Python Dictionaries

❮ Previous Next ❯

Dictionary

A dictionary is a collection which is unordered, changeable and indexed. In Python dictionaries are written with curly brackets, and they have keys and values.

Example

Create and print a dictionary:

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(thisdict)
```

Run example »

Accessing Items

You can access the items of a dictionary by referring to its key name:

Example

Get the value of the "model" key:

Run example »

There is also a method called `get()` that will give you the same result:

Example

Get the value of the "model" key:

```
x = thisdict.get("model")
```

Run example »

Change Values

You can change the value of a specific item by referring to its key name:

Example

Change the "year" to 2018:

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict["year"] = 2018
```

Run example »

Loop Through a Dictionary

You can loop through a dictionary by using a `for` loop.

Example

Print all key names in the dictionary, one by one:

```
for x in thisdict:
  print(x)
```

Run example »

Example

Print all *values* in the dictionary, one by one:

```
for x in thisdict:
  print(thisdict[x])
```

Run example »

Example

You can also use the `values()` function to return values of a dictionary:

```
for x in thisdict.values():
  print(x)
```

Run example »

Example

Loop through both *keys* and *values*, by using the `items()` function:

```
for x, y in thisdict.items():
  print(x, y)
```

Dictionary Length

To determine how many items (key-value pairs) a dictionary have, use the `len()` method.

Example

Print the number of items in the dictionary:

```
print(len(thisdict))
```

Run example »

Adding Items

Adding an item to the dictionary is done by using a new index key and assigning a value to it:

Example

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict["color"] = "red"
print(thisdict)
```

Run example »

Removing Items

There are several methods to remove items from a dictionary:

The `del` keyword removes the item with the specified key name:

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
del thisdict["model"]
print(thisdict)
```

Run example »

Example

The `pop()` method removes the item with the specified key name:

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict.pop("model")
print(thisdict)
```

Run example »

Example

The `popitem()` method removes the last inserted item (in versions before 3.7, a random item is removed instead):

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
```

Run example »

Example

The `del` keyword removes the item with the specified key name:

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
del thisdict["model"]
print(thisdict)
```

Run example »

Example

The `del` keyword can also delete the dictionary completely:

```
thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
del thisdict
print(thisdict) #this will cause an error because "thisdict" no longer
exists.
```

Run example »

Example

The `clear()` keyword empties the dictionary:

```
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict.clear()
print(thisdict)
```

Run example »

The dict() Constructor

It is also possible to use the `dict()` constructor to make a dictionary:

Example

```
thisdict = dict(brand="Ford", model="Mustang", year=1964)
# note that keywords are not string literals
# note the use of equals rather than colon for the assignment
print(thisdict)
```

Run example »

Dictionary Methods

Python has a set of built-in methods that you can use on dictionaries.

Method	Description
clear()	Removes all the elements from the dictionary
copy()	Returns a copy of the dictionary
fromkeys()	Returns a dictionary with the specified keys and values
get()	Returns the value of the specified key
items()	Returns a list containing the a tuple for each key value pair

pop()	Removes the element with the specified key
popitem()	Removes the last inserted key-value pair
setdefault()	Returns the value of the specified key. If the key does not exist: insert the key, with the specified value
update()	Updates the dictionary with the specified key-value pairs
values()	Returns a list of all the values in the dictionary

Test Yourself With Exercises

Exercise:

Use the `get` method to print the value of the "model" key of the `car` dictionary.

```
car =   {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(                )
```

Submit Answer »

❮ Previous Next ❯

Top 10 Tutorials

HTML Tutorial
CSS Tutorial
JavaScript Tutorial
How To Tutorial
W3.CSS Tutorial
Bootstrap Tutorial
SQL Tutorial
PHP Tutorial
jQuery Tutorial
Python Tutorial

Top 10 References

HTML Reference
CSS Reference
JavaScript Reference
W3.CSS Reference
Bootstrap Reference
SQL Reference
PHP Reference
HTML Colors
jQuery Reference
Python Reference

Top 10 Examples

HTML Examples
CSS Examples
JavaScript Examples
How To Examples
W3.CSS Examples
Bootstrap Examples
PHP Examples
jQuery Examples
Angular Examples
XML Examples

Web Certificates

HTML Certificate
CSS Certificate
JavaScript Certificate
jQuery Certificate
PHP Certificate
Bootstrap Certificate
XML Certificate