

Este material apresenta alguns exemplos de utilização da ferramenta REGEX - 101, a qual pode ser útil para alunos das disciplinas de Compiladores e LFAC. Ademais, ilustra brevemente outras ferramentas relacionadas com expressões regulares.

Sumário

1	REGEX - 101: principais características e funcionalidades	1
1.1	Exemplos Básicos	2
1.2	Exemplos para validação de Endereço	3
1.3	Exemplos para validação de Endereço com Repetição	4
1.4	Exemplos para validação de Endereço Composto	5
2	Outras ferramentas	7

1 REGEX - 101: principais características e funcionalidades

A ferramenta REGEX - 101 <https://regex101.com/> tem diversas características interessantes que podem facilitar o seu uso, bem como auxiliar no entendimento das expressões regulares.

A Figura 1 representa uma visão geral das principais funcionalidades da ferramenta.

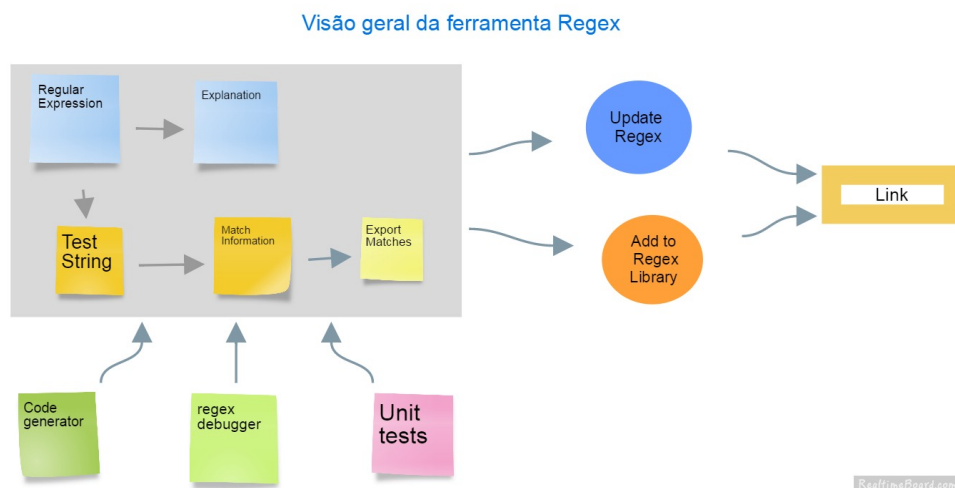


Figura 1: REGEX - visão geral

É possível enumerar algumas das principais características da ferramenta:

- *login* via Google ou GitHub.
- *syntax highlighting*.
- descrição e explicação detalha de cada expressão regular.

- *debugger* para entender todo caminho percorrido na verificação de uma dada entrada.
- lista de testes.
- geração de código em três linguagens (python, java script, php).
- modificadores específicos, como *i*, para indicar *case insensitive*, ao invés de *case sensitive*.
- biblioteca de expressões regulares, como diversos exemplo, entre eles:
 - Validação de URL.
 - Código posta britânico.
 - Endereço de e-mail.
 - Endereço de e-mail do Gmail.
 - Número de telefone.
 - Validação de senha.
 - Verificador de tags Html.
 - Verificador de Sql.
 - entre outros.
- Guia de referência rápida para consulta de operadores que podem ser utilizados:
 - Classe de caracteres.
 - Tokens em geral.
 - Quantificadores (operadores de concatenação).
 - Construtores de grupos.

1.1 Exemplos Básicos

As Figuras 2, 3, 4, 5, 6, 7 e 8 ilustram exemplos básicos de utilização da ferramenta. Onde considera-se que as expressões regulares envolvem apenas os símbolos *a*, *b*. Cada figura ilustra uma etapa apresentada (anteriormente) na Fig. 1.

