Serializer Class

```
from rest_framework import serializers
class StudentSerializer(serializers.Serializer):
    name = serializers.CharField(max_length=100)
    roll = serializers.IntegerField()
    city = serializers.CharField(max_length=100)
```

Serializer Field

Serializer fields handle converting between primitive values and internal datatypes. They also deal with validating input values, as well as retrieving and setting the values from their parent objects.

Syntax:-

```
from rest_framework import serializers
serializers.Field_Name( )
```

Example:-

```
from rest_framework import serializers
serializers.CharField( )
```

Serializer Field

CharField - A text representation. Optionally validates the text to be shorter than max_length and longer than min_length.

Syntax:- CharField(max_length=None, min_length=None, allow_blank=False, trim_whitespace=True)

- max_length - Validates that the input contains no more than this number of characters.
- min_length - Validates that the input contains no fewer than this number of characters.
- allow_blank - If set to True then the empty string should be considered a valid value. If set to False then the empty string is considered invalid and will raise a validation error. Defaults to False.
- trim_whitespace - If set to True then leading and trailing whitespace is trimmed. Defaults to True.
- The allow_null option is also available for string fields, although its usage is discouraged in favor of allow_blank.

Serializer Field

IntegerField - An integer representation.

Syntax:- IntegerField(max_value=None, min_value=None)

- max_value Validate that the number provided is no greater than this value.
- min_value Validate that the number provided is no less than this value.

FloatField - A floating point representation.

Syntax:- FloatField(max_value=None, min_value=None)

- max_value Validate that the number provided is no greater than this value.
- min_value Validate that the number provided is no less than this value.

Serializer Field

DecimalField - A decimal representation, represented in Python by a Decimal instance.

Syntax:- DecimalField(max_digits, decimal_places, coerce_to_string=None, max_value=None, min_value=None)

- max_digits The maximum number of digits allowed in the number. It must be either None or an integer greater than or equal to decimal_places.
- decimal_places The number of decimal places to store with the number.
- coerce_to_string Set to True if string values should be returned for the representation, or False if Decimal objects should be returned. Defaults to the same value as the COERCE_DECIMAL_TO_STRING settings key, which will be True unless overridden. If Decimal objects are returned by the serializer, then the final output format will be determined by the renderer. Note that setting localize will force the value to True.
- max_value Validate that the number provided is no greater than this value.
- min_value Validate that the number provided is no less than this value.

Serializer Field

localize Set to True to enable localization of input and output based on the current locale. This will also force coerce_to_string to True. Defaults to False. Note that data formatting is enabled if you have set USE_L10N=True in your settings file.

rounding Sets the rounding mode used when quantising to the configured precision. Valid values are decimal module rounding modes. Defaults to None.

SlugField - A RegexField that validates the input against the pattern [a-zA-Z0-9_-]+

Syntax:- SlugField(max_length=50, min_length=None, allow_blank=False)

Serializer Field

EmailField - A text representation, validates the text to be a valid e-mail address.

Syntax:- EmailField(max_length=None, min_length=None, allow_blank=False)

BooleanField - A boolean representation.

Syntax:- BooleanField()

NullBooleanField - A boolean representation that also accepts None as a valid value.

Syntax:- NullBooleanField()

Serializer Field

URLField - A RegexField that validates the input against a URL matching pattern. Expects fully qualified URLs of the form http://<host>/<path>.

Syntax:- URLField(max_length=200, min_length=None, allow_blank=False)

FileField - A file representation. Performs Django's standard FileField validation.

Syntax:- FileField(max_length=None, allow_empty_file=False, use_url=UPLOADED_FILES_USE_URL)

max_length - Designates the maximum length for the file name.

allow_empty_file - Designates if empty files are allowed.

use_url - If set to True then URL string values will be used for the output representation. If set to False then filename string values will be used for the output representation. Defaults to the value of the UPLOADED_FILES_USE_URL settings key, which is True unless set otherwise.

Serializer Field

ImageField - An image representation. Validates the uploaded file content as matching a known image format.

Syntax:- ImageField(max_length=None, allow_empty_file=False, use_url=UPLOADED_FILES_USE_URL)

max_length - Designates the maximum length for the file name.

allow_empty_file - Designates if empty files are allowed.

use_url - If set to True then URL string values will be used for the output representation. If set to False then filename string values will be used for the output representation. Defaults to the value of the UPLOADED_FILES_USE_URL settings key, which is True unless set otherwise.

