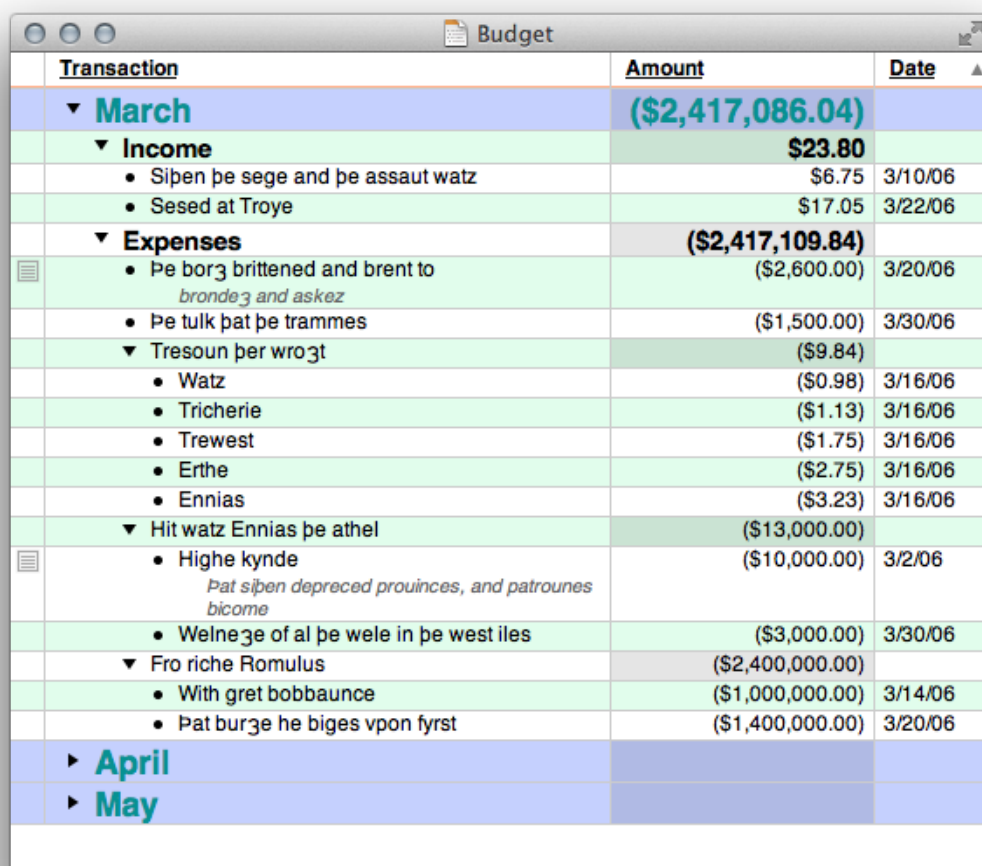


org-mode: 最好的文档编辑利器，没有之一

尽管按照 org-mode [官方](#) 的说法，Org 是一个基于快速高效的文本方式来实现做笔记、管理待办事项（TODO list）以及做项目计划的模式（Org is a mode for keeping notes, maintaining TODO lists, and doing project planning with a fast and effective plain-text system），但 Org-mode 首先是最好的文档编辑利器，没有之一。

我之前用过很多年 MS Word，也尝试过 OpenOffice/LibreOffice Writer，以及 iWorks Pager，但都不理想，写文档是没有痛快淋漓的感觉。直到后来发现了 [Omni Outliner](#)，才终于找到了写作的乐趣。但是了解了 org-mode 之后，发现原来一切都是浮云。只有 Org-mode 才是终极的解决之道。使用 org-mode 写文档的时候，你只需要关注内容本身，而不需要写上几个字，选中它们按 Ctl-B，或者停下来用鼠标去点击“标题 1”，更甚觉得那个标题格式不顺眼，开始去调整样式，而停下写作的思路。