Estes exemplos ainda podem ser acessados por meio dos seguintes links:

<http://regex101.com/r/bH81N0/3>

<http://regex101.com/r/bH81N0/4>

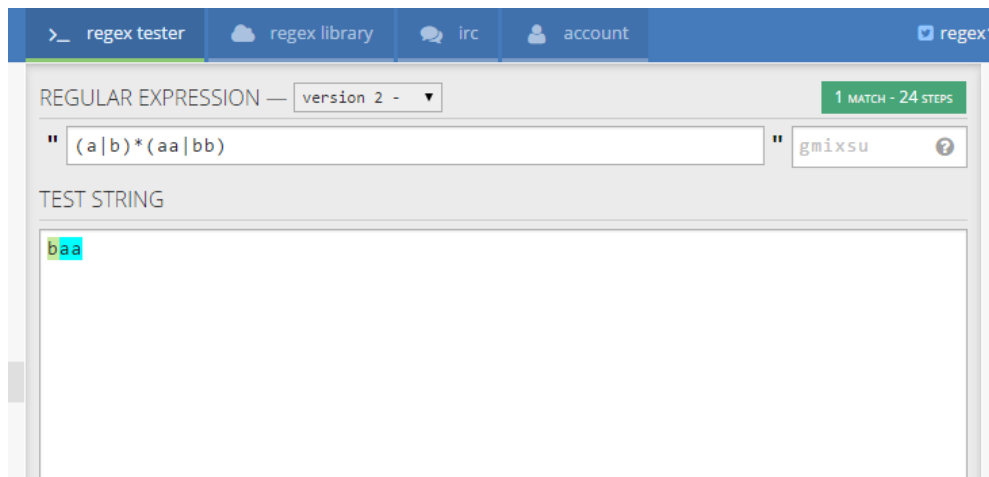


Figura 2: Exemplo Básico 1-A

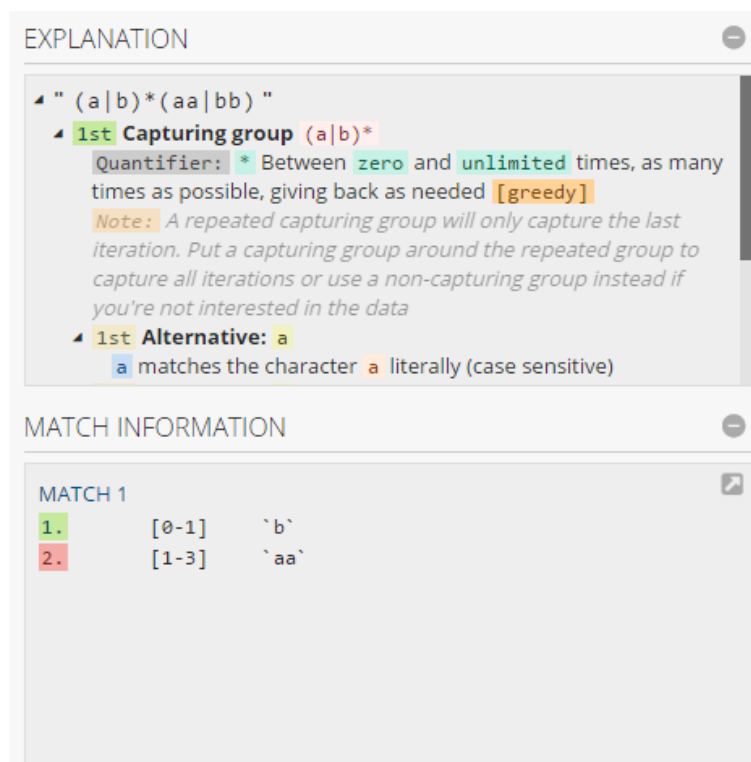


Figura 3: Exemplo Básico 1-B

1.2 Exemplos para validação de Endereço

As Figuras 9, 10 e 11 ilustram exemplos para validação de endereço (residencial ou similar) com diferentes formatos.

