
django_dicom
Release 0.1.0

Jul 19, 2022

Contents:

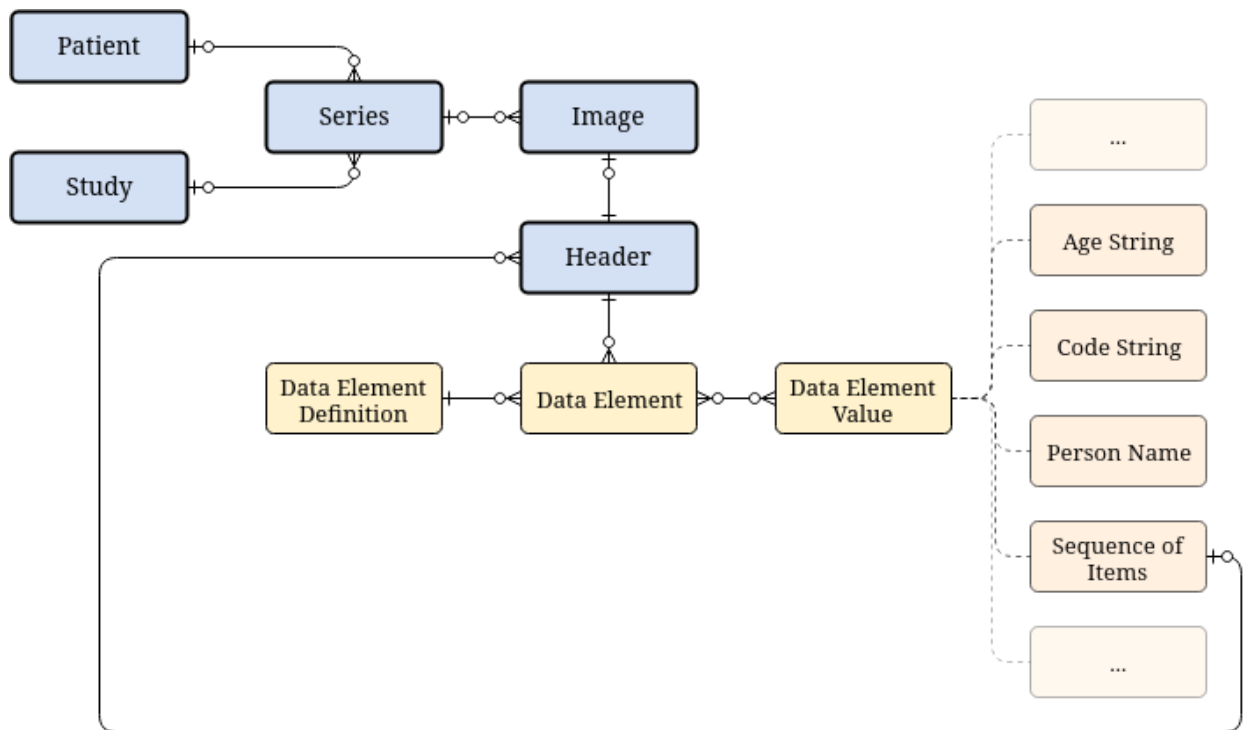
1	Overview	1
2	Installation	3
3	User Guide	5
3.1	Import Modes	5
3.2	Importing Data	5
3.3	Reading Header Information	6
3.4	Reading Series Data	8
4	Reference	11
4.1	Module contents	11
4.2	Subpackages	11
4.3	Submodules	58
4.4	django_dicom.urls module	58
5	Resources	59
5.1	General	59
5.2	Other	59
6	Indices and tables	61
	Python Module Index	63
	Index	65

CHAPTER 1

Overview

django_dicom is a reusable Django application built to maintain a database of DICOM data. It was created to support the *pylabber* project, but does not depend on it.

DICOM header information is represented using *Model* subclasses that represent the various entities it may contain (see the [DICOM standard specification](#) and [this blog post](#) for more information) and provides utility methods to import data and easily maintain the DICOM entities and their relationship.



The fundamental entities (colored in blue) will be created automatically whenever data is *imported to the database* by reading the required header information using *dicom_parser*.

The *import mode* configuration will determine which data elements will be serialized to the database under that *Header*.

CHAPTER 2

Installation

To install the latest version of *django_dicom*, simply run:

```
pip install django_dicom
```

Once installed:

- Add *django_dicom* to the `INSTALLED_APPS`:

Listing 1: settings.py

```
INSTALLED_APPS = [  
    ...  
    'django_dicom',  
]
```

- Include *django_dicom*'s URLs:

Listing 2: urls.py

```
urlpatterns = [  
    ...  
    path("api/", include("django_dicom.urls", namespace="dicom")),  
]
```

- Create the database tables:

```
python manage.py migrate django_dicom
```

Warning: *django_dicom* can only integrate with a PostgreSQL database. In order to set-up your project with PostgreSQL follow [this DjangoGirls tutorial](#).

- Start the development server and visit <http://127.0.0.1:8000/admin/> or <http://127.0.0.1:8000/dicom/>.

Once *django_dicom* is properly *installed* into your Django project, enter your project's shell by running `./manage.py shell` from within your project's root directory and with your *virtual environment* activated.

3.1 Import Modes

Currently, three import modes are available:

- Minimal: Do not create *Header* or *DataElement* instances at all.
- Normal: Create *Headers* and save only *standard data elements* to the database.
- Full: Create *Headers* and save all data elements to the database.

By default, *django_dicom*'s import mode will be set to `Normal`.

3.1.1 Changing Import Mode

In order to change the import mode, in your *project settings* add:

```
DICOM_IMPORT_MODE = "minimal" # or "full"
```

3.2 Importing Data

To import a local repository of *.dcm* files, use the *ImageManager*'s *import_path()* method:

```
>>> from django_dicom.models import Image

>>> path = '/path/to/local/dcm/repo/'
>>> results = Image.objects.import_path(path)
Importing DICOM data: 4312image [34:24, 2.09image/s]
```

(continues on next page)

(continued from previous page)

```
Successfully imported DICOM data from '/path/to/local/dcm/repo/'!
Created:          4312

>>> results
{'created': 4312, 'existing': 0}
```

You can verify the addition of new data to the database by querying the desired [DICOM](#) entity:

```
>>> from django_dicom.models import Image, Patient

>>> Image.objects.count()
4312
>>> Patient.objects.count()
3
```

3.3 Reading Header Information

django_dicom uses the [dicom_parser](#) package to read header information and store it as an *Header* instance. Some information is already available to us directly through the values stored in the associated *Image*, *Series*, *Patient*, and *Study* instances. However, *Header* instances are collections of *DataElement* instances, each storing a queryable reference to both the raw and parsed values of a given data element.

By default, *django_dicom* will store all official DICOM tags, skipping any private data elements. These will still be easily readable, however, we will not be able to query them from the database. For more information about import modes, see [Import Modes](#).

3.3.1 Reading a Single Image's Header

To read a single *Image* instance's header, we can simply use the *instance* property to retrieve [dicom_parser](#)'s representation of a DICOM image (*Image*):

```
>>> from django_dicom.models import Image

>>> image = Image.objects.first()
>>> image.instance.header
```

Tag	Keyword	VR	VM	Value
(0008, 0005)	SpecificCharacterSet	Code String	1	ISO_IR 100
(0008, 0008)	ImageType	Code String	5	['ORIGINAL', 'PRIMARY', ...]
(0008, 0012)	InstanceCreationDate	Date	1	2019-12-18
(0008, 0013)	InstanceCreationTime	Time	1	08:54:41.479000
(0008, 0016)	SOPClassUID	Unique Identifier	1	1.2.840.10008.5.1.4.1.1.4
(0008, 0018)	SOPInstanceUID	Unique Identifier	1	1.3.12.2.1107.5.2.43.660...
(0008, 0020)	StudyDate	Date	1	2019-12-18
... Private Data Elements				

```
=====
Tag          Keyword          VR          VM  Value
```

(continues on next page)

(continued from previous page)

```
(0019, 0010)                                Long String  1  SIEMENS MR HEADER
(0019, 1008)  CsaImageHeaderType             Unknown    1  b'IMAGE NUM 4 '
(0019, 1009)  CsaImageHeaderVersion??        Unknown    1  b'1.0 '
(0019, 100b)  SliceMeasurementDuration       Unknown    1
↪ 3535
...

>>> image.instance.header.get('PixelSpacing')
[0.48828125, 0.48828125]
```

For more information on `dicom_parser`'s classes and functionality, see [the documentation](#).

3.3.2 Querying the Database

According to the chosen *import mode*, no, some, or all of the saved images' data elements will be serialized to the database as `DataElement` instances. Each `DataElement` represents a single tag within a single header, however, the actual information is available its *definition* and *value* attributes. These attributes associate the `DataElement` instance with reusable `DataElementDefinition` and `DataElementValue` instances, thereby preventing data duplication simplifying queries.

To better understand the way header information is serialized in the database, let's query all the `Series` instances in which the underlying image headers contain a data element named `ImageType` which contains a value of `MOSAIC`.

First, we'll have a look at the relevant `DataElementDefinition` instance:

```
>>> from django_dicom.models import DataElementDefinition
>>> definition = DataElementDefinition.objects.get(keyword="ImageType")
>>> definition
<DataElementDefinition:
Tag              (0008, 0008)
Keyword          ImageType
Value Representation  CS
Description      Image Type>
```

Now, let's select any `DataElementValue` instances satisfying our condition:

```
>>> from django_dicom.models import CodeString
>>> values = CodeString.objects.filter(value="MOSAIC")
>>> values.count()
9
```

This means there are 9 different distinct `ImageType` values that contain the string `MOSAIC`. If we want all the related `DataElement` instances, we could:

```
>>> from django_dicom.models import DataElement
>>> data_elements = DataElement.objects.filter(definition=definition, _values__
↪ in=values)
>>> data_elements.count()
42236
```

Two important things to note here:

- We used the `_values` attribute to access the raw relationship between the `DataElement` and its associated `DataElementValues`.

- We used `__in` to query this [ManyToMany relationship](#) and retrieve data elements containing any of the desired values. This is necessary because DICOM elements may have multiple values (for more information see [value multiplicity](#)).

Now, if we would like to query all the images to which these data elements belong:

```
>>> image_ids = data_elements.values_list('header__image', flat=True)
>>> images = Image.objects.filter(id__in=image_ids)
```

or the series:

```
>>> from django_dicom.models import Series
>>> series_ids = images.values_list('series', flat=True)
>>> series = Series.objects.filter(id__in=series_ids)
>>> series.count()
409
>>> series.first().description
'IR-DTI_30dir_3iso'
```

3.4 Reading Series Data

django_dicom relies on *dicom_parser* and *pydicom*'s to convert an *Image*'s pixel data from bytes to a *NumPy ndarray*.

Let's query an anatomical ("MPRAGE"¹) *Series* instance we've added based on its *Series Description*:

```
>>> from django_dicom.models import Series
>>> series = Series.objects.filter(description__contains="MPRAGE").first()
```

Great! now all we need to do in order to get a *NumPy ndarray* of the underlying data would be to use the *data* property:

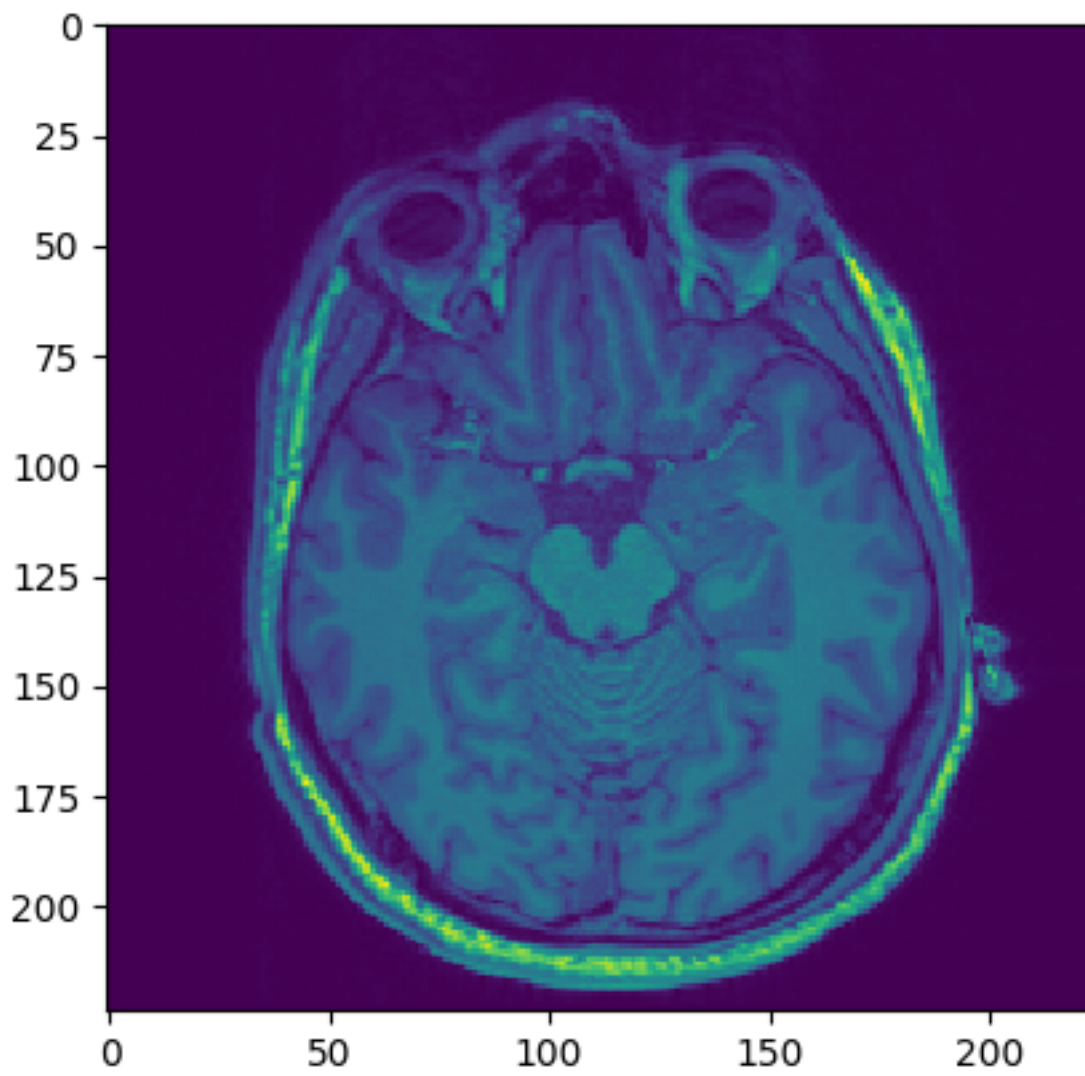
```
>>> data = series.get_data()
>>> series.data.shape
(224, 224, 208)
```

To inspect a particular slice, we could use *matplotlib*:

```
>>> import matplotlib.pyplot as plt
>>> plt.imshow(series.data[:, :, 100])
>>> plt.show()
```

This should return a figure similar to this:

¹ Brant-Zawadzki, M., Gillan, G. D., & Nitz, W. R. (1992). MP RAGE: a three-dimensional, T1-weighted, gradient-echo sequence—initial experience in the brain. *Radiology*, 182(3), 769-775.



