

Expressões Regulares na Prática!

Gustavo Yudi Bentinezi Matsuzake



Universidade Tecnológica Federal do Paraná
Coders UTFPR

matsuzake@alunos.utfpr.edu.br
<https://github.com/yudi-matsuzake/regex-coders>

10 de Novembro de 2015

Sumário |

Introdução

O que é?

Parte prática!

Você não vai aprender isso nessa apresentação!

Você vai aprender muito isso!

Livro

Metacaracteres

Caracteres e Metacaracteres

Observações

Tipo Representante

Tipo Quantificador

Tipo Âncora

Outros

Sumário II

Exercícios

Aplicações

Referências

FIM

Introdução

O que é?

Parte prática!

Você não vai aprender isso nessa apresentação!

Você vai aprender muito isso!

Livro

Metacaracteres

Caracteres e Metacaracteres

Observações

Tipo Representante

Tipo Quantificador

Tipo Âncora

Outros

Exercícios

Aplicações

Referências

Definição

Expressão Regular é:

- ▶ Uma "linguagem de programação"[1];
- ▶ Intimamente relacionada aos **DFA**, que são *autômatos*, analisadores sintáticos[1];
- ▶ Uma maneira sucinta e **finita** de representar uma linguagem infinita - regular;
- ▶ É um método formal de se especificar um padrão de texto[2].

Mesmo, Eu.

"Uma imagem vale mais do que mil palavras. Uma Expressão Regular¹ vale infinitas."

¹bombeável

Observação importante:

Diferença da teoria e da prática.

Vamos deixar claro nosso escopo!

Para a felicidade de alguns, infelicidade de outros...

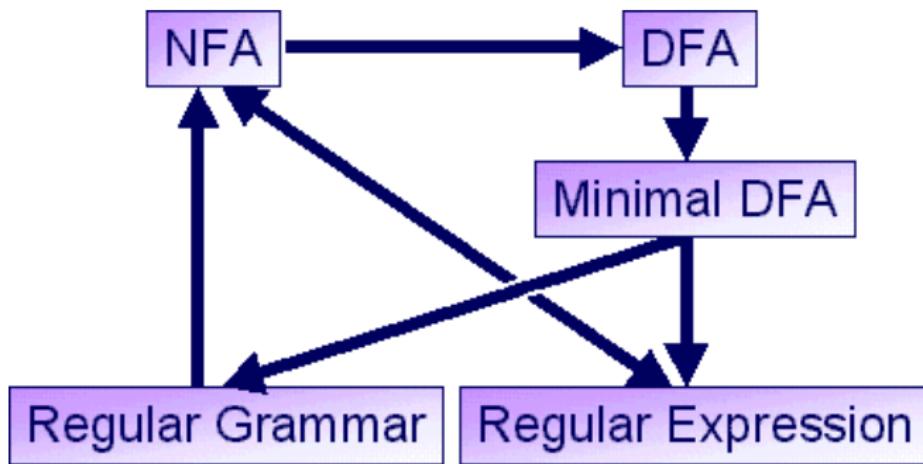
- ▶ Nosso escopo nessa apresentação é mais **prático** e não teórico;
- ▶ Vamos aprender **como** usar;
- ▶ Vamos aprender **onde** usar;
- ▶ Vamos aprender **porque** usar.