Este exemplo ainda podem ser acessados por meio do seguinte link:

<http://regex101.com/r/bH81N0/5>

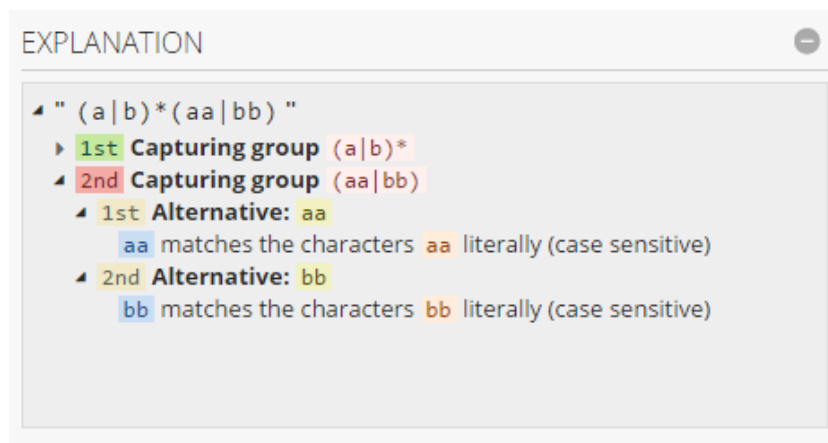


Figura 4: Exemplo Básico 1-C



Figura 5: Exemplo Básico 1-D

1.3 Exemplos para validação de Endereço com Repetição

As Figuras 13 e 14 ilustram exemplos para validação de endereço com repetição de informações.

Este exemplo ainda podem ser acessados por meio do seguinte link:

<https://regex101.com/r/bH81N0/6>

STATUS: DONE!

☒ DISPLAY POSITION IN PATTERN ☐ DISABLE INTER

MATCH 1 - FINISHED IN 25 STEPS

1	"(a b)*(aa bb)"	b	a	a
2	"(a b)*(aa bb)"	a	a	a
3	"(a b)*(aa bb)"	a	a	a
4	"(a b)*(aa bb)"	a	a	a
5	"(a b)*(aa bb)"	a	a	a
6	"(a b)*(aa bb)"	a	a	a
7	"(a b)*(aa bb)"	a	a	a
8	"(a b)*(aa bb)"	a	a	a
9	"(a b)*(aa bb)"	a	a	a
10	"(a b)*(aa bb)"	a	a	a
11	"(a b)*(aa bb)"	a	a	a
12	"(a b)*(aa bb)"	a	a	a
13	"(a b)*(aa bb)"	a	a	a
14	"(a b)*(aa bb)"	a	a	a
15	"(a b)*(aa bb)"	a	a	a
16	"(a b)*(aa bb)"	a	a	a
17	"(a b)*(aa bb)"	a	a	a
18	"(a b)*(aa bb)"	a	a	a
19	"(a b)*(aa bb)"	a	a	a
20	"(a b)*(aa bb)"	a	a	a
21	"(a b)*(aa bb)"	a	a	a
22	"(a b)*(aa bb)"	a	a	a
23	"(a b)*(aa bb)"	a	a	a
24	"(a b)*(aa bb)"	a	a	a
25	"(a b)*(aa bb)"	a	a	a
#	Match found in 25 step(s)			

Figura 6: Exemplo Básico 1-E

REGULAR EXPRESSION — version 2 - 1 MATCH - 24 STEPS

"(a|b)*(aa|bb)" "gmixsu

CREATE TEST

given the string test string

assert that capture group 1 equals string value

TEST LIST

given the string	abb	assert that	regex	matches	pass
given the string	ab	assert that	regex	does not match	pass
given the string	aba	assert that	regex	matches	fail
given the string	baacasinha	assert that	matched result	equals	aa
error: expected aa but got baa instead.					
given the string	baacasinha	assert that	matched result	contains	aa
given the string	olabb22	assert that	capture group 1	equals	a
given the string	olabb33	assert that	capture group 2	starts with	bb
given the string	olabaa44	assert that	capture group 1	equals	ab
error: expected ab but got b instead.					