4.1 Module contents

This application manages [DICOM](#) files by defining [Django models](#) that represent the various DICOM entities.

4.2 Subpackages

4.2.1 Exceptions

Module contents

Definition of custom exceptions that may be raised by *django_dicom*. For more information about error and exceptions see the [Python docs](#).

Submodules

`django_dicom.exceptions.import_error` module

Definition of the `DicomImportError` class.

exception `django_dicom.exceptions.dicom_import_error.DicomImportError`
Bases: `Exception`

Raised whenever importing new DICOM data to the database fails.

4.2.2 Filters

Module contents

Provides filter classes for the [REST API](#) endpoints (for more information see the [DRF docs](#) and [django-filter's integration with DRF docs](#)).

Submodules

django_{dicom}.filters.image_filter module

Definition of the *ImageFilter* class.

```
class djangodicom.filters.image_filter.ImageFilter (data=None, queryset=None, *, request=None, prefix=None)
```

Bases: `django_filters.rest_framework.filterset.FilterSet`

Provides filtering functionality for the *ImageViewSet*.

Available filters are:

- *id*: Primary key
- *series_uid*: Series instance UID (contains)
- *series_description*: Series description (contains)
- *number*: Series number (exact)
- *created_after_date*: Create after date
- *created_before_date*: Create before date
- *created_after_time*: Create after time
- *created_before_time*: Create before time

```
class Meta
```

Bases: `object`

```
fields = ('id', 'uid', 'number')
```

```
model
```

alias of `djangodicom.models.image.Image`

```
base_filters = {'created_after_date': <django_filters.filters.DateField object>, 'cr
```

```
declared_filters = {'created_after_date': <django_filters.filters.DateField object>,
```

django_{dicom}.filters.patient_filter module

Definition of the *FilterSet* subclass that will be assigned to the *PatientViewSet*'s *filter_class* attribute value.

```
class djangodicom.filters.patient_filter.PatientFilter (data=None, *, query-set=None, request=None, prefix=None)
```

Bases: `django_filters.rest_framework.filterset.FilterSet`

Provides filtering functionality for the *PatientViewSet*.

Available filters are:

- *id*: Primary key

- *uid*: Patient UID (contains, icontains, or exact)
- *born_after_date*: Earliest date of birth
- *born_before_date*: Latest date of birth
- *name_prefix*: Any of the existing *name_prefix* values in the database
- *given_name*: Given name value (contains, icontains, or exact)
- *middle_name*: Middle name value (contains, icontains, or exact)
- *family_name*: Family name value (contains, icontains, or exact)
- *name_suffix*: Any of the existing *name_prefix* values in the database
- *sex*: Any of the sex options defined in the `Sex Enum`
- *study__id*: Related *Study* ID

class Meta

Bases: `object`

fields = ('id', 'uid')

model

alias of `django_dicom.models.patient.Patient`

base_filters = {'date_of_birth': <django_filters.filters.DateFromToRangeFilter object>

declared_filters = {'date_of_birth': <django_filters.filters.DateFromToRangeFilter ob

filter_by_study (*queryset*: `django.db.models.query.QuerySet`, *name*: `str`, *value*: `int`) → `django.db.models.query.QuerySet`

Returns all *Patient* instances that have *Series* instances belonging to the *Study* with the specified *value* as primary key.

Used by *study__id*.

Parameters

- **queryset** (`QuerySet`) – *Patient* instances
- **name** (`str`) – Name of the queried filter field
- **value** (`int`) – *Study* primary key

Returns Filtered *Patient* instances

Return type `QuerySet`

django_dicom.filters.series_filter module

Definition of the *SeriesFilter* class.

class `django_dicom.filters.series_filter.SeriesFilter` (*data*=None, *queryset*=None, *, *request*=None, *prefix*=None)

Bases: `django_filters.rest_framework.filterset.FilterSet`

Provides filtering functionality for the *SeriesViewSet*.

Available filters are:

- *id*: Primary key
- *uid*: Series Instance UID

- *patient_id*: Related *Patient* instance's primary key
- *study_uid*: Related *Study* instance's *uid* value
- *study_description*: Related *Study* instance's *description* value (in-icontains)
- *modality*: Any of the values defined in *Modality*
- *description*: Series description value (contains, icontains, or exact)
- *number*: Series number value
- *protocol_name*: Protocol name value (contains)
- *scanning_sequence*: Any combination of the values defined in *ScanningSequence*
- *sequence_variant*: Any combination of the values defined in *SequenceVariant*
- *echo_time*: *echo_time* value
- *inversion_time*: *inversion_time* value
- *repetition_time*: *repetition_time* value
- *flip_angle*: Any of the existing *flip_angle* in the database
- *date_after*: Exact *date* value
- *date_before*: Create before date
- *time_after*: Create after time
- *time_before*: Create before time
- *manufacturer*: Any of the existing *manufacturer* in the database
- *manufacturer_model_name*: Any of the existing *manufacturer_model_name* in the database
- *device_serial_number*: Any of the existing *device_serial_number* in the database
- *institution_name*: Any of the existing *institution_name* in the database
- *pulse_sequence_name*: *pulse_sequence_name* value (in-icontains)
- *sequence_name*: *sequence_name* value (in-icontains)

class Meta

Bases: *object*

fields = ('id', 'uid', 'number', 'patient__id')

model

alias of *django_dicom.models.series.Series*

base_filters = {'date': <django_filters.filters.DateRangeFilter object>, 'description

declared_filters = {'date': <django_filters.filters.DateRangeFilter object>, 'descrip

django_dicom.filters.series_filter.filter_array (*queryset*:
django.db.models.query.QuerySet,
field_name: *str*, *value*: *list*)

Returns an exact lookup for a PostgreSQL *ArrayField*.

Parameters

- **queryset** (*QuerySet*) – The filtered queryset
- **field_name** (*str*) – The name of the field the queryset is being filtered by
- **value** (*list*) – The values to filter by

`django_dicom.filters.series_filter.filter_header` (*queryset:*
django.db.models.query.QuerySet,
field_name: str, values: str)

Returns a desired lookup for a DicomHeader field.

Parameters

- **queryset** (*QuerySet*) – The filtered queryset
- **field_name** (*str*) – The name of the field the queryset is being filtered by
- **values** (*dict*) – The fields and values to filter by

`django_dicom.filters.series_filter.filter_in_string` (*queryset:*
django.db.models.query.QuerySet,
field_name: str, values: list)

Returns a in-contains mixed lookup with ‘or’ between values for a CharField.

Parameters

- **queryset** (*QuerySet*) – The filtered queryset
- **field_name** (*str*) – The name of the field the queryset is being filtered by
- **values** (*str*) – The values to filter by

django_dicom.filters.study_filter module

Definition of the *StudyFilter* class.

class `django_dicom.filters.study_filter.StudyFilter` (*data=None, queryset=None, *,*
request=None, prefix=None)

Bases: `django_filters.rest_framework.filterset.FilterSet`

Provides filtering functionality for the *StudyViewSet*.

Available filters are:

- *id*: Primary key
- *uid*: Study instance UID
- *description*: Study description (contains, icontains, or exact)
- *created_after_date*: Create after date
- *created_before_date*: Create before date
- *created_after_time*: Create after time
- *created_before_time*: Create before time

class Meta

Bases: `object`

fields = ('id',)

model

alias of `django_dicom.models.study.Study`

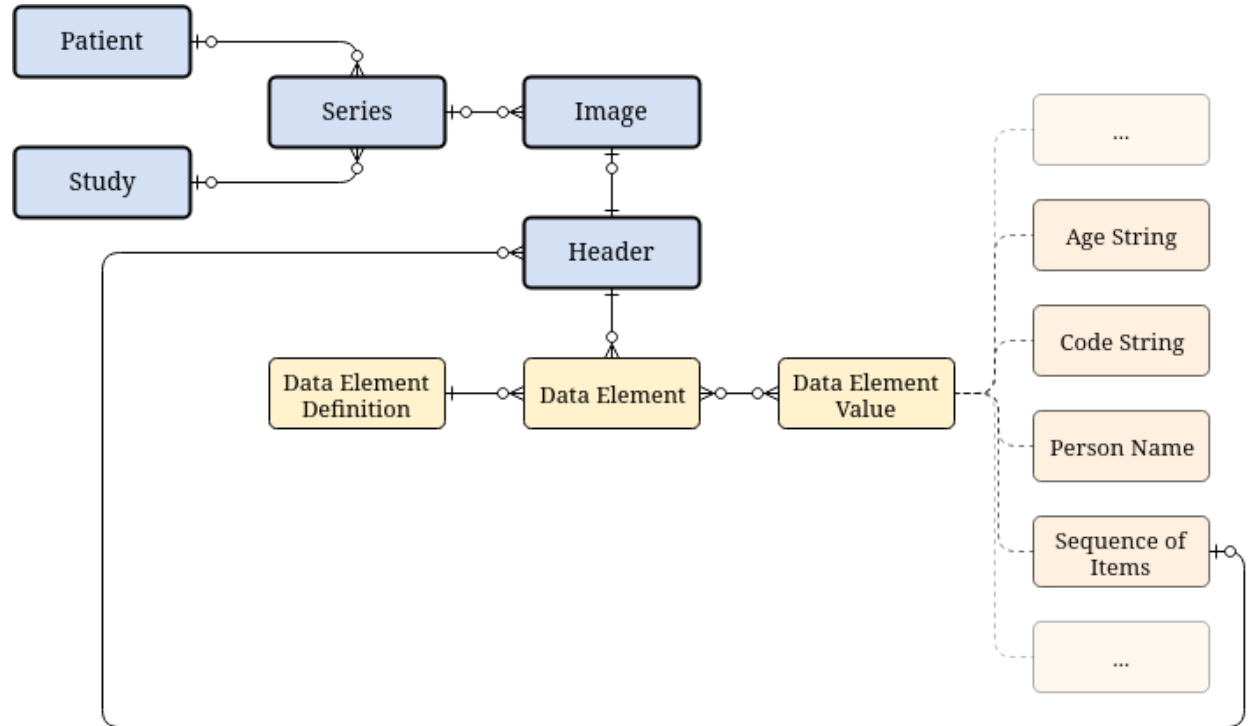
base_filters = {'date': <django_filters.filters.DateFromToRangeFilter object>, 'descr

declared_filters = {'date': <django_filters.filters.DateFromToRangeFilter object>, 'd

4.2.3 Models

Module contents

Creates `Model` subclasses to represent the various DICOM entities.



Subpackages

django_dicom.models.managers package

Module contents

`Manager` subclasses for some of the app's models.

References

- Django's documentation on managers.

Subpackages

django_dicom.models.managers.values package

Module contents

Custom managers for specific `DataElementValue` subclasses.

Submodules

django_dicom.models.managers.values.sequence_of_items module

Definition of the *SequenceOfItemsManager* class.

class django_dicom.models.managers.values.sequence_of_items.**SequenceOfItemsManager**
 Bases: *django_dicom.models.managers.data_element_value.DataElementValueManager*

Custom manager for the *SequenceOfItems* model.

from_dicom_parser (*data_element: dicom_parser.data_element.DataElement*) → tuple
 Create a sequence of items by reading the included headers and populating the database accordingly.

Parameters *data_element* (*DicomDataElement*) – Sequence of items data element

Returns The data element and whether it was created or not

Return type Tuple[*DataElementValue*, bool]

Submodules

django_dicom.models.managers.data_element module

Definition of a custom *Manager* for the *DataElement* model.

class django_dicom.models.managers.data_element.**DataElementManager**
 Bases: *django.db.models.manager.Manager*

Custom *Manager* for the *DataElement* model.

create_from_dicom_parser (*header, definition, data_element: dicom_parser.data_element.DataElement*)

Creates a new instance under *header* using the provided *definition* and *data_element*.

Parameters

- **header** (*Header*) – The header instance with which the created data element should be associated.
- **definition** (*DataElementDefinition*) – The data element definition of the created data element
- **data_element** (*dicom_parser.data_element.DataElement*) – Object representing a single data element in memory

Returns The created instance

Return type *DataElement*

from_dicom_parser (*header, data_element: dicom_parser.data_element.DataElement*)
 Creates a new instance under *header* using the provided *dicom_parser.data_element.DataElement* instance.

Parameters

- **header** (*Header*) – The header instance with which the created data element should be associated.
- **data_element** (*dicom_parser.data_element.DataElement*) – Object representing a single data element in memory

Returns The created instance

Return type *DataElement*

django_dicom.models.managers.data_element_definition module

Definition of a custom *Manager* for the *DataElementDefinition* model.

class django_dicom.models.managers.data_element_definition.**DataElementDefinitionManager**
 Bases: django.db.models.manager.Manager

Custom *Manager* for the *DataElementDefinition* model.

from_dicom_parser (*data_element: dicom_parser.data_element.DataElement*) → tuple
 Gets or creates a *DataElementDefinition* instance from a *dicom_parser* *DataElement*.

Parameters *data_element* (*dicom_parser.data_element.DataElement*) – Object representing a single data element in memory

Returns *data_element_definition*, created

Return type tuple

django_dicom.models.managers.data_element_definition.**data_element_to_definition** (*data_element: dicom_parser.data_element.DataElement*) → dict

Converts a *dicom_parser* *DataElement* to a dictionary of keyword arguments that may be used to instantiate a *DataElementDefinition* instance.

