django-instant Documentation

Release 0.1

synw

Nov 22, 2018

Install

Instant delivers events in real time to the userland using the Centrifugo websockets server.

CHAPTER 1

Install and configure

1. Install Django Instant:

```
pip install django-instant
```

Add to installed apps:

```
"corsheaders",
"instant",
```

2. Install the websockets server

An installer is available for Linux only:

```
python3 manage.py installws
```

This will download the Centrifugo websockets server, install it and update your settings.py file with the appropriate configuration.

For other systems you have to install Centrifugo manually (use version 1 as this module is not compatible with Centrifugo 2 yet)

1.1 Settings

Configure the middleware:

```
MIDDLEWARE = [
    ...
    'corsheaders.middleware.CorsMiddleware',
    'django.middleware.common.CommonMiddleware',
    ...
]
```

Note: if you use the management command to install the server the settings below will already be configured.

```
# required settings
CENTRIFUGO_SECRET_KEY = "70b651f6-775a-4949-982b-b387b31c1d84" # the_key_that_is_in_
↪config.json
SITE_SLUG = "my_site" # used internaly to prefix the channels
SITE_NAME = "My site"

CORS_ORIGIN_WHITELIST = ('localhost:8001',)

# optionnal settings
CENTRIFUGO_HOST = 'http://ip_here' #default: localhost
CENTRIFUGO_PORT = 8012 # default: 8001
CENTRIFUGO_PROXY = True # default: False - remove port from URL if you are behind a␣
↪proxy.
```

By default the events are published using python. A go module is available to perform the publish operations in order to leave the main process alone as much as possible. This might be usefull when lots of messages are sent.

Note: when this option is enabled there is no error handling. This option is recommended when you need higher performance and don't care about error messages.

You might have to make the *instant/go/publish* file executable with *chmod*

```
INSTANT_BROADCAST_WITH = 'go'
```

Performance test: 1000 messages:

- Python: 2.96 seconds
- Go: 1.41 seconds

10000 messages:

- Python: 29.57 seconds
- Go: 14.44 seconds

Note: this test uses the standard publish function so that each new event sent makes an new connection to Centrifugo.

1.2 Templates

Include the template `{% include "instant/client.html" %}` anywhere: nothing will be displayed it is the engine. See next section for messages handling.

1.3 Run the websockets server

If you used the installer:

```
python3 manage.py runws
```

Otherwise run the Centrifugo server normally

1.4 Frontend

A demo frontend is available. To use it:

```
pip install django-vitevue
```

Add *"vv"*, to installed apps

Set the urls:

```
from instant.views import instant_auth

urlpatterns = [
     # ...
     url(r'^centrifuge/auth/$', instant_auth, name='instant-auth'),
     url('^instant/', include('instant.urls')),
     ]
```

Login as superuser and go to */instant/*

CHAPTER 2

Declare channels

To declare new channels in settings.py use this format:

```
["channel_name", ["/path/where/to/connect/it"]]
```

If the second element of the tuple is set the channels will be connected only for the listed path. If not set the channel will autoconnect on every path. Example:

```
INSTANT_SUPERUSER_CHANNELS=[
   ["$mysite_admin1", ["/a/path", "/another/path"]],
   ['$mysite_admin2']
]
INSTANT_STAFF_CHANNELS=[
   ["$mysite_staff1", ["/a/path"]],
   ['$mysite_staff2'],
]
INSTANT_USERS_CHANNELS=[
   ['$mysite_users1'],
]
INSTANT_PUBLIC_CHANNELS=[
   ['mysite_public1'],
   ['mysite_public2'],
]
```

Important: every private channel name must start with a dollar sign

Be sure to configure auth urls if you use private channels: urls.py:

```
from instant.views import instant_auth

urlpatterns = [
     # ...
     url(r'^centrifuge/auth/$', instant_auth, name='instant-auth'),
     ]
```

CHAPTER 3

Default channel

Deprecation warning: this is going to be deprecated in favour of declarative channels.

A default public channel is available and enabled by default: to disable it: settings:

```
INSTANT_ENABLE_PUBLIC_CHANNEL = False
```

This channel is going to be disabled by default in next versions and then deprecated

CHAPTER 4

Send messages

To publish messages from python code:

```
from instant.producers import publish

# fire a message on the public channel whith error handling
err = publish(message='Hello world', channel="public", event_class="infos",
              data={"myfield":"my_value"})
if err != None:
   print("Error", str(err))

# send an instant debug message during development
publish("Something happened somewhere in the code", channel="$debug",
        event_class='debug')
```

Required parameters: `message` and `channel`

To send extra data pass some json into `data`. Note: for now this won't work if you use the Go engine.

Note: if no channel is provided the events are published to the default public channel. This behavior is going to be deprecated as the public channel is going to be removed in future versions so be sure to specify a channel.