Figura 7: Exemplo Básico 1-F

1.4 Exemplos para validação de Endereço Composto

REGULAR EXPRESSION — version 2 - 1 MATCH - 30 STEPS

CREATE TEST

given the string `test_string`

assert that `regex` matches

TEST LIST

- given the string `abb` assert that `regex` matches **pass**
- given the string `ab` assert that `regex` does not match **pass**
- given the string `aba` assert that `regex` matches **fail**
- given the string `baacasinha` assert that `matched result` equals `aa` **fail**
error: expected `aa` but got `baa` instead.
- given the string `baacasinha` assert that `matched result` contains `aa` **pass**
- given the string `olabb22` assert that `capture group 1` equals `a` **pass**
- given the string `olabb33` assert that `capture group 2` starts with `bb` **fail**
error: expected `bb` but got `a` instead.
- given the string `olabaa44` assert that `capture group 1` equals `ab` **pass**

EXPLANATION

1st Capturing group `((a|b)*(aa|bb))`

2nd Capturing group `(a|b)*`

3rd Capturing group `(aa|bb)`

1st Alternative: `aa`
`aa` matches the characters `aa` literally (case sensitive)

2nd Alternative: `bb`
`bb` matches the characters `bb` literally (case sensitive)

MATCH INFORMATION

MATCH 1

Index	Match	Start	End	Text
1	<code>[0-2]</code>	0	2	<code>'ab'</code>
2	<code>[1-2]</code>	1	2	<code>'b'</code>
3	<code>[2-4]</code>	2	4	<code>'aa'</code>

QUICK REFERENCE

CATEGORIES: General tokens

FLAGS/MODIFIERS: Global

Figura 8: Exemplo Básico 2

REGULAR EXPRESSION — version 2 - 1 MATCH - 12 STEPS

TEST STRING

`Rua:Monteiro,12`

EXPLANATION

1st Capturing group `(Rua|Avenida)`

1st Alternative: `Rua`
`Rua` matches the characters `Rua` literally (case sensitive)

2nd Alternative: `Avenida`
`Avenida` matches the characters `Avenida` literally (case sensitive)

`[A-Z]` match a single character present in the list below
`A-Z` a single character in the range between `A` and `Z` (case sensitive)

MATCH INFORMATION

MATCH 1

Index	Match	Start	End	Text
1	<code>[0-4]</code>	0	4	<code>'Rua-'</code>

Figura 9: Exemplo Validação de Endereço 3-A

As Figuras 15, 16, 17 ilustram exemplos para validação de endereço composto. Este exemplo ainda podem ser acessados por meio do seguinte link:
<https://regex101.com/r/bH81N0/7>

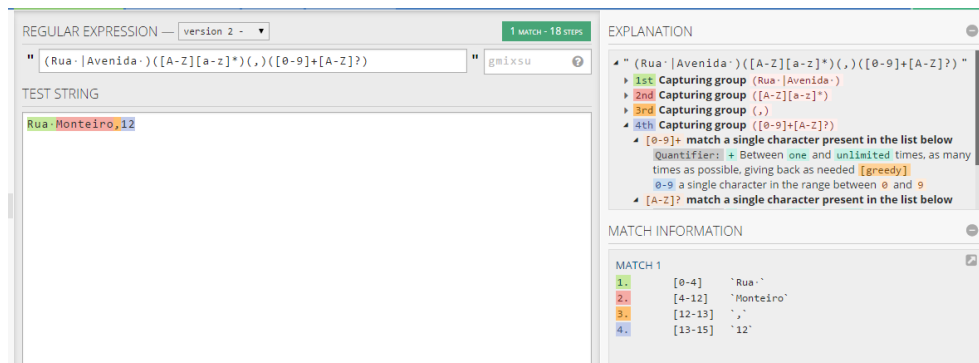


Figura 10: Exemplo Validação de Endereço 3-B



Figura 11: Exemplo Validação de Endereço 3-C

2 Outras ferramentas

Em relação a expressões regulares existem diversas ferramentas além da REGEX - 101.

