乱谈AsciiDoc的书籍编写

houqp

乱谈AsciiDoc的书籍编写

houqp

目录

插图清单

表格清单

范例清单

前言

写文档什么的最讨厌了。

第 1 章 一个demo

这是一个最简单的示例，从书的第一页起到这一行的所有内容可以通过以下的代码生成：

```
乱谈AsciiDoc的书籍编写
======================
houqp
v0.1, 2011-09
:doctype: book

[preface]
前言
----

写文档什么的最讨厌了。

一个demo
--------
这是一个最简单的示例，从书的第一页起到这一行的所有内容可以通过以下的代码生成：
```

很简单吧，AsciiDoc属于轻量级标记语言，其代码本身就有很高的可读性。这就是AsciiDoc的魅力，用简单的标记做出不简单的效果。

第 2 章 工具链的使用

这一章介绍AsciiDoc环境的搭建，在搭建过程中可以使用第 1 章 一个demo中的代码进行测试。

2.1. 英文文档的制作

如果你只需要编写英文书籍，那么几乎不需要改动什么，AsciiDoc所带的工具就能帮你生成你所需要的输出。

例 2.1. pdf的生成

```
a2x -v -f pdf --fop book.txt 1
```

1 -f指定输出的文档格式，而--fop则指定所使用的后端。用AsciiDoc生成pdf有两种后端可选，一个是fop，另一个是dblatex。关于两种后端的优劣，请参考[pdf_backend]。

改动-f的参数你就能生成HTML格式的书籍：

例 2.2. HTML的生成

```
a2x -v -f xhtml book.txt 1
```

2.2. 中文支持

如果你用上面的方法生成中文文档，会发现所有的中文都变成了“#”号。这是因为fop后端找不到对应的中文字体，所以你需要做一些另外的设置。

AsciiDoc生成文档的方式是先转换为DocBook文档，再使用DocBook的工具链进行编译。在编写DocBook文档时，我们通常使用XSL来进行输出排版的调整，因此在编译AsciiDoc文档时同样可以使用XSL。

例 2.3. 中文支持的XSL

```
<?xml version='1.0' encoding='UTF-8'?>

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns="http://www.w3.org/1999/xhtml"
                version="1.0">

  <xsl:param name="l10n.gentext.language" select="'zh_cn'"/>
  <xsl:param name="body.font.family">
    WenQuanYi Micro Hei, AR PL KaitiM GB, AR PL SungtiL GB,  1
  </xsl:param>
  <xsl:param name="monospace.font.family">
    WenQuanYi Micro Hei, AR PL SungtiL GB
  </xsl:param>
  <xsl:param name="title.font.family">
    WenQuanYi Micro Hei, AR PL SungtiL GB
  </xsl:param>

</xsl:stylesheet>
```

1 1 这里指定了正文部分选用字体的顺序。

指定完中文字体后还不够，因为你还要告诉fop去哪里找这些字体，所以我们还需要一个fop.xconf的文件：

例 2.4. fop.xconf的中文设置

```
<?xml version="1.0"?>

<fop version="1.0">

  <base>.</base>

  <renderers>
    <renderer mime="application/pdf">
      <fonts>

        <font metrics-url="./fop/fonts/wqy-microhei.xml" kerning="yes" embed-url="./fop/
fonts/wqy-microhei.ttc"> 1
          <font-triplet name="WenQuanYi Micro Hei" style="normal" weight="normal"/> 2
        </font>

        <font kerning="yes" embed-url="./fop/fonts/gbsnOOlp.ttf">
          <font-triplet name="AR PL SungtiL GB" style="normal" weight="normal"/>
        </font>

        <font kerning="yes" embed-url="./fop/fonts/gkai00mp.ttf">
          <font-triplet name="AR PL KaitiM GB" style="normal" weight="normal"/>
        </font>

      </fonts>
    </renderer>
  </renderers>
</fop>
```

1 这里指定中文字体文件所在目录。如果你使用相对路径的话这个路径就相对于你运行a2x时所在的目录。

2 这里的意思是当使用WenQuanYi Micro Hei这个字体时，选用字体文件./fop/fonts/wqy-microhei.ttc。

经过这一番折腾后你应该就能生成中文pdf了：

例 2.5. 中文pdf的生成

```
a2x -v -L -f pdf --fop --xsl-file=xsl/fo.xsl --fop-opts="-c fop/fop.xconf" book.txt 1
```

1 需要用--xsl-file参数给出xsl文件所在位置，用--fop-opts参数给出fop.xconf所在位置。

AsciiDoc的中文支持目前还不完善，这不是它的错，主要问题处出在FOP后端对字体的处理方式上。如果你需要生成粗体或斜体的话可以参考[fop_faq]或[fop_chinese]。

