Markdown语法

原文地址 shukebeta翻译

概述

哲学

Markdown 的目标是易读易写。

Markdown强调可读性高于一切。一份Markdown格式的文档应该能直接以纯文本方式发布，而不致一眼看过去满眼都是标签和格式化指令。Markdown的语法确实受了几种现有的text转HTML过滤器影响 - - 包括 Setext, atx, Textile, reStructuredText,Grutatext, 和 EtText -- 其中对Markdown语法影响最大的单一来源是纯文本的Email格式。

为实现这一目标，Markdown的语法几乎全部由标点符号构成，这些标点符号都是精心挑选而来，尽量做到能望文生义。如星号括着一个单词（Markdown中表示强调）看上去就像 *强调*。
Markdown的列表看上去就像列表；Markdown的引文就象引文，和你使用email时的感觉一样。

内嵌HTML

Markdown的语法为“方便地在网上写作”这一目标而生。

Markdown不是HTML替代品，也不是为了终接HTML。它的语法非常简单，只相当于HTML标签的一个非常非常小的子集。它并非是为了更容易输入HTML标签而创造一种新语法。在我看来，HTML标签已经够容易书写的了。Markdown的目标是让（在网上）让读文章、写文章、修改文章更容易。HTML是一种适合发表的格式；而Markdown是一种书写格式。正因如此，Markdown的格式化语法仅需解决用纯文本表达的问题。

对Markdown语法无法支持的格式，你可以直接用HTML。你不需要事先声明或者使用什么定界符来告诉Markdown要写HTML了，你直接写就是了。

唯一的限制是那些块级HTML元素 -- 如 `<div>`，`<table>`，`<pre>`，`<p>`等等 -- 必须使用空行与相邻内容分开，并且块元素的开始和结束标签之前不要留有空格或TAB。Markdown足够聪明不会添加额外的(也是不必要的) `<p>` 标签包住这些块元素标签。

下面这个例子，在一篇Markdown文章中添加了一个HTML表格：

```
这是一个普通的段落。

<table>
    <tr>
        <td>Foo</td>
    </tr>
</table>

这是另一个普通的段落。
```

注意一点，不要在块级HTML元素内使用Markdown格式化命令，Markdown不会处理它们。比如你不要在一个HTML块中使用 `*emphasis*` 这样的Markdown格式化命令。

行内HTML标签 -- 如 `<span>`，`<cite>`，或 `<del>` -- 在一个Markdown段落里、列表中、或者标题中 - - 随便用。 如果需要，你甚至可以用HTML标签代替Markdown格式化命令。比方你可以直接用HTML标签 `<a>` 或 `<img>` 而不使用Markdown的链接和图片语法，随你的便。

不同于这些块级HTML元素，在HTML行内元素内的Markdown语法标记会被正确处理。

自动转换特殊字符

在HTML中，有两个字符需要特殊对待：`<`和`&`。`<`用于标签开始，`&`用于标识HTML实体。如果打算把它们当成普通字符，你必须使用反引号转义它们，如`<`和`&`。

对一些互联网作家来说，`&`符号特别使人烦恼。如果你打算写'AT&T'，你就得写 '`AT&T`'。甚至在URL中也得想着转义`&`符号。比方你打算写：

```
http://images.google.com/images?num=30&q=larry+bird
```

你就得在A标签中把`href`属性中的URL编码成：

```
http://images.google.com/images?num=30&amp;q=larry+bird
```

不用说，这很容易忘。这往往是那些良构HTML站点中最容易出错的地方。

在Markdown中，你尽管自然的使用这些字符，只需要关心那些必要的转义。如果使用在HTML实体中使用&符号，它会保持不变；而在其它场合，它会转换成`&`。

所以，如果你打算在文章中书写版权符号，你可以这样写：

```
&copy;
```

Markdown不会碰它。然而如果你书写

```
AT&T
```

Markdown就会把它翻译成：

```
AT&amp;T
```

类似的，既然Markdown支持内嵌HTML，如果你使用`<`作为HTML标签定界符，Markdown就会把它们当成HTML标签定界符。可是如果你书写：

```
4 < 5
```

Markdown就会把它翻译成：

```
4 &lt; 5
```

然而，在Mardown代码行内标记和块级标记之中，`<`和`&`始终会被自动编码。这使得在Markdown文件中书写HTML代码更容易.(相对于纯HTML。如果想在纯在纯HTML里贴一段HTML代码，那才是糟糕透顶，必须对代码中的每一个`<`和`&`都转义才成。)

