Бесплатная электронная книга

УЧУСЬ Markdown

Free unaffiliated eBook created from
Stack Overflow contributors.

#markdown

Около

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: markdown

It is an unofficial and free Markdown ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official Markdown.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

глава 1: Начало работы с Markdown

Examples

Основное форматирование

Каждый тип формата связан с тегом HTML.

Первая рубрика относится к тегу `<h1>` и визуализируется как:

Привет, мир

и это написано подчеркиванием текста с `=` :

```
Hello World
===========
```

или путем добавления `#` к тексту:

```
# Hello World
```

Второй заголовок относится к тегу `<h2>` и визуализируется как:

Привет, мир

и это написано подчеркивание текста с `-` :

```
Hello World
-----------
```

или путем добавления `##` к тексту:

```
## Hello World
```

(Число `#` символов перед текстом соответствует номеру заголовка. `###` будет `<h3>` и т. Д.).

Код Inline ссылается на `<pre><code>` и визуализируется как:

```
Hello World
```

и это написано в тексте с помощью `` ` `` :

```
`Hello World`
```

Блоки кода относятся к тегу `<pre><code>` и визуализируются как:

```
Hello World
```

и это написано добавлением 4 (пробелы) перед каждой строкой:

```
    Hello World
```

Цитата относится к тегу `<blockquote>` и визуализируется как:

> Привет, мир

и он написан путем добавления a `>` перед каждым абзацем:

```
> Hello World
```

Акцент относится к тегу `<em>` и визуализируется как:

Привет, мир

и это написано sourroundind текст с `*`

```
*Hello World*
```

Полужирный относится к тегу `<strong>` и визуализируется как:

Привет, мир

и это написано путем окружения текста 2 `*`

```
 **Hello World**
```

Сильный акцент относится к тегу `<em><strong>` и он визуализируется как:

Привет, мир

и это написано путем окружения текста 3 `*`

```
***Hello World***
```

Прочитайте Начало работы с Markdown онлайн: https://riptutorial.com/ru/markdown/topic/529/начало-работы-с-markdown

глава 2: Диалекты / Ароматизаторы

замечания

Вариации Markdown с различным синтаксисом называются «ароматами». Markdown flavors перечислены на странице https://github.com/jgm/CommonMark/wiki/Markdown-Flavors .

Укус Markdown может быть реализован на различных языках программирования и программных приложениях. Библиотеки, реализующие анализатор Markdown, перечислены в https://github.com/markdown/markdown.github.com/wiki/Implementations и https://www.w3.org/community/markdown/wiki/MarkdownImplementations .

Несколько вкусов Markdown и различия между ними описаны на сайте http://flavoredmarkdown.com .

Чтобы протестировать и сравнить код Markdown во многих разных вариантах, а также реализации этих вкусов, вы можете использовать онлайн-инструмент Babelmark .

Examples

Устранение переполнения стека

Это аромат уценки, который используется Stack Overflow и другими сайтами обмена Stack. Когда вы отвечаете на вопрос или добавляете документацию, вы используете эту уценку. Этот ответ сделан из указателя SO

См. Официальную документацию

Главные вещи, которые добавляет SO markdown, находятся в разделе «Добавления стека Exchange» на этой веб-странице. В частности, SO добавляет *теги,* такие как `[tag:tag]` и ярлыки, такие как `[meta]` (не в документах), *спойлеры* :

> Это спойлер

```
>! This is a spoiler
```

и индивидуальные языковые преданности

```
<!-- language: java -->
```
This text is formatted as if it were Java code
```
```

```
This text is formatted as if it were Java code
```

Doxygen Markdown

Doxygen - это обычно используемый инструмент для документирования кода (для языков, включая C ++, C # и Java), который также поддерживает использование Markdown. Помимо стандартного синтаксиса Markdown, существует ряд элементов, специфичных для Doxygen .

Основными функциями являются использование тегов `@ref` для ссылок и `@page` , `@section/@subsection` и `@anchor` которые они могут ссылаться.

@ref <name> ["text"]

Этот элемент создает ссылку (то есть ссылку) на именованный раздел, подраздел, страницу или привязку, которые были определены в другом месте документации. (см. ссылку на Doxygen)

Первый параметр (`name`) должен соответствовать имени раздела, подраздела, страницы или привязки, к которым вы хотите привязать.

Второй необязательный параметр (`"text"`) должен быть заключен в двойные кавычки и будет определять, какая ссылка будет отображаться на странице. Если не используется, ссылка будет отображаться как заголовок, используемый в ссылке.

