

Chega de dores de cabeça para criar expressões regulares

# O Feiticeiro do Shell



Para os geeks, expressões regulares podem ser um interessante exercício mental, mas a selva de abreviações e símbolos arcanos pode ser aterrorizante para novatos. Entre no mundo do editor *txt2regex*. **POR ANDREAS KNEIB**

**T**entar encontrar a expressão regular certa para ter como retorno a cadeia de caracteres correta é uma experiência familiarmente frustrante para a maioria dos leitores. Assim também é o sentimento de resignação que nos acomete ao descobrir que as **regex** funcionam na linguagem Perl, mas não no código Lisp no qual, por um acaso, você vinha trabalhando.

Entremos no assistente do regex, o *txt2regex*, que exploraremos neste artigo. O assistente processa padrões de texto para diversos programas e linguagens, desde *awk*, passando por *Emacs*, até *Perl*, *procmail*, *sed* e *vim*. O programa compreende principalmente um script

em shell que exige o Bash versão 2.04 ou posterior.

Embora quase todas as distribuições Linux atuais possam satisfazer essa exigência, você pode querer fazer um teste para certificar-se de que está do lado do bem. Digite `echo $BASH_VERSION` ou `bash -version` para fazê-lo. Se o número de sua versão for grande o bastante, você pode continuar a instalação.

Se sua distribuição não incluir o assistente, vá à página de download do projeto em [1] para baixar um arquivo tar. Usuários do Debian podem simplesmente digitar `apt-get install txt2regex` para instalar a ferramenta.

Em seguida, descompacte o arquivo e vá para o diretório criado digitando:

```
~ > tar xvzf txt2regex-0.7.tgz
~ > cd txt2regex-0.7
```

Uma vez que não é necessário compilar o programa, você pode simplesmente digitar: `make install` como *root*:

```
~ > su
Password: senha
root# make install
```

Porém, há uma desvantagem nesse

método, pois ele guarda os componentes do programa nos diretórios `/usr/bin` e `/usr/share/locale`, onde nada de estranho à sua distribuição deve ser instalado. Para modificar os diretórios-alvo, passe as variáveis `BINDIR` e `LOCALEDIR` para `make install`. Ver Figura 1.

Isso coloca os componentes do programa em `/usr/local/bin` e `/usr/local/share/locale`. Há outro método possível, que é editar as variáveis no Makefile `txt2regex-0.7/Makefile`. A variável `MANDIR` não é usada na versão 0.7 do Makefile. Você pode copiar, por exemplo, `txt2regex-0.7/txt2regex.man` para `/usr/local/man/man1/`, para conseguir chamar as páginas de manual no futuro (`man txt2regex`):

```
root# cp txt2regex.man /usr/local/man/man1/txt2regex.1
```

Nosso programa já está pronto para rodar. Digite *txt2regex* num terminal para iniciá-lo.

## Opções

Primeiramente, vamos dar uma olhada nas opções de linha de comando da ferramenta. Há uma configuração básica para o fundo do console virtual em modo texto ou Xterm. A saída do assistente será mais facilmente legível se

## GLOSSÁRIO

**Regex:** Abreviação para expressão regular; uma combinação de caracteres especiais que os programas podem usar para buscar trechos específicos de texto.

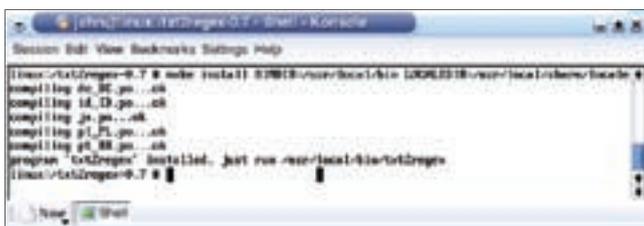


Figura 1: Passando alvos enquanto compila.

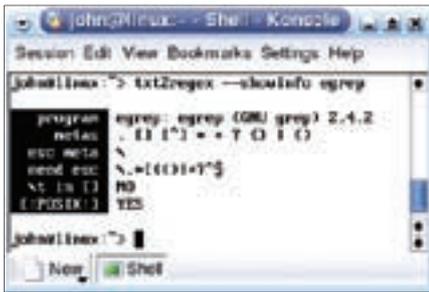


Figura 2: Exibição de meta-caracteres.

you opt for a light background; the parameter `--whitebg` takes care of that:

```
~ > txt2regex --whitebg
```

It's possible to remove the color of `txt2regex` by specifying `--nocolor`. The option `--prog` is also necessary to specify the programs for which the script will display the regex. The following example tells the tool to display the regular expressions for the editors `vim` and `emacs` and the Visual Basic Script and Perl:

```
~ > txt2regex --prog vim,emacs,vbscript,perl
```

To display all the possible variations of the regex, choose the option `--all`. Note that when you close the program, an error message appears, alerting you that the screen does not have enough lines, even if you are using a large Xterm. To avoid this, you should enter the command `export COLUMNS=LINES` in Bash before starting `txt2regex --all`.

