**Markdown** é uma ferramenta de conversão de *text-to-HTML*. Com ele é possível você marcar títulos, listas, tabelas, etc., de forma muito mais limpa, legível e precisa, do que se fosse fazer com HTML.

Onde usar Markdown

Existem vários lugares que você pode usar **Markdown**: No **Github** mesmo, você pode usar no *README.md*, que é um arquivo que fica na raiz do seu projeto, e é renderizado pelo **Github** abaixo da lista de arquivos. Aquele texto que você lê quando acessa um repositório é um arquivo *README.md*, escrito em **Markdown**.

Como eu começo a escrever em Markdown

Se você já tiver um *parser*, você só precisa criar um arquivo com uma dessas extensões: mark, markdown, md, mdml, mdown, text, mdtext, mdtxt, mdwn, mkd, mkdn.

Entre essas, a mais utilizada é a *md*. Crie um arquivo com uma dessas extensões e você já estará apto para começar a escrever em **Markdown**!

Agora, se você não tiver um *parser*, pode usar uma ferramenta online. Existem muitas ferramentas de *parse* de **Markdown** para **HTML**. Vou recomendar uma para que você possa acompanhar esse post, mas buscando por **Markdown Editor** no Google, você vai encontrar muitas outras!

Gosto bastante do [Stack Edit](https://stackedit.io/editor" \t "_blank). Ele converte **Markdown** para **HTML** enquanto você digita. É bastante simples para ver o resultado :)

Vamos aprender então como escrever em **Markdown**!

Tìtulos (<h1> a <h6>)

Para marcar um título, você vai usar # a quantidade de vezes que irá representar o nível do título. Exemplo:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 | # Título nível 1 ## Título nível 2 ### Título nível 3 #### Título nível 4 ##### Título nível 5 ###### Título nível 6 |

Parseia para:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 | <h1>Título nível 1</h1> <h2>Título nível 2</h2> <h3>Título nível 3</h3> <h4>Título nível 4</h4> <h5>Título nível 5</h5> <h6>Título nível 6</h6> |

Bastante simples, não? O h1 e o h2 ainda podem ser escritos da seguinte forma:

|  |  |
| --- | --- |
| 1 2 3 4 5 | Título de nível 1 ==================   Título de nível 2 ------------------ |

Parágrafos e quebras de linha (<p> e <br />)

Para gerar parágrafos, basta você escrever o texto em uma linha:

|  |  |
| --- | --- |
| 1 2 3 | Este é um parágrafo.   Este é outro parágrafo. |

Isso gera:

|  |  |
| --- | --- |
| 1 2 | <p>Este é um parágrafo.</p> <p>Este é outro parágrafo.</p> |

Note que eu pulei uma linha entre os parágrafos. Se eu não fizesse isso, o código gerado seria:

|  |  |
| --- | --- |
| 1 | <p>Este é um parágrafo. Este é outro parágrafo.</p> |

*Mas ele não deveria usar um para quebrar linha?*

Isso é muito particular de cada parser. Alguns quebram linha quando você dá enter. Mas a documentação do **Markdown** diz que, para quebras de linha, você precisa deixar dois espaços no final da linha:

|  |  |
| --- | --- |
| 1 2 | Primeira linha do parágrafo.·· Segunda linha do parágrafo. |

Coloquei o ·· no final da primeira linha somente para facilitar a visualização. Você deve substituir esse símbolo por dois espaços em branco. Isso deve gerar:

|  |  |
| --- | --- |
| 1 2 3 4 | <p>  Primeira linha do parágrafo.<br />  Segunda linha do parágrafo. </p> |

Por isso, se você estiver usando o [.editorconfig](http://editorconfig.org/) no seu projeto, deixe a opção trim\_trailing\_whitespace como false para arquivos **Markdown**. Assim, os espaços adicionais não serão removidos :)

Ênfase (<strong> e <em>)

Para enfatizar uma palavras (<em>), usamos um \* ou \_:

|  |  |
| --- | --- |
| 1 | Javascript é \_cool\_! |

ou:

|  |  |
| --- | --- |
| 1 | Javascript é \*cool\*! |

Que irá gerar:

|  |  |
| --- | --- |
| 1 2 3 | <p>  Javascript é <em>cool</em>! </p> |

O mais utilizado para ênfase (<em>) é o *underline*.

