Fusion Data Framework Documentation

Release 0.0.0

John Schmitt, David R. Smith, Kevin Tritz, Howard Yuh

October 06, 2015

CONTENTS

1 About FDF **1**

2 User Guide **2**

3 Developer Guide **3**

4 Usage Examples **4**

5 Package Reference **5**
- 5.1 Module factory.py . . . 5
- 5.2 Class Machine . . . 5
- 5.3 Class Shot . . . 6
- 5.4 Class Logbook . . . 6
- 5.5 Module fdf_signal.py . . . 6
- 5.6 Class Signal . . . 6
- 5.7 Module fdf_globals.py . . . 6

6 License **7**

7 Indices and tables **8**

Python Module Index **9**

Index **10**

CHAPTER

ONE

ABOUT FDF

Fusion Data Framework (FDF) is a data access, management, and visualization framework for magnetic fusion experiments.

Repository: https://github.com/Fusion-Data-Framework/fdf

HTML Documentation or PDF Documentation

Submit bugs or feature requests: https://github.com/Fusion-Data-Framework/fdf/issues

Created by:

- John Schmitt, Princeton Plasma Physics Lab
- David R. Smith, U. Wisconsin-Madison
- Kevin Tritz, The Johns Hopkins U.
- Howard Yuh, Nova Photonics

To contribute to the FDF project, please contact John, David, Kevin, or Howard.

CHAPTER

TWO

USER GUIDE

This guide is for people who want to use FDF on the PPPL Linux cluster. If you wish to contribute to the FDF code repository (https://github.com/Fusion-Data-Framework/fdf), see developer-guide.

HTML documentation is available here: http://fusion-data-framework.github.io/fdf/

To use FDF on the PPPL Linux cluster (`portal.pppl.gov`), load the module nstx/fdf (you may need to unload other nstx modules):

```
[sunfire06:~] % module load nstx/fdf
[sunfire06:~] % module list
Currently Loaded Modulefiles:
1) torque/2.5.2      5) idl/8.2           9) java/v1.6
2) moab/5.4.0        6) nstx/treedefs    10) nstx/mdsplus5
3) ppplcluster/1.1   7) nstx/epics       11) nstx/fdf
4) freetds/0.91      8) nstx/idldirs
```

Verify that python points to `/p/fdf/anaconda/bin/python`:

```
[sunfire06:~] % which python
/p/fdf/anaconda/bin/python
```

If python does not point to `/p/fdf/anaconda/bin/python`, then PATH contains to a different python distribution ahead of `/p/fdf/anaconda/bin`. In this case, you need to modify PATH so `/p/fdf/anaconda/bin` is the first python distribution in PATH.

Finally, you can launch python and import the FDF package:

```
[sunfire06:~] % python
Python 2.7.10 |Anaconda 2.3.0 (64-bit)| (default, Sep 15 2015, 14:50:01)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
>>> import fdf
>>>
```

CHAPTER

THREE

DEVELOPER GUIDE

CHAPTER

FOUR

USAGE EXAMPLES

CHAPTER

FIVE

PACKAGE REFERENCE

5.1 Module factory.py

Root module for the FDF package.

Classes

- Machine - root class for the FDF package
- Shot - shot container class
- Logbook - logbook connection class
- Container - diagnostic container class
- Node - mdsplus signal node class
- FdfError - error class for FDF package

Usage:

```
>>> import fdf
>>> nstx = fdf.Machine('nstx')
>>> nstx.s140000.logbook()
>>> nstx.addshots(xp=1048)
>>> nstx.s140000.mpts.plot()
```

5.2 Class Machine

class factory.**Machine**(*name='nstx'*, *shotlist=[]*, *xp=[]*, *date=[]*)

Factory root class that contains shot objects and MDS access methods.

Basic class initialization is performed as follows: >>>nstx = Machine(name='nstx')

the Machine class contains a model shot object: nstx.s0

shot data can be accessed directly through the Machine class: >>> nstx.s141398 >>> nstx.s141399

alternatively, a list of shot #'s may be provided during initialization: >>>nstx = Machine(name='nstx', shotlist=[141398, 141399])

or added later using the addshot method: >>>nstx.addshot([141398, 141399])

5.3 Class Shot

class factory.**Shot**(*shot*, *root=None*, *parent=None*)

5.4 Class Logbook

class factory.**Logbook**(*name='nstx'*, *root=None*)

5.5 Module fdf_signal.py

fdf-signals.py - module containing Signal class

Classes

- Signal - signal class for data objects

Created on Tue Jun 23 2015

@author: hyuh

5.6 Class Signal

class fdf_signal.**Signal**(***kwargs*)

sig=fdf.Signal(signal_ndarray, units='m/s', axes=['radius','time'], axes_values=[ax1_1Darray, ax2_1Darray], axes_units=['s','cm']

e.g.: mds.Signal(np.arange((20*10)).reshape((10,20)), units='keV', axes=['radius','time'], axes_values=[100+np.arange(10)*5, np.arange(20)*0.1], axes_units=['s','cm'])

or an empty signal: s=mds.Signal() default axes order=[time, space] sig=fdf.Signal(units='m/s', axes=['radius','time'], axes_values=[radiusSignal, timeSignal])

5.7 Module fdf_globals.py

fdf_globals.py contains package-level constants

Created on Thu Jun 18 11:18:16 2015

@author: ktritz

CHAPTER

SIX

LICENSE

The MIT License (MIT)

Copyright (c) 2015 David R. Smith, Kevin Tritz, Howard Yuh

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search

f

factory, 5
fdf_globals, 6
fdf_signal, 6

F

factory (module), 5
fdf_globals (module), 6
fdf_signal (module), 6

L

Logbook (class in factory), 6

M

Machine (class in factory), 5

S

Shot (class in factory), 6
Signal (class in fdf_signal), 6