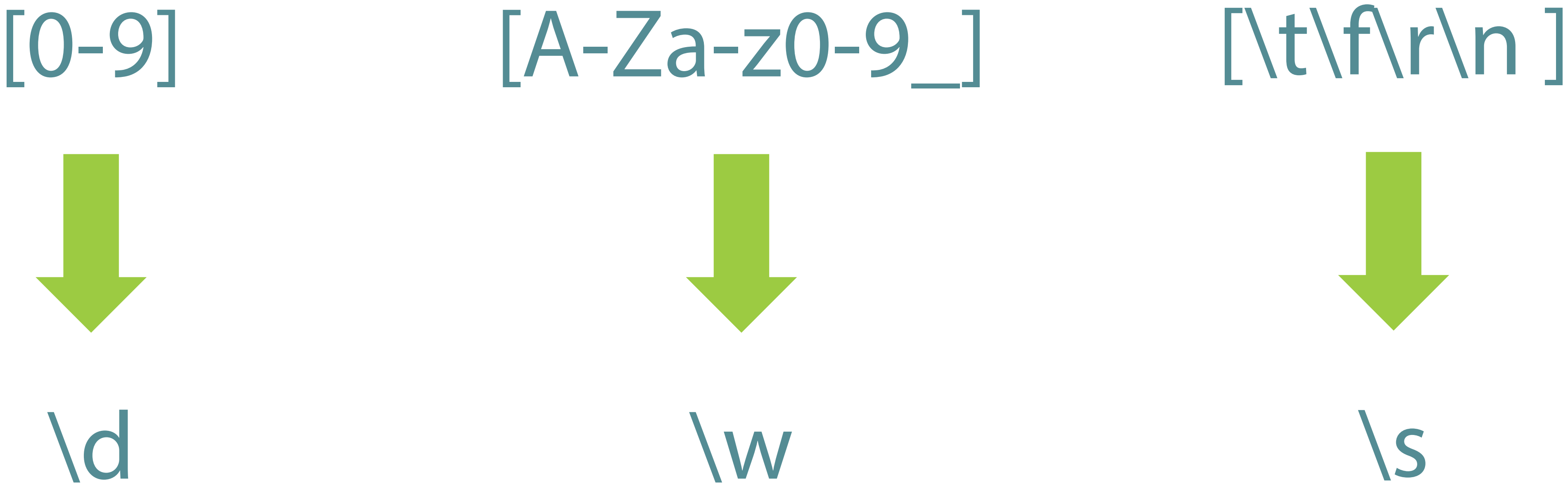
Shortcodes

Common Shortcodes



[0-9]

\d

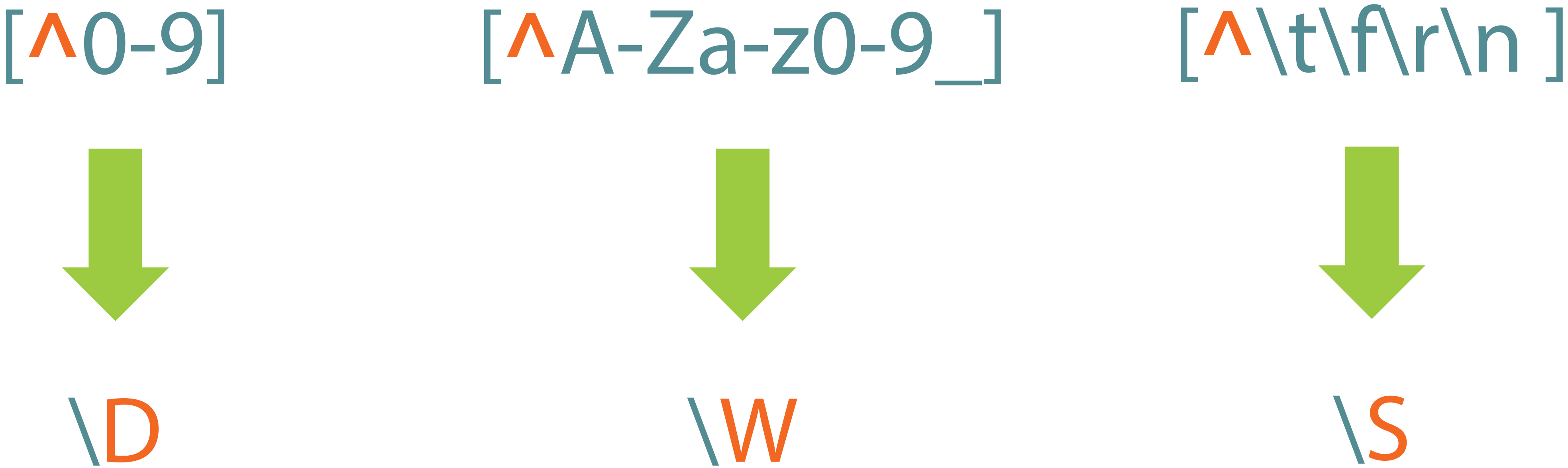
[A-Za-z0-9_]