Nada de lógica

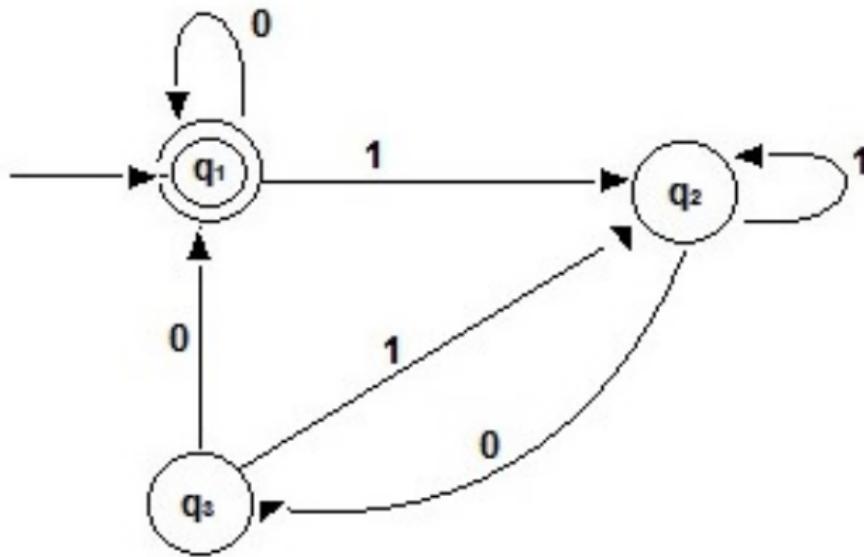
List of Identities:

1. $P \Leftrightarrow (P \vee P)$ ----- idempotence of \vee
2. $P \Leftrightarrow (P \wedge P)$ ----- idempotence of \wedge
3. $(P \vee Q) \Leftrightarrow (Q \vee P)$ ----- commutativity of \vee
4. $(P \wedge Q) \Leftrightarrow (Q \wedge P)$ ----- commutativity of \wedge
5. $[(P \vee Q) \vee R] \Leftrightarrow [P \vee (Q \vee R)]$ ----- associativity of \vee
6. $[(P \wedge Q) \wedge R] \Leftrightarrow [P \wedge (Q \wedge R)]$ ----- associativity of \wedge
7. $\neg(P \vee Q) \Leftrightarrow (\neg P \wedge \neg Q)$ ----- DeMorgan's Law
8. $\neg(P \wedge Q) \Leftrightarrow (\neg P \vee \neg Q)$ ----- DeMorgan's Law
9. $[P \wedge (Q \vee R)] \Leftrightarrow [(P \wedge Q) \vee (P \wedge R)]$ ----- distributivity of \wedge over \vee
10. $[P \vee (Q \wedge R)] \Leftrightarrow [(P \vee Q) \wedge (P \vee R)]$ ----- distributivity of \vee over \wedge
11. $(P \vee \text{True}) \Leftrightarrow \text{True}$
12. $(P \wedge \text{False}) \Leftrightarrow \text{False}$
13. $(P \vee \text{False}) \Leftrightarrow P$
14. $(P \wedge \text{True}) \Leftrightarrow P$
15. $(P \vee \neg P) \Leftrightarrow \text{True}$
16. $(P \wedge \neg P) \Leftrightarrow \text{False}$
17. $P \Leftrightarrow \neg(\neg P)$ ----- double negation
18. $(P \rightarrow Q) \Leftrightarrow (\neg P \vee Q)$ ----- implication
19. $(P \leftrightarrow Q) \Leftrightarrow [(P \rightarrow Q) \wedge (Q \rightarrow P)]$ ----- equivalence
20. $[(P \wedge Q) \rightarrow R] \Leftrightarrow [P \rightarrow (Q \rightarrow R)]$ ----- exportation
21. $[(P \rightarrow Q) \wedge (P \rightarrow \neg Q)] \Leftrightarrow \neg P$ ----- absurdity
22. $(P \rightarrow Q) \Leftrightarrow (\neg Q \rightarrow \neg P)$ ----- contrapositive

Nada de transição de DFA para NFA



Não vamos aprender autômatos



Isso vai fazer parte da sua realidade

`^ * [A-Za-z0-9_] +:(.*)$`

Isso não vai te dar mais pânico!

