

# REGEX AND REGEXR

---

An Introduction to Regular Expressions and Their Applications in Technical Services

# WHAT IS REGEX?

---

# Regular Expressions

Regular expressions are patterns of characters that allow for more options when searching and editing documents

What does that mean?

# Example: Let's search Mother Goose

Jack and Jill went up the hill  
To fetch a pail of water;  
Jack fell down, and broke his crown,  
And Jill came tumbling after.

# Let's find words that rhyme end in -ill

Search term:	Result:
ill	The pattern "ill" anywhere in the document

Jack and Jill went up the hill  
To fetch a pail of water;  
Jack fell down, and broke his crown,  
And Jill came tumbling after.

# The problem?

We're looking for words that end in -ill. What if the poem went like this?

Jack and Jill went up the hill  
To fetch a filling of water;  
Jack fell down, and broke his crown,  
And Jill came tumbling after.

The search would return the "ill" in "filling," and that isn't what we want.

# The solution?

With regular expressions, one can search for words that end in -ill, giving us only those words which rhyme

Search term:	Result:
<code>\w*ill\b</code>	A pattern of characters ending in "ill" anywhere in the document

# So what can regular expressions do?

- Search by character type
- Search by character variations
- Search by character quantity
- Search by position
- Find and replace



# REGEX BASICS

---

# Character Classes

- Character classes allow you to search for patterns by [character type](#)

`\w`

- Word characters

`\d`

- Digit characters

`\s`

- Space characters

# Using word characters `\w` A-Za-z0-9\_

Search term:	Result:
<code>h\wt</code>	A pattern containing "h," followed by one word character, followed by "t"

It was `hot` so I wore a `hat` when I `hit` a home run.

# Using digit characters `\d` 0-9

Search term:	Result:
<code>\d</code>	A pattern of a single digit

Why was 6 afraid of 7? Because 7, 8, 9.

Also: g2g, bye!

# Using space characters (spaces, tabs, breaks)

Search term:	Result:
<code>\s</code>	Matches all spaces, tabs, and breaks in the document
<code>\t</code>	Matches all tabs in the document
<code>\r\n</code>	Matches all patterns of line feed and carriage return (all new line feeds)

## BONUS: The wildcard character .

Search term:	Result:
.	Any character anywhere in the document
.ill	Any character followed by the pattern "ill" (see below)

Jack and Jill went up the hill  
To fetch a filling of water;  
Jack fell down, and broke his crown,  
And Jill came tumbling after.

# Character Sets

- Character sets allow you to search for patterns using [character variations](#)

[ ]

- Character set

[^ ]

- Negated character set

[ - ]

- Character set range

( | )

- Alternation

# Using character sets [ ]

Search term:	Result:
h[oa]t	A pattern containing "h," followed by an "o" or an "a," followed by "t"

It was **hot** so I wore a **hat** when I hit a home run.



# Using negated character sets [^]

Search term:	Result:
<code>h[^oat]</code>	A pattern containing "h," followed by one letter that is not an "o" or an "a," followed by "t"

It was hot so I wore a hat when I **hit** a home run.

# Using character set ranges [ - ]

Search term:	Result:
<code>h[a-i]t</code>	A pattern containing "h," followed by a character within the range of "a" through "i," followed by "t"

It was hot so I wore a **hat** when I **hit** a home run.

# Using character set alternation ( | )

Search term:	Result:
<code>h(a i o)t</code>	A pattern containing "h," followed by either an "a" or an "i" or an "o," followed by "t"

It was **hot** so I wore a **hat** when I **hit** a home run.

# Character Quantifiers

- Character quantifiers allow you to search for patterns by [character quantity](#)

{ }

- Single quantifier for the previous character

{ , }

- Quantifier range for the previous character

?

- 0 to 1 of the previous character

\*

- 0 or more of the previous character

+

- 1 or more of the previous character

# Using a character quantifier { }

Search term:	Result:
<code>be{2}</code>	A pattern containing a "b" and two "e's"

If I could be a thing with wings, I'd be a **bee**.

# Using a character quantifier range { , }

Search term:	Result:
<code>be{1,2}</code>	A pattern containing a "b" followed at least 1 "e" and at most 2 "e's"

If I could **be** a thing with wings, I'd **be** a **bee**.

# Using the 0-1 character quantifier ?

Search term:	Result:
<code>g?lass</code>	A pattern with 0 or 1 "g" followed by "lass"

Pour a **glass** for the **lass**!

*(Of lemonade, that is.)*

# Using the 0 or more character quantifier \*

Search term:	Result:
Pas*	A pattern of a "P," an "a", and 0 or more "s's"

Pass the peas, please, Pa!



