gtrending

Release 0.4.0

hedyhli

Jan 20, 2023

CONTENTS:

Fetch repositories and developers from GitHub Trending.

Here's a simple example, get the trending python projects and display their full names:

```
from gtrending import fetch_repos


repos = fetch_repos(language="python")  # Returns a dictionary

for repo in repos:
    print(repo["fullname"])  # "user/repo" for each repo
```

CHAPTER

ONE

INSTALLATION

Install gtrending on PyPI using your favorite package manage. For example:

```
pip3 install gtrending
```

CHAPTER

TWO

CONTRIBUTE

- Issue Tracker: https://github.com/hedyhli/gtrending/issues
- Source Code: https://github.com/hedyhli/gtrending

CHAPTER

THREE

SUPPORT

If you are having issues, please open an issue on the github issue tracker as linked above.

CHAPTER

FOUR

LICENSE

The project is licensed under the MIT license.

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

5.1 Fetch Module

The `gtrending.fetch` module includes functions for fetching trending repositories and developers from GitHub Trending.

5.1.1 Trending Repositories

The `fetch_repos()` function can be used for fetching trending repositories on GitHub.

A **list of dictionaries** containing metadata about the trending repositories is returned.

Parameters

- `language` - the programming language to search for (eg: python)
- `spoken_language_code` - the code of the spoken language to search for (eg: en)
- `since` - the date range for the search period. Either one of: **daily**, **weekly**, and **monthly**

Note: All parameters are case-insensitive

The default arguments are:

- `language = ""` (all languages included)
- `spoken_language_code = ""` (all spoken languages included)
- `since = "daily"` (trending today)

The following are all valid examples:

```
# Any language, any spoken language, trending today
fetch_repos()

# JavaScript, any spoken language, trending today
```

(continues on next page)

(continued from previous page)

```
fetch_repos("javascript")
# Chinese, any language, trending today
fetch_repos(spoken_language_code="zh")
# Any language, any spoken language, trending this month
fetch_repos(since="monthly")

# JavaScript, Chinese, trending this month
fetch_repos("javascript", "zh", "monthly")
```

Example return values

```
>>> fetch_repos()
[
  {
    "author": "ShoufaChen",
    "name": "DiffusionDet",
    "avatar": "https://github.com/ShoufaChen.png",
    "description": "PyTorch implementation of DiffusionDet (https://arxiv.org/abs/2211.
↪09788)",
    "url": "https://github.com/ShoufaChen/DiffusionDet",
    "language": "Python",
    "languageColor": "#3572A5",
    "stars": 610,
    "forks": 19,
    "currentPeriodStars": 121,
    "builtBy": [
      {
        "username": "ShoufaChen",
        "href": "https://github.com/ShoufaChen",
        "avatar": "https://avatars.githubusercontent.com/u/28682908"
      }
    ],
    "fullname": "ShoufaChen/DiffusionDet"
  },
  {
    "author": "FlagAI-Open",
    "name": "FlagAI",
    "avatar": "https://github.com/FlagAI-Open.png",
    "description": "FlagAI (Fast LArge-scale General AI models) is a fast, easy-to-use␣
↪and extensible t
  oolkit for large-scale model.",
    "url": "https://github.com/FlagAI-Open/FlagAI",
    "language": "Python",
    "languageColor": "#3572A5",
    "stars": 323,
    "forks": 42,
    "currentPeriodStars": 38,
    "builtBy": [
      {
        "username": "marscrazy",
```

(continues on next page)

(continued from previous page)


```
        "href": "https://github.com/marscrazy",
        "avatar": "https://avatars.githubusercontent.com/u/4392184"
      },
      {
        "username": "Anhforth",
        "href": "https://github.com/Anhforth",
        "avatar": "https://avatars.githubusercontent.com/u/94831503"
      },
      {
        "username": "920232796",
        "href": "https://github.com/920232796",
        "avatar": "https://avatars.githubusercontent.com/u/32668889"
      },
      {
        "username": "ZhaodongYan1",
        "href": "https://github.com/ZhaodongYan1",
        "avatar": "https://avatars.githubusercontent.com/u/26128888"
      },
      {
        "username": "BAAI-OpenPlatform",
        "href": "https://github.com/BAAI-OpenPlatform",
        "avatar": "https://avatars.githubusercontent.com/u/107522723"
      }
    ],
    "fullname": "FlagAI-Open/FlagAI"
  },
]
```


The key `currentPeriodStars` is the increase of stars for the current trending period - as specified by the `since` argument.

