CookBook по регулярным выражениям (REGEX)

???????? ?????? ????????? (????????? ????? gm):

- https://regexr.com/
- https://regex101.com/

??? ?????:

- https://habr.com/ru/articles/545150/
- https://regex.sorokin.engineer/ru/latest/regular_expressions.html

? KUMA ??? ?????? ??????? ????????? ? ???????? (????????????) ?????? ???? ?????????, ?????? ?????????? "priority" ??? ??????: `(?P<priority>\d|\d{2}|1[1-8]\d|19[01])`

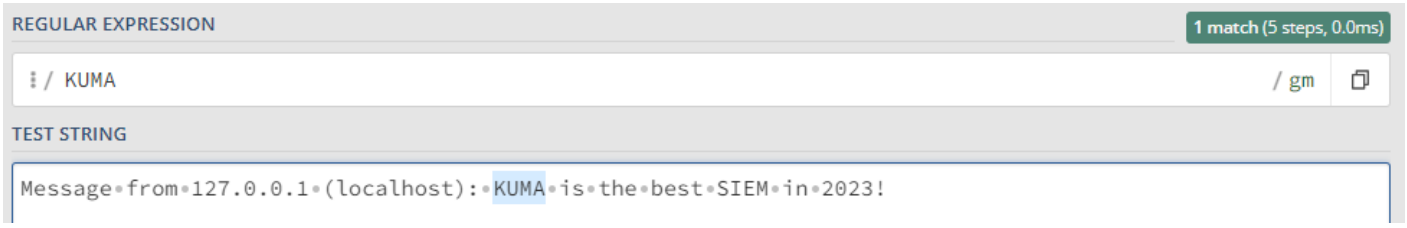
???? ?????? ?? ????? ? ????????, ?? ????? ???????????? ?? ?????????? ??????, ??????: `(?:\d|\d{2}|1[1-8]\d|19[01])`

?????????? ??????, ???????? ?????????? ?? ???????? ?????????:

```
Message from 127.0.0.1 (localhost): KUMA is the best SIEM in 2023!
```

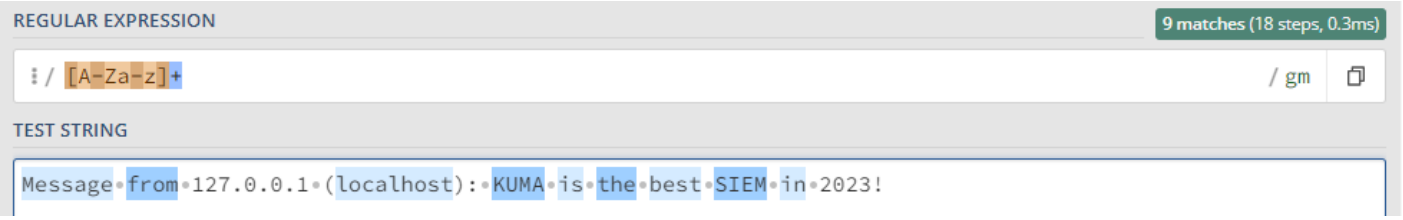
Захватить строку KUMA

KUMA *?????? ?????? ???????????? ?????? KUMA.*



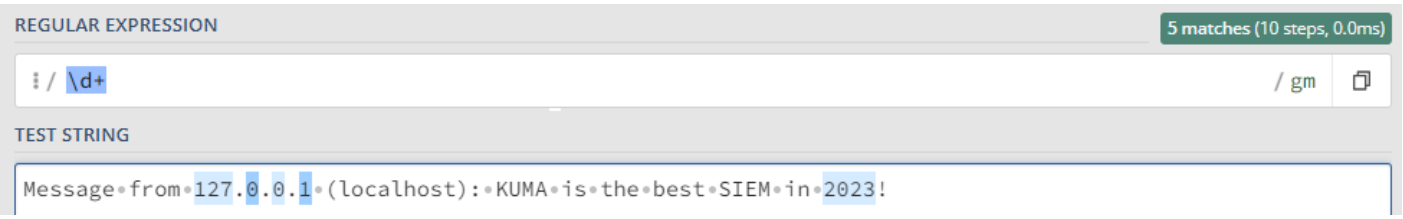
Захватить строку содержащую только буквы

[A-Za-z]+ *???? ?????? ([]) ???????? ? ???????? (A-Z) ? ?????????? (a-z) ??????? ?? ????? ? ????? (+).*



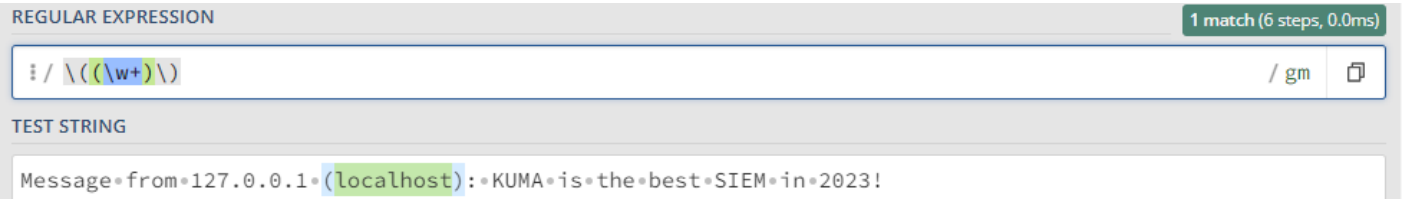
Захватить строку содержащую только числа

\d+ *???? ?? ?????? \d, ??? ???????? ???????????? [0-9] ?? ?????? ? ????? ????????? (+).*



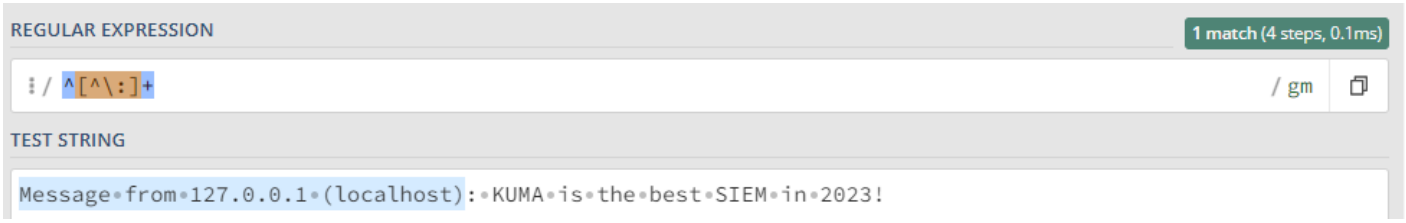
Захватить строку внутри круглых скобок

\((\w+)\) *???? ?? ?????? \w, ??? ???????? ???????????? [a-zA-Z0-9_] ?? ?????? ? ????? ????????? (+), ??? ???? ?????????? ??????? ?????? ? ??????? ????????? ????? \ ? ?????? ???? ?????????? ? ?????? ???????? ???????? ()*



