Markdown

Markdown (маркдаун) — облегчённый язык разметки, созданный с целью написания максимально читаемого и удобного для правки текста, но пригодного для преобразования в языки для продвинутых публикаций

Элементы разметки

Абзацы и переводы строки

Абзац записывается в Markdown весьма просто: это одна или несколько строк текста, отделённые от окружающего текста одной (или более чем одной) пустой строкою. (Пустой строкою считается всякая, которая выглядит пустой; строка, не содержащая ничего, кроме пробелов и символов табуляции, считается пустой.) Обычные параграфы не следует снабжать отступом из пробелов или символов табуляции.

Следствием правила «одна или несколько строк текста» является поддержка Маркдауном таких абзацев текста, в которых имеются «жёсткие переводы строки» (то есть такие места́, в которых автор нажимал на кнопку «ввод», также называемую «Enter»).

Чтобы вставить видимый перенос строки (элемент) посредством Markdown, надобно окончить строку двумя (или более) пробелами, и только затем нажать на «ввод» («Enter»).

Безусловно, это требует несколько больших усилий, нежели простое правило «каждый перевод строки преобразуется в и становится видимым», однако такое правило для Markdown не подходит. Некоторые (рассматриваемые далее) особенности синтаксиса Markdown — email-подобное блочное цитирование, многоабзацные элементы списков — гораздо лучше выглядят и работают, если их форматировать «жёсткими переводами строки».

Заголовки Markdown поддерживает два стиля заголовков: подчёркнутые и выделенные символом «#». Подчёркнутые заголовки подчёркиваются знаками равенства (если заголовок первого уровня) или дефисами (если второго уровня). Вот пример:

Заголовок первого уровня (соответствует H1 в HTML)

Заголовок второго уровня, H2

Годится какое угодно количество подчёркивающих символов «=» или «-». Заголовки, выделенные символом «#», используют от одного до шести таких символов подряд в начале строки; количество символов соответствует уровню заголовка (от первого до шестого). Вот пример:

Заголовок первого уровня

Заголовок второго уровня

Заголовок шестого уровня (H6)

При желании можно снабжать эти заголовки «закрывающими» символами «#». (Это не обязательно, но вполне возможно, если автор текста полагает, что с «закрывающими» символами «#» заголовок выглядит красивее.) Количество таких конечных символов не обязано соответствовать количеству начальных символов. Уровень заголовка определяется только количеством начальных символов «#»:

Заголовок первого уровня

Заголовок второго уровня

Заголовок третьего уровня

Цитаты

Markdown использует email-подобный стиль использования символов > для оформления цитат. Поэтому тем авторам, которым привычна электронная почта, уже́ известен и маркдауновский способ создания цитат. Лучше всего

цитата выглядит, если оформить её «жёсткими переводами строки» (см. выше), и каждую строку начать символом «>»:

> Это цитата, состоящая из двух абзацев текста. Вы видите, что первый абзац состоит из нескольких строк, каждой из которых предшествует символ закрывающей угловой скобки.
>
> Это второй абзац текста. Обратите внимание на то, что символом цитирования снабжён также и промежуток между абзацами.
>
> Это цитата, состоящая из двух абзацев текста. Вы видите, что первый абзац состоит из нескольких строк, но только первой из них предшествует символ закрывающей угловой скобки.
>
> Это второй абзац текста. Цитаты могут быть вложенными (то есть цитатами внутри цитат), для их разметки используются дополнительные уровни «>»:
>
> Это первый уровень цитирования.
>
> > Это вложенная цитата.
>
> Возвращаемся на первый уровень цитирования. Цитаты могут содержать и другие элементы Markdown — заголовки, списки, кодовые блоки:
>
> ## Это заголовок.
>
> 1. Это первый элемент нумерованного списка.
> 2. Это второй элемент нумерованного списка.
>
> Вот пример кода:

Списки

Markdown позволяет составлять нумерованные и ненумерованные списки.

