



# Regular Expression Wizardry

*Josiah Mory*

*@kickinbahrk*

---



*Let's set the stage...*

`/^[\\w+\\-\\.]+@[a-z\\d\\-]+(\\. [a-z\\d\\-]+)*\\. [a-z]+$/i`







`/^[\\w+\\-\\.]+@[a-z\\d\\-]+(\\. [a-z\\d\\-]+)*\\. [a-z]+$/i`





*Maybe someone was having  
a bad day*

What is a Regular  
Expression?



# Regular Expression

=

a pattern describing a  
specific string of text

**/regex-pattern-here/**



/[a-z]/

# Regular Expression "Engine"

=

a piece of software which  
processes regular expressions  
and tries to match the pattern  
to the given string



The syntax and  
behavior of a particular  
engine

=

*regular expression flavor*

# 3 Things You Do with Regular Expressions



# *Search*

---

a string to see if it matches your  
pattern

# *Extract*

---

a string (or part of a string) that matches  
your pattern



# *Replace*

---

a string by replacing parts that match  
with other text

# “Find-and-Replace on Steroids”

## - Dan Nguyen

---



# *Literal Characters*

---

# Most Characters:

- a - z
- A - Z
- 0 - 9



# *Special Characters*

---

12 special or (meta) characters

\, ^, \$, ., |, ?, \*, +, (, ), [, [



\* If you want to use any of these characters as a literal in a regex, you need to escape them with a backslash

## Common Metacharacters

^	[	.	\$
{	*	(	\
+	)		?
<	>		

The escape character is usually \

## Quantifiers

*	0 or more	{3}	Exactly 3
+	1 or more	{3,}	3 or more
?	0 or 1	{3,5}	3, 4 or 5

Add a ? to a quantifier to make it ungreedy.



## Character Classes

<code>\c</code>	Control character
<code>\s</code>	White space
<code>\S</code>	Not white space
<code>\d</code>	Digit
<code>\D</code>	Not digit
<code>\w</code>	Word
<code>\W</code>	Not word
<code>\x</code>	Hexadecimal digit
<code>\O</code>	Octal digit

## Groups and Ranges

<code>.</code>	Any character except new line ( <code>\n</code> )
<code>(a b)</code>	a or b
<code>(...)</code>	Group
<code>(?:...)</code>	Passive (non-capturing) group
<code>[abc]</code>	Range (a or b or c)
<code>[^abc]</code>	Not (a or b or c)
<code>[a-q]</code>	Lower case letter from a to q
<code>[A-Q]</code>	Upper case letter from A to Q
<code>[0-7]</code>	Digit from 0 to 7
<code>\x</code>	Group/subpattern number "x"

Ranges are inclusive.

## Anchors

<code>^</code>	Start of string, or start of line in multi-line pattern
<code>\A</code>	Start of string
<code>\$</code>	End of string, or end of line in multi-line pattern
<code>\Z</code>	End of string
<code>\b</code>	Word boundary
<code>\B</code>	Not word boundary
<code>\&lt;</code>	Start of word
<code>\&gt;</code>	End of word

## String Replacement

<code>\$n</code>	nth non-passive group
<code>\$2</code>	"xyz" in <code>/^(abc(xyz))\$/</code>
<code>\$1</code>	"xyz" in <code>/^(?:abc)(xyz)\$/</code>
<code>\$`</code>	Before matched string
<code>\$'</code>	After matched string
<code>\$+</code>	Last matched string
<code>\$&amp;</code>	Entire matched string

Some regex implementations use `\` instead of `$`.



```
gandalf_quote1 = "You shall not pass! -Gandalf"
```

```
the_grey = " The Grey"
```

```
console.log(gandalf_quote + the_grey);
```

