
django-dynamic-forms

Release 0.4.0

April 09, 2015

1	Installation	3
2	Usage	5
2.1	Using Custom Templates	5
2.2	Third Party Apps	6
3	Problems	7
3.1	My form is not shown	7
4	Changes	9
4.1	v0.5 (under development)	9
4.2	v0.4	9
4.3	v0.3.4	9
4.4	v0.3.3	9
4.5	v0.3.2	9
4.6	v0.2	10
4.7	v0.1	10
5	References	11
5.1	Actions	11
5.2	Admin	13
5.3	Fields	13
5.4	Form fields	13
5.5	Forms	17
5.6	Middlewares	18
5.7	Models	18
5.8	Settings	21
5.9	Views	22
5.10	Contrib packages	23
6	Indices and tables	25
	Python Module Index	27

Contents:

Warning: django-dynamic-forms 0.4.x will be the latest version branch that supports Django < 1.7!

Installation

Warning: `django-dynamic-forms` 0.4.x will be the latest version branch that supports Django < 1.7!

Install **django-dynamic-forms** into your virtual environment or your site-packages using pip:

```
$ pip install django-dynamic-forms
```

If you already use the wheel package format you can use the wheel build:

```
$ pip install --use-wheel django-dynamic-forms
```

To make **django-dynamic-forms** available in your Django project, you first have to add it to the `INSTALLED_APPS` in your `settings.py`. If you are unsure where to put it, just append it:

```
INSTALLED_APPS = (  
    ...  
    'dynamic_forms',  
    ...  
)
```

To make Django aware of the dynamic forms while processing the requests / responses you need to add the `FormModelMiddleware` to the list of `MIDDLEWARE_CLASSES`. The best place is probably at the end of the list. If your forms are not shown please refer to the *known problems* section of the documentation:

```
MIDDLEWARE_CLASSES = (  
    ...  
    'dynamic_forms.middlewares.FormModelMiddleware'  
)
```

Last but not least you need to add the `'dynamic_forms.urls'` urlpatterns to your project's URL patterns:

```
urlpatterns = patterns('',  
    ...  
    url(r'^dynamic_forms/',  
        include('dynamic_forms.urls', namespace='dynamic_forms')),  
    ...  
)
```

Important: Make sure that you get the namespace straight: `dynamic_forms`!

Finally you have to update your database. If you use Django >= 1.7 you need to run:

```
$ python manage.py migrate dynamic_forms
```

If you use [South](#) you need to run:

```
$ python manage.py syncdb  
$ python manage.py migrate
```

otherwise, if you don't use South or if you use Django<1.7, you have to run:

```
$ python manage.py syncdb
```


2.1 Using Custom Templates

django-dynamic-forms comes with a basic template that just displays the form or a success page. You can customize these templates to your needs.

2.1.1 The Form Template

The following code shows the default template rendering a dynamic form.

```
{% load i18n %}
<h2>{{ name }}</h2>
<form method="post" action="{{ submit_url }}">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">{% trans "Submit" %}</button>
</form>
```

The `DynamicFormView` exposes three variables to the template context related to the form:

form An instance of the form that will be shown on this page. As the form is a normal Django form, all rules from the [Django documentation](#) apply.

model An instance of the form model providing the form and assigned to this URL.

name The form's name as defined in `dynamic_forms.models.FormModel.name`.

success_url The URL the form will be submitted to as defined in `dynamic_forms.models.FormModel.submit_url`. This is *not* the `success_url`!

2.1.2 The Success Template

The following code shows the success template after a successful form submit.

```
{% load i18n %}
<h2>{% trans "Success" %}</h2>
<p>{% trans "Form submitted successfully" %}</p>
{% if data %}<p>{% blocktrans with link=data.show_url_link %}For your convenience you can see your data</p>
```

The `DynamicTemplateView` exposes two variables to the template context related to the form:

model An instance of the form model assigned to this URL.

data If an instance of `FormModelData` if a existing `display_key` is given and the form model (`model`) has set `allow_display` to `True`.

2.2 Third Party Apps

2.2.1 django-simple-captcha

`django-simple-captcha` provides easy CAPTCHA support for Django forms. This contrib package integrates `django-simple-captcha` into **django-dynamic-forms** allowing users to add a CAPTCHA field to their dynamic forms.