Parameters *data_element* (*dicom_parser.data_element.DataElement*) – Object representing a single data element in memory

Returns *DataElementDefinition* instantiation keyword arguments

Return type dict

django_dicom.models.managers.data_element_value module

Definition of a custom *InheritanceManager* for the *DataElementValue* model.

For more information about the *InheritanceManager* class, see *django-model-utils*'s *InheritanceManager* documentation.

class django_dicom.models.managers.data_element_value.**DataElementValueManager**
 Bases: model_utils.managers.InheritanceManager

Custom *InheritanceManager* for the *DataElementValue* model.

from_dicom_parser (*data_element: dicom_parser.data_element.DataElement*) → Tuple
 Get or create some *DataElementValue* subclass instances from a *dicom_parser* *DataElement*.

Parameters *data_element* (*dicom_parser.data_element.DataElement*) – Object representing a single data element in memory

Returns *data_element_value* or *data_element_values*, created

Return type Tuple[Union[*DataElementValue*, List[*DataElementValue*, ..]], bool]

get_or_create_from_nonsequence (*data_element*: *dicom_parser.data_element.DataElement*)
→ Tuple

Get or create some *DataElementValue* subclass instances from a non-Sequence of Items *dicom_parser.DataElement*.

Parameters *data_element* (*dicom_parser.data_element.DataElement*) – Object representing a single data element in memory

Returns *data_element_value* or *data_element_values*, created

Return type Tuple[Union[*DataElementValue*, List[*DataElementValue*, ..]], bool]

handle_invalid_data (*ValueModel*, *data_element*: *dicom_parser.data_element.DataElement*, *error*: *Exception*) → Tuple

If reading the value from the data element using *dicom_parser* raises an exception, this method is called to create an “empty” instance of the *ValueModel* (i.e. with *raw* and *value* set to *None*) and log the exception in the *warnings* field.

Parameters

- **ValueModel** (*DataElementValue*) – Some *DataElementValue* subclass used to instantiate values
- **data_element** (*dicom_parser.data_element.DataElement*) – Object representing a single data element in memory
- **error** (*Exception*) – The raised exception

Returns *data_element_value*, created

Return type tuple

handle_multiple_values (*ValueModel*, *data_element*: *dicom_parser.data_element.DataElement*)
→ Tuple[list, bool]

Handles data elements with a *value_multiplicity* greater than 1.

Parameters

- **ValueModel** (*DataElementValue*) – Some *DataElementValue* subclass used to instantiate values
- **data_element** (*dicom_parser.data_element.DataElement*) – Object representing a single data element in memory

Returns *data_element_values*, *any_created*

Return type Tuple[List[*DataElementValue*, ..], bool]

handle_no_value (*ValueModel*) → Tuple

Handles data elements with a *value_multiplicity* of 0. Returns an “empty” instance of the *ValueModel* (i.e. with *raw* and *value* set to *None*).

Parameters **ValueModel** (*DataElementValue*) – Some *DataElementValue* subclass used to instantiate values

Returns *dicom_data_element*, created

Return type Tuple[DicomDataElement, bool]

handle_single_value (*ValueModel*, *data_element*: *dicom_parser.data_element.DataElement*) → tuple

Handles data elements with a *value_multiplicity* of 1.

Parameters

- **ValueModel** (*DataElementValue*) – Some *DataElementValue* subclass used to instantiate values
- **data_element** (*dicom_parser.data_element.DataElement*) – Object representing a single data element in memory

Returns *data_element_value*, created

Return type *Tuple*[*DataElementValue*, *bool*]

handle_value_multiplicity (*ValueModel*, *data_element:* *dicom_parser.data_element.DataElement*) → *Tuple*

Handles the creation of the *DataElementValue* subclass instances according to the *dicom_parser.DataElement*'s *value_multiplicity* attribute.

Parameters

- **ValueModel** (*DataElementValue*) – Some *DataElementValue* subclass used to instantiate values
- **data_element** (*dicom_parser.data_element.DataElement*) – Object representing a single data element in memory

Returns *data_element_value* or *data_element_values*, created

Return type *Tuple*[*Union*[*DataElementValue*, *List*[*DataElementValue*, ..]], *bool*]

django_{dicom}.models.managers.dicom_entity module

Definition of a custom *Manager* for the *DicomEntity* model.

class *django_{dicom}.models.managers.dicom_entity.DicomEntityManager*

Bases: *django.db.models.manager.Manager*

Custom *Manager* for the *DicomEntity* model.

from_header (*header*) → *Tuple*

Get or create an instance using the provided header.

Parameters **header** (*django_{dicom}.models.header.Header*) – Header instance to query this entity's information from

Returns *dicom_entity*, created

Return type *Tuple*[*DicomEntity*, *bool*]

django_{dicom}.models.managers.header module

Definition of a custom *Manager* for the *Header* model.

class *django_{dicom}.models.managers.header.HeaderManager*

Bases: *django.db.models.manager.Manager*

Custom *Manager* for the *Header* model.

from_dicom_parser (*header: dicom_parser.header.Header*, ***kwargs*)

Creates a new instance from a *dicom_parser* *dicom_parser.header.Header*.

Parameters **header** (*dicom_parser.header.Header*) – Object representing an entire DICOM header in memory

Returns Created instance

Return type `django_dicom.models.header.Header`

Raises `DicomImportError` – DICOM header read error

django_dicom.models.managers.image module

Definition of the `ImageManager` class.

class `django_dicom.models.managers.image.ImageManager`

Bases: `django_dicom.models.managers.dicom_entity.DicomEntityManager`

Custom `Manager` for the `Image` model.

TEMP_DCM_FILE_NAME = `'tmp.dcm'`

Name given to DICOM files that need to be saved locally in order to be read.

create_from_dcm (*path*: `pathlib.Path`, *autoremove*: `bool = True`)

Creates an `Image` instance from a given path.

Parameters

- **path** (`pathlib.Path`) – Local `.dcm` file path
- **autoremove** (`bool`, *optional*) – Whether to remove the local copy of the `.dcm` file under `MEDIA_ROOT` if creation fails, by default `True`

Returns The created image

Return type `Image`

get_or_create (**args*, ***kwargs*) → `Tuple`

Overrides `get_or_create()` to call `get_or_create_from_dcm()` in case the `dcm` keyword argument is provided.

Returns image, created

Return type `Tuple[Image, bool]`

get_or_create_from_dcm (*path*: `pathlib.Path`, *autoremove*: `bool = True`) → `Tuple`

Gets or creates an `Image` instance based on the contents of the provided `.dcm` path.

Parameters

- **path** (`pathlib.Path`) – Local `.dcm` file path
- **autoremove** (`bool`, *optional*) – Whether to remove the local copy of the `.dcm` file under `MEDIA_ROOT` if creation fails, by default `True`

Returns image, created

Return type `Tuple[Image, bool]`

import_path (*path*: `pathlib.Path`, *progressbar*: `bool = True`, *report*: `bool = True`, *persistent*: `bool = True`, *pattern*: `bool = '*.dcm'`, *autoremove*: `bool = True`) → `django.db.models.query.QuerySet`

Iterates the given directory tree and imports any `.dcm` files found within it.

Parameters

- **path** (`pathlib.Path`) – Base path for recursive `.dcm` import
- **progressbar** (`bool`, *optional*) – Whether to display a progressbar or not, by default `True`

- **report** (*bool*, *optional*) – Whether to print out a summary report when finished or not, by default True
- **persistent** (*bool*, *optional*) – Whether to continue and raise a warning or to raise an exception when failing to read a DICOM file’s header
- **pattern** (*str*, *optional*) – Globbing pattern to use for file import

Returns The created *Image* instances

Return type `QuerySet`

```
report_import_path_results (path: pathlib.Path, counter: dict) → None
```

Reports the result of a recursive path import.

Parameters

- **path** (`pathlib.Path`) – Base path of DICOM data import
- **counter** (*dict*) – Dictionary containing *created* and *existing* keys containing the number of files which fit in each category.

```
store_image_data (image_data: _io.BufferedReader) → pathlib.Path
```

Stores binary image data to a temporary local path under the project's `MEDIA_ROOT`.

Parameters `image_data` (`io.BufferedReader`) – Binary DICOM image data

Returns Path of the created file

Return type `pathlib.Path`

django_dicom.models.utils package

Module contents

Utilities for the *models* module.

Submodules

django_dicom.models.utils.fields module

Custom `Field` subclasses.

[illegible]

Bases: `django.contrib.postgres.fields.array.ArrayField`

A field that allows us to store an array of choices. Uses Django 1.9's postgres ArrayField and a MultipleChoiceField for its formfield.

formfield (***kwargs*)

Return a `django.forms.Field` instance for this field.

django_dicom.models.utils.help_text module

`help_text` string constants for the various fields.

django_dicom.models.utils.logs module

Log messages to be used by models.

django_dicom.models.utils.meta module

`django_dicom.models.utils.meta.get_model(model_name: str) → django.db.models.base.Model`

django_dicom.models.utils.utils module

class `django_dicom.models.utils.utils.ImportMode`

Bases: `enum.Enum`

An enumeration.

FULL = 'Full'

MINIMAL = 'Minimal'

NORMAL = 'Normal'

`django_dicom.models.utils.utils.check_element_inclusion(data_element) → bool`

`django_dicom.models.utils.utils.get_dicom_root() → pathlib.Path`

`django_dicom.models.utils.utils.get_group_model()`

`django_dicom.models.utils.utils.get_import_configuration() → dict`

`django_dicom.models.utils.utils.get_import_mode() → django_dicom.models.utils.utils.ImportMode`

`django_dicom.models.utils.utils.get_mri_root() → pathlib.Path`

`django_dicom.models.utils.utils.get_subject_model()`

`django_dicom.models.utils.utils.snake_case_to_camel_case(string: str) → str`

django_dicom.models.utils.validators module

Definition of custom field validations.

`django_dicom.models.utils.validators.validate_file_extension(value)`

django_dicom.models.values package

Module contents

Definition of *DataElementValue* subclasses for every type of value representation (VR).

Hint: For more information about value representations see the [official DICOM standard \(part 05, section 6.2\)](#).

Submodules

django_dicom.models.values.age_string module

Definition of the *AgeString* model.

```
class django_dicom.models.values.age_string.AgeString(*args, **kwargs)
    Bases: django_dicom.models.values.data_element_value.DataElementValue
    A Model representing a single AgeString data element value.

    raw
        Overrides raw to assign a CharField.

    value
        Overrides value to assign a FloatField.
```

django_dicom.models.values.application_entity module

Definition of the *ApplicationEntity* model.

```
class django_dicom.models.values.application_entity.ApplicationEntity(*args,
                                                                           **kwargs)
    Bases: django_dicom.models.values.data_element_value.DataElementValue
    A Model representing a single ApplicationEntity data element value.

    raw
        Overrides raw to assign a CharField.

    value
        Overrides value to assign a CharField.
```

django_dicom.models.values.code_string module

Definition of the *CodeString* model.

```
class django_dicom.models.values.code_string.CodeString(*args, **kwargs)
    Bases: django_dicom.models.values.data_element_value.DataElementValue
    A Model representing a single CodeString data element value.

    raw
        Overrides raw to assign a CharField.

    value
        Overrides value to assign a CharField.
```

django_dicom.models.values.csa_header module

Definition of the *CsaHeader* model.

```
class django_dicom.models.values.csa_header.CsaHeader(*args, **kwargs)
    Bases: django_dicom.models.values.data_element_value.DataElementValue
    A Model representing a single Siemens' CSA header value.
```

Hint: For more information about CSA headers, see [dicom_parser's CSA headers documentation](#).

raw

Overrides `raw` to assign a `TextField`.

to_html (*verbose: bool = False, **kwargs*) → str

Returns the HTML representation of this instance.

If the *verbose* keyword argument is passed as *True*, returns the entire header as JSON. Otherwise, returns an HTML link to this instance in the admin site.

Parameters *verbose* (*bool, optional*) – Whether to return all of the header information or just a link, by default *False*

Returns HTML text containing either JSON encoded header information or a link to the admin site

Return type str

value

Overrides `value` to assign a `JSONField`.

django_dicom.models.values.data_element_value module

Definition of the `DataElementValue` model.

class django_dicom.models.values.data_element_value.**DataElementValue** (**args, **kwargs*)

Bases: `django.db.models.base.Model`

A parent `Model` representing a single value contained by some `DataElement` instance. If the data element has value multiplicity greater than 1, it will have multiple instances of this model associated with it, each with its own *index* value.

admin_link

Creates an HTML tag to link to this instance within the admin site.

Returns Link to this instance in the admin site

Return type str

agestring

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

`Place.restaurant` is a `ReverseOneToOneDescriptor` instance.

applicationentity

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

`Place.restaurant` is a `ReverseOneToOneDescriptor` instance.

codestring

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):  
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

csaheader

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):  
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

data_element_set

Accessor to the related objects manager on the forward and reverse sides of a many-to-many relation.

In the example:

```
class Pizza (Model):  
    toppings = ManyToManyField(Topping, related_name='pizzas')
```

Pizza.toppings and Topping.pizzas are ManyToManyDescriptor instances.

Most of the implementation is delegated to a dynamically defined manager class built by create_forward_many_to_many_manager() defined below.

date

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):  
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

datetime

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):  
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

decimalstring

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):  
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

floatingpointdouble

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

floatingpointsingle

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

get_raw_peek (*size: int = 100*) → str

Returns a truncated string of the raw data element's value (appended with "... " if changed).

Parameters *size* (*int*, *optional*) – Maximal string length, by default 100

Returns Truncated string

Return type str

index

If the value is one of a number of values within a DataElement (a DataElement with a value multiplicity that is greater than 1), this field keeps the index of this value.

integerstring

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

longstring

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

longtext

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

objects = <django_dicom.models.managers.data_element_value.DataElementValueManager object>

otherword

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

personname

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

raw = None

Raw data element value (meant to be overridden by child models).

sequenceofitems

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

shortstring

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

shorttext

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

signedlong

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```


Place.restaurant is a ReverseOneToOneDescriptor instance.

signedshort

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

time

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

to_html (**kwargs) → str

Returns the HTML representation of this instance. This method simply returns the *value*, child models should override to provide an appropriate representation.

