[Dictionary](http://www.pythonforbeginners.com/dictionary/) ***4***

This is a new serie of articles here at Python for beginners, that are supposed

to be a starting point for completely beginners of Python.

See it as a cheat sheet, reference, manual or whatever you want.

The purpose is to very short write down the basics of Python.

This page will describe how to use dictionaries in Python.

What is a Dictionary?

Dictionary is another data type in Python.

Dictionaries are collections of items that have a "key" and a "value".

Python dictionaries are also known as associative arrays or hash tables.

They are just like lists, except instead of having an assigned index number,  
you make up the index.

Dictionaries are unordered, so the order that the keys are added doesn’t

necessarily reflect what order they may be reported back.

Use {} curly brackets to construct the dictionary.

Look up the value associated with a key using []

Provide a key and a value.

A colon is placed between key and value (key:value)

Each key must be unique and each key can be in the dictionary only once.

How to create a Dictionary?

To create a dictionary, provide the key and the value and make sure that each

pair is separated by a comma

# This is a list

mylist = ["first","second","third"]

# This is a dictionary

mydictionary = {0:"first",1:"second",2:"third"}

**Let’s create some random dictionaries:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | # Empty declaration + assignment of key-value pair  emptyDict = {}  emptyDict['key4']=’value4′    # Create a three items dictionary  x = {"one":1,"two":2,"three":3}    #The name of the dictionary can be anything you like  dict1 = {'abc': 456};  dict2 = {'abc':123,98.6:37}; |

Accessing / Getting values

To access dictionary elements, you can use the square brackets along with the key

to obtain its value.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | data = {'Name':'Zara','Age':7,'Class':'First'};    # Get all keys  data.keys()    # Get all values  data.values()    # Print key1  print data['Name']    # Prints 7  print data['Age']    # Prints name and age  print 'Name', data['Name'];  print 'Age', data['Age']; |

Looping through a Dictionary

A for loop on a dictionary iterates over its keys by default.

The keys will appear in an arbitrary order.

The methods dict.keys() and dict.values() return lists of the keys or values

explicitly.

There's also an items() which returns a list of (key, value) tuples, which is the

most efficient way to examine all the key value data in the dictionary.

All of these lists can be passed to the sorted() function.

Basic syntax for looping through dictionaries (keys and values)

for key in data:

print data[key]

**Let’s say that you have a dictionary called “data”**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | data = {          'key1': 'value1',          'key2': 'value2',          'key3': 'value3'          }    for key, value in data.items():      print key,value    Looping their values directory (not in order)  for value in data.values():      print value |

Updating a Dictionary

Creates a new entry if the key is not already in dictionary.

Overwrites the previous value if the key is already present.

You can update a dictionary by:

adding a new entry or item (i.e., a key-value pair)

modifying an existing entry

deleting an existing entry

**Let’s see how that works:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | data = {'Name':'Zara','Age':7,'Class':'First'};    data['Age'] = 8;                    # update existing entry  data['School'] = "DPS School";      # Add new entry    print "data['Age']: ", data['Age'];  print "data['School']: ", data['School']; |

**Let’s take one more example:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | birthday = {}  birthday['Darwin'] = 1809  birthday['Newton'] = 1942  # oops  birthday['Newton'] = 1642  print birthday |

Delete a key / value

The "del" operator does deletions.

In the simplest case, it can remove the definition of a variable, as if that variable  
had not been defined.

Del can also be used on list elements or slices to delete that part of the list and to  
delete entries from a dictionary.

|  |  |
| --- | --- |
| 1  2  3 | data = {'a':1,'b':2,'c':3}  del dict['b']   ## Delete 'b' entry  print dict      ## {'a':1, 'c':3} |

Example from Google Class

Let see this example from <https://developers.google.com/edu/python/dict-files>

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | ## Can build up a dict by starting with the the empty dict {}  ## and storing key/value pairs into the dict like this:  ## dict[key] = value-for-that-key    dict = {}  dict['a'] = 'alpha'  dict['g'] = 'gamma'  dict['o'] = 'omega'    print dict          ## {'a': 'alpha', 'o': 'omega', 'g': 'gamma'}    print dict['a']     ## Simple lookup, returns 'alpha'    dict['a'] = 6       ## Put new key/value into dict    'a' in dict         ## True    ## print dict['z']                  ## Throws KeyError    if 'z' in dict: print dict['z']     ## Avoid KeyError  print dict.get('z')                 ## None (instead of KeyError) |

A for loop on a dictionary iterates over its keys by default.

The keys will appear in an arbitrary order.

The methods dict.keys() and dict.values() return lists of the keys or values  
explicitly.

There's also an items() which returns a list of (key, value) tuples, which is the  
most efficient way to examine all the key value data in the dictionary.

All of these lists can be passed to the sorted() function.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | ## Note that the keys are in a random order.  for key in dict: print key  ## prints a g o    ## Exactly the same as above  for key in dict.keys(): print key    ## Get the .keys() list:  print dict.keys()  ## ['a', 'o', 'g']    ## Likewise, there's a .values() list of values  print dict.values()  ## ['alpha', 'omega', 'gamma']    ## Common case -- loop over the keys in sorted order,  ## accessing each key/value  for key in sorted(dict.keys()):      print key, dict[key]    ## .items() is the dict expressed as (key, value) tuples  print dict.items()  ##  [('a', 'alpha'), ('o', 'omega'), ('g', 'gamma')]    ## This loop syntax accesses the whole dict by looping  ## over the .items() tuple list, accessing one (key, value)  ## pair on each iteration.  for k, v in dict.items(): print k, '>', v  ## a > alpha    o > omega     g > gamma |

Dict Formatting

The % operator works conveniently to substitute values from a dict into a string by name

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | hash = {}  hash['word'] = 'garfield'  hash['count'] = 42  s = 'I want %(count)d copies of %(word)s' % hash  # %d for int, %s for string    # Will give you:  >> 'I want 42 copies of garfield' |

Common Dictionary Operations

# create an empty dictionary

x = {}

# create a three items dictionary

x = {"one":1, "two":2, "three":3}

# get a list of all the keys

x.keys()

# get a list of all the values

x.values()

# add an entry

x["four"]=4

# change an entry

x["one"] = "uno"

# delete an entry

del x["four"]

# make a copy

y = x.copy()

# remove all items

x.clear()

#number of items

z = len(x)

# test if has key

z = x.has\_key("one")

# looping over keys

for item in x.keys(): print item

# looping over values

for item in x.values(): print item

# using the if statement to get the values

if "one" in x:

print x['one']

if "two" not in x:

print "Two not found"

if "three" in x:

del x['three']

Sources

[https://developers.google.com/edu/python/dict-files](https://developers.google.com/edu/python/dict-files" \o "google" \t "_blank)

[http://docs.python.org/2/tutorial/datastructures.html](http://www.pythonforbeginners.com/dictionary/onclick=)

[http://software-carpentry.org/3\_0/py04.html](http://software-carpentry.org/3_0/py04.html" \o "software" \t "_blank)

[http://yuji.wordpress.com](http://www.pythonforbeginners.com/dictionary/onclick=)

<http://www.tutorialspoint.com/python/python_dictionary.htm>