
Firenado Framework Documentation

Release 0.0.1.0

Flavio Garcia

October 09, 2016

1	User's guide	3
1.1	Introduction	3
1.2	Instalation	3
1.3	Usage	3
2	Configuration	5
3	Components	7
4	Release notes	9
4.1	What's new in Firenado 0.1.5.3	9
4.2	What's new in Firenado 0.1.5.2	9
4.3	What's new in Firenado 0.1.5.1	10
4.4	What's new in Firenado 0.1.5.0	10
4.5	What's new in Firenado 0.1.4	11
4.6	What's new in Firenado 0.1.3	11
4.7	What's new in Firenado 0.1.2	12
4.8	What's new in Firenado 0.1.1	12
4.9	What's new in Firenado 0.1.0	13
5	Indices and tables	15
5.1	Discussion and support	15

Firenado is a Python web framework that extends [Tornado](#) adding new features such as but not limited to application loose couple components, server side session layer, yaml based configuration files.

Contents:

1.1 Introduction

Firenado is a Python web framework that primarily extends the [Tornado](#) framework and runs over it's web server.

A Firenado application is organized in loose couple components and it is wired using yaml config files. The framework makes possible to develop components that can be shared between applications and distributed separately.

When you develop with Firenado you can use the our server side session layer based on files or redis, and you can develop a custom storage.

Other features are offered shipped at the framework core or via components.

1.2 Instalation

```
pip install firenado
```

1.3 Usage

Creating and running a new application:

```
firenado project init helloworld
cd helloworld
firenado app run
```

By default an application will be created with a redis based session and a redis data source defied and linked to the session.

Firenado don't install redispy so it is necessary to either install it or turn the session as file based. You can disable the session engine too.

To change the session type to file go to helloworld/conf/firenado.yml and change the session definition to:

```
# Session types could be:
# file or redis.
session:
  type: file
  enabled: true
  # Redis session handler configuration
  # data:
```

```
#   source: session
# File session handler related configuration
path: /tmp
```

If your helloworld project isn't on the python path just go helloworld/conf/firenado.yml and configure the application settings:

```
app:
  component: helloworld
  data:
  sources:
    # Set here references from sources defined on data.sources
    - session
  pythonpath: ..
  port: 8888
```

This web site and all documentation is licensed under Creative Commons 3.0.

Configuration

Firenado extends the Tornado web framework which uses ini files to help developers to configure their applications.

As the ini configuration from Tornado is pretty easy and straight forward to use, ini files(specially the one Tornado offers) just define key/value structures.

If someone wants to define hierarchical configuration structures it is necessary create indexes that represent the hierarchy(i.e. my.hierarchical.index) but the file will be read as key/value. Either the developer assumes the index is structured hierarchically or some extra development is needed to represent this data as should be in memory.

This is not a problem if the application configuration is simple but if the project requires lots of configurable parameters than you need a better option.

Firenado uses yaml files instead. TODO give the yaml benefits.

The app is bootstrapped using the app configuration file. This file will overload default parameters defined by the framework and system levels.

The framework config file will define the framework parts like:

- Firenado components offered by the framework
- Data connectors, used to connect to databases
- Log configuration
- Firenado cli management commands
- Session handlers and encoders

On the system config file level it is possible to define system level:

- Firenado components
- Data sources
- Custom data connectors
- Log configuration
- Custom cli management commands
- Custom Session handlers and encoders

The app config file will define the application component and tornado behaviour(ie. port or socket, number of processes forked, etc). It is also possible to configure on this level:

- Data sources used by the application
- If session is enabled and what session handler and encoder is being used by

the application - Override framework and system configuration

Components

Firenado is organized in components. In order to run a Firenado application it is necessary create a component and set it as the app component.

We can say that a Firenado application is a component.

Release notes

4.1 What's new in Firenado 0.1.5.3

4.1.1 October 09, 2016

We are pleased to announce the release of Firenado 0.1.5.3.

We added couple features to give more flexibility to configure the application and check if the request was sent by a mobile device.

The Tornado examples to handle Facebook and Twitter Oauth were added to the project and a Google one was created based on the Twitter one.

Here are the highlights:

New Features

- Create method to check if the handler is dealing with a mobile. [#161](#)
- Add static_url_prefix to the config file. [#162](#)
- Add settings configuration list to the app config. [#163](#)

Examples

- Create examples using facebook, twitter and google authentication. [#164](#)

4.2 What's new in Firenado 0.1.5.2

4.2.1 September 18, 2016

We are pleased to announce the release of Firenado 0.1.5.2.

