Matching at Beginning or End

In Python regular expressions, you can use special characters to specify that a pattern should match at the beginning or end of a string. Here are examples of matching patterns at the beginning or end:

1. Caret `^` - Beginning of a String:

The caret `^` asserts that the pattern following it should match at the beginning of the string.

```python
import re

text = "apple banana cherry"

# Match 'apple' only if it appears at the beginning of the string
pattern = re.compile(r'^apple')
match = pattern.search(text)
if match:
    print("Pattern found:", match.group())
else:
    print("Pattern not found")
```

In this example, the pattern `^apple` matches "apple" only if it appears at the beginning of the string.

2. Dollar `$` - End of a String:

The dollar `$` asserts that the pattern preceding it should match at the end of the string.

```python
# Match 'cherry' only if it appears at the end of the string
pattern = re.compile(r'cherry$')
match = pattern.search(text)
if match:
    print("Pattern found:", match.group())
else:
    print("Pattern not found")
```

Here, the pattern `cherry$` matches "cherry" only if it appears at the end of the string.

3. Anchoring at Both Ends:

You can combine `^` and `$` to ensure that the entire string matches a specific pattern.

```python
# Match the entire string if it consists of exactly three lowercase letters
pattern = re.compile(r'^[a-z]{3}$')
```

```
strings = ["abc", "abcd", "ab", "abcde"]

for s in strings:
    if pattern.match(s):
        print(f"Pattern found for {s}: {pattern.match(s).group()}")
    else:
        print(f"Pattern not found for {s}")
```

In this example, the pattern `^[a-z]{3}$` matches strings that consist of exactly three lowercase letters.

4. Multiline Mode:

In multiline mode, the `^` and `$` anchors also match the beginning and end of each line within a multiline string.

```python
multiline_text = "Line 1\nLine 2\nLine 3"

# Match lines that start with 'Line'
pattern = re.compile(r'^Line', re.MULTILINE)
matches = pattern.findall(multiline_text)
print(matches)
# Output: ['Line', 'Line', 'Line']
```

Here, the `^` anchor in multiline mode matches the start of each line in the multiline string.

Understanding how to anchor patterns at the beginning or end of a string is crucial for building effective regular expressions, especially when you want to enforce specific constraints on the structure of the string.