To use it make sure you installed **django-simple-captcha**:

```
$ pip install django-simple-captcha
```

Next put `'captcha'` and `'dynamic_forms.contrib.simple_captcha'` in the `INSTALLED_APPS`:

```
INSTALLED_APPS = (
    ...
    'captcha',
    'dynamic_forms.contrib.simple_captcha',
    ...
)
```

Problems

3.1 My form is not shown

If you are sure you followed the *Installation* instructions, there are several reasons:

1. You might have misspelled the URL path. Please check again.
2. There is another view that maps to the URL you defined in your model. 1. If you have Django's flatpages framework installed, check please check that there is not page mapping to this URL.
3. An error occurs while constructing the form or rendering the template. Set `DEBUG = True` in your `settings.py` and have a look at the exception that is raised.

Changes

4.1 v0.5 (under development)

Warning: `django-dynamic-forms` 0.5.x will only support Django \geq 1.7!

4.2 v0.4

Warning: `django-dynamic-forms` 0.4.x will be the latest version branch that supports Django $<$ 1.7!

- Added support for Django 1.8 and experimental support for Django 1.9. (#19)
- Removed `django-appconf` dependency.
- Added per form email receivers (thanks to Nar Chhantyal). (#13)
- Increased form fields' label `max_length` to 255 (thanks to Nar Chhantyal). (#18)

4.3 v0.3.4

- Fixed a issue with missing migrations on Python 2. (#11)

4.4 v0.3.3

- Updated Portuguese translation (thanks Gladson Simplicio). (#8)

4.5 v0.3.2

- Introduced the settings variables `DYNAMIC_FORMS_FORM_TEMPLATES` and `DYNAMIC_FORMS_SUCCESS_TEMPLATES` to make defining the templates for the form and success display easier and more usable for non-developers. (#1)
- Allowed delayed registration of *actions* and *dynamic form fields*.

- Allowed dynamic fields to exclude their value from the `mapped_data` by overriding `do_display_data()`.
- Added support for `django-simple-captcha`. (#2)
- Added Portuguese translation (thanks Gladson Simplicio). (#4)
- Replaced `formfields.dynamic_form_field_registry` with `formfields.formfield_registry` and deprecated the former.
- Fixed sorting of actions and field types by their label. (#5)
- Allowed users to get a link to see the data they submitted before at a later time. (#6)

4.6 v0.2

- Fixed some packaging issues. (thanks Jannis Leidel)
- Added Django 1.7's `db.migrations`.
- Moved to `tox` for development testing.

4.7 v0.1

- Initial release

Warning: You should not import the `ActionRegistry` directly! Always use the singleton instance `action_registry`!

```
>>> from dynamic_forms.actions import action_registry
```

get (*key*)

Parameters **key** (*str*) – The key to get an action

Returns Either the action previously registered or `None` if no action with the given key has been found.

get_as_choices ()

Changed in version 0.3: Returns a generator instead of a list

Returns a generator that yields all registered actions as 2-tuples in the form (*key*, *label*).

register (*func*, *label*)

Registers the function *func* with the label *label*. The function will internally be referred by its full qualified name:

```
'%s.%s' % (func.__module__, func.__name__)
```

Parameters

- **func** (*callable*) – The function to register.
- **label** (*str*) – A string / unicode giving the action a human readable name

unregister (*key*)

Looks up the given key in the internal dictionary and deletes the action if it exists.

Parameters **key** (*str*) – The key an action is assigned to

`dynamic_forms.actions.action_registry`

The singleton instance of the `ActionRegistry`.

Action registry utilities

`@dynamic_forms.actions.formmodel_action` (*label*)

Registering various actions by hand can be time consuming. This function decorator eases this heavily: given a string as the first argument, this decorator registers the decorated function withing the `action_registry` with its fully dotted Python path.

Usage:

```
@formmodel_action('My super awesome action')
def my_action(form_model, form):
    # do something with the data ...
```

This is equivalent to:

```
def my_action(form_model, form):
    # do something with the data ...
```

```
action_registry.register(my_action, 'My super awesome action')
```


5.1.2 Default Actions

`dynamic_forms.actions.dynamic_form_send_email(form_model, form)`

Sends the data submitted through the form `form` via e-mail to all recipients listed in `DYNAMIC_FORMS_EMAIL_RECIPIENTS`.

