Python Regex Non-Greedy

Summary: in this tutorial, you'll learn about the regex non-greedy (or lazy) quantifiers that match their preceding elements as few times as possible.

Introduction to the regex non-greedy (or lazy) quantifiers

Quantifiers (https://www.pythontutorial.net/python-regex/python-regex-quantifiers/) allow you to match their preceding elements a number of times. Quantifiers work in one of two modes: greedy (https://www.pythontutorial.net/python-regex/python-regex-greedy/) and non-greedy (lazy).

When quantifiers work in the greedy mode, they are called greedy quantifiers. Similarly, when quantifiers work in the non-greedy mode, they're called non-greedy quantifiers or lazy quantifiers.

By default, quantifiers work in the greedy mode. It means the greedy quantifiers will match their preceding elements as much as possible to return to the biggest match possible.

On the other hand, the non-greedy quantifiers will match as little as possible to return the smallest match possible. non-greedy quantifiers are the opposite of greedy ones.

To turn greedy quantifiers into non-greedy quantifiers, you add an extra question mark (`?`) to the quantifiers. The following table shows the greedy and their corresponding non-greedy quantifiers:

Greedy quantifier	Lazy quantifier	Meaning
*	*?	Match its preceding element zero or more times.
+	+?	Match its preceding element one or more times.
?	??	Match its preceding element zero or one time.
{ *n* }	{ *n* }?	Match its preceding element exactly *n* times.
{ *n* ,}	{ *n* ,}?	Match its preceding element at least *n* times.
{ *n* , *m* }	{ *n* , *m* }?	Match its preceding element from *n* to *m* times.

Python regex non-greedy quantifiers example

The following program uses the non-greedy quantifier (`+?`) to match the text within the quotes (`""`) of a button element:

```
import re

s = '<button type="submit" class="btn">Send</button>'

pattern = '".+?"'
matches = re.finditer(pattern, s)

for match in matches:
    print(match.group())
```

Output:

```
"submit"
"btn"
```

Summary

- Non-greedy quantifiers match their preceding elements as little as possible to return the smallest possible match.
- Add a question mark (?) to a quantifier to turn it into a non-greedy quantifier.