Resource Sharing

Release Beta 1.0

Radhesh Sarma, Simran Sahni, Nikhil Krishna, Abhirath Singh

Apr 10, 2020

CONTENTS:

CHAPTER

ONE

ALL ABOUT US

We are Django developers from BITS Pilani Hyderabad Campus

CHAPTER

TWO

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

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

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>

CHAPTER

THREE

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'

CHAPTER

FOUR

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

declared_fields = {'captcha': <captcha.fields.ReCaptchaField object>, 'password': <d

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

declared_fields = {'captcha': <captcha.fields.ReCaptchaField object>, 'password1': <

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.f

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

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

property media

PYTHON MODULE INDEX

INDEX