Parameters

- **form_model** (*dynamic_forms.models.ModelForm*) – The instance of the model defining the form.
- **form** (*dynamic_forms.forms.ModelFormForm*) – The instance of the submitted form. One can get the data either using `form.cleaned_data` or, if the labels defined in the `form_model` for each field are needed, in the appropriate order by calling `get_mapped_data()`.

`dynamic_forms.actions.dynamic_form_store_database(form_model, form)`

This action takes the mapped data from the `form` and serializes it as JSON. This value is then stored in the `FormModelData`.

See also:

`dynamic_form_store_database()` for a detailed explanation of the arguments.

New in version 0.3: To allow linking to a stored data set, the action now returns the inserted object.

5.2 Admin

5.3 Fields

class `dynamic_forms.fields.TextMultiSelectField([separate_values_by='\n', **options])`

Provides multiple choice field storage for strings without limiting the total length of the string or giving any restrictions of which characters are not allowed because they are used to split the input value into its different choices.

Parameters *separate_values_by* (*str*) – The string used to split the input value into its choices.
Defaults to `'\n'`.

See also:

The respective form field as part of **django-dynamic-forms** `dynamic_forms.forms.MultiSelectFormField`.
The common field options and the specifics for the `django.db.models.TextField`.

Note: The implementation is based on <http://djangosnippets.org/snippets/2753/> but has been modified to the needs of this project. Thus, there is no conversion of the selected items to `int` or similar.

5.4 Form fields

`dynamic_forms.formfields.format_display_type(cls_name)`

`dynamic_forms.formfields.load_class_from_string(cls_string)`

5.4.1 DynamicFormFieldRegistry

`class dynamic_forms.formfields.DynamicFormFieldRegistry(object)`

`get(key)`

`get_as_choices()`

Changed in version 0.3: Returns a generator instead of a list

Returns a generator that yields all registered dynamic form fields as 2-tuples in the form (key, display_type).

`register(cls)`

`unregister(key)`

`dynamic_forms.formfields.formfield_registry`

New in version 0.3: Use this instead of `dynamic_form_field_registry`

`dynamic_forms.formfields.dynamic_form_field_registry`

Deprecated since version 0.3: Deprecated in favor of `formfield_registry`

`@dynamic_forms.formfields.dynamic_form_field(cls)`

A class decorator to register the class as a dynamic form field in the `DynamicFormFieldRegistry`.

5.4.2 Base Form Field Classes

DFFMetaclass

`class dynamic_forms.formfields.DFFMetaclass`

Metaclass that inspects the Meta class of a class inheriting from `BaseDynamicFormField` and merges the different attributes that are later being passed to the respective `django.forms.Field`.

You are free to add an attribute `_exclude` of type list or tuple to the Meta class of a field to exclude any attributes inherited from a super `DynamicFormField`. Look at the implementation of the `BooleanField` for an example.

BaseDynamicFormField

`class dynamic_forms.formfields.BaseDynamicFormField`

`cls`

None

`display_type`

None

`widget`

None

`options`

`class Meta`

`help_text`

[six.string_types, ‘’, (forms.CharField, forms.Textarea)]

```

required
    [bool, True, forms.NullBooleanField]

dynamic_forms.formfields.__init__(name, label, widget_attrs={}, **kwargs)

dynamic_forms.formfields.__str__()
dynamic_forms.formfields.__unicode__()

dynamic_forms.formfields.construct([**kwargs])

dynamic_forms.formfields.contribute_to_form(form)

dynamic_forms.formfields.get_display_type()
    Returns a class's display_type is defined or calls format_display_type() with the class's
    name.

    This function is only available to the class itself. It is not callable from an instance.

dynamic_forms.formfields.get_widget_attrs()

dynamic_forms.formfields.set_options([**kwargs])

dynamic_forms.formfields.options_valid()

classmethod dynamic_forms.formfields.do_display_data()
    Default: True

```

5.4.3 Default Fields

```
class dynamic_forms.formfields.BooleanField
```

```

cls
    'django.forms.BooleanField

display_type
    'Boolean

class Meta

    _exclude
        ('required',)

```

```
class dynamic_forms.formfields.ChoiceField
```

```

cls
    'django.forms.ChoiceField

display_type
    'Choices

class Meta

    choices
        [six.string_types, '', (forms.CharField, forms.Textarea)]

dynamic_forms.formfields.construct([**kwargs])

dynamic_forms.formfields.options_valid()