@section <имя-раздела> (название раздела)

Этот элемент определяет имя раздела. Он визуально эквивалентен элементу `#` в Markdown, однако он также определит ссылку, которая может быть связана с другими разделами вашей документации. (см. ссылку на Doxygen)

Первый параметр `section-name` определяет `section-name` ссылки, которое может использоваться элементом `@ref` . Это не может содержать пробелов.

Второй `section title` параметра - это строка слов (которая может быть разделена пробелами), которая определяет, как заголовок раздела появится на вашей странице.

пример

```
@section Intro Introduction

This is some text in my introduction.

@section Body Body Paragraph
```

```
This is some text in my body, where I refer to the @ref Intro.
```

Выход

Вступление

Это текст в моем вступительном слове.

Пункт органа

Это текст в моем теле, где я ссылаюсь на введение.

NB: Слово Введение выше появится как ссылка, которая перейдет к заголовку Введение.

GitHub Flavored Markdown

GitHub Flavored Markdown (иногда сокращенно до GFM) упрощает работу с уценкой на GitHub.com .

К основным функциям GFM относятся:

- отступ кода
- поддержка списка задач
- простое решение проблемы GitHub
- автоматическое имя пользователя GitHub и обнаружение SHA
- автоматическое обнаружение URL-адресов
- поддержка emoji

Примеры GFM

Подсветка синтаксиса

С Markdown блок кода может быть сгенерирован с тремя backticks:

```
    ```
```

Без подсветки синтаксиса код, написанный на C, выглядит следующим образом:

```
#include <stdio.h>

int main(void)
{
    printf("Hello World\n");
```
```

```
    return (0);
}
```

Однако ***c*** подсветкой синтаксиса код, написанный на C, выглядит следующим образом:

```c
#include <stdio.h>

int main(void)
{
    printf("Hello World\n");
    return (0);
}
```

Чтобы выделить код, просто добавьте необязательный идентификатор языка, чтобы включить подсветку синтаксиса в вашем защищенном блоке кода.

```
```C
<code text here>
```
```

Эти усовершенствования предназначены для улучшения качества документации и разговора, включенных в файлы `README` , gists , запросы на загрузку и проблемы на платформе.

Прочитайте Диалекты / Ароматизаторы онлайн: https://riptutorial.com/ru/markdown/topic/1865/диалекты---ароматизаторы

глава 3: Заголовки

Examples

Заголовки Atx-Style

Текст с префиксом от одного до шести фунтов (символы хеша, `#`) становится заголовком `<h1>` через `<h6>`, в зависимости от того, сколько значков фунта было использовано.

```
# This is a first-level (`<h1>`) header
## This is a second-level (`<h2>`) header
### And so on.
```

Это заголовок первого уровня (`<h1>`)

Это заголовок второго уровня (`<h2>`)

И так далее.

Заголовки в стиле Atx могут быть необязательно закрыты, добавляя завершающие значки фунта, которые игнорируются.

```
### This is a header with some trailing hashes ###
```

Это заголовок с некоторыми конечными хэшами

В настоящее время вкус уценки Stack Exchange, похоже, поддерживает до трех уровней заголовка (`###`), хотя Markdown поддерживает весь путь до 6 .

Заголовки Setext-Style

Чтобы создать заголовок первого уровня (`<h1>`), используйте знак равенства (`=`) в строке под текстом:

```
All About Dogs
==============
```

Все о собаках

Используйте дефисы (`-`) для заголовков второго уровня (`<h2>`):

```
The Debut Novel
---------------
```

Дебютный роман

Строка под заголовком может быть любой длины.

```
Another header
--

Another header
-
```

Другой заголовок

Другой заголовок

Прочитайте Заголовки онлайн: https://riptutorial.com/ru/markdown/topic/532/заголовки

глава 4: Изображений

Синтаксис

- ! [Alt текст] (/ путь / в / img.jpg)
- ! [Alt текст] (/ путь / to / img.jpg «Дополнительное название»)
- ! [Alt текст] [id]

  ... промежуточный контент ...

  [id]: /path/to/img.jpg «Необязательный заголовок»

замечания

Синтаксис для изображений такой же, как для ссылок , но с восклицательным знаком `!` перед текстом (который используется как альт-текст).

Examples

Встроенные изображения

При добавлении этого типа URL изображения включается в место, где будет отображаться изображение. Если вам нужно добавить одно и то же изображение несколько раз, вы должны каждый раз включать его URL.

Источник Markdown

