finpandas

Release 0.1.0alpha

Matthew Beveridge

Jan 08, 2022

GETTING STARTED:

Public financial analysis tools in Python.

CHAPTER

ONE

INSTALL

Version 0.1.0alpha

For now, you will need to install from GitHub:

```
git clone git@github.com:mattbev/finpandas.git
cd finpandas
pip install .
```

In the future, this package will be released on PyPI.

CHAPTER

TWO

LICENSE

MIT License

Copyright (c) 2021 Matthew Beveridge

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

THREE

FINPANDAS PACKAGE

finpandas package.

Used to analyze SEC fundamental data.

3.1 Subpackages

3.1.1 finpandas.core package

finpandas core utilities

Submodules

finpandas.core.connections module

finpandas base module.

class DatabaseConnection(*database: str*, *username: str = 'finpandas'*, *password: Optional[str] = None*, *_host: str = 'database-blacktip.cpeql2xeyjqq.us-east-2.rds.amazonaws.com'*, *_port: int = 3306*)

Bases: `object`

Establish a connection with the database.

Parameters

- **username** (*str*) – your username
- **password** (*str*) – your password

database: str

dispose()
closes the database connection

password: str = None

query(*command: str*)
query information directly from the database

Parameters **command** (*str*) – the string mySQL command to execute

Returns the results of the query

Return type sqlalchemy.engine.Result

username: str = 'finpandas'

finpandas.core.datasources module

Used for analysis of public companies filing with the United States SEC.

class Fundamentals(*database: str = 'xbrl'*, *username: str = 'finpandas'*, *password: Optional[str] = None*, *_host: str = 'database-blacktip.cpeql2xeyjqq.us-east-2.rds.amazonaws.com'*, *_port: int = 3306*)

Bases: *finpandas.core.connections.DatabaseConnection*

Fundamentals API

Parameters **database** (*str, optional*) – database to connect to. Defaults to 'xbrl'.

database: str = 'xbrl'

ten_k(*ticker_or_cik: Union[str, int]*, *years: Union[int, Iterable[int]]*) → *finpandas.dataframes.forms.Form10K*

get company 10-Ks for selected years

Parameters

- **ticker_or_cik** (*Union[str,int]*) – company's ticker or SEC issued Central Index Key (CIK)
- **years** (*Union[int,Iterable[int]]*) – a year or iterable of years of interest

Returns a representation of the company's 10Ks

Return type *Form10K*

ten_q(*ticker_or_cik: Union[str, int]*, *periods: Union[int, Iterable[int], Tuple[Union[str, int], Union[str, int]], Iterable[Tuple[Union[str, int], Union[str, int]]]]*) → *finpandas.dataframes.forms.Form10Q*

get company 10-Qs for selected periods

Parameters

- **ticker_or_cik** (*Union[str,int]*) – company's ticker or SEC issued Central Index Key (CIK)
- **periods** (*Union[int,Iterable[int],Tuple[Union[str,int],Union[str,int]], Iterable[Tuple[Union[str,int],Union[str,int]]]]*) – the year of interest, will get all quarters; years of interest, will get all quarters; (year, quarter) pair; (year, quarter) pairs of interest

Raises ***NestedDepthError*** – inputted periods is not one of the correct formats

Returns a representation of the company's 10Qs

Return type *Form10Q*

Example

Querying 10-Q reports for Apple Inc.:

```
df = instance.ten_q("AAPL", 2019)
df = instance.ten_q("AAPL", [2018, 2019])
df = instance.ten_q("AAPL", (2019, "q1"))
df = instance.ten_q("AAPL", [(2019, "q1"), (2019, "q2")]
```

class Stocks(*database: str = 'stocks'*, *username: str = 'finpandas'*, *password: Optional[str] = None*, *_host: str = 'database-blacktip.cpeql2xeyjqq.us-east-2.rds.amazonaws.com'*, *_port: int = 3306*)

Bases: *finpandas.core.connections.DatabaseConnection*

Stocks API

Parameters **database** (*str, optional*) – database to connect to. Defaults to 'stocks'.

database: str = 'stocks'

price(*ticker_or_cik: Union[str, int]*, *period_start: str*, *period_end: Optional[str] = None*) → pandas.core.frame.DataFrame

queries the price of a given equity over a period

Parameters

- **ticker_or_cik** (*Union[str,int]*) – company or fund's ticker or SEC issued Central Index Key (CIK)
- **period_start** (*str*) – the starting day for the period, in year-month-day format (e.g. 2020-06-30).
- **period_end** (*str, optional*) – the ending day for the period, in year-month-day format (e.g. 2020-06-30). Defaults to None, indicating a single day period_start.