\w

[\t\f\r\n]

\s

Negated Shortcodes

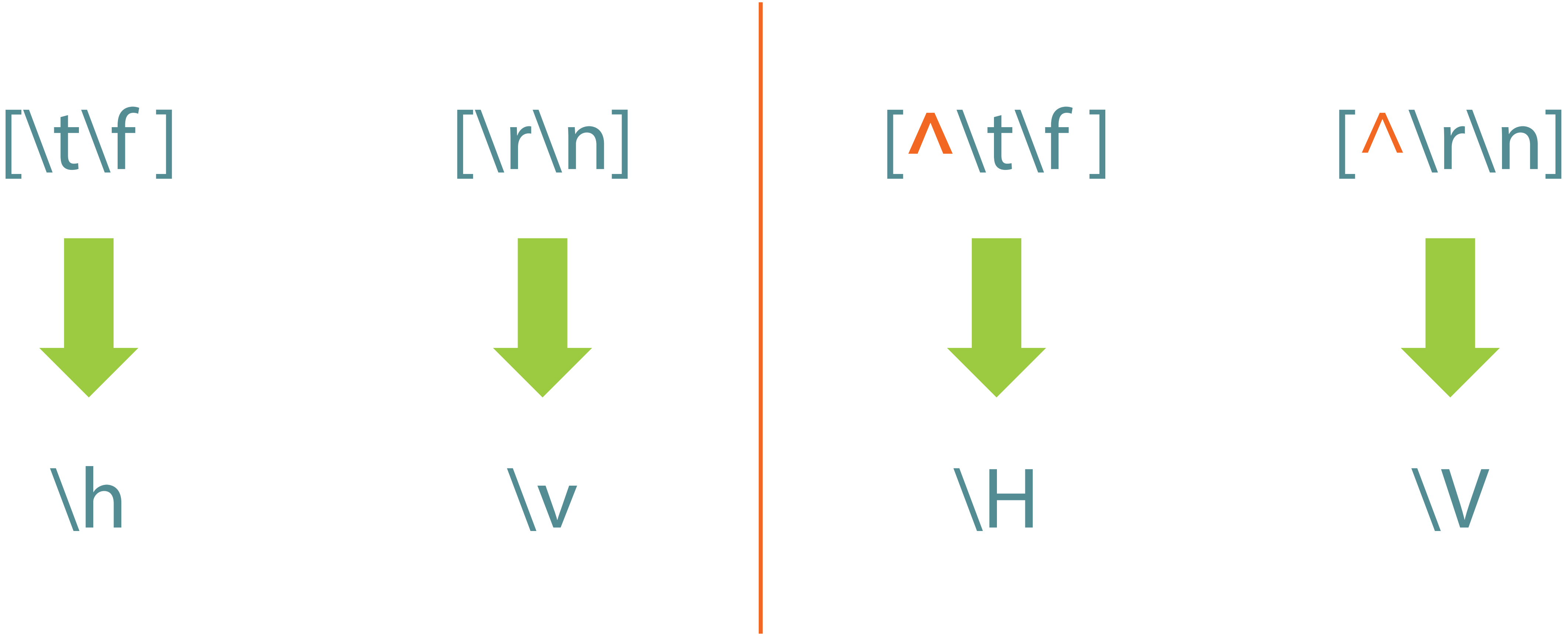


[^0-9] → \D

[^A-Za-z0-9_] → \W

[^\t\f\r\n] → \S

PCRE 7.2+ Shortcodes



[\t\f]

\h

[\r\n]

\v

[^\t\f]

\H

[^\r\n]

\V

PCRE 7.2 was released in June 2007

PCRE vs. POSIX Syntax

PCRE (common)	POSIX
\s	[:space:]
\s+	[[:space:]]+
[\s\d]+	[[:space:][:digit:]]+

Character classes		PCRE		POSIX	
[0-9]	[^0-9]	\d	\D	[[:digit:]]	[^[:digit:]]
[A-Za-z0-9_]	[^A-Za-z0-9_]	\w	\W	[[:word:]]	[^[:word:]]
[\t\f\r\n \v]	[^\t\f\r\n \v]	\s	\S	[[:space:]]	[^[:space:]]
[\t\f]	[^\t\f]	\h	\H	[[:blank:]]	[^[:blank:]]
[\r\n]	[^\r\n]	\v	\V	-	-

