PY0101EN-2-3-Sets

January 17, 2022

1 Sets in Python

Estimated time needed: **20** minutes

1.1 Objectives

After completing this lab you will be able to:

- Work with sets in Python, including operations and logic operations.

Table of Contents

```
<ul>
    <li>
        <a href="https://#set">Sets</a>
        <ul>
            <li><a href="https://content/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_c
            <li><a href="op">Set Operations</a></li>
            <li><a href="https://logic/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_co
        </ul>
    </li>
    <li>
        <a href="https://#quiz">Quiz on Sets</a>
    </li>
</ul>
```

Sets

Set Content

A set is a unique collection of objects in Python. You can denote a set with a pair of curly brackets {}. Python will automatically remove duplicate items:

[5]:
```
# Create a set

set1 = {"pop", "rock", "soul", "hard rock", "rock", "R&B", "rock", "disco"}
set1
print(set1)
```

```
{'disco', 'soul', 'pop', 'R&B', 'hard rock', 'rock'}
```

The process of mapping is illustrated in the figure:

You can also create a set from a list as follows:

```
[3]: # Convert list to set

     album_list = [ "Michael Jackson", "Thriller", 1982, "00:42:19", \
                   "Pop, Rock, R&B", 46.0, 65, "30-Nov-82", None, 10.0]
     album_set = set(album_list)
     album_set
     print(album_set)
     print(type(None))
```

```
{65, '30-Nov-82', 10.0, 'Thriller', 46.0, '00:42:19', None, 'Michael Jackson',
'Pop, Rock, R&B', 1982}
<class 'NoneType'>
```

Now let us create a set of genres:

```
[7]: # Convert list to set

     music_genres = set(["pop", "pop", "rock", "folk rock", "hard rock", "soul", \
                         "progressive rock", "soft rock", "R&B", "disco"])
     music_genres
```

```
[7]: {'R&B',
      'disco',
      'folk rock',
      'hard rock',
      'pop',
      'progressive rock',
      'rock',
      'soft rock',
      'soul'}
```

Set Operations

Let us go over set operations, as these can be used to change the set. Consider the set A:

```
[8]: # Sample set

     A = set(["Thriller", "Back in Black", "AC/DC"])
     A
```

```
[8]: {'AC/DC', 'Back in Black', 'Thriller'}
```

We can add an element to a set using the add() method:

```
[9]: # Add element to set
```

```
A.add("NSYNC")
A
```

[9]:
```
{'AC/DC', 'Back in Black', 'NSYNC', 'Thriller'}
```

If we add the same element twice, nothing will happen as there can be no duplicates in a set:

[10]:
```
# Try to add duplicate element to the set

A.add("NSYNC")
A
```

[10]:
```
{'AC/DC', 'Back in Black', 'NSYNC', 'Thriller'}
```

We can remove an item from a set using the remove method:

[11]:
```
# Remove the element from set

A.remove("NSYNC")
A
```

[11]:
```
{'AC/DC', 'Back in Black', 'Thriller'}
```

We can verify if an element is in the set using the in command:

[12]:
```
# Verify if the element is in the set

"AC/DC" in A
```

[12]:
```
True
```

Sets Logic Operations

Remember that with sets you can check the difference between sets, as well as the symmetric difference, intersection, and union:

Consider the following two sets:

[13]:
```
# Sample Sets

album_set1 = set(["Thriller", 'AC/DC', 'Back in Black'])
album_set2 = set([ "AC/DC", "Back in Black", "The Dark Side of the Moon"])
```

[16]:
```
# Print two sets

album_set1, album_set2
print(album_set1, album_set2)
```

```
{'Back in Black', 'AC/DC', 'Thriller'} {'Back in Black', 'AC/DC', 'The Dark Side
of the Moon'}
```

As both sets contain AC/DC and Back in Black we represent these common elements with the intersection of two circles.

