/ Reg(ex){2}plained /

Einführung in Reguläre Ausdrücke

Fabian Zimmermann

Motivation

Some developers, when confronted with a problem say:

“I know! I’ll use regular expressions!”

Now they have two problems.

Damian Conway:

Everything You Know About Regexes Is Wrong

Was sind regex?

```
/ a(.*b){2}\1 /g
/ a(.*b){2}\1 /g
/ search / flags
/ a(.*b){2}\1 / xyz$1 /g
/ a(.*b){2}\1 / xyz$1 /g
/ search / replace / flags
```

Anwendungen

Programmiersprachen

Perl

```
$text ~= s/a*/b/g
$0, $1, $2
```

Perl

$text ~= s/a*/b/g

$0, **$1**, $2

Java

```
Pattern p = Pattern.compile("a*");
Matcher matcher = p.matcher(text);
matcher.replaceAll("b")
```

Python

```
import re
text = re.sub(r'a*', r'b', text)
```

Typescript

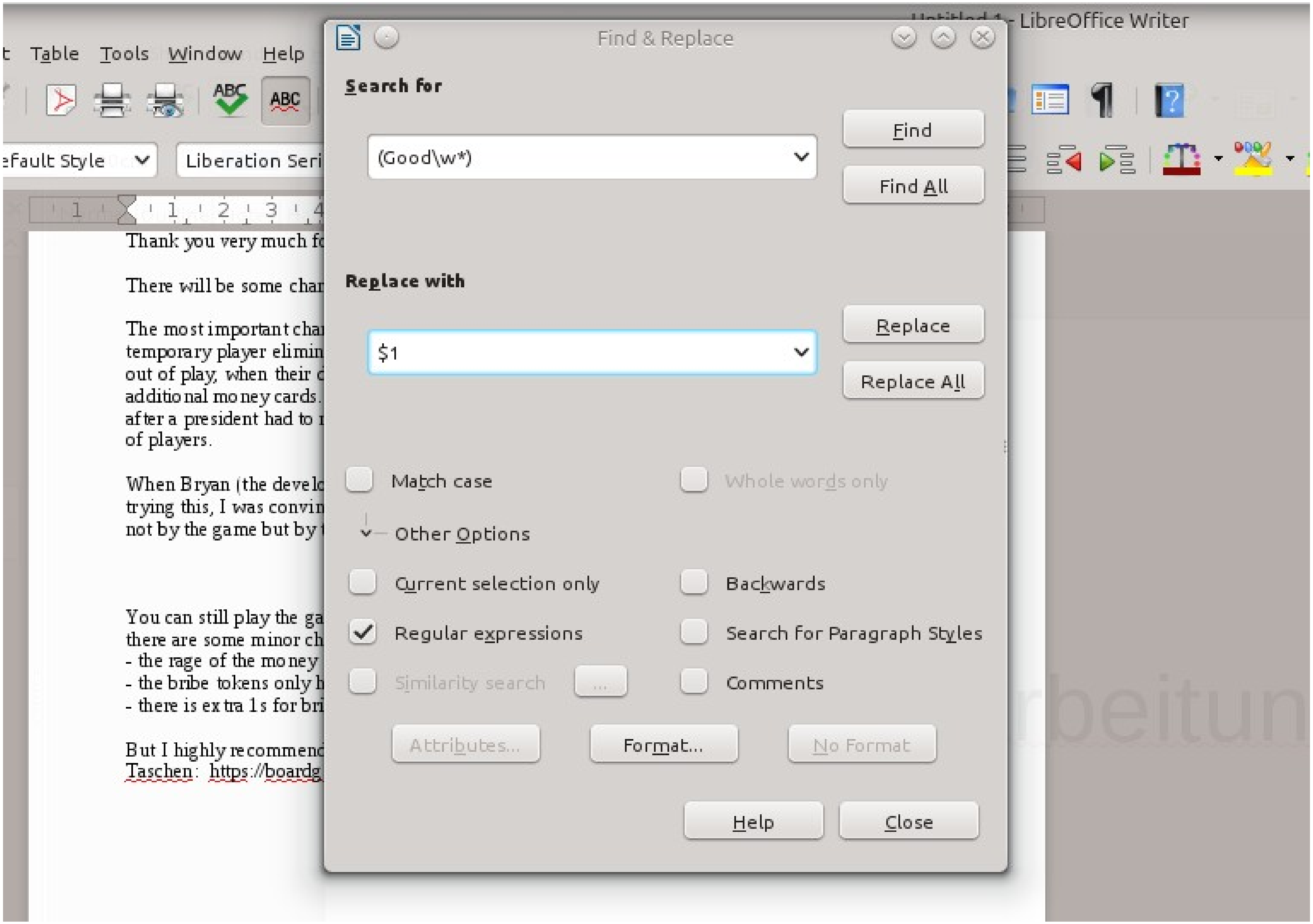
```
let regex = /a*/g;
text = text.replace(re, "b");
```