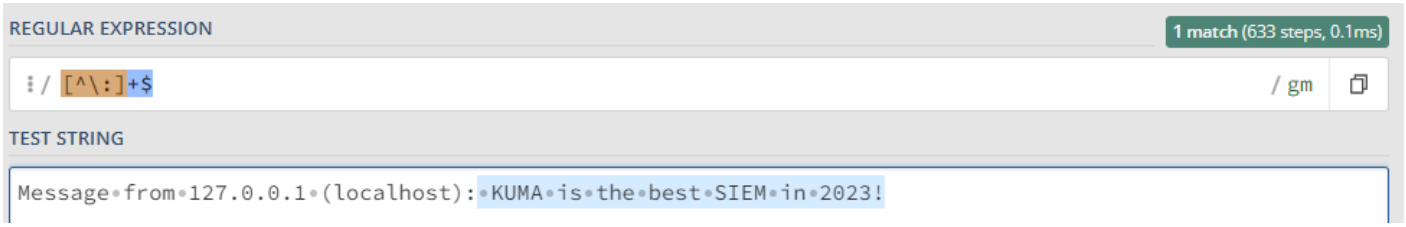
Захватить строку до двоеточия

^[^\:]+ *???? ? ?????? ?????? ^, ????? ??????????? ? ?????? ??? ????? ????????? (?????? ????????? ???????????) [^\:] ?? ?????? ? ????? ????????? (+)*



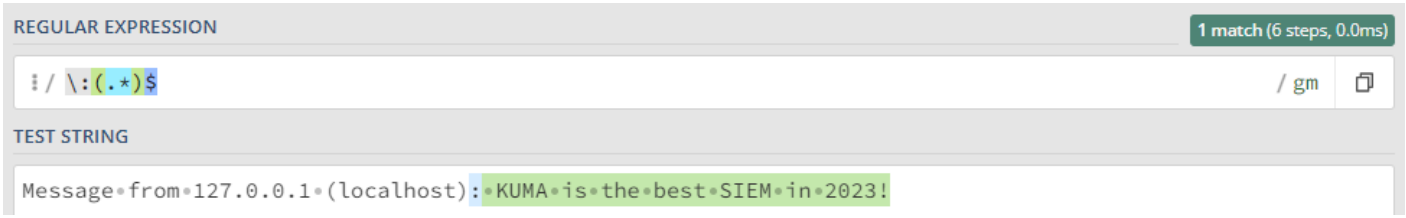
Захватить строку после двоеточия

[^\:]+$ *?????, ???????? ????????????? ????, ??????????? ?? ????????, ?.?. ??? ????? ????? ?????? (633 ????). ???? ??? ????? ????????? (?????? ????????? ???????????) [^\:] ?? ?????? ? ????? ????????? (+), ?? ?? ????? ?????? $*



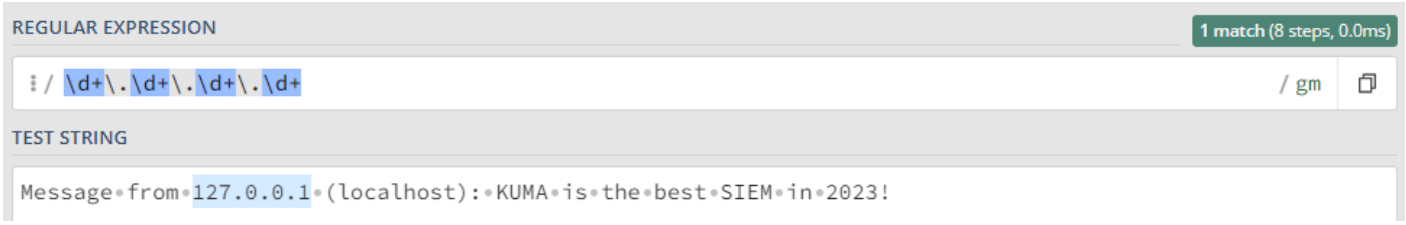
? ????? ?????? ????? ???????????? ?????????

\:(.*)$ *???? ? ?????? ????????? \:, ????? ??????????? ??? ??????? ?? ???? ? ????? ????????? (*), ? ????? ??? ??? ??? ????? ? ?????? ()*



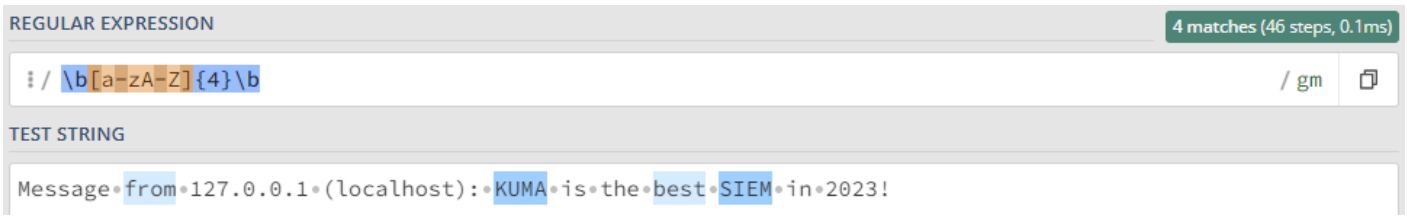
Захватить IP-адрес

\d+\.\d+\.\d+\.\d+ *???? ????? ?? ?????? ? ????? \d+, ? ?????? ? ??? 4 ????*



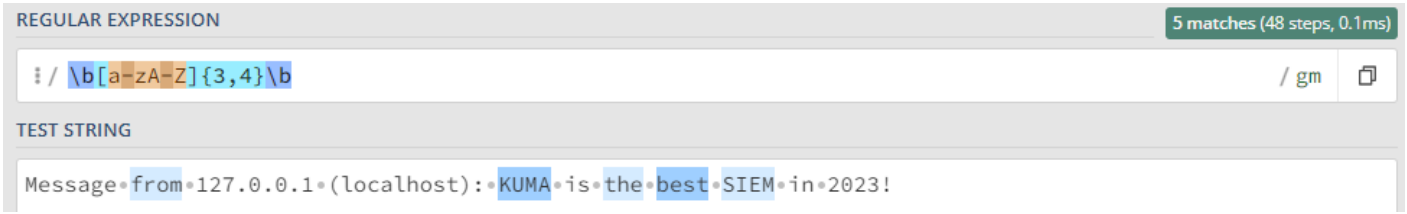
Захватить слова состоящие из 4 букв

\b[a-zA-Z]{4}\b *???? ?????? 4 ???????? ?? ???? ? ????????????? ?? (boundary) \b*