```

```
class dynamic_forms.formfields.DateField

    cls
        'django.forms.DateField

    display_type
        'Date

    class Meta

        localize
            [bool, True, forms.NullBooleanField]

class dynamic_forms.formfields.DateTimeField

    cls
        'django.forms.DateTimeField

    display_type
        'Date and Time'

    class Meta

        localize
            [bool, True, forms.NullBooleanField]

class dynamic_forms.formfields.EmailField

    cls
        'django.forms.EmailField

    display_type
        'Email

class dynamic_forms.formfields.IntegerField

    cls
        'django.forms.IntegerField

    display_type
        'Integer

    class Meta

        localize
            [bool, True, forms.NullBooleanField]

        max_value
            [int, None, forms.IntegerField]

        min_value
            [int, None, forms.IntegerField]

class dynamic_forms.formfields.MultiLineTextField
```

```

    cls
        'django.forms.CharField

    display_type
        'Multi Line Text

    widget
        'django.forms.widgets.Textarea
class dynamic_forms.formfields.SingleLineTextField

    cls
        'django.forms.CharField

    display_type
        'Single Line Text

    class Meta

        max_length
            [int, None, forms.IntegerField]

        min_length
            [int, None, forms.IntegerField]
class dynamic_forms.formfields.TimeField

    cls
        'django.forms.TimeField

    display_type
        'Time

    class Meta

        localize
            [bool, True, forms.NullBooleanField]

```

5.5 Forms

5.5.1 Form Fields

class `dynamic_forms.forms.MultiSelectFormField` (`[separate_values_by='\n', **options]`)
 Provides multiple choice field storage for string objects without limiting the total length of the string.

Parameters `separate_values_by` (*str*) – The string used to split the input value into its choices.
 Defaults to `'\n'`.

See also:

The respective database field as part of **django-dynamic-forms** `dynamic_forms.fields.TextMultiSelectField`.
The core form field arguments and the specifics for the `django.forms.MultipleChoiceField`.

Note: The implementation is based on <http://djangosnippets.org/snippets/2753/> but has been modified to the needs of this project.

5.5.2 Forms

```
class dynamic_forms.forms.FormModelForm(model[, *args, **kwargs])
```

```
    get_mapped_data([exclude_missing=False])
```

Returns a dictionary sorted by the position of the respective field in its form.

Parameters `exclude_missing` (*boolean*) – If `True`, non-filled fields (those whose value evaluates to `False`) are not present in the returned dictionary. Default: `False`

5.6 Middlewares

```
class dynamic_forms.middlewares.FormModelMiddleware
```

This middleware intercepts all HTTP 404 responses and checks if there is a form mapped to this URL. This way an explicit URL mapping from the projects `ROOT_URLCONF` cannot accidentally be overridden by wrong setting for `submit_url` or `success_url` on `dynamic_forms.models.FormModel`.

This technique is comparable to the one used by Django's `FlatpageFallbackMiddleware`.

```
    process_response(request, response)
```

The algorithm that decides if and which form to display works like this:

- 1.If the `status_code` for response is **not** 404 (NOT FOUND) this the `FormModelMiddleware` will return the response as-is and will not modify it. Thus, server error (5xx) will also not be affected by the middleware.
- 2.If there is a `FormModel` whose `submit_url` matches the request's `path_info`, this model is used to construct and render the view.
- 3.If there is a `FormModel` whose `success_url` matches the request's `path_info`, this model is used to display the success page.

Note: Since the `success_url` of a `FormModel` is not necessarily be unique, the first model that matches the request path will be used.

- 4.If any errors occur while processing a form the original request is returned (if `DEBUG = True` the respective exception is raised).

5.7 Models

5.7.1 FormModel

```
class dynamic_forms.models.FormModel
```

```
    name
        django.db.models.CharField
        •max_length = 50
        •unique = True
    submit_url
        django.db.models.CharField
```

```

        •max_length = 100
        •unique = True
success_url
    django.db.models.CharField
        •max_length = 100
        •blank = True
        •default = ''
actions
    dynamic_forms.fields.TextMultiSelectField
        •default = ''
        •choices = dynamic_forms.actions.ActionRegistry.get_as_choices()
form_template
    django.db.models.CharField
        •max_length = 100
        •choices = dynamic_forms.conf.DYNAMIC_FORMS_FORM_TEMPLATES
success_template
    django.db.models.CharField
        •max_length = 100
        •choices = dynamic_forms.conf.DYNAMIC_FORMS_SUCCESS_TEMPLATES
allow_display
    django.db.models.BooleanField
        •default = False
recipient_email
    django.db.models.EmailField
fields
    Related name by FormFieldModel
data
    Related name by FormModelData
class Meta

    ordering
        ['name']

dynamic_forms.models.__str__()
dynamic_forms.models.__unicode__()

dynamic_forms.models.get_fields_as_dict()
dynamic_forms.models.save([*args, **kwargs])

