DRF框架

序列化类的功能：进行序列化和反序列化。

序列化: 将对象转化为字典数据。

反序列化:

1）数据校验。

2）数据保存(新增&更新)

1. 序列化器-序列化

1）序列化单个对象

```
book = BookInfo.objects.get(id=1)
serializer = BookInfoSerializer(book)
serializer.data
```

2）序列化多个对象

```
books = BookInfo.objects.all()
serializer = BookInfoSerializer(books, many=True)
serializer.data
```

3）关联对象的序列化

```
1. 将关联对象序列化为关联对象主键 PrimaryKeyRelatedFieldå
2. 将关联对象使用指定的序列化器进行序列化
3. 将关联对象序列化为关联对象模型类__str__方法的返回值 StringRelatedField

注意：如果关联对象有多个，定义字段时，需要添加many=True
```

2. 序列化器-反序列化

1）反序列化之数据校验

```
data = {'btitle': 'python'}

serializer = BookInfoSerializer(data=data)
serializer.is_valid() # 调用此方法会对传递的data数据内容进行校验，校验成功返回True，否则返回False
serializer.errors # 获取校验失败的错误信息
serializer.validated_data # 获取校验之后的数据

# 补充验证行为
1. 对对应的字段指定validators
2. 序列化器类中定义对应的方法validate_<field_name>对指定的字段进行校验
3. 如果校验需要结合多个字段内容，定义validate方法
```

2）反序列化之数据保存(新增或更新)

```
# 数据校验之后，调用此方法可以进行数据保存，可能会调用序列化器类中的create或update
serializer.save()

# 调用create，创建序列化器对象时只传递了data
data = {'btitle': 'python'}
serializer = BookInfoSerializer(data=data)
serializer.is_valid()
serializer.save()

# 调用update，创建序列化器对象时传递了data和对象
book = BookInfo.objects.get(id=1)
data = {'btitle': 'python'}
serializer = BookInfoSerializer(book, data=data)
serializer.is_valid()
serializer.save()
```

3. 使用Serializer改写Django自定义RestAPI接口

将序列化和反序列化部分代码使用序列化器完成。

4. APIView视图类&Request对象&Response对象

1）APIView视图类

APIView是DRF框架中所有视图类的父类。

继承自APIView之后，处理函数中的request参数不再是Django原始的HttpRequest对象，而是由DRF框架封装的Request类的对象。

进行异常处理。

认证&权限&限流。

2）Request类

request.data: 包含传递的请求体数据(比如之前的request.body和request.POST)，并且已经转换成了字典或类字典类型。

request.query_params: 包含查询字符串参数(相当于之前的request.GET)

3）Response类

通过Response返回响应数据时，会根据前端请求头中的`Accept`转化成对应的响应数据类型，仅支持json和html，默认返回json。

4）补充

类视图对象`self.kwargs`保存这从url地址中提取的所有命名参数。

5. 使用APIView改写Django自定义RestAPI接口

将获取参数以及响应部分代码进行改写。

6. GenericAPIView视图类

APIView类的子类，封装了操作序列化器和操作数据库的方法。

过滤&分页。

属性：

serializer_class: 指定视图使用的序列化器类

queryset: 指定视图使用的查询集

方法:

get_serializer_class: 返回当前视图使用的序列化器类

get_serializer: 创建一个序列化器类的对象

get_queryset: 获取当前视图使用的查询集

get_object: 获取单个对象，默认根据主键进行查询