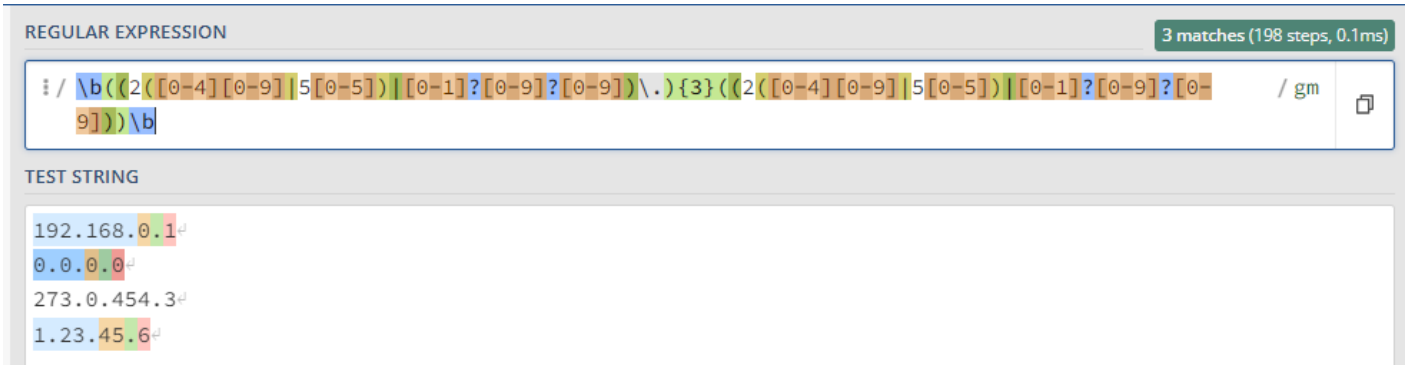
Захватить слова состоящие от 3 до 4 букв

\b[a-zA-Z]{3,4}\b *???? ?????? 4 ???????? ?? ???? ? ????????????? ?? (boundary) \b*



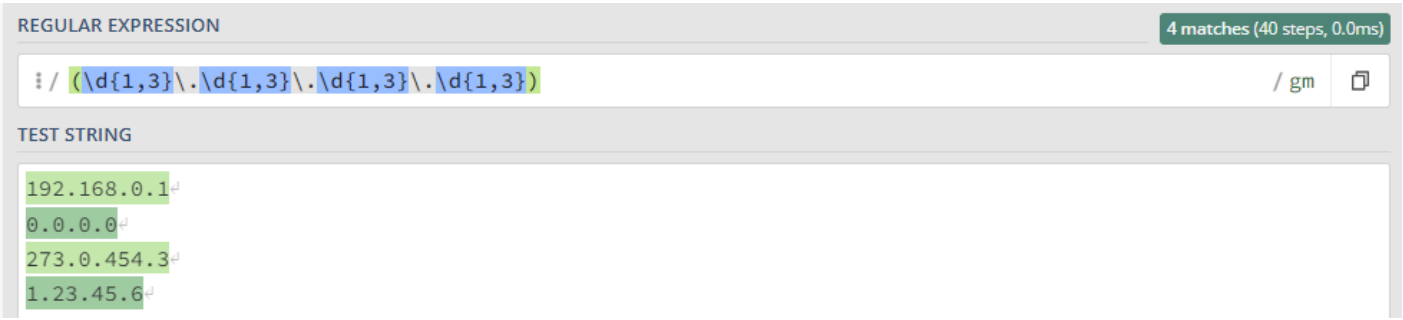
Захватить IPv4 адрес

\b((2([0-4][0-9]|5[0-5])|[0-1]?[0-9]?[0-9])\.){3}((2([0-4][0-9]|5[0-5])|[0-1]?[0-9]?[0-9]))\b



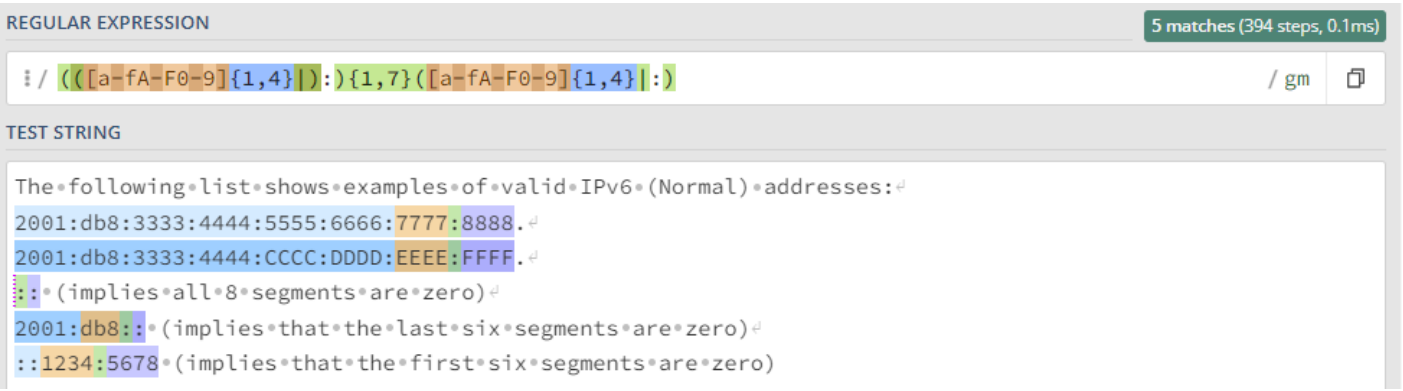
????? ??????? ???????, ?? ???????, ??? ?????????:

(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})



Захватить IPv6 адрес

(([a-fA-F0-9]{1,4}|):){1,7}([a-fA-F0-9]{1,4}|:)

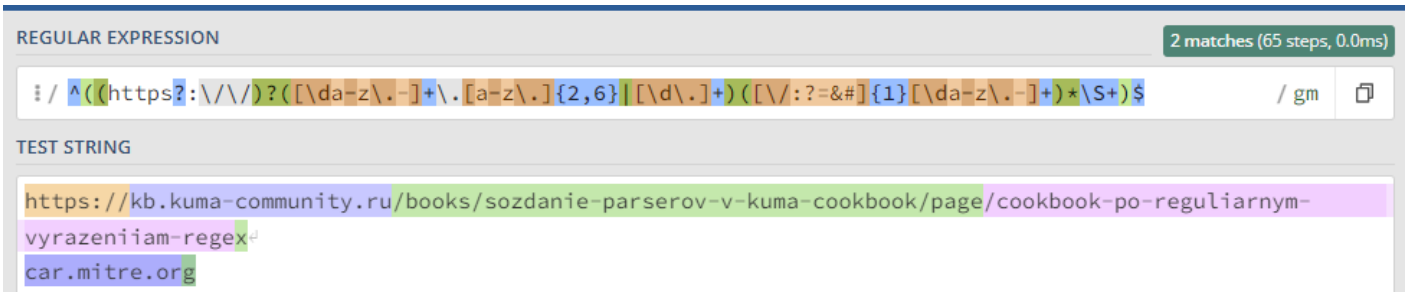


Захватить HASH сумму

- MD **^[a-fA-F0-9]{32}$**
- SHA1 **^[a-fA-F0-9]{40}$**
- SHA256 **^[a-fA-F0-9]{64}$**
- SHA512 **^[a-fA-F0-9]{128}$**

Захватить URL адрес

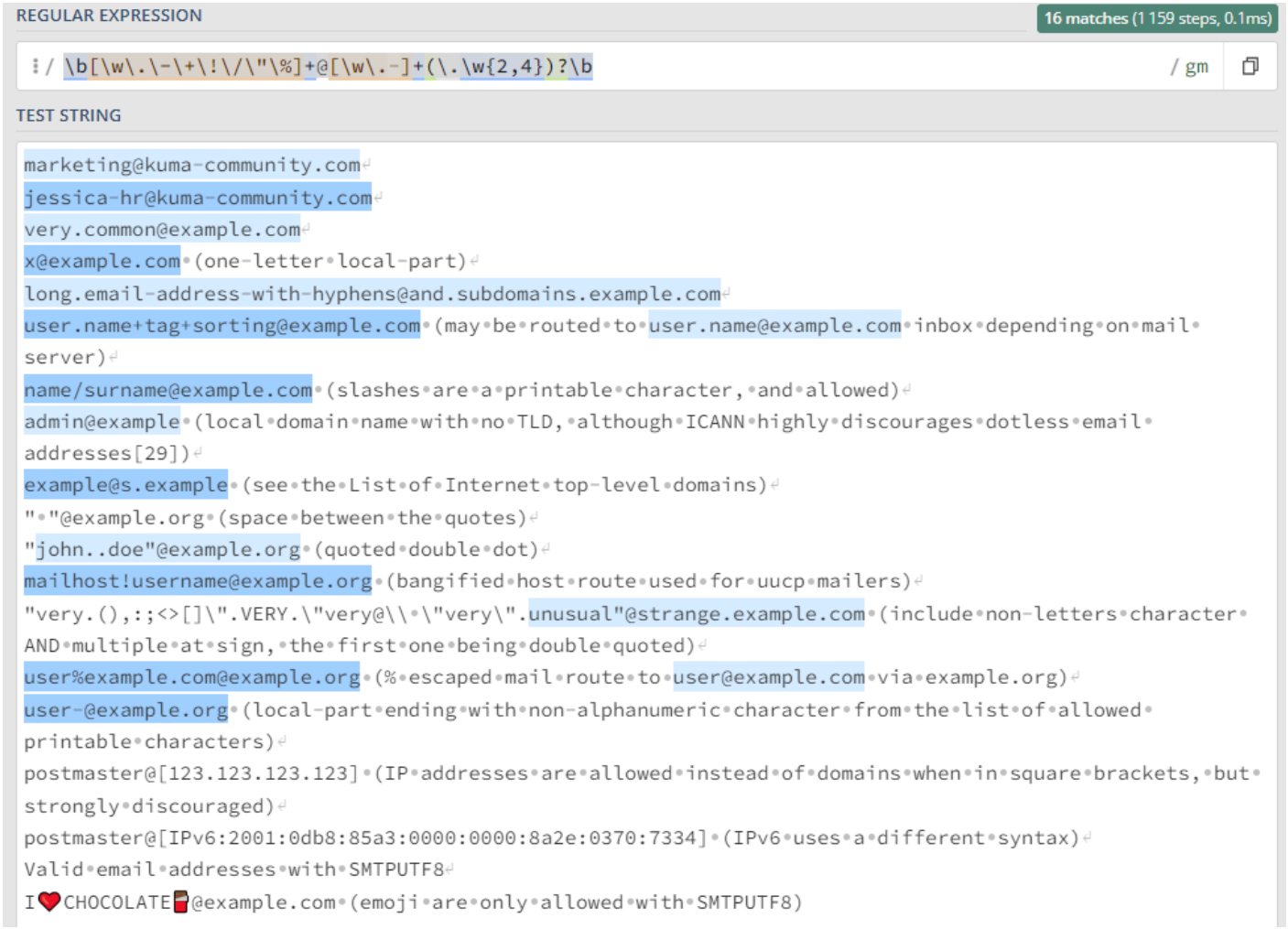
^((https?:\/\/)?([\da-z\.-]+\.[a-z\.]{2,6}|[\d\.]+)([\/:?=&#]{1}[\da-z\.-]+)*\S+)$



Захватить EMAIL адрес

?????????? ????? ??? ???? ???????? ???????

