<https://ilyachch.gitbook.io/django-rest-framework-russian-documentation/overview/navigaciya-po-api/serializers>

Сериализаторы позволяют преобразовывать сложные данные, такие как querysets и экземпляры моделей, в нативные типы данных Python, которые затем могут быть легко срендерены в JSON, XML или другие типы контента. Сериализаторы также обеспечивают десериализацию, позволяя преобразовать спарсенные данные обратно в сложные типы после проверки входящих данных.

def update(self, instance, validated\_data):  
 raise NotImplementedError('`update()` must be implemented.')  
  
def create(self, validated\_data):  
 raise NotImplementedError('`create()` must be implemented.')

default save():

def save(self, \*\*kwargs):

...

validated\_data = {\*\*self.validated\_data, \*\*kwargs}  
  
 if self.instance is not None:  
 self.instance = self.update(self.instance, validated\_data)  
 assert self.instance is not None, (  
 '`update()` did not return an object instance.'  
 )  
 else:  
 self.instance = self.create(validated\_data)  
 assert self.instance is not None, (  
 '`create()` did not return an object instance.'  
 )  
  
 return self.instance

Класс ModelSerializer совпадает с обычным классом Serializer, за исключением того, что:

* Он автоматически генерирует набор полей на основе модели.
* Он автоматически генерирует валидаторы для сериализатора, такие как unique\_together.
* Он включает простые стандартные реализации .create() и .update().

Класс HyperlinkedModelSerializer похож на класс ModelSerializer, за исключением того, что для представления отношений он использует гиперссылки, а не первичные ключи.

Если поле используется для представления отношения ко многим, вы должны добавить флаг many=True в поле сериализатора.