nsd1905_devweb_day04

django api

```
# 使用python shell
(nsd1905) [root@room8pc16 mysite]# python manage.py shell
>>> from polls.models import Question, Choice
# 在表中添加记录，方法一：Qeustion实例
>>> q1 = Question(question_text='你期待哪个公司给你发Offer？', pub_date='2019-10-28')
>>> q1.save()
# 在表中添加记录，方法二：
# django为每一个class都创建了一个名为objects的管理器，通过这个管理可以实现对模型的操作，如增删改查等
>>> q2 = Question.objects.create(question_text='毕业聚餐去哪里？', pub_date='2019-10-28
10:00:00')

# 添加选项，方法一：
>>> c1 = Choice(choice_text='阿里巴巴', question=q1)
>>> c1.save()
# 添加选项，方法二：
>>> c2 = Choice.objects.create(choice_text='腾讯', question=q1)
# 添加选项，方法三：每个问题的实例都有一个名为choice_set的管理器，通过它，可以反向创建选项实例。管理器的
名字构成：选项class名_set，全部小写字母。如果选项class是XuanXiang，那么问题实例的管理器名，就叫
xuanxiang_set
>>> c3 = q1.choice_set.create(choice_text='达内')

# 查询所有的问题,返回所有问题实例的列表
>>> qset1 = Question.objects.all()
>>> qset1
>>> qset1[0]
<Question: 问题: 第一份工作，你期待的工资是多少？>
>>> q1 = qset1[0]
>>> q1.question_text
'第一份工作，你期待的工资是多少？'
>>> q1.pub_date
datetime.datetime(2019, 10, 26, 16, 49)
>>> q1.id  # 没有在模型中声明主键，django自动添加名为id的主键
1
>>> for q in qset1:
...   print(q.question_text, q.pub_date)
...

# 排序，默认是升序排列
>>> qset2 = Question.objects.order_by('pub_date')
>>> for q in qset2:
...   print(q.question_text, q.pub_date)
...
你计划去哪个城市工作？ 2019-10-01 12:00:00
第一份工作，你期待的工资是多少？ 2019-10-26 16:49:00
```

```
你期待哪个公司给你发Offer？ 2019-10-28 00:00:00
毕业聚餐去哪里？ 2019-10-28 10:00:00
# 降序排列
>>> qset3 = Question.objects.order_by('-pub_date')
>>> for q in qset3:
...   print(q.question_text, q.pub_date)
...
毕业聚餐去哪里？ 2019-10-28 10:00:00
你期待哪个公司给你发Offer？ 2019-10-28 00:00:00
第一份工作，你期待的工资是多少？ 2019-10-26 16:49:00
你计划去哪个城市工作？ 2019-10-01 12:00:00

# get方法取出一个实例，如果多于一个实例或是0个，都报错
>>> Question.objects.get(id=10)  # 没有取到，报错
>>> Question.objects.get(id__lt=10)  # id小于10的，有4项，报错
>>> Question.objects.get(id=1)
<Question: 问题: 第一份工作，你期待的工资是多少？>

# filter方法取出实例列表，列表长度不限
>>> Question.objects.filter(id__lt=10)
<QuerySet [<Question: 问题: 第一份工作，你期待的工资是多少？>, <Questio>, <Question: 问题: 你期待
哪个公司给你发Offer？>, <Question: 问题: 毕业聚餐去哪里？>]>
>>> Question.objects.filter(id__gt=10)
<QuerySet []>

# 过滤条件。在django中使用的方式是:属性__条件=值
>>> Question.objects.filter(id=1)  # 是以下方式的简写
>>> Question.objects.filter(id__exact=1)
>>> Question.objects.filter(id__gt=1)  # id>1
>>> Question.objects.filter(id__lt=3)  # id<3
>>> Question.objects.filter(id__lte=3) # id<=3
>>> Question.objects.filter(id__gte=1) # id>=1
>>> Question.objects.filter(pub_date__year=2019)# pub_date.year=2019
>>> Question.objects.filter(pub_date__day=28) # pub_date.day=28
# question_text.startswith('你')
>>> Question.objects.filter(question_text__startswith='你')

# 修改
>>> q1 = Question.objects.get(question_text='你期待哪个公司给你发Offer ？')
>>> q1.question_text = '你心仪的公司是哪一家？'
>>> q1.save()
# 删除
>>> q2 = Question.objects.get(question_text='毕业聚餐去哪里？')
>>> q2.delete()
```