Returns HTML representation of this instance

Return type str

uniqueidentifier

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

unknown

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

unlimitedtext

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

unsignedlong

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant(Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

unsignedshort

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant(Model):
    place = OneToOneField(Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

value = None

Interpreted data element value (meant to be overridden by child models).

warnings

If any warnings were raised by *dicom_parser*, log them in the database.

django_{dicom}.models.values.date module

Definition of the *Date* model.

```
class djangodicom.models.values.date.Date(*args, **kwargs)
    Bases: djangodicom.models.values.data_element_value.DataElementValue
    A Model representing a single Date data element value.

    raw
        Overrides raw to assign a CharField.

    value
        Overrides value to assign a CharField.
```

django_{dicom}.models.values.datetime module

Definition of the *DateTime* model.

```
class djangodicom.models.values.datetime.DateTime(*args, **kwargs)
    Bases: djangodicom.models.values.data_element_value.DataElementValue
    A Model representing a single DateTime data element value.

    get_next_by_value(*, field=<django.db.models.fields.DateTimeField: value>, is_next=True,
                      **kwargs)

    get_previous_by_value(*, field=<django.db.models.fields.DateTimeField: value>,
                          is_next=False, **kwargs)

    raw
        Overrides raw to assign a CharField.

    value
        Overrides value to assign a CharField.
```

django_{dicom}.models.values.decimal_string module

Definition of the *DecimalString* model.

```
class djangodicom.models.values.decimal_string.DecimalString(*args, **kwargs)
    Bases: djangodicom.models.values.data_element_value.DataElementValue
    A Model representing a single DecimalString data element value.

    raw
        Overrides raw to assign a CharField.

    value
        Overrides value to assign a FloatField.
```

django_{dicom}.models.values.floating_point_double module

Definition of the *FloatingPointDouble* model.

```
class djangodicom.models.values.floating_point_double.FloatingPointDouble(*args,
                                                                            **kwargs)
    Bases: djangodicom.models.values.data_element_value.DataElementValue
    A Model representing a single FloatingPointDouble data element value.

    raw
        Overrides raw to assign a CharField.

    value
        Overrides value to assign a FloatField.
```

django_{dicom}.models.values.floating_point_single module

Definition of the *FloatingPointSingle* model.

```
class djangodicom.models.values.floating_point_single.FloatingPointSingle(*args,
                                                                              **kwargs)
    Bases: djangodicom.models.values.data_element_value.DataElementValue
    A Model representing a single FloatingPointSingle data element value.

    raw
        Overrides raw to assign a CharField.

    value
        Overrides value to assign a FloatField.
```

django_{dicom}.models.values.integer_string module

Definition of the *IntegerString* model.

```
class djangodicom.models.values.integer_string.IntegerString(*args, **kwargs)
    Bases: djangodicom.models.values.data_element_value.DataElementValue
    A Model representing a single IntegerString data element value.

    raw
        Overrides raw to assign a CharField.
```

value

Overrides *value* to assign a `IntegerField`.

`django_dicom.models.values.integer_string.MAX_VALUE = 2147483647`

Maximal *IntegerString* value.

`django_dicom.models.values.integer_string.MIN_VALUE = -2147483648`

Minimal *IntegerString* value.

django_dicom.models.values.long_string module

Definition of the *LongString* model.

```
class django_dicom.models.values.long_string.LongString(*args, **kwargs)
    Bases: django_dicom.models.values.data_element_value.DataElementValue
    A Model representing a single LongString data element value.
```

raw

Overrides *raw* to assign a `CharField`.

value

Overrides *value* to assign a `CharField`.

django_dicom.models.values.long_text module

Definition of the *LongText* model.

```
class django_dicom.models.values.long_text.LongText(*args, **kwargs)
    Bases: django_dicom.models.values.data_element_value.DataElementValue
    A Model representing a single LongText data element value.
```

raw

Overrides *raw* to assign a `CharField`.

value

Overrides *value* to assign a `TextField`.

django_dicom.models.values.other_word module

Definition of the *OtherWord* model.

```
class django_dicom.models.values.other_word.OtherWord(*args, **kwargs)
    Bases: django_dicom.models.values.data_element_value.DataElementValue
    A Model representing a single OtherWord data element value.
```

raw

Overrides *raw* to assign a `BinaryField`.

value

Overrides *value* to assign a `IntegerField`.

django_{dicom}.models.values.person_name module

Definition of the *PersonName* model.

```
class djangodicom.models.values.person_name.PersonName (*args, **kwargs)
    Bases: djangodicom.models.values.data_element_value.DataElementValue
    A Model representing a single PersonName data element value.

    raw
        Overrides raw to assign a BinaryField.

    to_html (**kwargs) → str
        Returns the HTML representation of this instance.

        Returns HTML representation of this instance

        Return type str

    value
        Overrides value to assign a JSONField.
```

django_{dicom}.models.values.sequence_of_items module

Definition of the *SequenceOfItems* model.

```
class djangodicom.models.values.sequence_of_items.SequenceOfItems (*args,
                                                                    **kwargs)
    Bases: djangodicom.models.values.data_element_value.DataElementValue
    A django.db.models.Model representing the value stored in a single DataElement with a value representation (VR) of SQ.

    Sequence of Items data elements are arrays of nested headers, and therefore this model does not override the raw and value field definitions (and so they remain None).
```

header_set
 Accessor to the related objects manager on the reverse side of a many-to-one relation.
 In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Parent.children is a *ReverseManyToOneDescriptor* instance.

Most of the implementation is delegated to a dynamically defined manager class built by *create_forward_many_to_many_manager()* defined below.

```
objects = <djangodicom.models.managers.values.sequence_of_items.SequenceOfItemsManager>
```

```
to_html (verbose: bool = False, **kwargs) → str
    Returns the HTML representation of this instance. This method simply returns the value, child models should override to provide an appropriate representation.

    Returns HTML representation of this instance

    Return type str
```

django_dicom.models.values.short_string module

Definition of the *ShortString* model.

```
class django_dicom.models.values.short_string.ShortString(*args, **kwargs)
    Bases: django_dicom.models.values.data_element_value.DataElementValue
    A Model representing a single ShortString data element value.

    raw
        Overrides raw to assign a CharField.

    value
        Overrides value to assign a CharField.
```

django_dicom.models.values.short_text module

Definition of the *ShortText* model.

```
class django_dicom.models.values.short_text.ShortText(*args, **kwargs)
    Bases: django_dicom.models.values.data_element_value.DataElementValue
    A Model representing a single ShortText data element value.

    raw
        Overrides raw to assign a CharField.

    value
        Overrides value to assign a CharField.
```

django_dicom.models.values.signed_long module

Definition of the *SignedLong* model.

```
django_dicom.models.values.signed_long.MAX_VALUE = 2147483647
    Maximal SignedLong value.

django_dicom.models.values.signed_long.MIN_VALUE = -2147483648
    Minimal SignedLong value.

class django_dicom.models.values.signed_long.SignedLong(*args, **kwargs)
    Bases: django_dicom.models.values.data_element_value.DataElementValue
    A Model representing a single SignedLong data element value.

    raw
        Overrides raw to assign an IntegerField.

    value
        Overrides value to assign an IntegerField.
```

django_dicom.models.values.signed_short module

Definition of the *SignedShort* model.

```
django_dicom.models.values.signed_short.MAX_VALUE = 32767
    Maximal SignedShort value.
```

`django_dicom.models.values.signed_short.MIN_VALUE = -32768`
 Minimal *SignedShort* value.

class `django_dicom.models.values.signed_short.SignedShort(*args, **kwargs)`
 Bases: `django_dicom.models.values.data_element_value.DataElementValue`
 A `Model` representing a single *SignedShort* data element value.

raw
 Overrides `raw` to assign an `IntegerField`.

value
 Overrides `value` to assign an `IntegerField`.

django_dicom.models.values.time module

Definition of the *Time* model.

class `django_dicom.models.values.time.Time(*args, **kwargs)`
 Bases: `django_dicom.models.values.data_element_value.DataElementValue`
 A `Model` representing a single *Time* data element value.

raw
 Overrides `raw` to assign a `CharField`.

to_html (`**kwargs`) → `str`
 Returns the HTML representation of this instance.
 Returns HTML representation of this instance
 Return type `str`

value
 Overrides `value` to assign a `TimeField`.

django_dicom.models.values.unique_identifier module

Definition of the *UniqueIdentifier* model.

class `django_dicom.models.values.unique_identifier.UniqueIdentifier(*args, **kwargs)`
 Bases: `django_dicom.models.values.data_element_value.DataElementValue`
 A `Model` representing a single *UniqueIdentifier* data element value.

raw
 Overrides `raw` to assign a `CharField`.

value
 Overrides `value` to assign a `CharField`.

django_dicom.models.values.unknown module

Definition of the *Unknown* model.

class `django_dicom.models.values.unknown.Unknown(*args, **kwargs)`
 Bases: `django_dicom.models.values.data_element_value.DataElementValue`
 A `Model` representing a single *Unknown* data element value.

raw
 Overrides *raw* to assign a `TextField`.

value
 Overrides *value* to assign a `TextField`.

django_{dicom}.models.values.unlimited_text module

Definition of the *UnlimitedText* model.

```
class djangodicom.models.values.unlimited_text.UnlimitedText (*args, **kwargs)
    Bases: djangodicom.models.values.data_element_value.DataElementValue
    A Model representing a single Unknown data element value.
```

raw
 Overrides *raw* to assign a `TextField`.

value
 Overrides *value* to assign a `TextField`.

django_{dicom}.models.values.unsigned_long module

Definition of the *UnsignedLong* model.

```
djangodicom.models.values.unsigned_long.MAX_VALUE = 4294967296
    Maximal UnsignedLong value.
```

```
class djangodicom.models.values.unsigned_long.UnsignedLong (*args, **kwargs)
    Bases: djangodicom.models.values.data_element_value.DataElementValue
    A Model representing a single UnsignedLong data element value.
```

raw
 Overrides *raw* to assign a `PositiveIntegerField`.

value
 Overrides *value* to assign a `PositiveIntegerField`.

django_{dicom}.models.values.unsigned_short module

Definition of the *UnsignedShort* model.

```
djangodicom.models.values.unsigned_short.MAX_VALUE = 65536
    Maximal UnsignedShort value.
```

```
class djangodicom.models.values.unsigned_short.UnsignedShort (*args, **kwargs)
    Bases: djangodicom.models.values.data_element_value.DataElementValue
    A Model representing a single UnsignedShort data element value.
```

raw
 Overrides *raw* to assign a `PositiveIntegerField`.

value
 Overrides *value* to assign a `PositiveIntegerField`.

definition

The *DataElementDefinition* # noqa: E501 instance holding information about this element's DICOM tag.

header

The *Header* instance to which this data element belongs.

objects = <django_dicom.models.managers.data_element.DataElementManager object>

to_html (**kwargs) → str

Returns an HTML representation of this instance.

Any keyword arguments will be passed to the associated *DataElementValue* subclass instances.

Returns HTML representaion of this instance

Return type str

to_verbose_dict () → dict

Returns a dictionary representation of this instance.

Returns This instance's information

Return type dict

to_verbose_series () → pandas.core.series.Series

Returns a *Series* representation of this instance.

Returns This instance's information

Return type pandas.Series

value

Returns the value or values (according to the *value multiplicity*) of the associated *DataElementValue* instances.

Returns Data element value

Return type Any

value_multiplicity

Returns the number of *DataElementValue* related to this instance.

Returns Value multiplicity

Return type int

Hint: For more information see the DICOM standard's definition of *value multiplicity*.

django_dicom.models.data_element_definition module

Definition of the *DataElementDefinition* class.

class django_dicom.models.data_element_definition.DataElementDefinition(*args,
**kwargs)

Bases: django.db.models.base.Model

A model representing a single DICOM data element definition.

Notes

For more information see the [DICOM standard's documentation](#) as well as [registry of data elements](#).

admin_link

Returns an HTML tag linking to this instance in the admin site.

Returns HTML link to this instance

Return type `str`

data_element_set

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

description

A short description of this data element.

get_value_representation_display (*, *field=<django.db.models.fields.CharField: value_representation>*)

keyword

Most data elements have some keyword that facilitates querying header information.

objects = `<django_dicom.models.managers.data_element_definition.DataElementDefinitionManager>`

tag

Data element tags are an ordered pair of 16-bit unsigned integers representing the *Group Number* followed by *Element Number* and they are represented in the database using an array (list) of two four character strings.

to_dict () → dict

Returns a dictionary representation of this instance.

Returns This instance's dictionary representation

Return type `dict`

to_series () → `pandas.core.series.Series`

Returns a `Series` representation of this instance.

Returns Series representation of this instance

Return type `pandas.Series`

value_representation

The value representation (VR) defines the type of information stored in the data element. This will be used to determine which `DataElementValue` subclass is instantiated to save this information to the database.

django_dicom.models.dicom_entity module

Definition of the `DicomEntity` class. This [abstract model](#) serves as a base class for the various [DICOM entities](#) (e.g. `Study` and `Patient`).

```
class django_dicom.models.dicom_entity.DicomEntity(*args, **kwargs)
    Bases: django_extensions.db.models.TimeStampedModel

    Abstract model providing common functionality for DICOM entities. For more information about DICOM
    entites, see this introduction.

    FIELD_TO_HEADER = {}
        A dictionary used to convert field names to header keywords.

    class Meta
        Bases: object

        abstract = False

    admin_link
        Calls get\_admin\_link\(\) to return a link to this instance's page in the admin site.

        Returns HTML element

        Return type str

    get_admin_link(text: str = None) → str
        Returns an HTML link to this instance's page in the admin site.

        Parameters text (str, optional) – Text to display, by default None

        Returns HTML element

        Return type str

    get_header_fields() → list
        Returns a list of the derived model's fields which represent DICOM header information.

        Returns Fields representing DICOM header information

        Return type list

    classmethod get_header_keyword(field_name: str) → str
        Returns the data element keyword to return the requested field's value from header data. Relies on the
        derived model's FIELD_TO_HEADER class attribute. If no matching key is found, will simply return the
        field's name in CamelCase formatting (the formatting of pydicom's header keywords).

        Returns pydicom header keyword

        Return type str

    is_header_field(field: django.db.models.fields.Field) → bool
        Returns a boolean indicating whether the provided field represents DICOM header information or not.

        Parameters field (Field) – Field in question

        Returns Whether this field represent DICOM header information or not

        Return type bool

    log_creation() → None
        Logs the creation of a new instance of the calling Entity.

    objects

    save(*args, **kwargs)
        Overrides save\(\) to create logs and update the instance's fields from header information if provided.

    update_fields_from_header(header, exclude: list = None) → None
        Update fields from header data.

        Parameters
```

- **header** (`dicom_parser.header.Header`) – DICOM header data
- **exclude** (`list`, `optional`) – Field names to exclude, default is []

django_dicom.models.header module

Definition of the *Header* class.

class django_dicom.models.header.**Header** (*args, **kwargs)
 Bases: model_utils.models.TimeStampedModel

A model representing a single DICOM Data Set.

admin_link

Creates an HTML tag to link to this instance within the admin site.

