Introduction to datetime modules

- A datetime object is a single object containing all the information from a date object and a time object.

- Like a date object, datetime assumes the current Gregorian calendar extended in both directions; like a time object, datetime assumes there are exactly 3600*24 seconds in every day.

- To work with date and time, we need to import the module ‘datetime’

- Using datetime module, we can achieve the following things like get current date, month, day, year of current date, difference between two dates, date arithmetic.

- The `time` module in Python provides a set of functions to work with time-related operations, such as timekeeping, formatting, and time conversions. This module is part of the Python Standard Library and is available in all Python installations, making it a convenient and essential tool for a wide range of applications. In this day 84 tutorial, we'll explore the `time` module in Python and see how it can be used in different scenarios.

Conversion Functions Used In DateTime Module

- There are 2 functions used in datetime module. When we get the date, the default format we get is a date object.
- To modify the default format of the datetime object, we need to convert it to string format. This can be achieved by using

```
from datetime import datetime

today = datetime.now()
print ("default format: ", today)
#default format:  2024-11-27 17:39:44.013683

format_date = datetime.strftime(today, "%d-%m-%Y %H:%M:%S")
print ("formatted date: ", format_date)
#formatted date:  27-11-2024 17:39:44
```

Conversion Functions Used In DateTime Module

```
from datetime import datetime

today = datetime.now()
print ("default format: ", today)

format_date = datetime.strftime(today, "%d-%m-%Y %H:%M:%S")
print ("formatted date: ", format_date)

datestr = "12-07-2019 11:45:42"

str2date = datetime.strptime(datestr, "%d-%m-%Y %H:%M:%S")
print (str2date)
```

Output-

default format: 2024-11-27 17:44:42.025873
formatted date: 27-11-2024 17:44:42
2019-07-12 11:45:42

time.time()

- The `time.time()` function returns the current time as a floating-point number, representing the number of seconds since the epoch (the point in time when the time module was initialized). The returned value is based on the computer's system clock and is affected by time adjustments made by the operating system, such as daylight saving time. Here's an example:
- The epoch is a reference point in time used by the operating system to calculate and represent time. For most systems, including Unix/Linux and modern Python implementations, the epoch is January 1, 1970, 00:00:00 UTC (Coordinated Universal Time). The value returned by time.time() is the number of seconds that have elapsed since this epoch.
- import time
- print(time.time())
- # Output: 1602299933.233374

- As you can see, the function returns the current time as a floating-point number, which can be used for various purposes, such as measuring the duration of an operation or the elapsed time since a certain point in time.

- ## time.sleep()
- The `time.sleep()` function suspends the execution of the current thread for a specified number of seconds. This function can be used to pause the program for a certain period of time, allowing other parts of the program to run, or to synchronize the execution of multiple threads. Here's an example:

- import time
- print("Start:", time.time())
- time.sleep(2)
- print("End:", time.time())
- # Output:
- # Start: 1602299933.233374
- # End: 1602299935.233376

time.strftime()

- The `time.strftime()` function formats a time value as a string, based on a specified format. This function is particularly useful for formatting dates and times in a human-readable format, such as for display in a GUI, a log file, or a report. Here's an example:

- import time
-
  t = time.localtime()
- formatted_time = time.strftime("%Y-%m-%d %H:%M:%S", t)
-
  print(formatted_time)
- # Output: 2022-11-08 08:45:33

- import datetime
- #To get current GM time
- print("Current GM time:",time.gmtime())
- #This returns a time structure containing 9 values - year, month,day, hour, minute, sec, day of week, day of year and daylight savings.
- Output-
- Current GM time: time.struct_time(tm_year=2024, tm_mon=11, tm_mday=27, tm_hour=18, tm_min=0, tm_sec=22, tm_wday=2, tm_yday=332, tm_isdst=0)

- #To get current local time
- print("Current local time:",time.localtime())
- #This also returns a time structure containing 9 values - year, month,day, hour, minute, sec, day of week, day of year and daylight savings.

- Output-
- Current local time: time.struct_time(tm_year=2024, tm_mon=11, tm_mday=27, tm_hour=18, tm_min=0, tm_sec=22, tm_wday=2, tm_yday=332, tm_isdst=0)

```
#To extract today's date in a specified string format
print("Today's date using time module",time.strftime("%m-%m/%Y"))
#Python additionally allows use of  datetime module
#Prints today's date
print("Today's date using datetime module:", datetime.date.today())
#To extract today's date in a specified string format
print("Today's date (dd/mm/yyyy) using datetime module:",
datetime.date.today().strftime("%d/%m/%Y"))
#To convert a date in string format to datetime value
print("Today's date (dd/mm/yyyy):", datetime.datetime.strptime("17/04/1","%y/%d/%m"))
```

```
Today's date using time module 11-11/2024
Today's date using datetime module: 2024-11-27
Today's date (dd/mm/yyyy) using datetime module: 27/11/2024
Today's date (dd/mm/yyyy): 2017-01-04 00:00:00
```

- Python **timedelta** object is used to perform datetime manipulations in an easy way. The timedelta class is part of datetime module.
-
- Python timedelta object represents a duration of time. We can create its object using following factory method.
-
- datetime.timedelta(days=0, seconds=0, microseconds=0, milliseconds=0, minutes=0, hours=0, weeks=0)
-
- **datetime.timedelta example:**
- ----------
- from datetime import datetime, timedelta
-
- current_datetime = datetime.now()
- print(current_datetime) #2024-11-27 18:16:42.941286

- # future dates
- one_year_future_date = current_datetime + timedelta(days=365)
- print('One year from now Date:', one_year_future_date)
- #One year from now Date: 2025-11-27 18:18:00.826089
-
- # past dates
- three_days_before_date = current_datetime - timedelta(days=3)
- print('Three days before Date:', three_days_before_date)
- #Three days before Date: 2024-11-24 18:19:00.396589