Enhancing Your MadCap Flare Skills with Regular Expressions

PRESENTED BY

Jenny Pittman, Sr. Technical Writer, BeyondTrust

PREVIEWS OF COMING ATTRACTIONS

- What is a regular expression?
- Why would I want to use regular expressions?
- How has BeyondTrust has used regular expressions?
- How do regular expressions work?
- What are best practices for using regular expressions?
- What if I want to go even deeper?

What is a regular expression?

WHAT IS A REGULAR EXPRESSION?

- “A regular expression is a pattern that the regular expression engine attempts to match in input text. A pattern consists of one or more character literals, operators, or constructs.” https://docs.microsoft.com/en-us/dotnet/standard/base-types/regular-expression-language-quick-reference

AGAIN, *WHAT* IS A REGULAR EXPRESSION?

- A way to search for a range of characters
- A way to search for “this *or* that”
- A way to limit your search to “this *but not* that”
- A way to limit your search to “this *if* that”

regex or regexp

BE AWARE!

- This presentation gives examples for MadCap Flare's regex parser.
- Other software programs may use different parsers.

- For our purposes, a parser has *nothing* to do with a parsec.

Why use regular expressions?

REFINE THE SEARCH

- Standard search finds content based on:
  - Words or phrases
  - Element type (<p>, <h1>, , <MadCap:conditionalText>, etc.)
  - Attribute (style, class, condition, etc.)

```
<h1>Introduction</h1>
<h2>Intro</h2>
<h1 class="red">Intro</h1>
```

- Regex search finds content based on:
  - Multiple factors (this text in this or that element with this attribute)
  - Beginning, middle, or end of the line
  - Beginning, middle, or end of the topic

```
<h\d[^>]>Intro(duction)?</h\d>
```

```
<h1>Introduction</h1>
<h2>Intro</h2>
<h1 class="red">Intro<h1>
```

EXPAND THE REPLACE

- Standard search replaces x with y
- Regex search can:
  - Modify or remove tags while keeping the content

    Intro to <h1>Intro</h1> *and* Outro to <h1>Outro</h1>
  - Modify or delete text that may be formatted in multiple ways

    Note:, Notes, Note:
  - Replace some but not all instances of a word

    Change “blue” to “red” unless part of the word “blueprint”

How we’ve used regexes

THE LEGEND OF REGEX

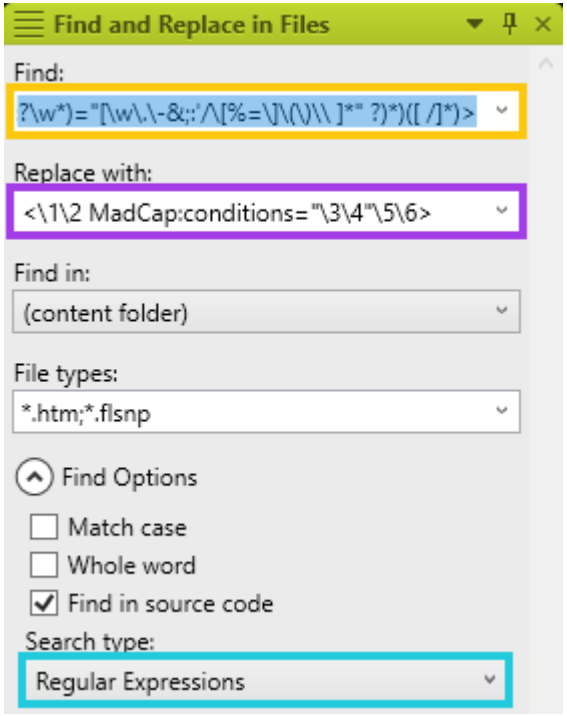
Find text

Replace with text

The original text

The text after find and replace

Set the search to **Regular Expressions**



GET RID OF HARD-CODED NOTES

- Modified our stylesheet to automatically include “Note:” and “Important!”
- Used regex to delete hard-coded text

```
<(b|strong)>Note( | )*(</\1>)?( | )*\:(</\1>)?( | )*(</\1>)?
```

```

```

```
<p class="note"><b>Note:</b> Be sure to drink your Ovaltine.</p>
```

```
<p class="note">Be sure to drink your Ovaltine.</p>
```

MAKE SIMPLE COMMANDS BOLD

- Bolded one-word, unformatted “click” commands

(c|C)lick(ing)? (the)?(OK|Add|Edit|Close|Next|Save|Delete|Enter)(button)?