```

5.7.2 FormFieldModel

```
class dynamic_forms.models.FormFieldModel
```

```

parent_form
    django.db.models.ForeignKey
        •Foreign key to FormModel
        •on_delete = django.db.models.CASCADE

field_type
    django.db.models.CharField
        •max_length = 255
        •choices = dynamic_forms.formfields.DynamicFormFieldRegistry.get_as_choices()

label
    django.db.models.CharField
        •max_length = 255

name
    django.db.models.CharField
        •max_length = 50
        •blank = True

_options
    django.db.models.TextField
        •blank = True
        •null = True

position
    django.db.models.SmallIntegerField
        •blank = True
        •default = 0

options
    Property wrapping JSON serialization and deserialization around the _options.

class Meta

    ordering
        ['parent_form', 'position']

    unique_together
        ("parent_form", "name",)

dynamic_forms.models.__str__()
dynamic_forms.models.__unicode__()

dynamic_forms.models.generate_form_field(form)

dynamic_forms.models.get_form_field_kwargs()

dynamic_forms.models.save([*args, **kwargs])

```

5.7.3 FormModelData

```

class dynamic_forms.models.FormModelData

```



```

form
    django.db.models.ForeignKey
        •Foreign key to FormModel
        •on_delete = django.db.models.SET_NULL
        •null = True

value
    django.db.models.TextField
        •blank = True
        •default = ''

submitted
    django.db.models.DateTimeField
        •auto_now_add = True

__str__()
__unicode__()

pretty_value()

show_url
    If the form this data set belongs to has allow_display == True, return the permanent URL. If displaying is not allowed, return an empty string.

show_url_link
    Similar to show_url but wraps the display key in an <a>-tag linking to the permanent URL.

```

5.8 Settings

5.8.1 DYNAMIC_FORMS_EMAIL_RECIPIENTS

`dynamic_forms.conf.DYNAMIC_FORMS_EMAIL_RECIPIENTS`

A list of email addresses. Used to define the recipients form data will be send to if the action `dynamic_form_send_email()` is activated.

Defaults to all email addresses defined in the `ADMINS` setting.

5.8.2 DYNAMIC_FORMS_FORM_TEMPLATES

`dynamic_forms.conf.DYNAMIC_FORMS_FORM_TEMPLATES`

New in version 0.3.

A tuple of 2-tuples passed to the `FormModel`'s `form_template` attribute. This setting provides easier and less error-prone definition of the form template.

Defaults to:

```

(
    ('dynamic_forms/form.html', _('Default form template')),
)

```

5.8.3 DYNAMIC_FORMS_SUCCESS_TEMPLATES

`dynamic_forms.conf.DYNAMIC_FORMS_SUCCESS_TEMPLATES`

New in version 0.3.

A tuple of 2-tuples passed to the `FormModel`'s *success_template* attribute. This setting provides easier and less error-prone definition of the form template.

```
(
    ('dynamic_forms/form_success.html', _('Default success template')),
)
```

5.9 Views

`class dynamic_forms.views.DynamicFormView`

form_valid (*form*)

Instantiates an empty dict `self.action_results` that takes the return values of every action that is called, unless the return value of that action is `None`.

get_success_url ()

If the `dynamic_form_store_database` action is active for the current form, include the `display_key` for the newly created data set.

`class dynamic_forms.views.DynamicTemplateView`

get_context_data (***kwargs*)

If a `display_key` query parameter is given and the key belongs to a `FormModel` that has `allow_display=True`, add the data to the normal Django *TemplateView* context.

