public openplanets / fido

- Code
- Network
- Pull Requests 0
- Issues 10
- Wiki
- Graphs

Python

1. Python 100%

Format Identification for Digital Objects (FIDO) is a Python command-line tool to identify the file formats of digital objects. It is designed for simple integration into automated work-flows. — Read more

http://www.openplanetsfoundation.org/software/fido

- ZIP

- HTTP
- Git Read-Only

https://github.com/openplanet

Read-Only access

- Tags 14
- Downloads 0

- *branch:* master
  Switch branches/tags
  Filter branches/tags
  - Branches
  - Tags

  Nothing to show

  master

- Files
- Commits
- Branches 1

Latest commit to the **master** branch

BUGFIXES for fido.py, update_signatures.py and prepare.py

commit 297e8022dd

techmaurice authored 24 days ago

fido /

name	age	history message

fido	24 days ago	BUGFIXES for fido.py, update_signatures.py and prepare.py [techmaurice]
.gitignore	a year ago	Substantial simplifications and changes. (1) moved run.py to fido.py.… [adamfarquhar]
LICENSE.txt	a year ago	Folder with jar and jar build files [techmaurice]
README.in	5 months ago	Version 1.0 updated readme [techmaurice]
README.txt	5 months ago	Version 1.0 updated readme [techmaurice]
todo.txt	9 months ago	Fixed fido.py and prepare.py; FIDO supports all PRONOM sigs now [techmaurice]

README.txt

```
usage: fido.py [-h] [-v] [-q] [-recurse] [-zip] [-nocontainer] [-input INPUT]
               [-useformats INCLUDEPUIDS] [-nouseformats EXCLUDEPUIDS]
               [-matchprintf FORMATSTRING] [-nomatchprintf FORMATSTRING]
               [-bufsize BUFSIZE] [-container_bufsize CONTAINER_BUFSIZE]
               [-loadformats XML1,...,XMLn] [-confdir CONFDIR]
               [FILE [FILE ...]]

Format Identification for Digital Objects (fido). FIDO is a command-line tool
to identify the file formats of digital objects. It is designed for simple
integration into automated work-flows.

positional arguments:
  FILE                  files to check. If the file is -, then read content
                        from stdin. In this case, python must be invoked with
                        -u or it may convert the line terminators.

optional arguments:
  -h, --help            show this help message and exit
  -v                    show version information
  -q                    run (more) quietly
  -recurse              recurse into subdirectories
  -zip                  recurse into zip and tar files
  -nocontainer          disable deep scan of container documents, increases
                        speed but may reduce accuracy with big files
  -input INPUT          file containing a list of files to check, one per
                        line. - means stdin
  -useformats INCLUDEPUIDS
                        comma separated string of formats to use in
                        identification
  -nouseformats EXCLUDEPUIDS
                        comma separated string of formats not to use in
                        identification
  -matchprintf FORMATSTRING
                        format string (Python style) to use on match. See
                        nomatchprintf, README.txt.
  -nomatchprintf FORMATSTRING
                        format string (Python style) to use if no match. See
                        README.txt
  -bufsize BUFSIZE      size (in bytes) of the buffer to match against
                        (default=131072 bytes)
  -container_bufsize CONTAINER_BUFSIZE
                        size (in bytes) of the buffer to match against
                        (default=524288 bytes)
  -loadformats XML1,...,XMLn
                        comma separated string of XML format files to add.
  -confdir CONFDIR      configuration directory to load_fido_xml, for example,
                        the format specifications from.

Open Planets Foundation (http://www.openplanetsfoundation.org)
See License.txt for license information.
Download from: http://github.com/openplanets/fido/downloads
Author: Adam Farquhar, 2010
Maintainer: Maurice de Rooij (OPF/NANETH), 2011, 2012
FIDO uses the UK National Archives (TNA) PRONOM File Format and Container descriptions.
PRONOM is available from http://www.nationalarchives.gov.uk/pronom/

Installation
------------

Any platform
1. Download the latest zip release from http://github.com/openplanets/fido/downloads
   (or use the big Downloads button on http://github.com/openplanets/fido)
2. Unzip into some directory
3. Open a command shell, cd to the directory that you placed the zip contents into and cd into folder 'fido'
4. You should now be able to see the help text:
   python fido.py -h
5. Before identifying files with FIDO for the first time, please update signatures first
   using the 'update_signatures.py' script (see below for instructions).

Updating signatures
-------------------

To update FIDO with the latest PRONOM file format definitions, run:
   python update_signatures.py
This is an interactive CLI script which downloads the latest PRONOM signature file and signatures. Please note that it can take a while tc

If you are having trouble running the script due to firewall restrictions, see OPF wiki: http://wiki.opf-labs.org/display/PT/Command+Line+

Please note that this WILL NOT update the container signature file located in the 'conf' folder.
The reason for this that the PRONOM container signature file contains special types
of sequences which need to be tested before FIDO can use them. If there is an update available
for the PRONOM container signature file it will show up in a next commit.

Dependencies
```

```
------------

FIDO 1.0 and later will run on Python 2.7 with no other dependencies.

Format Definitions
------------------

By default, FIDO loads format information from two files conf/formats.xml
and conf/format_extensions.xml. Addition format files can be specified using
the -loadformats command line argument.  They should use the same syntax as
conf/format_extensions.xml. If more than one format file needs to be specified,
then they should be comma separated as with the -formats argument.

Output
------

Output is controlled with the two parameters matchprintf and nomatchprintf.
Each is a string that may contain formating information.  They have access to
an object called info with the following fields:

printmatch: info.version (file format version X), info.alias (format also called X), info.apple_uti (Apple Uniform Type Identifier), info.

printnomatch: info.count (file N)

The defaults for FIDO 1.0 are:
  printmatch:
    "OK,%(info.time)s,%(info.puid)s,%(info.formatname)s,%(info.signaturename)s,%(info.filesize)s,\"%(info.filename)s\",\"%(info.mimetype)s

  printnomatch:
    "KO,%(info.time)s,,,,%(info.filesize)s,\"%(info.filename)s\",,\"%(info.matchtype)s\"\n"

It can be useful to provide an empty string for either, for example to ignore all failed matches, or all successful ones (see examples bel
Note that a newline needs to be added to the end of the string using \n.

Matchttypes
-----------

FIDO returns the following matchtypes:
- fail:      the object could not be identified with signature or file extension
- extension: the object could only be identified by file extension
- signature: the object has been identified with (a) PRONOM signature(s)
- container: the object has been idenfified with (a) PRONOM container signature(s)

(In some cases multiple results are returned.)

Examples running FIDO
---------------------

Identify all files in the current directory and below, sending output
into file-info.csv
   python fido.py -recurse . > file-info.csv

Do the same as above, but also look inside of zip or tar files:
   python fido.py -recurse -zip . > file-info.csv

Take input from a list of files:
Linux:
   ls > files.txt
   python fido.py -input files.txt
Windows:
   dir /b > files.txt
   python fido.py -input files.txt

Take input from a pipe:
Linux:
   find . -type f | python fido.py -input -
Windows:
   dir /b | python fido.py -input -

Only show files that could not be identified.
   python fido.py -matchprintf "" .

Only show files that could be identified.
   python fido.py -nomatchprintf "" .

Deep scan of container objects
------------------------------

By default, when FIDO detects that a file is a container (compound) object,
it will start a deep (complete) scan of the file using the PRONOM container signatures.
When identifying big files, this behaviour can cause FIDO to slow down sigificantly.
You can disable deep scanning by invoking FIDO with the '-nocontainer' argument.
While disabling deep scan speeds up identification, it may reduce accuracy.

At the moment (version 1.0) FIDO is not yet able to perform scanning containers which are
passed through STDIN. A workaround would be to save the stream to a temporary file and have
FIDO identify this file.

License information
-------------------

See the file "LICENSE.txt" for information on the history of this
software, terms & conditions for usage, and a DISCLAIMER OF ALL
```

```
WARRANTIES...
```