\1lick\2 \3\4\5

Click OK, then finish by clicking the Close button.

Click **OK**, then finish by clicking the **Close** button.

MAKE EACH TOPIC'S FIRST LINE AN H1

- Replaced starting paragraphs, H2s, and H3s to satisfy SEO needs

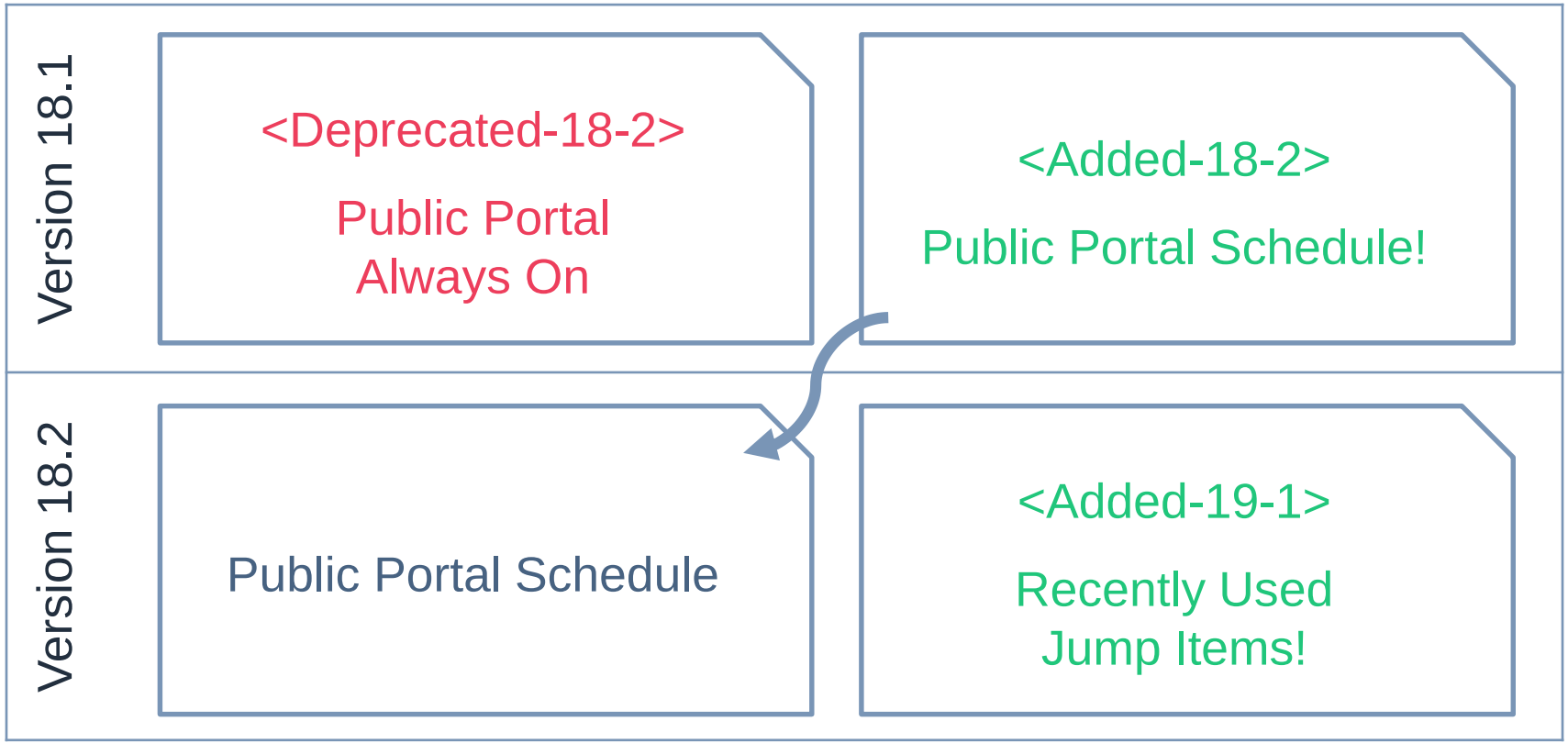
```
(<body>\s*)<(p|h[^1])([^>]*)>(.*)</\2>
```

```
\1<h1\3>\4</h1>
```

```
<h2 class="style">Header Text</h2>
```

```
<h1 class="style">Header Text</h1>
```