PCRE vs. POSIX

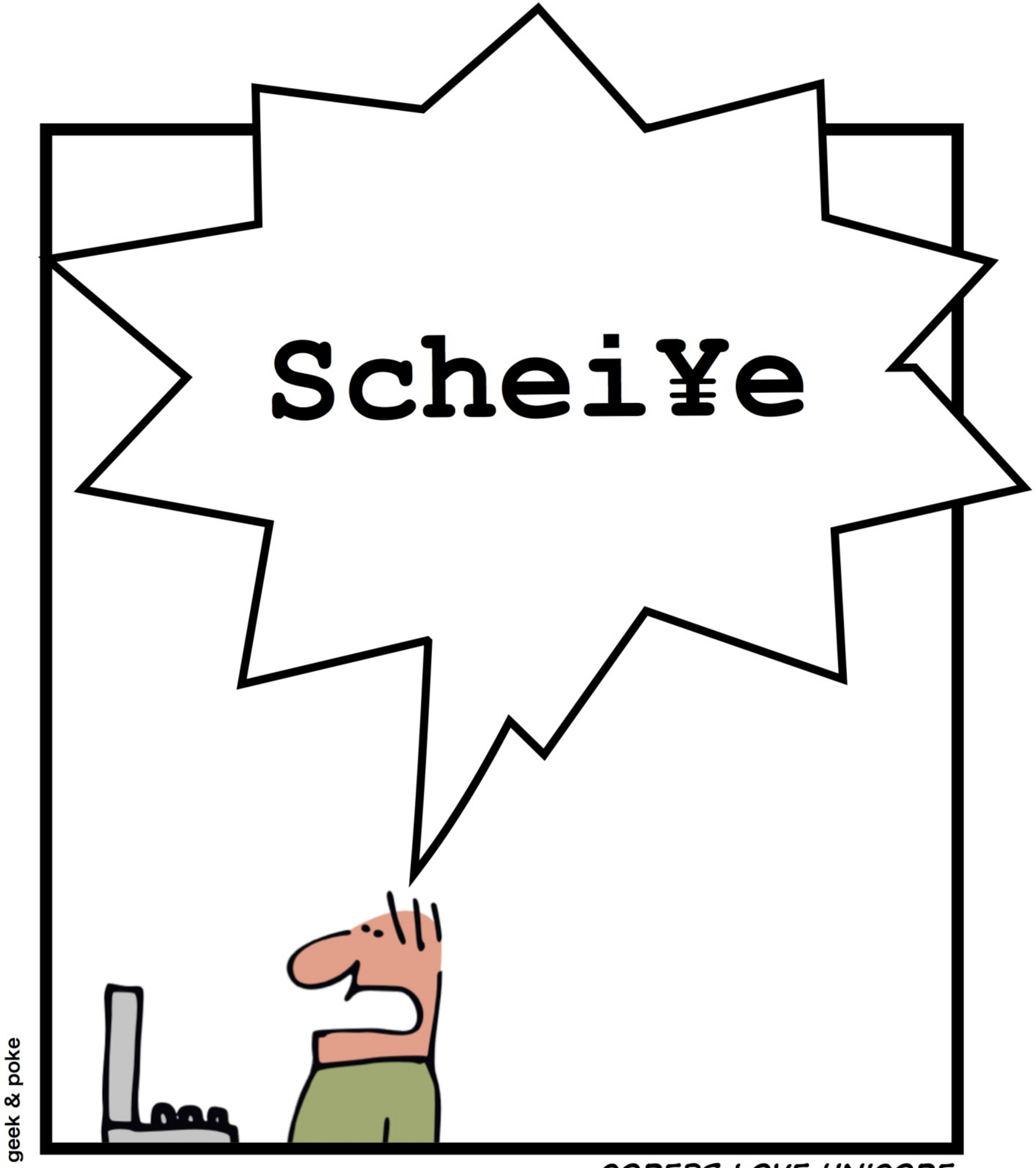
Locale

English (en)

déjà vu   [\w]+   ✗

French (fr)

déjà vu   [\w]+   ✓



CODERS LOVE UNICODE

\b

Uses \w and \W to find boundaries.

Locale

English (en)

déjà vu [\w]+ ✗

French (fr)

déjà vu [\w]+ ✓

Photo by Ilya Haykinson

Pitfalls

[^\D\S]

PCRE
vs.
POSIX

Locale

Engines
&
Implementations

Inconsistent Implementations

```
[ \f\n\r\t]
```

```
[\f\n\r\t\v\x85\p{Z}]
```

```
[ \f\n\r\t\x0B]
```

```
[ \f\n\r\t\v\u1680\u180e
\u2000\u2001\u2002
\u2003\u2004\u2005
\u2006\u2007\u2008
\u2009\u200a\u2028
\u2029\u202f\u205f\u3000]
```

```
[ \f\n\r\t\v]
```

```
[\f\n\r\t\p{Z}]
```

To Use or Not to Use ?