([0-9]{1,3}\.){2}[0-9]{1,3}-[0-9]{2}

Isso será fácil de entender (:

`[+-]?[0-9]{1,3}(\.[0-9]{3})?([0-9]{2})?`

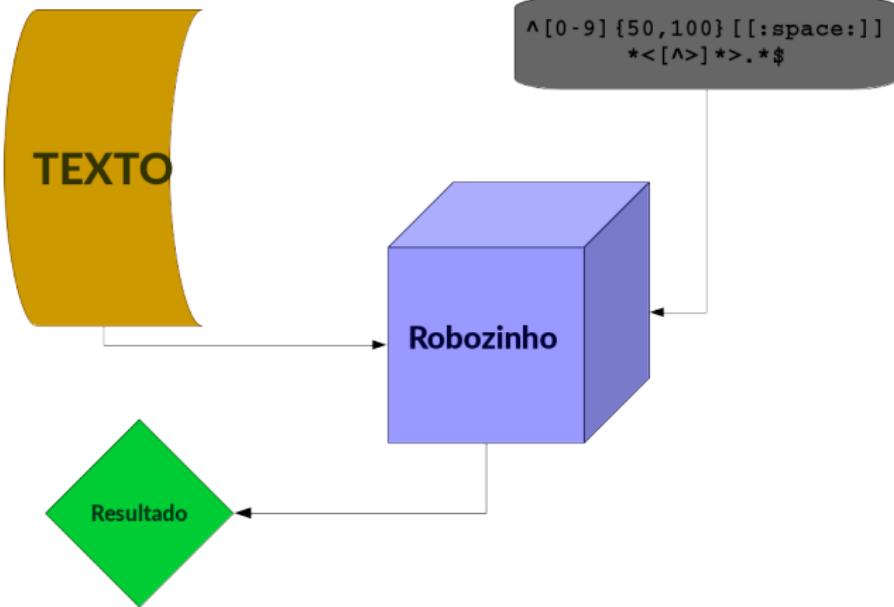


Figura: Como vamos abordar as RE

[Expressões Regulares] Uma abordagem divertida



Figura: Essa apresentação foi inspirada, causada e é sobre esse livro.

Introdução

O que é?

Parte prática!

Você não vai aprender isso nessa apresentação!

Você vai aprender muito isso!

Livro

Metacaracteres

Caracteres e Metacaracteres

Observações

Tipo Representante

Tipo Quantificador

Tipo Âncora

Outros

Exercícios

Aplicações

Referências

FIM

Você sabe o que são os metacaracteres?

- ▶ Se você não conhecia essas belezinhas de *expressões regulares* você só utilizou caracteres literais durante toda sua vida;
- ▶ Metacaracteres, você, você metacaracteres;
- ▶ Como isso vai fazer diferença na minha vida?
- ▶ Diferença dos robozinhos.



Figura: Expressões Regulares salvando o dia

Tipos

Representante, Quantificador e Âncora

Grupo

(. . .)

Escape

\n

Ponto



O ponto

- ▶ O ponto casa com **TUDO**.
- ▶ i.e., o ponto casa com qualquer caractere!
- ▶ **Pergunta:** o ponto casa com um ponto?

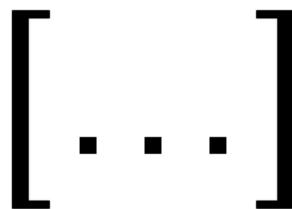


Figura: O ponto é o coringa solitário.

Casamento do ponto

Expressão	Casa com...
n.o	nao, não, n40, noo...
.u	au, bu, du...
12.45	12:45, 12 45, 12.45...

Lista



A lista

- ▶ A lista casa com **TODOS** os caracteres dentro dela.
- ▶ **Exemplos:** [abc], [123], [letras]...
- ▶ **Dentro da lista todo mundo é LITERAL.**
 - ▶ O ponto dentro da lista é literal;

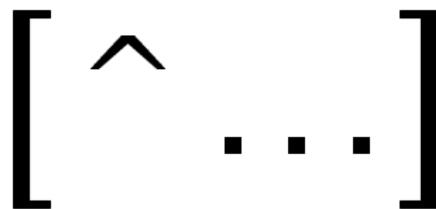
Mentiroso!