块级元素

段落和换行

一个段落由一行或多个相关文本行构成。段落之间用一个或多个空行分隔。（一个空行就是一个看上去什么也没有的行 - - 如果一行什么也没有或者只有空格和TAB都会被视为空行）正常的段落不要以空白或TAB字符开始。

一行或多个相关文本行意味着Markdown支持“硬折行”。这一点与其它text转HTML的程序完全不同（包括Moveable Type的“Convert Line Breaks”选项），它们会将段落中的每一个换行符转换成`<br />`标签。

如果你确实需要使用Markdown插入一个`<br />`换行符，只需要在每一行的末尾以两个或更多个空格符号结束，然后再打回车键。

没错，在Markdown里生成一个`<br />`稍稍有一点麻烦，但那种简单的“把每一个换行符都转换成`<br />`规则”并不适用于Markdown。Markdown Email风格的 blockquoting 和 multi-

paragraph list items更好用 -- 并且更美观 -- 在你用换行符对其格式化时。

标题

Markdown 支持两种风格的标题，Setext 和 atx.

Setext-风格的一级标题下面一行使用等号符号，二级标题下面使用连字符符号，例如：

```
这是一个一级标题
==============

这是一个二级标题
--------------
```

至少有一个`=`和`-`就能正常工作。

Atx-风格的标题在每行的开头使用1 - 6个井号字符，分别对应标题级别1 - 6。例如：

```
# 这是一级标题

## 这是二级标题

###### 这是六级标题
```

如果愿意, 你也可以 "结束" atx-风格的标题。这纯粹是美观考虑--如果你觉得这样会看上更舒服些的话。结束用的井号个数随便，不必与起始井号数量相同 (起始井号的数量决定标题级别)：

```
# 这是一级标题 #

## 这是二级标题 ##
```

```
### 这是三级标题 ######
```

引用块

Markdown使用Email风格的 `>` 字符引用块。如果你熟悉Email中的引用块，你就知道在Markdown中如何使用引用块。如果每一行你都使用硬换行并在行首放一个`>`符号，看上去会很美观：

```
> This is a blockquote with two paragraphs. Lorem ipsum dolor sit amet,
> consectetuer adipiscing elit. Aliquam hendrerit mi posuere lectus.
> Vestibulum enim wisi, viverra nec, fringilla in, laoreet vitae, risus.
>
> Donec sit amet nisl. Aliquam semper ipsum sit amet velit. Suspendisse
> id sem consectetuer libero luctus adipiscing.
```

（如果觉得每行写一个`>`太累，）Markdown允许你偷懒，你只需在硬换行段落的第一行之前放一个`>`号:

```
> This is a blockquote with two paragraphs. Lorem ipsum dolor sit amet,
consectetuer adipiscing elit. Aliquam hendrerit mi posuere lectus.
Vestibulum enim wisi, viverra nec, fringilla in, laoreet vitae, risus.

> Donec sit amet nisl. Aliquam semper ipsum sit amet velit. Suspendisse
id sem consectetuer libero luctus adipiscing.
```

只需要多加一个`>`，就得到嵌套的引用块(即引用块中的引用块):

```
> This is the first level of quoting.
>
> > This is nested blockquote.
>
> Back to the first level.
```

引用块中可包含其它Markdown元素，如标题、列表和代码块：

```
> ## This is a header.
>
> 1.   This is the first list item.
> 2.   This is the second list item.
>
> Here's some example code:
>
>     return shell_exec("echo $input | $markdown_script");
```

是个象样的文本编辑器都能实现Email风格的引用。比如在BBEdit里，你就可以选中一些文字之后从Text菜单里选择引用级别。

列表

Markdown 支持有序列表和无序列表

无序列表可使用星号、加号和连字符（这几个符号是等价的，你喜欢哪个就用哪个）作为列表标记：

```
*   Red
*   Green
*   Blue
```

等同于：

```
+   Red
+   Green
+   Blue
```

也等同于：

```
-   Red
-   Green
```

```
-   Blue
```

有序列表则使用数字加英文句点：

```
1.  Bird
2.  McHale
3.  Parish
```

有一点需要注意，你在列表中输入的标记数字并不会反映到Markdown输出的HTML之中。上面这个列表Markdown会输出为：

```
<ol>
<li>Bird</li>
<li>McHale</li>
<li>Parish</li>
</ol>
```