- Aprenda RegEx - aprenda.vidageek.net/aprenda/regex

é um tutorial que auxilia no entendimento de expressões regulares por meio de exemplos.

- HiFi RegEx tool - <http://www.gethifi.com/tools/regex>

esta ferramenta é similar a REGEX - 101, a qual aceita uma expressão regular, um texto de entrada e gera o respectivo código em *JavaScript*.

- Debuggex - www.debuggex.com

ferramenta que gera diagramas de transição a partir de uma expressão regular. Com essa ferramenta é possível testar entradas.

A Figura 18 ilustra um exemplo de utilização desta ferramenta.

- Regexp - regxp.com

ferramenta que gera diagramas de transição a partir de uma expressão regular. Essa ferramenta não permite testar entradas. Porém, ela gera o diagrama em formato SVG.

A Figura 19 ilustra um exemplo de utilização desta ferramenta.

REGULAR EXPRESSION — version 2 — 1 MATCH - 18 STEPS

" (?P<tipo>Rua|Avenida|(?P<nome>[A-Z][a-z]*)(?P<delimitador>,(?P<numero>[0-9]+[A-Z]?))" gmixsu ?

CREATE TEST

given the string test_string

assert that capture_group 4 equals string value

TEST LIST 6/9

given the string	assert that	regex	matches	pass
Rua Futuro,78	regex	matches	pass	pass
Avenida Caxanga,120A	regex	matches	pass	pass
Avenida Vicente Machado,45B	regex	matches	fail	fail
CÂMPUS PONTA GROSSA Avenida Monteiro Lobato, s/n - Km 04 CEP 84016-21	matched result	contains	fail	fail
Avenida Caxanga,1230B	capture group 4	starts with	12	pass
Rua Osorio,127	capture group 2	starts with	Os	pass
Rua Osorio,127,Osorio,128	regex	matches	pass	pass
Rua Osorio,127,Osorio,128	capture group 4	equals	127	pass
Rua Osorio,127,Osorio,128	capture group 4	equals	128	fail

error: expected 128 but got 127 instead.

Figura 12: Exemplo Validação de Endereço 3-D

REGULAR EXPRESSION — version 5 — 1 MATCH - 25 STEPS

" (?P<tipo>Rua|Avenida|(?P<nome>[A-Z][a-z]*)(?P<delimitador>,(?P<numero>[0-9]+[A-Z]?)(?P<delimitador>,(?P<nome>[A-Z][a-z]*)(?P<delimitador>,(?P<numero>[0-9]+[A-Z]?))" gmixsu ?

TEST STRING

Rua Osorio,127,Osorio,128

EXPLANATION

- " (?P<tipo>Rua|Avenida|(?P<nome>[A-Z][a-z]*)(?P<delimitador>,(?P<numero>[0-9]+[A-Z]?)(?P<delimitador>,(?P<nome>[A-Z][a-z]*)(?P<delimitador>,(?P<numero>[0-9]+[A-Z]?))"
 - (?P<tipo>Rua|Avenida|) Named capturing group tipo
 - (?P<nome>[A-Z][a-z]*) Named capturing group nome
 - (?P<delimitador>,) Named capturing group delimitador
 - (?P<numero>[0-9]+[A-Z]*) Named capturing group numero
 - (?P<delimitador>) matches the same text as most recently matched by the capturing group named delimitador
 - (?P<nome>) matches the same text as most recently matched by the capturing group named nome
 - (?P<delimitador>) matches the same text as most recently matched by the capturing group named delimitador
 - (?P<numero>[0-9]+[A-Z]*) Named capturing group numero2

MATCH INFORMATION

MATCH 1	tipo	nome	delimitador	numero	numero2
[0-4]	Rua	[4-10]	Osorio	[10-11]	,
[11-14]	127	[22-25]	128		

Figura 13: Exemplo Validação de Endereço 4-A

REGULAR EXPRESSION — version 5 - 1 MATCH - 25 STEPS

" (?P<tipo>Rua·|Avenida·)(?P<nome>[A-Z][a-z]*) (?P<delimitador>·)(?P<numero>[0-9]+[A-Z]?)(?P=delimitador)(?P=nome)(?P=delimitador)(?P<numero2>[0-9]+[A-Z]?)" gmixsu ?