# Using the 1 or more character quantifier +

Search term:	Result:
princes+	The pattern "prince" followed by at least one "s"

Many **princes** wanted to marry the **princess**.

# Location Anchors

- Location anchors allow you to search for patterns by their [position](#)

^

- Searches at the beginning of a line

\$

- Searches at the end of a line

\b

- Word boundary

# Searching the beginning of a line ^

Search term:	Result:
<code>^Mary</code>	Finds the pattern "Mary" at the beginning of a line

**Mary** had a little lamb,  
His fleece was white as snow,  
And everywhere that Mary went,  
The lamb was sure to go.

# Searching the end of a line \$

Search term:	Result:
merrily,\$	Finds the pattern "merrily," at the end of a line

Row, row, row your boat,  
Gently down the stream;  
Merrily, merrily, merrily, merrily,  
Life is but a dream.

# Searching with a word boundary \b

Search term:	Result:
fish\b	Finds the pattern "fish" followed by a word boundary

The little stars were the herring fish  
That lived in that beautiful sea —  
"Now cast your nets wherever you wish —  
Never afraid are we";  
So cried the stars to the fishermen three:  
Wynken, Blynken, and Nod.

# Capturing Groups

- Capturing groups allow you to perform [find and replace](#)

()

- Captures a regular expression

\$1, \$2,  
\$3, etc.

- References the capture

# Using a capture ( )

Search term:	Result:
(a) (penny)	Finds the pattern "a" and stores it as the first reference; finds the pattern "penny" and stores it as the second reference

Hot cross buns!  
Hot cross buns!  
one a penny, two a penny,  
Hot cross buns!

# Using a capture ( ) \$1 \$2

Search term:	Result:
(a) (penny)	Finds the pattern "a" and stores it for reference as the first capture; finds the pattern "penny" and stores it as the second reference
\$1	References the first capture
\$2	References the second capture

Hot cross buns!  
Hot cross buns!  
one a penny, two a penny,  
Hot cross buns!



# Using a capture ( ) \$1 \$2

Find:	Replace:
(a) (penny)	\$1-\$2

Hot cross buns!

Hot cross buns!

one a-penny, two a-penny,

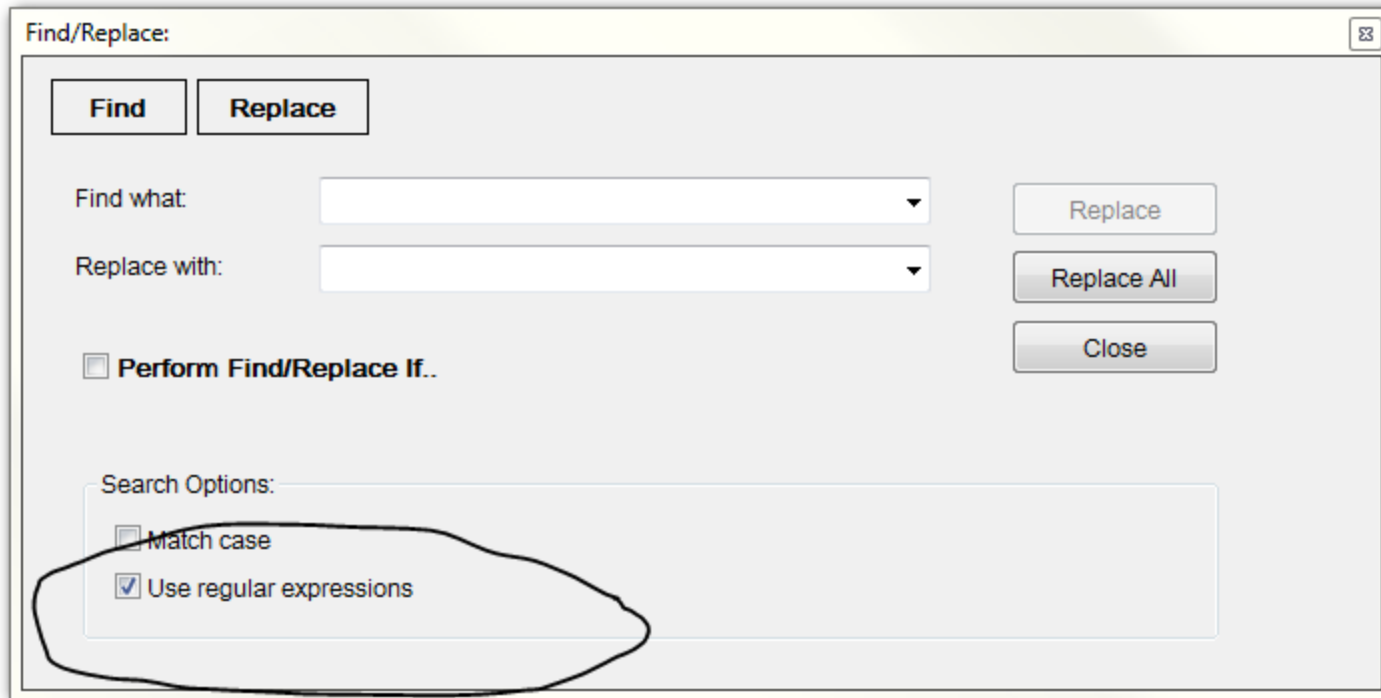
Hot cross buns!

