Desconocido

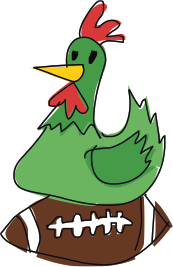
### Diccionarios en Python

En Python, un *dict* (abreviatura de *diccionario* ) es una colección de cosas, como las listas y las tuplas. La diferencia entre los dicts y las listas o tuplas es que cada elemento de un dict tiene una *clave* y un *valor* correspondiente *.*

Por ejemplo, supongamos que tenemos una lista de personas y sus deportes favoritos. Podríamos poner esta información en una lista Python, con el nombre de la persona seguido de su deporte, así

>>> favorite\_sports = ['Ralph Williams, Football',  
 'Michael Tippett, Basketball',  
 'Edward Elgar, Baseball',  
 'Rebecca Clarke, Netball',  
 'Ethel Smyth, Badminton',  
 'Frank Bridge, Rugby']

Si te pregunto cuál es el deporte favorito de Rebecca Clarke, podrías hojear la lista y encontrar que la respuesta es el netball. Pero, ¿y si la lista incluyera a 100 (o muchas más) personas?



Si almacenamos esta misma información en un diccionario, con el nombre de la persona como clave y su deporte favorito como valor, el código sería así:

>>> favorite\_sports = {'Ralph Williams' : 'Football',   
 'Michael Tippett' : 'Basketball',  
 'Edward Elgar' : 'Baseball',  
 'Rebecca Clarke' : 'Netball',  
 'Ethel Smyth' : 'Badminton',  
 'Frank Bridge' : 'Rugby'}

Utilizamos dos puntos para separar cada clave de su valor, y cada clave y cada valor van rodeados de comillas simples. Observa también que los elementos de un diccionario se encierran entre llaves ({}), no entre paréntesis o corchetes. El resultado es un dict (donde cada clave apunta a un valor concreto), como se muestra en [la Tabla 3-1 .](ch03.xhtml#ch03tab01)

Tabla**3-1:** Claves que apuntan a valores en un diccionario de deportes favoritos

| Clav**e** | Valo**r** |
| --- | --- |
| Ralph Williams | Fútbol |
| Michael Tippett | Baloncesto |
| Edward Elgar | Béisbol |
| Rebecca Clarke | Netball |
| Ethel Smyth | Bádminton |
| Frank Bridge | Rugby |

Ahora, para obtener el deporte favorito de Rebecca Clarke, accedemos a nuestro diccionario favorite\_sports utilizando su nombre como clave, así:

>>> print(favorite\_sports['Rebecca Clarke'])  
Netball

Y la respuesta es Netball.

Para eliminar un valor de un diccionario, utiliza su clave. Por ejemplo, eliminemos Ethel Smyth:

>>> del favorite\_sports['Ethel Smyth']  
>>> print(favorite\_sports)  
{'Rebecca Clarke': 'Netball', 'Michael Tippett': 'Basketball',   
'Ralph Williams': 'Football', 'Edward Elgar': 'Baseball',   
'Frank Bridge': 'Rugby'}

Para sustituir un valor en un dict, también utilizamos su clave. Supongamos que necesitamos cambiar el deporte favorito de Ralph Williams de Fútbol a Hockey sobre hielo. Podemos hacerlo así

>>> favorite\_sports['Ralph Williams'] = 'Ice Hockey'  
>>> print(favorite\_sports)  
{'Rebecca Clarke': 'Netball', 'Michael Tippett': 'Basketball',   
'Ralph Williams': 'Ice Hockey', 'Edward Elgar': 'Baseball',   
'Frank Bridge': 'Rugby'}

Sustituimos el deporte favorito Fútbol por Hockey sobre hielo utilizando la clave Ralph Williams.

Como ves, trabajar con diccionarios es parecido a trabajar con listas y tuplas, salvo que no puedes unir diccionarios con el operador más ( + ). Si intentas hacerlo, recibirás un mensaje de error, como en el ejemplo siguiente:

>>> favorite\_sports = {'Rebecca Clarke': 'Netball',  
 'Michael Tippett': 'Basketball',  
 'Ralph Williams': 'Ice Hockey',  
 'Edward Elgar': 'Baseball',  
 'Frank Bridge': 'Rugby'}  
>>> favorite\_colors = {'Malcolm Warner' : 'Pink polka dots',  
 'James Baxter' : 'Orange stripes',  
 'Sue Lee' : 'Purple paisley'}  
>>> favorite\_sports + favorite\_colors  
  
Traceback (most recent call last):  
File "<pyshell>", line 1, in <module>  
 favorite\_sports + favorite\_colors  
TypeError: unsupported operand type(s) for +: 'dict' and 'dict'

Unir diccionarios no tiene sentido para Python, así que se da por vencido.

[anterior](ch03_11.html)[Subtema 12 de 14: (Ver todo)](ch03.html)[siguiente](ch03_13.html)