CREATE TEST

given the string test string

assert that capture group 5 equals string value

TEST LIST 3/7

given the string	Avenida Vicente Machado,45B	assert that	regex	matches	fail
given the string	CÂMPUS PONTA GROSSA Avenida Monteiro Lobato, s/n - Km 04 CEP 84016-210 - Ponta Grossa - PR - Brasil	assert that	matched result	contains	Mon teiro Loba
given the string	Rua Osorio,127	assert that	capture group 2	starts with	Os
given the string	Rua Osorio,127,Osorio,128	assert that	regex	matches	pass
given the string	Rua Osorio,127,Osorio,128	assert that	capture group 4	equals	127
given the string	Rua Osorio,127,Osorio,128	assert that	capture group 4	equals	128
error: expected 128 but got 127 instead.					
given the string	Rua Osorio,127,Osorio,128	assert that	capture group 5	equals	128

Figura 14: Exemplo Validação de Endereço 4-B

REGULAR EXPRESSION — version 6 - 1 MATCH - 35 STEPS

" (?P<tipo>Rua\s|Avenida\s)(?P<nome1>[A-Z][a-z]*) (?P<nomeN>(\s[A-Z][a-z]*)*) (?P<delimitador>,\s)(?P<numero>[0-9]+[A-Z]?)" gmixsu ?

TEST STRING

Rua Joaquim De Paula Xavier, 335B

EXPLANATION

- " (?P<tipo>Rua\s|Avenida\s)(?P<nome1>[A-Z][a-z]*) (?P<nomeN>(\s[A-Z][a-z]*)*) (?P<delimitador>,\s)(?P<numero>[0-9]+[A-Z]?)"
 - > (?P<tipo>Rua\s|Avenida\s) Named capturing group tipo
 - > (?P<nome1>[A-Z][a-z]*) Named capturing group nome1
 - > (?P<nomeN>(\s[A-Z][a-z]*)*) Named capturing group nomeN
 - > (?P<delimitador>,\s) Named capturing group delimitador
 - > matches the character , literally
 - > \s match any white space character [\r\n\t\f]
 - > (?P<numero>[0-9]+[A-Z]? Named capturing group numero

MATCH INFORMATION

MATCH 1		
tipo	[0-4]	'Rua'
nome1	[4-11]	'Joaquim'
nomeN	[11-27]	'De Paula Xavier'
4.	[20-27]	'-Xavier'
delimitador	[27-29]	','
numero	[29-33]	'335B'

Figura 15: Exemplo Validação de Endereço Composto 5-A

The screenshot displays a regular expression validation interface. The **REGULAR EXPRESSION** field contains the pattern: `"(?P<tipo>Rua\s|Avenida\s)(?P<nome1>[A-Z][a-z]*)(?P<nomeN>\s[A-Z][a-z]*)(?P<delimitador>,\s)(?P<numero>[0-9]+[A-Z])?"`. The **TEST STRING** field contains the address: `Rua JOAQUIM de paula Xavier, 335b`. The **EXPLANATION** panel provides a detailed breakdown of the pattern, identifying named capturing groups for `tipo`, `nome1`, `nomeN`, `delimitador`, and `numero`, along with their respective character classes and the `i` (insensitive) modifier. The **MATCH INFORMATION** panel shows the successful match for the entire string, with sub-matches for each component: `tipo` (0-4), `nome1` (4-11), `nomeN` (11-27), `delimitador` (27-29), and `numero` (29-33).