- ▶ Lembra que eu falei que todo mundo dentro lista era literal?
Então... **EU MENTI MUHAHAHA.**

Mentiroso (nem tanto, vai)!

- ▶ Lembra que eu falei que todo mundo dentro lista era literal?
Então...
 - ▶ Todo caractere que se acha fora da lista, lá dentro não serve pra nada ou tem outro significado;
 - ▶ É como se ela tivesse um mundinho dentro dela, só dela, onde ela dita as regras.

Listas Negadas



A lista negada

- ▶ A lista negada casa com **TODOS** os caracteres que **NÃO** estão dentro dela.
- ▶ Ou seja, ela não é tão seletiva quanto a lista, tampouco liberal quanto o ponto. Ela sabe com quem não quer casar;
- ▶ **Exemplos:** [^abc], [^123], [^letras]...

Intervalos

- ▶ Como você faria uma expressão regular para casar com dois números consecutivos?

Intervalos

- ▶ Como você faria uma expressão regular para casar com dois números consecutivos?
 - ▶ "[0123456789][0123456789]"?

Intervalos

- ▶ O traço (-) é um operador não literal dentro da lista;
- ▶ Ele foi criado para representar intervalos:
 - ▶ [0-9] é igual a [0123456789];
 - ▶ [a-z] é igual a [abcdefghijklmnopqrstuvwxyz];
 - ▶ [A-Z] é igual a [ABCDEFGHIJKLMNOPQRSTUVWXYZ];
 - ▶ [0-9a-zA-Z]
 - ▶ [-~]

Intervalos

- ▶ O traço (-) é o único operador não literal dentro da lista;
- ▶ Ele foi criado para representar intervalos:
 - ▶ [0-9] é igual a [0123456789];
 - ▶ [a-z] é igual a [abcdefghijklmnopqrstuvwxyz];
 - ▶ [A-Z] é igual a [ABCDEFGHIJKLMNOPQRSTUVWXYZ];
 - ▶ [0-9a-zA-Z];
- ▶ [-~] ???

Exceções

- ▶ E se eu quiser colocar um traço (-) na lista?
- ▶ E se eu quiser colocar os colchetes ([e]) na lista?
- ▶ E se eu quiser colocar um circunflexo na lista?
- ▶ E se...

Exceções

- ▶ **E se eu quiser colocar um traço (-) na lista?**
 - ▶ Devemos color o traço no começo ou no final da lista;
 - ▶ [-0-9] → Casa com 0 a 9 e traço;
 - ▶ [a-f-] → Casa com a a f e traço;
- ▶ **E se eu quiser colocar os colchetes ([e]) na lista?**
- ▶ **E se eu quiser colocar um circunflexo na lista?**
- ▶ **E se...**

Exceções

- ▶ E se eu quiser colocar um traço (-) na lista?
- ▶ E se eu quiser colocar os colchetes ([e]) na lista?
 - ▶ O "colchete abrir" podemos colocar em qualquer lugar:
 - ▶ [[0-9];
 - ▶ [0-4[*.<ç>];
 - ▶ O "colchete de fechar" deve ser o primeiro da lista;
 - ▶ []0-9] → Casa com 0 a 9 ou com].
 - ▶ [0-9]] → Tá errado.
- ▶ E se eu quiser colocar um circunflexo na lista?
- ▶ E se...

Exceções

- ▶ E se eu quiser colocar um traço (-) na lista?
- ▶ E se eu quiser colocar os colchetes ([e]) na lista?
- ▶ E se eu quiser colocar um circunflexo na lista?
 - ▶ [a-z^] → casa com caracteres de a a z e ^;
 - ▶ [^a-z] → lista negada: casa com qualquer coisa que não seja de a a z.
- ▶ E se...

Exceções

- ▶ E se eu quiser colocar um traço (-) na lista?
- ▶ E se eu quiser colocar os colchetes ([e]) na lista?
- ▶ E se eu quiser colocar um circunflexo na lista?
- ▶ E se...
 - ▶ Mais alguma coisa/dúvida?

