

Introduction to Python Dictionary

Python Dictionary

Dictionary in Python is an unordered collection of data values, used to store data values like a map, which unlike other Data Types that hold only single value as an element, Dictionary holds **key:value** pair. Key value is provided in the dictionary to make it more optimized.

Note – Keys in a dictionary doesn't allows Polymorphism.\

Each key is separated from its value by a colon (:), the items are separated by commas, and the whole thing is enclosed in curly braces. An empty dictionary without any items is written with just two curly braces, like this: {}.

Creating a Dictionary

In Python, a Dictionary can be created by placing sequence of elements within curly {} braces, separated by 'comma'. Dictionary holds a pair of values, one being the Key and the other corresponding pair element being its **Key:value**. Values in a dictionary can be of any datatype and can be duplicated, whereas keys can't be repeated and must be *immutable*.

```
# Creating a Dictionary
```

```
# with Integer Keys
```

```
Dict = {1: 'Geeks', 2: 'For', 3: 'Geeks'}
```

```
print("\nDictionary with the use of Integer Keys: ")
```

```
print(Dict)
```

```
# Creating a Dictionary
```

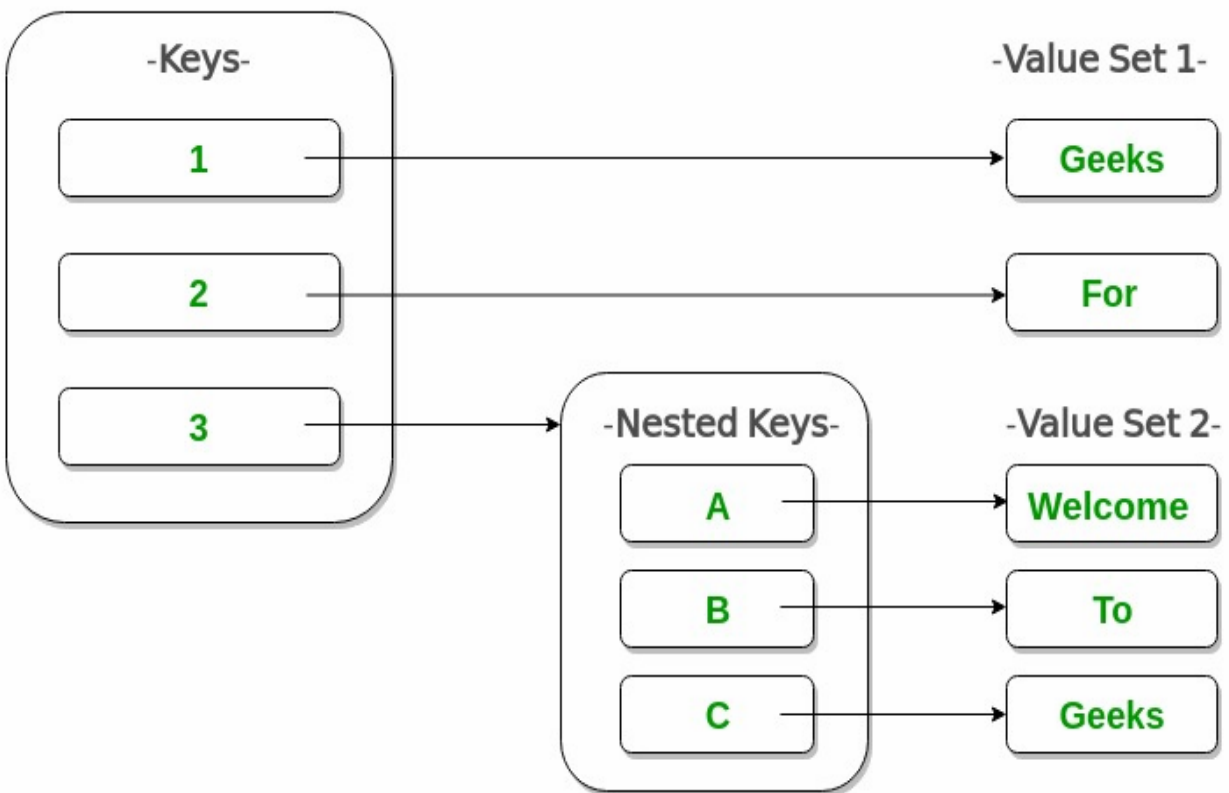
```
# with Mixed keys
```

```
Dict = {'Name': 'Geeks', 1: [1, 2, 3, 4]}
```

```
print("\nDictionary with the use of Mixed Keys: ")
```

```
print(Dict)
```

Nested Dictionary:



Creating a Nested Dictionary

as shown in the below image

```
Dict = {1: 'Geeks', 2: 'For',
        3: {'A': 'Welcome', 'B': 'To', 'C': 'Geeks'}}
```

```
print(Dict)
```

Output:

```
{1: 'Geeks', 2: 'For', 3: {'A': 'Welcome', 'B': 'To', 'C': 'Geeks'}}
```

Accessing Values in Dictionary

To access dictionary elements, you can use the familiar square brackets along with the key to obtain its value

```
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}  
print "dict['Name']: ", dict['Name']  
print "dict['Age']: ", dict['Age']
```

When the above code is executed, it produces the following result –

```
dict['Name']: Zara  
dict['Age']: 7
```

Updating Dictionary

You can update a dictionary by adding a new entry or a key-value pair, modifying an existing entry, or deleting an existing entry as shown below in the simple example –

```
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}  
dict['Age'] = 8; # update existing entry  
dict['School'] = "DPS School"; # Add new entry  
  
print "dict['Age']: ", dict['Age']  
print "dict['School']: ", dict['School']
```

When the above code is executed, it produces the following result –

```
dict['Age']: 8  
dict['School']: DPS School
```

-Delete Dictionary Elements

You can either remove individual dictionary elements or clear the entire contents of a dictionary. You can also delete entire dictionary in a single operation.

To explicitly remove an entire dictionary, just use the **del** statement. Following is a simple example –

```
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}  
del dict['Name']; # remove entry with key 'Name'  
dict.clear();    # remove all entries in dict  
del dict ;       # delete entire dictionary  
  
print "dict['Age']: ", dict['Age']  
print "dict['School']: ", dict['School']
```