Returns a representation of the company's historical stock price

Return type pd.DataFrame

sector(*ticker_or_cik: Union[str, int]*) → str

queries the sector of a given entity

Parameters **ticker_or_cik** (*Union[str,int]*) – company or fund's ticker or SEC issued Central Index Key (CIK)

Returns the sector

Return type str

3.1.2 finpandas.dataframes package

modified dataframe classes

Submodules

finpandas.dataframes.forms module

Abstract data types for US public fundamentals: 10K and 10Q.

class Form(*dataframe: pandas.core.frame.DataFrame*)

Bases: pandas.core.frame.DataFrame

Abstract Data Type for SEC forms. Subclass of pandas DataFrame.

asset_sheet() → pandas.core.frame.DataFrame

generates a data sheet of "Asset" values

Returns subset of the whole dataframe filtered by assets

Return type pd.DataFrame

asset_turnover(*as_list: bool = False*) → pandas.core.frame.DataFrame

calculates asset turnover = revenue / assets

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the asset turnover

Return type pd.DataFrame

balance_sheet() → pandas.core.frame.DataFrame

Returns subset of the whole dataframe filtered by values on a balance sheet

Return type pd.DataFrame

book_value(*as_list: bool = False*) → pandas.core.frame.DataFrame

calculates the book value = total assets - total liabilities

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the book values

Return type pd.DataFrame

cash_flow() → pandas.core.frame.DataFrame

Returns subset of the whole dataframe filtered by values on a cash flow statement

Return type pd.DataFrame

cash_turnover(*as_list: bool = False*) → pandas.core.frame.DataFrame

calculates cash turnover = revenue / cash and cash equivalents at carrying value

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the cash turnover

Return type pd.DataFrame

current_asset_turnover(*as_list: bool = False*) → pandas.core.frame.DataFrame

calculates current asset turnover = revenue / current assets

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the current asset turnover

Return type pd.DataFrame

current_ratio(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates the current ratio = current assets / current liabilities

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the current ratio values

Return type pd.DataFrame

debt_capitalization(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates debt capitalization = liabilities / (liabilities + stockholder equity)

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the debt capitalization

Return type pd.DataFrame

debt_sheet() → pandas.core.frame.DataFrame
generates a data sheet of "Debt" values

Returns subset of the whole dataframe filtered by debts

Return type pd.DataFrame

debt_to_assets(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates the debt-to-equity ratio = total liabilities / total assets

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the debt-to-assets ratio

Return type pd.DataFrame

debt_to_equity(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates the debt-to-equity ratio = total liabilities / total stockholders equity

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the debt-to-equity ratio

Return type pd.DataFrame

ebit(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates EBIT = Net Income + Taxes

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the EBIT

Return type pd.DataFrame

ebitda(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates EBITDA = Net Income + Taxes + Depreciation + Amortization

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the EBITDA

Return type pd.DataFrame

effective_tax_rate(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates effective tax rate = income tax / EBIT

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the effective tax rate

Return type pd.DataFrame

gross_margin(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates gross margin

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the gross margin

Return type pd.DataFrame

inventory_turnover(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates inventory tower = revenue / inventory net

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the inventory turnover

Return type pd.DataFrame

liability_sheet() → pandas.core.frame.DataFrame
generates a data sheet of "Liability" values

Returns subset of the whole dataframe filtered by liabilities

Return type pd.DataFrame

net_income(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates net income

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the net income

Return type pd.DataFrame

net_working_capital(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates net working capital = current assets - liabilities

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the effective tax rate

Return type pd.DataFrame

net_working_capital_per_share(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates net working capital = current assets - liabilities

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the effective tax rate

Return type pd.DataFrame

quick_ratio(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates the quick ratio = (current assets - inventory) / current liabilities

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the quick ratio values

Return type pd.DataFrame

receivables_turnover(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates receivables turnover = revenue / account receivable

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the receivables turnover

Return type pd.DataFrame

return_on_assets(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates ROA = net income / average total assets

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the ROA values

Return type pd.DataFrame

return_on_equity(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates ROE = net income / total stockholders equity

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the ROE values

Return type pd.DataFrame

revenue(*as_list: bool = False*) → pandas.core.frame.DataFrame
calculates revenue

Parameters **as_list** (*bool, optional*) – returns values as a list if True, as a dataframe otherwise. Defaults to False.

Returns the revenue

Return type pd.DataFrame

search(*regex: str*, *index: str = 'tag'*, *axis: int = 0*) → pandas.core.frame.DataFrame
generates a data sheet based on a keyword regex

Parameters

- **regex** (*str*) – the keyword regular expression
- **index** (*str, optional*) – dataframe index to search along (“tag”, “uom”, “fy”, “fp”, etc.). Defaults to “tag”.

- **axis** (*int, optional*) – the axis the index belongs to ("tag", "uom" on 0, "fy", "fp" on 1). Defaults to 0.

