Rex Jones II

How To Use String Methods

Python Video = https://youtu.be/hJwRkQuTv9A

In this tutorial session, we are going to look at String methods. When it comes to strings, they are immutable. That means, the string value which are a series of characters between the quotes cannot be changed. However, we can create a copy of the string then modify the string or get something from the string.

For Python, there is a difference between methods and functions. The main difference involves an object. A method depends on an object but a function does not depend on an object. Therefore a method is dependent while a function is independent of an object.

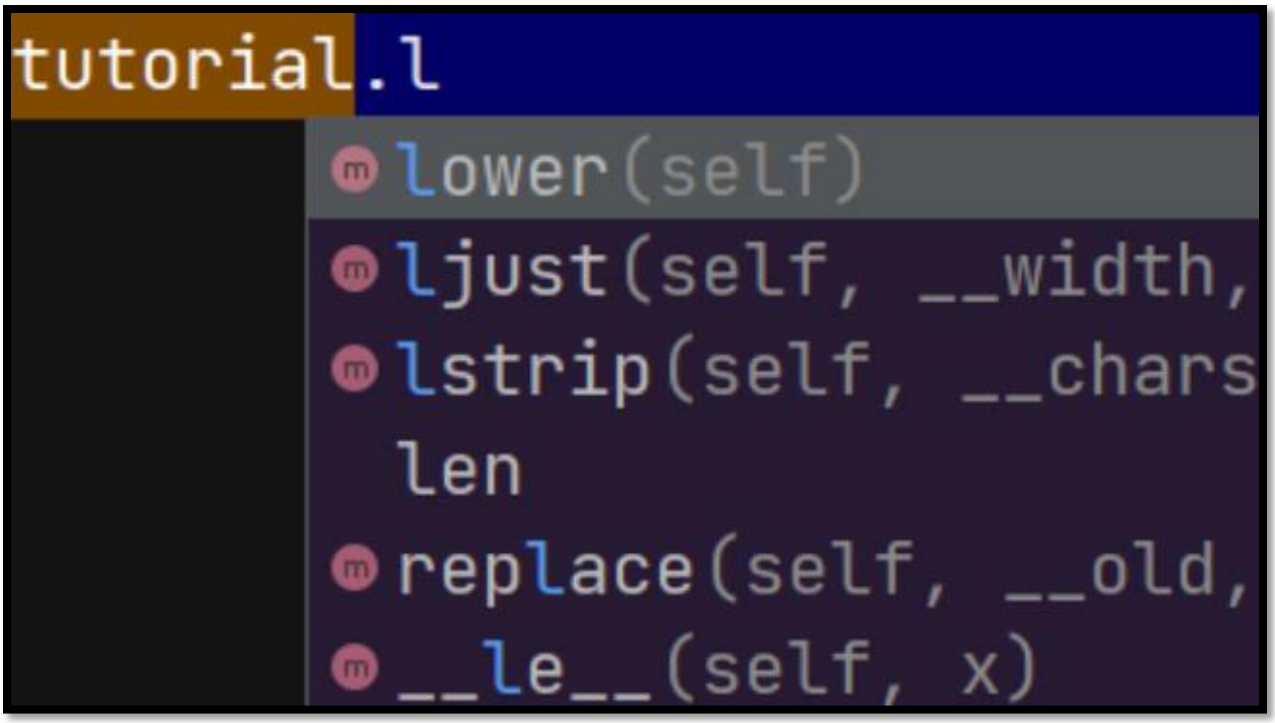
A method is attached to an object after writing a name and dot operator. That attachment provides access to the data. There are a few differences but one more difference is how we call the method and how we call the function. A method cannot be only called by its name. However, a function can be called only by its name.

Our focus will be Python methods but let me show you an example of a function. The name is tutorial = with a value of “Python for Programming/Automation!”. We write the letter ‘l’ to return the number of characters for len. Do you see the small ‘f’ before len?

