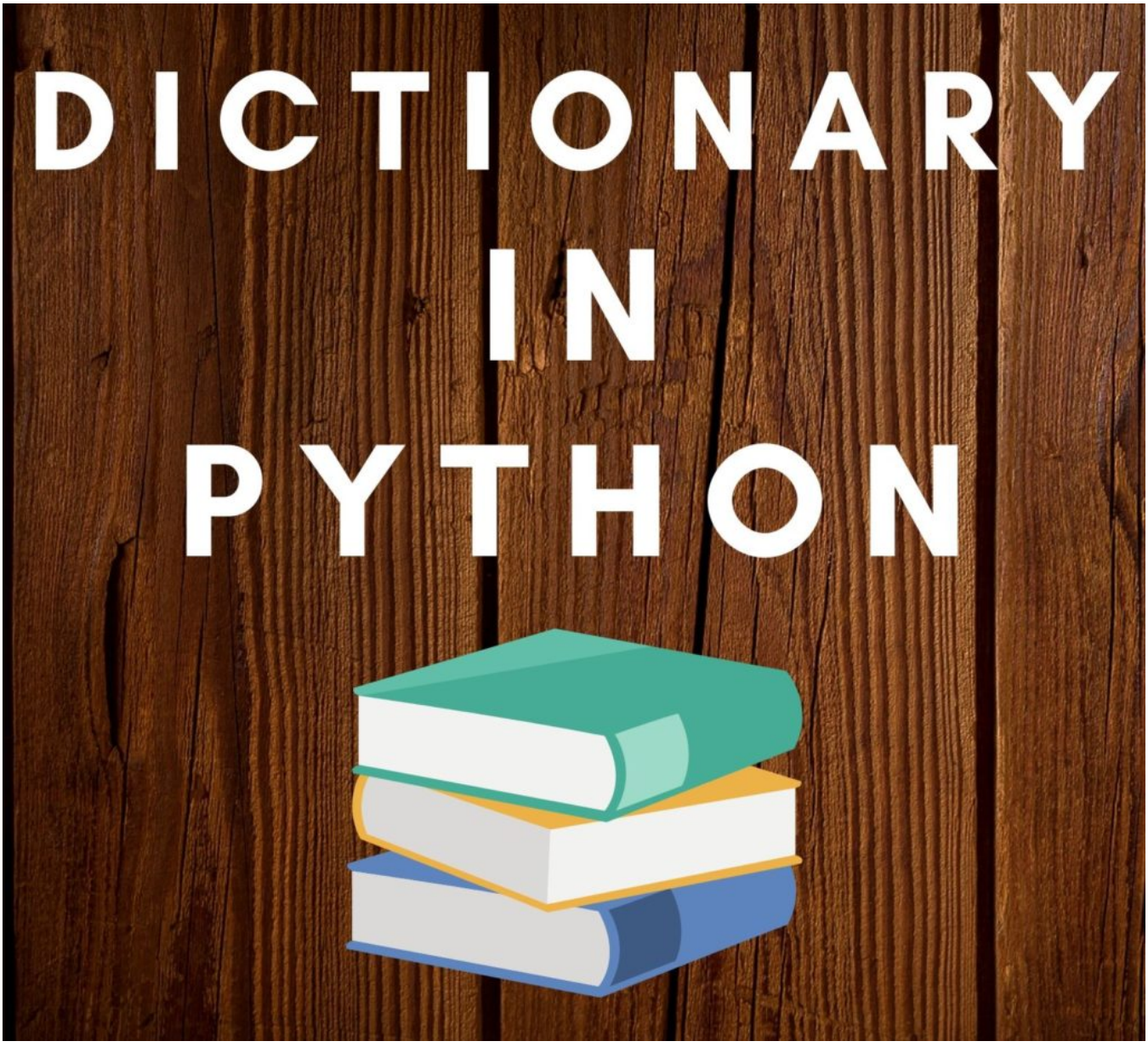
Dictionary in Python-Complete Tutorial

Python / March 19, 2020



This **Dictionary in Python-Complete Tutorial** is made for you to give complete knowledge of the Dictionary in Python. At the end of this tutorial, you will have full knowledge of the **Dictionary in Python**.

Hello, & Welcome! to the **Dictionary in Python-Complete Tutorial**.