Advantages

- Adjust to locale

Disadvantages

- Adjust to locale
- Inconsistent implementations
- Low portability
- Difficult to unit test

Unicode Shortcodes

Is Unicode Supported ?

Engine support

Compilation

Input encoding

Graphmeme Clusters

à ↔ à

U+00E0

U+0061 + U+0300

Codepoints vs. Graphmeme Clusters

à ↔ à

U+00E0

U+0061 + U+0300

Codepoints vs. Graphmeme Clusters

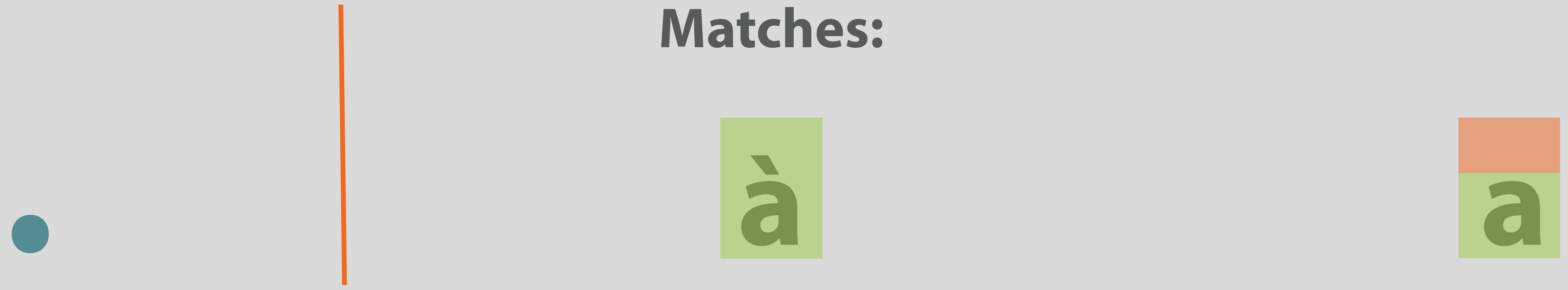
à

à

U+00E0

U+0061 + U+0300

Matches:



Unicode Wildcard *

\X

- Matches graphmeme clusters
  - ~ equivalent to \P{M}\p{M}*+
- Matches new line

* Included in the PCRE standard,
but not widely supported (yet)

Unicode Range Identifiers

- Categories
- Scripts
- Blocks
- Binary properties

Unicode Shortcode Syntax

Positive

```
\p{Identifier}
```

Negative

```
\P{Identifier}
```

Unicode Shortcodes

\p{..}

\P{..}

- {Identifier}
- Mind: Capitalization
- Can be used anywhere

Unicode Range Identifiers

- Categories
- Scripts
- Blocks
- Binary properties

Category-based Unicode Shortcodes

Letters	\p{L}
Marks	\p{M}
Numbers	\p{N}
Punctuation	\p{P}
Symbols	\p{S}
Separators	\p{Z}
Other	\p{C}

\p{Ll}	Letter: Lowercase
\p{Lm}	Letter: Mark/modifier
\p{Lo}	Letter: Other
\p{Lt}	Letter: Titlecase
\p{Lu}	Letter: Uppercase

