery.min.js' %}"></script>
<script src="{% static 'js/bootstrap.min.js' %}"></script>
<script type="text/javascript">
    $('#linux-carousel').carousel({
        interval : 3000
    });
</script>
</body>
</html>

# 修改index.html去除共性部分，个性部分放到相应的block中
{% extends 'basic.html' %}
{% load static %}
{% block title %}投票首页{% endblock %}
{% block content %}
    <h1 class="text-center text-warning">投票首页</h1>
    <div class="h4">
        <ol>
            {% for question in questions %}
                <li>
                    <a href="{% url 'detail' question.id %}" target="_blank">
                        {{ question.question_text }}
                    </a>
                    {{ question.pub_date }}
                </li>
            {% endfor %}
        </ol>
    </div>
{% endblock %}
```

## 修改detail.html和result.html

```
# templates/detail.html
```

```
{% extends 'basic.html' %}
{% load static %}
{% block title %}投票详情{% endblock %}
{% block content %}
    <h1 class="text-center text-warning">
        {{ question_id }}号问题投票详情
    </h1>
{% endblock %}

# templates/result.html
{% extends 'basic.html' %}
{% load static %}
{% block title %}投票结果{% endblock %}
{% block content %}
    <h1 class="text-center text-warning">
        {{ question_id }}号问题投票结果
    </h1>
{% endblock %}
```

## 完成投票详情页

通过问题取出所有的选项：

```
(nsd1906) [root@room8pc16 mysite]# python manage.py shell
>>> from polls.models import Question
>>> q1 = Question.objects.get(id=1)
>>> q1
<Question: 问题:第一份工作，你期待的工资是多少？>
# 问题实例有一个名为choice_set的选项集，它也是一个管理器，可以通过这个管理器管理该问题的选项
>>> q1.choice_set.all()
>>> q1.choice_set.order_by('id')
```

```
# 取出某一问题的实例，交给模板处理
# polls/views.py
def detail(request, question_id):
    # 在数据库中取出具体的问题
    question = Question.objects.get(id=question_id)
    # 字典的内容将会成为模板文件的变量，字典的key是变量名，val是变量值
    return render(request, 'detail.html', {'question': question})

# 修改模板
# templates/detail.html
{% extends 'basic.html' %}
{% load static %}
{% block title %}投票详情{% endblock %}
{% block content %}
    <h1 class="text-center text-warning">
        {{ question.id }}号问题投票详情
```

```
    </h1>
    <h2>{{ question.question_text }}</h2>
    <div class="h4">
        <form action="{% url 'vote' question.id %}" method="post">
            {% comment %}csrf_token是安全选项，必须设置{% endcomment %}
            {% csrf_token %}
            {% for choice in question.choice_set.all %}
                <div class="radio">
                    <label>
                        <input type="radio" name="choice_id" value="{{ choice.id }}">
                        {{ choice.choice_text }}
                    </label>
                </div>
            {% endfor %}
            <div class="form-group">
                <input class="btn btn-primary" type="submit" value="提 交">
            </div>
        </form>
    </div>
{% endblock %}
```

# 实现投票功能

- 投票功能应该是执行一个函数，该函数能够把选项的votes字段加1
- 在django中访问某一url将会调用相关函数
- 所以，要想实现投票功能，应该在detail页面中，提交表单时，提交到一个url，该url调用相关函数。函数用于在数据库中找到选项，把选项的votes字段值加1.

```
# polls/urls.py
urlpatterns = [
    ... ...
    url(r'^(\d+)/vote/$', views.vote, name='vote'),
]

# polls/views.py
from django.shortcuts import render, redirect

def vote(request, question_id):
    # 取出问题
    question = Question.objects.get(id=question_id)
    # request是用户的请求，它有很多属性，客户端提交的数据保存到了
    # request.POST字典中，choice_id是表单的属性
    choice_id = request.POST.get('choice_id')
    # 通过问题取出选项实例
    choice = question.choice_set.get(id=choice_id)
    # 将选项的votes属性值加1
    choice.votes += 1
    choice.save()
    # 投票结束后，跳转到投票结果页
    return redirect('result', question.id)
```

## 完成投票结果页

```
# 修改函数，取出问题
# polls/views.py
def result(request, question_id):
    question = Question.objects.get(id=question_id)
    return render(request, 'result.html', {'question': question})

# templates/result.html
{% extends 'basic.html' %}
{% load static %}
{% block title %}投票结果{% endblock %}
{% block content %}
    <h1 class="text-center text-warning">
        {{ question.id }}号问题投票结果
    </h1>
    <h2>{{ question.question_text }}</h2>
    <table class="table table-striped table-hover h4">
        <thead class="bg-primary">
            <tr>
                <th>选项</th>
                <th>票数</th>
            </tr>
        </thead>
        {% for choice in question.choice_set.all %}
            <tr>
                <td>{{ choice.choice_text }}</td>
                <td>{{ choice.votes }}</td>
            </tr>
        {% endfor %}
    </table>
    <h5>
        <a href="{% url 'index' %}">返回首页</a>
    </h5>
{% endblock %}
```