BTW，Omni Outliner 似乎也是从 Org-mode 找到的灵感，有图为证：



Transaction	Amount	Date
▼ March	(\$2,417,086.04)	
▼ Income	\$23.80	
• Sipen be sege and be assault watz	\$6.75	3/10/06
• Sesed at Troye	\$17.05	3/22/06
▼ Expenses	(\$2,417,109.84)	
• Pe bor3 brittended and brent to bronde3 and askez	(\$2,600.00)	3/20/06
• Pe tulk pat be trammes	(\$1,500.00)	3/30/06
▼ Tresoun per wro3t	(\$9.84)	
• Watz	(\$0.98)	3/16/06
• Tricherie	(\$1.13)	3/16/06
• Trewest	(\$1.75)	3/16/06
• Erthe	(\$2.75)	3/16/06
• Ennias	(\$3.23)	3/16/06
▼ Hit watz Ennias be athel	(\$13,000.00)	
• Highe kynde Pat sipen deprecd provinces, and patrounes bicomie	(\$10,000.00)	3/2/06
• Welne3e of al be wele in be west iles	(\$3,000.00)	3/30/06
▼ Fro riche Romulus	(\$2,400,000.00)	
• With gret bobbaunce	(\$1,000,000.00)	3/14/06
• Pat bur3e he biges vpon fyrst	(\$1,400,000.00)	3/20/06
► April		
► May		

其功能与 Org-mode 几乎一样，而且这货居然支持 emacs 快捷键！

尽管 Omni Outliner 以 GUI 的方式实现了 org-mode 的功能，但是并不是很理想：它的内容是“所见即所得”的，很多时候你难以更改其样式，而且只能在 Mac OS 下使用。而 Org-mode 使用文本方式，具有如下优势：

- 格式通用，系统无关，软件无关
- 体积小，速度快
- “所想即所得”，比“所见即所得”更人性化

可以说，正是由于有了 Org-Mode，Emacs 处理文本的能力才得到了大幅度的提高，使得 Emacs 能够被非程序员接受。如果说 LaTeX 是排版的终极，那么 Org-mode 就是编辑的终极。Emacs 22 以后的版本已经集成了 org-mode，打开 .org 扩展的文件会自动进入 org 模式。此外，Vim 下面也有了对应的 Org-mode。

本文介绍最基本的编辑、格式化文本以及导出功能，后续再讨论高级玩法。

Table of Contents

- [1 用大纲组织内容](#)
 - [1.1 定义标题](#)
 - [1.2 大纲相关的快捷键](#)
 - [1.2.1 折叠大纲](#)
 - [1.2.2 在大纲之间移动](#)
 - [1.2.3 基于大纲的编辑](#)
 - [1.3 大纲的显示方式](#)
- [2 超链接和图文混排](#)
 - [2.1 创建链接](#)
 - [2.2 内部链接](#)
 - [2.3 显示图片](#)
 - [2.4 创建链接](#)
 - [2.5 内部链接](#)
 - [2.6 显示图片](#)
- [3 轻量级标记语言](#)
 - [3.1 字体](#)
 - [3.2 表格](#)
 - [3.2.1 创建和转换表格](#)
 - [3.2.2 调整和区域移动](#)
 - [3.2.3 编辑行和列](#)
 - [3.3 段落](#)
 - [3.4 列表](#)
 - [3.4.1 列表操作快捷键](#)
 - [3.5 分隔线](#)

- [4 标签](#)
 - [4.1 tag 的作用](#)
 - [4.2 标记 tag](#)
 - [4.3 预定义 tag](#)
 - [4.4 按 tag 搜索](#)
- [5 导出和发布](#)
 - [5.1 准备工作](#)
 - [5.1.1 文档元数据](#)
 - [5.1.2 内容元数据](#)
 - [5.1.3 嵌入 Html](#)
 - [5.1.4 包含文件](#)
 - [5.1.5 嵌入 LaTeX](#)
 - [5.2 导出](#)
 - [5.3 发布](#)

1 用大纲组织内容

尽管 Org-mode 的功能不断丰富，现在已经可以记笔记，管理个人事务，制定项目计划以及很多其他的用途，但是最初和最基本的功能还是通过大纲(outline)的方式来编辑文档。而且，无论是笔记管理，任务管理还是项目计划的编写，都是以对内容进行高效的组织（organization)为基础的。

在编辑文档，尤其是大型文档的时候，对内容的组织就显得尤为重要。经常需要在文档中快速定位，只关注某一部分的内容，Word 之类的编辑器，通过文档结构图来定位文档位置，速度很慢，而且很多时候不能满足编辑的需要。尽管 Word 也提供了“大纲视图”，但是，唉……不说也罢，如果那个功能真的好，也就不需要 Omni Outliner 了。

Org-mode 天然支持大纲视图，通过在文档中定义标题，可以方便的浏览每个小节，从而把握文档的总体内容。Org 是基于 Outline 模式的，它提供了更灵活的编辑结构文件的命令。比如折叠文档，针对大纲的编辑功能等，极其强大。