Alternative Syntaxes: ⚠

\pX

\p{Category}

\w ≠ \p{L}

Close approximation:

```
[\p{L}\p{M}\p{Nd}\p{Nl}\p{Pc}\u200c\u200d]
```

Unicode Range Identifiers

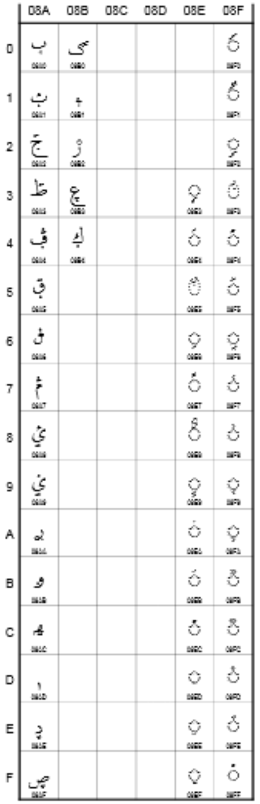
Categories

Scripts

Blocks

Binary properties

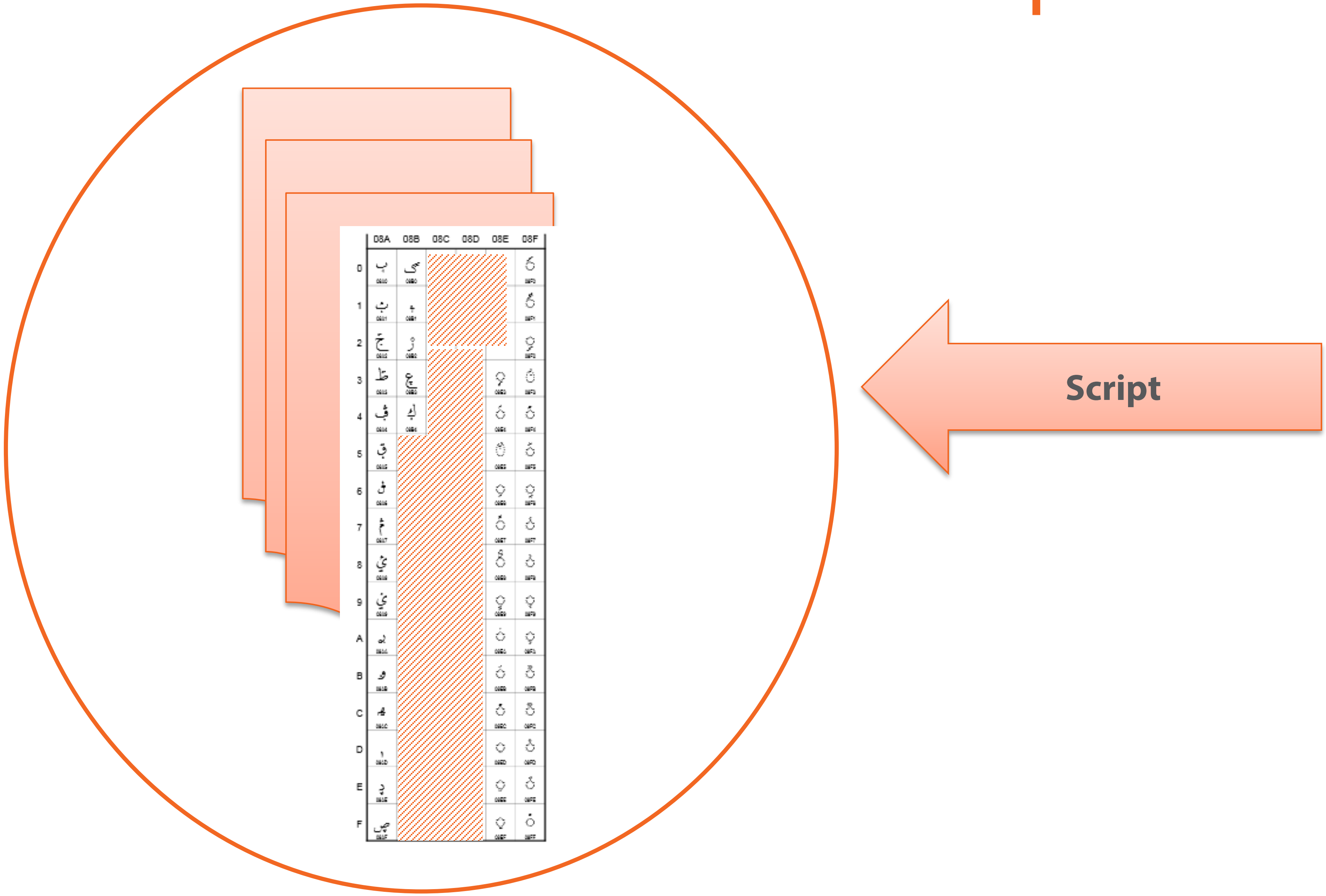
Blocks vs. Scripts



	08A	08B	08C	08D	08E	08F
0	ࢠ 08A0	ࢰ 08B0				◌ࣰ 08F0
1	ࢡ 08A1	ࢱ 08B1				◌ࣱ 08F1
2	ࢢ 08A2	ࢲ 08B2				◌ࣲ 08F2
3	ࢣ 08A3	ࢳ 08B3			◌ࣣ 08E3	◌ࣳ 08F3
4	ࢤ 08A4	ࢴ 08B4			◌ࣤ 08E4	◌ࣴ 08F4
5	ࢥ 08A5				◌ࣥ 08E5	◌ࣵ 08F5
6	ࢦ 08A6				◌ࣦ 08E6	◌ࣶ 08F6
7	ࢧ 08A7				◌ࣧ 08E7	◌ࣷ 08F7
8	ࢨ 08A8				◌ࣨ 08E8	◌ࣸ 08F8
9	ࢩ 08A9				◌ࣩ 08E9	◌ࣹ 08F9
A	ࢪ 08AA				◌࣪ 08EA	◌ࣺ 08FA
B	ࢫ 08AB				◌࣫ 08EB	◌ࣻ 08FB
C	ࢬ 08AC				◌࣬ 08EC	◌ࣼ 08FC
D	ࢭ 08AD				◌࣭ 08ED	◌ࣽ 08FD
E	ࢮ 08AE				◌࣮ 08EE	◌ࣾ 08FE
F	ࢯ 08AF				◌࣯ 08EF	◌ࣿ 08FF