Casamento das listas

Expressão	Casa com...
n[aÃAÃ]o	nao, nÃo, nAo, nÃo
12[:.]45	12:45, 12.45, 12 45
funcao.[ch]	funcao.c, funcao.h
[][-]	???

Casamento das listas

Expressão	Casa com...
n[aÃAÃ]o	nao, nÃo, nAo, nÃo
12[:.]45	12:45, 12.45, 12 45
funcao.[ch]	funcao.c, funcao.h
[][-]	[,], -

Opcional

?

O opcional

- ▶ O opcional torna um caractere ou metacaractere de representação anterior em opcional;
- ▶ Sua **cardinalidade** é 0 ou 1;

O casamento do opcional

Expressão	Casa com...
fala[r!]?	fala, falar, fala!
pessoas?	pessoa, pessoas
(hiper)?mercado	hipermercado, mercado

Asterisco



O asterisco

- ▶ O asterisco é o gulosso;
- ▶ A cardinalidade dele é 0 ou 1 até **infinito!**

O casamento do asterisco

Expressão	Casa com...
an^*a	, ana, annnnna, annnnnna...
$[ar]^*a$	arara, a, aararrarra...
10^*	1, 100, 1000000000...

Gulodice sem fim



Apresentando a gulodice infinita

- ▶ Esses dois operadores juntos são muito úteis;
- ▶ Com grandes poderes vêm grandes responsabilidades;
- ▶ Esses operadores juntos casam com qualquer coisa, em qualquer quantidade;

Máquina de estados

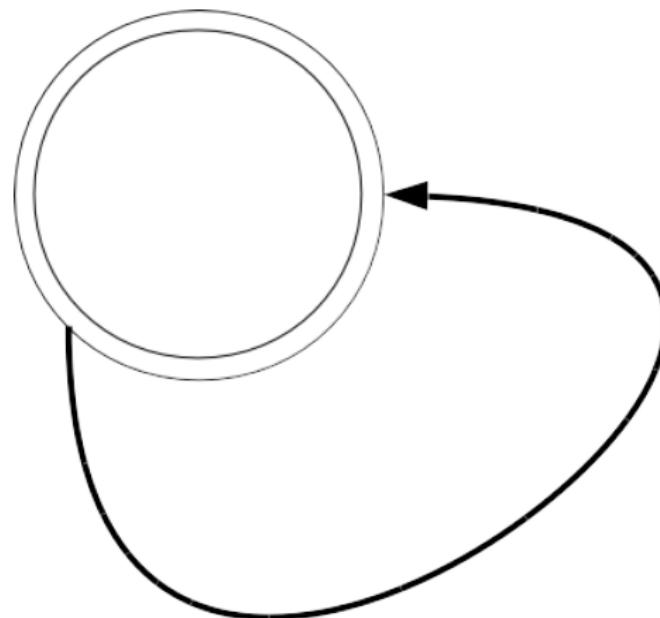
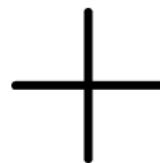


Figura: Máquina de estados da expressão $*$

Mais



O mais

- ▶ É igual ao asterisco só que não é opcional;
- ▶ **Tem** que ter o caractere que está sendo quantificado;
- ▶ A cardinalidade do operador mais é 1 até infinito.

Casamento do Mais

Expressão	Casa com...
an+a	ana, annna, annnnnnnna
cobras+	cobra, cobrasssssssss
10+	10, 100, 100

Chaves

{ m , n }

As Chaves

- ▶ As chaves tem um comportamento que consegue simular os operadores opcional(?), asterisco(*) e mais(+);
- ▶ Ele pode as seguintes "caras":
 1. **{n}**: repetir **n** vezes;
 2. **{n,m}**: repetir de **n** até **m**;
 3. **{n,}**: repetir de **n** até infinito;
 4. **,m}**: repetir até **m** vezes.
- ▶ **Pergunta:** Como o operador consegue simular os operadores ?, * e + ?