Note:-

The FileField and ImageField classes are only suitable for use with MultiPartParser or FileUploadParser. Most parsers, such as e.g. JSON don't support file uploads.

Requires either the Pillow package.

Serializer Field

DateField - A date representation.

Syntax:- DateField(format=api_settings.DATE_FORMAT, input_formats=None)

format - A string representing the output format. If not specified, this defaults to the same value as the DATE_FORMAT settings key, which will be 'iso-8601' unless set. Setting to a format string indicates that to_representation return values should be coerced to string output. Format strings are described below. Setting this value to None indicates that Python date objects should be returned by to_representation. In this case the date encoding will be determined by the renderer.

input_formats - A list of strings representing the input formats which may be used to parse the date. If not specified, the DATE_INPUT_FORMATS setting will be used, which defaults to ['iso-8601'].

Serializer Field

TimeField - A time representation.

Syntax:- TimeField(format=api_settings.TIME_FORMAT, input_formats=None)

format - A string representing the output format. If not specified, this defaults to the same value as the TIME_FORMAT settings key, which will be 'iso-8601' unless set. Setting to a format string indicates that to_representation return values should be coerced to string output. Format strings are described below. Setting this value to None indicates that Python time objects should be returned by to_representation. In this case the time encoding will be determined by the renderer.

input_formats - A list of strings representing the input formats which may be used to parse the date. If not specified, the TIME_INPUT_FORMATS setting will be used, which defaults to ['iso-8601'].

Serializer Field

DateTimeField - A date and time representation.

Syntax:- DateTimeField(format=api_settings.DATETIME_FORMAT, input_formats=None, default_timezone=None)

format - A string representing the output format. If not specified, this defaults to the same value as the DATETIME_FORMAT settings key, which will be 'iso-8601' unless set. Setting to a format string indicates that to_representation return values should be coerced to string output. Format strings are described below. Setting this value to None indicates that Python datetime objects should be returned by to_representation. In this case the datetime encoding will be determined by the renderer.

input_formats - A list of strings representing the input formats which may be used to parse the date. If not specified, the DATETIME_INPUT_FORMATS setting will be used, which defaults to ['iso-8601'].

default_timezone - A pytz.timezone representing the timezone. If not specified and the USE_TZ setting is enabled, this defaults to the current timezone. If USE_TZ is disabled, then datetime objects will be naive.

Serializer Field

DurationField - A Duration representation. The validated_data for these fields will contain a datetime.timedelta instance. The representation is a string following this format '[DD] [HH:[MM:]]ss[.uuuuuu]'.

Syntax:- DurationField(max_value=None, min_value=None)

max_value Validate that the duration provided is no greater than this value.

min_value Validate that the duration provided is no less than this value.

RegexField - A text representation, that validates the given value matches against a certain regular expression.

Syntax:- RegexField(regex, max_length=None, min_length=None, allow_blank=False)

regex argument may either be a string, or a compiled python regular expression object.

Serializer Field

UUIDField - A field that ensures the input is a valid UUID string. The to_internal_value method will return a uuid.UUID instance. On output the field will return a string in the canonical hyphenated format.

Syntax:- UUIDField(format='hex_verbose')

format: Determines the representation format of the uuid value

'hex_verbose' - The canonical hex representation, including hyphens: "5ce0e9a5-5ffa-654b-cee0-1238041fb31a"

'hex' - The compact hex representation of the UUID, not including hyphens: "5ce0e9a55ffa654bcee01238041fb31a"

'int' - A 128 bit integer representation of the UUID: "123456789012312313134124512351145145114"

'urn' - RFC 4122 URN representation of the UUID: "urn:uuid:5ce0e9a5-5ffa-654b-cee0-1238041fb31a" Changing the format parameters only affects representation values. All formats are accepted by to_internal_value

Serializer Field

FilePathField -A field whose choices are limited to the filenames in a certain directory on the filesystem.

