4ch Documentation

Release 0.2.0

plausibility

December 19, 2016

Contents

fourch (stylised as 4ch) is an easy-to-implement Python wrapper for 4chan's JSON API, as provided by moot.

It uses the documentation of the 4chan API located at: https://github.com/4chan/4chan-API

This is based off of the API last updated Aug 12, 2014. (4chan-API commit: 1b2bc7858afc555127b8911b4d760480769872a9)

CHAPTER 1

Contents

1.1 Usage

4ch is easy to use and implement, here are some examples. For more thorough documentation, visit the Automated Documentation.

1.1.1 Neat examples

Here are some examples to show you what sort of stuff you can do with 4ch.

List all image URLs

You can easily access all the images in a `thread` with the `t.images` property.

```
## images.py
import sys
import fourch

def main():
    # Display usage and help if they've mucked up.
    if len(sys.argv) is not 3:
        print "usage: python {0} <board> <thread>".format(sys.argv[0])
        return

    b = fourch.board(sys.argv[1])
    t = b.thread(sys.argv[2])
    for i in t.images:
        print i

if __name__ == "__main__":
    main()
```

This can be utilized to download all the images in a thread like so:

```
$ python images.py g 123456789 | xargs wget
```

This will simply pass all of the image URLs to wget, downloading them.

Get thread meta-data

This will simply list information about a given thread.

```
## thread_data.py
import sys
import fourch

def main():
    if len(sys.argv) is not 3:
        print "usage: python {0} <board> <thread>".format(sys.argv[0])
        return

    b = fourch.board(sys.argv[1])
    t = b.thread(sys.argv[2])

    print 'Thread:\t\t', t
    print 'Sticky:\t\t', t.sticky
    print 'Closed:\t\t', t.closed
    print 'OP:\t\t', t.op
    print 'Post #:\t\t', t.op.number
    print 'Post time:\t', t.op.now
    print 'Timestamp:\t', t.op.timestamp
    print 'Filemd5:\t', t.op.file.md5
    print 'Filemd5 base64:\t', t.op.file.md5b64
    print 'File url:\t', t.op.file.url
    print 'Subject:\t', t.op.subject
    print 'Comment (text):\n', t.op.comment_text
    print 'Replies:\t', len(t.replies)

if __name__ == "__main__":
    main()
```

View all threads in a page

Note: If you set `update_each` to `True`, this method can be costly – bandwidth wise – as well as slow, as it has to GET each thread.

You can easily view all the threads on any given page of a board with the `fourch.board.page()` method. This method pulls in all the threads in a given page.

If you want to get all the threads, as well as their replies (e.g., a full thread, not just what's shown), you will have to set `update_each` to `True` in the method call: `b.page(page=0, update_each=True)`

```
## all_threads.py
import sys
import fourch

def main():
    if len(sys.argv) is not 3:
        print "usage: python {0} <board> <page>".format(sys.argv[0])
        return

    b = fourch.board(sys.argv[1])
    thr = b.page(page=sys.argv[2])
```

```
    for t in thr:
        print t.op.url
        print t.op.comment_text, "\n"

if __name__ == "__main__":
    main()
```

This will simply pull in all the threads from whatever page you specify, and print out the URL, as well as the comment.

1.2 Automated Documentation

This documentation is automatically built from the latest version of 4ch, located at the git repo.

fourch.**board**
alias of *fourch.board*

fourch.**thread**
alias of *fourch.thread*

fourch.**reply**
alias of *fourch.reply*

f

B

F

R

T