Another important parameter is `--history`. It loads a previously defined expression:

```
~ > txt2regex --history 'http://www.linux-magazine.com'
```

With this, the text `http://www.linux-magazine.com` will be displayed in various programming languages. To complete an expression, the program displays a shortcut for the "history" option. Ad-

## INFORMAÇÕES

[1] Página oficial do Txt2regex:  
<http://txt2regex.sourceforge.net/>

[2] Aurélio Marinho Jargas, autor do programa:  
<http://www.aurelio.net>

dition the parameter `--showinfo` to get a list of the meta-characters of the program. See Figure 2.

The first line contains the name of the program, `egrep` in our case, followed by details about the meta-characters, the characters especially that need to be escaped or the Posix classes. For more information about `egrep`, type `man egrep`.

## Pergunta e Resposta

After starting the assistant (see Figure 3) by typing `txt2regex` in a console, you will notice the options at the top of the terminal: `quit`, `reset`, `color` and `programs`. The "end" key closes the editor, which displays a regex that can be accessed through the option `--history`. It also shows the text pattern specified by the regex, for example: *Start match at beginning of line, followed by a string...*

By pressing the key `0` to call the option `reset`, the program resets all the previous inputs. Pressing the asterisk key, `*`, turns the color on or off. Pressing the key `/` to access `Programs`, the most important menu of tools. It shows a general view of the programs whose regular expressions you want to use (see Figure 4).

Next comes a session of questions and answers, helping you to define your regular expression. The first set of data, *start to match*: with respect to the beginning of the search pattern. It can be *on the line beginning* (at the beginning of the line) or *in any part of the line* (in any part of the line). Respond by pressing `1` (for the beginning of the line) or `2` (for any part of the line). Your input will be converted to a regex language and appears in boxes of the application specified.

After finishing this selection, the

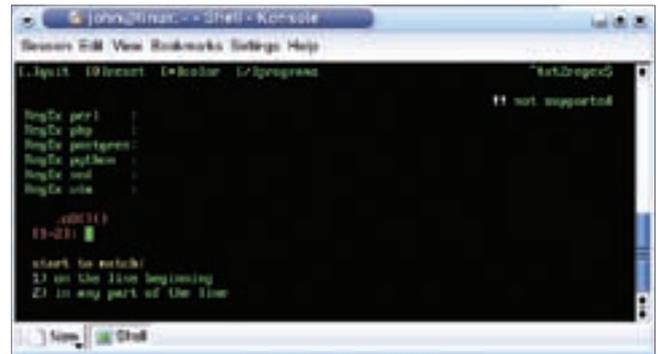


Figura 3: Txt2regex - aparência inicial.

group following is called *followed by*. It has a list of nine numbered items, ranging from *any character* (any character), passing through *forbidden characters list* (a list of prohibited characters), to *anything* (anything).

After configuring the character options, you have to answer *which* you want. The editor asks *how many times* the character should occur, offering seven options that range from *once* to *at most N times*.

After finishing this step, you return to the general view with the title *followed by*. Selecting an option *String*, you should enter the set of characters you want. Item `7`, *POSIX combination*, gives access to a list in alphabetical order that includes *letters* (letters), *numbers* (numbers), *hexadecimal numbers* (hexadecimal numbers) and *graphic chars* (graphic characters). The "end" key closes the menu Posix.

The assistant uses this method to compile a text pattern for you, without the need to worry about the specific aspects of the characters, special characters, escape or other things. After finishing to create your regex, just press the "end" key.

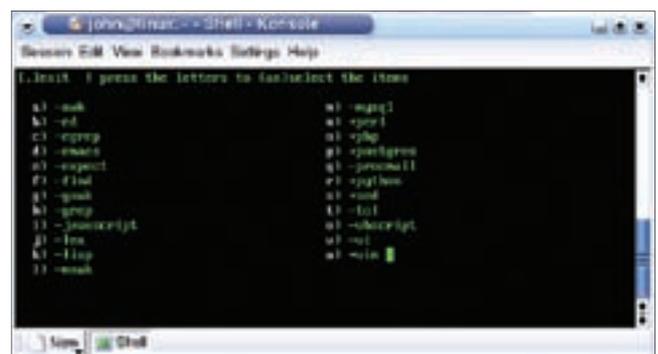


Figura 4: Seleção de linguagens de programação.