Typescript

```
let regex = RegEx(a*);
text = text.replace(re, "b");
```

command line tools

Textverarbeitung



Untitled 1 - LibreOffice Writer

Thank you very much fo

There will be some char

The most important char
temporary player elimin
out of play, when their c
additional money cards.
after a president had to r
of players.

When Bryan (the develo
trying this, I was convin
not by the game but by t

You can still play the ga
there are some minor ch
- the rage of the money
- the bribe tokens only h
- there is extra 1s for bri

But I highly recommend
Taschen: https://boardg

Find & Replace

Search for: (Good\w*)

Replace with: $1

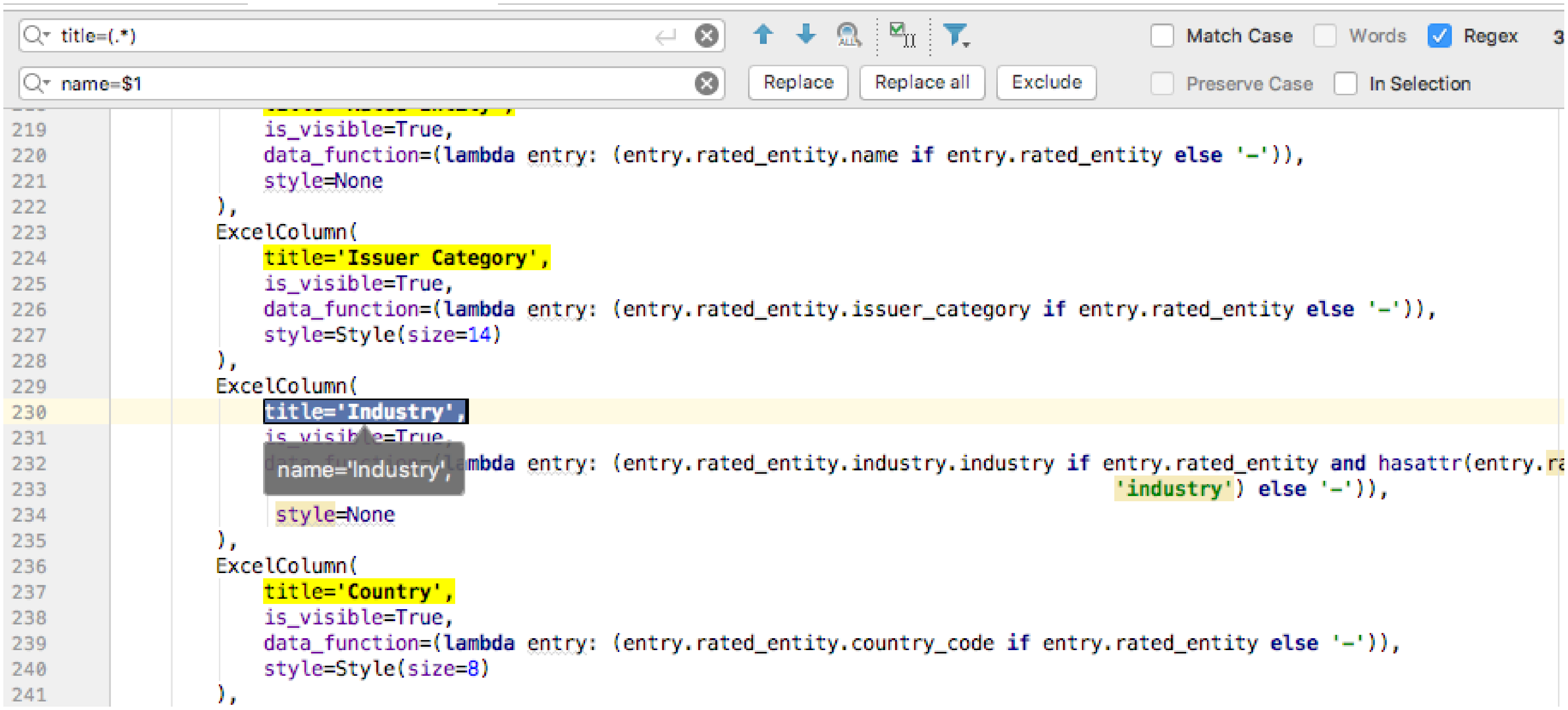
Match case: ☐ Whole words only: ☐

Current selection only: ☐ Backwards: ☐

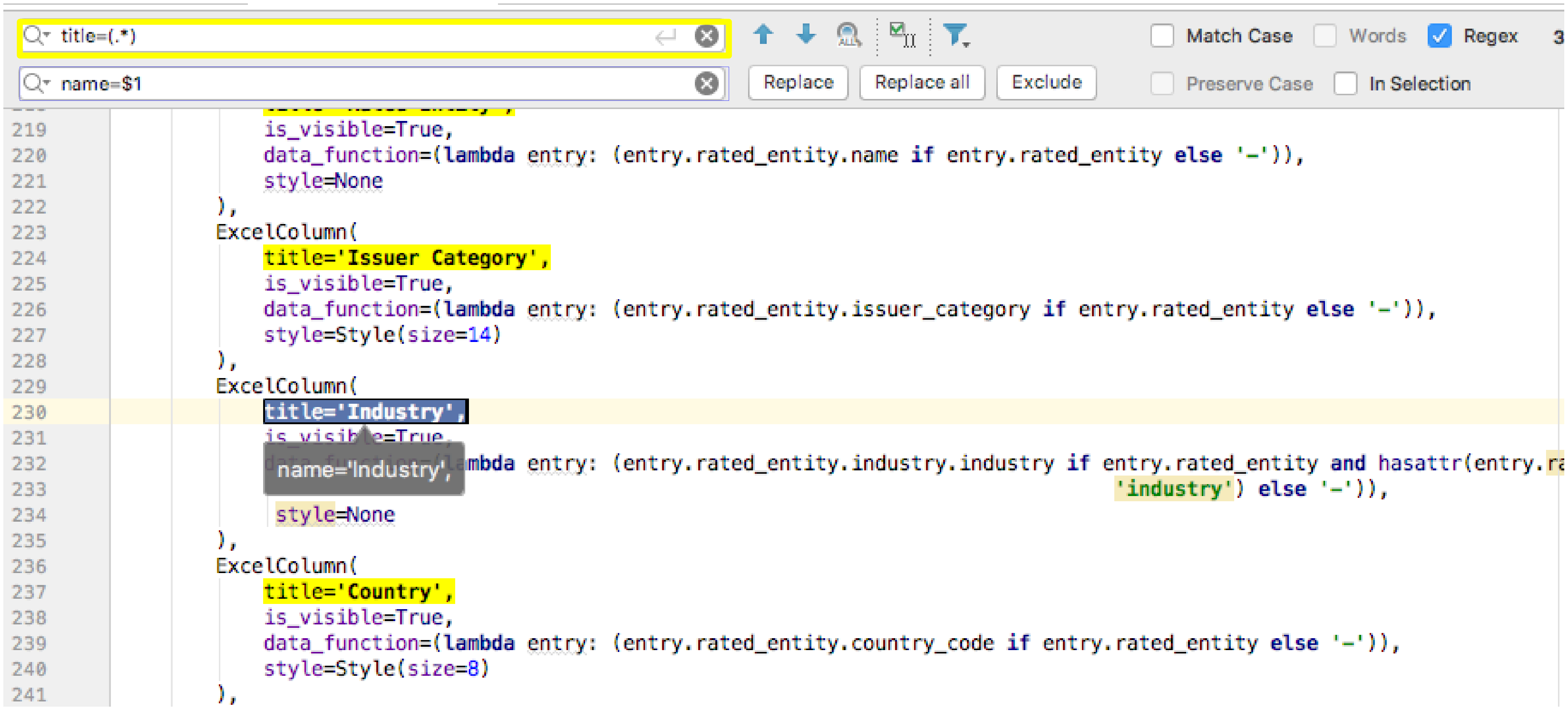
Regular expressions: ☑ Search for Paragraph Styles: ☐