`class dynamic_forms.views.DynamicDataMixin`

slug_field = `u'display_key'`

`'display_key'`

slug_url_kwarg = `u'display_key'`

`'display_key'`

template_name_404 = `u'dynamic_forms/data_set_404.html'`

`'dynamic_forms/data_set_404.html'`

`class dynamic_forms.views.DynamicDataSetDetailView`

model = `<class 'dynamic_forms.models.FormModelData'>`

The `FormDataModel`

template_name = `u'dynamic_forms/data_set.html'`

`'dynamic_forms/data_set.html'`

5.10 Contrib packages

5.10.1 django-simple-captcha

```
class dynamic_forms.contrib.simple_captcha.models.CaptchaField

    cls
        'captcha.fields.CaptchaField'

    display_type
        'CAPTCHA'

    class Meta

        _exclude
            ('required',)

    classmethod dynamic_forms.contrib.simple_captcha.models.do_display_data()
        Default: True
```

Indices and tables

- *genindex*
- *modindex*
- *search*

a

`dynamic_forms.actions`, [11](#)
`dynamic_forms.admin`, [13](#)

c

`dynamic_forms.conf`, [21](#)
`dynamic_forms.contrib.simple_captcha`, [6](#)
`dynamic_forms.contrib.simple_captcha.models`,
[23](#)

f

`dynamic_forms.fields`, [13](#)
`dynamic_forms.formfields`, [13](#)
`dynamic_forms.forms`, [17](#)

m

`dynamic_forms.middlewares`, [18](#)
`dynamic_forms.models`, [18](#)

v

`dynamic_forms.views`, [22](#)

Symbols

__init__() (in module dynamic_forms.formfields), 15
 __str__() (dynamic_forms.models.ModelFormData method), 21
 __str__() (in module dynamic_forms.formfields), 15
 __str__() (in module dynamic_forms.models), 19, 20
 __unicode__() (dynamic_forms.models.ModelFormData method), 21
 __unicode__() (in module dynamic_forms.formfields), 15
 __unicode__() (in module dynamic_forms.models), 19, 20
 _exclude (dynamic_forms.contrib.simple_captcha.models.CaptchaField attribute), 23
 _exclude (dynamic_forms.formfields.BooleanField.Meta attribute), 15
 _options (dynamic_forms.models.FormFieldModel attribute), 20

A

action_registry (in module dynamic_forms.actions), 12
 ActionRegistry (class in dynamic_forms.actions), 11
 actions (dynamic_forms.models.ModelForm attribute), 19
 allow_display (dynamic_forms.models.ModelForm attribute), 19

B

BaseDynamicFormField (class in dynamic_forms.formfields), 14
 BaseDynamicFormField.Meta (class in dynamic_forms.formfields), 14
 BooleanField (class in dynamic_forms.formfields), 15
 BooleanField.Meta (class in dynamic_forms.formfields), 15

C

CaptchaField (class in dynamic_forms.contrib.simple_captcha.models), 23

CaptchaField.Meta (class in dynamic_forms.contrib.simple_captcha.models), 23
 ChoiceField (class in dynamic_forms.formfields), 15
 ChoiceField.Meta (class in dynamic_forms.formfields), 15
 choices (dynamic_forms.formfields.ChoiceField.Meta attribute), 15
 cls (dynamic_forms.contrib.simple_captcha.models.CaptchaField attribute), 23
 cls (dynamic_forms.formfields.BaseDynamicFormField attribute), 14
 cls (dynamic_forms.formfields.BooleanField attribute), 15
 cls (dynamic_forms.formfields.ChoiceField attribute), 15
 cls (dynamic_forms.formfields.DateField attribute), 16
 cls (dynamic_forms.formfields.DateTimeField attribute), 16
 cls (dynamic_forms.formfields.EmailField attribute), 16
 cls (dynamic_forms.formfields.IntegerField attribute), 16
 cls (dynamic_forms.formfields.MultiLineTextField attribute), 16
 cls (dynamic_forms.formfields.SingleLineTextField attribute), 17
 cls (dynamic_forms.formfields.TimeField attribute), 17
 construct() (in module dynamic_forms.formfields), 15
 contribute_to_form() (in module dynamic_forms.formfields), 15

D

data (dynamic_forms.models.ModelForm attribute), 19
 DateField (class in dynamic_forms.formfields), 15
 DateField.Meta (class in dynamic_forms.formfields), 16
 DateTimeField (class in dynamic_forms.formfields), 16
 DateTimeField.Meta (class in dynamic_forms.formfields), 16
 DFFMetaClass (class in dynamic_forms.formfields), 14
 display_type (dynamic_forms.contrib.simple_captcha.models.CaptchaField attribute), 23
 display_type (dynamic_forms.formfields.BaseDynamicFormField attribute), 14