Codepoint Block

Blocks vs. Scripts



Script

Script

- \p{Scriptname}
- \p{IsScriptName}
- \p{script=ScriptName}
- \p{sc=ScriptName}

Block

- \p{Blockname}
- \p{InBlockName}
- \p{IsBlockName} XML
- \p{block=BlockName}
- \p{blk=BlockName}

\p{Cyrillic}

\p{InCyrillic}

\p{InCyrillic_Supplementary}

Photo by Michael Coghlan

Modifiers

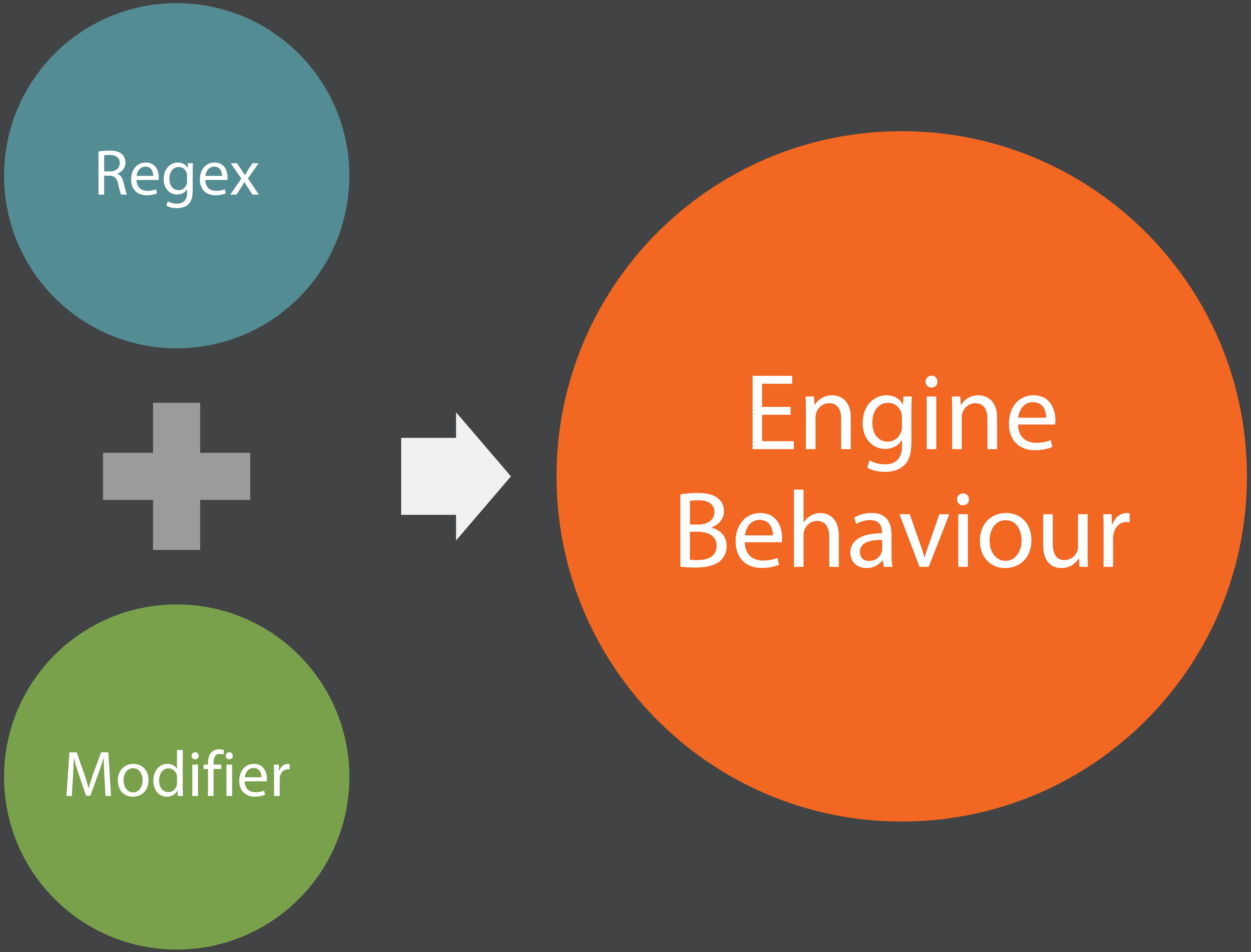

Regex
Modifier
Engine Behaviour

Photo by Dean Hochman

```
/[a-z0-9]+/im
```

Modifiers

Applying Modifiers

/regex/m

m/regex/

match('regex', modifiers)

new Re(/regex/, flags)

preg_match()

vs.

preg_match_all()

