

Python Documentation

version

April 06, 2020

Contents

Welcome to ResourceSharing's documentation!	1
All about us	1
Models	1
Views	3
Forms	3
Indices and tables	5
Index	7
Python Module Index	9

Welcome to ResourceSharing's documentation!

All about us

We are Django developers from BITS Pilani Hyderabad Campus

Models

```
class blog.models.Task (id, author, content, date_created, resource_type)
```

exception DoesNotExist

exception MultipleObjectsReturned

author

Accessor to the related object on the forward side of a many-to-one or one-to-one (via ForwardOneToOneDescriptor subclass) relation.

In the example:

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

Child.parent is a ForwardManyToOneDescriptor instance.

author_id

content

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

date_created

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

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

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

get_resource_type_display (*, field=<django.db.models.fields.CharField: resource_type>)

id

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

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

resource_type

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

```
class users.models.CustomUser (id, password, last_login, is_superuser, username, first_name, last_name, is_staff, is_active, date_joined, mobile_number, address, email)
```

exception DoesNotExist

exception MultipleObjectsReturned

address

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

emailaddress_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_next_by_date_joined (*, field=<django.db.models.fields.DateTimeField: date_joined>, is_next=True, **kwargs)

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

groups

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.

id

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

logentry_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.

mobile_number

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

task_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.

user_permissions

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.

```
class users.models.userdata (id, name, email, content)
```

exception DoesNotExist

exception MultipleObjectsReturned

content

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

email

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

id

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

name

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

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

Views

```
blog.views.home (request)
```

```
blog.views.remove (request, item_id)
```

```
blog.views.task_list (request)
```

```
class pages.views.AboutPageView (**kwargs)
```

```
    template_name = 'pages/about.html'
```

```
class pages.views.HomePageView (**kwargs)
```

```
    template_name = 'pages/home.html'
```

Forms

```
class users.forms.CustomUserChangeForm (*args, **kwargs)
```

```
class Meta
```

```
    fields = ('email', 'username', 'mobile_number', 'address')
```

```
    model
```

alias of `users.models.CustomUser`

```
base_fields = {'address': <django.forms.fields.CharField object>, 'captcha': <captcha.fields.ReCaptchaField
object>, 'email': <django.forms.fields.EmailField object>, 'mobile_number': <django.forms.fields.CharField object>,
'password': <django.contrib.auth.forms.ReadOnlyPasswordHashField object>, 'username':
<django.forms.fields.CharField object>}
```

```
declared_fields = {'captcha': <captcha.fields.ReCaptchaField object>, 'password':
<django.contrib.auth.forms.ReadOnlyPasswordHashField object>}
```

property media

```
class users.forms.CustomUserCreationForm(*args, **kwargs)
```

```
class Meta
```

```
fields = ('email', 'username', 'mobile_number', 'address')
```

```
model
```

alias of `users.models.CustomUser`

```
base_fields = {'address': <django.forms.fields.CharField object>, 'captcha': <captcha.fields.ReCaptchaField
object>, 'email': <django.forms.fields.EmailField object>, 'mobile_number': <django.forms.fields.CharField object>,
'password1': <django.forms.fields.CharField object>, 'password2': <django.forms.fields.CharField object>,
'username': <django.contrib.auth.forms.UsernameField object>}
```

```
declared_fields = {'captcha': <captcha.fields.ReCaptchaField object>, 'password1':
<django.forms.fields.CharField object>, 'password2': <django.forms.fields.CharField object>}
```

property media

```
class users.forms.PostForm(data=None, files=None, auto_id='id_%s', prefix=None, initial=None,
error_class=<class 'django.forms.utils.ErrorList'>, label_suffix=None,
empty_permitted=False, instance=None, use_required_attribute=None, renderer=None)
```

```
class Meta
```

```
fields = ('name', 'email', 'content')
```

```
model
```

alias of `users.models.userdata`

```
base_fields = {'content': <django.forms.fields.CharField object>, 'email': <django.forms.fields.EmailField
object>, 'name': <django.forms.fields.CharField object>}
```

```
declared_fields = {}
```