```
Picture of Duck:

![Duck](http://i.stack.imgur.com/ukC2U.jpg)
```

Выход HTML

Изображение Duck:

Изображения ссылочного стиля

В этом типе добавления изображения одно изображение можно использовать несколько раз, не дублируя его URL, что делает его хорошим выбором при одновременном использовании одного изображения в документе.

Источник Markdown

```
Picture of Duck:

![Duck][1]

Same picture of Duck:

![Same Duck][1]

[1]: http://i.stack.imgur.com/ukC2U.jpg
```

Выход HTML

Изображение Duck:

Та же картина утки:

Прочитайте Изображений онлайн: https://riptutorial.com/ru/markdown/topic/698/изображений

глава 5: Код

Синтаксис

- *inline* : «Код между backticks» *или* <code> Код между этими HTML-тегами </ code>
- *multiline* : ____Код, который имеет отступы в каждой строке *или* <pre> <code> Несколько строк кода между этими тегами HTML </ code> </ pre>

Examples

Встроенный код

Markdown поддерживает добавление встроенного кода, полученного `like this` , путем обертывания текста в обратные ссылки:

```
`code here`
```

Кроме того, вы можете поместить свой встроенный код между тегами HTML `<code>` и `</code>` .

Рассмотрим следующий код уценки:

```
`This` is an inline code block! <code>This</code> is one too!
```

Это даст следующий результат:

> `This` встроенный блок кода! `This` тоже!

Если вам нужно включить обратную линию внутри встроенного кода, вы можете использовать несколько backticks для начала и завершения встроенного блока кода, например:

```
``code containing a backtick (`) character``
```

Это даст следующий результат:

> `code containing a backtick (`) character`

Используйте `\` чтобы избежать обратных шагов. Например:

```
\`a\`
```

будут представлены как

`a`

Выделение синтаксиса (StackExchange)

На сайтах StackExchange фрагменты кода могут предоставлять необязательную подсветку синтаксиса. На таких сайтах, как Stack Overflow, язык по умолчанию выводится из тегов, используемых в соответствующем вопросе (если применимо). Кроме того, язык подсветки синтаксиса кода фрагмента кода также может быть определен путем добавления комментария HTML к тексту .

```
<!-- language: lang-vb -->

    Sub ShowVB()
    Dim i As Long
    For i = 1 To 2
        If i = 3 Then
            MsgBox "How did that happen?"
        End If
    Next
    End Sub
```

Такой комментарий изменит язык подсветки синтаксиса для всех последующих фрагментов кода, что может быть весьма полезным, особенно если в одном сообщении задействовано несколько языков.

Вышеуказанное будет отображаться с подсветкой Visual Basic следующим образом:

```
Sub ShowVB()
Dim i As Long
For i = 1 To 2
    If i = 3 Then
        MsgBox "How did that happen?"
    End If
Next
End Sub
```

Отступные кодовые блоки

Вы можете создавать многострочные фрагменты кода, вставляя каждую строку с не менее чем четырьмя пробелами или одной вкладкой:

```
    #include <stdio.h>

    int main() {
      printf("Hello World!\n");
      return 0;
    }
```

Огороженные кодовые блоки

Некоторые парсеры позволяют назначать код, добавляя три обратных элемента до и после раздела кода.

```
```
This is an HTML example!
```
```

Необязательно, многие парсеры позволяют добавлять подсветку синтаксиса, указав язык кода сразу после первого набора обратных ссылок:

```
```html
<p><em>This</em> is an HTML example!</p>
```
```

Результат:

```
<p><em>This</em> is an HTML example!</p>
```

Блокированные блоки кода внутри списков

При добавлении отступов кодовых блоков внутри списка вам сначала нужна пустая строка, а затем дальнейший отступ кода. Различные вкусы Markdown имеют для этого разные правила.

1. StackExchange требует, чтобы код был отступом на 8 символов вместо обычного 4. *(Пространства заменены на * для ясности)* :

   ```
   1.*Listitem1
   2.*Listitem2

   ********code here
   3.*Listitem3
   ```

2. В таких спецификациях, как CommonMark, требуется, чтобы в блоке кода было отложено 4 символа, из которых начинается текст элемента списка. *(Для ясности пробелы заменены на *)* :

   ```
   1.****Listitem1
   2.****Listitem2

   **********code here
   3.****Listitem3
   ```

Прочитайте Код онлайн: https://riptutorial.com/ru/markdown/topic/553/код

глава 6: соединение

Examples

Встроенная ссылка

Форма для ссылки в уценке выглядит следующим образом.