Argument validation

Parameters `language` and `spoken_language_code` only accept values in the correct format. **ValueError** is thrown for invalid values.

Both parameters are case-insensitive.

Language

Valid values must be one of the list of languages, returned from `languages_params()`.

To check if a value is valid before passing to `fetch_repos()`, use `check_language(language)`:

```
>>> check_language("python")
True
>>> check_language("Ruby")
True
>>> check_language("TeaScript")  # Does not exist
False
>>> check_language("")
False
```

See the *ParamUtils module* for usage details on other parameter validation functions.

Spoken language code

Valid values must be one of the list of spoken language **codes**, returned from `spoken_languages_codes()`.

Note: This is the spoken language **code**, as in "en"/"es"/"ko", and not the spoken language name itself (such as "english").

To check if a value is valid before passing to `fetch_repos()`, use `check_spoken_language_code(code)`:

```
>>> check_spoken_language_code("el")
True
>>> check_spoken_language_code("ZH")
True
>>> check_spoken_language_code("ZZ")  # Does not exist
False
>>> check_spoken_language_code("")
False
```

See the *ParamUtils module* for functions that validate the spoken language name, and convert between the name and the code.

5.1.2 Trending Developers

The `fetch_developers()` function can be used for fetching trending developers.

A **list of dictionaries** containing metadata about the trending developers as well as the repository they are trending for is returned.

Parameters

- `language` - the programming language to search for (eg: python)
- `since` - the date range for the search period. Either one of: **daily**, **weekly**, or **monthly**

Note: All parameters are case-insensitive

The default arguments are:

- `language = ""` (all languages included)
- `since = "daily"`

The following are all valid examples:

```
# Any language, trending today
fetch_developers()

# Python, trending today
fetch_developers("python")
```

(continues on next page)

(continued from previous page)

```
# Any language, trending this week
fetch_developers(since="weekly")

# Rust, trending this month
fetch_developers("rust", "monthly")
```

Example return values

```
>>> fetch_developers()
[
  {
    "username": "stedolan",
    "name": "Stephen Dolan",
    "url": "https://github.com/stedolan",
    "sponsorUrl": null,
    "avatar": "https://avatars.githubusercontent.com/u/79765",
    "repo": {
      "name": "jq",
      "description": "Command-line JSON processor",
      "url": "https://github.com/stedolan/jq"
    }
  },
  {
    "username": "wcandillon",
    "name": "William Candillon",
    "url": "https://github.com/wcandillon",
    "sponsorUrl": null,
    "avatar": "https://avatars.githubusercontent.com/u/306134",
    "repo": {
      "name": "can-it-be-done-in-react-native",
      "description": "\u269b\ufe0f \ud83d\udcfa Projects from the \u201cCan it be done␣
↪in React Native?\u201d YouTube series",
      "url": "https://github.com/wcandillon/can-it-be-done-in-react-native"
    }
  },
  ...
]
```

Argument validation

The `language` parameter raises **ValueError** for invalid language values or non-existent languages.

Language

Valid values must be one of the list of languages, returned from `languages_params()`.

To check if a value is valid before passing to `fetch_developers()`, use `check_language(language)`

```
>>> check_language("python")
True
>>> check_language("Ruby")
True
>>> check_language("TeaScript")  # Does not exist
False
>>> check_language("")
False
```

See the *ParamUtils module* for usage details on parameter validation functions such as conversion between language argument formats, and validating language arguments.

Optional parameters to specify the programming language, spoken language, and the date range can be used.

5.1.3 Parameters

All parameters are case-insensitive.

- **Programming language**: `language` (str), eg: "python", "vimscript".

  This specifies the primary language that repositories are written in, when searching trending repositories, and the primary language a developer's trending repository is written in, when searching for trending developers

- **Spoken language**: `spoken_language_code` (str), eg: "en", "es".

  This specifies the primary spoken language of the trending repository, or of the trending developer's repository.