REMOVE OR CHANGE ATTRIBUTES: OUR PROCESS



REMOVE OR CHANGE ATTRIBUTES

- Modified classes, styles, conditions, and other attributes

```
<(\w+:?\w* ?)((?: (?:\w+:?\w*)="[^"]*" ?)*) MadCap:conditions="([\w\.\-, ]*)(?:,?Release\.Added\-RS\-18\-2,?)([\w\.\-, ]*)"((?: (?:\w+:?\w*)="[^"]*" ?)*)([ /]*)>
```

```
<\1\2 MadCap:conditions="\3\4"\5\6>
```

```
<p MadCap:conditions="Release.Added-RS-18-2,Default.PrintOnly">
```

```
<p MadCap:conditions="Default.PrintOnly">
```

How do regular expressions work?

Special Characters and their Superpowers

\	( )	[ ]	{ }
.	|	-	< >
*	?	+	!
:	$	^	=

Note:

To use any of these as a literal, you must precede it with a backslash!

- To search for an asterisk: *
- To search for a backslash: \\

ABC’S AND 123’S

- To search for any letter, number, or underscore: \w
- To search for a space, tab, or line break: \s

Note:

This does not search for non-breaking spaces, coded in Flare as

- To search for any character except newlines, use a dot: .
- To specifically search for any number: \d
- To specifically search for a tab: \t
- To specifically search for a line break: \r\n

IT’S A GROUP EFFORT

- Group and capture with parentheses ( )
  - Searches for a string as a single token
  - Treats (cat) as one single search term that cannot be broken up; finds cat, catalog, and concatenate but not act
  - Used with repetition and backreference
- Find this or that with (|)
  - Use (cat|dog) to find either cat or dog

CAPTURE THE FLAG (OR DON'T)

- Use backreference to find the same captured group twice
  - Use (\w*) \1 to find apple apple, grape grape, etc.
  - Use (\w*) is as \1 does to find Pretty is as pretty does
- Use backreference in the replace field to keep a captured group as it was found
  - Find (\w*) and (\w*) and replace with \2 and \1 to replace sugar and spice with spice and sugar (or apples and oranges with oranges and apples)
- Group but don't capture with (?:) to keep your backreference from exceeding the Flare limit of 9

PICK A CARD, ANY CARD

- Find any matching character with square brackets [ ]
  - Called a character class or character set
  - Find any letter or number: [a-z0-9]

Note:

Unlike some parsers, Flare is not case-sensitive unless you check **Match case** in the **Find options**.

  - Find any letter between a and n: [a-n]
  - Find any vowel: [aeiou]

Note:

By itself, searches for only one instance.

BUT NOT THAT CARD

- Find text that does not contain any specified character [^]
  - Find cat or cast but not cart: ca[^r]?t
  - Find cat but not cast or cart: ca[^rs]?t
- Find text that does not contain a specified string (?!)
  - Find The book was great but not The movie was great: The (?!movie)\w+ was great
  - Find I love ice cream sandwiches or I love tomato sandwiches but not I love tomato tofu sandwiches: I love (?!tomato tofu)[\w]* sandwiches

SET BOUNDARIES

- To define the beginning or end of a word: \b

Note:

Use two to duplicate Flare's built-in **Whole word** search option: \bcast\b finds cast but not castle or podcast.

