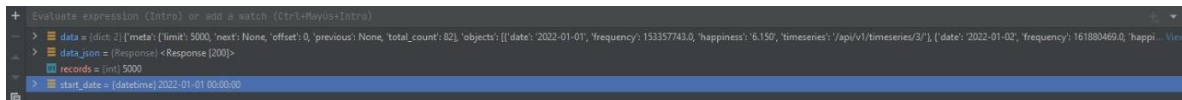


Escrito por: Daniela Martínez Madrid

En la línea 99 la variable data tiene estructura de diccionario, con dos índices: 'meta' y 'objects'.



```
+ Evaluate expression (Intro) or add a watch (Ctrl+Mayus+Intro)
> data = (dict: 2) {'meta': {'limit': 5000, 'next': None, 'offset': 0, 'previous': None, 'total_count': 82}, 'objects': [{'date': '2022-01-01', 'frequency': 153357743.0, 'happiness': 6.150, 'timeseries': '/api/v1/timeseries/3/1'}]}
> data_json = (Response) <Response [200]>
> records = (int) 5000
> start_date = (datetime) 2022-01-01 00:00:00
```

Donde el índice 'meta' le corresponde diccionario de índices: 'limit', 'next', 'offset', 'previous' y 'total account'. En el caso del índice 'objects' le corresponde una lista de diccionarios, donde cada uno de estos diccionarios contiene los índices: 'date', 'frequency', 'happiness' y 'timeseries'.

Ahora el código en la línea 99, la variable data va a ser guardada de una nueva forma, haciendo uso de list comprehension, aquí se va a generar la variable como una lista de listas, de la siguiente manera:

La primera parte del código de la línea 99: `[datetime.datetime.strptime(d['date'], "%Y-%m-%d"), d['frequency'], float(d['happiness'])]`, indica que vamos a generar una lista de tres entradas, así:

`datetime.datetime.strptime(d['date'], "%Y-%m-%d")`: lee un formato tipo string, para que el método `strptime` lo convierta en un objeto de `datetime`, por lo que la fecha se guardará aquí tendrá es de clase `datetime`.

`d['frequency']`: guarda la frecuencia.

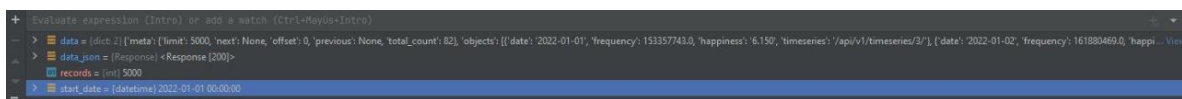
`float(d['happiness'])`: guarda el score de felicidad.

Ya escrito el código de la lista, en la segunda parte se hace un list comprehension para generar una nueva lista a partir de la dada, así:

```
data = [ [datetime.datetime.strptime(d['date'], "%Y-%m-%d"), d['frequency'], float(d['happiness'])]
for d in data['objects']]
```

Lo subrayado en rojo es la lista que ya se explico anteriormente y lo que hace el código nuevo es hacer un recorrido (usando `for`) por todos las entradas de la lista `data['objects']`, recuerde que cada entrada de la lista es un diccionario, que contiene la información solicitada para crear la lista roja.

La variable data, ya ejecutada la línea 102 tiene esta presentación:

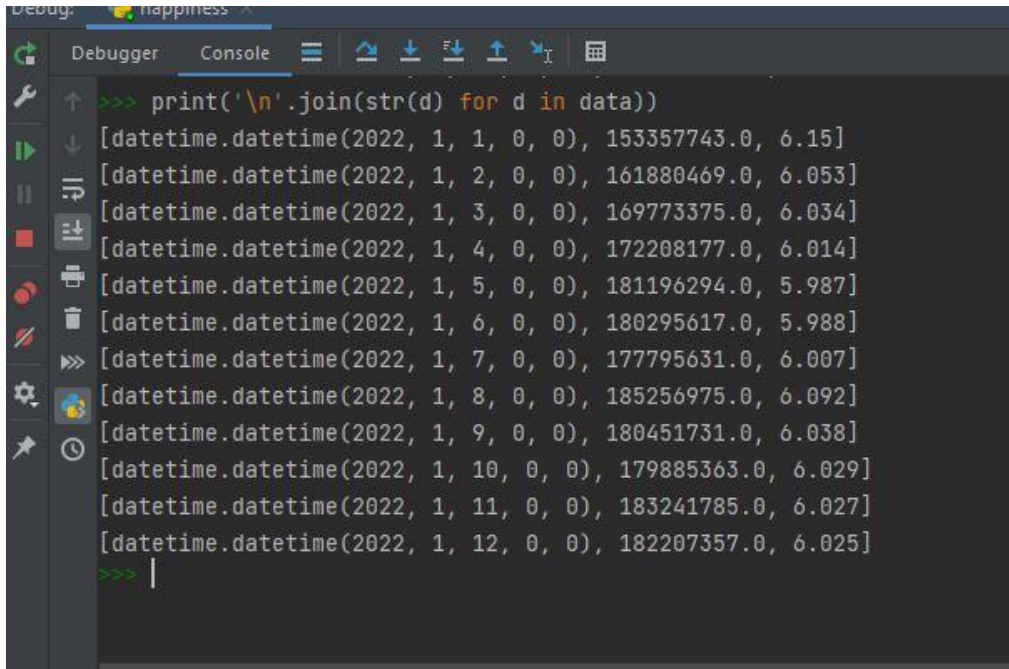


```
+ Evaluate expression (Intro) or add a watch (Ctrl+Mayus+Intro)
> data = (list: 2) [[{'date': '2022-01-01', 'frequency': 153357743.0, 'happiness': 6.150, 'timeseries': '/api/v1/timeseries/3/1'}], {'date': '2022-01-02', 'frequency': 161880469.0, 'happiness': 6.150, 'timeseries': '/api/v1/timeseries/3/1'}]]
> data_json = (Response) <Response [200]>
> records = (int) 5000
> start_date = (datetime) 2022-01-01 00:00:00
```

Es decir, una lista de listas, por ejemplo, la primera entrada es la lista:

`[datetime.datetime(2022,1,1,0,0),153357743.0,6.15]`. Como ya antes dicho, fecha, frecuencia y score de felicidad.

Ahora, nuestra variable data, en este momento es de tipo lista. Sin embargo, en la línea 116 se crea un código que permite que su visualización sea más cómoda, y nuevamente usando list comprehension se genera una variable tipo string, que presenta cada lista por línea.



```
Debugger Console
>>> print('\n'.join(str(d) for d in data))
[datetime.datetime(2022, 1, 1, 0, 0), 153357743.0, 6.15]
[datetime.datetime(2022, 1, 2, 0, 0), 161880469.0, 6.053]
[datetime.datetime(2022, 1, 3, 0, 0), 169773375.0, 6.034]
[datetime.datetime(2022, 1, 4, 0, 0), 172208177.0, 6.014]
[datetime.datetime(2022, 1, 5, 0, 0), 181196294.0, 5.987]
[datetime.datetime(2022, 1, 6, 0, 0), 180295617.0, 5.988]
[datetime.datetime(2022, 1, 7, 0, 0), 177795631.0, 6.007]
[datetime.datetime(2022, 1, 8, 0, 0), 185256975.0, 6.092]
[datetime.datetime(2022, 1, 9, 0, 0), 180451731.0, 6.038]
[datetime.datetime(2022, 1, 10, 0, 0), 179885363.0, 6.029]
[datetime.datetime(2022, 1, 11, 0, 0), 183241785.0, 6.027]
[datetime.datetime(2022, 1, 12, 0, 0), 182207357.0, 6.025]
>>> |
```