- display_type (dynamic_forms.formfields.BooleanField attribute), 15
 - display_type (dynamic_forms.formfields.ChoiceField attribute), 15
 - display_type (dynamic_forms.formfields.DateField attribute), 16
 - display_type (dynamic_forms.formfields.DateTimeField attribute), 16
 - display_type (dynamic_forms.formfields.EmailField attribute), 16
 - display_type (dynamic_forms.formfields.IntegerField attribute), 16
 - display_type (dynamic_forms.formfields.MultiLineTextField attribute), 17
 - display_type (dynamic_forms.formfields.SingleLineTextField attribute), 17
 - display_type (dynamic_forms.formfields.TimeField attribute), 17
 - do_display_data() (in module dynamic_forms.contrib.simple_captcha.models), 23
 - do_display_data() (in module dynamic_forms.formfields), 15
 - dynamic_form_field() (in module dynamic_forms.formfields), 14
 - dynamic_form_field_registry (in module dynamic_forms.formfields), 14
 - dynamic_form_send_email() (in module dynamic_forms.actions), 13
 - dynamic_form_store_database() (in module dynamic_forms.actions), 13
 - dynamic_forms.actions (module), 11
 - dynamic_forms.admin (module), 13
 - dynamic_forms.conf (module), 21
 - dynamic_forms.contrib.simple_captcha (module), 6
 - dynamic_forms.contrib.simple_captcha.models (module), 23
 - dynamic_forms.fields (module), 13
 - dynamic_forms.formfields (module), 13
 - dynamic_forms.forms (module), 17
 - dynamic_forms.middlewares (module), 18
 - dynamic_forms.models (module), 18
 - dynamic_forms.views (module), 22
 - DYNAMIC_FORMS_EMAIL_RECIPIENTS (in module dynamic_forms.conf), 21
 - DYNAMIC_FORMS_FORM_TEMPLATES (in module dynamic_forms.conf), 21
 - DYNAMIC_FORMS_SUCCESS_TEMPLATES (in module dynamic_forms.conf), 22
 - DynamicDataMixin (class in dynamic_forms.views), 22
 - DynamicDataSetDetailView (class in dynamic_forms.views), 22
 - DynamicFormFieldRegistry (class in dynamic_forms.formfields), 14
 - DynamicFormView (class in dynamic_forms.views), 22
 - DynamicTemplateView (class in dynamic_forms.views), 22
- ## E
- EmailField (class in dynamic_forms.formfields), 16
- ## F
- field_type (dynamic_forms.models.FormFieldModel attribute), 20
 - fields (dynamic_forms.models.FormModel attribute), 19
 - form (dynamic_forms.models.FormModelData attribute), 20
 - form_template (dynamic_forms.models.FormModel attribute), 19
 - form_valid() (dynamic_forms.views.DynamicFormView method), 22
 - format_display_type() (in module dynamic_forms.formfields), 13
 - formfield_registry (in module dynamic_forms.formfields), 14
 - FormFieldModel (class in dynamic_forms.models), 19
 - FormFieldModel.Meta (class in dynamic_forms.models), 20
 - FormModel (class in dynamic_forms.models), 18
 - FormModel.Meta (class in dynamic_forms.models), 19
 - formmodel_action() (in module dynamic_forms.actions), 12
 - FormModelData (class in dynamic_forms.models), 20
 - FormModelForm (class in dynamic_forms.forms), 18
 - FormModelMiddleware (class in dynamic_forms.middlewares), 18
- ## G
- generate_form_field() (in module dynamic_forms.models), 20
 - get() (dynamic_forms.actions.ActionRegistry method), 12
 - get() (dynamic_forms.formfields.DynamicFormFieldRegistry method), 14
 - get_as_choices() (dynamic_forms.actions.ActionRegistry method), 12
 - get_as_choices() (dynamic_forms.formfields.DynamicFormFieldRegistry method), 14
 - get_context_data() (dynamic_forms.views.DynamicTemplateView method), 22
 - get_display_type() (in module dynamic_forms.formfields), 15
 - get_fields_as_dict() (in module dynamic_forms.models), 19
 - get_form_field_kwargs() (in module dynamic_forms.models), 20

get_mapped_data() (dynamic_forms.forms.ModelFormForm method), 18
 get_success_url() (dynamic_forms.views.DynamicFormView method), 22
 get_widget_attrs() (in module dynamic_forms.formfields), 15