1.1 定义标题

要实现大纲，首先要定义标题。用 emacs 新建一个 orgmode.org，输入如下内容：

```
* org-mode
** 大纲
正在编写大纲
** 轻量级标记语言
* 可以导出其他格式
支持 html, pdf 等格式
```

注意：

1. * 要位于每行的行首
2. * 之后要有一个空格，然后再输入标题

3. 连续几个*就表示是第几级大纲，最多支持 10 级。

此时看起来应该是这个样子：

```
* org-mode
** 大纲
正在编写大纲
** 轻量级标记语言
* 可以导出其他格式
支持html,pdf等格式
█
```

org-mode

觉得没什么出奇的地方，只是改变了一些颜色？其真正的用处在于可以通过大纲操作文档，包括折叠，定位和编辑。而这些操作都通过快捷键实现，非常有效率。尤其是对大文档。

1.2 大纲相关的快捷键

1.2.1 折叠大纲

快捷键	命令	说明
S-TAB	org-shifttab	循环切换整个文档的大纲状态（三种状态：折叠，打开下一级，打开全部）
TAB	org-cycle	循环切换光标所在大纲的状态

1.2.2 在大纲之间移动

快捷键	命令	说明
C-c C-n/p		下/上一标题
C-c C-f/b		下/上一标题（仅限同级标题）
C-c C-u		跳到上一级标题
C-c C-j		切换到大纲浏览状态

1.2.3 基于大纲的编辑

快捷键	命令	说明
M-RET		插入一个同级标题
M-S-RET		插入一个同级 TODO 标题
M-LEFT/RIGHT		将当前标题升/降级
M-S-LEFT/RIGHT		将子树升/降级

快捷键	命令	说明
M-S-UP/DOWN		将子树上/下移
C-c *		将本行设为标题/正文
C-c C-w		将子树或区域移动到另一标题处（跨缓冲区）
C-x n s/w		只显示当前子树/返回
C-c C-x b		在新缓冲区显示当前分支（类似 C-x n s）
C-c /		只列出包含搜索结果的大纲，并高亮，支持多种搜索方式
C-c C-c		取消高亮

更多的快捷键可以通过 C-c C-x C-h 查看。

1.3 大纲的显示方式

默认的大纲显示没有缩进，显得有些乱。可以用 M-x org-indent-mode 切换到另一种显示方式：

```
***** 定义标题...
***** 大纲相关的快捷键...

***** 大纲的显示方式
默认的大纲显示没有缩进，显得有些乱。可以用 M-x org-indent-mode 切换
到 org-indent-mode 模式动态地实现，它会在每行前面加上一些前导空格。你
可以在 .org 文件中设置 org-startup-indent 为所有的文件打开 org-indent-mode 模式，或
在 .org 文件中设置 \#+STARTUP: indent 为单个文件打开缩进。

***** 文本列表...
*** 超链接和图文混排...
*** 轻量级标记语言...
*** 导出和发布...
```

如果想让某个文件默认用这种方式打开，可以在文件头部增加：

```
\#+STARTUP: indent
```

如果希望打开所有 org 文件都默认用这种方式，可以在 .emacs 中配置：

```
(setq org-startup-indent t)
```

2 超链接和图文混排

超链接也是组织内容的一种非常有效的方式。Org 支持多种超链接。对于符合要求的图片链接，可以形成图文混排。

2.1 创建链接

对于符合链接规则的内容，org-mode 会自动将其视为链接，包括文件、网页、邮箱、新闻组、BBDB 数据库项、IRC 会话和记录等。下面是一些例子：

http://www.astro.uva.nl/~dominik	on the web
file:/home/dominik/images/jupiter.jpg	file, absolute path
/home/dominik/images/jupiter.jpg	same as above
file:papers/last.pdf	file, relative path
file:projects.org	another Org file
docview:papers/last.pdf::NNN	open file in doc-view mode
at page NNN	
id:B7423F4D-2E8A-471B-8810-C40F074717E9	Link to heading by ID
news:comp.emacs	Usenet link
mailto:adent@galaxy.net	Mail link
vm:folder	VM folder link
vm:folder#id	VM message link
wl:folder#id	WANDERLUST message link
mhe:folder#id	MH-E message link
rmail:folder#id	RMAIL message link
gnus:group#id	Gnus article link
bbdb:R.*Stallman	BBDB link (with regexp)
irc:/irc.com/#emacs/bob	IRC link
info:org:External%20links	Info node link (with
encoded space)	

