Mozilla Developer Services Dashboard Documentation

Release 1.0

Luke Crouch

March 17, 2016

Contents

moz-dev-dash is the code for Mozilla Developer Services dashboard. It combines:

- Firefox Accounts Authentication (via django-allauth)
- (Soon) Domain Verification (via ?)
- Mozilla Push Service metrics

CHAPTER 1

Resources

Code https://github.com/mozilla-services/push-dev-dashboard

License MPL2

Documentation http://push-dev-dashboard.readthedocs.org/

Issues https://github.com/mozilla-services/push-dev-dashboard/issues

IRC irc://irc.mozilla.org/mds

Mailing list https://lists.mozilla.org/listinfo/mds-public

Servers https://moz-push-dash.herokuapp.com (stage)

CHAPTER 2

Contents

2.1 Development

2.1.1 Requirements

- postgres for database
- python 2.7, virtualenv, & pip for app server

2.1.2 Install Locally

1. Clone and change to the directory:

```
git clone git@github.com:mozilla-services/push-dev-dashboard.git
cd push-dev-dashboard
```

2. Create and activate a virtual environment (Can also use virtualenvwrapper):

```
virtualenv env
source env/bin/activate
```

3. Install requirements:

```
pip install -r requirements.txt
```

4. Source the `.env` file to set environment config vars (Can also use autoenv):

```
source .env
```

5. **'Migrate'_** DB tables

```
python manage.py migrate
```

6. Create a superuser:

```
python manage.py createsuperuser
```

2.1.3 Run it

1. Source the `.env` file to set environment config vars (Can also use autoenv):

```
source .env
```

2. Activate the virtual environment (Can also use virtualenvwrapper):

```
source env/bin/activate
```

3. Run it:

```
python manage.py runserver
```

2.1.4 Enable Firefox Accounts Auth

To enable Firefox Accounts authentication, you can use our local development OAuth client app.

1. Add a django-allauth social app for Firefox Accounts (Log in as the superuser account you created):
   - Provider: Firefox Accounts
   - Name: fxa
   - Client id: 7a4cd4ca0fb1b5c9
   - Secret key: c10059ba24e6715a1b6f2c80f1cc398fb6a39ca18bc7554e894b36ea85b88eeb
   - Sites: example.com -> Chosen sites
2. Log out of the admin account
3. Sign in with a Firefox Account at http://127.0.0.1:8000.

2.1.5 Run the Tests

1. Install test requirements:

```
pip install requirements-test.txt
```

2. Running the test suite:

```
python manage.py test
```

2.1.6 Working on Docs

Install doc requirements:

```
pip install -r requirements-docs.txt
```

Building the docs is easy:

```
cd docs
sphinx-build . html
```

Read the beautiful docs:

```
open html/index.html
```

2.1.7 What to work on

We have Issues.

2.2 Deployment

moz-dev-dash is designed with 12-factor app philosophy, so it runs easily on heroku, and deis. You can most easily deploy your changes to your own heroku app with heroku toolbelt.

2.2.1 Deploy your own (on Heroku)

1. Create a heroku remote. We suggest naming it moz-dev-dash-*username*:

```
heroku apps:create moz-dev-dash-username
```

2. Set the heroku app to use the "multi" buildpack:

```
heroku buildpacks:set https://github.com/ddollar/heroku-buildpack-multi.git
```

3. Push code to the heroku remote:

```
git push heroku master
```

4. Migrate DB tables on heroku:

```
heroku run python manage.py migrate
```

5. Create a superuser on heroku:

```
heroku run python manage.py createsuperuser
```

6. Open the new heroku app:

```
heroku open
```

2.2.2 Deploy your own (on Mozilla Deis)

To deploy the app to the Mozilla Cloud Ops Dev Deis cluster, you will need to request a dev IAM account from Mozilla Cloud Ops.

1. Install the deis client.
2. Register/login with the Mozilla Deis controller:

```
deis register http://deis.apps.dev.mozaws.net
deis login
```

3. Add your public key:

```
deis keys:add
```

4. Create the application:

```
deis create
```

5. Set the deis app to use the "multi" buildpack:

```
deis config:set BUILDPACK_URL=https://github.com/heroku/heroku-buildpack-multi
```

6. Push code to the deis remote:

```
git push deis master
```

7. Create an RDS Postgres instance in us-east-1 with default settings except:
   - DB Instance Class: db.t2.micro
   - Allocated Storag: 5 GB
   - VPC: vpc-9c2b0ef8
8. In the RDS Instance configuration details, click the "Security Groups". (Usually something like "rds-launch-wizard-N (sg-abcdef123)")
9. In the security group, under the "Inbound" tab, change the source to allow the deis cluster hosts:

```
10.21.0.0/16
```

10. Set the `DATABASE_URL` environment variable to match the RDS DB:

```
deis config:set DATABASE_URL=postgres://username:password@endpoint/dbname
```

11. Migrate DB tables on the new RDS instance:

```
deis run python manage.py migrate
```

12. Dock to app instance to create a superuser:

```
deisctl dock <app-name>
/app/.heroku/python/bin/python manage.py createsuperuser
```

13. Open the new deis app:

```
deis open
```

2.2.3 Enable Firefox Accounts Auth

To enable Firefox Account sign-ins on your own app, you will need to create your own Firefox Accounts OAuth Client for your app domain.

1. Go to register your own Firefox Accounts OAuth Client:
   - Client name: moz-dev-dash-username
   - Redirect URI: https://<app-domain>/accounts/fxa/login/callback/
   - Trusted Mozilla Client: **CHECKED**

   Be sure to copy the client secret - you can't see it again.
2. Go to https://<app-domain>/admin/socialaccount/socialapp/add/ to *Enable Firefox Accounts Auth* like a local machine; this time using your own new Firefox Accounts OAuth Client ID and Secret
3. Sign in at https://<app-domain>/ with a Firefox Account.