Similarity search: ☐ Comments: ☐

IDEs



```
            is_visible=True,
            data_function=(lambda entry: (entry.rated_entity.name if entry.rated_entity else '-')),
            style=None
        ),
        ExcelColumn(
            title='Issuer Category',
            is_visible=True,
            data_function=(lambda entry: (entry.rated_entity.issuer_category if entry.rated_entity else '-')),
            style=Style(size=14)
        ),
        ExcelColumn(
            title='Industry',
            is_visible=True,
            data_function=(lambda entry: (entry.rated_entity.industry.industry if entry.rated_entity and hasattr(entry.ra
                                                                          'industry') else '-')),
            style=None
        ),
        ExcelColumn(
            title='Country',
            is_visible=True,
            data_function=(lambda entry: (entry.rated_entity.country_code if entry.rated_entity else '-')),
            style=Style(size=8)
        ),
```



title=(.*)

name=$1

```
            is_visible=True,
            data_function=(lambda entry: (entry.rated_entity.name if entry.rated_entity else '-')),
            style=None
        ),
        ExcelColumn(
            title='Issuer Category',
            is_visible=True,
            data_function=(lambda entry: (entry.rated_entity.issuer_category if entry.rated_entity else '-')),
            style=Style(size=14)
        ),
        ExcelColumn(
            title='Industry',
            is_visible=True,
            data_function=(lambda entry: (entry.rated_entity.industry.industry if entry.rated_entity and hasattr(entry.ra
                                                                                  'industry') else '-')),
            style=None
        ),
        ExcelColumn(
            title='Country',
            is_visible=True,
            data_function=(lambda entry: (entry.rated_entity.country_code if entry.rated_entity else '-')),
            style=Style(size=8)
        ),
```

name='Industry',

Regex Dialekte

BRE

ERE

EMACS

VIM

PCRE

PSIX

Regex Dialekte

BRE

ERE

EMACS

VIM

PCRE

PSIX

Regex Dialekte

BRE

ERE

EMACS

VIM

PCRE

PSIX

Perl **C**ompatible **R**egular **E**xpressions

Metacharacters

. \ ( ) { [

* + $ ^ | ?

Simple Regex

/a/g

*“Hey Boss! Th**a**t was quite **a** h**a**ul we got there, hunh? W**a**sn’t it? Wh**a**t’s that? Th**a**t’s my whole sh**a**re, eh? Is it? I don’t know… M**a**ybe it’s time somebody else c**a**ll the shots **a**round here. Wh**a**t’s th**a**t? I don’t got the muscle? Well, let’s see wh**a**t the other critters h**a**ve to s**a**y **a**bout that!” –SlimJim the We**a**sel*

Simple Regex

/Boss/g

“Hey ***Boss****! That was quite a haul we got there, hunh? Wasn’t it? What’s that? That’s my whole share, eh? Is it? I don’t know… Maybe it’s time somebody else call the shots around here. What’s that? I don’t got the muscle? Well, let’s see what the other critters have to say about that!” –SlimJim the Weasel*

Simple Regex

```
/(Boss|Weasel)/g
```

*“Hey **Boss**! That was quite a haul we got there, hunh? Wasn’t it? What’s that? That’s my whole share, eh? Is it? I don’t know… Maybe it’s time somebody else call the shots around here. What’s that? I don’t got the muscle? Well, let’s see what the other critters have to say about that!” –SlimJim the Weasel*

Boss oder Weasel

Character Classes

/[abc]/g

*“Hey Boss! Th**a**t w**a**s quite **a** h**a**ul we got there, hunh? W**a**sn’t it? Wh**a**t’s that? Th**a**t’s my whole sh**a**re, eh? Is it? I don’t know… M**a**y**b**e it’s time some**b**ody else **ca**ll the shots **a**round here.*

Findet a, b oder c

Ranges gehen auch: [abc] = [a-c]

Character Classes

/[a-z]/

*“Hey Boss! T**hat was quite a haul we got there**, **hunh**?*
*W**asn’t it**? **What’s that**?*

Jeder Kleinbuchstaben von a bis z

Character Classes

\d = [0-9]

Ziffern

Character Classes

\D = [^0-9]

Nicht-Zahlen

Character Classes

\w = [a-zA-Z_0-9]

Any word character

Character Classes

\W = [^a-zA-Z_0-9]

Any non word character

Character Classes

\s = [\t\n]

Any space character

Character Classes

●

Any character (except newline)

Flag s changes behavior of .