对于文件链接，可以用::后面增加定位符的方式链接到文件的特定位置。定位符可以是行号或搜索选项。
如：

file:~/code/main.c::255	进入到 255 行
file:~/xx.org::My Target	找到目标 '<<My Target>>'
file:~/xx.org/::#my-custom-id	查找自定义 id 的项

除了上述的自动链接外，还可以显示指定链接，采用如下格式：

```
[[link]][description]]  
[[link]]
```

显示指定的链接可以不显示原始的 URL 而是显示对该链接的描述。这种方式可以用相对路径链接本地文件。

对于显示指定的链接，即可以手工输入，也可以用 org-mode 提供的快捷键进行编辑：

快捷键	命令	说明
C-c l		保存链接
C-c C-l	org-insert-link	创建或修改链接，可以引用已保存的链接
C-c C-o	org-open-at-point	打开链接

C-c %	记录内部链接地址
C-c &	跳转到已记录的内部链接

2.2 内部链接

前面的例子都是外部链接，Org-mode 还支持内部链接：

```
定义锚点 #<<my-anchor>>
[[my-anchor][内部链接]]
```

脚注可以看作是一种特殊的内部链接，但是要求具有"fn:"前缀：

```
添加脚注链接 [[fn:footprint1][脚注 1]]
定义脚注 [fn:footprint1]
```

2.3 显示图片

尽管不看重"所见即所得"，但有时候能够看到图文混排的内容还是很有必要的。通过 iimage 这个 minor mode，可以在 Org-mode 中显示图片。

下载 iimage.el 文件扔到 Emacs 的目录里，然后在 .emacs 里添加下面的代码：

```
;; iimage mode
(autoload 'iimage-mode "iimage" "Support Inline image minor mode." t)
(autoload 'turn-on-iimage-mode "iimage" "Turn on Inline image minor
mode." t)
```

然后就可以用命令

M-x iimage-mode RET

在当前模式里启动 iimage 这个 minor mode。

iimage-mode 目前只能显示以文件方式链接的图片。

混排 超链接也是组织内容的一种非常有效的方式。Org 支持多种超链接。对于符合要求的图片链接，可以形成图文混排。

2.4 创建链接

对于符合链接规则的内容，org-mode 会自动将其视为链接，包括文件、网页、邮箱、新闻组、BBDB 数据库项、IRC 会话和记录等。下面是一些例子：

http://www.astro.uva.nl/~dominik	on the web
file:/home/dominik/images/jupiter.jpg	file, absolute path
/home/dominik/images/jupiter.jpg	same as above
file:papers/last.pdf	file, relative path
file:projects.org	another Org file
docview:papers/last.pdf::NNN	open file in doc-view mode
at page NNN	
id:B7423F4D-2E8A-471B-8810-C40F074717E9	Link to heading by ID
news:comp.emacs	Usenet link
mailto:adent@galaxy.net	Mail link
vm:folder	VM folder link
vm:folder#id	VM message link
wl:folder#id	WANDERLUST message link
mhe:folder#id	MH-E message link
rmail:folder#id	RMAIL message link
gnus:group#id	Gnus article link
bbdb:R.*Stallman	BBDB link (with regexp)
irc:/irc.com/#emacs/bob	IRC link

info:org:External%20links
encoded space)

Info node link (with

对于文件链接，可以用::后面增加定位符的方式链接到文件的特定位置。定位符可以是行号或搜索选项。
如：

file:~/code/main.c::255
file:~/xx.org::My Target
file:~/xx.org/::#my-custom-id

进入到 255 行
找到目标 '<<My Target>>'
查找自定义 id 的项

除了上述的自动链接外，还可以显示指定链接，采用如下格式：

[[link]][description]]
[[link]]

显示指定的链接可以不显示原始的 URL 而是显示对该链接的描述。这种方式可以用相对路径链接本地文件。

对于显示指定的链接，即可以手工输入，也可以用 org-mode 提供的快捷键进行编辑：

快捷键	命令	说明
C-c l		保存链接
C-c C-l	org-insert-link	创建或修改链接，可以引用已保存的链接
C-c C-o	org-open-at-point	打开链接

