django-dajaxice Documentation

Release 0.5.5

Jorge Bastida

October 11, 2013

CONTENTS

Dajaxixe is an Easy to use AJAX library for django. Its main goal is to trivialize the asynchronous communication within the django server code and your js code. Dajaxice uses the unobtrusive standard-compliant (W3C) XMLHttpRequest 1.0 object.

django-dajaxice is a **JS-framework agnostic** library and focuses on decoupling the presentation logic from the server-side logic. dajaxice only requieres **5 minutes to start working.**

Dajaxice has the following aims:

- Isolate the communication between the client and the server.
- JS Framework agnostic (No Prototype, JQuery... needed).
- Presentation logic outside the views (No presentation code inside ajax functions).
- Lightweight.
- Crossbrowsing ready.
- Unobtrusive standard-compliant (W3C) XMLHttpRequest 1.0 object usage.

CHAPTER

ONE

DOCUMENTATION

1.1 Installation

Follow this instructions to start using dajaxice in your django project.

1.1.1 Installing dajaxice

Add *dajaxice* in your project settings.py inside `INSTALLED_APPS`:

```
INSTALLED_APPS = (
        'django.contrib.auth',
        'django.contrib.contenttypes',
        'django.contrib.sessions',
        'django.contrib.sites',
        'dajaxice',
        ...
)
```

Ensure that your `TEMPLATE_LOADERS`, looks like the following. Probably you'll only need to uncomment the last line.:

```
TEMPLATE_LOADERS = (
    'django.template.loaders.filesystem.Loader',
    'django.template.loaders.app_directories.Loader',
    'django.template.loaders.eggs.Loader',
)
```

Ensure that `TEMPLATE_CONTEXT_PROCESSORS` has `django.core.context_processors.request`. Probably you'll only need to add the last line:

```
TEMPLATE_CONTEXT_PROCESSORS = (
    'django.contrib.auth.context_processors.auth',
    'django.core.context_processors.debug',
    'django.core.context_processors.i18n',
    'django.core.context_processors.media',
    'django.core.context_processors.static',
    'django.core.context_processors.request',
    'django.contrib.messages.context_processors.messages'
)
```

Add `dajaxice.finders.DajaxiceFinder` to `STATICFILES_FINDERS`:

```
STATICFILES_FINDERS = (
    'django.contrib.staticfiles.finders.FileSystemFinder',
    'django.contrib.staticfiles.finders.AppDirectoriesFinder',
    'dajaxice.finders.DajaxiceFinder',
)
```

1.1.2 Configure dajaxice url

Add the following code inside urls.py:

```
from dajaxice.core import dajaxice_autodiscover, dajaxice_config
dajaxice_autodiscover()
```

Add a new line in urls.py urlpatterns with this code:

```
urlpatterns = patterns('',
    ...
    url(dajaxice_config.dajaxice_url, include('dajaxice.urls')),
    ...
)
```

If you aren't using `django.contrib.staticfiles`, you should also enable it importing:

```
from django.contrib.staticfiles.urls import staticfiles_urlpatterns
```

and adding this line to the bottom of your urls.py:

```
urlpatterns += staticfiles_urlpatterns()
```

1.1.3 Install dajaxice in your templates

Dajaxice needs some JS to work. To include it in your templates, you should load `dajaxice_templatetags` and use `dajaxice_js_import` TemplateTag inside your head section. This TemplateTag will print needed js.

```
{% load dajaxice_templatetags %}

<html>
  <head>
    <title>My base template</title>
    ...
    {% dajaxice_js_import %}
  </head>
    ...
</html>
```

This templatetag will include all the js dajaxice needs.

1.1.4 Use Dajaxice!

Now you can create your first ajax function following the *Quickstart*.

1.2 Quickstart

1.2.1 Create your first ajax function

Create a file named `ajax.py` inside any of your apps. For example `example/ajax.py`.

Inside this file create a simple function that return json.:

```
from django.utils import simplejson

def sayhello(request):
    return simplejson.dumps({'message':'Hello World'})
```

Now you'll need to register this function as a dajaxice function using the `dajaxice_register` decorator:

```
from django.utils import simplejson
from dajaxice.decorators import dajaxice_register

@dajaxice_register
def sayhello(request):
    return simplejson.dumps({'message':'Hello World'})
```

1.2.2 Invoque it from your JS

You can invoque your ajax fuctions from javascript using:

```
onclick="Dajaxice.example.sayhello(my_js_callback);"
```

The function `my_js_callback` is your JS function that will use your example return data. For example alert the message:

```
function my_js_callback(data){
    alert(data.message);
}
```

That callback will alert the message `Hello World`.

