manga-dl

Release 0.1.6

Jun 06, 2018

CONTENTS

Contents:

CHAPTER

ONE

MANGA_DL PACKAGE

1.1 Subpackages

1.1.1 manga_dl.entities package

Submodules

manga_dl.entities.MangaChapter module

class manga_dl.entities.MangaChapter.**MangaChapter**(*chapter_number: float, pages: List[manga_dl.entities.MangaPage.MangaPage]*)

Bases: object

Class that models a Chapter of the manga

Contains scrapers links to the individual chapter pages

__init__(*chapter_number: float, pages: List[manga_dl.entities.MangaPage.MangaPage]*) → None
Initializes a Manga chapter with the contained pages.

Parameters

- **chapter_number** – The chapter number of this chapter
- **pages** – the chapter pages as list of MangaPage objects

Returns None

chapter_name = ''
The chapter's name. Is normally generated by the get_chapter_name method

chapter_number = -1.0
The chapter's chapter number Is a float to allow chapters like 5.5 or the like

get_chapter_name() → str
Generates a Chapter name for this chapter based on the chapter number, or returns the chapter name previously set by the set_chapter_name method.

Returns the chapter name as a string

get_pages() → List[manga_dl.entities.MangaPage.MangaPage]

Returns a list of manga pages belonging to this chapter

pages = []
The individual pages of the chapter

set_chapter_name(*name: str*) → None
Sets the chapter's name

Parameters **name** – the name of the chapter

Returns None

manga_dl.entities.MangaPage module

class manga_dl.entities.MangaPage.**MangaPage**(*page_number: int*, *image_url: str*)
Bases: object

Class that models a page in a Manga chapter

__init__(*page_number: int*, *image_url: str*) → None
Initializes the Manga Page with a page number and an image URL

Parameters

- **page_number** – the page number
- **image_url** – the image URL

Returns None

get_page_name() → str
Generates a Page name for this page based on the page number, or returns the page name previously set by the set_page_name method.

Returns the page name as a string

image_url = ''
The URL to the page's image

page_name = ''
The page's name. Is normally generated by the get_page_name method

page_number = -1
The page's page number

set_page_name(*name: str*) → None
Sets the page's name

Parameters **name** – the name of the page

Returns None

manga_dl.entities.MangaSeries module

exception manga_dl.entities.MangaSeries.**MangaScraperNotFoundError**
Bases: Exception

Exception raised when no applicable manga scraper was found

class manga_dl.entities.MangaSeries.**MangaSeries**(*url: str*, *root_directory: str*)
Bases: object

Class that models a Manga series. It is the entry point for all operations related to downloading, repairing and zipping manga series.

It offers an automatic scraper detection system that tries to find a fitting scraper for the URL provided

__init__ (*url: str*, *root_directory: str*) → None
Initializes the Manga series

Parameters

- **url** – the URL for where to look for volumes to scrapers
- **root_directory** – the directory in which the local copy of the series resides in

Raises MangaScraperNotFound, if no applicable manga scraper was found

static download_file (*options: Tuple[str, str, bool, bool, bool, bool]*) → None
Downloads a file, can also be used to repair previously downloaded files. Can be run in parallel using the multiprocessing Pool class. This limits the function to a single parameter, a Tuple in this case

Parameters **options** – Tuple containing the following parameters: url: the file's URL destination: the local destination for the file overwrite_existing: flag that enables overwriting

existing files

repair: flag that can be set to enable repair mode verbose: Sets the verbosity flag dry_run: Sets the dry run flag

Returns None

download_manga (*update: bool = False*, *repair: bool = False*)
Starts downloading the manga series

Parameters

- **update** – flag to set an update process, i.e. only downloads files that don't exist
- **repair** – flag to set a repair process, i.e. updates + checks if files are OK

Returns None

dry_run = False
Flag that can be set to disable any changes to the system, i.e. a dry run akin to the rsync dry run flag '-n'

max_threads = 1
Defines the maximum number of concurrent threads while scraping

repair () → None
Updates the current directory with volumes and chapters that do not exist yet. While doing so, every file is checked for consistency and replaced if needed.

Returns None

root_directory = ''
The root directory of the downloaded manga

scrape (*skip_existing_chapters: bool = False*) → None
Finds a list of all volumes using the scraper found in the __init__ method.

Parameters **skip_existing_chapters** – Can be set to skip existing chapters

Returns None

scraper = None
The scraper used to find the volumes belonging to the series

set_dry_run (*dry_run: bool = True*) → None
Sets the dry_run flag

Parameters **dry_run** – the new value of the dry_run flag, defaults to True

Returns None

set_maximum_thread_amount(*max_threads: int*) → None
Sets the maximum amount of threads to be used

Parameters **max_threads** – the new thread maximum

Returns None

set_verbose(*verbose: bool = True*) → None
Sets the verbosity flag

Parameters **verbose** – the new value of the verbosity flag, defaults to True

Returns None

update() → None
Updates the current directory with volumes and chapters that do not exist yet.

