Python Set Difference

Summary: in this tutorial, you'll learn about the Python Set difference and how to use it to find the difference between two or more sets.

Introduction to the Python Set difference

The difference between the two sets (https://www.pythontutorial.net/python-basics/python-set/) results in a new set that has elements from the first set which aren't present in the second set.

Suppose you have the following `s1` and `s2` sets:

```
s1 = {'Python', 'Java', 'C++'}
s2 = {'C#', 'Java', 'C++'}
```

The difference between `s1` and `s2` sets results in the following set with one element:

```
{'Python'}
```

...because there is only `'Python'` element from the first set that doesn't exist in the second set.

The set difference isn't commutative. The difference between the `s2` and `s1` sets returns the following set:

```
{'C#'}
```

The following Venn diagram illustrates the difference between the `s1` and `s2` sets:

And the following Venn diagram illustrates the difference between `s2` and `s1` sets:

In Python, you can use the set `difference()` method or set difference operator (`-`) to find the difference between sets.

1) Using Python Set difference() method to find the difference between sets

The `Set` type has a `difference()` method that returns the difference between two or more sets:

```
set1.difference(s2, s3, ...)
```

For example, you can use the set `difference()` method to find the difference between `s1` and `s2` sets:

```
s1 = {'Python', 'Java', 'C++'}
s2 = {'C#', 'Java', 'C++'}
s = s1.difference(s2)

print(s)
```

Output:

```
{'Python'}
```

And this example shows how to use the set `difference()` method to find the difference between `s2` and `s1` sets.

```
s1 = {'Python', 'Java', 'C++'}
s2 = {'C#', 'Java', 'C++'}
s = s2.difference(s1)

print(s)
```

Output:

```
{'C#'}
```

Note that the `difference()` method returns a new set. It doesn't change the original sets.

2) Using Python set difference operator (-) to find the difference between sets

Besides the `difference()` method, Python provides you with the set difference operator (`-`) that allows you to find the difference between sets.

```
s = s1 - s2
```

The following example uses the difference operator (`-`) to find the difference between the `s1` and `s2` sets:

```
s1 = {'Python', 'Java', 'C++'}
s2 = {'C#', 'Java', 'C++'}
s = s1 - s2

print(s)
```

And this example use the set difference operator to return the difference between `s2` and `s1` :

```
s1 = {'Python', 'Java', 'C++'}
s2 = {'C#', 'Java', 'C++'}
s = s2 - s1

print(s)
```

Output:

```
{'C#'}
```

The set difference() method vs set difference operator (-)

The set `difference()` method can accept one or more iterables (https://www.pythontutorial.net/python-basics/python-iterables/) (e.g., strings (https://www.pythontutorial.net/python-basics/python-string/) , lists (https://www.pythontutorial.net/python-basics/python-list/) , dictionaries (https://www.pythontutorial.net/python-basics/python-dictionary/)) while the set difference operator (`-`) only allows sets.

When you pass iterables to the set `difference()` method, it'll convert the iterables to sets before performing the difference operation.

The following shows how to use the set `difference()` method with a list:

```
scores = {7, 8, 9}
numbers = [9, 10]
new_scores = scores.difference(numbers)

print(new_scores)
```

However, if you use the set difference operator (`-`) with iterables, you'll get an error:

```
scores = {7, 8, 9}
numbers = [9, 10]
new_scores = scores - numbers

print(new_scores)
```

Error:

```
TypeError: unsupported operand type(s) for -: 'set' and 'list'
```

Summary

- A difference between two sets results in a new set containing elements in the first set that aren't present in the second set.
- Use the set `difference()` method or set difference operator (`-`) to find the difference between sets.