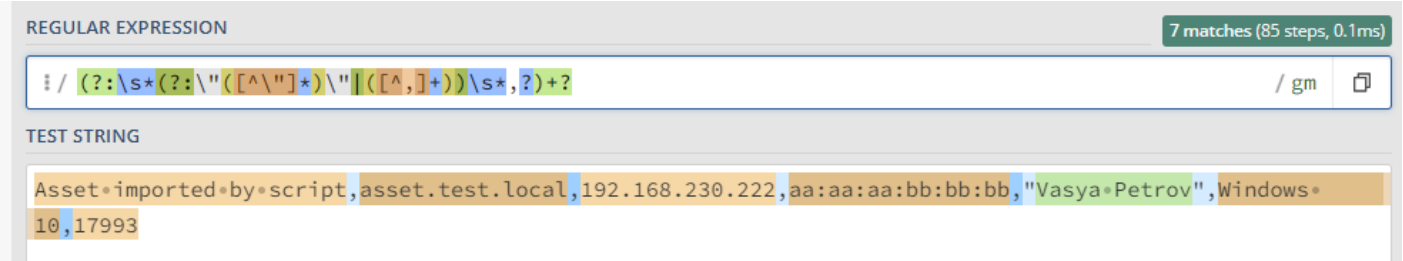
\b[\w\.\-\+\!\/\"\%]+@[\w\.-]+(\.\w{2,4})?\b



Захватить CSV структуру

????????? ?????? ?? ????????? ?? CSV

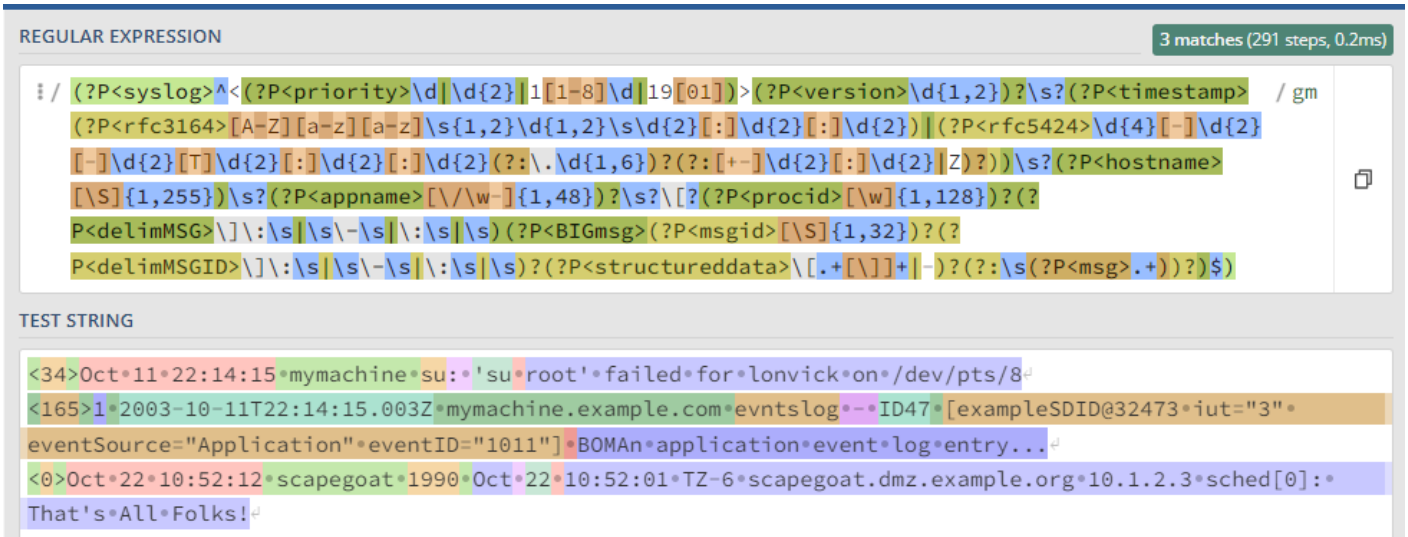
```
(?:\s*(?:\"([^\"]*)\"|([^,]+))\s*,?)+?
```



Захватить Syslog структуру

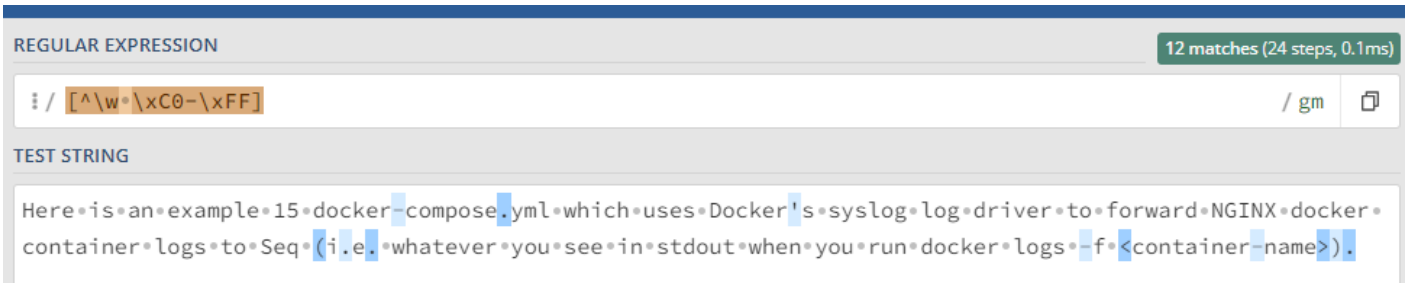
??????????? ????????? ?? rfc https://datatracker.ietf.org/doc/html/rfc5424 ? https://datatracker.ietf.org/doc/html/rfc3164

```
(?P<syslog>^<(?P<priority>\d|\d{2}|1[1-8]\d|19[01])>(?P<version>\d{1,2})?\s?(?P<timestamp>(?P<rfc3164>[A-Z][a-z][a-z]\s{1,2}\d{1,2}\s\d{2}[:]\d{2}[:]\d{2})|(?P<rfc5424>\d{4}[-]\d{2}[-]\d{2}[T]\d{2}[:]\d{2}[:]\d{2}(?:\.\d{1,6})?(?:[+-]\d{2}[:]\d{2}|Z)?))\s?(?P<hostname>[\S]{1,255})\s?(?P<appname>[\/\w-]{1,48})?\s?\[?(?P<procid>[\w]{1,128})?(?P<delimMSG>\]\:\s|\s\-\s|\:\s|\s)(?P<BIGmsg>(?P<msgid>[\S]{1,32})?(?P<delimMSGID>\]\:\s|\s\-\s|\:\s|\s)?(?P<structureddata>\[.+[\]]+|-)?(?:\s(?P<msg>.+))?)$)
```