H

help_text (dynamic_forms.formfields.BaseDynamicFormField attribute), 14

I

IntegerField (class in dynamic_forms.formfields), 16
 IntegerField.Meta (class in dynamic_forms.formfields), 16

L

label (dynamic_forms.models.FormFieldModel attribute), 20
 load_class_from_string() (in module dynamic_forms.formfields), 13
 localize (dynamic_forms.formfields.DateField.Meta attribute), 16
 localize (dynamic_forms.formfields.DateTimeField.Meta attribute), 16
 localize (dynamic_forms.formfields.IntegerField.Meta attribute), 16
 localize (dynamic_forms.formfields.TimeField.Meta attribute), 17

M

max_length (dynamic_forms.formfields.SingleLineTextField.Meta attribute), 17
 max_value (dynamic_forms.formfields.IntegerField.Meta attribute), 16
 min_length (dynamic_forms.formfields.SingleLineTextField.Meta attribute), 17
 min_value (dynamic_forms.formfields.IntegerField.Meta attribute), 16
 model (dynamic_forms.views.DynamicDataSetDetailView attribute), 22
 MultiLineTextField (class in dynamic_forms.formfields), 16
 MultiSelectFormField (class in dynamic_forms.forms), 17

N

name (dynamic_forms.models.FormFieldModel attribute), 20
 name (dynamic_forms.models.FormModel attribute), 18

O

options (dynamic_forms.formfields.BaseDynamicFormField attribute), 14

options (dynamic_forms.models.FormFieldModel attribute), 20
 options_valid() (in module dynamic_forms.formfields), 15
 ordering (dynamic_forms.models.FormFieldModel.Meta attribute), 20
 ordering (dynamic_forms.models.ModelForm.Meta attribute), 19

P

parent_form (dynamic_forms.models.FormFieldModel attribute), 19
 position (dynamic_forms.models.FormFieldModel attribute), 20
 pretty_value() (dynamic_forms.models.ModelFormData method), 21
 process_response() (dynamic_forms.middlewares.ModelFormMiddleware method), 18

R

recipient_email (dynamic_forms.models.ModelForm attribute), 19
 register() (dynamic_forms.actions.ActionRegistry method), 12
 register() (dynamic_forms.formfields.DynamicFormFieldRegistry method), 14
 required (dynamic_forms.formfields.BaseDynamicFormField.Meta attribute), 14

S

save() (in module dynamic_forms.models), 19, 20
 set_options() (in module dynamic_forms.formfields), 15
 show_url (dynamic_forms.models.ModelFormData attribute), 21
 show_url_link (dynamic_forms.models.ModelFormData attribute), 21
 SingleLineTextField (class in dynamic_forms.formfields), 17
 SingleLineTextField.Meta (class in dynamic_forms.formfields), 17
 slug_field (dynamic_forms.views.DynamicDataMixin attribute), 22
 slug_url_kwarg (dynamic_forms.views.DynamicDataMixin attribute), 22
 submit_url (dynamic_forms.models.ModelForm attribute), 18
 submitted (dynamic_forms.models.ModelFormData attribute), 21
 success_template (dynamic_forms.models.ModelForm attribute), 19
 success_url (dynamic_forms.models.ModelForm attribute), 19

T

`template_name` (`dynamic_forms.views.DynamicDataSetDetailView` attribute), [22](#)
`template_name_404` (`dynamic_forms.views.DynamicDataMixin` attribute), [22](#)
`TextMultiSelectField` (class in `dynamic_forms.fields`), [13](#)
`TimeField` (class in `dynamic_forms.formfields`), [17](#)
`TimeField.Meta` (class in `dynamic_forms.formfields`), [17](#)

U

`unique_together` (`dynamic_forms.models.FormFieldModel.Meta` attribute), [20](#)
`unregister()` (`dynamic_forms.actions.ActionRegistry` method), [12](#)
`unregister()` (`dynamic_forms.formfields.DynamicFormFieldRegistry` method), [14](#)

V

`value` (`dynamic_forms.models.FormModelData` attribute), [21](#)

W

`widget` (`dynamic_forms.formfields.BaseDynamicFormField` attribute), [14](#)
`widget` (`dynamic_forms.formfields.MultiLineTextField` attribute), [17](#)