- **Date range**: `since` (str). Must be one of daily, weekly, or monthly.

  The date range of which the repository or developer is trending for. This parameter affects the `currentPeriodStars` value of the resulting list of dictionaries returned. See the documentation page of *fetch_repos() function* for details.

Helper functions for the validation of these parameters are available in the *ParamUtils module*

5.1.4 Return values

Where a list of dictionaries are returned, here are the keys in each dictionary for repositories:

```
author              str
name                str
avatar              str (url)
description         str
url                 str
language            str
```

(continues on next page)

(continued from previous page)

```
languageColor       str (hex)
stars               int
forks               int
currentPeriodStars  int (stars increase for date range)
builtBy             list (dicts)
    username            str
    href                str
    avatar              str (url)
fullname            str ("user/repo")
```

And here are the keys for developers:

```
username         str
name             str
url              str
sponsorUrl       str
avatar           str (url)
repo             dict
    name             str
    description      str
    url              str
```

5.2 Utilities

The `gtrending.paramutils` module provides utility functions for search parameters, including:

- `language`,
- `spoken_language_code`, and
- `since`.

5.2.1 Terminology

For `language` and `spoken_language_code`, only values in the required form is accepted. For example, `emacs-lisp` instead of `Emacs Lisp`, or `es` instead of `Spanish`.

These values can be validated using functions with the corresponding prefix `language_*()` or `spoken_language_*()` in this module.

For **language**: The parameter value that should be used is named **"param"**.

For **spoken language**: The parameter value that should be used is named **"code"**.

The full name of both of these parameters, are named **"name"**.

Table 1: TLDR

Name	Meaning	Example
Language name	Full name of language	"Common Lisp"
Language param	Value used in *fetch* functions	*"common-lisp"*
Spoken language name	Full name of spoken language	"Italian"
Spoken language code	2-character code for spoken language; Value used in *fetch* functions	*"it"*

Values in *italics* indicate the value format that must be used in the fetch module.

Note: The `since` parameter has only one form, it can only be one of "daily", "weekly", or "monthly".

5.2.2 Conversion

Hence, conversion between the two formats may be done using `convert_*()` functions:

- `convert_language_name_to_param()` Converts a given name of a programming language, eg: "Emacs Lisp", into the correct parameter form to be used when calling functions in *gtrending.fetch module*, eg: "emacs-lisp".
- `convert_spoken_language_name_to_code()` Converts a given name of a spoken language, eg: "Spanish", into the correct parameter form (the language code), eg: "es"

5.2.3 Validation

Functions that can be used to validate arguments before passing to fetch-functions have the prefix `check_*()`.

Note: If an empty string is passed as an argument to these functions, `False` is returned, **even though all parameters are optional in the fetch module**.

- `check_language()` checks that the argument is a valid language parameter
- `check_spoken_language()` checks that the argument is a valid spoken language code
- `check_since()` checks that the argument is one of the valid options for date range: daily, weekly, and monthly.

These functions are called from within fetch-functions, where **ValueError**s are raised if: the argument is truthy, AND `False` returned when passed to corresponding `check_*()` functions.

For example, the following WILL raise ValueError:

```
fetch_repos("does-not-exist")
fetch_developers("common lisp")
fetch_developers(since="yearly")
fetch_repos(spoken_language_code="English")
```

And the following WILL NOT raise ValueError:

```
fetch_repos("")
fetch_developers(since=None)
fetch_developers(language="css")
```

5.2.4 Lists and dictionaries

Languages List - A list of dictionaries with param and name.:

```
>>> languages_list()
[
  {
    "param": "1c-enterprise",
    "name": "1C Enterprise"
  },
  {
    "param": "abap",
    "name": "ABAP"
  },

  ...

  {
    "param": "yara",
    "name": "YARA"
  },
  {
    "param": "zephir",
    "name": "Zephir"
  },
  {
    "param": "zimpl",
    "name": "Zimpl"
  },
]
```

Languages Dictionary - A dictionary of each param to its full name.:

```
>>> languages_dict()
{
    '1c-enterprise': '1C Enterprise',
    'abap': 'ABAP',
    'abnf': 'ABNF',
    ...
    'yara': 'YARA',
    'zephir': 'Zephir',
    'zimpl': 'Zimpl'
}
```