You can find the intersect of two sets as follow using &:

```
[17]: # Find the intersections

      intersection = album_set1 & album_set2
      intersection
```

```
[17]: {'AC/DC', 'Back in Black'}
```

You can find all the elements that are only contained in album_set1 using the difference method:

```
[18]: # Find the difference in set1 but not set2

      album_set1.difference(album_set2)
```

```
[18]: {'Thriller'}
```

You only need to consider elements in album_set1; all the elements in album_set2, including the intersection, are not included.

The elements in album_set2 but not in album_set1 is given by:

```
[ ]: album_set2.difference(album_set1)
```

You can also find the intersection of album_list1 and album_list2, using the intersection method:

```
[19]: # Use intersection method to find the intersection of album_list1 and␣
       ↪album_list2

      album_set1.intersection(album_set2)
```

```
[19]: {'AC/DC', 'Back in Black'}
```

This corresponds to the intersection of the two circles:

The union corresponds to all the elements in both sets, which is represented by coloring both circles:

The union is given by:

```
[20]: # Find the union of two sets

      album_set1.union(album_set2)
```

```
[20]: {'AC/DC', 'Back in Black', 'The Dark Side of the Moon', 'Thriller'}
```

And you can check if a set is a superset or subset of another set, respectively, like this:

[21]:
```
# Check if superset

set(album_set1).issuperset(album_set2)
```

[21]:
```
False
```

[22]:
```
# Check if subset

set(album_set2).issubset(album_set1)
```

[22]:
```
False
```

Here is an example where issubset() and issuperset() return true:

[23]:
```
# Check if subset

set({"Back in Black", "AC/DC"}).issubset(album_set1)
```

[23]:
```
True
```

[24]:
```
# Check if superset

album_set1.issuperset({"Back in Black", "AC/DC"})
```

[24]:
```
True
```

Quiz on Sets

Convert the list [‘rap’,‘house’,‘electronic music’, ‘rap’] to a set:

[26]:
```
# Write your code below and press Shift+Enter to execute
list4=['rap','house','electronic music', 'rap']
set4=set(list4)
print(set4)
```

```
{'electronic music', 'rap', 'house'}
```

Click here for the solution

```
set(['rap','house','electronic music','rap'])
```

Consider the list A = [1, 2, 2, 1] and set B = set([1, 2, 2, 1]), does sum(A) == sum(B)?

[40]:
```
# Write your code below and press Shift+Enter to execute
A = [1, 2, 2, 1]
B = set([1, 2, 2, 1])
print('sum of A:',sum(A))
print("sum of B:",sum(B))
print("ans:",sum(A) == sum(B))
```

```
sum of A: 6
sum of B: 3
ans: False
```

Click here for the solution

```
A = [1, 2, 2, 1]
B = set([1, 2, 2, 1])
print("the sum of A is:", sum(A))
print("the sum of B is:", sum(B))
```

Create a new set album_set3 that is the union of album_set1 and album_set2:

```
[41]: # Write your code below and press Shift+Enter to execute

album_set1 = set(["Thriller", 'AC/DC', 'Back in Black'])
album_set2 = set([ "AC/DC", "Back in Black", "The Dark Side of the Moon"])
album_set3 =album_set1.union(album_set2)
album_set4=album_set2.union(album_set1)
album_set4==album_set3
```

```
[41]: True
```

Click here for the solution

```
album_set3 = album_set1.union(album_set2)
album_set3
```

Find out if album_set1 is a subset of album_set3:

```
[42]: # Write your code below and press Shift+Enter to execute
album_set1.issubset(album_set3)
```

```
[42]: True
```

Click here for the solution

```
album_set1.issubset(album_set3)
```

The last exercise!

Congratulations, you have completed your first lesson and hands-on lab in Python.

1.2 Author

Joseph Santarcangelo

1.3 Other contributors

Mavis Zhou

1.4 Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2022-01-10	2.1	Malika	Removed the readme for GitShare
2020-08-26	2.0	Lavanya	Moved lab to course repo in GitLab

##