Figura 16: Exemplo Validação de Endereço Composto 5-B

This screenshot is identical to the one in Figure 16, showing the same regular expression validation tool interface. It displays the same pattern, test string, explanation, and match information, confirming the successful validation of the composite address "Rua JOAQUIM de paula Xavier, 335b".

Figura 17: Exemplo Validação de Endereço 5-C

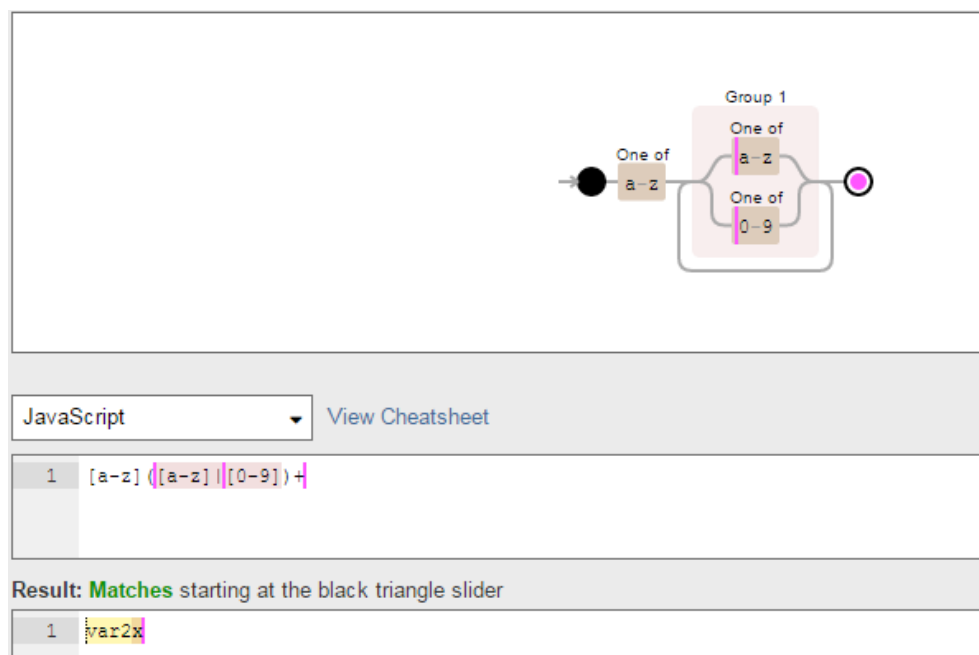


Figura 18: Exemplo da ferramenta Debuggex

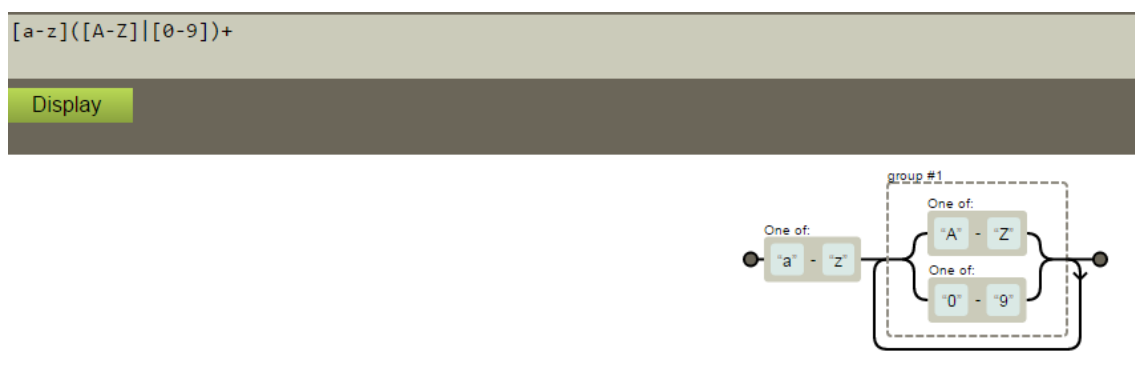


Figura 19: Exemplo da ferramenta Regexpal