This release add some features to secure cookies, prevent xsrf attacks and add login urls to the app.

Here are the highlights:

Features

- Add cookie_secret to the config core feature session tornadoweb. #123
- Add login url to the app config . #153
- Add xsrf_cookies to the app config. #154
- Add management task to generate random cookie secrets. #155

4.3 What's new in Firenado 0.1.5.1

4.3.1 September 10, 2016

We are pleased to announce the release of Firenado 0.1.5.1.

This release fixes some management tasks bugs and handles the sqlalchemy data source disconnect issue better.

Here are the highlights:

Bug Fixes

- Firenado project init is broken. #138
- Sub-command help is not being parsed correctly. #139
- Sqlalchemy session has errors after some time. #145

4.4 What's new in Firenado 0.1.5.0

4.4.1 September 01, 2016

We are pleased to announce the release of Firenado 0.1.5.0.

On this release the build was improved in order to provide extra requirements.

Right now it is possible to install redis and sqlalchemy using extra requirements with pip or on the requirements file.

Python 2.6 and 3.2 are not supported on this version to comply with Tornado 4.4.1.

Here are the highlights:

Refactory

- Depreciate python 2.6 and 3.2. #118
- Change application file from application.py to app.py. #120

New Features

- Create the static maps component. #57
- Add optional requirements to setup.py. #119

Bug Fixes

- Fix managment command help. [#13](#)
- Six is not being installed by setup.py. [#121](#)
- Sqlalchemy connection is missed when database service restarts or the connection is renewed. [#127](#)

4.5 What's new in Firenado 0.1.4

4.5.1 April 9, 2016

We are pleased to announce the release of Firenado 0.1.4.

On this release the focus was to get the framework refactored to be more “Pythonic”.

This version fixes couple bugs and introduce a way to set the application static_path via configuration.

Here are the highlights:

Refactoring

- Package refactoring to move code from `__init__.py`'s [#109](#)

New Features

- Added static_path to the config file [#104](#)

Bug Fixes

- A disabled session still needs a session type [#91](#)
- The service by decorator is broken [#107](#)
- The initialize method runs just if the component has a file defined [#103](#)

4.6 What's new in Firenado 0.1.3

4.6.1 January 31, 2016

We are pleased to announce the release of Firenado 0.1.3.

On this release yml config files will be used by default but yaml files will be considered also.

Services can be served by other services and couple more minor features were added to the framework.

No bug fixes were covered on this release.

Here are the highlights:

New Features

- Served_by decorator can be used by another service. [#91](#)
- Components can add ui_modules to an application. [#85](#)
- Tornado application debug mode added. [#84](#)
- Config files can be used with yml and yaml. Yml is default. [#83](#)
- Firenado was prepared to start with application types. Tornado is default. [#41](#)
- A Firenado Tornado application can be set to start listening on a unix socket. [#38](#)

4.7 What's new in Firenado 0.1.2

4.7.1 October 30, 2015

Hotfix release to solve a severe bug that was preventing firenado command to run.

Here are the highlights:

New Features

- Added root path to the assets component. [#77](#)

Bug Fixes

- Firenado doesn't run because core.management.tasks was not shipped on 0.1.1. [#76](#)

4.8 What's new in Firenado 0.1.1

4.8.1 October 28, 2015

We are pleased to announce the release of Firenado 0.1.1. Now we start to support Python 3 and have fixed some bugs fixed.

The project CI is being tracked by travis and tests are being added to the project.

Here are the highlights:

New Features

- Python 3 support
- Added installation capabilities to the framework.

Migration from iFlux to Firenado

- Security (It will be decided if this module still be needed on 0.1.2).

Bug Fixes

- Firenado requirements are not wired to the setup script. [#52](#)
- When Redis connection fails application exits without warning. [#58](#)
- Deadlock with the string generation on the util package. [#64](#)

4.9 What's new in Firenado 0.1.0

4.9.1 September 28, 2015

We are pleased to announce the release of Firenado 0.1. This was a complete rewrite from the iFlux framework and here are the highlights:

New Features

- Yaml based configuration files

Migration from iFlux to Firenado

- Data tools
- Modules
- Session
- Services

Migration changes

- Configuration
- Command line tool

Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)

5.1 Discussion and support

Report bugs on the [GitHub issue tracker](#).

Firenado is one of [Candango Open Source Group](#) initiatives. It is available under the [Apache License, Version 2.0](#).

This web site and all documentation is licensed under [Creative Commons 3.0](#).