即使你写成下面这样：

```
1.  Bird
1.  McHale
1.  Parish
```

甚至这样：

```
3. Bird
1. McHale
8. Parish
```

都会得到一模一样（但正确的）输出。要点在于，如果你愿意，就在你的Markdown有序列表里顺序使用数字（这样源代码里的顺序和生成的顺序会一致），如果你希望省点儿事，你就不用费心（去手工编号）。

如果你打算偷懒，记住列表的第一行使用数字 1。以后Markdown或许会支持有序列表从任意数字开始（译者注：这儿和前面的例子有点矛盾，原文如此）。

列表标记通常从左边界开始，至多可以有三个空格的缩进。列表标记之后至少要跟一个空格或TAB。

为了让列表看起来美观，你可以使用TAB缩进列表项内容，使其整齐:

```
*   Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
    Aliquam hendrerit mi posuere lectus. Vestibulum enim wisi,
    viverra nec, fringilla in, laoreet vitae, risus.
*   Donec sit amet nisl. Aliquam semper ipsum sit amet velit.
    Suspendisse id sem consectetuer libero luctus adipiscing.
```

不过如果你很懒，下面这样也行：

```
*   Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
Aliquam hendrerit mi posuere lectus. Vestibulum enim wisi,
viverra nec, fringilla in, laoreet vitae, risus.
*   Donec sit amet nisl. Aliquam semper ipsum sit amet velit.
Suspendisse id sem consectetuer libero luctus adipiscing.
```

如果列表项之间用空行分隔，Markdown就会在HTML输出中使用`<p>`标签包裹列表项。比如：

```
*   Bird
*   Magic
```

生成的HTML如下：

```
<ul>
<li>Bird</li>
<li>Magic</li>
</ul>
```

而这个：

```
*   Bird

*   Magic
```

生成的HTML是这样：

```
<ul>
<li><p>Bird</p></li>
<li><p>Magic</p></li>
</ul>
```

列表项有可能由多个段落组成，列表项的每个后续段落必须缩进至少4个空格或者一个TAB：

```
1.  This is a list item with two paragraphs. Lorem ipsum dolor
    sit amet, consectetuer adipiscing elit. Aliquam hendrerit
    mi posuere lectus.

    Vestibulum enim wisi, viverra nec, fringilla in, laoreet
    vitae, risus. Donec sit amet nisl. Aliquam semper ipsum
    sit amet velit.

2.  Suspendisse id sem consectetuer libero luctus adipiscing.
```

像上面这样缩进后续段落的每一行看起来很美观（但稍有些麻烦），如果你比较懒（和我一样），没问题，Markdown支持你：

```
*   This is a list item with two paragraphs.

    This is the second paragraph in the list item. You're
only required to indent the first line. Lorem ipsum dolor
sit amet, consectetuer adipiscing elit.

*   Another item in the same list.
```

要在列表项中使用引用，引用定界符 `>` 需要缩进：

```
*   A list item with a blockquote:

    > This is a blockquote
    > inside a list item.
```

要在列表项中使用代码块，代码块需要缩进两次 -- 8个空格或者两个TAB：

```
*   A list item with a code block:

        <code goes here>
```

有时候不小心会触发一个有序列表，比方在写类似下面这样的东西时：

```
1986. What a great season.
```

换言之， 以数字 + 句点 + 空格 序列起始的行会触发有序列表。为避免此情况，要对句点符号进行转义：

```
1986\. What a great season.
```

代码块

我们经常在写有关编程或标记语言源代码时用到预格式化的代码块。不像格式化普通段落，代码块中的行会按字面进行解释。Markdown对代码块同时使用 `<pre>` 和 `<code>` 标签包裹：

在Markdown中要生成一个代码块，只需要在代码块内容的每一行缩进至少四个空格或者一个TAB。比如：

```
This is a normal paragraph:
```

```
    This is a code block.
```

Markdown会生成：

```
<p>This is a normal paragraph:</p>

<pre><code>This is a code block.
</code></pre>
```

Markdown会从生成的代码块中删除一级缩进 -- 4个空格或者1个TAB。看下面这个例子：

```
Here is an example of AppleScript:

    tell application "Foo"
        beep
    end tell
```

会得到：

```
<p>Here is an example of AppleScript:</p>

<pre><code>tell application "Foo"
    beep
end tell
</code></pre>
```

代码块在遇到没有缩进的一行，或者文件末尾时自动结束。

在代码块中，`&`符号和`<`、`>`会自动转换成HTML实体。因此在Markdown中包含HTML源代码只是小菜一碟 - - 粘贴进去，缩进一下。剩下的脏活累活Markdown自会处理。看下面这个例子：

```
    <div class="sample_footer">
        &copy; 2004 Foo Corporation
    </div>
```