Returns Link to this instance in the admin site

Return type str

data_element_set

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

Parent.children is a ReverseManyToOneDescriptor instance.

Most of the implementation is delegated to a dynamically defined manager class built by create_forward_many_to_many_manager() defined below.

get_entity_uid (entity: django_dicom.models.dicom_entity.DicomEntity) → str

Returns the UID of the provided DICOM entity from the header information.

Parameters entity (*DicomEntity*) – The entity for which a UID is desired

Returns The provided DICOM entity’s unique identifier

Return type str

get_or_create_entity (entity: django_dicom.models.dicom_entity.DicomEntity) → tuple

Get or create a DICOM entity’s instance from this header.

Parameters entity (*DicomEntity*) – The desired DICOM entity’s model

Returns instance, created

Return type tuple

get_or_create_patient () → tuple

Get or create the *Patient* instance corresponding to this header.

Returns instance, created

Return type tuple

get_or_create_series () → tuple

Get or create the *Series* instance corresponding to this header.

Returns instance, created

Return type tuple

get_or_create_study () → tuple

Get or create the *Study* instance corresponding to this header.

Returns instance, created

Return type tuple

get_value_by_keyword (keyword: str) → Any

Returns a data element's value by keyword **from the database** (not from the DICOM header itself, ergo only elements that are saved to the database may be queried).

Parameters keyword (str) – Data element keyword

Returns Data element value

Return type Any

image

Accessor to the related object on the reverse side of a one-to-one relation.

In the example:

```
class Restaurant (Model) :
    place = OneToOneField (Place, related_name='restaurant')
```

Place.restaurant is a ReverseOneToOneDescriptor instance.

index

This Data Set's index in the sequence (if *parent* is not None).

instance

Caches the created `dicom_parser.header.Header` instance to prevent multiple reads.

Returns Header information

Return type `dicom_parser.header.Header`

objects = <django_dicom.models.managers.header.HeaderManager object>

parent

Data Set <http://dicom.nema.org/medical/dicom/current/output/chtml/part05/chapter_7.html>'_s 'may be nested. If this header (Data Set) is an item of some sequence, this field holds that reference.

to_html (verbose: bool = False, **kwargs) → str

Returns an HTML representation of this instance.

Parameters verbose (bool, optional) – Whether to return a JSON with all of the header information or just a link to this instance's admin page, by default False.

Returns HTML representaion of this instance

Return type str

to_verbose_list () → List[dict]

Returns a list of dictionaries containing the information from the data elements in this header.

Returns Header information as a list of dictionaries

Return type List[dict]

django_dicom.models.image module

Definition of the *Image* class.

```
class django_dicom.models.image.Image(*args, **kwargs)
    Bases: django_dicom.models.dicom_entity.DicomEntity

    A model to represent a single instance of the Image entity. This model is normally instantiated with the dcm
    field set to some .dcm file from which the header information is read.

    FIELD_TO_HEADER = {'date': 'InstanceCreationDate', 'number': 'InstanceNumber', 'time'
        A dictionary of DICOM data element keywords to be used to populate a created instance's fields.

    create_header_instance() → django_dicom.models.header.Header
        Creates a Header instance from a dicom_parser.header.Header.

        Returns Created instance

        Return type Header

    data
        Facilitates access to the Image instance's data.

        Returns The image's pixel data

        Return type np.ndarray

    date
        Instance Creation Date value.

    dcm
        A reference to the DICOM image file.

    default_path
        Default unique path for this image based on its header information.

        Returns Default image location

        Return type pathlib.Path

    get_absolute_url() → str
        Returns the absolute URL for this instance. For more information see the Django documentation.

        Returns This instance's absolute URL path

        Return type str

    get_default_path() → pathlib.Path
        Returns a unique default path under MEDIA_ROOT for this instance based on its header information.

        Returns This instance's default location

        Return type pathlib.Path

    header
        The associated Header instance representing this image's header information.

    instance
        Caches the created dicom_parser.image.Image instance to prevent multiple reads.

        Returns Image information

        Return type dicom_parser.image.Image

    logger = <Logger data.dicom.image (WARNING)>

    number
        Instance Number value.

    objects = <django_dicom.models.managers.image.ImageManager object>
```

patient

Returns the *Patient* associated with this image.

Returns Associated *Patient* instance

Return type *Patient*

rename (*target: pathlib.Path*) → None

Move the .dcm file this instance references to some target destination.

Parameters **target** (*pathlib.Path*) – Destination path

save (**args, rename: bool = True, **kwargs*)

Overrides *save()* to check for missing header information or associated DICOM entities and create them if a .dcm file is provided.

Parameters **rename** (*bool, optional*) – Whether to move the file this instance is a reference to to a default path under MEDIA_ROOT or not, by default True

sequence_type

Returns the MRI sequence type detected by *dicom_parser*.

Returns Sequence type identifier

Return type *str*

series

The *Series* instance to which this image belongs.

time

Instance Creation Time value.

uid

SOP Instance UID value.

warnings

In case any warnings are raised by *dicom_parser* upon reading the image's header information, they are stored in this field.

django_dicom.models.patient module

Definition of the *Patient* class.

class *django_dicom.models.patient.Patient* (**args, **kwargs*)

Bases: *django_dicom.models.dicom_entity.DicomEntity*

A model to represent a single instance of the *Patient* entity.

FIELD_TO_HEADER = {'date_of_birth': 'PatientBirthDate', 'sex': 'PatientSex', 'uid':

A dictionary of DICOM data element keywords to be used to populate a created instance's fields.

date_of_birth

Patient Birth Date value.

family_name

Patient's Name value.

get_absolute_url () → *str*

Returns the absolute URL for this instance.

Returns This instance's absolute URL path

Return type *str*

get_full_name () → str
Returns the first and last names of the patient.
Returns Patient's first and last names.
Return type str

get_sex_display (*,field=<django.db.models.fields.CharField: sex>)

given_name
Patient's Name value.

logger = <Logger data.dicom.patient (WARNING)>

middle_name
Patient's Name value.

n_images
Returns the number of associated images.
See also:

- `query_n_images()`

Returns Number of associated images
Return type int

n_series
Returns the number of associated series.
Returns Number of associated series
Return type int

n_studies
Returns the number of associated studies.
See also:

- `query_n_studies()`

Returns Number of associated studies
Return type int

name_prefix
Patient's Name value.

name_suffix
Patient's Name value.

objects = <django.db.models.manager.DicomEntityManagerFromPatientQuerySet object>

query_n_images () → int
Returns the number of associated images.
See also:

- `n_images()`

Returns Number of associated images
Return type int

query_n_studies() → int

Returns the number of associated studies.

See also:

`n_studies()`

Returns Number of associated studies

Return type int

query_research_subject()

Returns the associated research subject, if such a model is registered and a matching instance exists.

See also:

- `research_subject()`

Returns Associated research subject

Return type Subject

research_subject

Returns the associated research subject, if such a model is registered and a matching instance exists.

See also:

- `query_research_subject()`

Returns Associated research subject

Return type Subject

series_set

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

sex

Patient's Sex value.

uid

Patient ID value.

update_fields_from_header(header, exclude: list = None) → None

Overrides `update_fields_from_header()` to handle setting the name parts.

Parameters

- **header** (`Header`) – DICOM header information.
- **exclude** (`list`, *optional*) – Field names to exclude (the default is [], which will not exclude any header fields).

update_patient_name (*header*) → None

Parses the patient's name from the DICOM header and updates the instance's fields.

Parameters **header** (*Header*) – A DICOM image's *Header* instance.

django_dicom.models.series module

Definition of the *Series* class.

class django_dicom.models.series.**Series** (*args, **kwargs)

Bases: *django_dicom.models.dicom_entity.DicomEntity*

A model to represent a single instance of the *Series* entity.

FIELD_TO_HEADER = {'date': 'SeriesDate', 'description': 'SeriesDescription', 'mr_acq'

A dictionary of DICOM data element keywords to be used to populate a created instance's fields.

MR_ACQUISITION_2D = '2D'

MR Acquisition Type value.

MR_ACQUISITION_3D = '3D'

MR_ACQUISITION_TYPE_CHOICES = (('2D', '2D'), ('3D', '3D'))

body_part_examined

Body Part Examined value.

data

Returns the `dicom_parser.series.Series.data` property's value.

Returns Series data

Return type `np.ndarray`

date

Series Date value.

datetime

Returns a `datetime.datetime` object by combining the values of the *date* and *time* fields.

Returns Series datetime

Return type `datetime.datetime`

description

Series Description value.

device_serial_number

Device Serial Number value.

echo_time

Echo Time value.

echo_train_length

Echo Train Length value.

flip_angle

Flip Angle value.

get_absolute_url () → str

Returns the absolute URL for this instance. For more information see the [Django documentation](#).

Returns This instance's absolute URL path

Return type `str`

get_file_paths () → Tuple[pathlib.Path]

get_modality_display (*, field=<django.db.models.fields.CharField: modality>)

get_mr_acquisition_type_display (*, field=<django.db.models.fields.CharField: mr_acquisition_type>)

get_path () → pathlib.Path

Returns the base directory containing the images composing this series.

Returns This series's base directory path

Return type str

get_patient_position_display (*, field=<django.db.models.fields.CharField: patient_position>)

get_sample_header () → dicom_parser.header.Header

Return a sample `Header` instance for this series.

See also:

- `sample_header()`

Returns Sample image header information

Return type DicomHeader

get_scanning_sequence_display () → list

Returns the display value of this instance's `scanning_sequence` attribute.

Returns Verbose scanning sequence values

Return type list

get_sequence_type_display (*, field=<django.db.models.fields.CharField: sequence_type>)

get_sequence_variant_display () → list

Returns the display value of this instance's `sequence_variant` attribute.

Returns Verbose sequence variant values

Return type list

image_set

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

instance

Caches the created `dicom_parser.series.Series` instance to prevent multiple reads.

Returns Series information

Return type `dicom_parser.series.Series`

institution_name

Institution Name value.

inversion_time
Inversion Time value.

logger = <Logger data.dicom.series (WARNING)>

magnetic_field_strength
Magnetic Field Strength value.

manufacturer
Manufacturer value.

manufacturer_model_name
Manufacturer's Model Name value.

missing_relation
Returns whether this instance misses an associated *Patient* or *Study*.

Returns Whether this instance has missing relationships

Return type bool

modality
Modality value.

mr_acquisition_type
A wrapper for a deferred-loading field. When the value is read from this object the first time, the query is executed.

number
Series Number value.

operators_name
Operator's Name value.

path
Returns the base path of this series' images.

Returns Series directory path

Return type `pathlib.Path`

patient
The *Patient* instance to which this series belongs.

patient_position
Patient Position value.

pixel_spacing
Pixel Spacing value.

protocol_name
Protocol Name value.

pulse_sequence_name
Pulse Sequence Name value.

repetition_time
Repetition Time value.

sample_header
Return a sample *Header* instance for this series.

See also:

- `get_sample_header()`

Returns Sample image header information

Return type DicomHeader

save (*args, **kwargs) → None

Overrides *save()* to create any missing related DICOM entities if required.

scanning_sequence

Scanning Sequence value.

sequence_name

Sequence Name value.

sequence_type

Scanning sequence type identifier, as detected by *dicom_parser*.

sequence_variant

Sequence Variant value.

slice_thickness

Slice Thickness value.

spatial_resolution

Returns the 3D spatial resolution of the instance by combining the values of the *pixel_spacing* and *slice_thickness* fields.

Returns (x, y, z) resolution in millimeters

Return type tuple

study

The *Study* instance to which this series belongs.

time

Series Time value.

uid

Series Instance UID value.

update_sequence_type (save: bool = True)

Checks the sequence type identifier detected by *dicom_parser* and updates the serialized value if required.

Parameters **save** (bool) – Whether to save changes or not, default is True

django_{dicom}.models.study module

Definition of the *Study* class.

class django_{dicom}.models.study.**Study** (*args, **kwargs)

Bases: *django_{dicom}.models.dicom_entity.DicomEntity*

A model to represent a single instance of the *Study* entity.

FIELD_TO_HEADER = {'date': 'StudyDate', 'description': 'StudyDescription', 'time':

A dictionary of DICOM data element keywords to be used to populate a created instance's fields.

date

Study Date value.

description

Study Description value.

get_absolute_url()

Returns the absolute URL for this instance. For more information see the [Django documentation](#).

Returns This instance's absolute URL path

Return type `str`

logger = `<Logger data.dicom.study (WARNING)>`

n_images

Returns the number of associated images.

See also:

`query_n_images()`

Returns Number of associated images

Return type `int`

n_patients

Returns the number of associated patients.

See also:

`query_n_patients()`

Returns Number of associated patients

Return type `int`

n_series

Returns the number of associated series.

Returns Number of associated series

Return type `int`

objects = `<django.db.models.manager.DicomEntityManagerFromStudyQuerySet object>`

query_n_images() → `int`

Returns the number of associated images.

See also:

`n_images()`

Returns Number of associated images

Return type `int`

query_n_patients() → `int`

Returns the number of associated patients.

See also:

`n_patients()`

Returns Number of associated patients

Return type `int`

series_set

Accessor to the related objects manager on the reverse side of a many-to-one relation.

In the example:

```
class Child(Model):
    parent = ForeignKey(Parent, related_name='children')
```

`Parent.children` is a `ReverseManyToOneDescriptor` instance.

Most of the implementation is delegated to a dynamically defined manager class built by `create_forward_many_to_many_manager()` defined below.

time

Study Time value.

uid

Study Instance UID value.

4.2.4 Serializers

Module contents

Provides `serializers` for the `REST API`.

