Python Regex search()

Summary: in this tutorial, you'll learn how to use the Python regex `search()` function to return the first match of a pattern in a string.

Introduction to the Python regex search() function

The regex `search()` is a function in the built-in `re` module that deals with regular expressions (https://www.pythontutorial.net/python-regex/python-regular-expressions/). The `search()` function has the following syntax:

```
re.search(pattern, string, flags=0)
```

In this syntax:

- pattern is a regular expression that you want to search for in the string.
- string is the input string.
- `flags` is one or more regular expression flags (https://www.pythontutorial.net/python-regex/python-regex-flags/) that modify the standard behavior of the pattern.

The `search()` function scans the `string` from left to right and finds the first location where the `pattern` produces a match. It returns a `Match` object if the search was successful or `None` otherwise.

Python regex search() function examples

Let's take some examples of using the `search()` function.

1) Using the Python regex search() fuction to find the first match

The following example uses the `search()` function to find the first number in the string:

```
import re

s = 'Python 3 was released on Dec 3, 2008'
pattern = '\d+'

match = re.search(pattern, s)

if match is not None:
    print(match.group())
else:
    print('No match found')
```

Output:

```
<re.Match object; span=(7, 8), match='3'>
```

In this example, the pattern `\d+` matches one or more digits. The `search()` returns a `Match` object.

To get the match, you can call the `group()` method of the `Match` object like this:

```
import re
```

```
s = 'Python 3 was released on Dec 3, 2008'
pattern = '\d+'


match = re.search(pattern, s)

if match is not None:
    print(match.group())
```

Output:

```
3
```

2) Using the Python regex search() function to find the first word that matches a pattern

The following example uses the `search()` function to search the first word that ends with the literal string `thon` in a string:

```
import re

s = 'CPython, IronPython, or Cython'
pattern = r'\b((\w+)thon)\b'

match = re.search(pattern, s)

if match is not None:
    print(match.groups())
```

Output:

```
('CPython', 'CPy')
```

The pattern `r'\b((\w+)thon)\b'` has two capturing groups:

- `(\w+)` – captures the characters at the beginning of the word.
- `((\w+)thon)` – captures the whole word.

The `search()` function returns the first location where it finds the match. To access all the groups in a match, you use the `groups()` method of the match object.

As clearly shown in the output, the `groups()` method returns a tuple that contains all the groups.

3) Using the Python regex search() function with a regex flag

The following example uses the `search()` function to find the first `python` word in a string:

```
import re

s = 'Python or python'
pattern = r'\bpython\b'

match = re.search(pattern, s)
print(match)
```

It returns the word `python` with the letter `p` in lowercase as specified in the pattern.

```
<re.Match object; span=(10, 16), match='python'>
```

To match the word case-insensitively, you can pass the `re.IGNORECASE` flag to the third argument of the `search()` function. For example:

```
import re

s = 'Python or python'
pattern = r'\bpython\b'
```

```
match = re.search(pattern, s, re.IGNORECASE)
print(match)
```

In this example, the `search()` function returns the word `Python` with the letter `P` in uppercase:

```
<re.Match object; span=(0, 6), match='Python'>
```

Summary

- Use the regex `search()` function to return the first match of a pattern in a string or `None` if the search was unsuccessful.