```
[Shown Text](Link)
```

Например, это приведет вас к Example.com, который создается с помощью

```
[This will take you to Example.com](http://www.example.com)
```

Ссылка ссылки

URL-адреса ссылок могут быть указаны позже в документе.

уценка

```
[Text1][1] will link to the first link, and [Text2][2] to the second.
You [can reuse][1] names, and give longer names [like this one][a link].
You can also link text [like this] without giving the reference an explicit name.

[1]: http://www.google.com
[2]: http://stackoverflow.com/
[a link]: http://example.org/
[like this]: http://stackexchange.com/
```

Выход

> Text1 будет ссылаться на первую ссылку, а Text2 - на вторую. Вы можете повторно использовать имена и давать более длинные имена, подобные этому . Вы также можете связать текст таким образом, не указывая ссылку на явное имя.

Именованные якоря (ссылка на фрагмент страницы)

создать пункт назначения

```
<a id="destinationLinkName"></a>
```

ссылка на пункт назначения

```
[link text](#destinationLinkName)
```

Прочитайте соединение онлайн: https://riptutorial.com/ru/markdown/topic/545/соединение

глава 7: Создание списков

Examples

Нумерованные списки

```
1. Lists
2. Can be
3. Numbered
```

1. Списки
2. Может быть
3. нумерованный

Обратите внимание, что сами числа игнорируются:

```
1. This is the first item
5. This is the fifth item
7. This is the seventh item
```

1. Это первый элемент
2. Это пятый пункт
3. Это седьмой пункт

Однако первое число используется для начала нумерации:

```
3. This list starts at #3
2. However, this item is #4, despite being prefixed with `2.`
```

3. Этот список начинается с # 3
4. Однако этот элемент №4, несмотря на то, что он имеет префикс `2.`

Это можно использовать для возобновления списка после того, как он был прерван другим текстом / изображением / таблицей / и т. Д.

```
My very favorite colors are:
1. Blue
5. Red

(I like red because that's the best flavor of Skittle. But I digress.)

3. Orange
9. [etc]
```

Мои самые любимые цвета:

1. синий

2. красный

(Я люблю красный, потому что это лучший вкус Кэйтта, но я отвлекаюсь.)

3. оранжевый
4. [так далее]

Вы также можете аннотировать элемент списка таким образом, не прерывая нумерацию:

```
My very favorite colors are:
1. Blue
5. Red
(I like red because that's the best flavor of Skittle. But I digress.)
9. Orange
11. [etc]
```

Обратите внимание , что нет пустой строки между 5. и вводным заявлением, и **есть** два пробела в конце «Red», так что мы получаем:

Мои самые любимые цвета:

1. синий
2. красный
   (Я люблю красный, потому что это лучший вкус Кэйтта, но я отвлекаюсь.)
3. оранжевый
4. [так далее]

Без двух пространств этот раздел будет:

1. синий
2. Красный (я люблю красный, потому что это лучший вкус Кэйтта, но я отвлекаюсь).

... из-за того, как Markdown рассматривает разрывы строк.

Буклетные списки

```
Characters for bulleted lists:
* Asterisks
+ Plus signs
- Minus signs
```

Персонажи для маркированных списков:

- Звездочки
- Плюс знаки
- Минусы

Пожалуйста, обратите внимание:
Для достижения наилучших результатов вы должны использовать один и тот же символ, потому что, как вы можете видеть в приведенном ниже примере, разные знаки делают перерыв в списке

```
Characters for bulleted lists:
* Asterisks 1
* Asterisks 2
* Asterisks 3
+ Plus signs 1
+ Plus signs 2
+ Plus signs 3
- Minus signs 1
- Minus signs 2
- Minus signs 3
```

Персонажи для маркированных списков:

- Звездочки 1
- Звездочки 2
- Звездочки 3

- Плюс знаки 1
- Плюс знаки 2
- Плюс знаки 3

- Минусы 1
- Минусы 2
- Минусы 3

Вложенные списки