第 3 章 AsciiDoc基础

在这一章里我会介绍一些用AsciiDoc编写文档所需用到的最基本的语法，整个教程我都会以写书为例，这里也不例外。看完本章后你应该就能写出一本非常朴素的书了！

3.1. 章节

写书前当然要先列个提纲，拟个目录，再往里面慢慢充实内容。先来看看一本书通常会包含哪些结构：

例 3.1. 一本书的骨架

```
书名
====
作者名
v0.1, 2012-12
:doctype: book 1

[preface]
前言标题
-------- 2
前言前言前言……

第一章标题
----------
内容内容内容……

第二章标题
----------
内容内容内容……

[appendix]
附录标题
--------
内容内容内容……

[bibliography]
参考文献标题
------------
内容内容内容……

[index]
索引标题
--------
```

1 这里的每一个章节其实就是AsciiDoc里面的1级section，当我们将doctype设为book的时候1级section在语义上就成为书本里的章节了，而0级section就成为了书本的题目。

2 注意标题标示的长度应与标题文字的长度一致。

我们写书的时候每一章下面还会有子章节。AsciiDoc最多支持4级子目录，每一级的目录标题都需要用不同的语法来标示（废话……）。请看下面的示例：

例 3.2. 章节的嵌套

```
第二章标题
----------
内容……

第二章二级标题
~~~~~~~~~~~~~~~
内容……

第二章三级标题
^^^^^^^^^^^^^^^
内容……

第二章四级标题
+++++++++++++++
内容……
```

或许有人会觉到写个标题还要用另一整行来标示，好麻烦呀。其实AsciiDoc支持两种标题的标记语法。一种是双行标题（Two line title），另一种是单行标题（One line title）。双行标题就是我们前面所用的那种，而单行标题则允许用户将标题的定义与内容写在一行里面：

例 3.3. 单行标题

```
= 书书名 =
== 第一章标题 ==
=== 第一章二级标题 ===
==== 第一章三级标题 ====
===== 第一章四级标题 ======
```

在写书时具体使用哪种格式则纯粹是个人喜好了。如果你想再偷懒一点，单行标题右边的“=”也是可以省略的。

3.2. 段落

AsciiDoc非常轻量，以至于它对于段落这种结构没有特别的标记语法。如果硬要说有的话那段落的标记就是 空行 。每一个段落在AsciiDoc中用空行来分隔，就这么简单。不过注意的是每一段的开头都要在左边顶格写，如果有空格的话就成另一种标记了，这个在第 5.2.1 节 “literal段落”中会有专门的介绍。

3.3. 字体样式

这里介绍几个常用的字体样式，它们是：

粗体 ：

```
*粗体* 1
```

1 如果加粗字体两旁有其他中文字体的话，需要用空格隔开，英文字体则无此问题。

斜体 ：

```
'斜体'
```

下划线 ：

```
[underline]#下划线#
```

大号 ：

```
[big]#大号#
```

有色字体 ：

```
[red]#有色字体#
```

需要注意的是中文文档的pdf的输出还不支持这些字体属性。这跟fop对字体处理的方式有关系。目前我还没找到完美的解决方案，只有一些取巧的办法，详见第 2 章 工具链的使用。

第 4 章 写书常用的元素

4.1. 脚注

AsciiDoc为脚注提供了三种相关的语法:

例 4.1. 脚注代码

```
只给定脚注内容footnote:[脚注示例1]。

给定脚注内容和脚注IDfootnoteref:[ft_ex, 含ID的脚注示例]。

对给定ID的脚注的引用footnoteref:[ft_ex]。
```

例 4.2. 脚注效果

只给定脚注内容[1]。

给定脚注内容和脚注ID[2]。

对给定ID的脚注的引用[2]。

4.2. 索引

AsciiDoc有两种生成索引的方式:

例 4.3. 索引代码

```
indexterm:[隐式一级索引(1), 隐式二级索引(1), 隐式三级索引(1)]
隐式索引在原文中不会显式。
(((隐式一级索引(2), 隐式二级索引(2), 隐式三级索引(2)))
indexterm:[隐式一级索引(3)] 1

((显式索引(2))) 2
```

1 二级索引和三级索引都是可以省略的。
2 显式索引的内容会同时出现在原文中，而隐式则不会。

例 4.4. 索引效果

隐式索引在原文中不会显式。

显式索引(1)

索引条目效果请参考本文档的“索引”。

4.3. 内部引用

写书时常常需要引用书本其他部分的文字，使用AsciiDoc的内部引用就能很方便的管理这些引用，只要改一个地方，所有引用都会跟着变动，不需要人工干预。

[1] 脚注示例1
[2] 含ID的脚注示例

内部引用的对象是id，你可以在文档的任何部位创建引用id，然后在任何部位对这些id进行引用。创建标记使用“[[]]”，引用则使用“<<>>”。

例 4.5. 引用代码