# PRACTICAL APPLICATIONS

---

# How would you...

- Search for an author when you aren't sure if his last name is Larson, Larsson, or Larsen?
- Find all the 12-digit barcodes in a document?
  - And then capture just the last 7 digits of that barcode?
- Add a full-stop to 100 fields that don't have one?
- Change all the "\$a Bible \$x Dictionaries" subject headings to "\$a Bible \$v Dictionaries"
  - And, while incorrect, for fun: how would we reorder those subfields if we wanted to?



# PRACTICE WITH REGEXR

---

# RegExr <http://www.regexr.com/>

- RegExr is one of several websites which allows you to practice regular expressions on a block of text

Regex goes here



The screenshot shows the RegExr website interface. At the top, there is a blue header with the word "Expression" and icons for "share", "save", and "flags". Below the header, the regex expression `/([A-Z])\w+/g` is entered in a text field, and a button indicates "19 matches". The main area is titled "Text" and contains a sample document. The text in the document is as follows:

```
Welcome to RegExr v2.1 by gskinner.com!  
  
Edit the Expression & Text to see matches. Roll over matches or the expression for details.  
Undo mistakes with ctrl-z. Save Favorites & Share expressions with friends or the Community.  
Explore your results with Tools. A full Reference & Help is available in the Library, or  
watch the video Tutorial.  
  
Sample text for testing:  
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ  
0123456789 _+-. ,!@#$%^&*();\|/|<>"'  
12345 -98.7 3.141 .6180 9,000 +42  
555.123.4567 +1-(800)-555-2468  
foo@demo.net bar.ba@test.co.uk  
www.demo.com http://foo.co.uk/  
http://regexr.com/foo.html?q=bar
```

In the screenshot, the words "Welcome", "RegExr", "v2.1", "by", "gskinner.com!", "Edit", "Expression", "Text", "Roll", "over", "matches", "or", "the", "expression", "for", "details.", "Undo", "mistakes", "with", "ctrl-z.", "Save", "Favorites", "&", "Share", "expressions", "with", "friends", "or", "the", "Community.", "Explore", "your", "results", "with", "Tools.", "A", "full", "Reference", "&", "Help", "is", "available", "in", "the", "Library,", "or", "watch", "the", "video", "Tutorial.", "Sample", "text", "for", "testing:", "ABCDEFGHIJKLMNOPQRSTUVWXYZ", "0123456789", " \_+-. ,!@#\$%^&\*();\|/|<>"', "12345", "-98.7", "3.141", ".6180", "9,000", "+42", "555.123.4567", "+1-(800)-555-2468", "foo@demo.net", "bar.ba@test.co.uk", "www.demo.com", "http://foo.co.uk/", and "http://regexr.com/foo.html?q=bar" are all highlighted in blue.

Document text goes here (the text which matches the regex is highlighted in blue)

# RegExr <http://www.regexr.com/>

The screenshot shows the RegExr v2.1 web application. The interface is dark-themed with a light blue header. On the left is a 'Library' sidebar with navigation links: Help, Reference, Cheatsheet, Examples, Community, and Favourites. The 'Cheatsheet' link is circled in yellow. The main content area has a light blue header with the text 'Expression' and icons for 'share', 'save', and 'flags'. Below this, the regex expression `/([A-Z])\w+/g` is shown, with a '19 matches' button to its right. The 'Text' section contains the following content:

Welcome to RegExr v2.1 by gskinner.com!

Edit the Expression & Text to see matches. Roll over matches or the expression for details. Undo mistakes with ctrl-z. Save Favorites & Share expressions with friends or the Community. Explore your results with Tools. A full Reference & Help is available in the Library, or watch the video Tutorial.

Sample text for testing:

```
abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ
0123456789 _+-. ,!@#$$%^&*();\|/|<>'
12345 -98.7 3.141 .6180 9,000 +42
555.123.4567 +1-(800)-555-2468
foo@demo.net bar.ba@test.co.uk
www.demo.com http://foo.co.uk/
http://regexr.com/foo.html?q=bar
```