

# Regular Expression Basics: Takeaways

by Dataquest Labs, Inc. - All rights reserved © 2021

## Syntax

---

### REGULAR EXPRESSION MODULE

- Importing the regular expression module:

```
import re
```

- Searching a string for a regex pattern:

```
re.search(r"blue", "Rhythm and blues")
```

---

### PANDAS REGEX METHODS

- Return a boolean mask if a regex pattern is found in a series:

```
s.str.contains(pattern)
```

- Extract a regex capture group from a series:

```
s.str.extract(pattern_with_capture_group)
```

---

### ESCAPING CHARACTERS

- Treating special characters as ordinary text using backslashes:

```
\[pdf\]
```

## Concepts

- Regular expressions, often referred to as regex, are a set of syntax components used for matching sequences of characters in strings.
- A pattern is described as a regular expression that we've written. We say regular expression has matched if it finds the pattern exists in the string.
- Character classes allow us to match certain classes of characters.
- A set contains two or more characters that can match in a single character's position.
- Quantifiers specify how many of the previous characters the pattern requires.
- Capture groups allow us to specify one or more groups within our match that we can access separately.
- Negative character classes are character classes that match every character except a character class.
- An anchor matches something that isn't a character, as opposed to character classes which match specific characters.
- A word boundary matches the space between a word character and a non-word character, or a word character and the start/end of a string
- Common character classes:

Character Class	Pattern	Explanation
Set	<code>[fud]</code>	Either <code>f</code> , <code>u</code> , or <code>d</code>
Range	<code>[a-e]</code>	Any of the characters <code>a</code> , <code>b</code> , <code>c</code> , <code>d</code> , or <code>e</code>
Range	<code>[0-3]</code>	Any of the characters <code>0</code> , <code>1</code> , <code>2</code> , or <code>3</code>
Range	<code>[A-Z]</code>	Any uppercase letter
Set + Range	<code>[A-Za-z]</code>	Any uppercase or lowercase character
Digit	<code>\d</code>	Any digit character (equivalent to <code>[0-9]</code> )
Word	<code>\w</code>	Any digit, uppercase, or lowercase character (equivalent to <code>[A-Za-z0-9_]</code> )
Whitespace	<code>\s</code>	Any space, tab or linebreak character
Dot	<code>.</code>	Any character except newline

- Common quantifiers:

Quantifier	Pattern	Explanation
Zero or more	<code>a*</code>	The character <code>a</code> zero or more times
One or more	<code>a+</code>	The character <code>a</code> one or more times
Optional	<code>a?</code>	The character <code>a</code> zero or one times
Numeric	<code>a{3}</code>	The character <code>a</code> three times
Numeric	<code>a{3,5}</code>	The character <code>a</code> three, four, or five times
Numeric	<code>a{,3}</code>	The character <code>a</code> one, two, or three times
Numeric	<code>a{8,}</code>	The character <code>a</code> eight or more times

- Common negative character classes:

Character Class	Pattern	Explanation
Negative Set	<code>[^fud]</code>	Any character except <code>f</code> , <code>u</code> , or <code>d</code>
Negative Set	<code>[^1-3Z\s]</code>	Any characters except <code>1</code> , <code>2</code> , <code>3</code> , <code>Z</code> , or whitespace characters
Negative Digit	<code>\D</code>	Any character except digit characters
Negative Word	<code>\W</code>	Any character except word characters
Negative Whitespace	<code>\S</code>	Any character except whitespace characters

- Common anchors:

Anchor	Pattern	Explanation
Beginning	<code>^abc</code>	Matches <code>abc</code> only at the start of a string
End	<code>abc\$</code>	Matches <code>abc</code> only at the end of a string
Word boundary	<code>s\b</code>	Matches <code>s</code> only when it's followed by a word boundary
Word boundary	<code>s\B</code>	Matches <code>s</code> only when it's not followed by a word boundary

## Resources

- [re module](#)
- [Building regular expressions](#)