GitHub Links

GitHub

- About
- Blog
- Features
- Contact & Support
- Training
- GitHub Enterprise
- Site Status

Clients

- GitHub for Mac
- GitHub for Windows
- GitHub for Eclipse
- GitHub Mobile Apps

Tools

- Gauges: Web analytics
- Speaker Deck: Presentations
- Gist: Code snippets

Extras

- Job Board
- GitHub Shop
- The Octodex

Documentation

- GitHub Help
- Developer API
- GitHub Flavored Markdown
- GitHub Pages

- Terms of Service
- Privacy
- Security

Powered by the Dedicated Servers and Cloud Computing of Rackspace Hosting®

Markdown Cheat Sheet

Format Text

Headers

```
# This is an <h1> tag
## This is an <h2> tag
###### This is an <h6> tag
```

Text styles

```
*This text will be italic*
_This will also be italic_
**This text will be bold**
__This will also be bold__

*You **can** combine them*
```

Lists

Unordered

```
* Item 1
* Item 2
  * Item 2a
  * Item 2b
```

Ordered

```
1. Item 1
2. Item 2
3. Item 3
   * Item 3a
   * Item 3b
```

Miscellaneous

Images

```
![GitHub Logo](/images/logo.png)
Format: ![Alt Text](url)
```

Links

```
http://github.com - automatic!
[GitHub](http://github.com)
```

Blockquotes

```
As Kanye West said:

> We're living the future so
> the present is our past.
```

Code Examples in Markdown

Syntax highlighting with GFM

```
```javascript
function fancyAlert(arg) {
  if(arg) {
    $.facebox({div:'#foo'})
  }
}
```
```

Or, indent your code 4 spaces

```
Here is a Python code example
without syntax highlighting:

    def foo:
      if not bar:
        return true
```

Inline code for comments

```
I think you should use an
`<addr>` element here instead.
```

Something went wrong with that request. Please try again. Dismiss

Looking for the GitHub logo?

- **GitHub Logo**

Download

- **The Octocat**

Download