



# Expressões Regulares

As expressões regulares são estruturas formadas por uma sequência de caracteres que especificam um padrão formal que servem para validar, extrair ou mesmo substituir caracteres dentro de uma String

A screenshot of a Mac OS X desktop environment showing a terminal window and a code editor.

The terminal window is titled "regexp\_1.js — javascriptmasterclass". It contains the following text:

```
rodrigobranas:javascriptmasterclass $ node
> /john@gmail.com/;
/john@gmail.com/
> █
```

The code editor window is titled "JS regexp\_1.js". It contains the following code:

```
1 /john@gmail.com/;
```

regexp\_2.js — javascriptmasterclass

JS regexp\_2.js ×

```
1 new RegExp("john@gmail.com");
2
```

TERMINAL ⚙️ 1: node

```
rodrigobranas:javascriptmasterclass $ node
> new RegExp("john@gmail.com");
/john@gmail.com/
> █
```

JS regexp\_3.js ×

regexp\_3.js — javascriptmasterclass

1 let regExp = /john@gmail.com/;  
2 let result = regExp.test("john@gmail.com");  
3 console.log(result);  
4

TERMINAL ⚡ 1: bash

```
rodrigobranas:javascriptmasterclass $ node regexp/regExp_3.js
true
rodrigobranas:javascriptmasterclass $
```

The screenshot shows a terminal window with two panes. The left pane displays the code for 'regexp\_4.js' in a code editor. The right pane shows the terminal output.

**Code Editor (Left Pane):**

```
JS regexp_4.js ×
1 let regExp = /john@gmail.com/;
2 let result = regExp.exec("john@gmail.com");
3 console.log(result);
4
```

**Terminal (Right Pane):**

```
regexp_4.js — javascriptmasterclass
TERMINAL ... 1: bash
rodrigobranas:javascriptmasterclass $ node regexp/regExp_4.js
[ 'john@gmail.com', index: 0, input: 'john@gmail.com' ]
rodrigobranas:javascriptmasterclass $
```

A screenshot of a Mac OS X desktop environment showing a terminal window and a code editor.

The terminal window is titled "regexp\_5.js — javascriptmasterclass". It contains the following text:

```
rodrigobranas:javascriptmasterclass $ node regexp/regexp_5.js
john@gmail.com
8
E-mail: john@gmail.com
rodrigobranas:javascriptmasterclass $
```

The code editor window is titled "regexp\_5.js" and shows the following JavaScript code:

```
JS regexp_5.js ×
1 let regExp = /john@gmail.com/;
2 let result = regExp.exec("E-mail: john@gmail.com");
3 console.log(result[0]);
4 console.log(result.index);
5 console.log(result.input);
6
```

# Metacaracteres - Parte 1

. - Representa qualquer caractere

regexp\_6.js — javascriptmasterclass

JS regexp\_6.js ×

```
1 let regExp = /john@gmail.com/;
2 let result = regExp.exec("E-Mail: john@gmailxcom");
3 console.log(result[0]);
4 console.log(result.index);
5 console.log(result.input);
6
```

TERMINAL ... 1: bash

```
rodrigobranas:javascriptmasterclass $ node regexp/regexp_6.js
john@gmailxcom
8
E-Mail: john@gmailxcom
rodrigobranas:javascriptmasterclass $
```

# Escapando caracteres especiais

\ - A barra é utilizada antes de caracteres especiais, com o objetivo de escapá-los

A screenshot of a Mac OS X desktop environment showing a terminal window. The window title is "regexp\_7.js — javascriptmasterclass". The terminal tab bar shows "1: bash". The terminal pane displays the output of a Node.js script named "regexp\_7.js".

```
JS regexp_7.js x
1 let regExp = /john@gmail\.com/;
2 let result = regExp.exec("E-Mail: john@gmailxcom");
3 console.log(result);
4

TERMINAL ... 1: bash +
rodrigobranas:javascriptmasterclass $ node regexp/regexp_7.js
null
rodrigobranas:javascriptmasterclass $
```

# Iniciando e finalizando com um determinado caractere

^ - Inicia com um determinado caractere

\$ - Finaliza com um determinado caractere

regexp\_8.js — javascriptmasterclass

JS regexp\_8.js ×

```
1 let regExp = /^john@gmail\.com$/;
2 let result1 = regExp.exec("E-Mail: john@gmail.com");
3 console.log(result1);
4 let result2 = regExp.exec("john@gmail.com");
5 console.log(result2);
6
```

TERMINAL ... 1: bash

```
rodrigobranas:javascriptmasterclass $ node regexp/regexp_8.js
null
[ 'john@gmail.com', index: 0, input: 'john@gmail.com' ]
rodrigobranas:javascriptmasterclass $
```

# Grupos de caracteres

[abc] - Aceita qualquer caractere dentro do grupo, nesse caso a, b e c

[^abc] - Não aceita qualquer caractere dentro do grupo, nesse caso a, b ou c

[0-9] - Aceita qualquer caractere entre 0 e 9

[^0-9] - Não aceita qualquer caractere entre 0 e 9

# Quantificadores

Os quantificadores podem ser aplicados a caracteres, grupos, conjuntos ou metacaracteres.

