python-networkdays

Release 0.1

github.com/cadu-leite

Jun 21, 2020

CONTENTS:

1 networkdays 1

2 Installation 3

3 Features 5

4 Examples 7

5 Other similar projects 9

6 Indices and tables 11

Python Module Index 13

Index 15

CHAPTER

ONE

NETWORKDAYS

class `networkdays.networkdays.`**`JobSchedule`**(*project_duration_hours*, *workhours_per_day*, *date_start*, *networkdays=None*)

`days`()

`job_workdays`()

list workdays for a given job duration

Returns workday datetime.date list

Return type list

`months`(*year=None*)

return a weeks *iterATOR*

Parameters **`year`** (*`None, optional`*) – Description

Returns Description

Return type TYPE

`weeks`(*year=None*, *month=None*)

return an *interator* for ISO format see https://docs.python.org/3/library/datetime.html#datetime.date.isocalendar)

Parameters

- **`year`** (*`None, optional`*) – filter per year
- **`month`** (*`None, optional`*) – filter per month

Returns weeks iso numbers based

Return type iter

`years`()

Its not duration

class `networkdays.networkdays.`**`Networkdays`**(*date_start*, *date_end=None*, *holidays={}*, *weekdaysoff={6, 7}*)

`holidays`()

`networkdays`()

NetWorkDays like Excel Networkdays function. Given 2 dates, it returns the number of days between the dates, minus holidays, minus week days off (ex.: saturday and sunday).

The *weekdaysoff* indicates days not worked per week in iso format (ex for Sun and Sat => {6,7})

Holidays may be any date, datetime.date object, in a year.

Parameters

- **`date_start`** (*`datetime.date`*) – initial date
- **`date_end`** (*`datetime.date`*) – last date - if none, is the same as *date_start* plus 1 year.
- **`workdays`** (*`set`*) – set (list) of working days in ISO format, Monday is 1 and Sunday is 7.
- **`holidays`** (*`set`*) – datetime object set, indicating days off.
- **`weekdaysoff`** (*`set`*) – set of weekdays not working, default is Saturday and Sunday {6,7}.

Returns list of work days.

ex.:

networkdays(datetime.date(2020,1,1), datetime.date(2020,2,31), holiday=datetime.date(2020,1,1), weekdaysoff={6,7}

)

`weekends``()`

- Business days calendar.
- JobSchedule on business days.

Tip: Just Python built-in libs, no dependencies

Networkdays: Return working days between two dates exclude weekends and holidays.

- just like spreadsheets *networdays* function
- exclude Holidays
- Exclude "days off" per week.

Job schedule: Calculate the period for a given job hours, based on *Networdays*.

Table of Contents

- *python-networkdays's documentation*
  - *Installation*
  - *Features*
  - *Examples*
    - *Networkdays.networkdays()*
    - *Networkdays.jobschedule()*
  - *Other similar projects*
  - *Indices and tables*

CHAPTER TWO

INSTALLATION

python-networkdays can be installed from PyPI using pip

```
pip install python-networkdays
```

Tip: note that the package name is different from the importable name

Page on Pypi: https://pypi.org/project/python-networkdays/

There is no dependencies.

CHAPTER THREE

FEATURES

- Return a list of business days between 2 dates.
- Exclude weekends by default
- Custom “days off” may be informed as list like {1,2,3,4,5,6,7}, where 1 is Monday default is {6,7} = (Sat, Sun).
- How many business days between two dates.
- How many days off, including holidays and weekends.
- Return a list of business days for a given number of hours
- Return a list of Years, months or weeks for a given number of hours
- **No Pandas or NumPy dependencies**

CHAPTER

FOUR

EXAMPLES

4.1 Networkdays.networkdays()

```
import datetime
from networkdays import networkdays

HOLIDAYS  = { datetime.date(2020, 12, 25),}

day
# you have methods to get holidays and weekends date list as well.
# here i just got the size of each set
print(f'''
Bussiness days: {len(days.networkdays())}
    {days.networkdays()[:2]}
    ...{days.networkdays()[-2:]}

Weekends:       {len(days.weekends())}
    {days.weekends()[:2]}
    ...{days.weekends()[-2:]}

Holidays:       {len(days.holidays())}
''')
```

```
Bussiness days: 22
    [datetime.date(2020, 12, 1), datetime.date(2020, 12, 2)]
    ...[datetime.date(2020, 12, 30), datetime.date(2020, 12, 31)]

Weekends:       8
    [datetime.date(2020, 12, 5), datetime.date(2020, 12, 6)]
    ...[datetime.date(2020, 12, 26), datetime.date(2020, 12, 27)]

Holidays:       1
```

4.2 Networkdays.jobschedule()

```
# jobSchedule
import datetime
from networkdays import networkdays

# Distribute the 600 hrs of effort, starting on december 1, 2020 working 8hrs per day.
jobschedule = networkdays.JobSchedule(600, 8, datetime.date(2020, 12, 1),␣
↪networkdays=None)
job_dates = jobschedule.job_workdays()

# print results ...
print(f'''

bussines days:          {jobschedule.bussines_days}
calendar days:          {jobschedule.total_days}
starts - ends:          {jobschedule.prj_starts} - {jobschedule.prj_ends}

years:                  {list(jobschedule.years())}
months:                 {list(jobschedule.months())}
weeks (ISO):            {list(jobschedule.weeks())}

days:
    {list(jobschedule.days())[:2]} ...\n ...{list(jobschedule.days())[-2:]}

Works days dates on january:
    {list(jobschedule.days())[:2]} ...\n ...{list(jobschedule.days())[-2:]}
''')
```

```
bussines days:          54
calendar days:          73 days, 0:00:00
starts - ends:          12/01/20 - 02/12/21

years:                  [2020, 2021]
months:                 [12, 1, 2]
weeks (ISO):            [49, 50, 51, 52, 53, 1, 2, 3, 4, 5, 6]

days:
    [datetime.date(2020, 12, 1), datetime.date(2020, 12, 2)] ...
 ...[datetime.date(2021, 2, 11), datetime.date(2021, 2, 12)]

Works days dates on january:
    [datetime.date(2020, 12, 1), datetime.date(2020, 12, 2)] ...
 ...[datetime.date(2021, 2, 11), datetime.date(2021, 2, 12)]
```

CHAPTER

FIVE

OTHER SIMILAR PROJECTS

When I start to code, I did check for some similar projects.

I knew about python-dateutil, a great project I use for years… I'd like something more straightforward or simpler.

After to publish the python-networkdays on PyPi I found some others 8(

- workdays : A 5 years old project, looks the same as **networkdays_**
- timeboard : A more complex but powerful project
- python-dateutil is great, powerful but even more complex.
- python-bizdays : Quick simple and direct …

I will try to keep this list updated…

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

n

networkdays.networkdays, 1

INDEX

D

days() (*networkdays.networkdays.JobSchedule method*), 1

H

holidays() (*networkdays.networkdays.Networkdays method*), 1

J

job_workdays() (*networkdays.networkdays.JobSchedule method*), 1

JobSchedule (*class in networkdays.networkdays*), 1

M

module
    networkdays.networkdays, 1

months() (*networkdays.networkdays.JobSchedule method*), 1

N

Networkdays (*class in networkdays.networkdays*), 1

networkdays() (*networkdays.networkdays.Networkdays method*), 1

networkdays.networkdays
    module, 1

W

weekends() (*networkdays.networkdays.Networkdays method*), 2

weeks() (*networkdays.networkdays.JobSchedule method*), 1

Y

years() (*networkdays.networkdays.JobSchedule method*), 1