В качестве маркеров ненумерованного списка используются звёздочки, или плюсы, или дефисы; эти символы могут для этой цели использоваться взаимозаменяемо.

Скажем, вот этот пример:

- Красный
- Зелёный
- Синий совершенно равносилен вот этому:
- Красный
- Зелёный
- Синий или вот этому:
- Красный
- Зелёный
- Синий Нумерованные списки используют в качестве маркеров числа с точкою:

1. Чкалов
2. Байдуков
3. Беляков Конкретные числа, которые используются при разметке списка, не оказывают никакого влияния на производящийся Маркдауном итоговый HTML-код. По вышеприведённому списку Markdown изготовит такой HTML-код:

Чкалов

Байдуков

Беляков

Если вместо этого записать список так:

1. Чкалов
2. Байдуков
3. Беляков или даже так:
4. Чкалов
5. Байдуков
6. Беляков

Однако в любом случае нумерация нумерованного списка должна начинаться с единицы, хотя создатель Markdown предполагает, что в дальнейшем

развитии Markdown может начать поддерживать нумерацию, начинающуюся произвольным числом.

Маркеры списков обычно начинаются с начала строки, но им могут предшествовать до трёх пробелов. За маркером должен следовать либо пробел (один или более), либо символ табуляции.

Для красоты можно снабжать последующие строки списка отступами:

- Это первый элемент списка. Он содержит жёсткие переносы строк, и в начале каждой строки находится отступ, соответствующий началу текста после маркера, расположенного в первой строке.
- Это второй элемент списка. Он оформлен сходным образом: в начале каждой строки его находится отступ, соответствующий началу текста после маркера, расположенного в первой строке. Но если нет надобности, то на это можно не тратить время:
- Это первый элемент списка. Достаточно поставить маркер списка в начале первой строки его, и все остальные строки можно начинать от левого края без отступа.
- Это второй элемент списка. Он оформлен сходным образом: в начале первой строки его находится маркер, а остальные строки следуют за нею безо всякого дополнительного отступа. Если элементы списка разделяются пустыми строками, то Markdown обернёт эти элементы тэгами

  в своём итоговом HTML-коде. Скажем, вот этот список:

- Святослав
- Владимир будет преобразован к виду

Святослав

Владимир

а зато вот этот:

- Святослав
- Владимир будет преобразован к виду

Святослав

Владимир

Элементы списка могут состоять из нескольких абзацев. Второй, и третий, и каждый последующий абзац элемента должны снабжаться отступом из четырёх пробелов или одного символа табуляции:

1. Это элемент списка, состоящий из двух абзацев. Нетрудно заметить, что оба они снабжены жёсткими переносами строк и отступами, поэтому выглядят красиво.

   Это второй абзац первого элемента. Обратите внимание на отступ перед его первой строкою.

2. Это второй элемент списка. Список выглядит красиво, если отступом снабжать каждую строку. Но если нет времени этим заниматься, то для Markdown достаточно снабдить им только первую строку абзаца:

3. Это элемент списка, состоящий из двух абзацев. И хотя не все строки абзацев не снабжаются отступами, Markdown всё же способен понять, что второй абзац является частью списка.

   Это второй абзац первого элемента. Обратите внимание, что отступ перед первой строкою абзаца вполне достаточен для Markdown; все остальные строки абзаца внутри элемента списка можно начать и таким образом, как если бы абзац этот не был частью списка.

4. Это второй элемент списка. Если внутри элемента списка располагается цитата, то надобно снабжать отступом разделители «>», цитате предшествующие:

- Этот элемент списка содержит цитату:

  > Вот эта цитата, содержащаяся внутри элемента списка. Чтобы поместить кодовый блок внутри элемента списка, этот кодовый блок надобно снабдить двукратным отступом, то есть либо восемью пробелами, либо двумя символами табуляции:

- Этот элемент списка содержит кодовый блок:

```
первая строка кода;
вторая строка кода;
...
```

Блоки кода

Блоки отформатированного кода приводятся в статьях о программировании или о разметке текста, когда возникает надобность процитировать исходники. В отличие от обычных абзацев, переносы строк в кодовом блоке воспринимаются буквально. Markdown обрамляет кодовый блок сразу двумя элементами HTML — и

, и .