(?m)

g* i m s x

Modifiers

g*

i

m

s

x

g

- GLOBAL
- Return all matches vs. first match
- Non-overlapping

Modifiers

g*

i

m

s

x

i

- CASE-INSENSITIVE
- Mind locales
  - German: FUSSBALL vs. fußball

Modifiers

g*

i

m

s

x

m

- MULTILINE
- Affects ^ and $ behaviour

Modifiers

g*

i

m

s

x

s

- DOTALL or SINGLELINE
- Affects . (dot) to match \n
- Slow ^{n}

Modifiers

g*

i

m

s

x

X

- EXTENDED

```
/^((
   25[0-5]|                # Match 250-255 range
   2[0-4][0-9]|            # Match 200-249 range
   [01]?[0-9]{1,2}         # Match   0-199 range
)\.){3}                    # Repeat 3 times with period
(25[0-5]|2[0-4][0-9]|[01]?[0-9]{1,2}) # and once without
$/x
```

Modifiers

g*

i

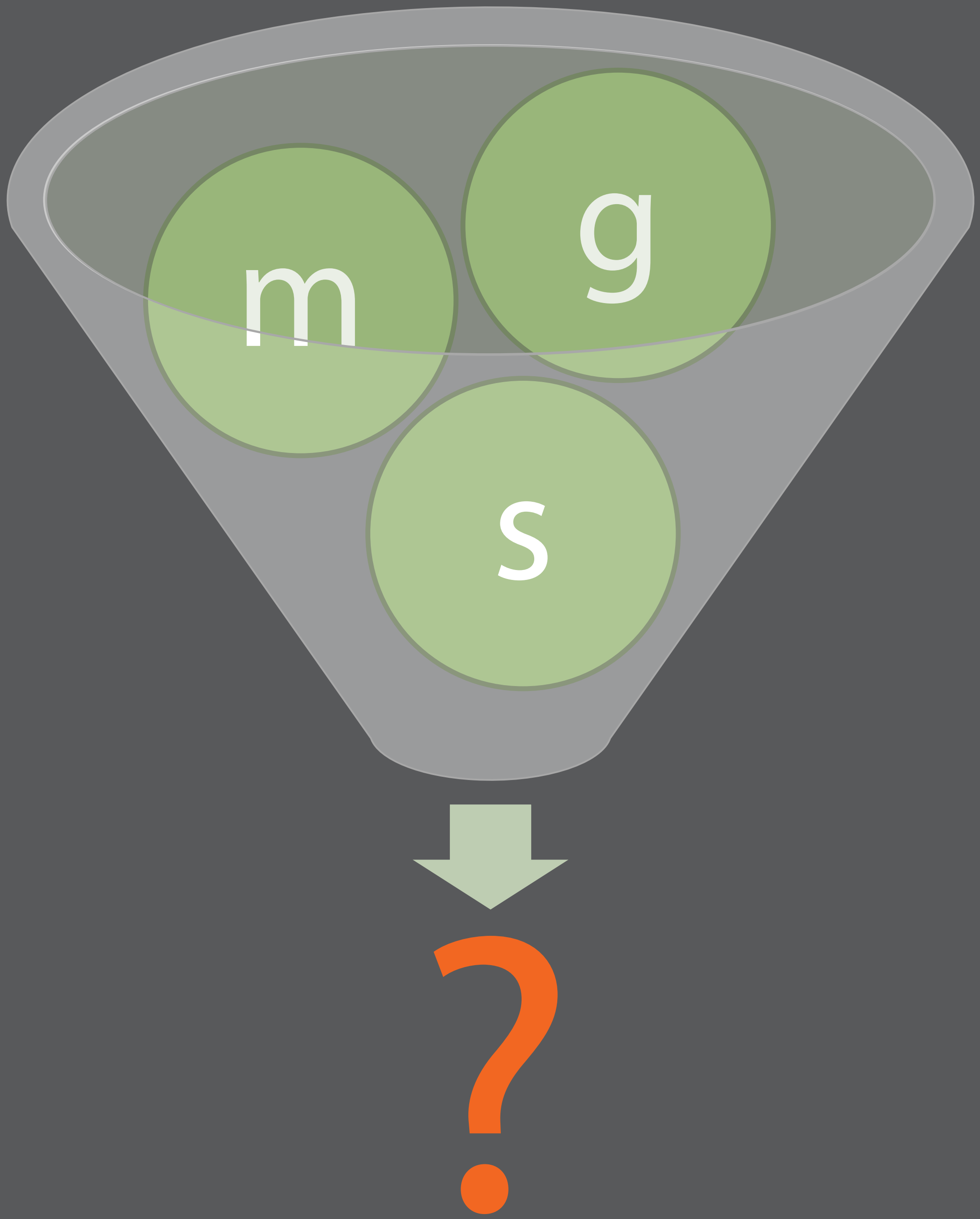
m

s

x

X

- EXTENDED
- Ignore whitespace & # to end of line
- Mind: escaping
  - \x20
  - \#



m
g
s
?