1.2.3 How can I do a GET request instead of a POST one?

When you register your functions as ajax functions, you can choose the http method using:

```
from django.utils import simplejson
from dajaxice.decorators import dajaxice_register

@dajaxice_register(method='GET')
def saybye(request):
    return simplejson.dumps({'message':'Bye!'})
```

This function will be executed doing a GET request and not a POST one.

1.2.4 Can I combine both?

Yes! You can register a function as many times as you want, for example:

```
from django.utils import simplejson
from dajaxice.decorators import dajaxice_register

@dajaxice_register(method='POST', name='user.update')
@dajaxice_register(method='GET', name='user.info')
def list_user(request):
    if request.method == 'POST':
        ...
    else:
        ...
```

In this case you'll be able to call this two JS functions:

```
Dajaxice.user.info( callback );
Dajaxice.user.update( callback );
```

The first one will be a GET call and the second one a POST one.

1.3 Custom error callbacks

1.3.1 How dajaxice handle errors

When one of your functions raises an exception dajaxice returns as response the `DAJAXICE_EXCEPTION` message. On every response `dajaxice.core.js` checks if that response was an error or not and shows the user a default error message `Something goes wrong`.

1.3.2 Customize the default error message

This behaviour is configurable using the new `Dajaxice.setup` function.

```
Dajaxice.setup({'default_exception_callback': function(){ alert('Error!'); }});
```

1.3.3 Customize error message per call

In this new version you can also specify an error callback per dajaxice call.

```
function custom_error(){
    alert('Custom error of my_function.');
}

Dajaxice.simple.my_function(callback, {'user': 'tom'}, {'error_callback': custom_error});
```

1.4 Utils

1.4.1 dajaxice.utils.deserialize_form

Using `deserialize_form` you will be able to deserialize a query_string and use it as input of a Form:

```
from dajaxice.utils import deserialize_form

@dajaxice_register
def send_form(request, form):
    form = ExampleForm(deserialize_form(form))
    if form.is_valid():
        ...
    ...
```

1.5 Production Environment

Since `0.5` dajaxice takes advantage of `django.contrib.staticfiles` so deploying a dajaxice application live is much easy than in previous versions.

You need to remember to run `python manage.py collectstatic` before deploying your code live. This command will collect all the static files your application need into `STATIC_ROOT`. For further information, this is the Django static files docuemntation

1.6 Migrating to 0.5

1.6.1 Upgrade to django 1.3 or 1.4

Dajaxice `0.5` requires `django>=1.3`, so in order to make dajaxice work you'll need to upgrade your app to any of these ones.

- Django 1.3 release notes
- Django 1.4 release notes

1.6.2 Make django static-files work

Add this at the beginning of your `urls.py` file:

```
from django.contrib.staticfiles.urls import staticfiles_urlpatterns
```

and add this line to the bottom of your urls.py:

```
urlpatterns += staticfiles_urlpatterns()
```

Add a new staticfiles finder named `dajaxice.finders.DajaxiceFinder` to the list of `STATICFILES_FINDERS`:

```
STATICFILES_FINDERS = ('django.contrib.staticfiles.finders.FileSystemFinder',
                       'django.contrib.staticfiles.finders.AppDirectoriesFinder',
                       'dajaxice.finders.DajaxiceFinder')
```

1.6.3 Update dajaxice core url

Add `dajaxice_config` to the list of modules to import:

```
# Old import
from dajaxice.core import dajaxice_autodiscover

# New import
from dajaxice.core import dajaxice_autodiscover, dajaxice_config
```

And replate your old dajaxice url with the new one:

```
# Old style
(r'^%s/' % settings.DAJAXICE_MEDIA_PREFIX, include('dajaxice.urls')),

# New style
url(dajaxice_config.dajaxice_url, include('dajaxice.urls')),
```

1.6.4 Done!

Your app should be working now! You can now read the *quickstart* to discover some of the new dajaxice features.

1.7 Available Settings

1.7.1 DAJAXICE_MEDIA_PREFIX

This will be the namespace that dajaxice will use as endpoint.