```
1. Lists can be nested
    * Four spaces
        - Eight spaces
            + Twelve spaces
2. And back
```

1. Списки могут быть вложенными
    - Четыре пробела
        - Восемь пробелов
            - Двенадцать пробелов
2. И назад

Прочитайте Создание списков онлайн: https://riptutorial.com/ru/markdown/topic/554/создание-списков

глава 8: Спойлеры Markdown или скрытый текст

замечания

Этот синтаксис spoiler специфичен для Stack Exchange и не является частью стандартного синтаксиса уценки.

Examples

Создание спойлеров

Спойлеры используются, чтобы скрыть текст или изображения, которые в противном случае негативно повлияли бы на опыт другого пользователя. Они могут быть созданы с помощью `>!`

```
>!This is hidden until your cursor hovers on top of it
```

> Это скрыто, пока ваш курсор не окажется на вершине

Примечание. Это не входит в стандартную спецификацию разметки, и она не будет отображаться во всех анализаторах уценки.

Прочитайте Спойлеры Markdown или скрытый текст онлайн:
https://riptutorial.com/ru/markdown/topic/531/спойлеры-markdown-или-скрытый-текст

глава 9: таблицы

замечания

Таблицы поддерживаются только в определенных вариантах Markdown, включая Markdown Extra и Github Flavored Markdown , но не в исходном синтаксисе Markdown или в CommonMark .

Таблицы Markdown также не поддерживаются на сайтах Stack Exchange (за исключением бета-версии Documentation).

Examples

Создание таблицы

Таблицы Markdown физически представлены с помощью тире - для разделения строки заголовка из содержимого и трубы | для столбцов.

колонка	колонка
клетка	клетка

производится

```
Column | Column
------ | ------
Cell   | Cell
```

Вы также можете заполнить таблицу любым способом -

Письмо	цифра	символ
	4	$
	365	(
б		^

Код этой таблицы:

```
Letter | Digit | Character
------ | ------|----------
a      | 4     | $
       | 365   | (
```

```
b      |       | ^
```

Markdown игнорирует интервал. Та же таблица может быть написана следующим образом:

```
Letter|Digit|Character
---|---|---
a|4|$
 |365|(
b| |^
```

и все еще показывают то же самое:

Письмо	цифра	символ
	4	$
	365	(
б		^

ПРИМЕЧАНИЕ . Если вам нужен столбец void, вы должны добавить пробел между трубами

Как вы можете видеть, код таблицы не должен представлять интервал таблицы - это выполняется в уценке.

Вы должны захотеть выровнять содержимое таблицы. Все, что вам нужно сделать, это добавить несколько двоеточий таким образом:

Выравнивание столбца:

: используется для выравнивания столбца. Выравнивание по левому краю - это стандарт.

```
Column | Column | Column
:----- | :----: | -----:
Left   | Center | Right
align  | align  | align
```

колонка	колонка	колонка
Оставил	Центр	Правильно
выравнивать	выравнивать	выравнивать

Труба в содержимом ячейки

Если вы хотите использовать символ канала (|) в содержимом ячейки, вам нужно будет избежать его с помощью обратной косой черты.

```
Column | Column
------ | ------
\| Cell \|| \| Cell \|
```

Это приводит к следующей таблице:

колонка	колонка
| Сотовый |	| Сотовый |

Прочитайте таблицы онлайн: https://riptutorial.com/ru/markdown/topic/533/таблицы

глава 10: Форматирование текста

замечания

Форматирование текста в методах уценки обычно требует символов как в начале, так и в конце текста.

Examples

Смелый

Жирный текст может быть создан окружающим текстом с двойными звездочками или двойными подчеркиваниями:

```
**Bolded text**

__Also bolded text__
```

Результат:

Полужирный текст

Также полужирный текст

курсивный

Курсив может быть создан окружающим текстом либо звездочками, либо символами подчеркивания:

```
*Italicized text*

_Also italicized_
```

Результат:

Курсивный текст

Также выделено курсивом

Зачеркнутый

Чтобы создать ~~удачный текст~~ , обведите текст с помощью `~~double tildes~~` .

Примечание: на StackExchange это форматирование не включено. Вместо этого используйте тег html `<s>text</s>` . (В чате вы можете использовать `---three hyphens---` .)

Полужирный + Курсив

```
Creating ***bold italic*** text is simply a matter of using both
**bold** (two asterisks) and *italic* (one asterisk) at the same time,
for a total of three asterisks on either side of the text you want to format at once.
```

Создание ***жирного курсивного*** текста - это просто вопрос об использовании как **полужирных** (двух звездочек), так и *курсив* (одна звездочка) одновременно, в общей сложности три звездочки по обе стороны от текста, который вы хотите отформатировать сразу.

Горизонтальные правила