Returns subset of the whole dataframe filtered by the regex

Return type pd.DataFrame

class Form10K(*dataframe: pandas.core.frame.DataFrame*)
Bases: *finpandas.dataframes.forms.Form*

Abstract Data Type for the SEC Form 10-K

class Form10Q(*dataframe: pandas.core.frame.DataFrame*)
Bases: *finpandas.dataframes.forms.Form*

Abstract Data Type for the SEC Form 10-Q

finpandas.dataframes.prices module

Abstract data types for historical pricing.

class HistoricalPrices(*dataframe: pandas.core.frame.DataFrame*)
Bases: pandas.core.frame.DataFrame

Abstract Data Type for historical prices. Subclass of pandas DataFrame.

class HistoricalStockPrices(*dataframe: pandas.core.frame.DataFrame*)
Bases: *finpandas.dataframes.prices.HistoricalPrices*

Abstract Data Type for historical stock prices

3.1.3 finpandas.resources package

resource files

3.1.4 finpandas.utils package

utilities

Submodules

finpandas.utils.errors module

Custom errors.

exception NestedDepthError(*input_depth: int*, *correct_depth: int*, *message='iterable depth of ({input_depth}) invalid for required depth ({correct_depth})'*)
Bases: Exception

Exception raised when the inputted nested iterable is the incorrect depth

Parameters

- **input_depth** (*int*) – the depth of the inputted iterable
- **correct_depth** (*int*) – the correct depth of the inputted iterable

- **message** (*str, optional*) – printed output. Defaults to "the nested depth of the iterable ({input_depth}) is invalid for required depth ({correct_depth})".

exception TickerNotFoundError(*ticker: str*, *message="inputted ticker '{ticker}' could not be found"*)
Bases: Exception

Exception raised when the inputted ticker is not found

Parameters

- **ticker** (*str*) – the ticker
- **message** (*str, optional*) – printed output. Defaults to "the inputted ticker '{ticker}' could not be found".

finpandas.utils.functional module

helper functions

class SafeList(*iterable=()*, */*)
Bases: list

subclass of list datatype that allows safe indexing

get(*index: int*, *default: Optional[object] = None*) → object
safe indexing method

Parameters

- **index** (*int*) – [description]
- **default** (*object, optional*) – what to return if indexing fails. Defaults to None.

Returns either the value at that index or a default value

Return type object

count_object_methods(*obj: object*)
get the number of callable object methods

Parameters **obj** (*object*) – any object

Returns the number of object class methods

Return type int

get_profile_by_cik(*cik: Union[str, int]*) → Union[None, Tuple[int, str, str]]
gets the company profile using the CIK number

Parameters **cik** (*Union[str,int]*) – the SEC issued Central Index Key (CIK)

Returns (cik, ticker, title)

Return type Union[None,Tuple[int,str,str]]

get_profile_by_ticker(*ticker: str*) → Union[None, Tuple[int, str, str]]
gets the company profile using the ticker

Parameters **ticker** (*str*) – the ticker of the company

Returns (cik, ticker, title)

Return type Union[None,Tuple[int,str,str]]

get_profile_by_title(*title: str*) → Union[None, Tuple[int, str, str], Iterable[Tuple[int, str, str]]]
gets the company profile using the title

Parameters **title** (*str*) – the company's name in glob form

Returns (cik, ticker, title)

Return type Union[None,Tuple[int,str,str],Iterable[Tuple[int,str,str]]]

nested_depth(*iterable: Iterable*) → int
calculate the nested depth of an iterable

Parameters **iterable** (*Iterable*) – the iterable to calculate the depth of

Returns the depth

Return type int

ticker_or_cik_parser(*ticker_or_cik: Union[str, int]*) → int
takes in a ticker or CIK number and returns just the CIK number. a convenience function for internal use.

Parameters **ticker_or_cik** (*Union[str,int]*) – ticker or SEC issued Central Index Key (CIK)

Raises ***TickerNotFoundError*** – if the ticker is not found in our database

Returns the CIK number

Return type int

finpandas.utils.jobs module

Utility for multiprocessing finpandas jobs

class Jobs
Bases: object

add_job(*f: Callable*, **args*) → None
add a job to be executed

Parameters **f** (*Callable*) – the function to execute with the job, followed in-order by any arguments.

execute() → Iterable[Any]
execute all jobs

Returns the output of each job

Return type Iterable

parmap(*f: Callable*, *X: Iterable[Any]*) → Iterable[Any]
parralelized mapping function

Parameters

- **f** (*Callable*) – function to parallelize operation over
- **X** (*Iterable*) – the arguments to the function, over N function calls.

Returns the N function outputs

Return type Iterable

spawn(*f: Callable*) → Callable
[summary]

Parameters **f** (*Callable*) – function to spawn a pipe for

Returns a function that executes the spawning of the pipe

Return type Callable

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

f

INDEX

J

L

M

N

P

Q

R

S

T

U