{n} - Quantifica um número específico

{n,} - Quantifica um número mínimo

{n,m} - Quantifica um número mínimo e um número máximo

? - Zero ou um

\* - Zero ou mais

+ - Um ou mais

regexp\_9.js — javascriptmasterclass

JS regexp\_9.js ×

```
1 let regExp = /^[a-z]+@[a-z]+\.[a-z]{3}$/;
2 let result = regExp.exec("jane@hotmail.com");
3 console.log(result[0]);
4 console.log(result.index);
5 console.log(result.input);
6
```

TERMINAL ... 1: bash

```
rodrigobranas:javascriptmasterclass $ node regexp/regexp_9.js
jane@hotmail.com
0
jane@hotmail.com
rodrigobranas:javascriptmasterclass $
```

The screenshot shows a Mac OS X desktop environment with a terminal window open. The title bar of the terminal window reads "regexp\_10.js — javascriptmasterclass". The main pane of the terminal displays the output of running the script:

```
rodrigobranas:javascriptmasterclass $ node regexp/regexp_10.js
mary@hotmail.com.br
.br
0
mary@hotmail.com.br
rodrigobranas:javascriptmasterclass $
```

The terminal window has a standard OS X interface with red, yellow, and green close buttons at the top left. It includes a search icon, a refresh icon, and a menu icon at the top right. The bottom right corner of the window has a small black square icon.

The code in the script is as follows:

```
1 let regExp = /^[a-z]+@[a-z]+(\.[a-z]{2,3})+$/;
2 let result = regExp.exec("mary@hotmail.com.br");
3 console.log(result[0]);
4 console.log(result[1]);
5 console.log(result.index);
6 console.log(result.input);
7
```

# Metacaracteres

\w - Representa o conjunto [a-zA-Z0-9\_]

\W - Representa o conjunto [^a-zA-Z0-9\_]

\d - Representa o conjunto [0-9]

\D - Representa o conjunto [^0-9]

\s - Representa um espaço em branco

\S - Representa um não espaço em branco

\n - Representa uma quebra de linha

\t - Representa um tab

A screenshot of a Mac OS X desktop environment showing a terminal window. The window title is "regexp\_11.js — javascriptmasterclass". The terminal pane displays the output of running the JavaScript file "regexp\_11.js" with the command "node regexp/regexp\_11.js". The output shows the input string "mary@hotmail.com.br" being matched by the regular expression, with the result object containing the match, index, and input properties.

```
JS regexp_11.js ×
regexp_11.js — javascriptmasterclass
1  let regExp = /^w+@\w+(\.\w{2,3})+$/;
2  let result = regExp.exec("mary@hotmail.com.br");
3  console.log(result[0]);
4  console.log(result[1]);
5  console.log(result.index);
6  console.log(result.input);
7

TERMINAL  ...
1: bash
rodrigobranas:javascriptmasterclass $ node regexp/regexp_11.js
mary@hotmail.com.br
.br
0
mary@hotmail.com.br
rodrigobranas:javascriptmasterclass $
```

# Grupos de Captura

() - Determina um grupo de captura para realizar a extração de valores de uma determinada String

A screenshot of a Mac OS X desktop environment showing a terminal window. The window title is "regexp\_12.js — javascriptmasterclass". The terminal pane displays the output of running the JavaScript file "regexp\_12.js" with the command "node regexp/regexp\_12.js". The output shows the input string "mary@hotmail.com" being parsed by a regular expression to extract the local part ("mary") and the domain part ("hotmail.com"). The index of the match is 0, and the input string remains unchanged.

```
JS regexp_12.js × regexp_12.js — javascriptmasterclass TERMINAL 1: bash + = rodrigobranas:javascriptmasterclass $ node regexp/regexp_12.js mary@hotmail.com mary hotmail.com 0 mary@hotmail.com rodrigobranas:javascriptmasterclass $
```

```
1 let regExp = /([a-z]+)@([\.\w\w]+)/;
2 let result = regExp.exec("mary@hotmail.com");
3 console.log(result[0]);
4 console.log(result[1]);
5 console.log(result[2]);
6 console.log(result.index);
7 console.log(result.input);
8
```

# Modificadores

i - Case-insensitive matching

g - Global matching

m - Multiline matching

A screenshot of a Mac OS X desktop environment showing a terminal window. The window title is "regexp\_13.js — javascriptmasterclass". The terminal pane shows the command "node regexp/regexp\_13.js" followed by the output of the script. The script itself is visible in the main editor area.

JS regexp\_13.js ×

```
1 let regExp = /[a-z]+@[\.a-z]+/g;
2 let result1 = regExp.exec("mary@hotmail.com;john@gmail.com")
3 console.log(result1[0]);
4 console.log(result1.index);
5 let result2 = regExp.exec("mary@hotmail.com;john@gmail.com")
6 console.log(result2[0]);
7 console.log(result2.index);
8
```

TERMINAL ... 1: bash

```
rodrigobranas:javascriptmasterclass $ node regexp/regexp_13.js
mary@hotmail.com
0
john@gmail.com
17
rodrigobranas:javascriptmasterclass $
```