Submodules

django_dicom.serializers.image_serializer module

Definition of the `ImageSerializer` class.

```
class django_dicom.serializers.image_serializer.ImageSerializer(instance=None,
                                                                data=<class
                                                                'rest_framework.fields.empty'>,
                                                                **kwargs)
```

Bases: `rest_framework.serializers.ModelSerializer`

A serializer class for the `Image` model.

class Meta

Bases: `object`

fields = ('id', 'series', 'number', 'date', 'time', 'uid')

model

alias of `django_dicom.models.image.Image`

django_dicom.serializers.patient_serializer module

Definition of the `PatientSerializer` class.

```
class django_dicom.serializers.patient_serializer.PatientSerializer(instance=None,
                                                                    data=<class
                                                                    'rest_framework.fields.empty'>,
                                                                    **kwargs)
```

Bases: `rest_framework.serializers.HyperlinkedModelSerializer`

A serializer class for the `Patient` model.

Submodules

django_dicom.utils.html module

Definition of the *Html* class.

django_dicom.utils.html.ADMIN_VIEW_NAMES = {'DataElement': 'admin:django_dicom_dataelement
Admin site views by model name, used to generate the appropriate URLs.

class django_dicom.utils.html.Html

Bases: *object*

Automates some HTML generation for the admin site.

BREAK = '
'

HORIZONTAL_LINE = '<hr style="background-color: {color};">'

classmethod admin_link(model_name: str, pk: int, text: str = None) → str

Returns a link to the admin site page of the provided model instance.

Parameters

- **model_name** (*str*) – Model to be linked
- **pk** (*int*) – Instance to be linked
- **text** (*str*, *optional*) – Text to display for the link, by default None (uses the instance's primary key)

Returns HTML link

Return type *str*

classmethod break_html(pieces) → str

Add break (
) tags between an iterable of HTML snippets.

Parameters **pieces** (*Iterable*) – HTML snippets

Returns Joined HTML

Return type *str*

classmethod horizontal_line(color: str = 'black') → str

Returns an <hr> tag with the desired color.

Parameters **color** (*str*, *optional*) – Color to style the line with, by default “black”

Returns HTML horizontal line

Return type *str*

classmethod json(value) → str

Formats a JSON string for display in the browser.

Parameters **value** (*str*) – JSON string

Returns Formatted JSON HTML

Return type *str*

4.2.6 Views

Module contents

Definition of the app's `rest_framework.viewsets.ViewSet` subclasses.

Submodules

django_{dicom}.views.defaults module

Definition of the *DefaultsMixin* mixin.

```
class djangodicom.views.defaults.DefaultsMixin
    Bases: object

    Default settings for view authentication, permissions and filtering.

    authentication_classes = (<class 'rest_framework.authentication.BasicAuthentication'>,
    filter_backends = (<class 'django_filters.rest_framework.backends.DjangoFilterBackend'>,
    permission_classes = (<class 'rest_framework.permissions.IsAuthenticated'>,)

```

django_{dicom}.views.image module

Definition of the *ImageViewSet* class.

```
class djangodicom.views.image.ImageViewSet (**kwargs)
    Bases: djangodicom.views.defaults.DefaultsMixin, rest_framework.viewsets.ModelViewSet

    API endpoint that allows images to be viewed or edited.

    filter_class
        alias of djangodicom.filters.image_filter.ImageFilter

    get_queryset()
        Get the list of items for this view. This must be an iterable, and may be a queryset. Defaults to using
        self.queryset.

        This method should always be used rather than accessing self.queryset directly, as self.queryset gets evaluated only once, and those results are cached for all subsequent requests.

        You may want to override this if you need to provide different querysets depending on the incoming request.

        (Eg. return a list of items that is specific to the user)

    ordering_fields = ('series', 'number', 'date', 'time')

    pagination_class
        alias of djangodicom.views.pagination.StandardResultsSetPagination

    parser_classes = (<class 'rest_framework.parsers.MultiPartParser'>,)

    queryset

    search_fields = ('number', 'date', 'time', 'uid')

    serializer_class
        alias of djangodicom.serializers.image_serializer.ImageSerializer

```

django_{dicom}.views.pagination module

Definition of the *StandardResultsSetPagination* class.

```
class djangodicom.views.pagination.StandardResultsSetPagination
```

Bases: *rest_framework.pagination.PageNumberPagination*

Default pagination parameters. This didn't work as part of the DefaultsMixin and therefore has to be defined separately in the 'pagination_class' configuration.

max_page_size = 10000

page_size = 25

page_size_query_param = 'page_size'

django_{dicom}.views.patient module

Definition of the *PatientViewSet* class.

```
class djangodicom.views.patient.PatientViewSet (**kwargs)
```

Bases: *django_{dicom}.views.defaults.DefaultsMixin*, *rest_framework.viewsets.ModelViewSet*

API endpoint that allows patients to be viewed or edited.

aggregate (*request*) → *rest_framework.response.Response*

Returns related model counts if count filtering is enabled.

Parameters **request** (*Request*) – API request

Returns Aggregated queryset or informational message

Return type *Response*

download_series_set (*request, uid*)

filter_class

alias of *django_{dicom}.filters.patient_filter.PatientFilter*

get_queryset () → *django.db.models.query.QuerySet*

Overrides the parent *get_queryset* () method to apply aggregated annotation if count filtering is enabled.

Returns *Patient* queryset

Return type *QuerySet*

ordering_fields = ('id', 'uid', 'family_name', 'given_name', 'sex', 'date_of_birth', '')

pagination_class

alias of *django_{dicom}.views.pagination.StandardResultsSetPagination*

queryset

serializer_class

alias of *django_{dicom}.serializers.patient_serializer.PatientSerializer*

django_{dicom}.views.series module

Definition of the *SeriesViewSet* class.

```
class django_dicom.views.series.SeriesViewSet (**kwargs)
    Bases: django_dicom.views.defaults.DefaultsMixin, rest_framework.viewsets.ModelViewSet

    API endpoint that allows series to be viewed or edited.

    filter_class
        alias of django_dicom.filters.series_filter.SeriesFilter

    get_csv (request: rest_framework.request.Request) → rest_framework.response.Response

    get_manufacturers (request)

    get_queryset ()
        Get the list of items for this view. This must be an iterable, and may be a queryset. Defaults to using self.queryset.

        This method should always be used rather than accessing self.queryset directly, as self.queryset gets evaluated only once, and those results are cached for all subsequent requests.

        You may want to override this if you need to provide different querysets depending on the incoming request.

        (Eg. return a list of items that is specific to the user)

    listed_zip (request: rest_framework.request.Request, series_ids: str) → django.http.response.FileResponse

    ordering_fields = ('study', 'patient', 'number', 'date', 'time', 'scanning_sequence',

    pagination_class
        alias of django_dicom.views.pagination.StandardResultsSetPagination

    queryset

    search_fields = ('study', 'patient', 'body_part_examined', 'number', 'description', 'd

    serializer_class
        alias of django_dicom.serializers.series_serializer.SeriesSerializer

    to_zip (request: rest_framework.request.Request, pk: int) → django.http.response.FileResponse
```

django_dicom.views.study module

Definition of the *StudyViewSet* class.

```
class django_dicom.views.study.StudyViewSet (**kwargs)
    Bases: django_dicom.views.defaults.DefaultsMixin, rest_framework.viewsets.ModelViewSet

    API endpoint that allows studies to be viewed or edited.

    aggregate (request) → rest_framework.response.Response
        Returns related model counts if count filtering is enabled.

        Parameters request (Request) – API request

        Returns Aggregated queryset or informational message

        Return type Response

    filter_class
        alias of django_dicom.filters.study_filter.StudyFilter
```

get_queryset () → django.db.models.query.QuerySet

Overrides the parent *get_queryset* () method to apply aggregated annotation if count filtering is enabled.

Returns Study queryset

Return type QuerySet

pagination_class

alias of *django_dicom.views.pagination.StandardResultsSetPagination*

queryset

serializer_class

alias of *django_dicom.serializers.study_serializer.StudySerializer*

4.3 Submodules

4.4 django_dicom.urls module

Definition of the app's URLs.

django_dicom.urls.**path** (route, view, kwargs=None, name=None, *, Pattern=<class 'django.urls.resolvers.RoutePattern'>)

5.1 General

- [Wikipedia](#).
- [The official DICOM standard website](#).
- [NiBabel's introduction to DICOM](#).
- [DICOM is Easy](#) - wonderful blog dedicated to the DICOM file format and software programming for medical applications.
- [DICOM Library](#) - free online medical images, signals or video files sharing and anonymizing service for educational and scientific purposes. Includes a good [introduction to the DICOM standard](#) as well as a [handy table of official DICOM tags](#).
- [DICOM Standard Browser](#) - an online DICOM encyclopedia.

5.2 Other

- *The first step for neuroimaging data analysis: DICOM to NIfTI conversion*¹ - contains valuable information about the way different vendors store data in the DICOM format ([PDF](#)).
- [Siemens mosaic format](#) - Excellent NiBabel article.
- Information on GE private tags can be found in the [Discovery VCT DICOM Conformance Statement](#) as well as in the [GE MR DICOM Conformance Statement](#).
- [NA-MIC Wiki page on DICOM for DWI and DTI](#) - vendor-specific information about DICOM and DWI/DTI data encoding.

¹ Li, X., Morgan, P. S., Ashburner, J., Smith, J., & Rorden, C. (2016). The first step for neuroimaging data analysis: DICOM to NIfTI conversion. *Journal of neuroscience methods*, 264, 47-56.

CHAPTER 6

Indices and tables

- `genindex`
- `modindex`
- `search`

Python Module Index

d

`django_dicom`, 11

`django_dicom.exceptions`, 11

`django_dicom.exceptions.dicom_import_error`, 11

`django_dicom.filters`, 12

`django_dicom.filters.image_filter`, 12

`django_dicom.filters.patient_filter`, 12

`django_dicom.filters.series_filter`, 13

`django_dicom.filters.study_filter`, 15

`django_dicom.models`, 16

`django_dicom.models.data_element`, 37

`django_dicom.models.data_element_definition`, 38

`django_dicom.models.dicom_entity`, 39

`django_dicom.models.header`, 41

`django_dicom.models.image`, 42

`django_dicom.models.managers`, 16

`django_dicom.models.managers.data_element`, 17

`django_dicom.models.managers.data_element_definition`, 18

`django_dicom.models.managers.data_element_value`, 18

`django_dicom.models.managers.dicom_entity`, 20

`django_dicom.models.managers.header`, 20

`django_dicom.models.managers.image`, 21

`django_dicom.models.managers.values`, 16

`django_dicom.models.managers.values.sequence_of_items`, 17

`django_dicom.models.patient`, 44

`django_dicom.models.series`, 47

`django_dicom.models.study`, 50

`django_dicom.models.utils`, 22

`django_dicom.models.utils.fields`, 22

`django_dicom.models.utils.help_text`, 22

`django_dicom.models.utils.logs`, 23

`django_dicom.models.utils.meta`, 23

`django_dicom.models.utils.utils`, 23

`django_dicom.models.utils.validators`, 23

`django_dicom.models.values`, 23

`django_dicom.models.values.age_string`, 24

`django_dicom.models.values.application_entity`, 24

`django_dicom.models.values.code_string`, 24

`django_dicom.models.values.csa_header`, 24

`django_dicom.models.values.data_element_value`, 25

`django_dicom.models.values.date`, 30

`django_dicom.models.values.datetime`, 30

`django_dicom.models.values.decimal_string`, 31

`django_dicom.models.values.floating_point_double`, 31

`django_dicom.models.values.floating_point_single`, 31

`django_dicom.models.values.integer_string`, 31

`django_dicom.models.values.long_string`, 32

`django_dicom.models.values.long_text`, 32

`django_dicom.models.values.other_word`, 32

`django_dicom.models.values.person_name`, 33

`django_dicom.models.values.sequence_of_items`, 33

`django_dicom.models.values.short_string`, 34

`django_dicom.models.values.short_text`, 34

`django_dicom.models.values.signed_long`, 34

django_dicom.models.values.signed_short,
34
django_dicom.models.values.time, 35
django_dicom.models.values.unique_identifier,
35
django_dicom.models.values.unknown, 35
django_dicom.models.values.unlimited_text,
36
django_dicom.models.values.unsigned_long,
36
django_dicom.models.values.unsigned_short,
36
django_dicom.models.values.vr_to_model,
37
django_dicom.serializers, 52
django_dicom.serializers.image_serializer,
52
django_dicom.serializers.patient_serializer,
52
django_dicom.serializers.series_serializer,
53
django_dicom.serializers.study_serializer,
53
django_dicom.urls, 58
django_dicom.utils, 53
django_dicom.utils.html, 54
django_dicom.views, 55
django_dicom.views.defaults, 55
django_dicom.views.image, 55
django_dicom.views.pagination, 56
django_dicom.views.patient, 56
django_dicom.views.series, 56
django_dicom.views.study, 57

A

abstract (*django_dicom.models.dicom_entity.DicomEntity.Meta attribute*), 13
attribute), 40
 admin_link (*django_dicom.models.data_element.DataElement attribute*), 14
attribute), 37
 admin_link (*django_dicom.models.data_element_definition.DataElementDefinition attribute*), 39
body_part_examined
 admin_link (*django_dicom.models.dicom_entity.DicomEntity (django_dicom.models.series.Series attribute)*), 40
attribute), 47
 admin_link (*django_dicom.models.header.Header attribute*), 41
break_html() (*django_dicom.utils.html.Html class method*), 54
 admin_link (*django_dicom.models.values.data_element_value.DataElementValue attribute*), 25
 admin_link() (*django_dicom.utils.html.Html class method*), 54
 ADMIN_VIEW_NAMES (in module *django_dicom.utils.html*), 54
 AgeString (class in *django_dicom.models.values.age_string*), 24
 agestring (*django_dicom.models.values.data_element_value.DataElementValue attribute*), 25
 aggregate() (*django_dicom.views.patient.PatientViewSet method*), 56
 aggregate() (*django_dicom.views.study.StudyViewSet method*), 57
 ApplicationEntity (class in *django_dicom.models.values.application_entity*), 24
 applicationentity (*django_dicom.models.values.data_element_value.DataElementValue attribute*), 25
 authentication_classes (*django_dicom.views.defaults.DefaultsMixin attribute*), 55

