Data Scientist
Codecademy
Python Fundamentals
Dictionary

Introduction to Dictionary

A dictionary is an unordered set of key: value pairs.

It provides us a way to map pieces of data to each other so that we can quickly find values that are associated with one another.

- a dictionary begins and ends with curly braces {}
- each item consists of a key and a value
- each key : value pair is separated by a comma

Invalid Keys

We can have a list or dictionary as a value of an item in a dictionary, but we cannot use these data types as keys of the dictionary. Trying to do so will return TypeError: unhashable type.

Dictionaries in Python rely on each key having a hash value, a specific identifier for the key. If key can change, that hash value would not be reliable. So keys must always be unchangeable, hashable data types, like numbers or strings.

Empty Dictionary

A dictionary doesn't have to contain anything.

Add a Key

To add a single key: value pair we can use the syntax:

```
dictionary[key] = value
```

Add Multiple Keys

If we want to add multiple key: value pairs at once, we can use the .update() method.

Overwrite Values

When we add a key: value pair using this syntax:

```
dictionary[key] = value
```

if that key already exists in the dictionary then we overwrite its previous value with the new value we just added.

Dict Comprehensions

Python allows us to create a dictionary using a dict comprehension, with this syntax

```
dictionary = {key:value for key, value in zip(list1,list2)}
```

zip() combines two lists into a iterator of tuples with the list elements paired together

Get a key

Once a dictionary is defined we can use keys to access values in a dictionary.

Get an Invalid key

If we try to access a key which does not exists then we get keyError. We can avoid this by first checking if the key exists in the dictionary using ‘in’ keyword.

```
if key in dict: print(dict[key])
```

Try/ Except to Get a Key

```
key_to_check = “some_key”
try:
```

```
        print(dict[key_to_check])
    except KeyError:
        print("That key doesn't exist!")
```

When we try to access a key that doesn't exist, the program will fo into except block and print appropriate message.

Safely Get a key

Dictionaries have a .get() method to search for a value instead of the dict[key] notation we have been using. If the key does not exists it returns None by default.

We can specify a default value to be returned if a key does not exists by passing it as the second argument to .get() method.

Delete a key

We can use .pop() to remove a key : value pair from a dictionary. We can provide a default value to return if the key does not exist in the dictionary.

Get all keys

We can use list() function to get a list of all the keys in dictionary.

Dictionaries also have a .keys() method which returns a dict_keys object. The dict_keys object is view only and cannot be modified in any manner.

Get all Values

Dictionaries have a .values() method that returns a dict_values object with all of the values in the dictionary.

Get all Items

We can get both the keys and the values with the .items() method. It returns a dict_list object. Each element of the object is tuple.