Quantifier

/a{5}/g

“aaaab dasd ***aaaaa***

Fünf mal a

Quantifier

/b{5,7}/g

“aaaab dasd ***bbbbbb***

Zwischen 5 und 7 mal b

Quantifier

/c{10,}/g

“aaaab dasd ***ccccccccccccccccc***

Mindestens 10 mal c

Quantifier

{0,1} = ?

0 oder 1 mal

Quantifier

{1,} = +

Mindestens 1 mal

Quantifier

{0,} = *

Beliebig oft (auch 0 mal)

Greedyness

/B.*s/g

“Hey ***Boss! That was quite a haul we got there, hunh? Wasn’t it? What’s that? That’s my whole share, eh? Is it? I don’t know… Maybe it’s time somebody else call the shots*** *around here.*

Regexes mit Quantifier sind greedy

Greedyness

/B.*?s/g

*"Hey **Bos**s! That was quite a haul we got there, hunh? Wasn't it? What's that? That's my whole share, eh? Is it? I don't know… Maybe it's time somebody else call the shots around here.*

? nach einem Quantifier macht diesen non-greedy

Anchors

^: Beginn

$: Ende

Anchors

\b: Wortgrenze

\B: Nichtwortgrenze

Capture Groups

/(B.*?ss).*/g

*“Hey **Boss! That was quite a haul we got there, hunh? Wasn’t it? What’s that? That’s my whole share, eh? Is it? I don’t know… Maybe it’s time somebody else call the shots around here**.*

$1 = Boss: $1 … $n enthalten den Inhalt der 1. bis n. Capture Group

$0 enthält alles, das gematcht wurde

Backreferences

/(\d*)a\1/g

Mit \1, \2 … kann auf die 1., 2. … Capture Group referenziert werden

*a**123a123**4*

Regex = Reguläre Sprachen?

Durch back references können regex deutlich mehr matchen als nur reguläre Sprachen

Non-Capture Groups

```
/(?:a.*b)/g
```

(?:) : non-capturing group

Positive Forward Look-ahead

/the\s(?=critter)/g

GoodCritters is a game for 4-8 criminal critters who pull of heists and fight over the loot! Whoever’s the chosen boss can distribute the loot however they like, but it’s the crew ta has the final say. If the crew doesn’t like the split, they might just tell the boss to take a hike and put some other critter in charge! In the end, ***the*** *critter that collects the most valuable stash of loot wins!*

match “the “ followed by “critter”

Negative Forward Look-ahead

/the\s(?!critter)/g

GoodCritters is a game for 4-8 criminal critters who pull of heists and fight over ***the*** *loot! Whoever's* ***the*** *chosen boss can distribute* ***the*** *loot however they like, but it's* ***the*** *crew ta has* ***the*** *final say. If* ***the*** *crew doesn't like the split, they might just tell* ***the*** *boss to take a hike and put some other critter in charge! In* ***the*** *end, the critter that collects* ***the*** *most valuable stash of loot wins!*

match "the " not followed by "critter"

Password Checks

```
/^(?=.*\d)(?=.*[a-z]).{8,}$/g
```

Mindestens eine Zahl, mindestens ein Kleinbuchstabe, mindestens 8 Zeichen

Exponentieller Rechenaufwand

/(a*a*)*b/

aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

Quellen

Damian Conway:

EverythingYouKnowAboutRegexesIsWrong

http://yowconference.com.au/slides/yowwest2015/Conway-EverythingYouKnowAboutRegexesIsWrong.pdf

Lea Verou:

Best of Fluent 2012: /Reg(exp){2}lained/: Demystifying Regular Expressions

https://www.youtube.com/watch?v=EkluES9Rvak

https://regexr.com/