Part 11- Dictionary in Python-Complete Tutorial

In this article of Python Tutorial, you will learn following-

1. What is Dictionary in Python?
2. How to Create a Dictionary in Python?
3. Accessing Elements from Dictionary.
4. How to check the availability of a key in the Dictionary?
5. Typecasting of Keys and Values into List.
6. Python Dictionary Append.
7. Removing Elements from Dictionary.
8. Python Nested Dictionary.
9. Update value in Dictionary.
10. Copy a Dictionary.

What is Dictionary in Python?

Dictionary is a new data type and its a mapping type, which means it is a set of key, value pairs. Dictionary is an unordered collection of data values. A Dictionary is changeable and indexed.

How to Create a Dictionary in Python?

For creating a dictionary, a curly bracket {} is used. Let's see how to create a dictionary in python-

```
ninja_belts= {"crystal":"red","ryu":"black"}
print(ninja_belts)

OUTPUT-
{'crystal' : 'red', 'ryu' : 'black'}
```

Here, “crystal” is a **Key** and “red” is **value**. Key and value are put into ” ” because values are by default strings.

Values can be of any data type and it can repeat, but a key must be unique and fixed type.

You can create a dictionary by using a built-in function dict(), like that-

```
dict= {}
```

Accessing Elements from Dictionary.

If you want to access value from a dictionary, you can do that by **Key**, whatever the key is assigned to the value. This key can be used inside a square bracket [] or with the get() method.

Let’s have a look at the example of a **square bracket []** method-

```
ninja_belts= {"crystal":"red","ryu":"black"}
print(ninja_belts['ryu'])

OUTPUT-
red
```

Let’s have a look at an example of **get()** method-

```
ninja_belts= {"crystal":"red","ryu":"black"}
print(ninja_belts.get('ryu'))

OUTPUT-
red
```

How to check the availability of a Key in the Dictionary?

If you want to check that that certain key is present in the dictionary or not, then you need to use an '**in**' keyword in python.

The syntax for checking a key in a dictionary-

```
key in dictionary
```

-> Here, a **key** may be any particular key name, which we are looking for.

-> **in** is a keyword in python.

-> **Dictionary** is a particular dictionary in which we are checking.

Let's understand with the help of an example-

```
>>>ninja_belts= {"crystal":"red","ryu":"black"}
>>>'yoshi' in ninja_belts

OUTPUT-
False
```

Here, we are getting **False** as an output because **'Yoshi'** is not present in the dictionary **ninja_belts**.

Another Method-

There is one more method for checking a key in a dictionary. Let's see in the example below-

```
>>>ninja_belts= {"crystal":"red","ryu":"black"}
>>> ninja_belts.keys()

OUTPUT-
dict_keys(['crystal', 'ryu'])
```

Here, we are getting the full list of **keys** in the '**ninja_belts**' dictionary. Therefore, you can check the key for which you are looking for,

Typecasting of Keys and Values into List.

If you want to typecast keys of a dictionary into a list, so you can do it by using a list keyword.

Let's see in the example–

```
>>>ninja_belts= {"crystal":"red","ryu":"black"}
>>> list(ninja_belts.keys())

OUTPUT-
['crystal', 'ryu']
```

Here, we are getting all the keys in the form of a list in output.

You can also typecast the **values of a dictionary** into a list. First, we get all the values of the dictionary and then store them in a variable in the form of a list.

Let's see in the example–

```
>>>ninja_belts= {"crystal":"red","ryu":"black"}
>>> ninja_belts.values()

output-
dict_values(['red', 'black'])
```

Here, we are getting all the values of the dictionary, now we store values in the variable in the form of a list.

Let's see in the example below–

```
>>> val=list(ninja_belts.values())
>>> val

OUTPUT-
['red', 'black']
```

Here, '**val**', is the variable in which we are storing our values in the form of a list.

By doing this, you can typecast keys and values into the list.

Now, you can work on that list, like you can count how many instances in values, which means you can check how many times one value is used.

Let's see in the example–

```
>>> vals.count('black')

OUTPUT-
1
```

The output is 1 because 'black' is used only 1 time in the dictionary.

Python Dictionary Append.

You can append a new key into the dictionary very easily. Without wasting your time, Let's see in the example–

```
>>> ninja_belts['Yoshi']='red'
>>> ninja_belts

OUTPUT-
{'crystal': 'red', 'ryu': 'black', 'Yoshi': 'red'}
```

Here, we have added a new **key as 'Yoshi'** and **value as 'red'**.

Removing Elements from Dictionary.

You can remove elements from the dictionary in various ways. We will discuss approx all the ways to remove an item from the dictionary.