C-c %	记录内部链接地址
C-c &	跳转到已记录的内部链接

2.5 内部链接

前面的例子都是外部链接，Org-mode 还支持内部链接：

定义锚点 #<<my-anchor>>
[[my-anchor]][内部链接]]

脚注可以看作是一种特殊的内部链接，但是要求具有"fn:"前缀：

添加脚注链接 [[fn:footprint1]][脚注 1]]
定义脚注 [fn:footprint1]

2.6 显示图片

尽管不看重"所见即所得"，但有时候能够看到图文混排的内容还是很有必要的。通过 iimage 这个 minor mode，可以在 Org-mode 中显示图片。

下载 iimage.el 文件扔到 Emacs 的目录里，然后在 .emacs 里添加下面的代码：

;; iimage mode
(autoload 'iimage-mode "iimage" "Support Inline image minor mode." t)


```
(autoload 'turn-on-iimage-mode "iimage" "Turn on Inline image minor
mode." t)
```

然后就可以用命令

M-x iimage-mode RET

在当前模式里启动 iimage 这个 minor mode。

iimage-mode 目前只能显示以文件方式链接的图片。

3 轻量级标记语言

前面的大纲和超链接都是使用标记来定义的。实际上，Org 现在已经成为一种专门的轻量级标记语言，与 Markdown、reStructuredText、Textile、RDoc、MediaWiki 等并列。

相对于重量级标记语言（如 html, xml），轻量级标记语言的语法简单，书写容易。即使不经过渲染，也可以很容易阅读。用途越来越广泛。比如，gitHub 的 README 文档除了支持纯文本外，还支持丰富的轻量级标记语言，其中就包括 Org。

关于这些语言的对比，可以参考[这里](#)。下面来看一下 Org 还支持哪些标记。

3.1 字体

```
*粗体*
/斜体/
+删除线+
_下划线_
下标： H2O
上标： E=mc2
等宽字： =git= 或者 ~git~
```

3.2 表格

Org 能够很容易地处理 ASCII 文本表格。任何以 ‘|’ 为首个非空字符的行都会被认为是表格的一部分。’|’ 也是列分隔符。一个表格是下面的样子：

```
| Name | Pone | Age |
|-----+-----+-----|
| Peter | 1234 | 17 |
| Anna | 4321 | 25 |
```

你可能认为要录入这样的表格很繁琐，实际上你只需要输入表头 “[Name|Pone|Age]” 之后，按 C-c RET，就可以生成整个表格的结构。类似的快捷键还有很多：

3.2.1 创建和转换表格

快捷键	命令	说明
C-c 竖线		创建或转换成表格

3.2.2 调整和区域移动

快捷键	命令	说明
C-c C-c		调整表格，不移动光标
TAB		移动到下一区域，必要时新建一行
S-TAB		移动到上一区域
RET		移动到下一行，必要时新建一行

3.2.3 编辑行和列

快捷键	命令	说明
M-LEFT/RIGHT		移动列
M-UP/DOWN		移动行
M-S-LEFT/RIGHT		删除/插入列
M-S-UP/DOWN		删除/插入行
C-c -		添加水平分割线
C-c RET		添加水平分割线并跳到下一行
C-c ^		根据当前列排序，可以选择排序方式

3.3 段落

对于单个回车换行的文本，认为其属于同一个段落。在导出的时候将会转化为不换行的同一段。如果要新起一个段落，需要留出一个空行。这点与 MediaWiki 类似。

3.4 列表

Org 能够识别有序列表、无序列表和描述列表。

- 无序列表项以 ‘-’、 ‘+’ 或者 ‘*’ 开头。
- 有序列表项以 ‘1.’ 或者 ‘1)’ 开头。
- 描述列表用 ‘::’ 将项和描述分开。
- 有序列表和无序列表都以缩进表示层级。只要对齐缩进，不管是换行还是分块都认为是处于当前列表项。

同一列表中的项的第一行必须缩进相同程度。当下一行的缩进与列表项的的开头的符号或者数字相同或者更小时，这一项就结束了。当所有的项都关上时，或者后面有两个空行 时，列表就结束了。例如：

```
My favorite scenes are (in this order)
1. The attack of the Rohirrim
2. Eowyn's fight with the witch king
  + this was already my favorite scene in the book
  + I really like Miranda Otto.
Important actors in this film are:
```

- Elijah Wood :: He plays Frodo
- Sean Austin :: He plays Sam, Frodo's friend.