property media

```
class blog.forms.PostForm(data=None, files=None, auto_id='id_%s', prefix=None, initial=None,
error_class=<class 'django.forms.utils.ErrorList'>, label_suffix=None,
empty_permitted=False, instance=None, use_required_attribute=None, renderer=None)
```

```
class Meta
```

```
fields = ('content', 'resource_type')
```

```
model
```

alias of `blog.models.Task`


```
base_fields = {'captcha': <captcha.fields.ReCaptchaField object>, 'content': <django.forms.fields.CharField object>, 'resource_type': <django.forms.fields.TypedChoiceField object>}
```

```
declared_fields = {'captcha': <captcha.fields.ReCaptchaField object>}
```

property media

Indices and tables

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

Index

A

AboutPageView (class in pages.views)
address (users.models.CustomUser attribute)
author (blog.models.Task attribute)
author_id (blog.models.Task attribute)

B

base_fields (blog.forms.PostForm attribute)
(users.forms.CustomUserChangeForm attribute)
(users.forms.CustomUserCreationForm attribute)
(users.forms.PostForm attribute)

blog.forms

module

blog.models

module

blog.views

module

C

content (blog.models.Task attribute)
(users.models.userdata attribute)
CustomUser (class in users.models)
CustomUser.DoesNotExist
CustomUser.MultipleObjectsReturned
CustomUserChangeForm (class in users.forms)
CustomUserChangeForm.Meta (class in users.forms)
CustomUserCreationForm (class in users.forms)
CustomUserCreationForm.Meta (class in users.forms)

D

date_created (blog.models.Task attribute)
declared_fields (blog.forms.PostForm attribute)
(users.forms.CustomUserChangeForm attribute)
(users.forms.CustomUserCreationForm attribute)
(users.forms.PostForm attribute)

E

email (users.models.userdata attribute)
emailaddress_set (users.models.CustomUser attribute)

F

fields (blog.forms.PostForm.Meta attribute)

(users.forms.CustomUserChangeForm.Meta attribute)

(users.forms.CustomUserCreationForm.Meta attribute)

(users.forms.PostForm.Meta attribute)

G

get_next_by_date_created() (blog.models.Task method)
get_next_by_date_joined() (users.models.CustomUser method)
get_previous_by_date_created() (blog.models.Task method)
get_previous_by_date_joined() (users.models.CustomUser method)
get_resource_type_display() (blog.models.Task method)
groups (users.models.CustomUser attribute)

H

home() (in module blog.views)
HomePageView (class in pages.views)

I

id (blog.models.Task attribute)
(users.models.CustomUser attribute)
(users.models.userdata attribute)

L

logentry_set (users.models.CustomUser attribute)

M

media() (blog.forms.PostForm property)
(users.forms.CustomUserChangeForm property)
(users.forms.CustomUserCreationForm property)
(users.forms.PostForm property)
mobile_number (users.models.CustomUser attribute)
model (blog.forms.PostForm.Meta attribute)
(users.forms.CustomUserChangeForm.Meta attribute)
(users.forms.CustomUserCreationForm.Meta attribute)
(users.forms.PostForm.Meta attribute)

module

blog.forms
blog.models
blog.views

pages.views
users.forms
users.models
users.views

N

name (users.models.userdata attribute)

O

objects (blog.models.Task attribute)
(users.models.userdata attribute)

P

pages.views

module

PostForm (class in blog.forms)

(class in users.forms)

PostForm.Meta (class in blog.forms)

(class in users.forms)

R

remove() (in module blog.views)

resource_type (blog.models.Task attribute)

T

Task (class in blog.models)

Task.DoesNotExist

Task.MultipleObjectsReturned

task_list() (in module blog.views)

task_set (users.models.CustomUser attribute)

template_name (pages.views>AboutPageView attribute)
(pages.views.HomePageView attribute)

U

user_permissions (users.models.CustomUser attribute)

userdata (class in users.models)

userdata.DoesNotExist

userdata.MultipleObjectsReturned

users.forms

module

users.models

module

users.views

module

Python Module Index

b

blog

[blog.forms](#)

[blog.models](#)

[blog.views](#)

p

pages

[pages.views](#)

u

users

[users.forms](#)

[users.models](#)

[users.views](#)