Syntax:- FilePathField(path, match=None, recursive=False, allow_files=True, allow_folders=False, required=None, **kwargs)

path - The absolute filesystem path to a directory from which this FilePathField should get its choice.

match - A regular expression, as a string, that FilePathField will use to filter filenames.

recursive - Specifies whether all subdirectories of path should be included. Default is False.

allow_files - Specifies whether files in the specified location should be included. Default is True. Either this or allow_folders must be True.

allow_folders - Specifies whether folders in the specified location should be included. Default is False. Either this or allow_files must be True.

Serializer Field

IPAddressField - A field that ensures the input is a valid IPv4 or IPv6 string.

Syntax:- IPAddressField(protocol='both', unpack_ipv4=False, **options)

protocol Limits valid inputs to the specified protocol. Accepted values are 'both' (default), 'IPv4' or 'IPv6'. Matching is case insensitive.

unpack_ipv4 Unpacks IPv4 mapped addresses like ::ffff:192.0.2.1. If this option is enabled that address would be unpacked to 192.0.2.1. Default is disabled. Can only be used when protocol is set to 'both'.

Serializer Field

ChoiceField - A field that can accept a value out of a limited set of choices.

Used by ModelSerializer to automatically generate fields if the corresponding model field includes a choices=… argument.

Syntax:- ChoiceField(choices)

choices - A list of valid values, or a list of (key, display_name) tuples.

allow_blank - If set to True then the empty string should be considered a valid value. If set to False then the empty string is considered invalid and will raise a validation error. allow_blank should be preferred for textual choices. Defaults to False.

html_cutoff - If set this will be the maximum number of choices that will be displayed by a HTML select drop down. Can be used to ensure that automatically generated ChoiceFields with very large possible selections do not prevent a template from rendering. Defaults to None.

html_cutoff_text - If set this will display a textual indicator if the maximum number of items have been cutoff in an HTML select drop down. Defaults to "More than {count} items…"

allow_null - allow_null should be preferred for numeric or other non-textual choices.

Serializer Field

MultipleChoiceField - A field that can accept a set of zero, one or many values, chosen from a limited set of choices. Takes a single mandatory argument. to_internal_value returns a set containing the selected values.

Syntax:- MultipleChoiceField(choices)

choices - A list of valid values, or a list of (key, display_name) tuples.

allow_blank - If set to True then the empty string should be considered a valid value. If set to False then the empty string is considered invalid and will raise a validation error. allow_blank should be preferred for textual choices. Defaults to False. Defaults to False.

html_cutoff - If set this will be the maximum number of choices that will be displayed by a HTML select drop down. Can be used to ensure that automatically generated ChoiceFields with very large possible selections do not prevent a template from rendering. Defaults to None.

html_cutoff_text - If set this will display a textual indicator if the maximum number of items have been cutoff in an HTML select drop down. Defaults to "More than {count} items…"

allow_null - allow_null should be preferred for numeric or other non-textual choices.

Serializer Field

ListField - A field class that validates a list of objects.

Sntax:- ListField(child=<A_FIELD_INSTANCE>, allow_empty=True, min_length=None, max_length=None)

child - A field instance that should be used for validating the objects in the list. If this argument is not provided then objects in the list will not be validated.

allow_empty - Designates if empty lists are allowed.

min_length - Validates that the list contains no fewer than this number of elements.

max_length - Validates that the list contains no more than this number of elements.

Serializer Field

DictField - A field class that validates a dictionary of objects. The keys in DictField are always assumed to be string values.

Syntax:- DictField(child=<A_FIELD_INSTANCE>, allow_empty=True)

child - A field instance that should be used for validating the values in the dictionary. If this argument is not provided then values in the mapping will not be validated.

allow_empty - Designates if empty dictionaries are allowed.

Serializer Field

HStoreField - A preconfigured DictField that is compatible with Django's postgres HStoreField.

Syntax:- HStoreField(child=<A_FIELD_INSTANCE>, allow_empty=True)

child - A field instance that is used for validating the values in the dictionary. The default child field accepts both empty strings and null values.

allow_empty - Designates if empty dictionaries are allowed.

Note that the child field must be an instance of CharField, as the hstore extension stores values as strings.

Serializer Field

JSONField - A field class that validates that the incoming data structure consists of valid JSON primitives. In its alternate binary mode, it will represent and validate JSON-encoded binary strings.