B

base_filters (*django_dicom.filters.image_filter.ImageFilter attribute*), 12

base_filters (*django_dicom.filters.patient_filter.PatientFilter attribute*), 13
 base_filters (*django_dicom.filters.series_filter.SeriesFilter attribute*), 14
 base_filters (*django_dicom.filters.study_filter.StudyFilter attribute*), 15
 body_part_examined
 BREAK (*django_dicom.utils.html.Html attribute*), 54
 break_html() (*django_dicom.utils.html.Html class method*), 54
 C
 check_element_inclusion() (in module *django_dicom.models.utils.utils*), 23
 ChoiceArrayField (class in *django_dicom.models.utils.fields*), 22
 CodeString (class in *django_dicom.models.values.code_string*), 24
 codestring (*django_dicom.models.values.data_element_value.DataElementValue attribute*), 25
 create_from_dcm() (*django_dicom.models.managers.image.ImageManager method*), 21
 create_from_dicom_parser() (*django_dicom.models.managers.data_element.DataElementManager method*), 17
 create_header_instance() (*django_dicom.models.image.Image method*), 43
 CsaHeader (class in *django_dicom.models.values.csa_header*), 24
 csaheader (*django_dicom.models.values.data_element_value.DataElementValue attribute*), 26
 D
 data (*django_dicom.models.image.Image attribute*), 43

data (django_dicom.models.series.Series attribute), 47	data_element_definition (django_dicom.models.data_element_definition.DataElementDefinition attribute), 43
data_element_set (django_dicom.models.data_element_definition.DataElementDefinition attribute), 39	data_element_set (django_dicom.models.image.Image attribute), 43
data_element_set (django_dicom.models.header.Header attribute), 41	data_element_set (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
data_element_set (django_dicom.models.values.data_element_value.DataElementValue attribute), 26	data_element_set (django_dicom.models.data_element.DataElement attribute), 37
data_element_to_definition() (in module description (django_dicom.models.data_element_definition.DataElementDefinition), attribute), 39	data_element_to_definition() (in module description (django_dicom.models.series.Series attribute), 18
DataElement (class in django_dicom.models.data_element), 37	DataElement (class in django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
DataElementDefinition (class in django_dicom.models.data_element_definition), 38	DataElementDefinition (class in django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
DataElementDefinitionManager (class in django_dicom.models.managers.data_element_definition), 18	DataElementDefinitionManager (class in django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
DataElementManager (class in django_dicom.models.managers.data_element), 17	DataElementManager (class in django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
DataElementValue (class in django_dicom.models.values.data_element_value), 25	DataElementValue (class in django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
DataElementValueManager (class in django_dicom.models.managers.data_element_value), 18	DataElementValueManager (class in django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
Date (class in django_dicom.models.values.date), 30	Date (class in django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
date (django_dicom.models.image.Image attribute), 43	date (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
date (django_dicom.models.series.Series attribute), 47	date (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
date (django_dicom.models.study.Study attribute), 50	date (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
date (django_dicom.models.values.data_element_value.DataElementValue attribute), 26	date (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
date_of_birth (django_dicom.models.patient.Patient attribute), 44	date_of_birth (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
DateTime (class in django_dicom.models.values.datetime), 30	DateTime (class in django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
datetime (django_dicom.models.series.Series attribute), 47	datetime (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
datetime (django_dicom.models.values.data_element_value.DataElementValue attribute), 26	datetime (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
dcm (django_dicom.models.image.Image attribute), 43	dcm (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
DecimalString (class in django_dicom.models.values.decimal_string), 31	DecimalString (class in django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
decimalstring (django_dicom.models.values.data_element_value.DataElementValue attribute), 26	decimalstring (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
declared_filters (django_dicom.filters.image_filter.ImageFilter attribute), 12	declared_filters (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
declared_filters (django_dicom.filters.patient_filter.PatientFilter attribute), 13	declared_filters (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
declared_filters (django_dicom.filters.series_filter.SeriesFilter attribute), 14	declared_filters (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55
declared_filters (django_dicom.filters.study_filter.StudyFilter attribute), 15	declared_filters (django_dicom.models.defaults.Mixin (class in django_dicom.views.defaults), 55

(module), 20
 django_dicom.models.managers.header
 (module), 20
 django_dicom.models.managers.image (mod-
 ule), 21
 django_dicom.models.managers.values
 (module), 16
 django_dicom.models.managers.values.sequence_of_items
 (module), 17
 django_dicom.models.patient (module), 44
 django_dicom.models.series (module), 47
 django_dicom.models.study (module), 50
 django_dicom.models.utils (module), 22
 django_dicom.models.utils.fields (mod-
 ule), 22
 django_dicom.models.utils.help_text
 (module), 22
 django_dicom.models.utils.logs (module),
 23
 django_dicom.models.utils.meta (module),
 23
 django_dicom.models.utils.utils (module),
 23
 django_dicom.models.utils.validators
 (module), 23
 django_dicom.models.values (module), 23
 django_dicom.models.values.age_string
 (module), 24
 django_dicom.models.values.application_entity
 (module), 24
 django_dicom.models.values.code_string
 (module), 24
 django_dicom.models.values.csa_header
 (module), 24
 django_dicom.models.values.data_element_value
 (module), 25
 django_dicom.models.values.date (module),
 30
 django_dicom.models.values.datetime
 (module), 30
 django_dicom.models.values.decimal_string
 (module), 31
 django_dicom.models.values.floating_point
 (module), 31
 django_dicom.models.values.floating_point
 (module), 31
 django_dicom.models.values.integer_string
 (module), 31
 django_dicom.models.values.long_string
 (module), 32
 django_dicom.models.values.long_text
 (module), 32
 django_dicom.models.values.other_word
 (module), 32
 django_dicom.models.values.person_name
 (module), 33
 django_dicom.models.values.sequence_of_items
 (module), 33
 django_dicom.models.values.short_string
 (module), 34
 django_dicom.models.values.short_text
 (module), 34
 django_dicom.models.values.signed_long
 (module), 34
 django_dicom.models.values.signed_short
 (module), 34
 django_dicom.models.values.time (module),
 35
 django_dicom.models.values.unique_identifier
 (module), 35
 django_dicom.models.values.unknown (mod-
 ule), 35
 django_dicom.models.values.unlimited_text
 (module), 36
 django_dicom.models.values.unsigned_long
 (module), 36
 django_dicom.models.values.unsigned_short
 (module), 36
 django_dicom.models.values.vr_to_model
 (module), 37
 django_dicom.serializers (module), 52
 django_dicom.serializers.image_serializer
 (module), 52
 django_dicom.serializers.patient_serializer
 (module), 52
 django_dicom.serializers.series_serializer
 (module), 53
 django_dicom.serializers.study_serializer
 (module), 53
 django_dicom.urls (module), 58
 django_dicom.utils (module), 53
 django_dicom.utils.html (module), 54
 django_dicom.views (module), 55
 django_dicom.views.defaults (module), 55
 django_dicom.views.image (module), 55
 django_dicom.views.pagination (module), 56
 django_dicom.views.patient (module), 56
 django_dicom.views.series (module), 56
 django_dicom.views.study (module), 57
 download_series_set ()
 (django_dicom.views.patient.PatientViewSet
 method), 56
E
 echo_time (django_dicom.models.series.Series attribute), 47
 echo_train_length
 (django_dicom.models.series.Series attribute),

get_entity_uid() (*django_dicom.models.header.Header* method), 41
 get_file_paths() (*django_dicom.models.series.Series* method), 47
 get_full_name() (*django_dicom.models.patient.Patient* method), 44
 get_group_model() (in module *django_dicom.models.utils.utils*), 23
 get_header_fields() (*django_dicom.models.dicom_entity.DicomEntity* method), 40
 get_header_keyword() (*django_dicom.models.dicom_entity.DicomEntity* class method), 40
 get_import_configuration() (in module *django_dicom.models.utils.utils*), 23
 get_import_mode() (in module *django_dicom.models.utils.utils*), 23
 get_manufacturers() (*django_dicom.views.series.SeriesViewSet* method), 57
 get_modality_display() (*django_dicom.models.series.Series* method), 48
 get_model() (in module *django_dicom.models.utils.meta*), 23
 get_mr_acquisition_type_display() (*django_dicom.models.series.Series* method), 48
 get_mri_root() (in module *django_dicom.models.utils.utils*), 23
 get_next_by_value() (*django_dicom.models.values.datetime.DateTime* method), 30
 get_or_create() (*django_dicom.models.managers.image.ImageManager* method), 21
 get_or_create_entity() (*django_dicom.models.header.Header* method), 41
 get_or_create_from_dcm() (*django_dicom.models.managers.image.ImageManager* method), 21
 get_or_create_from_nonsequence() (*django_dicom.models.managers.data_element_value.DataElementValueManager* method), 18
 get_or_create_patient() (*django_dicom.models.header.Header* method), 41
 get_or_create_series() (*django_dicom.models.header.Header* method), 41
 get_or_create_study() (*django_dicom.models.header.Header* method), 41
 get_path() (*django_dicom.models.series.Series* method), 48
 get_patient_position_display() (*django_dicom.models.series.Series* method), 48
 get_previous_by_value() (*django_dicom.models.values.datetime.DateTime* method), 30
 get_queryset() (*django_dicom.views.image.ImageViewSet* method), 55
 get_queryset() (*django_dicom.views.patient.PatientViewSet* method), 56
 get_queryset() (*django_dicom.views.series.SeriesViewSet* method), 57
 get_queryset() (*django_dicom.views.study.StudyViewSet* method), 57
 get_raw_peek() (*django_dicom.models.values.data_element_value.DataElementValue* method), 27
 get_sample_header() (*django_dicom.models.series.Series* method), 48
 get_scanning_sequence_display() (*django_dicom.models.series.Series* method), 48
 get_sequence_type_display() (*django_dicom.models.series.Series* method), 48
 get_sequence_variant_display() (*django_dicom.models.series.Series* method), 48
 get_sex_display() (*django_dicom.models.patient.Patient* method), 45
 get_subject_model() (in module *django_dicom.models.utils.utils*), 23
 get_value_by_keyword() (*django_dicom.models.header.Header* method), 42
 get_value_model() (in module *django_dicom.models.values.vr_to_model*), 37
 get_value_model_name() (in module *django_dicom.models.values.vr_to_model*), 37
 get_value_representation_display() (*django_dicom.models.data_element_definition.DataElementDefinitionManager* method), 39
 given_name (*django_dicom.models.patient.Patient* attribute), 45

H

handle_invalid_data() (*django_dicom.models.managers.data_element_value.DataElementValueManager* method), 19
 handle_multiple_values() (*django_dicom.models.managers.data_element_value.DataElementValueManager* method), 19

method), 19
 handle_no_value() (django_{dicom}.models.managers.data_element_value.DataElementValueManager method), 19
 handle_single_value() (django_{dicom}.models.managers.data_element_value.DataElementValueManager method), 19
 handle_value_multiplicity() (django_{dicom}.models.managers.data_element_value.DataElementValueManager method), 20
 Header (class in django_{dicom}.models.header), 41
 header (django_{dicom}.models.data_element.DataElement integerstring (django_{dicom}.models.values.data_element_value.DataElementValueManager attribute), 38
 header (django_{dicom}.models.image.Image attribute), 43
 header_set (django_{dicom}.models.values.sequence_of_items.SequenceOfItems attribute), 33
 HeaderManager (class in django_{dicom}.models.managers.header), 20
 HORIZONTAL_LINE (django_{dicom}.utils.html.Html attribute), 54
 horizontal_line() (django_{dicom}.utils.html.Html class method), 54
 Html (class in django_{dicom}.utils.html), 54
I
 Image (class in django_{dicom}.models.image), 42
 image (django_{dicom}.models.header.Header attribute), 42
 image_set (django_{dicom}.models.series.Series attribute), 48
 ImageFilter (class in django_{dicom}.filters.image_filter), 12
 ImageFilter.Meta (class in django_{dicom}.filters.image_filter), 12
 ImageManager (class in django_{dicom}.models.managers.image), 21
 ImageSerializer (class in django_{dicom}.serializers.image_serializer), 52
 ImageSerializer.Meta (class in django_{dicom}.serializers.image_serializer), 52
 ImageViewSet (class in django_{dicom}.views.image), 55
 import_path() (django_{dicom}.models.managers.image.ImageManager class method), 21
 ImportMode (class in django_{dicom}.models.utils.utils), 23
 index (django_{dicom}.models.header.Header attribute), 42
 index (django_{dicom}.models.values.data_element_value.DataElementValue attribute), 27
 instance (django_{dicom}.models.header.Header attribute), 42
 instance (django_{dicom}.models.image.Image attribute), 43
 instance (django_{dicom}.models.series.Series attribute), 48
 institution_name (django_{dicom}.models.series.Series attribute), 48
 integerstring (django_{dicom}.models.values.data_element_value.DataElementValueManager class in django_{dicom}.models.values.integer_string), 31
 integerstring (django_{dicom}.models.values.data_element_value.DataElementValueManager attribute), 27
 inversion_time (django_{dicom}.models.series.Series attribute), 48
 items_in_sequence_of_items() (django_{dicom}.models.dicom_entity.DicomEntity method), 40
J
 json() (django_{dicom}.utils.html.Html class method), 54
K
 keyword (django_{dicom}.models.data_element_definition.DataElementDefinition attribute), 39
L
 listed_zip() (django_{dicom}.views.series.SeriesViewSet method), 57
 log_creation() (django_{dicom}.models.dicom_entity.DicomEntity method), 40
 logger (django_{dicom}.models.image.Image attribute), 43
 logger (django_{dicom}.models.patient.Patient attribute), 45
 logger (django_{dicom}.models.series.Series attribute), 49
 logger (django_{dicom}.models.study.Study attribute), 51
 LongString (class in django_{dicom}.models.values.long_string), 32
 longstring (django_{dicom}.models.values.data_element_value.DataElementValue attribute), 27
 longtext (class in django_{dicom}.models.values.long_text), 32
 longtext (django_{dicom}.models.values.data_element_value.DataElementValue attribute), 27
M
 magnetic_field_strength (django_{dicom}.models.series.Series attribute), 49