Inline Modifiers

Setting:

(?i)

```
(?i)caseless(?-i)cased(?i)caseless
```

```
cased(case(?i)insen|sitive)cased
```

Inline Modifiers

Setting:

(?i)

Unsetting:

(?-i)

Combined:

(?im-sx)

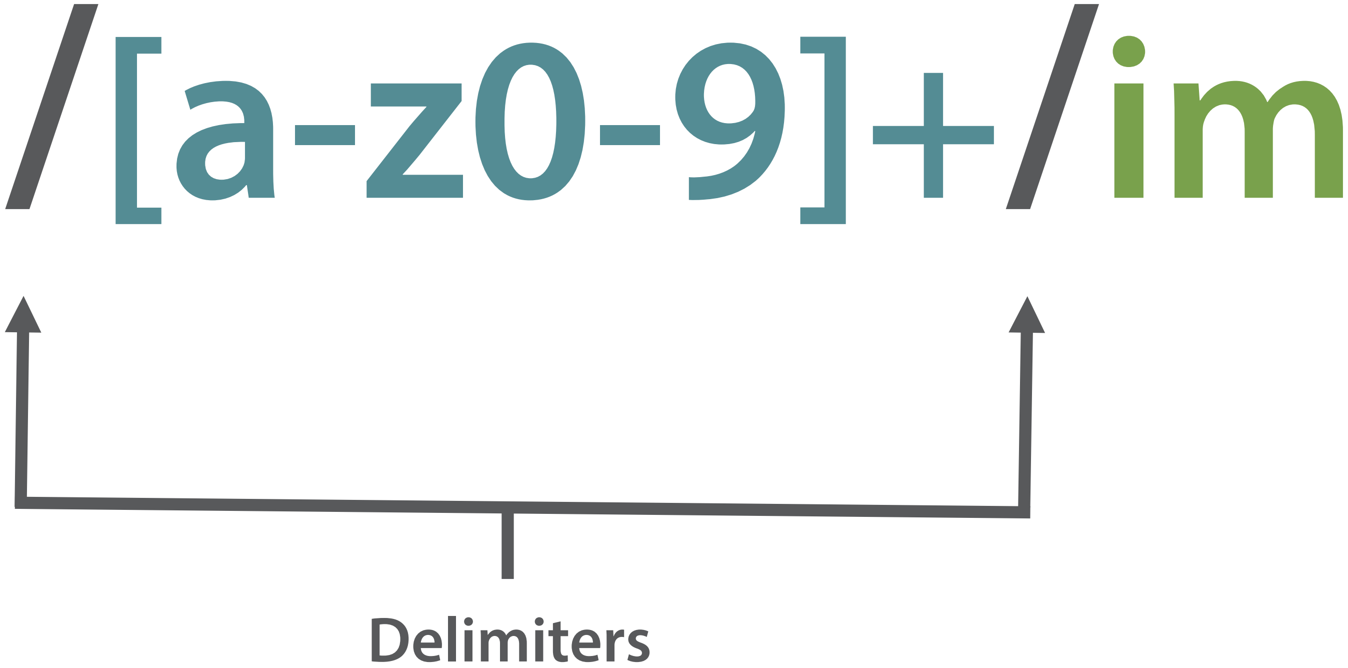
Apply to subpattern (non-capturing):

(?i:subp)

Explore

S	*Pattern Analysis*
U	*Ungreedy*
l	Locale
p	Preserve
y	*Sticky*
D	*Dollar end-only*
a	*Ascii*
r	Return replacement, don't modify
u	*UTF-8*
e	*EUC-JP*
o	Interpolation only
c	No position reset

Delimiters



/[a-z0-9]+/im

Delimiters

Delimiters

/.../

- Enclose the pattern
- Not always needed
- Alternative delimiters

Alternative Delimiter Requirements

Non-alphanumeric

Non-backslash

Non-whitespace

Alternative Delimiters

```
![0-9]+!
#[0-9]+#
@[0-9]+@
`[0-9]+`
~[0-9]+~
%[0-9]+%
```

Did you know ?

You can use brackets as delimiters:

```
(p[at]{2}te(rn))   {p[at]{2}te(rn)}
[p[at]{2}te(rn)]   <p[at]{2}te(rn)>
```

```
/http:\/\/\p{L}+\.[a-z]+\//
```

vs.

```
`http://\p{L}+\.[a-z]+/`
```

Choose Wisely

Next up:

Working with Matches

Juliette Reinders Folmer

@jrf_nl | regexcheatsheets.com