Syntax: JSONField(binary, encoder)

binary - If set to True then the field will output and validate a JSON encoded string, rather than a primitive data structure. Defaults to False.

encoder - Use this JSON encoder to serialize input object. Defaults to None.

ReadOnlyField - A field class that simply returns the value of the field without modification.

This field is used by default with ModelSerializer when including field names that relate to an attribute rather than a model field.

Syntax: ReadOnlyField()

Serializer Field

HiddenField - A field class that does not take a value based on user input, but instead takes its value from a default value or callable. The HiddenField class is usually only needed if you have some validation that needs to run based on some pre-provided field values, but you do not want to expose all of those fields to the end user.

Syntax:- HiddenField()

Serializer Field

ModelField - A generic field that can be tied to any arbitrary model field. The ModelField class delegates the task of serialization/deserialization to its associated model field. This field can be used to create serializer fields for custom model fields, without having to create a new custom serializer field.

This field is used by ModelSerializer to correspond to custom model field classes.

Syntax:- ModelField(model_field=<Django ModelField instance>)

The ModelField class is generally intended for internal use, but can be used by your API if needed. In order to properly instantiate a ModelField, it must be passed a field that is attached to an instantiated model. For example:
ModelField(model_field=MyModel()._meta.get_field('custom_field'))

Serializer Field

SerializerMethodField - This is a read-only field. It gets its value by calling a method on the serializer class it is attached to. It can be used to add any sort of data to the serialized representation of your object.

Syntax: SerializerMethodField(method_name=None)

method_name - The name of the method on the serializer to be called. If not included this defaults to get_<field_name>.

The serializer method referred to by the method_name argument should accept a single argument (in addition to self), which is the object being serialized. It should return whatever you want to be included in the serialized representation of the object.

Core Arguments

label - A short text string that may be used as the name of the field in HTML form fields or other descriptive elements.

validators - A list of validator functions which should be applied to the incoming field input, and which either raise a validation error or simply return. Validator functions should typically raise serializers.ValidationError, but Django's built-in ValidationError is also supported for compatibility with validators defined in the Django codebase or third party Django packages.

error_messages - A dictionary of error codes to error messages.

help_text - A text string that may be used as a description of the field in HTML form fields or other descriptive elements.

Core Arguments

required - Normally an error will be raised if a field is not supplied during deserialization. Set to false if this field is not required to be present during deserialization.

Setting this to False also allows the object attribute or dictionary key to be omitted from output when serializing the instance. If the key is not present it will simply not be included in the output representation.

Defaults to True.

default - If set, this gives the default value that will be used for the field if no input value is supplied. If not set the default behaviour is to not populate the attribute at all.

The default is not applied during partial update operations. In the partial update case only fields that are provided in the incoming data will have a validated value returned.

Note that setting a default value implies that the field is not required. Including both the default and required keyword arguments is invalid and will raise an error.

Core Arguments

initial - A value that should be used for pre-populating the value of HTML form fields.

style - A dictionary of key-value pairs that can be used to control how renderers should render the field.

Example:-

```
password = serializers.CharField(
        max_length=100,
        style={'input_type': 'password', 'placeholder': 'Password'}
    )
```

Core Arguments

read_only - Read-only fields are included in the API output, but should not be included in the input during create or update operations. Any 'read_only' fields that are incorrectly included in the serializer input will be ignored.

Set this to True to ensure that the field is used when serializing a representation, but is not used when creating or updating an instance during deserialization.

Defaults to False

write_only - Set this to True to ensure that the field may be used when updating or creating an instance, but is not included when serializing the representation.

Defaults to False

Core Arguments

allow_null - Normally an error will be raised if None is passed to a serializer field. Set this keyword argument to True if None should be considered a valid value.

Note that, without an explicit default, setting this argument to True will imply a default value of null for serialization output, but does not imply a default for input deserialization.

Defaults to False

source - The name of the attribute that will be used to populate the field. May be a method that only takes a self argument, such as URLField(source='get_absolute_url'), or may use dotted notation to traverse attributes, such as EmailField(source='user.email'). When serializing fields with dotted notation, it may be necessary to provide a default value if any object is not present or is empty during attribute traversal.

The value source='*' has a special meaning, and is used to indicate that the entire object should be passed through to the field. This can be useful for creating nested representations, or for fields which require access to the complete object in order to determine the output representation.

Defaults to the name of the field.