完善首页，将所有的问题取出，发往前台页面

```
# polls/views.py
from django.shortcuts import render
from .models import Question

def index(request):
```

```
    questions = Question.objects.order_by('-pub_date')
    return render(request, 'index.html', {'questions': questions})

# templates/index.html  # 显示问题
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>投票首页</title>
</head>
<body>
<div class="container">
    <div class="content">
        <h1>投票首页</h1>
        {{ questions }}
    </div>
</div>
</body>
</html>
# 启动开发服务器，测试。开发服务器运行在0.0.0.0:80。注意，如果不是管理员，不能使用小于1024的端口。
(nsd1905) [root@room8pc16 mysite]# python manage.py runserver 0:80

# 修改模板文件，将问题逐一展示
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>投票首页</title>
</head>
<body>
<div class="container">
    <div class="content">
        <h1>投票首页</h1>
        <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>
        达内云计算学院 NSD1905
    </div>
</div>
</body>
</html>

# 引入bootstrap
# django默认到应用的static目录下寻找静态文件
```

```
(nsd1905) [root@room8pc16 mysite]# cp -r ../../day02/static polls/
{% 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 class="content h4">
        <h1 class="text-center text-warning">投票首页</h1>
        <ol style="margin: 20px 0">
            {% 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 class="h4 text-center">
        达内云计算学院 NSD1905
```

```
    </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>
```

实现模板继承

为了使得各个页面具有相同的页面风格，可以创建一个基础页面，将共性内容写到基础页面。其他各个页面继承基础页面，把个性内容写到自己的页面中。

```
# 将index.html，复制一份取名为basic.html
# 在basic.html中，将个性内容用block替代
{% 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 NSD1905
    </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，个性内容写到对应的block中
{% extends 'basic.html' %}
{% load static %}
{% block title %}投票首页{% endblock %}
{% block content %}
    <div class="content h4">
        <h1 class="text-center text-warning">投票首页</h1>
        <ol style="margin: 20px 0">
            {% 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 %}
```

制作投票详情页

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

# templates/detail.html
{% extends 'basic.html' %}
{% load static %}
{% block title %}投票详情页{% endblock %}
```

```
{% block content %}
    <div class="content h4">
        <h1 class="text-center text-warning">
            {{ question.id }}号问题的投票详情页
        </h1>
        <h3>
            {{ question.question_text }}
        </h3>
        <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 %}
```

实现投票功能

投票功能是将数据库中某一选项的votes字段值加1。为了实现该功能，需要执行函数，函数首先取出选项，再把选项votes字段值进行修改。为了执行函数，可以访问一个URL。

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

# polls/views.py
def vote(request, question_id):
    question = Question.objects.get(id=question_id)
    # 当用户提交表单时，request.POST是一个字典，里面记录了与POST相关的数据
    # choice_id是detail.html页面中单选按钮的name，值是选项的id
    choice_id = request.POST.get('choice_id')
    choice = question.choice_set.get(id=choice_id)
    choice.votes += 1
    choice.save()

    # 这里返回使用的不是render，因为render直接返回页面，URL不变，也就是
    # http://x.x.x.x/polls/2/vote显示的是2号问题的投票结果，这是不合理的
    # 应该由http://x.x.x.x/polls/2/result/显示投票结果。所以使用redirect
    return redirect('result', question_id)

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

制作投票结果页

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

# templates/result.html
{% extends 'basic.html' %}
{% load static %}
{% block title %}投票结果页{% endblock %}
{% block content %}
    <div class="content h4">
        <h1 class="text-center text-warning">
            {{ question.id }}号问题的投票结果
        </h1>
        <table class="table table-striped table-hover">
            <thead class="bg-primary">
                <tr>
                    <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 %}
```