That ‘f’ let’s us know this is a function. Recall, I said a function can be called only by its name. We are calling len with no other names. Let’s print the number of characters by passing in tutorial inside the len function print(len(tutorial)) and run. The console shows 34.

```
tutorial = "Python for Programming/Automation!"
print(len(tutorial))
```

I’m going to write an ‘l’ one more time. Do you see a small ‘m’ before any of these words? We do not see an ‘m’ because a method needs an object to be called. Now watch what happens when I write print(tutorial.l).



At this point, the intellisense is filled with “m’s”. Let’s select lower(). The purpose of this method is to convert all of the characters to lowercase. I’m going to print and you will see how they are lowercase.

```
34
python for programming/automation!
```

Bingo, python for programming/automation. We can also convert all of the characters to uppercase by changing lower to upper. Let's Run

```
tutorial = "Python for Programming/Automation!"
print(len(tutorial))
print(tutorial.lower())
print(tutorial.upper())
```

```
34
python for programming/automation!
PYTHON FOR PROGRAMMING/AUTOMATION!
```

Now, we see PYTHON FOR PROGRAMMING/AUTOMATION in all uppercase letters. In addition to converting the strings to upper or lower case. We can also check to verify if the string is upper or lower by writing tutorial.isupper(). We can also write islower(). This will check to see if all of the characters are uppercase. This will return a True of False value. It will be True if the string is completely upper case or False if the string is not completely upper case. Let's print() and run. The console shows False.

Now, something else that's unique about Python. It can combine the string methods into 1 statement. So I can check to see if it is upper after converting the string to upper. For example, tutorial.upper() will convert the string to upper. Now, I will check to see if it is upper. Print and run.

```
tutorial = "Python for Programming/Automation!"
print(len(tutorial))
print(tutorial.lower())
print(tutorial.upper())
print(tutorial.isupper())
print(tutorial.upper().isupper())
```

It shows True because they were all uppercase before it verified if they were uppercase. Now, there are scenarios when you may want to find some characters and replace some characters. To find a character, we write tutorial.find(" "). Let's find the letter 'a'. There are 2 characters in this string with a lowercase 'a' but the interpreter will find the first character. Let's print() and see what the console show us.

```
print(tutorial.find("a"))
```

```
16
```

It shows 16 because 'a' in Programming is the 16th character. Let's count. The find method is case sensitive. That means, this 'a' is different from a capital 'A'. print(tutorial.find("A")). Run and we see 23. I have a question for you. What do you think will happen if we try to find a letter that is not in this string? For example, if I write tutorial.find("B"). There is no "B" so when we write print(). The console returns a -1. One more thing, I want to show you about the find method. We can search for a range of characters. Let's search for Python by writing print(tutorial.find("Python")). It shows 0 because 'P' is index 0.

```
print(tutorial.find("a"))
print(tutorial.find('A'))
print(tutorial.find('B'))
print(tutorial.find("Python"))
```

When it comes to the replace method. We write tutorial.replace(). Let's replace "Python for". We place the value we want to replace in the 1st set of quotes then we add a comma, and add more quotes. In the 2nd set of quotes is the value we want to replace it with. How about "Learn"? print() and run. Learn

Programming/Automation! Let's search and replace a single character by writing tutorial.replace("f", "F"). print() and run. The console shows a capital 'F' Python For Programming/Automation.

```
print(tutorial.replace("Python for","Learn"))
print(tutorial.replace('f','F'))
```

```
34
python for programming/automation!
PYTHON FOR PROGRAMMING/AUTOMATION!
False
True
16
23
-1
0
Learn Programming/Automation!
Python For Programming/Automation!
```

Contact Info

- ✔ Email Rex.Jones@Test4Success.org
- ✔ YouTube https://www.youtube.com/c/RexJonesII/videos
- ✔ Facebook https://facebook.com/JonesRexII
- ✔ Twitter https://twitter.com/RexJonesII
- ✔ GitHub https://github.com/RexJonesII/Free-Videos
- ✔ LinkedIn https://www.linkedin.com/in/rexjones34/