Use one to define only one side of the word boundary: \bcast finds cast and castle but not podcast, while cast\b finds cast and podcast but not castle.

- To define the beginning of a line: ^
- To define the end of a line: $

SMALL, MEDIUM, OR LARGE?

- Find the character or group 0 or 1 times: ?
  - Use It’s (not)?raining to find both It’s raining and It’s not raining
- Find the character or group 1 or more times: +
  - Use ho+p to find both hop and hoop (and hooooooooooop)
- Find the character or group 0 or more times: *
  - Use I’m [\w]*ready to find both I’m ready and I’m almost ready (and I’m definitely almost certainly ready)

WOULD YOU LIKE TO SUPERSIZE THAT?

- Find the character or group exactly x times: {x}
  - Use ho{2}p to find hoop but not hop (or hooooooooooop)
- Find the character or group at least x times but no more than y times: {x,y}
  - Use \b\w{5,7}\b to find Psycho but neither Jaws nor Casablanca

Another look at the examples

GET RID OF HARD-CODED NOTES

- <(b|strong)>Note(|)*(</\1>)?(|)*\:(</\1>)?(|)*(</\1>)?
- Find Note or Note
- Find zero or more spaces
- Find or

- Why not use \2 for the second instance of (|)?
- Once a capturing group has been found the first time, all backreferences equal that text

MAKE SIMPLE COMMANDS BOLD

- (c|C)lick(ing)? (the)?(OK|Add|Edit|Close|Next|Save)(button)?
- \1lick\2 \3\4\5
- Find click, Click, clicking, or Clicking
- Find zero or one instances of the
- Find OK, Add, Edit, or another specified word
- Find zero or one instances of button

MAKE EACH TOPIC'S FIRST LINE AN H1

- (<body>\s*)<(p|h[^1])([^>]*)>(.*)</\2>
- \1<h1\3>\4</h1>
- Find the <body> tag followed by zero or more spaces, tabs, or line breaks
- Find p or any header tag that is not h1
- Find zero or more characters that are not >
- Find zero or more characters other than line breaks
- Find the closing p or header tag

REMOVE OR CHANGE ATTRIBUTES

- <(\w+:?\w* ?)((?: (?:\w+:?\w*)="[^"]*" ?)*) MadCap:conditions="([\w\.\-,]*)(?:,?Release\.Added\-RS\-18\-2,?)([\w\.\-,]*)"((?: (?:\w+:?\w*)="[^"]*" ?)*)([/]*)>
- <\1\2 MadCap:conditions="\3\4"\5\6>
- Find any opening tag, including MadCap:x tags
- Do not explicitly capture this group (still captured as part of the larger group)
- Find zero or more attributes, including MadCap:x attributes, with a definition including any characters other than "
- Find zero or more additional conditions
- Find the condition Release.Added-RS-18-2, optionally preceded or followed by a comma
- Find the closing bracket, preceded by zero or more slashes or spaces

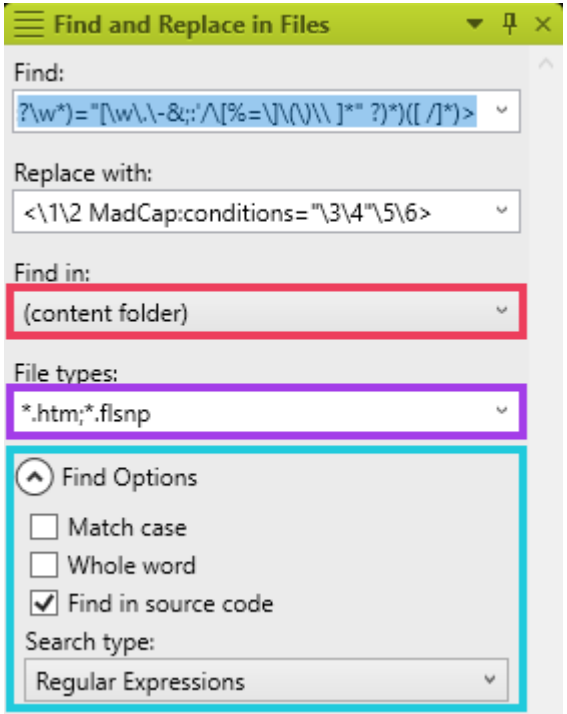
Top tips!