Чтобы создать блок кода в Markdown, достаточно просто снабдить каждую строку блока отступом, состоящим из четырёх пробелов или одного символа табуляции. Например, вот этот текст:

Это обычный абзац.

```
А это кодовый блок.
```

будет преобразован Маркдауном в следующий HTML-код:

Это обычный абзац.

Один уровень отступа (4 пробела или 1 символ табуляции) устраняются из каждой строки кодового блока. Например, вот этот текст:

Это пример на языке AppleScript:

```
tell application ”Foo”
    beep
end tell
```

будет преобразован к виду

Это пример на языке AppleScript:

Блок кода продолжается до тех пор, покуда не отыщется строка без отступа (или конец текста).

Внутри кодового блока амперсанды («&») и угловые скобки («<» и «>») автоматически преобразуются в HTML-сущности (HTML entities). Поэтому нетрудно использовать Markdown для включения примеров на языке HTML: достаточно вставить их и снабдить отступом, а Markdown позаботится о кодировании амперсандов и угловых скобок. Скажем, вот этот пример:

```
<div class=”footer”>
    &copy; 2004 Foo Corporation
</div>
```

примет вид следующего HTML-кода:

```
<code>&lt;div class=”footer”&gt;
    &amp;copy; 2004 Foo Corporation
&lt;/div&gt;
</code>
```

Обычный синтаксис Markdown не обрабатывается внутри блоков кода. Например, звёздочки внутри кодового блока считаются просто звёздочками в буквальном смысле. Так что нетрудно использовать Markdown, чтобы на нём писать о его собственном синтаксисе, приводя наглядные примеры.

Горизонтальная черта Чтобы создать горизонтальную черту (соответствующую HTML-элементу

), достаточно поместить три (или более) дефиса, или звёздочки, или символа подчёркивания, на отдельной строке текста. Если угодно, между дефисами или звёздочками можно располагать пробелы. Каждая из следующих строк соответствует горизонтальной черте:

Гиперссылки

Markdown поддерживает три стиля оформления гиперссылок:

с немедленным указанием адреса; подобные сно́скам, простую вставку URL. Первая пара стилей предполагает, что помимо URLa существует ещё текст ссылки; в разметке он [заключается в квадратные скобки].

Гиперссылка с немедленным указанием адреса

Чтобы создать ссылку с немедленным указанием адреса, надобно немедленно после закрывающей квадратной скобки поместить обычные (круглые) скобки; в этих круглых скобках приводится тот URL, на который указывает гиперссылка, а за ним может ещё быть указан (в кавычках) всплывающий заголовок-подсказка ссылки. Вот пример:

Это пример ссылки с немедленным указанием адреса. Эта ссылка снабжена всплывающим заголовком-подсказкою.

А для этой ссылки заголовок не указан. Из этого примера выйдет следующий HTML-код:

Это пример ссылки с немедленным указанием адреса. Эта ссылка снабжена всплывающим заголовком-подсказкою.

А для этой ссылки заголовок не указан.

Для ссылок на локальный ресурс (то есть расположенный на том же сервере) можно использовать относительные пути:

Подробности приводятся на странице About. Гиперссылка, подобная сноске

Ссылка, подобная сноске, вместо целевого адреса использует вторую пару квадратных скобок, внутри которых помещается метка, идентификатор ссылки:

Это пример ссылки, подобной сноске. Для разделения двух пар квадратных скобок может использоваться необязательный пробел:

Это пример ссылки, подобной сноске. Затем, где угодно в документе, следует определить эту метку ссылки, для чего используется отдельная строка следующего примерно вида:

Строка эта состоит из следующих элементов:

Идентификатор ссылки, окружённый квадратными скобками (которым может предшествовать необязательный отступ — от одного до трёх пробелов). Двоеточие, Один или несколько пробелов (или символов табуляции). URL гиперссылки. Необязательный заголовок (всплывающая подсказка) гиперссылки, заключённый либо в двойные или одиночные кавычки, либо в скобки. Три следующие определения ссылки совершенно равносильны:

```
[foo]: http://example.com/  "А здесь необязательный заголовок"
[foo]: http://example.com/  'А здесь необязательный заголовок'
[foo]: http://example.com/  (А здесь необязательный заголовок)
```

URL гиперссылки может (но не обязательно) быть помещён в угловые кавычки:

Можно поместить заголовок на следующей строке, а перед ним отступ из пробелов (или символов табуляции); с длинными URL такая запись выглядит красивее:

Такие определения ссылок используются только для создания самих ссылок при обработке текста в Markdown; из итогового HTML-кода определения ссылок убираются.

Идентификаторы ссылок могут состоять из букв, цифр, пробелов и знаков пунктуации, однако они не чувствительны к регистру. Например, вот эти две ссылки: [link text][a] [link text][A]

[link text][a] [link text][A] совершенно равносильны.

BibTeX

BibTeX — программное обеспечение для создания форматированных списков библиографии. BibTeX используется совместно с LaTeX'ом и входит во все известные дистрибутивы TeX и LaTeX.

BibTeX был создан Ореном Паташником и Лесли Лэмпортом в 1985 году. BibTeX позволяет легко работать со списками источников, отделяя библиографическую информацию от её представления. Принцип отделения содержимого от его представления использован как в самом LaTeX'е, так и в XHTML, CSS и др.

- список литературы генерируется автоматически по всем ссылкам [**?**, **?**] упомянутым в тексте;
- можно использовать единую библиографическую базу (bibфайл)во всех своих текстах, во всех работах отдела, и т. д.;
- легко обмениваться библиографическими базами с коллегами;
- нет необходимости помнить правила оформления библиографии, так как BibTeX делает эту работу автоматически с помощью стилевых bst-файлов.[@wiki] [@bibtex]

BibTeX использует bib-файлы специального текстового формата для хранения списков библиографических записей. Каждая запись описывает ровно одну публикацию — статью, книгу, диссертацию, и т. д.

Bib-файлы можно использовать для хранения библиографических баз данных. Многие программы, работающие с библиографиями, (такие, как JabRef) могут экспортировать ссылки в bib-формат.

Каждая запись выглядит следующим образом:

```
@ARTICLE{tag,
 author = {Список авторов},
 title = {Название статьи},
 year = {год},
journal = {Название журнала}
      }
```

BibTeX использует bst-файлы для описания того, как bib-записи преобразуются в текст на LaTeX'е. Каждый bst-файл представляет собой программу на простом стековом языке программирования, напоминающем Forth или PostScript. Есть программы, позволяющие генерировать .bst-файлы автоматически (например, custom-bib или Bib-it).

JabRef

JabRef — это система управления библиографической информацией, которая использует BibTeX, как нативный формат. JabRef предоставляет удобный интерфейс для редактирования файлов BibTeX, импортирования данных из онлайн научных баз данных и для поиска и управления BibTeX файлами. Приложение написано на языке программирования Java, и является кроссплатформенным.

Возможности JabRef

- Полностью совместим с BibTeX

-Полнотекстовый поиск по всей библиографии.