将显示为：

My favorite scenes are (in this order)

- 1. The attack of the Rohirrim
- 2. Eowyn's fight with the witch king
 - this was already my favorite scene in the book
 - I really like Miranda Otto.

Important actors in this film are:

Elijah Wood

He plays Frodo

Sean Austin

He plays Sam, Frodo's friend.

3.4.1 列表操作快捷键

为了便利，org-mode 也支持很多列表操作的快捷键，大部分都与大纲的快捷键类似：

快捷键	命令	说明
TAB		折叠列表项
M-RET		插入项
M-S-RET		插入带复选框的项
M-S-UP/DOWN		移动列表项
M-LEFT/RIGHT		升/降级列表项，不包括子项
M-S-LEFT/RIGTH		升/降级列表项，包括子项
C-c C-c		改变复选框状态
C-c -		更换列表标记（循环切换）

3.5 分隔线

五条短线或以上显示为分隔线。

4 标签

4.1 tag 的作用

对于信息的管理，有分类(category)和标签(tag)两种方式。这两种方式各有特点：

通常分类是固定的，很少变化，而 tag 随时可以增加。分类通常表现为树状结构，比较清晰，但是树状结构过于简单，不能表达复杂的信息。比如，如果有多个分类树，处理起来就会比较麻烦。

所以，这两种方式通常结合起来使用。比如 blog 系统中，通常既支持文章的分类（树），又支持为每篇文章作 tag 标记。

org-mode 作为[最好的文档编辑利器](#)，在支持文内大纲（也是树状结构）的同时，还方便的支持 tag 功能。tag 可以在多篇文档中共用。

4.2 标记 tag

在 Org-mode 中，可以对标题增加 tag 标记。标记的格式如下：

跟特留尼西特握手 :苦差:薪水:逃不掉:

而且 Org-mode 的标签自动按照大纲树的结构继承。即子标题自动继承父标题的标签。比如：

```
* Meeting with the French group      :work:
** Summary by Frank                  :boss:notes:
*** TODO Prepare slides for him      :action:
```

则最后一行标题具有 work, boss, notes, action 四个标签。

如果希望文档中的所有标题都具有某些标签，只需要定义文档元数据：

```
#+FILETAGS: :Peter:Boss:Secret:
```

如果手工输入标签，在标题后设置标签，键入:后，M-Tab 自动提供标签的补齐。

更方便的做法是在正文部分用 C-c C-q 或直接在标题上用 C-c C-c 创建标签，这种方式可以列出所有预定义的标签以便选取。

4.3 预定义 tag

上面提到，除了可以输入标签外，还可以从预定义的标签中进行选择。预定义的方式有两种：

- 在当前文件头部定义

这种方式预定义的标签只能在当前文件中使用。使用#+TAGS 元数据进行标记，如：

```
#+TAGS: { 桌面(d) 服务器(s) }  编辑器(e) 浏览器(f) 多媒体(m) 压缩(z)
```

每项之间必须用空格分隔，可以在括号中定义一个快捷键；花括号里的为标签组，只能选择一个对标签定义进行修改后，要在标签定义的位置按 C-c C-c 刷新才能生效。

- 在配置文件中定义 上面的标签定义只能在当前文件生效，如果要在所有的.org 文件中生效，需要在 Emacs 配置文件 .emacs 中进行定义：

```
(setq org-tag-alist '(
    (:startgroup . nil)
      ("桌面" . ?d) ("服务器" . ?s)
    (:endgroup . nil)
    ("编辑器" . ?e)
    ("浏览器" . ?f)
    ("多媒体" . ?m)
  ))
```

默认情况下，org 会动态维护一个 Tag 列表，即当前输入的标签若不在列表中，则自动加入列表以供下次补齐使用。

为了使这几种情况（默认列表、文件预设 tags，全局预设 tags）同时生效，需要在文件中增加一个空的 TAGS 定义：

```
#+TAGS:
```

4.4 按 tag 搜索

使用标签可以更好的管理内容。org-mode 提供了以下功能：

KEYS	COMMENT
C-c \	按 tag 搜索标题
C-c / m	搜索并按树状结构显示
C-c a m	按标签搜索多个文件（需要将文件加入全局 agenda）