```
You can create a horizontal break to divide your text by placing three (or more) underscores

___

or asterisks

***

or hyphens

---

on their own line.
```

Вы можете создать горизонтальный перерыв, чтобы разделить текст, разместив три (или более) подчеркивания

или звездочки

или дефисы

на их собственной линии.

Между символами могут быть пробелы, и за горизонтальным правилом может сразу следовать другая:

```
_ _ _
* * * *
and the spaces don't have to be evenly distributed
*  ****
```

и пространства не должны быть равномерно распределены

HTML

Некоторые теги HTML также могут использоваться в Markdown.

- `<b>bold</b>` **полужирный**
- `<i>italic</i>` *курсив*
- `<a href="http://stackoverflow.com/">link</a>` ссылку
- `<kbd>Ctrl</kbd>` Ctrl

Именованные якоря также могут использоваться для облегчения навигации по документу. Обратите внимание, что Stack Overflow Markdown, похоже, не поддерживает это.

```
<a name="heading"></a>
# Heading 1
Text under the heading
Click on a link like [Go to Heading 1](#heading1) to go to that named anchor.
```

Заголовок 1

Текст под заголовком Нажмите ссылку, например Go to Heading 1, чтобы перейти к названному якорю.

Подстрочный / Надстрочные

`x<sub>2</sub>` создает x ₂

`x<sup>2</sup>` дает x ²

Разрывы строк и абзацы

Завершите строку с двумя или более пробелами, чтобы создать разрыв строки.

```
Ending a line with no spaces
or with just one space
doesn't create a line beak.
Use two or more spaces
to create a line break.

Use an empty line to make a new paragraph.
```

> Завершение строки без пробелов или с помощью всего одного пробела не создает линейный клюв.
> Использовать два или более пробела
> для создания разрыва строки.
>
> Используйте пустую строку для создания нового абзаца.

Прочитайте Форматирование текста онлайн: https://riptutorial.com/ru/markdown/topic/549/форматирование-текста

глава 11: Цитаты

замечания

- Некоторые реализации Markdown, такие как Stack Exchange, поддерживают кавычки spoiler , которые выглядят одинаково, но скрывают содержимое цитаты, пока вы не нажмете на нее.

Examples

квотирование

Добавив `>` перед строкой, вы можете создать цитируемый текст!

```
> I am a quote
```

> Я цитата!

Вложенные котировки

Вы можете вложить котировки просто в том числе несколько `>` символов, например так:

```
> Often makes no sense.
> > Commenting above your quote…
```

> Часто не имеет смысла.
>
> > Комментируя выше цитаты ...

Прочитайте Цитаты онлайн: https://riptutorial.com/ru/markdown/topic/546/цитаты

кредиты

S. No	Главы	Contributors
1	Начало работы с Markdown	Community, Giacomo Garabello, hairboat, J F, Jeremy Banks, jkdev, Matt Clark, RamenChef, Raystafarian, TiernanO
2	Диалекты / Ароматизаторы	Doc, gemmakbarlow, jkdev, josephsw, yeungegs
3	Заголовки	Alex Warren, chrki, hairboat, jkdev, Nathan Arthur, Raystafarian, scenography, Typothecary, Wolfgang
4	Изображений	manetsus, Nathan Arthur, Wolfgang
5	Код	Christopher Muller, David Fullerton, DavidG, ganesshkumar, J F , Jens Erat, jkdev, John Slegers, Keith Hall, Luke Hefson, Marco Bonelli, Mario, Nathan Arthur, Raystafarian, Sweeper, Wolfgang
6	соединение	Ander Biguri, bumbeishvili, intboolstring, Nathan Arthur, Thunderforge, Wolfgang
7	Создание списков	Giacomo Garabello, hairboat, Raystafarian, Wolfgang
8	Спойлеры Markdown или скрытый текст	ganesshkumar, Raystafarian, Wolfgang
9	таблицы	CD.., ganesshkumar, Giacomo Garabello, Hexaholic, J F, jkdev, Raystafarian
10	Форматирование текста	Ajedi32, Ana, chrki, hairboat, Karsten 7., Keith Hall, Null, Paul Jacobson, Raystafarian, scenography, Wolfgang
11	Цитаты	intboolstring, Karsten 7., OverZealous, Quill, Wolfgang