Respostas

- ▶ ?: {1}
- ▶ *: {0,}
- ▶ +: {1,}

Circunflexo

^

O Circunflexo

- ▶ O circunflexo casa com o começo da linha;
- ▶ Não confunda com o caractere da lista!
- ▶ O circunflexo só tem função de âncora no começo da linha, obviamente;
- ▶ i.e. o caractere circunflexo (^) vai ser literal no meio da linha.

O casamento do circunflexo

Expressão	Casa com...
$\^a$	Todas as linhas que começam com a
$\^{\^}$	Todas as linhas que começam com ^

Cifrão

\$

O cifrão

- ▶ O cifrão casa com o final da linha;
- ▶ O cifrão é literal no meio da linha;

O casamento do cífrão

Expressão	Casa com...
a\$	Uma linha que termina com a
^.40\$	Uma linha com 40 caracteres
^\$???

O casamento do cífrão

Expressão	Casa com...
a\$	Uma linha que termina com a
^.{40}\$	Uma linha com 40 caracteres
^\$	Uma linha vazia

Borda

\b

A borda

- ▶ Casa com a borda das palavras;
- ▶ Definição de borda: espaço depois de letras;

O casamento da borda

Expressão	Casa com...
\dia	dia, diafragma, bom-dia!
\dia	dia, diafragma, bom-dia!
dia \	dia, melodia, bom-dia!
\dia \	dia, bom-dia!

Ou



O ou

- ▶ É um operador lógico, casa com a ou b, onde a e b são expressões regulares;
- ▶ **Exemplo:** $a|b$;

Casa com...?

mal(a|inha|eta)

Casa com...?

bols(a|inha)

Casa com...?

pern(a|inha|eta)

Casa com...?

punh(o|inho)

Retrovisor

\ 1 . . . \ 9

O retrovisor

- ▶ O retrovisor "olha" para os grupos "(...)";
- ▶ O vão de \1 a \9;

O casamento da borda

Expressão	Casa com...
(quero)-\1	quero-quero
([A-Za-z]+)-\1	qualquerpalavra-qualquerpalavra

Introdução

O que é?

Parte prática!

Você não vai aprender isso nessa apresentação!

Você vai aprender muito isso!

Livro

Metacaracteres

Caracteres e Metacaracteres

Observações

Tipo Representante

Tipo Quantificador

Tipo Âncora

Outros

Exercícios

Aplicações

Referências

Exercícios

1. Baixe um arquivo html de algum site, e extraia com o comando egrep todos os parágrafos entre <p> <\p>
2. Em um código-fonte, encontre todas as funções que retornam inteiros
3. Abra o arquivo /var/log/kern.log[0-9]?² do seu computador e execute o egrep para achar TODOS os ipv6. i.e., são conjuntos de hexadecimais que vão de 0 a 9, depois de A a F, separados por :. Perceba que o ipv6 pode ser simplificado, zeros seguidos se transformam em ::.

²Isso é uma expressão regular :)

Microsoft

VBScript



PostgreSQL



`#!/bin/bash`







Referências

-  John Hopcroft, Jeffrey Ullman, and Rajeev Motwani.
Introduction to Automata Theory Languages and Computation.
Pearson Education, 2 edition, 2001.
-  Aurelio Marinho Jargas.
Expressões Regulares - Uma abordagem Divertida.
Novatec, www.aurelio.net, 4 edition, 2012.
Obrigado Aurelio por apresentar esse mundo bem louco e empolgante do software livre, por me fazer se apaixonar por shell script, vim e regex. SÉDimais (você é demais!).

Dúvidas?

Expressões Regulares na Prática!

Gustavo Yudi Bentinezi Matsuzake



Universidade Tecnológica Federal do Paraná
Coders UTFPR

matsuzake@alunos.utfpr.edu.br

<https://github.com/yudi-matsuzake/regex-coders>

10 de Novembro de 2015