Markdown会生成：

```
        © 2004 Foo Corporation
```

Markdown不会解析代码块中的Markdown标记。如代码块中的星号就是星号，失去了它原来的Markdown含义。这意味着你能够使用Markdown编写Markdown自己的语法教程。（就象这篇文章一样）。

水平线

如果在一行里只放三个或更多个连字符，或星号或下划线，你就会得到一个水平线标记(`<hr />`)。下面每一行都会得到一个水平线：

```
* * *

***

*****

- - -

---------------------------------------
```

行内元素

链接

Markdown 支持两种风格的链接： 行内链接 和 引用链接.

两种风格的链接，链接文本都放在中括号之内[square brackets]。

要生成一个行内链接，在链接文本之后紧跟用一对小括号。小括号里放链接地址和可选的的链接title。如果提供链接title的话，链接title要用引号包起来。例如：

```
这是一个 [an example](http://example.com/ "Title") 行内链接。

[这个链接](http://example.net/) 没有title属性。
```

Markdown会生成：

```
<p>This is <a href="http://example.com/" title="Title">
an example</a> inline link.</p>

<p><a href="http://example.net/">This link</a> has no
title attribute.</p>
```

如果你打算引用一个本地资源或者同一站点的资源，可以使用相对路径：

```
如果想进一步了解我，请参阅我的 [关于我](/about/) 页。
```

引用风格的链接，在链接文本之后紧跟又一对中括号。这对中括号里放的是该链接的标识符（可以理解为别名）：

```
这是一个引用型链接 [示例][id]。
```

如果你嫌弃两对中括号过于亲密，Markdown允许你在两对中括号之间放一个空格：

```
这是一个引用型链接 [示例] [id]。
```

然后，我们可以在文档的任意位置，像下面这样定义链接标识与链接的对应关系（一行一个链接）：

```
[id]: http://example.com/  "Optional Title Here"
```

即：

- 中括号内放链接标识符(行前可选缩进，至多不超过三个空格)；
- 之后紧跟一个冒号；
- 再后面是一个或多个空格（TAB也行）；
- 接下来是链接URL；
- 最后面是可选的用双引号或单引号或小括号括起来的链接title。

下面三种链接定义方式是等价的：

```
[foo]: http://example.com/  "Optional Title Here"
[foo]: http://example.com/  'Optional Title Here'
[foo]: http://example.com/  (Optional Title Here)
```

注意： Markdown.pl 1.0.1 版本有一个已知的bug，用单引号作为链接title的定界符会出问题。

至于链接URL，还支持使用一对可选的尖括号包裹起来：

```
[id]: <http://example.com/>  "Optional Title Here"
```

你也可以将链接的title属性放在下一行并使用额外的空格或TAB填充，这样较长的URL会比较美观：

```
[id]: http://example.com/longish/path/to/resource/here
    "Optional Title Here"
```

链妆定义仅供Markdown解析器使用。最终输出的HTML当中不会包含链接定义。

链接标识符可以由字母、数字、空格和标点符号组成 - - 不区分大小写。下面这两个链接：

```
[link text][a]
[link text][A]
```

是等价的。

隐式链接标识 允许我们省略链接标识，这时链接文本本身就是链接标识。在链接文本之后加一对空的中括号 - - 例如，使用"Google"文本链接到google.com站点，可以这样写：

```
[Google][]
```

然后这样定义它的链接：

```
[Google]: http://google.com/
```

链接名字有可能包含空格，不过没问题，这种情况照样正常工作：

```
Visit [Daring Fireball][] for more information.
```

然后这样定义这个链接：

```
[Daring Fireball]: http://daringfireball.net/
```

链接定义可放于Markdown文档的任意位置。我建议把它们就近放到最先使用它的段落之后。不过如果你更喜欢放到文档末尾，当成某种形式的尾注，随你的便。

下面是一些引用链接的例子：

```
I get 10 times more traffic from [Google] [11] than from
[Yahoo] [12] or [MSN] [13].

  [1]: http://google.com/        "Google"
  [2]: http://search.yahoo.com/  "Yahoo Search"
  [3]: http://search.msn.com/    "MSN Search"
```