Spoken Languages List - A list of dictionaries of code and the list of names.:

```
>>> spoken_languages_list()
[
    {'code': 'ab', 'name': ['Abkhazian']},
    {'code': 'aa', 'name': ['Afar']},
    {'code': 'af', 'name': ['Afrikaans']},
    {'code': 'ak', 'name': ['Akan']},
    ...
    {'code': 'yo', 'name': ['Yoruba']},
```

(continues on next page)

(continued from previous page)

```
    {'code': 'za', 'name': ['Zhuang', 'Chuang']},
    {'code': 'zu', 'name': ['Zulu']}
]
```

Spoken Languages Dictionary - A dictionary of the languages codes to its list of names.:

```
>>> spoken_languages_dict()
{
    'aa': ['Afar'],
    'ab': ['Abkhazian'],
    'ae': ['Avestan'],
    'af': ['Afrikaans'],
    ...
    'yi': ['Yiddish'],
    'yo': ['Yoruba'],
    'za': ['Zhuang', 'Chuang'],
    'zh': ['Chinese'],
    'zu': ['Zulu'],
}
```

5.3 API Reference

5.3.1 fetch

Fetch trending repositories and developers using github-trending-api

gtrending.fetch.**fetch_developers**(*language: str | None = ''*, *since: str | None = 'daily'*) → List[dict]

Fetch trending developers on GitHub.

Parameters

- **language** (*str, optional*) – The programming language, eg: python
- **since** (*str, optional*) – The time range, choose from [daily, weekly, monthly].
- **"daily".** (*Defaults to*) –

Returns

A list of dictionaries containing information for each trending developer found

Return type

list(dict)

Raises

ValueError – When any of the arguments are invalid

Examples

```
fetch_developers()
fetch_repos(language="python")
fetch_repos("C", since="monthly")
```

gtrending.fetch.**fetch_repos**(*language: str | None = ''*, *spoken_language_code: str | None = ''*, *since: str | None = 'daily'*) → List[dict]

Fetch trending repositories on GitHub.

Parameters

- **language** (*str, optional*) – Filtering by language, eg: python, common-lisp
- **spoken_language_code** (*str, optional*) – The spoken language, eg: en for english
- **since** (*str, optional*) – The time range, choose from: [daily, weekly, monthly].
- **"daily".** (*Defaults to*) –

Note: spoken_language_code argument must be the language code ("en" and not "english"). To convert language name to language code, use convert_spoken_language_name_to_code().

Likewise, language argument must be the parameter value ("common-lisp" not "Common Lisp"). To convert the name to param, use convert_language_name_to_param().

Returns
A list of dictionaries containing information for each trending repository found

Return type
list(dict)

Raises
ValueError – When any of the arguments are invalid

Examples

```
fetch_repos()
fetch_repos(language="python")
fetch_repos("C", "zh", "monthly")
```

5.3.2 paramutils

Parameter utility functions

gtrending.paramutils.**check_language**(*language: str*) → bool

Check if the language parameter is valid.

Value that is already url-encoded would not be accepted.

Returns false for falsey values.

Examples

```
>>> check_language('python')
True
>>> check_language('py')
False
>>> check_language('GO')
True
>>> check_language('c%2B%2B')
False
>>> check_language('c++')
True
>>> check_language('')
False
>>> check_language('vim-script')
True
```

Parameters
language (*str*) – The language, eg: python (case-insensitive)

Returns
True for valid language, False otherwise

Return type
bool

gtrending.paramutils.**check_since**(*since: str*) → bool

Check if the time range value is correct.

Examples

```
>>> check_since('daily')
True
>>> check_since('DAILY')
True
>>> check_since('yearly')
False
```

Parameters
since (*str*) – The time range

Returns
True for valid parameter, False otherwise

Return type
bool

gtrending.paramutils.**check_spoken_language**(*spoken_language: str*) → bool

Check if the spoken language code or name exists.