Defaults to `dajaxice`

Optional: `True`

1.7.2 DAJAXICE_XMLHTTPREQUEST_JS_IMPORT

Include XmlHttpRequest.js inside dajaxice.core.js

Defaults to `True`

Optional: `True`

1.7.3 DAJAXICE_JSON2_JS_IMPORT

Include json2.js inside dajaxice.core.js

Defaults to `True`

Optional: `True`

1.7.4 DAJAXICE_EXCEPTION

Default data sent when an exception occurs.

Defaults to `"DAJAXICE_EXCEPTION"`

Optional: `True`

1.8 Changelog

1.8.1 0.5.5

- Return XMLHttpRequest from concreate functions as well as from function call.
- Fixed django 1.5 compatibility: Content-Type have to be application/x-www-form-urlencoded otherwise Django discards POST data.
- Fix JS generation errors
- Fix @dajaxice_register legacy decorator

1.8.2 0.5.4.1

- Fix JS generation errors.

1.8.3 0.5.4

- Fix JS generation errors.

1.8.4 0.5.3

- Fix some Windows bugs.
- Fix some JS generation errors.
- Make dajaxice use CSRF_COOKIE_NAME.

1.8.5 0.5.2

- Fix GET dajaxice requests in order to send args as part of the url.

1.8.6 0.5.1

- Make django-dajaxice work with django 1.3
- Fix installation steps
- Update json2.js

1.8.7 0.5

- General Project clean-up
- Django>=1.3 is now a requirement
- Fixed numerous CSRF issues
- Dajaxice now use django.contrib.staticfiles
- Fix SERVER_ROOT_URL issues
- Fixed js_core issues accepting multiple arguments

- New upgraded documentation
- Marketing site (http://dajaxproject.com) is now open-source
- Fix JS generation issues
- Travis-ci integration

1.8.8 0.2

- Fix bug with the 'is_callback_a_function' variable in dajaxice.core.js
- Fix bug with csrftoken in landing pages using dajaxice.
- Improve reliability handling server errors.
- Exception handling was fully rewritten. Dajaxice default_error_callback is now configurable using Dajaxice.setup.
- Custom error messages per dajaxice call.
- Dajaxice now propagate docstrings to javascript dajaxice functions.
- Added DAJAXICE_JS_DOCSTRINGS to configure docstrings propagation behaviour, default=False.
- Updated installation guide for compatibility with django 1.3
- dajaxice now uses the logger 'dajaxice' and not 'dajaxice.DajaxiceRequest'
- Documentation Updated.

1.8.9 0.1.8.1

- Fixed bug #25 related to CSRF verification on Django 1.2.5

1.8.10 0.1.8

- Add build dir to ignores
- Remove MANIFEST file and auto-generate it through MANIFEST.in
- Add MANIFEST to ignores
- Include examples and docs dirs to source distribution
- Add long_description to setup.py
- Fixed Flaw in AJAX CSRF handling (X-CSRFToken Django 1.2.5)

1.8.11 0.1.7

- Fixing dajaxice callback model to improve security against XSS attacks.
- Dajaxice callbacks should be passed as functions and not as strings.
- Old string-callback maintained for backward compatibility.(usage not recommended)
- New documentation using Sphinx
- Adding a decorators.py file with a helper decorator to register functions (Douglas Soares de Andrade)

1.8.12 0.1.6

- Fixing registration bugs
- Added some tests

1.8.13 0.1.5

- Now dajaxice functions must be registered using dajaxice_functions.register instead of adding that functions to DAJAXICE_FUNCTIONS list inside settings.py. This pattern is very similar to django.contrib.admin model registration.
- Now dajaxice functions could be placed inside any module depth.
- With this approach dajaxice app reusability was improved.
- Old style registration (using DAJAXICE_FUNCTIONS) works too, but isn't recommended.
- New tests added.

1.8.14 0.1.3

- CSRF middleware buf fixed
- Improved production and development logging
- New custom Exception message
- New notify_exception to send traceback to admins
- Fixed semicolon issues
- Fixed unicode errors
- Fixed generate_static_dajaxice before easy_install usage
- Fixed IE6 bug in dajaxice.core.js

1.8.15 0.1.2

- New and cleaned setup.py

1.8.16 0.1.1

- json2.js and XMLHttpRequest libs included
- New settings DAJAXICE_XMLHTTPREQUEST_JS_IMPORT and DAJAXICE_JSON2_JS_IMPORT

1.8.17 0.1.0

- dajaxice AJAX functions now receive parameters as function arguments.
- dajaxice now uses standard python logging
- some bugs fixed

1.8.18 0.0.1

- First Release

CHAPTER

TWO

INDICES AND TABLES

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