可以使用逻辑表达式限制条件，更准确灵活的搜索

+	和	a+b	同时有这两个标签
-	排除	a-b	有 a 但没有 b
	或	a b	有 a 或者有 b
&	和	a&b	同时有 a 和 b，可以用“+”替代

在查询视图中 C-c C-c 退出

5 导出和发布

更多：<http://orgmode.org/manual/Exporting.html>

Org-mode 可以完美的编辑，但是最终文档可能需要发布成其他的格式。Org-Mode 支持多种文档的输出，包括：

- 文本
- 网页

- PDF (需要 Latex 支持)
- XOXO
- FreeMind/Xmind
- Docbook
- iCalendar (苹果 iCal 文件)
-

5.1 准备工作

为了更好的发布文档，还需要做一些准备工作。主要是为文档添加一些”元数据“，使得发布的时候能更好地识别文档的内容。

5.1.1 文档元数据

具体包括：

```

#+TITLE:      the title to be shown (default is the buffer name)
#+AUTHOR:     the author (default taken from user-full-name)
#+DATE:       a date, an Org timestamp1, or a format string for
format-time-string
#+EMAIL:      his/her email address (default from user-mail-address)
#+DESCRIPTION: the page description, e.g. for the XHTML meta tag
#+KEYWORDS:   the page keywords, e.g. for the XHTML meta tag
#+LANGUAGE:   language for HTML, e.g. 'en' (org-export-default-
language)
#+TEXT:       Some descriptive text to be inserted at the beginning.
#+TEXT:       Several lines may be given.
#+OPTIONS:    H:2 num:t toc:t \n:nil @:t ::t |:t ^:t f:t TeX:t ...
#+BIND:       lisp-var lisp-val, e.g.: org-export-latex-low-levels
itemize
              You need to confirm using these, or configure org-
export-allow-BIND
#+LINK_UP:    the ``up'' link of an exported page
#+LINK_HOME:  the ``home'' link of an exported page
#+LATEX_HEADER: extra line(s) for the LaTeX header, like
\usepackage{xyz}
#+EXPORT_SELECT_TAGS: Tags that select a tree for export
#+EXPORT_EXCLUDE_TAGS: Tags that exclude a tree from export
#+XSLT:       the XSLT stylesheet used by DocBook exporter to
generate FO file

```

其中#+OPTIONS 是复合的选项，包括：

```

H:           set the number of headline levels for export
num:         turn on/off section-numbers
toc:         turn on/off table of contents, or set level limit (integer)
\n:         turn on/off line-break-preservation (DOES NOT WORK)
@:           turn on/off quoted HTML tags
::           turn on/off fixed-width sections
|:           turn on/off tables
^:           turn on/off TeX-like syntax for sub- and superscripts. If
              you write "a_{b}", a_{b} will be interpreted, but
              the simple a_b will be left as it is.
-:           turn on/off conversion of special strings.
f:           turn on/off footnotes like this[1].
todo:        turn on/off inclusion of TODO keywords into exported text
tasks:       turn on/off inclusion of tasks (TODO items), can be nil to
remove

```

	all tasks, todo to remove DONE tasks, or list of kwds to
keep	
pri:	turn on/off priority cookies
tags:	turn on/off inclusion of tags, may also be not-in-toc
<:	turn on/off inclusion of any time/date stamps like
DEADLINES	
*:	turn on/off emphasized text (bold, italic, underlined)
TeX:	turn on/off simple TeX macros in plain text
LaTeX:	configure export of LaTeX fragments. Default auto
skip:	turn on/off skipping the text before the first heading
author:	turn on/off inclusion of author name/email into exported file
email:	turn on/off inclusion of author email into exported file
creator:	turn on/off inclusion of creator info into exported file
timestamp:	turn on/off inclusion creation time into exported file
d:	turn on/off inclusion of drawers

这些元数据可以根据需要设置。建议放在文档的开头部分。如，本文采用的元数据如下：

```

#+TITLE: org-mode: 最好的文档编辑利器，没有之一
#+AUTHOR: Holbrook Wong
#+EMAIL: wanghaikuo@gmail.com
#+KEYWORDS: emacs, org-mode
#+OPTIONS: H:4 toc:t

```

5.1.2 内容元数据

通常在行首以“#+”开头，可以有多种用途。

- 分行区块

默认内容不换行，需要留出空行才能换行。定义了分行的区块可以实现普通换行：

```

#+BEGIN_VERSE
Great clouds overhead
Tiny black birds rise and fall
Snow covers Emacs
-- AlexSchroeder
#+END_VERSE