Returns false for falsey values

Examples

```
>>> check_spoken_language('english')
True
>>> check_spoken_language('en')
True
>>> check_spoken_language('EL')
True
>>> check_spoken_language('python')
False
```

Parameters
sl (*str*) – The spoken language, eg: English, or en for English

Returns
True for valid spoken language, False otherwise

Return type
bool

gtrending.paramutils.**check_spoken_language_code**(*code: str*) → bool

Check if the spoken language code exists, case-insensitive.

Returns false for falsey values.

Examples

```
>>> check_spoken_language_code('english')
False
>>> check_spoken_language_code('en')
True
>>> check_spoken_language_code('EL')
True
>>> check_spoken_language_code('py')
False
```

Parameters
sl (*str*) – The spoken language code, eg: en, es

Returns
True for valid spoken language code, False otherwise.

Return type
bool

gtrending.paramutils.**check_spoken_language_name**(*spoke_language: str*) → bool

Check if the spoken language name exists, case-insensitive.

Returns false for falsey values.

Examples

```
>>> check_spoken_language_name('english')
True
>>> check_spoken_language_name('en')
False
>>> check_spoken_language_name('Greek')
True
>>> check_spoken_language_name('')
False
```

Parameters
sl (*str*) – The spoken language, eg: English, Spanish

Returns
True for valid spoken language name, False otherwise

Return type
bool

gtrending.paramutils.**convert_language_name_to_param**(*language: str*) → str

Convert language name to value used as API parameter

Examples

```
>>> convert_language_name_to_param('Python')
'python'
>>> convert_language_name_to_param('')
''
>>> convert_language_name_to_param('perl 6')
'perl-6'
>>> convert_language_name_to_param('c++')
'c++'
>>> convert_language_name_to_param('C#')
'c#'
```

Parameters
language (*str*) – The language name, eg: "Vim script"

Returns
The language in parameter format, eg: vim-script

Return type
str

gtrending.paramutils.**convert_spoken_language_name_to_code**(*sl_name: str*) → str

Convert spoken language name to its code.

Returns an empty string for an invalid name.

Examples

```
>>> convert_spoken_language_name_to_code('Greek')
'el'
>>> convert_spoken_language_name_to_code('French')
'fr'
```

Parameters
sl_name (*str*) – The spoken language name

Returns
The corresponding spoken_language code

Return type
str

gtrending.paramutils.**languages_dict**() → dict

Dictionary of the language param for its name.

Example

```
{
    ...
    'restructuredtext': 'reStructuredText',
    'rexx': 'REXX',
    'rhtml': 'RHTML',
    'ring': 'Ring',
    'rmarkdown': 'RMarkdown',
    'robotframework': 'RobotFramework',
    'roff': 'Roff',
    'rouge': 'Rouge',
    'rpc': 'RPC',
    'rpm-spec': 'RPM Spec',
    'ruby': 'Ruby',
    'runoff': 'RUNOFF',
    'rust': 'Rust',
    'sage': 'Sage',
    'saltstack': 'SaltStack',
    'sas': 'SAS',
    'sass': 'Sass',
    'scala': 'Scala',
    'scaml': 'Scaml',
    'scheme': 'Scheme',
    'scilab': 'Scilab',
    'scss': 'SCSS',
    'sed': 'sed',
    'self': 'Self',
    'shaderlab': 'ShaderLab',
    'shell': 'Shell',
    ...
}
```

Returns
param: name for each language

Return type
dict

gtrending.paramutils.**languages_list**() → List[dict]

Fetch programming languages.

Example

```
[
    {'name': '1C Enterprise', 'param': '1c-enterprise'},
    {'name': 'ABAP', 'param': 'abap'},
    {'name': 'ABNF', 'param': 'abnf'},
    ...
    {'name': 'HTML+ERB', 'param': 'html+erb'},
    {'name': 'HTML+PHP', 'param': 'html+php'},
    {'name': 'HTTP', 'param': 'http'},
    {'name': 'Hy', 'param': 'hy'},
    {'name': 'HyPhy', 'param': 'hyphy'},
    {'name': 'IDL', 'param': 'idl'},
    {'name': 'Idris', 'param': 'idris'},
    {'name': 'IGOR Pro', 'param': 'igor-pro'},
    {'name': 'Inform 7', 'param': 'inform-7'},
    {'name': 'INI', 'param': 'ini'},
    {'name': 'Inno Setup', 'param': 'inno-setup'},
    {'name': 'Io', 'param': 'io'},
    {'name': 'Ioke', 'param': 'ioke'},
    {'name': 'IRC log', 'param': 'irc-log'},
    ...
]
```

Returns
A list of dictionaries containing languages, mapping the param value to its name

Return type
list(dict)

gtrending.paramutils.**languages_names**() → list

List of valid language names.