Захватить символы (не буквы и не цифры)

```
[^\w \xC0-\xFF]
```



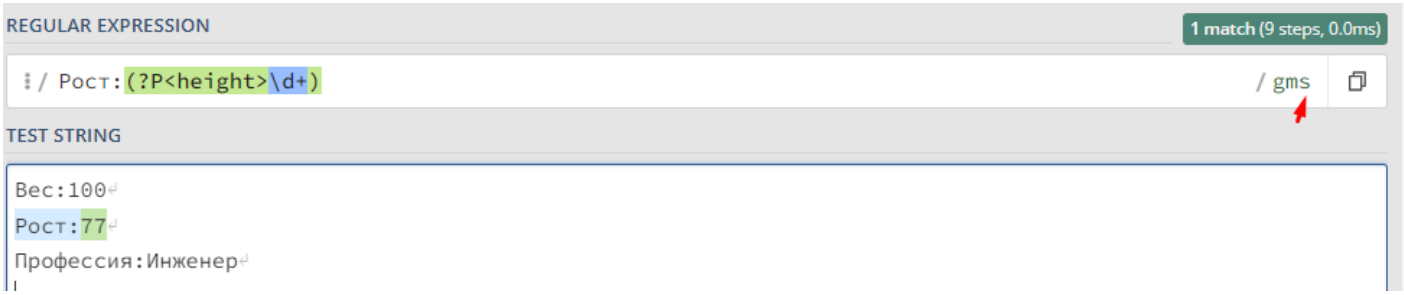
Работа с многострочным сообщением

? regexp ?????????? ????????? ?????:

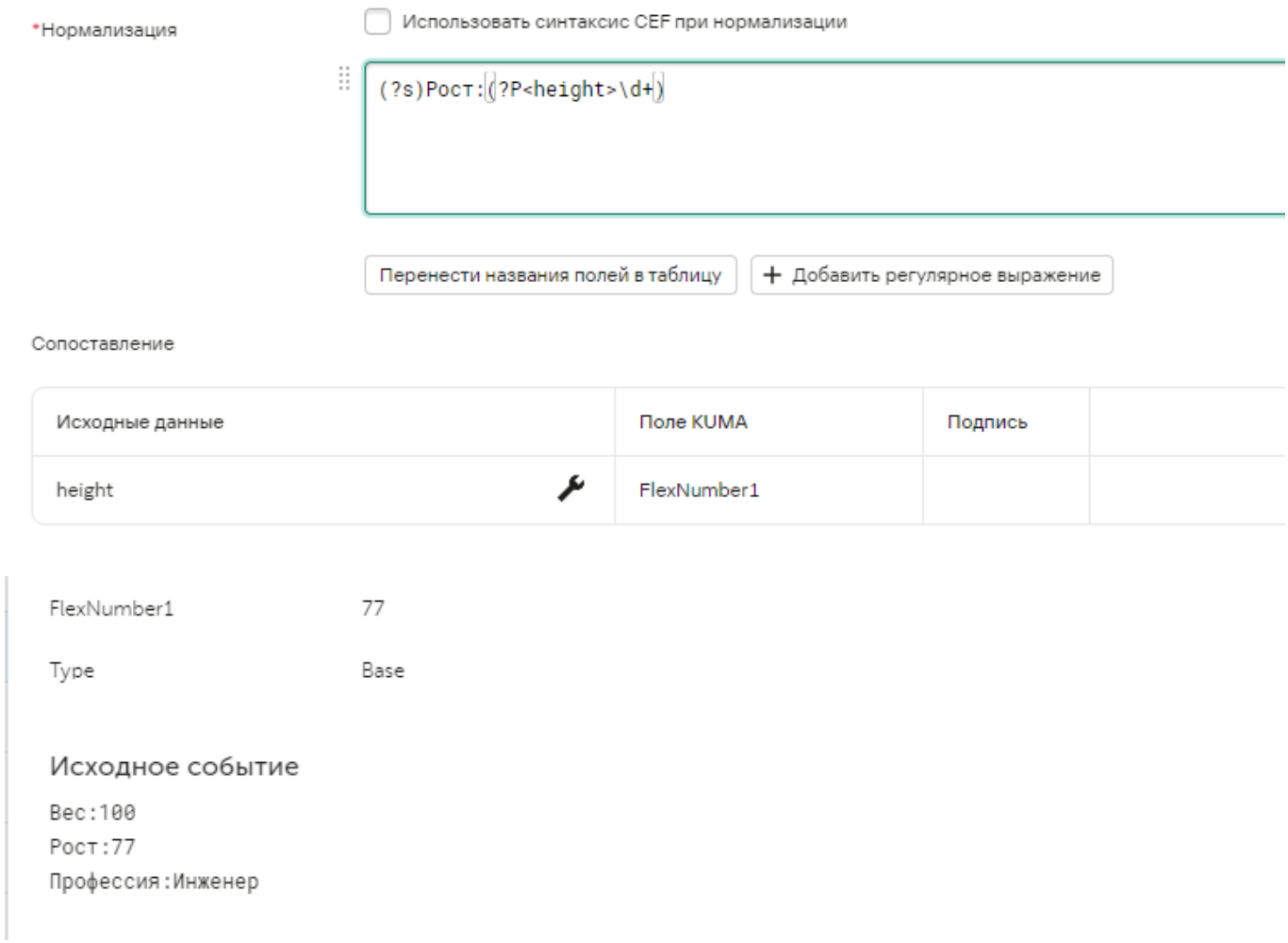
- `i` - ????????????? ? ???????? (?? ????????? false)
- `m` - ????????????? ?????, ^ ? $????????????? ?????? ??????/????? ? ?????????? ? ?????? ??????/????? (?? ????????? false)
- `s` - ????????? `.` (?????) ????????? ? `\n` (?? ????????? false)
- `U` - ?? ?????? ?????, ?????? ??????? ???????? x* ? x*?, x+ ? x+? ? ?. ?. (?? ????????? false)

????????? ?????: xyz (??????????) ??? -xyz (????????) ??? xy-z (?????????? xy, ???????? z).