Para dar forte ênfase em palavras (<strong>), você usa dois \*\* ou \_\_:

|  |  |
| --- | --- |
| 1 | \*\*Da2k\*\* é a pronúncia para \*\*Daciuk\*\*: DA-TWO-K! |

ou

|  |  |
| --- | --- |
| 1 | \_\_Da2k\_\_ é a pronúncia para \_\_Daciuk\_\_: DA-TWO-K! |

Que irá gerar:

|  |  |
| --- | --- |
| 1 2 3 | <p>  <strong>Da2k</strong> é a pronúncia para <strong>Daciuk:</strong> DA-TWO-K! </p> |

O mais utilizado para forte ênfase (<strong>) são dois asteriscos.

Links (<a>)

Para gerar links, você usa [](). Dentro dos colchetes você coloca o texto do link, e dentro dos parênteses, você coloca a URL:

|  |  |
| --- | --- |
| 1 | [Blog do Da2k](http://blog.da2k.com.br) |

Que irá gerar:

|  |  |
| --- | --- |
| 1 | <a href="http://blog.da2k.com.br">Blog do Da2k</a> |

Passando um texto após a URL, separando o link do texto por um espaço em branco, esse texto será usado como title:

|  |  |
| --- | --- |
| 1 | [Blog do Da2k](http://blog.da2k.com.br "Clique e acesse agora!") |

Vai gerar:

|  |  |
| --- | --- |
| 1 | <a href="http://blog.da2k.com.br" title="Clique e acesse agora!">Blog do Da2k</a> |

Links automáticos

Se o texto do seu link é o próprio link, você pode envolvê-lo entre < e >, que o link será gerado automaticamente:

|  |  |
| --- | --- |
| 1 | <https://www.google.com.br> |

Irá gerar:

|  |  |
| --- | --- |
| 1 | <a href="https://www.google.com.br">https://www.google.com.br</a> |

E isso funciona também para e-mails:

|  |  |
| --- | --- |
| 1 | <meu@email.com> |

Vai gerar:

|  |  |
| --- | --- |
| 1 | <a href="mailto:meu@email.com">meu@email.com</a> |

Da hora, não? ;)

Blocos de citação (<blockquote>)

Para criar blocos de citação, você usa o sinal de >:

|  |  |
| --- | --- |
| 1 2 3 4 | > Esse é um bloco de citação. > Ele pode ter várias linhas por parágrafo. > > Inclusive, dando um espaço, você tem um novo parágrafo. |

Que gera o seguinte:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 | <blockquote>  <p>  Esse é um bloco de citação.  Ele pode ter várias linhas.  </p>  <p>Inclusive, uma quebra de linha.</p> </blockquote> |

Listas (<ul> e <ol>)

Para listas não ordenadas (<ul>), você pode usar \*, + ou -. Veja:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 | \* Item 1 \* Item 2 \* Item 3   + Item 1 + Item 2 + Item 3   - Item 1 - Item 2 - Item 3 |

Os três formatos acima geram a mesma marcação:

|  |  |
| --- | --- |
| 1 2 3 4 5 | <ul>  <li>Item 1</li>  <li>Item 2</li>  <li>Item 3</li> </ul> |

E para listas ordenadas, você usa o número, seguido de ponto:

|  |  |
| --- | --- |
| 1 2 3 | 1. Item 1 2. Item 2 3. Item 3 |

Que irá gerar:

|  |  |
| --- | --- |
| 1 2 3 4 5 | <ol>  <li>Item 1</li>  <li>Item 2</li>  <li>Item 3</li> </ol> |