Example

```
[..., "HTML", "Makefile", "Lua", "M4", "Mathematica", "Emacs Lisp", ...]
```

Returns
List of capitalized names as strings

Return type
list(str)

gtrending.paramutils.**languages_params**() → List[str]

List of valid language params.

Example

```
[..., "html", "makefile", "lua", "m4", "mathematica", "emacs-lisp", ...]
```

Returns
List of languages' params as strings

Return type
list(str)

gtrending.paramutils.**spoken_languages_codes**() → list

List of valid spoken language codes

Example

::
["en", "es", "it", "fr", ...]

Returns
2-character codes as strings

Return type
list(str)

gtrending.paramutils.**spoken_languages_dict**() → dict

Dictionary of the spoken language code for its name

Example

```
{
    ...
    'nl': ['Dutch', 'Flemish'],
    'nn': ['Norwegian Nynorsk'],
    'no': ['Norwegian'],
    'nr': ['South Ndebele'],
    'nv': ['Navajo', 'Navaho'],
    'ny': ['Chichewa', 'Chewa', 'Nyanja'],
    'oc': ['Occitan'],
    'oj': ['Ojibwa'],
    'om': ['Oromo'],
    'or': ['Oriya'],
    'os': ['Ossetian', 'Ossetic'],
    'pa': ['Punjabi', 'Panjabi'],
    'pi': ['Pali'],
    'pl': ['Polish'],
    'ps': ['Pashto', 'Pushto'],
    'pt': ['Portuguese'],
```

(continues on next page)

(continued from previous page)

```
    'qu': ['Quechua'],
    'rm': ['Romansh'],
    'rn': ['Rundi'],
    'ro': ['Romanian', 'Moldavian', 'Moldovan'],
    'ru': ['Russian'],
    'rw': ['Kinyarwanda'],
    'sa': ['Sanskrit'],
    'sc': ['Sardinian'],
    'sd': ['Sindhi'],
    'se': ['Northern Sami'],
    'sg': ['Sango'],
    'si': ['Sinhala', 'Sinhalese'],
    'sk': ['Slovak'],
    ...
}
```

Returns
code: `[names, ...]` for each spoken language

Return type
dict

gtrending.paramutils.**spoken_languages_list**() → List[dict]

Fetch spoken languages.

Example

```
[
    ...
    {'code': 'bh', 'name': ['Bihari languages']},
    ...
    {'code': 'ca', 'name': ['Catalan', 'Valencian']},
    {'code': 'ch', 'name': ['Chamorro']},
    {'code': 'ce', 'name': ['Chechen']},
    {'code': 'ny', 'name': ['Chichewa', 'Chewa', 'Nyanja']},
    {'code': 'zh', 'name': ['Chinese']},
    ...
    {'code': 'cs', 'name': ['Czech']},
    {'code': 'da', 'name': ['Danish']},
    {'code': 'dv', 'name': ['Divehi', 'Dhivehi', 'Maldivian']},
    {'code': 'nl', 'name': ['Dutch', 'Flemish']},
    {'code': 'dz', 'name': ['Dzongkha']},
    {'code': 'en', 'name': ['English']},
    {'code': 'eo', 'name': ['Esperanto']},
    {'code': 'et', 'name': ['Estonian']},
    ...
    {'code': 'de', 'name': ['German']},
    {'code': 'el', 'name': ['Greek', 'Modern']},
    ...
]
```

Returns
A list dictionaries of spoken languages, mapping the code to the name

Return type
list(dict)

gtrending.paramutils.**spoken_languages_names**() → List[str]

List of valid spoken language names

Example

::

["English", "Spanish", "Italian", "French", ...]

Returns
Capitalized spoken language names as strings

Return type
list(str)

PYTHON MODULE INDEX

g

INDEX