CHAPTER 5

Client-side messages handlers

5.1 Default handler

The default messages handler is `templates/instant/handlers/default.js`. Ex usage:

```
if (channel === "somechan") {
   console.log(message)
}
```

5.2 Channel handlers

Each channel can have its own javascript handler: create it in the handlers directory: example: if a `$mychan` is declared in settings create a `templates/instant/handlers/$mychan.js` file to manage the handling logics for that channel:

```
if (event_class === "someclass") {
   console.log(message)
}
```

Available javascript variables in handlers:

`event_class` : class of the event

`channel` : name of the channel

`message` : text message

`data` : json payload

`site` : site slug

`uid` : unique id of the message

`timestamp` : date timestamp

5.3 Debug

Note: for javascript debugging you can set a `INSTANT_DEBUG = True` in settings.py

5.4 Connection handler

To perform custom actions on connect and disconnect events create templates:

Template *instant/events/connect.js* or *instant/events/disconnect.js*:

```
{% include "myapp/myjs.js %}
```

This javascript will be executed on the selected event

CHAPTER 6

Optionnal message labels

Some javascript is used in the demo frontend to format the messages label: `templates/instant/event_class_format.js`.

Some css classes are defined in `instant/static/instant.css`. To customize:

```
{
'default' : 'mq-label mq-default',
'important' : 'mq-label mq-important',
'ok' : 'mq-label mq-ok',
'info' : 'mq-label mq-info',
'debug' : 'mq-label mq-debug',
'warning' : 'mq-label mq-warning',
'error' : 'mq-label mq-error',
'object created' : 'mq-label mq-created',
'object edited' : 'mq-label mq-edited',
'object deleted' : 'mq-label mq-deleted',
}
```

Default icons (using font-awesome):

```
{
'default' : '<i class="fa fa-flash"></i>',
'important' : '<i class="fa fa-exclamation"></i>',
'ok' : '<i class="fa fa-thumbs-up"></i>',
'info' : '<i class="fa fa-info-circle"></i>',
'debug' : '<i class="fa fa-cog"></i>',
'warning' : '<i class="fa fa-exclamation"></i>',
'error' : '<i class="fa fa-exclamation-triangle"></i>',
'object edited' : '<i class="fa fa-pencil"></i>',
'object created' : '<i class="fa fa-plus"></i>',
'object deleted' : '<i class="fa fa-remove"></i>',
}
```

CHAPTER 7

Private channels

7.1 Authentication

To use private channels add this in you main url file:

```
from instant.views import instant_auth

urlpatterns = [
     # ...
     url(r'^centrifuge/auth/$', instant_auth, name='instant-auth'),
     ]
```

All the channels prefixed with a dollar sign $ are considered private.

```
from instant.producers import publish

publish(message='Private event', channel="$private_chan")
```

7.2 Security

All the messages sent to the Centrifugo server are signed using the secret key. When a client requests a connection to a private channels Centrifugo sends an ajax request to `/centrifuge/auth/` and expects to receive a signed response that will indicate if the user is authorized or not.

More info here about Centrifugo's auth mechanism.

CHAPTER 8

Custom channels

8.1 Custom javascript client

Create a `{% instant/extra_clients.js %}` template that contains something like:

```
{% if user.is_authenticated and request.path == "/private/" %}
     {% include "myapp/client.js" %}
{% endif %}
```

Edit `myapp/client.js`:

```
var my_callbacks = {
   "message": function(dataset) {
   // the instantDebug variable is set via INSTANT_DEBUG = True in settings.py
   if (instantDebug === true) { console.log('EVENT: '+JSON.stringify(dataset));};
   res = unpack_data(dataset);
   var message = res['message']
   var event_class = res['event_class']
   var message_label = res['message_label']
   var data = res['data']
   var channel = res['channel'];
   if ( data.hasOwnProperty('my_field') ) {
    my_field = data['myfield']
   }
   // do something with the data
   console.log(message);
},
{% include "instant/js/join_events.js" %}

var subscription = centrifuge.subscribe("$mychannel", my_callbacks);
```

8.2 Custom auth function

You can write a custom auth backend to authenticate the user. Example: urls.py:

```
from mymodule.views import mychan_auth_view

url(r'^centrifuge/auth/$', mychan_auth_view),
```

In views.py:

```
import json
from django.http import JsonResponse
from django.views.decorators.csrf import csrf_exempt
from django.http.response import Http404
from instant.utils import signed_response

@csrf_exempt
def mychan_auth_view(request):
    if not request.is_ajax() or not request.method == "POST":
        raise Http404
    data = json.loads(request.body)
    channels = data["channels"]
    client = data['client']
    response = {}
    for channel in channels:
        response[channel] = {"status","403"}
        if channel == "$mychannel":
            # checks come here
            if request.user.is_authenticated() and whatever():
               response[channel] = signed_response(channel, client)
    return JsonResponse(response)
```