换成隐式链接标识，也可以这么写：

```
I get 10 times more traffic from [Google][] than from
[Yahoo][] or [MSN][].

  [google]: http://google.com/        "Google"
  [yahoo]:  http://search.yahoo.com/  "Yahoo Search"
  [msn]:    http://search.msn.com/    "MSN Search"
```

上面两种写法最终得到HTML输出是一样的：

```
<p>I get 10 times more traffic from <a href="http://google.com/"
title="Google">Google</a> than from
<a href="http://search.yahoo.com/" title="Yahoo Search">Yahoo</a>
or <a href="http://search.msn.com/" title="MSN Search">MSN</a>.</p>
```

作为比较，下面这个段落使用Markdown的行内链接风格编写：

```
I get 10 times more traffic from [Google](http://google.com/ "Google")
than from [Yahoo](http://search.yahoo.com/ "Yahoo Search") or
[MSN](http://search.msn.com/ "MSN Search").
```

引用型链接的亮点并不在于它更容易书写，而在于引用型链接让你的文档可读性更好。看看上面的例子：使用引用型链接，段落本身仅81个字符；而使用行内链接的例子，是176个字符。最终输出的HTML则有234个字符。纯HTML中标记字符甚至超过了文本本身。

使用Markdown的引用型链接，源文档更接近于最终的浏览器输出效果。再加上Markdown允许将标记有关的元数据移到段落之外，你尽管添加链接，而不必担心打断文件的故事情节。

强调

Markdown使用星号(`*`)和下划线(`_`)作为表示强调。用一个 `*` 或 `_` 包裹的文本会使用 HTML `<em>` 标签包裹; 用两个 `*` 或 `_` 包裹的文本会使用HTML `<strong>` 标签包裹。如：

```
*single asterisks*

_single underscores_

**double asterisks**

__double underscores__
```

将会输出为：

```
<em>single asterisks</em>

<em>single underscores</em>

<strong>double asterisks</strong>

<strong>double underscores</strong>
```

你喜欢哪一种风格就用哪一种，唯一的限制就是起始字符与关闭字符必须一致。

强调符号可用于一个单词的一部分：

```
un*frigging*believable
```

不过如果你用空格包裹单独的 `*` 或 `_`，它们就失去了强调的含义，而成为字面上的星号或下划线。

如果不想让Markdown解释这两个元字符，就转义它：

```
\*this text is surrounded by literal asterisks\*
```

代码

要在行内表示部分代码，用反引号(`)包住它。与预格式代码块不同和，行内代码用于段落之内。例如：

```
Use the `printf()` function.
```

会生成：

```
<p>Use the <code>printf()</code> function.</p>
```

要在一个行内代码中使用反引号（`）本身，用多个反引号作为定界符包住它：

```
``There is a literal backtick (`) here.``
```

这样就会得到：

```
<p><code>There is a literal backtick (`) here.</code></p>
```

包住行内代码的反引号定界符可以包括空格 - - 即在起始反引号之后，结束反引号之前可以有一个空格。这使得我们能够在行内代码的开始或结束处使用反引号：

```
A single backtick in a code span: `` ` ``

A backtick-delimited string in a code span: `` `foo` ``
```

会生成：

```
<p>A single backtick in a code span: <code>`</code></p>

<p>A backtick-delimited string in a code span: <code>`foo`</code></p>
```

在行内代码中，`&`和`<`和`>`会自动编码为HTML实体，以方便包含HTML标签。Markdown会把下面这行：

```
Please don't use any `<blink>` tags.
```

转换为：

```
<p>Please don't use any <code><blink></code> tags.</p>
```

你也可以这样写：

```
`&#8212;` is the decimal-encoded equivalent of `&mdash;`.
```

会得到：

```
<p><code>&#8212;</code> is the decimal-encoded
equivalent of <code>&mdash;</code>.</p>
```

图片

必须承认，要以“自然的”语法把一个图片放到一个纯文本文档之中，确实是一个挑战。

Markdown使用了类似链接语法来表示图片，同样有两种风格：行内图片和引用图片。

行内图片语法示例：

```
![Alt text](/path/to/img.jpg)

![Alt text](/path/to/img.jpg "Optional title")
```

即：

- 一个感叹号`!`开头；
- 其后紧跟一对中括号，中括号内存放图片的alt`属性；
- 其后紧跟一对小括号，小括号内存放图片的URL或路径，及可选的用双引号或单引号或小括号括起来的图片`title`

引用图片语法如下：

```
![Alt text][id]
```

这里 "id" 是图片引用标识。图片引用定义的语法与链接定义完全相同：

```
[id]: url/to/image  "Optional title attribute"
```

