# Working with Dates and Times in Python: Takeaways ⤴

## Syntax

### IMPORTING MODULES AND DEFINITIONS

- Importing a whole module:

```
import csv

csv.reader()
```

- Importing a whole module with an alias:

```
import csv as c

c.reader()
```

- Importing a single definition:

```
from csv import reader

reader()
```

- Importing multiple definitions:

```
from csv import reader, writer

reader()

writer()
```

- Importing all definitions:

```
from csv import *
```

### WORKING WITH THE `DATETIME` MODULE

- All examples below presume the following import code:

```
import datetime as dt
```

- Creating datetime.datetime string given a month, year, and day:

```python
eg_1 = dt.datetime(1985, 3, 13)
```

- Creating a datetime.datetime object from a string:

```python
eg_2 = dt.datetime.strptime("24/12/1984", "%d/%m/%Y")
```

- Converting a datetime.datetime object to a string:

```python
dt_object = dt.datetime(1984, 12, 24)

dt_string = dt_object.strftime("%d/%m/%Y")
```

- Instantiating a datetime.time object:

```python
eg_3 = datetime.time(hour=0, minute=0, second=0, microsecond=0)
```

- Retrieving a part of a date stored in the datetime.datetime object:

```python
eg_1.day
```

- Creating a date from a datetime.datetime object:

```python
d2_dt = dt.datetime(1946, 9, 10)

d2 = d2_dt.date()
```

- Creating a datetime.date object from a string:

```python
d3_str = "17 February 1963"

d3_dt = dt.datetime.strptime(d3_str, "%d %B %Y")

d3 = d3_dt.date()
```

- Instantiating a datetime.timedelta object:

```python
eg_4 = dt.timedelta(weeks=3)
```

- Adding a time period to a datetime.datetime object:

```python
d1 = dt.date(1963, 2, 26)

d1_plus_1wk = d1 + dt.timedelta(weeks=1)
```

- Convert integers to objects based on their epoch time value:

```python
datetime.datetime.fromtimestamp(345521)
```

## Concepts

- The datetime module contains five classes:
  - `datetime.datetime` — For working with date and time data
  - `datetime.date` — For working with date data only
  - `datetime.time` — For working with time data only
  - `datetime.timedelta` — For representing time periods
  - `datetime.timezone` — For representing a specific time zone

- Time objects behave similarly to datetime objects for the following reasons:
  - They have attributes like `time.hour` and `time.second` that you can use to access individual time components.
  - They have a `time.strftime()` method, which you can use to create a formatted string representation of the object.
- The timedelta type represents a period of time, e.g. 30 minutes or two days.
- Epoch time represents time as an integer, counting the number of seconds since midnight on January 1, 1970. Epoch time is the default on the majority of the world's servers, which makes it an important time format to know and understand.
- Common format codes when working with `datetime.datetime.strptime` :

| Strftime Code | Meaning | Examples |
|---|---|---|
| `%d` | Day of the month as a zero-padded number[1] | `04` |
| `%A` | Day of the week as a word[2] | `Monday` |
| `%m` | Month as a zero-padded number[1] | `09` |
| `%Y` | Year as a four-digit number | `1901` |
| `%y` | Year as a two-digit number with zero-padding[1, 3] | `01` (2001) `88` (1988) |
| `%B` | Month as a word[2] | `September` |
| `%H` | Hour in 24 hour time as zero-padded number[1] | `05` (5am) `15` (3pm) |
| `%p` | AM or PM[2] | `AM` |
| `%I` | Hour in 12 hour time as zero-padded number[1] | `05` (5am, or 5pm if AM/PM indicates otherwise) |
| `%M` | Minute as a zero-padded number[1] | `07` |

*1. The strptime parser will parse non-zero padded numbers without raising an error.*

*2. Date parts containing words will be interpreted using the locale settings on your computer, so strptime won't be able to parse 'febrero' (february in Spanish) if your locale is set to an english language locale.*

*3. Year values from 00-68 will be interpreted as 2000-2068, with values 70-99 interpreted as 1970-1999.*

- Operations between timedelta, datetime, date, and time objects:

| | | Resultant |
|---|---|---|

- Epoch Time:

| Operation | Explanation | Notes | Resultant Type |
|---|---|---|---|
| datetime - datetime | Calculate the time between two specific dates/times | | timedelta |
| datetime - timedelta | Subtract a time period from a date or time. | | datetime |
| datetime + timedelta | Add a time period to a date or time. | | datetime |
| timedelta + timedelta | Add two periods of time together | There are 3,600 seconds in one hour | timedelta |
| timedelta - timedelta | Calculate the difference between two time periods. | There are 86,400 seconds in 24 hours | timedelta |
| timedelta / integer | Divide a time period by a number. | | timedelta |
| timedelta * integer | Multiply a time period by a number. | | timedelta |

| Epoch Time | Date/Time in UTC |
|---|---|
| 0 | Midnight, January 1, 1970 |
| 1 | One second after midnight, January 1, 1970 |
| 60 | One minute after midnight, January 1, 1970 |
| 3600 | One hour after, January 1, 1970 |
| 86400 | Midnight, January 2, 1970 |

## Resources

- Python Documentation – Datetime module
- Python Documentation: Strftime/Strptime Codes
- strftime.org
- Wikipedia – Epoch time