manufacturer (*django_dicom.models.series.Series attribute*), 49
 manufacturer_model_name (*django_dicom.models.series.Series attribute*), 49
 max_page_size (*django_dicom.views.pagination.StandardResultsSetPagination type attribute*), 56
 MAX_VALUE (in module *django_dicom.models.values.integer_string*), 32
 MAX_VALUE (in module *django_dicom.models.values.signed_long*), 34
 MAX_VALUE (in module *django_dicom.models.values.signed_short*), 34
 MAX_VALUE (in module *django_dicom.models.values.unsigned_long*), 36
 MAX_VALUE (in module *django_dicom.models.values.unsigned_short*), 36
 middle_name (*django_dicom.models.patient.Patient attribute*), 45
 MIN_VALUE (in module *django_dicom.models.values.integer_string*), 32
 MIN_VALUE (in module *django_dicom.models.values.signed_long*), 34
 MIN_VALUE (in module *django_dicom.models.values.signed_short*), 34
 MINIMAL (*django_dicom.models.utils.utils.ImportMode attribute*), 23
 missing_relation (*django_dicom.models.series.Series attribute*), 49
 modality (*django_dicom.models.series.Series attribute*), 49
 model (*django_dicom.filters.image_filter.ImageFilter.Meta attribute*), 12
 model (*django_dicom.filters.patient_filter.PatientFilter.Meta attribute*), 13
 model (*django_dicom.filters.series_filter.SeriesFilter.Meta attribute*), 14
 model (*django_dicom.filters.study_filter.StudyFilter.Meta attribute*), 15
 model (*django_dicom.serializers.image_serializer.ImageSerializer.Meta attribute*), 52
 model (*django_dicom.serializers.patient_serializer.PatientSerializer.Meta attribute*), 53
 model (*django_dicom.serializers.series_serializer.SeriesSerializer.Meta attribute*), 53
 model (*django_dicom.serializers.study_serializer.StudySerializer.Meta attribute*), 53
 MR_ACQUISITION_2D (*django_dicom.models.series.Series attribute*), 47
 MR_ACQUISITION_3D (*django_dicom.models.series.Series attribute*), 47
 MR_ACQUISITION_TYPE_CHOICES (*django_dicom.models.series.Series attribute*), 47
 N
 n_images (*django_dicom.models.patient.Patient attribute*), 45
 n_images (*django_dicom.models.study.Study attribute*), 51
 n_patients (*django_dicom.models.study.Study attribute*), 51
 n_series (*django_dicom.models.patient.Patient attribute*), 45
 n_series (*django_dicom.models.study.Study attribute*), 51
 n_studies (*django_dicom.models.patient.Patient attribute*), 45
 name_prefix (*django_dicom.models.patient.Patient attribute*), 45
 name_suffix (*django_dicom.models.patient.Patient attribute*), 45
 NORMAL (*django_dicom.models.utils.utils.ImportMode attribute*), 23
 number (*django_dicom.models.image.Image attribute*), 43
 number (*django_dicom.models.series.Series attribute*), 49
 O
 objects (*django_dicom.models.data_element.DataElement attribute*), 38
 objects (*django_dicom.models.data_element_definition.DataElementDefinition attribute*), 39
 objects (*django_dicom.models.dicom_entity.DicomEntity attribute*), 40
 objects (*django_dicom.models.header.Header attribute*), 42
 objects (*django_dicom.models.image.Image attribute*), 43
 objects (*django_dicom.models.patient.Patient attribute*), 45
 objects (*django_dicom.models.study.Study attribute*), 51
 objects (*django_dicom.models.values.data_element_value.DataElementValue attribute*), 27

objects (*django_dicom.models.values.sequence_of_items.SequenceOfItems* (class in *attribute*), 33
django_dicom.views.patient), 56
operators_name (*django_dicom.models.series.Series* permission_classes (*django_dicom.views.defaults.DefaultsMixin* *attribute*), 49
ordering_fields (*django_dicom.views.image.ImageViewSet* *attribute*), 55
attribute), 55
PersonName (class in *attribute*), 55
ordering_fields (*django_dicom.views.patient.PatientViewSet* *django_dicom.models.values.person_name*), *attribute*), 56
33
ordering_fields (*django_dicom.views.series.SeriesViewSet* *person_name* (*django_dicom.models.values.data_element_value.DataElementValue* *attribute*), 57
attribute), 28
OtherWord (class in *pixel_spacing* (*django_dicom.models.series.Series* *attribute*), 49
django_dicom.models.values.other_word), *attribute*), 32
49
protocol_name (*django_dicom.models.series.Series* *attribute*), 49
otherword (*django_dicom.models.values.data_element_value.DataElementValue* *attribute*), 28
pulse_sequence_name (*django_dicom.models.series.Series* *attribute*), 49
P
page_size (*django_dicom.views.pagination.StandardResultsSetPagination* *attribute*), 56
Q
page_size_query_param *query_n_images()* (*django_dicom.models.patient.Patient* *method*), 45
(*django_dicom.views.pagination.StandardResultsSetPagination* *attribute*), 56
query_n_images() (*django_dicom.models.study.Study* *method*), 51
pagination_class (*django_dicom.views.image.ImageViewSet* *method*), 51
attribute), 55
query_n_patients()
pagination_class (*django_dicom.views.patient.PatientViewSet* (*django_dicom.models.study.Study* *method*), 51
attribute), 56
query_n_studies()
pagination_class (*django_dicom.views.series.SeriesViewSet* (*django_dicom.models.patient.Patient* *method*), 46
attribute), 57
query_research_subject()
pagination_class (*django_dicom.views.study.StudyViewSet* *attribute*), 58
(*django_dicom.models.patient.Patient* *method*), 46
parent (*django_dicom.models.header.Header* *attribute*), 42
queryset (*django_dicom.views.image.ImageViewSet* *attribute*), 55
parser_classes (*django_dicom.views.image.ImageViewSet* *attribute*), 55
queryset (*django_dicom.views.patient.PatientViewSet* *attribute*), 56
path (*django_dicom.models.series.Series* *attribute*), 49
attribute), 56
path() (in module *django_dicom.urls*), 58
queryset (*django_dicom.views.series.SeriesViewSet* *attribute*), 57
Patient (class in *django_dicom.models.patient*), 44
attribute), 57
patient (*django_dicom.models.image.Image* *attribute*), 43
queryset (*django_dicom.views.study.StudyViewSet* *attribute*), 58
patient (*django_dicom.models.series.Series* *attribute*), 49
R
patient_position (*django_dicom.models.series.Series* *raw* (*django_dicom.models.values.age_string.AgeString* *attribute*), 24
attribute), 49
PatientFilter (class in *raw* (*django_dicom.models.values.application_entity.ApplicationEntity* *attribute*), 24
django_dicom.filters.patient_filter), 12
attribute), 24
PatientFilter.Meta (class in *raw* (*django_dicom.models.values.code_string.CodeString* *attribute*), 24
django_dicom.filters.patient_filter), 13
attribute), 24
PatientSerializer (class in *raw* (*django_dicom.models.values.csa_header.CsaHeader* *attribute*), 25
django_dicom.serializers.patient_serializer), *attribute*), 52
raw (*django_dicom.models.values.data_element_value.DataElementValue* *attribute*), 28
PatientSerializer.Meta (class in *raw* (*django_dicom.models.values.date.Date* *attribute*), 30
django_dicom.serializers.patient_serializer), *attribute*), 52
30

`django_dicom.models.values.short_string`), 34
`shortstring` (`django_dicom.models.values.data_element_value.DataElementValue` attribute), 28
`ShortText` (class in `django_dicom.models.values.short_text`), 34
`shorttext` (`django_dicom.models.values.data_element_value.DataElementValue` attribute), 28
`SignedLong` (class in `django_dicom.models.values.signed_long`), 34
`signedlong` (`django_dicom.models.values.data_element_value.DataElementValue` attribute), 28
`SignedShort` (class in `django_dicom.models.values.signed_short`), 35
`signedshort` (`django_dicom.models.values.data_element_value.DataElementValue` attribute), 29
`slice_thickness` (`django_dicom.models.series.Series` attribute), 50
`snake_case_to_camel_case` () (in module `django_dicom.models.utils.utils`), 23
`spatial_resolution` (`django_dicom.models.series.Series` attribute), 50
`StandardResultsSetPagination` (class in `django_dicom.views.pagination`), 56
`store_image_data` () (`django_dicom.models.managers.image.ImageManager` method), 22
`Study` (class in `django_dicom.models.study`), 50
`study` (`django_dicom.models.series.Series` attribute), 50
`StudyFilter` (class in `django_dicom.filters.study_filter`), 15
`StudyFilter.Meta` (class in `django_dicom.filters.study_filter`), 15
`StudySerializer` (class in `django_dicom.serializers.study_serializer`), 53
`StudySerializer.Meta` (class in `django_dicom.serializers.study_serializer`), 53
`StudyViewSet` (class in `django_dicom.views.study`), 57

T
`tag` (`django_dicom.models.data_element_definition.DataElementDefinition` attribute), 39
`TAG_TO_MODEL` (in module `django_dicom.models.values.vr_to_model`), 37
`TEMP_DCM_FILE_NAME` (`django_dicom.models.managers.image.ImageManager` attribute), 21
`Time` (class in `django_dicom.models.values.time`), 35
`time` (`django_dicom.models.image.Image` attribute), 44
`Time` (`django_dicom.models.series.Series` attribute), 50
`time` (`django_dicom.models.study.Study` attribute), 52
`time` (`django_dicom.models.values.data_element_value.DataElementValue` attribute), 29
`to_dict` () (`django_dicom.models.data_element_definition.DataElementDefinition` method), 38
`to_html` () (`django_dicom.models.data_element.DataElement` method), 38
`to_html` () (`django_dicom.models.header.Header` method), 42
`to_html` () (`django_dicom.models.values.csa_header.CsaHeader` method), 25
`to_html` () (`django_dicom.models.values.data_element_value.DataElementValue` method), 29
`to_html` () (`django_dicom.models.values.person_name.PersonName` method), 33
`to_html` () (`django_dicom.models.values.sequence_of_items.SequenceOfItems` method), 33
`to_html` () (`django_dicom.models.values.time.Time` method), 35
`to_series` () (`django_dicom.models.data_element_definition.DataElementDefinition` method), 39
`to_verbose_dict` () (`django_dicom.models.data_element.DataElement` method), 38
`to_verbose_list` () (`django_dicom.models.header.Header` method), 42
`to_verbose_series` () (`django_dicom.models.data_element.DataElement` method), 38
`to_zip` () (`django_dicom.views.series.SeriesViewSet` method), 57

U
`uid` (`django_dicom.models.image.Image` attribute), 44
`uid` (`django_dicom.models.patient.Patient` attribute), 46
`uid` (`django_dicom.models.series.Series` attribute), 50
`uid` (`django_dicom.models.study.Study` attribute), 52
`UniqueIdentifier` (class in `django_dicom.models.values.unique_identifier`), 35
`uniqueidentifier` (`django_dicom.models.values.data_element_value.DataElementValue` attribute), 29
`Unknown` (class in `django_dicom.models.values.unknown`), 39
`unknown` (`django_dicom.models.values.data_element_value.DataElementValue` attribute), 29
`UnlimitedText` (class in `django_dicom.models.values.unlimited_text`), 36

[unlimitedtext \(django_dicom.models.values.data_element_value.DicomEntityValue.values.long_text.LongText attribute\), 29](#)
[UnsignedLong \(class in value \(django_dicom.models.values.other_word.OtherWord attribute\), 32](#)
[36 value \(django_dicom.models.values.person_name.PersonName attribute\), 33](#)
[unsignedlong \(django_dicom.models.values.data_element_value.DicomEntityValue value \(django_dicom.models.values.short_string.ShortString attribute\), 34](#)
[UnsignedShort \(class in attribute\), 34](#)
[36 value \(django_dicom.models.values.short_text.ShortText attribute\), 34](#)
[unsignedshort \(django_dicom.models.values.data_element_value.DicomEntityValue.values.signed_long.SignedLong attribute\), 30](#)
[attribute\), 34](#)
[update_fields_from_header\(\) value \(django_dicom.models.values.signed_short.SignedShort attribute\), 35](#)
[\(django_dicom.models.dicom_entity.DicomEntity method\), 40](#)
[value \(django_dicom.models.values.time.Time attribute\), 35](#)
[update_fields_from_header\(\) value \(django_dicom.models.values.unique_identifier.UniqueIdentifier attribute\), 35](#)
[\(django_dicom.models.patient.Patient method\), 46](#)
[update_patient_name\(\) value \(django_dicom.models.values.unknown.Unknown attribute\), 36](#)
[\(django_dicom.models.patient.Patient method\), 46](#)
[value \(django_dicom.models.values.unlimited_text.UnlimitedText attribute\), 36](#)
[update_sequence_type\(\) value \(django_dicom.models.values.unsigned_long.UnsignedLong attribute\), 36](#)
[\(django_dicom.models.series.Series method\), 50](#)
[value \(django_dicom.models.values.unsigned_short.UnsignedShort attribute\), 36](#)

V

[validate_file_extension\(\) \(in module value_multiplicity](#)
[django_dicom.models.utils.validators\), 23](#)
[value \(django_dicom.models.data_element.DataElement attribute\), 38](#)
[value_representation](#)
[value \(django_dicom.models.values.age_string.AgeString \(django_dicom.models.data_element_definition.DataElementDefi](#)
[attribute\), 24](#)
[attribute\), 39](#)
[value \(django_dicom.models.values.application_entity.ApplicationEntity \(in module](#)
[attribute\), 24](#)
[django_dicom.models.values.vr_to_model\), 37](#)

W

[value \(django_dicom.models.values.code_string.CodeString attribute\), 24](#)
[warnings \(django_dicom.models.image.Image attribute\), 44](#)
[value \(django_dicom.models.values.csa_header.CsaHeader attribute\), 25](#)
[value \(django_dicom.models.values.data_element_value.DicomEntityValue value \(django_dicom.models.values.data_element_value.DataElement](#)
[attribute\), 30](#)
[attribute\), 30](#)
[value \(django_dicom.models.values.date.Date attribute\), 30](#)
[value \(django_dicom.models.values.datetime.DateTime attribute\), 30](#)
[value \(django_dicom.models.values.decimal_string.DecimalString attribute\), 31](#)
[value \(django_dicom.models.values.floating_point_double.FloatingPointDouble attribute\), 31](#)
[value \(django_dicom.models.values.floating_point_single.FloatingPointSingle attribute\), 31](#)
[value \(django_dicom.models.values.integer_string.IntegerString attribute\), 31](#)
[value \(django_dicom.models.values.long_string.LongString attribute\), 32](#)