```

- 缩进区块

通常用于引用，与默认格式相比左右都会留出缩进：

```

#+BEGIN_QUOTE
缩进区块
#+END_QUOTE

```

- 居中区块

```

#+BEGIN_CENTER
Everything should be made as simple as possible, \
but not any simpler
#+END_CENTER

```

- 代码区块

```

#+BEGIN_SRC ruby
require 'redcarpet'
md = Redcarpet.new("Hello, world.")
puts md.to_html

```

```
##+END_SRC
```

- 例子

: 单行的例子以冒号开头

```
##+BEGIN_EXAMPLE
多行的例子
使用区块
##+END_EXAMPLE
```

- 注释

以 ‘#’开头的行被看作注释，不会被导出

区块注释采用如下写法：

```
##+BEGIN_COMMENT
块注释
...
##+END_COMMENT
```

- 表格与图片

对于表格和图片，可以在前面增加标题和标签的说明，以方便交叉引用。

比如在表格的前面添加：

```
##+CAPTION: This is the caption for the next table (or link)
##+LABEL: tbl:table1
```

则在需要的地方可以通过

```
\ref{table1}
```

来引用该表格。

5.1.3 嵌入 Html

对于导出 html 以及发布，嵌入 html 代码就很有用。比如下面的例子适用于格式化为 cnblogs 的代码块：

```
##+BEGIN_HTML
<div class="cnblogs_Highlighter">
<pre class="brush:cpp">
int main()
{
    return 0;
}
</pre>
</div>
##+END_HTML
```

相当于在 cnblogs 的网页编辑器中插入 "c++" 代码。

5.1.4 包含文件

当导出文档时，你可以包含其他文件中的内容。比如，想包含你的 “.emacs” 文件，你可以用：

```
##+INCLUDE: "~/ .emacs" src emacs-lisp
```


可选的第二个第三个参数是组织方式（例如，“quote”，“example”，或者“src”），如果是“src”，语言用来格式化内容。组织方式是可选的，如果不给出，文本会被当作 Org 模式的正常处理。用 C-c ,可以访问包含的文件。

5.1.5 嵌入 LaTeX

对于需要包含数学符号和特殊方程的科学笔记，Org 模式支持嵌入 LaTeX 代码到文件中。你可以直接使用类 TeX 的宏来输入特殊符号，输入方程，或者整个 LaTeX 环境。

```
Angles are written as Greek letters \alpha, \beta and \gamma. The mass
if
the sun is  $M_{\text{sun}} = 1.989 \times 10^{30}$  kg. The radius of the sun is  $R_{\text{sun}}$ 
=
 $6.96 \times 10^8$  m. If  $a^2=b$  and  $b=2$ , then the solution must be either
 $a=+\sqrt{2}$  or  $a=-\sqrt{2}$ .
\begin{equation}
x=\sqrt{b}
\end{equation}
```

特殊设置之后，导出 HTML 时 LaTeX 代码片段会生成图片并包含进来。

5.2 导出

做好准备工作后，就可以导出了。使用命令：

C-c C-e

然后选择相应的格式，就可以导出对应的文件了。

5.3 发布

Org 包含一个发布管理系统，可以配置一个由相互链接的 Org 文件组成的工程项目的自动向 HTML 转换。你也可以设置 Org，将导出的 HTML 页面和相应的附件如图片，源代码文件等自动上传到服务器。

下面是一个例子：

```
(setq org-publish-project-alist
  '(("org"
    :base-directory "~/org/"
    :publishing-directory "~/public_html"
    :section-numbers nil
    :table-of-contents nil
    :style "<link rel=\"stylesheet\"
          href=\"../other/mystyle.css\"
          type=\"text/css\"/>")))
```

发布相关的命令：

命令	说明
C-c C-e C	提示指明一个项目，将所有的文件发布
C-c C-e P	发布包含当前文件的项目

命令	说明
C-c C-e F	只发布当前文件
C-c C-e E	发布所有项目

Org 用时间戳来查看文件是否改变。上面的命令只发布修改过的文件。你可以给它们加上前缀来强制重新发布所有的文件。

Date: 2012-04-15 17:59:22 CST

Author: Holbrook Wong

Org version 7.8.08 with Emacs version 23

[Validate XHTML 1.0](#)

**本人已在 github 上用 Jekyll 建立了新的博客：
<http://thinkinside.tk/>，本站文章会陆续迁移
过去**