Returns None

url = ''
The manga series' URL

verbose = False
Flag that can be set to enable console output

volumes = []
List of volumes of the series

zip(*zip_volumes: bool = False*, *zip_chapters: bool = False*)
Zips parts of the series together to enable reading in some manga readers, like ComicRack for android

Parameters

- **zip_volumes** – flag to enable zipping volumes
- **zip_chapters** – flag to enable zipping chapters

Returns None

zip_all() → None
Zips the series by Volume and then by chapter

Returns None

zip_chapters() → None
Zips the series by chapter

Returns None

zip_volumes() → None
Zips the series by volume

Returns None

manga_dl.entities.MangaVolume module

class manga_dl.entities.MangaVolume.**MangaVolume**(*volume_number: int, chapters: List[manga_dl.entities.MangaChapter.MangaChapter]*)

Bases: object

Class that models a Manga chapter.

Contains a list of chapters

__init__(*volume_number: int, chapters: List[manga_dl.entities.MangaChapter.MangaChapter]*) → None
Initializes a Manga volume with a list of chapters

Parameters

- **volume_number** – The volume number of this volume
- **chapters** – the chapters belonging to this volume

Returns None

chapters = []
List of Manga chapters in this volume

get_chapters() → List[manga_dl.entities.MangaChapter.MangaChapter]

Returns a list of manga chapters belonging to this volume

get_volume_name() → str
Generates a Volume name for this volume based on the volume number, or returns the volume name previously set by the set_volume_name method.

Returns the volume name as a string

set_volume_name(*name: str*) → None
Sets the volume's name

Parameters **name** – the name of the volume

Returns None

volume_name = ''
The volume's name. Is normally generated by the get_volume_name method

volume_number = -1
The volume's Volume Number

Module contents

1.1.2 manga_dl.scrapers package

Submodules

manga_dl.scrapers.GenericMangaScraper module

class manga_dl.scrapers.GenericMangaScraper.**GenericMangaScraper**
Bases: object

Class that models how a Manga Scraper should operate

static get_series_name(*manga_url: str*) → str
Parses the URL to determine the series name

Parameters **manga_url** – The URL to parse

Returns The series name

static scrape_volumes_from_url(*manga_url: str, manga_directory: str, skip_existing_chapters: bool = False, max_threads: int = 1, verbose: bool = False*) → List[manga_dl.entities.MangaVolume.MangaVolume]
Scrapes a given URL

Parameters

- **manga_url** – the given URL to scrape
- **manga_directory** – the manga directory, which can be used to skip existing chapters
- **skip_existing_chapters** – Flag that can be set to skip existing chapters, thereby increasing scraping speed
- **max_threads** – the maximum numbers of threads to use
- **verbose** – Sets the verbosity flag. Defaults to no output

Returns a list of volumes, which should also contain chapters

static url_match(*manga_url: str*) → bool
Checks if a URL matches the pattern expected by the scraper

Parameters **manga_url** – the URL to check

Returns True if it matches, False otherwise

manga_dl.scrapers.MangaFoxScraper module

class manga_dl.scrapers.MangaFoxScraper.**MangaFoxScraper**
Bases: *manga_dl.scrapers.GenericMangaScraper.GenericMangaScraper*

Class that models how a Manga Scraper should operate

static get_series_name(*manga_url: str*) → str
Returns the very end of the URL, as this is the name of the series

Parameters **manga_url** – The URL to parse

Returns The series name

static parse_page(*options: Tuple[int, bool, str]*) → manga_dl.entities.MangaPage.MangaPage
Parses a single page of a chapter. Can be run in parallel using multiprocessing.Pool, which is the reason why the arguments are all passed via a single Tuple, due to the limitations of Pool.map()

Parameters **options** – the options for the page to parse: image_number: The image number of the image page

to parse

verbose: Enabling or disabling verbose output chapter_base_url: The base URL of the chapter

Returns the scraped manga page object

static scrape_volumes_from_url(*manga_url: str*, *manga_directory: str*, *skip_existing_chapters: bool = False*, *max_threads: int = 1*, *verbose: bool = False*) → List[manga_dl.entities.MangaVolume.MangaVolume]
Scrapes a given URL from mangafox.me

Parameters

- **manga_url** – the given URL to scrape
- **manga_directory** – the manga directory, which can be used to skip existing chapters
- **skip_existing_chapters** – Flag that can be set to skip existing chapters, thereby increasing scraping speed

- **max_threads** – The maximum amount of threads that can be used
- **verbose** – Sets the verbosity flag. Defaults to no output

Returns a list of volumes, which should also contain chapters

static url_match(*manga_url: str*) → bool
Checks if a URL matches the pattern expected by the scraper

Parameters **manga_url** – the URL to check

Returns True if it matches, False otherwise

manga_dl.scrapers.MangaScraperManager module

class manga_dl.scrapers.MangaScraperManager.**MangaScraperManager**
Bases: object

Class that acts as a negotiator for the various manga scrapers

static get_scraper_for(*manga_url: str*) → type
Returns the correct scraper for a specified manga URL :param manga_url: the URL of the Manga series :return: The correct scraper, or None if none was found

static get_series_name_from_url(*manga_url: str*) → str
Tryes to figure out the name of a manga series from its URL

Parameters **manga_url** – The URL to check

Returns The series name, or an emtpy string if no applicable parser exists

scrapers = [<class 'manga_dl.scrapers.MangaFoxScraper.MangaFoxScraper'>]
A list of scrapers that are implemented

Module contents

1.2 Submodules

1.3 manga_dl.main module

manga_dl.main.**main**() → None
Parses CLI arguments and starts the program

Returns None

1.4 Module contents

CHAPTER

TWO

MANGA_DL

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

m

INDEX

Symbols

C

D

G

I

M

P