Comprensión de listas

El poder de Python va mucho más allá de lo que a primera vista podemos imaginar.

La comprensión de listas, del inglés **list comprehensions**, es una funcionalidad que nos permite crear listas avanzadas en una misma línea de código. Esto se ve mucho mejor en la práctica, así que a lo largo de esta lección vamos a trabajar distintos ejemplos.

Ejemplo 1

Crear una lista con las letras de una palabra:

```
# Método tradicional
lista = []
for letra in 'casa':
    lista.append(letra)
print(lista)
```

```
['c', 'a', 's', 'a']
```

```
# Con comprensión de listas
lista = [letra for letra in 'casa']
print(lista)
```

```
['c', 'a', 's', 'a']
```

Como vemos, gracias a la comprensión de listas podemos indicar directamente cada elemento que va a formar la lista, en este caso la letra, a la vez que definimos el for:

```
# La lista está formada por cada letra que recorremos en el for
lista = [letra for letra in 'casa']
```

Ejemplo 2

Crear una lista con las potencias de 2 de los primeros 10 números:

```
# Método tradicional
lista = []
for numero in range(0,11):
    lista.append(numero**2)
print(lista)
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

```
# Con comprensión de listas
lista = [numero**2 for numero in  range(0,11)]
print(lista)
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

De este ejemplo podemos aprender que es posible modificar al vuelo los elementos que van a formar la lista.

Ejemplo 3

Crear una lista con los todos los múltiples de 2 entre 0 y 10:

```
# Método tradicional
lista = []
for numero in range(0,11):
    lista.append(numero**2)
print(lista)
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

```
# Método tradicional
lista = []
for numero in range(0,11):
    if numero % 2 == 0:
        lista.append(numero)
print(lista)
```

```
# Con comprensión de listas
lista = [numero for numero in range(0,11) if numero % 2 == 0 ]
print(lista)
```

```
# Añadir los números del 0 al 10 cuando su módulo de 2 sea 0
[numero for numero in range(0,11) if numero % 2 == 0 ]
```

En este caso podemos observar que incluso podemos marcar una condición justo al final para añadir o no el elemento en la lista.

Ejemplo 4

Crear una lista de pares a partir de otra lista creada con las potencias de 2 de los primeros 10 números:

```
# Método tradicional
lista = []
for numero in range(0,11):
```

```
    lista.append(numero**2)

pares = []
for numero in lista:
    if numero % 2 == 0:
        pares.append(numero)

print(pares)
```

```
[0, 4, 16, 36, 64, 100]
```

```
# Con comprensión de listas
lista = [numero for numero in
            [numero**2 for numero in range(0,11)]
                if numero % 2 == 0 ]
print(lista)
```

```
[0, 4, 16, 36, 64, 100]
```

Crear listas a partir de listas anidadas nos permite llevar la comprensión de listas al siguiente nivel y además no hay un límite.

Última edición: 6 de Octubre de 2018