????????????? ???? s



????????????? ? KUMA:



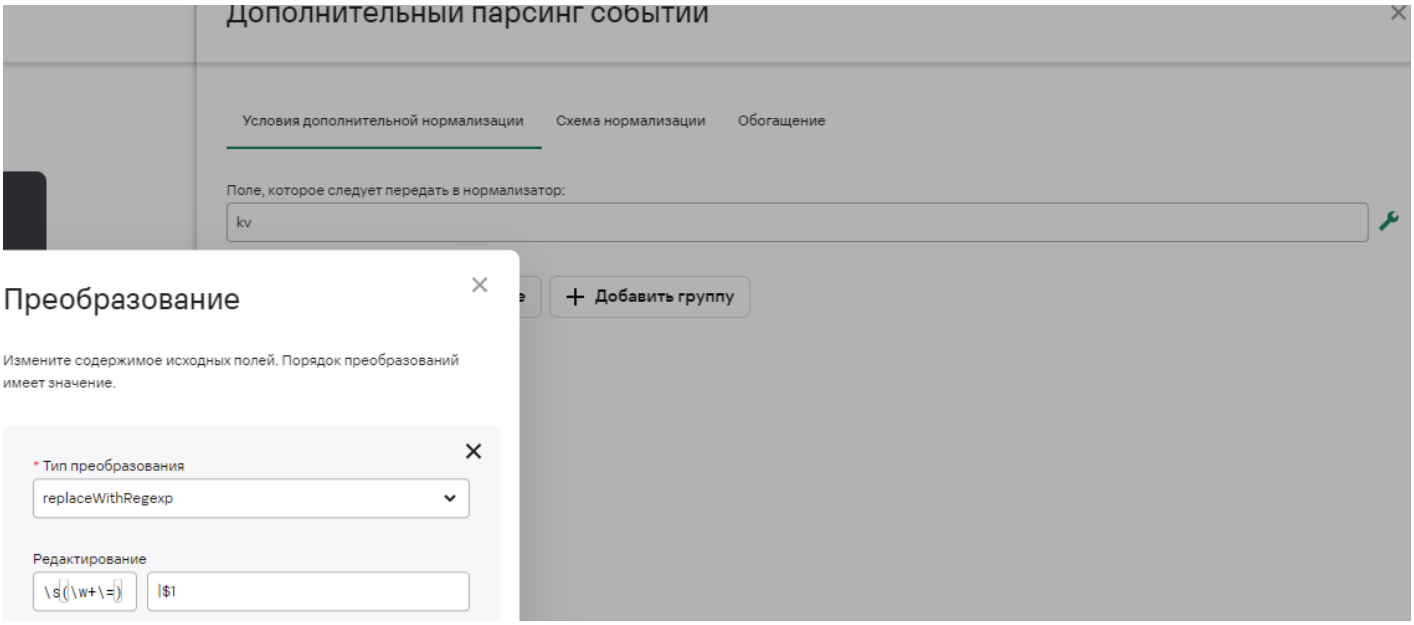
Замена разделителя в структуре KV

?????? ? ????????? KV ??????????? ?????? ??????? ??? ????????, ?????? ?????? ???????:

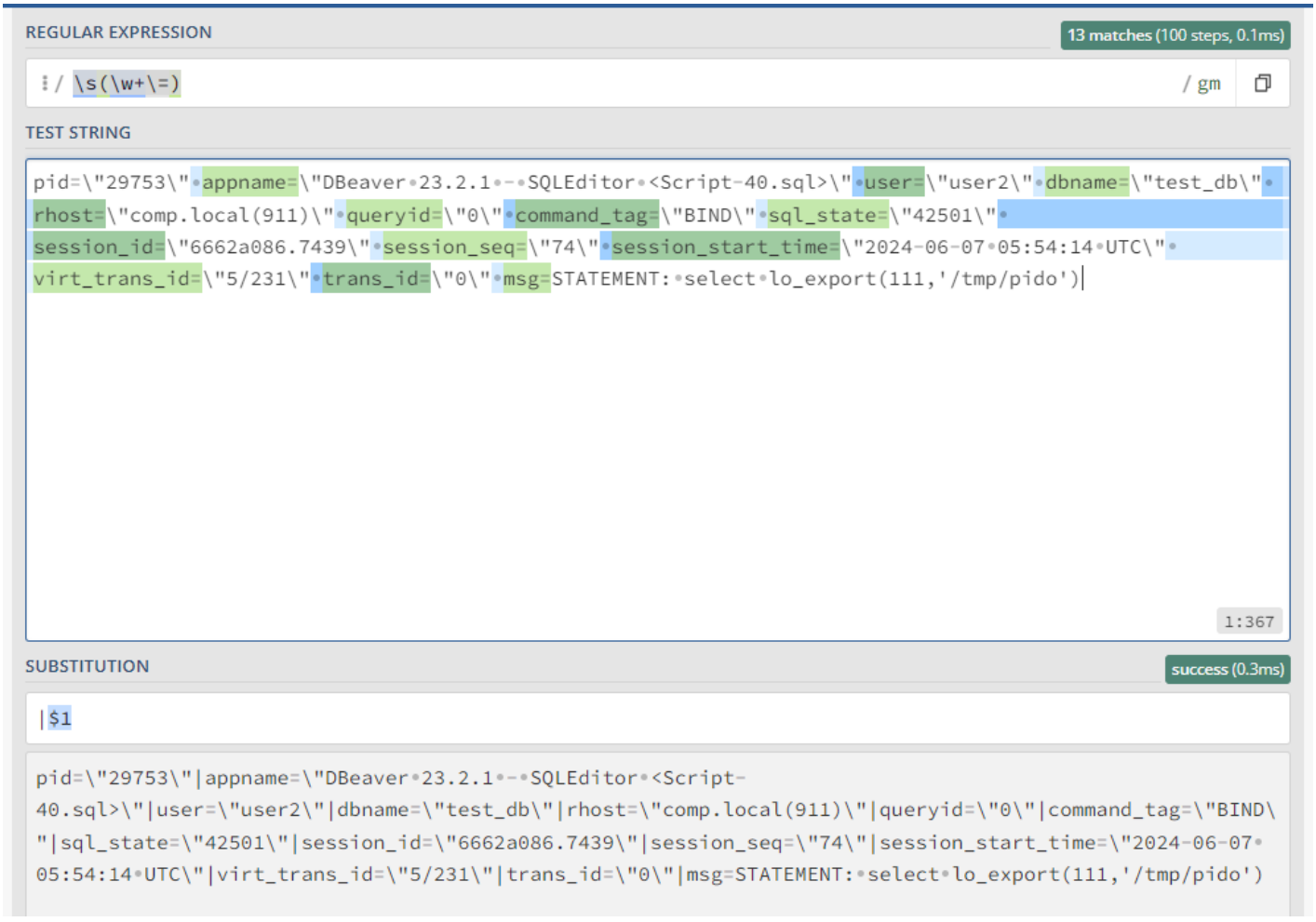
```
pid=\"29753\" appname=\"DBeaver 23.2.1 - SQLEditor <Script-40.sql>\" user=\"user2\" dbname=\"test_db\"
rhost=\"comp.local(911)\" queryid=\"0\" command_tag=\"BIND\" sql_state=\"42501\" session_id=\"6662a086.7439\"
session_seq=\"74\" session_start_time=\"2024-06-07 05:54:14 UTC\" virt_trans_id=\"5/231\" trans_id=\"0\"
msg=STATEMENT: select lo_export(111,'/tmp/pido')
```