- Импорт различных форматов: BibTeXML, CSA, Refer/Endnote, Web of Knowledge, SilverPlatter, Medline/Pubmed (xml), Scifinder, OVID, INSPEC, Biblioscape, Sixpack, JStor and RIS.
- Экспорт в разных форматах HTML, Docbook, BibTeXML, MODS, RTF, Refer/Endnote и OpenOffice.org.
- Группировка по любым полям BibTeX, ключевым словам.
- Интеграция с десктоп-окружением: запуск программ просмотра PDF/PS, браузера, вставка цитирований в LyX, Kile, LatexEDitor, Emacs, Vim и WinEdt, OpenOffice.org (с помощью плагина)

- Поддержка плагинов — расширений.
- Автоматическое создание BibTeX ключей

Интерфейс программы легко описывается по ввержнему меню окна программы: 1. File - группа для создания или открытия базы данных, bib-файла, коллекции bib-файлов, синхронизации БД с внешним источником, импорта и экспорта БД или ее частей. Также присутствует возможность сохранять различные сессии для работы разных пользователей. 2. Edit - стандартная группа команд для редактирования и подсветки результатов. 3. Search - поиск элементов, а также, обнаружение дупликатов и экспорт в буфер обмена команд на цитирование. 4. View - отображение таблиц и шрифтов. 5. BibTeX - создание новой библиографической записи с выбором ее типа, редактирование ее дополнительных параметров. 6. Tools - связь с внешними источниками или программами, экспортирование в редактор WinEdt, открытие привязанного текста статьи из pdf-файла или URL ссылки или DOI, автогенерация ключевых слов. 7. Plugins - подключение дополнительных программных модулей. 8. Options - свойства программы, установки по умолчанию, настройки экспорта. 9. Help - меню помощи и указатель на сайт разработчика. Сортировка по полю происходит нажатием на имя столбца. Для редактирования записи дважды кликните на ней, после этого перемещайтесь между горизонтальными вкладками с группами обязательных и необязательных полей. Последняя вкладка отвечает за содержимое bib-файла, которое можно редактировать вручную.

Pandoc

Pandoc — универсальная утилита («швейцарский нож») для работы с текстовыми форматами. Основная сфера применения — форматирование математических и технических текстов.

Pandoc представляет собой кроссплатформенную программу с командным интерфейсом, способную преобразовывать тексты в самых разнообразных разметках в многочисленные выходные форматы.

Так, например с использованием pandoc можно конвертировать исходные документы в разметках ASCIIdoc, Wiki, Markdown в HTML. Если установить LaTex, то становится возможным получение и PDF.

Действительно, pandoc справляется с конвертированием без каких-либо потерь информации. При конвертировании из формата MarkDown поддерживается чтение трех параметров метаданных — заголовка, автора и даты документа. Поддерживается так же передача параметров командной строки для установки некоторых специфических свойств, например языка документа. Есть возможность задать свой шаблон выходного документа, до некоторой степени видоизменяя его.

Основные команды

Ниже рассмотрим самые основные параметры и опции Pandoc, В остальных случаях желательно познакомится с довольно объёмным руководством на сайте или через руководство man.

Pandoc - программа консольная, и ей требуется передавать в качестве параметров имя входного файла, а и при помощи опции “-o” - целевого файла. Программа может понимать входной и выходной формат разметки по расширению файла.

pandoc input.md -o output.html

Однако можно указать входной и выходной форматы при помощи специальных опций. Выходной формат задаётся опциями “-t” или “–to”, а входной - опциями “-f” или “–from”. Например, следующая команда перекодирует файл input.md с разметкой Markdown в файл output.txt с html-разметкой

pandoc input.md -o output.txt -t html

Если опции форматов или расширения файлов не указаны или не известны для Pandoc, то по умолчанию программа считает форматом входного файла Mark-down, а выходного - HTML.

Математика

Интересной особенностью Pandoc является поддержка конвертирования математических формул из разметки LaTeX в представление HTML. Для вывода математики в HTML используются на выбор несколько специальных математических движков на основе MathML, Java-Script, онлайн-сервисов, код которых будет вставлен в сконвертированный HTML-файл.

Для конвертирования математических формул можно использовать следующие опции:

--mathml - преобразует формулы LaTeX в разметку MathML;

--webtex - преобразует формулы LaTeX при помощи онлайн-сервиса Google Chart AP

--mathjax - преобразует формулы LaTeX при помощи расширения MathJax для Media

--latexmathml - преобразует формулы LaTeX при помощи JS-библиотеки Latexmath