在写这篇文章时，Markdown还没有语法指定图片的大小，如果这一点对你特别重要，你可以直接使用`<img>`标签。

杂七杂八

自动链接

Markdown提供了一种快捷方式"自动地"定义链接和Email地址：直接用一对尖括号把URL或Email地址包住。这表示链接文本就是URL本身，Email文本就是Email本身。这样你就得到了一个可点击的链接，如：

```
<http://example.com/>
```

Markdown会将它转换为：

```
<a href="http://example.com/">http://example.com/</a>
```

自动Email地址工作方式相似，只有一点不同。Markdown自动的用一些十进制和十六进制数字表示你的Email，以防止遭遇垃圾邮件袭击。 例如：

```
<address@example.com>
```

会被转换为：

```
<a href="&#x6D;&#x61;i&#x6C;&#x74;&#x6F;:&#x61;&#x64;&#x64;&#x72;&#x65;
&#115;&#115;&#64;&#101;&#120;&#x61;&#109;&#x70;&#x6C;e&#x2E;&#99;&#111;
&#109;">&#x61;&#x64;&#x64;&#x72;&#x65;&#115;&#115;&#64;&#101;&#120;&#x61;
&#109;&#x70;&#x6C;e&#x2E;&#99;&#111;&#109;</a>
```

浏览器会将它渲染为一个可点击的链接，并正确显示 "address@example.com"。

(这种实体编码的小方法可以骗过一些收集邮件地址的机器人，不过它确实无法骗过所有的机器人。有总比没有强，聊胜于无。能阻止一点就阻止一点好了。)

反斜线转义

Markdown允许你使用反斜线转义那些Markdown元字符，让它们失去原有的“魔力”。举个例子，如果你确实想用星号包住一个词组（而不是想得到`<em>`标签），就可以在星号之前使用反斜线将其转义。即：

```
\*literal asterisks\*
```

Markdown中，以下字符支持使用反斜线转义：

```
\   反斜线
`   反引号
*   星号
_   下划线
{}  大括号
[]  中括号
```

()	小括号
#	井号
+	加号
-	减号（连字符）
.	句点
!	感叹号

markdown

shukebeta
发表于 2011-12-21 15:39

评论

时间 推荐

推荐
1

感谢shukebeta的辛勤翻译，收藏了。
个人认为这篇的译法更贴近我们的用语。“重译”是有价值的。


何逸勤
2011-12-21 22:49

推荐
1

之前markdown.tw网站也翻译过这篇：http://www.ituring.com.cn/article/504，您可以看看。


杨帆
2011-12-21 15:47

唉呀，事前没搜，重复劳动了。差不多四个小时呢。多谢老兄！ – shukebeta 2011-12-21 16:53

翻译了一遍，对于markdown的用法肯定印象深刻。 – 杨帆 2011-12-21 17:17

推荐
0

勘个误，在概述最后一段中“如果想在纯在纯HTML里贴一段HTML代码，那……” --> "去掉一个在纯"


十分之九
01-17 23:05

推荐
0

本文中小方块里包含红色的字是怎么实现的啊？


deephelper
2014-07-16 22:02

用半角反引号（`）把需要原文显示的内容包起来。比如 `hello,world` – shukebeta 2014-07-18 08:04

推荐

0

列表中写的链接不可用，怎么解决？

格瑞图图

2012-08-13 23:14

推荐

0

好详细！翻译不错。

fengmk2

2012-02-16 10:36

推荐

0

markdown里面比较麻烦的事情是绘制表格~~有什么好的解决方案么？

陈钢

2012-02-12 15:17

> php markdown extra 提供了一种增强型的markdown语法，支持用很简单快捷的方式画出表格。如果你用PHP,后端可以考虑用PHP markdown extra。http://michelf.com/projects/php-markdown/extra/ – shukebeta 2012-02-15 06:30

> 我是用来写一些简单的文档。为了方便输出，选用了markdown，但表格却比较麻烦。 – 陈钢 2012-02-20 09:03

推荐

0

对于俺这种不看HTML源码就难受的Web开发者而言，Markdown语法实在让人抓狂，嘿嘿

推荐阅读《偶得新玩具——通用文档转换器Pandoc》
（ http://www.ituring.com.cn/article/details/680 ）
及其它两篇与 pandoc 标签相关的文章。

高翌翔

2011-12-21 23:15

我要评论

需要登录后才能发言

记住我 登录

京ICP备11039595号 京公网安备11010502011375