Using pop() method-

The pop() method is used to remove the item with a particular key name. Let's see in the example-

```
>>>ninja_belts= {"crystal":"red","ryu":"black","Yoshi":"red"}
>>>ninja_belts.pop("Yoshi")
>>>ninja_belts

OUTPUT-
{'crystal': 'red', 'ryu': 'black'}
```

Using popitem() method-

The popitem() method item removes only the last item from the dictionary. Let's see in the example-

```
>>>ninja_belts= {"crystal":"red","ryu":"black","Yoshi":"red"}
>>>ninja_belts.popitem()
>>>ninja_belts

OUTPUT-
{'crystal': 'red', 'ryu': 'black'}
```

Using del Keyword-

The 'del' keyword deletes the element with the particular key name. Let's see in the example-

```
>>>ninja_belts= {"crystal":"red","ryu":"black","Yoshi":"red"}
>>>del ninja_belts["Yoshi"]
>>>ninja_belts

OUTPUT-
{'crystal': 'red', 'ryu': 'black'}
```

The “del” Keyword is also used for deleting the whole dictionary. Let’s see in the example-

```
>>> del ninja_belts
>>> ninja_belts

OUTPUT-
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'ninja_belts' is not defined
```

Using clear() method-

By using a clear() method, you can clear your dictionary, which means it makes your dictionary empty. Let’s see in the example-

```
>>>ninja_belts= {"crystal":"red","ryu":"black","Yoshi":"red"}
>>>ninja_belts.clear()
>>>ninja_belts

OUTPUT-
{}
```

Python Nested Dictionary.

You can create a nested dictionary in Python. A nested dictionary means, dictionary inside another dictionary.

Let's see how to create a nested dictionary in python-

```
>>>my_dict={1:"hello",2:"I",3:{'A':"am","B":"Alex","C":"John"}}
>>>my_dict

OUTPUT-
{1: 'hello', 2: 'I', 3: {'A': 'am', 'B': 'Alex', 'C': 'John'}}
```

Here, we first created a **my_dict** dictionary inside **my_dict**, we have created another dictionary as **3:{'A':"am","B":"Alex","C":"John"}**.

Update value in Dictionary.

You can change or update the value of the dictionary by pointing to its key name. Let's see in the example how to update the value in the dictionary.

```
>>>ninja_belts= {"crystal":"red","ryu":"black","Yoshi":"red"}
>>>ninja_belts["Yoshi"]="green"
>>> ninja_belts

OUTPUT-
{'crystal': 'red', 'ryu': 'black', 'Yoshi': 'green'}
```

Copy a Dictionary.

If you want to make a copy of your dictionary, then you can do it by a copy() method. Let's have a look in the example-

```
>>>ninja_belts= {"crystal":"red","ryu":"black","Yoshi":"red"}
>>>new_ninja_belts=ninja_belts.copy()
>>>new_ninja_belts

OUTPUT-
{'crystal': 'red', 'ryu': 'black', 'Yoshi': 'red'}
```

Here, we are copying our "**ninja_belts**" dictionary into the **"new_ninja_belts"** dictionary by just using a **copy()** method.

One more method to copy-

You can copy your dictionary by one more method that is a built-in method in python **dict()**. By using **dict()** method you can copy your dictionary. Let's see in the example-

```
>>>ninja_belts= {"crystal":"red","ryu":"black","Yoshi":"red"}
>>>new_ninja_belts=dict(ninja_belts)
>>>new_ninja_belts

OUTPUT-
{'crystal': 'red', 'ryu': 'black', 'Yoshi': 'red'}
```

That's all for the **Dictionary in Python**. I hope now you have a better understanding of the **Dictionary in Python**.

Congratulations! You successfully learned the **Dictionary in** Python.

In the next tutorial, we will start learning **Sets in Python**.

Till then, Enjoy Learning Python!

All the Best!

Are you ML Beginner and confused, from where to start ML, then read my BLOG – How do I learn Machine Learning?

If you are looking for Machine Learning Algorithms, then read my Blog – Top 5 Machine Learning Algorithm.

If you are wondering about Machine Learning, read this Blog- What is Machine Learning?

Thank YOU!

Though of the Day...

> *"Live as if you were to die tomorrow. Learn as if you were to live forever."*
>
> *– Mahatma Gandhi*

← Previous Post Next Post →

Leave a Comment

Your email address will not be published. Required fields are marked *

Type here..

Name*

Email*

Website

Save my name, email, and website in this browser for the next time I comment.

Post Comment »

Follow Us on Twitter-

Follow us on Pinterest-

Our Facebook Group-