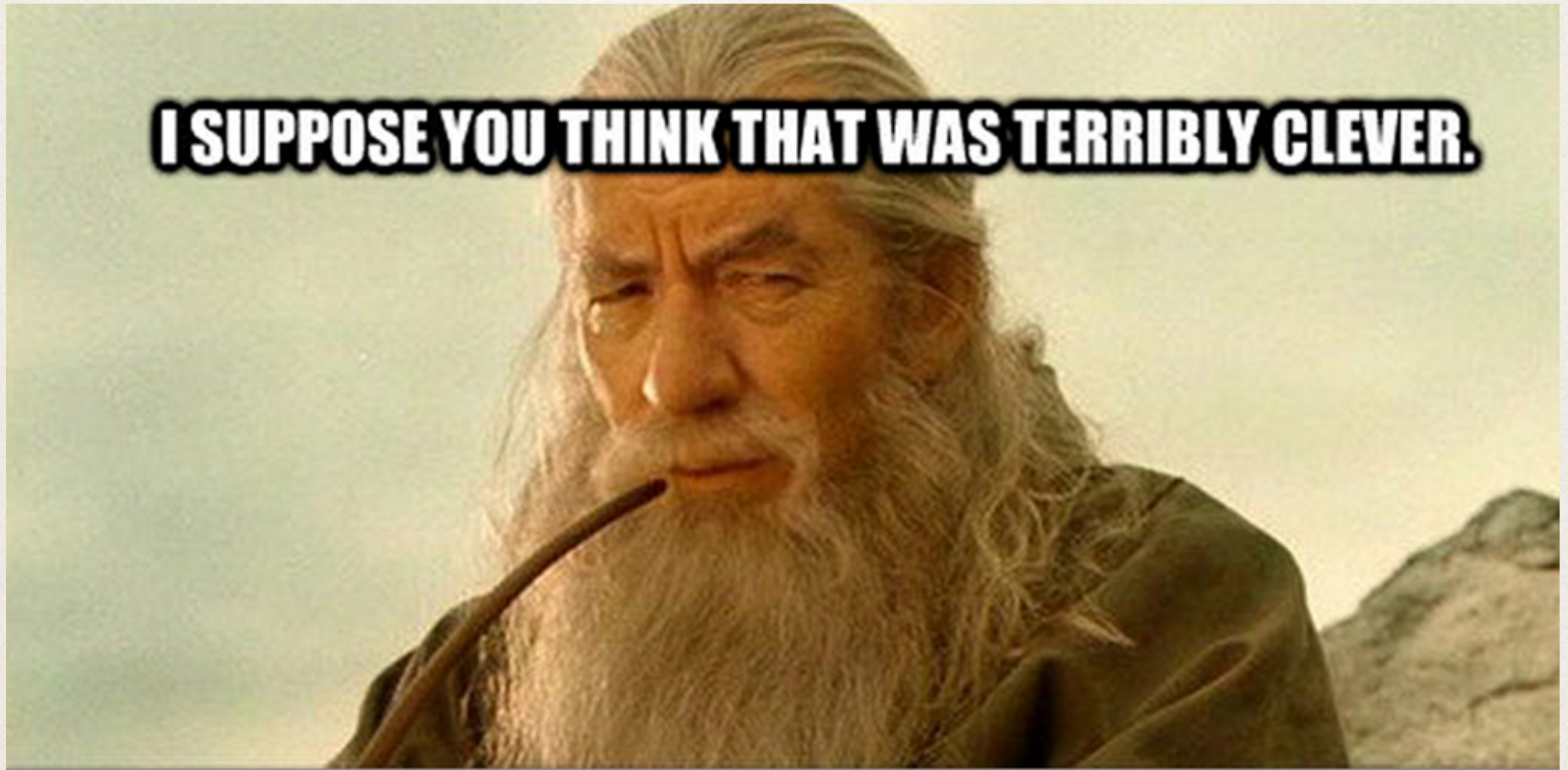
```
~> You shall not pass! -Gandalf The Grey
```

gandalf\_quote2 = "Yes, yes my dear sir and I do know your name Mr. Bilbo Baggins. And you do know my name, though you don't remember that I belong to it. I am Gandalf, and Gandalf, means me."

gandalf\_quote2 = "Yes, yes my dear sir and I do know your name Mr. Bilbo Baggins. And you do know my name, though you don't remember that I belong to it. I am Gandalf " + the\_grey + " and Gandalf " + the\_grey + " means me."



