

---

# **daylio-parser**

***Release 0.1.0***

**staticf0x**

**May 05, 2022**



**CONTENTS:**

<b>1</b>	<b>Installation</b>	<b>1</b>
<b>2</b>	<b>Config</b>	<b>3</b>
<b>3</b>	<b>Parser</b>	<b>5</b>
<b>4</b>	<b>PlotData</b>	<b>7</b>
<b>5</b>	<b>Stats</b>	<b>9</b>
	<b>Index</b>	<b>11</b>



## INSTALLATION

Via pip:

```
pip3 install --user daylio-parser
```

Via Pipenv:

```
pipenv install daylio-parser
```

Via Poetry:

```
poetry add daylio-parser
```



## CONFIG

daylio-parser comes with a default config that works for the default Daylio setup after installing the app. That is, there's just 5 moods, called `awful`, `bad`, `meh`, `good`, `rad`.

Each mood has its class:

### **class Mood**

**name:** `str`

Name of the mood, must correspond with mood name in the exported CSV.

**level:** `int`

Assigned numeral level for the mood (higher = better mood). The allowed levels are 1 to 5.

**color:** `str`

Any hex color.

**boundaries:** `Tuple[float, float]`

A tuple with lower and upper bound for the mood. Any average mood that falls within these boundaries will be colored using the `Mood.color`.

The whole mood config for your app will be constructed using the `MoodConfig` class.

### **class MoodConfig**(*mood\_list=None, color\_palette=None*)

Creates a config with `mood_list`. If the mood list isn't provided, `DEFAULT_MOODS` will be used. All moods are automatically colored using `color_palette` and boundaries are also calculated. Each boundary is exactly 1 in size, with the first one and the last one being only 0.5 in size.

#### **Parameters**

- **mood\_list** (`List[Tuple[int, str]]`) – A list of moods with (level, name)
- **color\_palette** (`List[str]`) – A list of colors (hex values or common names)

**from\_list**(*mood\_list, color\_palette=None*) → `None`

Updates the config with a new list of moods.

#### **Parameters**

- **mood\_list** (`List[Tuple[str, str]]`) – A list of moods with (level, name)
- **color\_palette** (`List[str]`) – A list of colors (hex values or common names)

**static from\_file**(*path*) → `MoodConfig`

Loads the `MoodConfig` from a JSON file. The file structure is:

```
{  
  "moods": [(level, name), ...],  
  "colors": [value, ...],  
}
```

**get**(*mood\_name*) → *Mood*

Returns a *Mood* by its name.

Raises *MoodNotFound* if the *mood\_name* doesn't exist.

**Parameters** *mood\_name* (*str*) – Mood name

**exception** *MoodNotFound*

Raised in cases when attempting to retrieve a non-existing *Mood*.



## PARSER

### **class Entry**

A class that holds data for an entry in the diary. One day can have multiple entries.

**datetime:** `datetime.datetime`

**mood:** `Mood`

**activities:** `List[str]`

**notes:** `str`

### **class Parser**(*config=None*)

Parser for the CSV file. If config is not provided, a default one will be created.

**Parameters** **config** (`MoodConfig`) – MoodConfig for the parser

**load\_csv**(*path*) → `List[Entry]`

Load entries from a CSV file.

**Parameters** **path** (*str*) – Path to the CSV file

**load\_from\_buffer**(*f*) → `List[Entry]`

Load entries from a file like object containing CSV data.

**Parameters** **f** – A file-like object



## PLOTDATA

**class** `PlotData`(*entries*, *config=None*)

A class that provides some data for easier plotting.

**Parameters**

- **entries** (*List* [`Entry`]) – A list of parsed entries
- **config** (`MoodConfig`) – MoodConfig for the parser (if none is provided, a default one will be created)

**split\_into\_bands**(*moods*) → `numpy.ma.MaskedArray`

**Parameters** **moods** (`numpy.ndarray`) – An array of mood values

**Return type** `numpy.ma.MaskedArray`

Splits input moods into bands, given their boundaries. See [Mood.boundaries](#).

**interpolate**(*avg\_moods=None*, *interpolate\_steps=360*)

Interpolates moods to make a smooth chart. Returns an array of dates and an array of moods.

**Parameters**

- **avg\_moods** – Average moods to iterate over. If not provided, these are generated by [Stats.average\\_moods\(\)](#)
- **interpolate\_steps** (*int*) – Number of steps for one day (midnight to midnight)

**Return type** `Tuple[numpy.ndarray, numpy.ndarray]`



**class MoodPeriod**

This class represents a period of moods.

**start\_date:** `datetime.date`

**end\_date:** `datetime.date`

**duration:** `int`

Length of the period as a number of days.

**avg\_mood:** `float`

Average mood for the whole period.

**class Stats**(*entries*, *config=None*)

A class for computing various stats from the entries.

**Parameters**

- **entries** (*List* [*Entry*]) – A list of parsed entries
- **config** (*MoodConfig*) – MoodConfig for the parser (if none is provided, a default one will be created)

**average\_moods()** → *List*[*Tuple*[*datetime.date*, *float*]]

Computes average mood for each day.

**activity\_moods()** → *Dict*[*str*, *Tuple*[*float*, *float*]]

Computes average mood and standard deviation for each activity. The returned dict has mood name as a key and (mean, std) as value.

**mean()** → *Tuple*[*float*, *float*]

Returns (mean, std) for all entries.

**rolling\_mean**(*N=5*)

Computes a rolling mean for the entries.

**Parameters** *N* (*int*) – Window size

**find\_high\_periods**(*threshold=4*, *min\_duration=4*) → *List*[*MoodPeriod*]

Find all periods of high moods.

**Parameters**

- **threshold** (*float*) – Find moods higher than this
- **min\_duration** (*int*) – Find periods longer than this

**find\_low\_periods**(*threshold=3, min\_duration=5*) → List[MoodPeriod]

Find all periods of low moods.

**Parameters**

- **threshold** (*float*) – Find moods higher than this
- **min\_duration** (*int*) – Find periods longer than this

## INDEX

### A

activities (*Entry attribute*), 5  
activity\_moods() (*Stats method*), 9  
average\_moods() (*Stats method*), 9  
avg\_mood (*MoodPeriod attribute*), 9

### B

boundaries (*Mood attribute*), 3

### C

color (*Mood attribute*), 3

### D

datetime (*Entry attribute*), 5  
duration (*MoodPeriod attribute*), 9

### E

end\_date (*MoodPeriod attribute*), 9  
Entry (*built-in class*), 5

### F

find\_high\_periods() (*Stats method*), 9  
find\_low\_periods() (*Stats method*), 9  
from\_file() (*MoodConfig static method*), 3  
from\_list() (*MoodConfig method*), 3

### G

get() (*MoodConfig method*), 4

### I

interpolate() (*PlotData method*), 7

### L

level (*Mood attribute*), 3  
load\_csv() (*Parser method*), 5  
load\_from\_buffer() (*Parser method*), 5

### M

mean() (*Stats method*), 9  
Mood (*built-in class*), 3  
mood (*Entry attribute*), 5

MoodConfig (*built-in class*), 3  
MoodNotFound, 4  
MoodPeriod (*built-in class*), 9

### N

name (*Mood attribute*), 3  
notes (*Entry attribute*), 5

### P

Parser (*built-in class*), 5  
PlotData (*built-in class*), 7

### R

rolling\_mean() (*Stats method*), 9

### S

split\_into\_bands() (*PlotData method*), 7  
start\_date (*MoodPeriod attribute*), 9  
Stats (*built-in class*), 9