# Convertir un string en fecha en Python

A la hora de programar hay ocasiones en que nos toca convertir un string en fecha (un objeto de tipo date, datetime o time). Por ejemplo: al procesar los datos de entrada que un usuario introduce por teclado, al obtener los parámetros de una petición web o al leer datos de un fichero de texto.

En todos los casos anteriores nos podemos encontrar con fechas/horas cuyo tipo real es string. Sin embargo, necesitamos convertirlas a un tipo de fecha/hora correcto (datetime, date o time) para poder realizar operaciones sobre ellas: [sumar o restar días](https://j2logo.com/operaciones-con-fechas-en-python/), obtener el día de la semana, etc.

En este post aprenderás cómo convertir un string a un objeto de tipo fecha/hora.

## Cómo convertir un string en fecha

Para convertir un string en un objeto de tipo fecha equivalente tienes que hacer uso del método strptime() de la clase datetime. Este método devuelve **siempre** un objeto datetime a partir de una cadena con un formato específico.

**NOTA:** No se puede convertir cualquier objeto de tipo string a un objeto datetime. La cadena debe tener un formato definido con antelación.

El método strptime() toma dos argumentos. El primero de ellos es la cadena a convertir y el segundo el formato que sigue dicha cadena. Para definir el formato se puede hacer uso de unos [códigos de formato](https://j2logo.com/convertir-un-string-en-fecha-en-python/#format-codes) que te enseñaré más adelante. Antes, veamos unos ejemplos.

## Ejemplo 1: Transformar un string en datetime

Supongamos que queremos convertir la cadena '20/04/2019' en un objeto datetime equivalente. Los pasos que tenemos que seguir son los siguientes (recuerda importar primero la clase datetime del módulo datetime):

from datetime import datetime

# Convertimos un string con formato <día>/<mes>/<año> en datetime

una\_fecha = '20/04/2019'

fecha\_dt = datetime.strptime(una\_fecha, '%d/%m/%Y')

print(fecha\_dt)

> 2019-04-20 00:00:00

# Comprobación del tipo del objeto fecha\_dt

print(type(fecha\_dt))

<class 'datetime.datetime'>

## Ejemplo 2: Pasar un string a datetime (otro formato distinto)

Veamos otro ejemplo con un formato de fecha/hora algo más complejo.

# Convertimos un string con formato

# <día> del <mes> de <año> a las <hora>:<minutos> en datetime

una\_fecha = '20 del 04 de 2019 a las 12:00'

fecha\_dt = datetime.strptime(una\_fecha, '%d del %m de %Y a las %H:%M')

print(fecha\_dt)

> 2019-04-20 12:00:00

## ValueError al convertir un string

Como te indicaba más arriba, la cadena a convertir debe tener un formato específico. Concretamente debe seguir el formato que se indica en el segundo parámetro con el que se llama a strptime(). En caso contrario, al llamar al método strptime() se lanzará una excepción ValueError.

una\_fecha = '20/04/2019'

fecha\_dt = datetime.strptime(una\_fecha, '%d-%m-%Y')

Traceback (most recent call last):

File "<input>", line 1, in <module>

File "/usr/local/Cellar/python/3.7.4\_1/Frameworks/Python.framework/Versions/3.7/lib/python3.7/\_strptime.py", line 577, in \_strptime\_datetime

tt, fraction, gmtoff\_fraction = \_strptime(data\_string, format)

File "/usr/local/Cellar/python/3.7.4\_1/Frameworks/Python.framework/Versions/3.7/lib/python3.7/\_strptime.py", line 359, in \_strptime

(data\_string, format))

ValueError: time data '20/04/2019' does not match format '%d-%m-%Y'

## Códigos de formato

A continuación se muestran los códigos de formato válidos que puedes usar para convertir un string en fecha (para aquellas plataformas con una implementación estándar de C).

| **Directiva** | **Significado** | **Ejemplo** |
| --- | --- | --- |
| %a | Nombre abreviado del día de la semana | Sun, Mon, … |
| %A | Nombre completo del día de la semana | Sunday, Monday, … |
| %w | Día de la semana como número decimal (domingo es 0) | 0, 1, …, 6 |
| %d | Día del mes como número decimal (añade cero delante) | 01, 02, …, 31 |
| %-d | Día del mes como número decimal | 1, 2, …, 30 |
| %b | Nombre del mes abreviado | Jan, Feb, …, Dec |
| %B | Nombre del mes completo. | January, February, … |
| %m | Mes como número decimal (añade cero delante) | 01, 02, …, 12 |
| %-m | Mes como número decimal | 1, 2, …, 12 |
| %y | Año sin siglo como número decimal (añade cero delante) | 00, 01, …, 99 |
| %-y | Año sin siglo como número decimal | 0, 1, …, 99 |
| %Y | Año con siglo como número decimal | 2013, 2019 etc. |
| %H | Hora (24 horas) como número decimal (añade cero delante) | 00, 01, …, 23 |
| %-H | Hora (24 horas) como número decimal | 0, 1, …, 23 |
| %I | Hora (12 horas) como número decimal (añade cero delante) | 01, 02, …, 12 |
| %-I | Hora (12 horas) como número decimal | 1, 2, … 12 |
| %p | AM o PM locales | AM, PM |
| %M | Minuto como número decimal (añade cero delante) | 00, 01, …, 59 |
| %-M | Minuto como número decimal | 0, 1, …, 59 |
| %S | Segundo como número decimal (añade cero delante) | 00, 01, …, 59 |
| %-S | Segundo como número decimal | 0, 1, …, 59 |
| %f | Microsegundos como número decimal (añade ceros delante) | 000000 – 999999 |
| %z | Compensación UTC con formato +HHMM o -HHMM |  |
| %Z | Nombre de la zona horaria |  |
| %j | Día del año como número decimal (añade cero delante) | 001, 002, …, 366 |
| %-j | Día del año como número decimal | 1, 2, …, 366 |
| %U | Número de semana del año (domingo como primer día de la semana). Los días de un nuevo año anteriores al primer domingo se consideran en la semana 0. | 00, 01, …, 53 |
| %W | Número de semana del año (lunes como primer día de la semana). Los días de un nuevo año anteriores al primer lunes se consideran en la semana 0. | 00, 01, …, 53 |
| %c | Representación apropiada de fecha y hora local | Mon Sep 30 07:06:05 2013 |
| %x | Representación apropiada de fecha local | 09/30/13 |
| %X | Representación apropiada de hora local | 07:06:05 |
| %% | El carácter ‘%’ literal | % |