TESTING, 1, 2, 3

- Test, test, test – that is, (test), \1, \1
- Commit to source control regularly throughout
- Regular text search to see how many results to expect
- Find with regex and check that results count isn't too high
- Find/replace a few with regex to make sure replace works
- Use in-topic find/replace to see where the regex is broken
- Find/replace all with regex and compare results count
- Commit, then regular text search to find unchanged files
- Update the regex and repeat the process

BONUS TIP!

- Regex searches can take a long time to run
- To cut down processing time, specify which file types to search in the **File types** box
- With **Find in**, pick a folder to break big searches into smaller chunks
- Remember your **Find Options**



The Really Complex Stuff

LOOKIN' AHEAD

- Find the character or group only if it's immediately followed by what's in the parentheses: (?=)
  - Use super(?=hero) to find superhero but not superpower
- Find the character or group only if it's *not* immediately followed by what's in the parentheses: (?!)
  - Use super(?!hero) to find superpower but not superhero

LOOKIN' BEHIND

- Find the character or group only if it's immediately preceded by what's in the parentheses: (?<=)
  - Use (?<=soft)ware to find software but not hardware
- Find the character or group only if it's *not* immediately preceded by what's in the parentheses: (?<!)
  - Use (?<!soft)ware to find hardware but not software

IFS, ANDS, AND BUTS

- If a is true, find b; otherwise, find c: (?( ))
- Given aircraft, airtime, watercraft, lifetime:
  - Use (?(?<=air)craft|time) to find aircraft and lifetime
    - Find craft if it's immediately preceded by air; otherwise, find time
  - Use (?(?<!air)craft|time) to find watercraft and airtime
    - Find craft if it's *not* immediately preceded by air; otherwise, find time
  - Use (?(?=craft)air|life) to find aircraft and lifetime
    - Find air if it's immediately followed by craft; otherwise find life
  - Use (?(?!craft)air|water) to find airtime and watercraft
    - Find air if it's *not* immediately followed by craft; otherwise find water

Try it out!

TRY IT: SWITCH REGULAR TEXT TO A VARIABLE

- Your project uses the word "yarn" throughout.
- One user needs "fiber" instead, and another "wool".
- You've created two variables: [%=Variables.yarn%] and [%=Variables.Yarn%].
- How do you replace "yarn" with these variables?

Tip:

Instead of using the XML editor default of <MadCap:variable name="Variables.Yarn" /> you can use [%=Variables.Yarn%], the code format. While this doesn't show the definition in the WYSIWYG, it renders correctly in the output, and it makes find/replace far easier.

TRY IT: CHANGE A HEADER TYPE

- Your project has topics that use H3 as their first header.
- Your webmaster says these must all be switched to H1.
- You've created a new style called h1.h3style.
- How do you replace H3s at the beginning of the topic but not in the middle?
- Bonus: How would you do this if some H3s have other classes you want to keep?

TRY IT: FIND EMPTY AND MISSING ALT TAGS

- Your webmaster wants all image alt text to be between 12 characters and 16 words.
- You suspect that many images have either:
  - Nothing between the quotation marks
  - Too-short or too-long descriptions
- How do you find errant images? (may take two searches)

SOURCES

- https://journalxtra.com/linux/bash/regular-expressions-a-quick-guide/
- https://thenewstack.io/dont-fear-regex-getting-started-regular-expressions/
- https://www.rexegg.com/
- https://www.regular-expressions.info/tutorial.html
- https://docs.microsoft.com/en-us/dotnet/standard/base-types/regular-expression-language-quick-reference