Alguns parsers renderizam automaticamente os próximos números, após o 1. Você só precisa usar \* para os itens do 2 em diante:

|  |  |
| --- | --- |
| 1 2 3 | 1. Item 1 \* Item 2 \* Item 3 |

Mas não são todos que renderizam dessa forma, então é bom ficar ligado ;)

Imagens (<img />)

Geração de imagens é bem parecido com a geração de links: você só precisa adicionar uma ! no início. E o texto que você coloca entre os colchetes, é usado como alt na imagem:

|  |  |
| --- | --- |
| 1 | ![Banana](http://cdn.osxdaily.com/wp-content/uploads/2013/07/dancing-banana.gif) |

Esse código vai gerar:

|  |  |
| --- | --- |
| 1 | <img src="http://cdn.osxdaily.com/wp-content/uploads/2013/07/dancing-banana.gif" alt="Banana" /> |

O title também funciona como no link:

|  |  |
| --- | --- |
| 1 | ![Banana](http://cdn.osxdaily.com/wp-content/uploads/2013/07/dancing-banana.gif "Olha a banana dançando!") |

Que gera:

|  |  |
| --- | --- |
| 1 | <img src="http://cdn.osxdaily.com/wp-content/uploads/2013/07/dancing-banana.gif" alt="Banana" title="Olha a banana dançando!" /> |

Tabelas (<table>)

Código inline e bloco (<code> e <pre>)

Vocế ainda pode adicionar trechos de código via **Markdown**. Para adicionar código a nível *inline*, você usa ```:

|  |  |
| --- | --- |
| 1 | O `<blockquote>` é uma tag HTML. |

Isso irá gerar:

|  |  |
| --- | --- |
| 1 2 3 | <p>  O <code>&lt;blockquote&gt;</code> é uma tag HTML! </p> |

E para gerar blocos de código, você simplesmente indenta o código 4 espaços (ou 1 tab) à frente do paràgrafo:

|  |  |
| --- | --- |
| 1 2 3 4 | Essa é a função sayHello():  function sayHello() {  return 'hi!';  } |

Que irá gerar:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 | <p>  Essa é a função sayHello():  <pre><code>function sayHello() {  return 'hi!';  }</code></pre> </p> |

Isso é como está na documentação. Mas a maior parte dos parses que eu conheço não funcionam dessa forma. Eles geram blocos de código usando três crases no início da primeira e última linha, para marcar o início e o fim do bloco:

[](http://blog.da2k.com.br/uploads/2015/02/code-block.png)

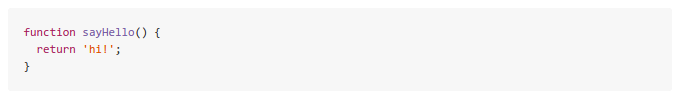
PS.: Tive que colocar como imagem, pois o meu parser não consegue escapar as 3 crases ¬¬

O **Github** inclusive [recomenda que se use as 3 crases](https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet#code), pois é mais fácil de visualizar e dar manutenção no código.

No **Github**, você ainda consegue definir qual a linguagem que está sendo utilizada, para que seja feito *code highlight* no seu código. Só passe a linguagem após as 3 crases, dessa forma:

[](http://blog.da2k.com.br/uploads/2015/02/code-block-language.png)

Que ó seu código será mostrado bonitinho assim:

[](http://blog.da2k.com.br/uploads/2015/02/code-block-highlighted.png)

Backslash scapes

Para escapar caracteres que são parseados pelo **Markdown**, você pode usar a barra invertida \ (*backslash*), seguida do caractere, para imprimí-lo literalmente. O escape funciona para os caracteres listados abaixo:

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 | \ backslash (barra invertida) ` backtick (crase) \* asterisk (asterisco) \_ underscore {} curly braces (chaves) [] square brackets (colchetes) () parentheses (parênteses) # hash mark (sustenido / hash / jogo da velha) + plus sign (sinal de "mais" ou somar) - minus sign (hyphen) (sinal de menos ou hífen) . dot (ponto) ! exclamation mark (ponto de exclamação) |