

Dictionary

Python dictionary is an unordered collection of items. While other compound data types have only value as an element, a dictionary has a key: value pair.

Dict Creation ¶

```
In [0]: #empty dictionary
my_dict = {}

#dictionary with integer keys
my_dict = {1: 'abc', 2: 'xyz'}
print(my_dict)

#dictionary with mixed keys
my_dict = {'name': 'satish', 1: ['abc', 'xyz']}
print(my_dict)

#create empty dictionary using dict()
my_dict = dict()

my_dict = dict([(1, 'abc'), (2, 'xyz')])    #create a dict with list of tuples
print(my_dict)

{1: 'abc', 2: 'xyz'}
{'name': 'satish', 1: ['abc', 'xyz']}
{1: 'abc', 2: 'xyz'}
```

Dict Access

```
In [0]: my_dict = {'name': 'satish', 'age': 27, 'address': 'guntur'}

#get name
print(my_dict['name'])

satish
```

```
In [0]: #if key is not present it gives KeyError  
print(my_dict['degree'])
```

```
-----  
KeyError                                Traceback (most recent call last)  
<ipython-input-5-c5aba24e1656> in <module>()  
      1 #if key is not present it gives KeyError  
----> 2 print(my_dict['degree'])  
  
KeyError: 'degree'
```

```
In [0]: #another way of accessing key  
print(my_dict.get('address'))
```

guntur

```
In [0]: #if key is not present it will give None using get method  
print(my_dict.get('degree'))
```

None

Dict Add or Modify Elements

```
In [0]: my_dict = {'name': 'satish', 'age': 27, 'address': 'guntur'}
```

```
#update name  
my_dict['name'] = 'raju'  
  
print(my_dict)
```

{'name': 'raju', 'age': 27, 'address': 'guntur'}

```
In [0]: #add new key  
my_dict['degree'] = 'M.Tech'  
  
print(my_dict)
```

{'name': 'raju', 'age': 27, 'address': 'guntur', 'degree': 'M.Tech'}

Dict Delete or Remove Element

```
In [0]: #create a dictionary
my_dict = {'name': 'satish', 'age': 27, 'address': 'guntur'}

#remove a particular item
print(my_dict.pop('age'))

print(my_dict)

27
{'name': 'satish', 'address': 'guntur'}
```

```
In [0]: my_dict = {'name': 'satish', 'age': 27, 'address': 'guntur'}

#remove an arbitarty key
my_dict.popitem()

print(my_dict)

{'name': 'satish', 'age': 27}
```

```
In [0]: squares = {2: 4, 3: 9, 4: 16, 5: 25}

#delete particular key
del squares[2]

print(squares)

{3: 9, 4: 16, 5: 25}
```

```
In [0]: #remove all items
squares.clear()

print(squares)

{}
```

```
In [0]: squares = {2: 4, 3: 9, 4: 16, 5: 25}

#delete dictionary itself
del squares

print(squares) #NameError because dict is deleted
```

```
-----
NameError                                Traceback (most recent call last)
<ipython-input-16-355e8277492b> in <module>()
      4 del squares
      5
----> 6 print(squares) #NameError because dict is deleted

NameError: name 'squares' is not defined
```

Dictionary Methods

```
In [0]: squares = {2: 4, 3: 9, 4: 16, 5: 25}
```

```
my_dict = squares.copy()
print(my_dict)
```

```
{2: 4, 3: 9, 4: 16, 5: 25}
```

```
In [0]: #fromkeys[seq[, v]] -> Return a new dictionary with keys from seq and value equal to v (defaults to None).
```

```
subjects = {}.fromkeys(['Math', 'English', 'Hindi'], 0)
print(subjects)
```

```
{'Math': 0, 'English': 0, 'Hindi': 0}
```

```
In [0]: subjects = {2:4, 3:9, 4:16, 5:25}
```

```
print(subjects.items()) #return a new view of the dictionary items (key, value)
```

```
dict_items([(2, 4), (3, 9), (4, 16), (5, 25)])
```

```
In [0]: subjects = {2:4, 3:9, 4:16, 5:25}
```

```
print(subjects.keys()) #return a new view of the dictionary keys
```

```
dict_keys([2, 3, 4, 5])
```

```
In [0]: subjects = {2:4, 3:9, 4:16, 5:25}
```

```
print(subjects.values()) #return a new view of the dictionary values
```

```
dict_values([4, 9, 16, 25])
```

```
In [0]: #get list of all available methods and attributes of dictionary
```

```
d = {}
print(dir(d))
```

```
['__class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__setitem__', '__sizeof__', '__str__', '__subclasshook__', 'clear', 'copy', 'fromkeys', 'get', 'items', 'keys', 'pop', 'popitem', 'setdefault', 'update', 'values']
```

Dict Comprehension

```
In [0]: #Dict comprehensions are just like list comprehensions but for dictionaries
```

```
d = {'a': 1, 'b': 2, 'c': 3}
for pair in d.items():
    print(pair)
```

```
('a', 1)
```

```
('c', 3)
```

```
('b', 2)
```

```
In [0]: #Creating a new dictionary with only pairs where the value is larger than 2  
d = {'a': 1, 'b': 2, 'c': 3, 'd': 4}  
new_dict = {k:v for k, v in d.items() if v > 2}  
print(new_dict)  
  
{'c': 3, 'd': 4}
```

```
In [0]: #We can also perform operations on the key value pairs  
d = {'a':1,'b':2,'c':3,'d':4,'e':5}  
d = {k + 'c':v * 2 for k, v in d.items() if v > 2}  
print(d)  
  
{'cc': 6, 'dc': 8, 'ec': 10}
```