**I SUPPOSE YOU THINK THAT WAS TERRIBLY CLEVER.**





# Lookaheads

---

✦ Negative - ?!

✦ Positive - ?=



# Negative Lookahead

(?!foo)

Asserts that what  
immediately follows the  
current position in the string  
is not foo

# Positive Lookahead

(?=foo)

Asserts that what  
immediately follows the  
current position in the string  
is foo



# Capture Groups

---

() allow us to group a RegEx  
together

Groups are numbered 1-99



You can call a Group using \$ and  
the number

E.g. \$1

To not capture a group in parens  
?:



/.\*\..com\/(?:(?:groups\/videos\)|  
(?:ondemand|channels)(?:(?:\Vless\)|(?:\V  
\w\*\V))|(?:video\V))?([0-9]+).\*/

/.\*\comV

(?:(?:groups\videosV)|(?:ondemand|  
channels)(?:(?:VlessV)|(?:V\w\*V))|(?:video  
V)))?([0-9]+).\*/



/.\*\comV

(?:(?:groups\videosV)

|(?:ondemand|channels)(?:(?:VlessV)|(?:V  
w\*V))|(?:videoV))?([0-9]+).\*/

/.\*\com\

(?:(?:groups\videos\

|(?:ondemand|channels)(?:(?:\less\

|(?:\w\*\|(?:video\|)?([0-9]+).\*/



/.\*\com\

(?:(?:groups\videos\

|(?:ondemand|channels)(?:(?:\less\

|(?:\w\*\

|(?:video\))?([0-9]+).\*\

/.\*\com\

(?:(?:groups\videos\

|(?:ondemand|channels)(?:(?:\less\

|(?:\w\*\

|(?:video\))?([0-9]+).\*/



/.\*\comV

(?:(?:groups\videosV)

|(?:ondemand|channels)(?:(?:VlessV)

|(?:V\w\*V))

|(?:videoV))?[0-9]+).\*\/

/.\*\comV

(?:(?:groups\videosV)

|(?:ondemand|channels)(?:(?:VlessV)

|(?:V\w\*V))

|(?:videoV))?([0-9]+).\*/



/.\*\comV

(?:(?:groups\videosV)

|(?:ondemand|channels)(?:(?:VlessV)

|(?:V\w\*V))

|(?:videoV))?([0-9]+).\*/

/.\*vimeo.com/  
(?:(?:groups/videos/)  
|(?:ondemand|channels)(?:(?:Vless/  
|(?:\w\*))  
?(?:video/))?(?:[0-9]+)?.\*



/\*\.com\

(?:(?:groups/videos\

|(?:ondemand|channels)(?:(?:\less\

|(?:\w\*\

|(?:video\))?([0-9]+).\*\

/\*vimeo.com\

(?:(?:groups/videos\

|(?:ondemand|channels)(?:(?:\less\

|(?:\w\*\

?(?:video\))?(?:[0-9]+)?.\*\

<https://vimeo.com/ondemand/less/101677664>

<https://vimeo.com/channels/everythinganimated>

<https://vimeo.com/152376671>

<https://www.youtube.com/watch?v=RvYmwo7Tlu0>



# Javascript The Good Parts (Chap 7) - Douglas Crockford

RegExer - RegEx Tester and Editor for Javascript

<http://regexr.com/>

Eloquent Javascript (Chap 9) - Marijn Haverbeke

[http://eloquentjavascript.net/09\\_regexp.html](http://eloquentjavascript.net/09_regexp.html)

MDN (Mozilla Developer Network) on Regular Expressions

[https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/  
Regular\\_Expressions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions)

Josiah's Github

<https://github.com/kickinbahr/>

[Regular\\_Expressions\\_LV\\_DotNet\\_Meetup\\_01-2016](#)