? ??????? ???????? ?? ????? msg ??? ???????, ? ??? ????? ??????????? ?????????, ??????? ?????, ???? ???????? ???????? ??? ???????, ???? ?????????????? ??????? ??????? ? ??????? ???????????, ???????? |

??? ????? ?????????? ??????? replaceWithRegexp:


Дополнительный парсинг событий
Условия дополнительной нормализации
Схема нормализации
Обогащение
Поле, которое следует передать в нормализатор:
kv
Добавить группу
Преобразование
Измените содержимое исходных полей. Порядок преобразований имеет значение.
Тип преобразования
replaceWithRegexp
Редактирование
\s(\w+\=)
|$1

??? ??? ???????? ????? ??????:


REGULAR EXPRESSION
13 matches (100 steps, 0.1ms)
/ \s(\w+\=)
/ gm
TEST STRING
pid=\"29753\"•appname=\"DBeaver•23.2.1•-•SQLEditor•<Script-40.sql>\"•user=\"user2\"•dbname=\"test_db\"•rhost=\"comp.local(911)\"•queryid=\"0\"•command_tag=\"BIND\"•sql_state=\"42501\"•session_id=\"6662a086.7439\"•session_seq=\"74\"•session_start_time=\"2024-06-07•05:54:14•UTC\"•virt_trans_id=\"5/231\"•trans_id=\"0\"•msg=STATEMENT:•select•lo_export(111,'/tmp/pido')
1:367
SUBSTITUTION
success (0.3ms)
|$1
pid=\"29753\"|appname=\"DBeaver•23.2.1•-•SQLEditor•<Script-40.sql>\"|user=\"user2\"|dbname=\"test_db\"|rhost=\"comp.local(911)\"|queryid=\"0\"|command_tag=\"BIND\"|sql_state=\"42501\"|session_id=\"6662a086.7439\"|session_seq=\"74\"|session_start_time=\"2024-06-07•05:54:14•UTC\"|virt_trans_id=\"5/231\"|trans_id=\"0\"|msg=STATEMENT:•select•lo_export(111,'/tmp/pido')

??? ??? ??? ???????? ? KUMA:

Service	TEST BORIS (TCP/5577)
Type	Base
Extra	appname: \"DBeaver 23.2.1 - SQLEditor <Script-40.sql>\" command_tag: \"BIND\" dbname: \"test_db\" delim: • kv: pid=\"29753\"•appname=\"DBeaver 23.2.1 - SQLEditor <Script-40.sql>\"•user=\"user2\"•dbname=\"test_db\"•rhost=\"comp.local(911)\"•queryid=\"0\"•command_tag=\"BIND\"•sql_state=\"42501\"•session_id=\"6662a086.7439\"•session_seq=\"74\"•session_start_time=\"2024-06-07 05:54:14 UTC\"•virt_trans_id=\"5/231\"•trans_id=\"0\"•msg=STATEMENT: select lo_export(111,/tmp/pido) ms: .685 msg: STATEMENT: select lo_export(111,/tmp/pido) pid: \"29753\" queryid: \"0\" rfc5424_modif: 2024-06-07 05:54:14.685 rhost: \"comp.local(911)\" session_id: \"6662a086.7439\" session_seq: \"74\" session_start_time: \"2024-06-07 05:54:14 UTC\" sql_state: \"42501\" trans_id: \"0\" user: \"user2\" virt_trans_id: \"5/231\"

Общая шпаргалка

Anchors

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

Character Classes

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

POSIX

[:upper:]	Upper case letters
[:lower:]	Lower case letters
[:alpha:]	All letters
[:alnum:]	Digits and letters
[:digit:]	Digits
[:xdigit:]	Hexadecimal digits
[:punct:]	Punctuation
[:blank:]	Space and tab
[:space:]	Blank characters
[:cntrl:]	Control characters
[:graph:]	Printed characters
[:print:]	Printed characters and spaces
[:word:]	Digits, letters and underscore

Assertions

?=	Lookahead assertion
?!	Negative lookahead
?<=	Lookbehind assertion
?!= or ?<!	Negative lookbehind
?>	Once-only Subexpression
?()	Condition [if then]
?()|	Condition [if then else]
?#	Comment

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.

Escape Sequences

\	Escape following character
\Q	Begin literal sequence
\E	End literal sequence

"Escaping" is a way of treating characters which have a special meaning in regular expressions literally, rather than as special characters.

Common Metacharacters

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

The escape character is usually \

Special Characters

\n	New line
\r	Carriage return
\t	Tab
\v	Vertical tab
\f	Form feed
\xxx	Octal character xxx
\xhh	Hex character hh

Groups and Ranges

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

Ranges are inclusive.

Pattern Modifiers

g	Global match
i *	Case-insensitive
m *	Multiple lines
s *	Treat string as single line
x *	Allow comments and whitespace in pattern
e *	Evaluate replacement
U *	Ungreedy pattern

* PCRE modifier

String Replacement

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

Some regex implementations use \ instead of $.

Revision #13
Created 2023-10-03 13:16:14 UTC by Boris Rzr
Updated 2024-12-03 11:31:20 UTC by Boris Rzr