nsd1904_devweb_day04

操作模型

```
# 加载python shell
(nsd1904) [root@room8pc16 mysite]# python manage.py shell
# 导入模型
>>> from polls.models import Question, Choice
# 创建问题实例，方法一：
>>> q1 = Question(question_text="你计划哪个城市工作？", pub_date="2019-09-27 10:00:00")
>>> q1.save()
# 创建问题实例，方法二：
# 每个实体类都有一个名为objects的管理器，通过这个管理器实现CRUD等操作
>>> q2 = Question.objects.create(question_text="你期待哪家公司给你发Offer？", pub_date="2019-10-01 12:00:00")

# 创建选项实例，方法一：
>>> c1 = Choice(choice_text="阿里巴巴", question=q2)
>>> c1.save()
# 创建选项实例，方法二：
>>> c2 = Choice.objects.create(choice_text="Tencent", question=q2)
# 创建选项实例，方法三：
# 问题和选项存在一对多的关系，一个问题可以有很多选项。每个问题的实例都有一个名为choice_set的管理器，通过它，可以创建选项。如果选项模型名为XuanXiang，那么问题的管理器就叫xuanxiang_set
>>> c3 = q2.choice_set.create(choice_text="jd")

# 查询：get必须只返回一个结果
>>> q1 = Question.objects.get(id=1)
>>> q1
<Question: 问题: 你期待的工资是多少？>

# 查询：all返回所有实例构成的实例列表
>>> Question.objects.all()

# 查询：filter返回0到多个实例构成的列表
>>> Question.objects.filter(id=10)
<QuerySet []>

# get和filter都采用相同的条件，只是返回值不一样。条件的书写方式使用双下划线来表示属性或方法，而不是句点
>>> s1 = 'Hello World'
>>> s1.startswith('He')
True
>>> Question.objects.filter(question_text__startswith="你")

>>> from datetime import datetime
>>> t = datetime.now()
>>> t.month
9
```

```
>>> Question.objects.filter(pub_date__month=9)
<QuerySet [<Question: 问题: 你期待的工资是多少？>, <Question: 问题: 十一ion: 问题: 你计划哪个城市工作？>]>

# 查询条件的比较也都是通过双下划线来实现的，=是简写：
>>> Question.objects.get(id=1)  等价于
>>> Question.objects.filter(id__exact=1)

# 数字其他比较
>>> Question.objects.filter(id__gt=1)   # >1
>>> Question.objects.filter(id__gte=1)  # >=1
>>> Question.objects.filter(id__lt=1)   # <1
>>> Question.objects.filter(id__lte=1)  # <=1

# 字符串相关查询
>>> Question.objects.filter(question_text__contains="工资")  # 包含
>>> Question.objects.filter(question_text__endswith='？')

# 排序
>>> Question.objects.order_by('pub_date')   # 默认升序排列
>>> Question.objects.order_by('-pub_date')   # 降序排列
>>> Question.objects.filter(pub_date__month=9).order_by('-pub_date')

# 修改，实际上就是把实例变量重新赋值
>>> c1 = Choice.objects.get(choice_text='jd')
>>> c1.choice_text = "京东"
>>> c1.save()

# 删除，通过实例的delete方法实现
>>> c2 = Choice.objects.get(choice_text="Tencent")
>>> c2.delete()
```

修改投票首页

```
# polls/views.py
from .models import Question
def index(request):
    questions = Question.objects.order_by('-pub_date')
    return render(request, 'index.html', {'questions': questions})

# templates/index.html
# {{ forloop.counter }}是循环内部变量，记录的是第几次循环
# {% url %}是模板语法，表示url的名称，detail接收一个参数，所以要把question.id传过去
<body>
<h1>投票首页</h1>
{% for question in questions %}
<div>
    {{ forloop.counter }}.
    <a href="{% url 'detail' question.id %}" target="_blank">
        {{ question.question_text }}
    </a>
    {{ question.pub_date }}
```

```
</div>
{% endfor %}
</body>
```

引入bootstrap

```
# 把day02中的static目录拷贝到polls目录
(nsd1904) [root@room8pc16 mysite]# cp -r ../../day02/static polls/
# 修改index.html
<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>

    <h1 class="text-center text-warning">投票首页</h1>
    <div class="h4" style="margin: 30px 0">
        {% for question in questions %}
        <div>
            {{ forloop.counter }}.
            <a href="{% url 'detail' question.id %}" target="_blank">
                {{ question.question_text }}
            </a>
            {{ question.pub_date }}
        </div>
        {% endfor %}
    </div>

    <div class="h4 text-center">
```

```
        达内云计算学院 <a href="#">nsd1904</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>
```

使用模板扩展

- 为了方便地实现统一的界面风格，可以使用模板扩展
- 把各个页面相同的内容放到基础模板中
- 制作个性化页面的时候，扩展于基础模板

```
# 拷贝index.html，改名为base.html
# 在base.html中，把个性化内容删除，使用block占位
{% 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>

    {% block content %}{% endblock %}

    <div class="h4 text-center">
        达内云计算学院 <a href="#">nsd1904</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，把共性内容删除，个性内容放到block中
{% extends 'base.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" style="margin: 30px 0">
        {% for question in questions %}
        <div>
            {{ forloop.counter }}.
            <a href="{% url 'detail' question.id %}" target="_blank">
                {{ question.question_text }}
            </a>
            {{ question.pub_date }}
        </div>
        {% endfor %}
    </div>
{% endblock %}
```

完成投票详情页

```
# polls/views.py
def detail(request, question_id):
    question = Question.objects.get(id=question_id)
    return render(request, 'detail.html', {'question': question})

# 修改模板文件templates/detail.html
# 说明，{% csrf_token %}是django防止跨站攻击所必须的，没有此项，提交时将会出现403错误
```

```
# form表单的method设置为post，默认是get
{% extends 'base.html' %}
{% load static %}
{% block title %}投票详情页{% 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" style="margin: 30px 0">
        <form action="" method="post">
            {% 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 %}
```

完成投票功能

- 投标功能，是把某一投票项的votes字段值加1
- 实现投票功能，需要调用函数
- 调用函数是通过访问url来实现的

```
# polls/urls.py
    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)
    # 通过用户提交的表单，取出用户选择项目的id
    print(dir(request))   # 在终端上打印request所有属性
    choice_id = request.POST.get('choice_id')
    # 通过问题的choice_set管理器取出选项实例
    choice = question.choice_set.get(id=choice_id)
    # 修改选项的票数
    choice.votes += 1
    choice.save()
    # 重定向到结果页
    return redirect('result', question.id)

# 修改detail.html中form表单的action
        <form action="{% url 'vote' question.id %}" method="post">
```

```
# 投票测试
```

完成投票结果页

```
# 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 'base.html' %}
{% load static %}
{% block title %}投票结果页{% endblock %}
{% block content %}
    <h1  class="text-center text-warning">{{ question.id }}号问题投票结果</h1>
    <div class="h4" style="margin: 30px 0">
        <table class="table table-hover table-striped">
            <thead>
                <tr class="bg-primary h3">
                    <td colspan="2">
                        {{ question.question_text }}
                    </td>
                </tr>
            </thead>
            {% for choice in question.choice_set.all %}
                <tr>
                    <td>{{ choice.choice_text }}</td>
                    <td>{{ choice.votes }}</td>
                </tr>
            {% endfor %}
        </table>
    </div>
{% endblock %}
```