```
[[ref_exp, 自定义标记标签]] 1

<<ref_exp>> 2

<<ref_exp, 自定义引用标题>> 3
```

1 ref_exp就是id，后面的自定义标记标签可以省略，AsciiDoc会自动根据当前的位置生成标签。
2 引用时可以只给出id，这样AsciiDoc会自动当前位置显示自定义或系统生成的标签。
3 也可以在引用的时候指定要显示的内容。

例 4.6. 引用效果

自定义标记标签

自定义引用标题

引用也有第二种表示方式，例如：

```
anchor:ref_exp[自定义标记标签]

xref:ref_exp[自定义引用标题]
```

4.4. 附录，索引与参考目录的撰写

在第 3.1 节 “章节”中我们提到过，AsciiDoc以章节（更准确的说是section）来对书本进行划分。但是我们平时写书的时候还会涉及到特殊的章节，例如附录，索引表，参考目录等等。这些部分应该是与章节（或者说section）是同级关系。为了能够方便的编写这一类特殊的章节，AsciiDoc引入了“section标记模板”这个概念。

你可以把section标记模板理解为一个特殊的章节（section），你给定一个模板的名字，然后这个章节里面的内容就会根据对应的模板来进行渲染。

AsciiDoc目前支持的模板有：

- abstract
- preface
- colopho
- dedicatio
- glossar
- bibliograph
- synopsi
- appendi

- index

使用模板，你能够很方便的创建一个附录：

例 4.7. 附录代码

```
[appendix]
附录示例
========

附录里面也是可以定义子附录的。

附录子目录
----------

当然你还可以继续深入下去。
```

效果可以参考本文档的附录。

定义参考目录也可以使用类似的方法：

例 4.8. 参考目录代码

```
[bibliography]
参考文献目录
============

参考文献目录可以分多个区域：

[bibliography] 1
.书籍 2
- [[[taoup]]] Eric Steven Raymond. 'The Art of Unix 3
  Programming'. Addison-Wesley. ISBN 0-13-142901-9.

[bibliography]
.论文
- [[[abc2003]]] Gall Anonim. 'An article', Whatever. 2003.
```

1 2 每个区域以[bibliography]开头，并使用块标题来指定区域的题目。关于块标题，请参考第 5.4 节“块标题”。

3 引用条目的格式相对于BibTex或Docbook来说要简单很多，AsciiDoc的条目直接用列表表示。通常会给前面加个id，方便在正文内部进行引用。

索引表目的撰写则最为简单：

例 4.9. 索引表目代码

```
[index]
索引标题 1
--------
```

1 只需给定标题即可，其他什么都不用写，系统会自动生成索引的条目。

第 5 章 AsciiDoc常用标记介绍

这一章全是些很无聊的东西，扫一眼留个映象即可。以后用到时可以再查阅官方文档，那里有更专业的介绍。

5.1. 列表

列表是我平时写文档时最常用的标记，顾名思义，它用于表现你要逐点罗列的条目。其实就是HTML中的li标签。

列表主要分四种，分别适用于不同的场景：

- 无序列表
- 有序列表
- 标签列表
- callout列表

5.1.1. 无序列表

无序列表的每一个条目以某个无任何含义的符号（默认是圆点）开头，语法如下：

例 5.1. 无序列表语法

```
- 社会主义好
- 自由软件好
```

无序列表的显示效果如下：

例 5.2. 无序列表效果

- 社会主义好
- 自由软件好

列表是支持多级嵌套的，当你要表示多级列表时，请使用如下语法：

例 5.3. 无序列表的多级嵌套

```
- 第一级 1
* 第二级 2
** 第三级
*** 第四级
**** 第五级
***** 第六级 3
```

1 最高一级用“-”表示。
2 接下来用一个或多个“*”表示。
3 最多支持5个“*”，也就是6级嵌套。

5.1.2. 有序列表

有序列表的每一个条目都以一个序号开头，并按照序号的增减顺序排列，语法如下：

例 5.4. 有序列表语法

```
2. 阿拉伯数字作为序号（显式）1
a. 小写字母作为序号（显式）
a. 小写字母作为序号（显式）2
F. 大写字母作为序号（显式）3
ii) 小写罗马字母作为序号（显式）
IX) 大写罗马字母作为序号（显式）


. 阿拉伯数字作为序号（隐式）4
.. 小写字母作为序号（隐式）
.. 小写字母作为序号（隐式）
... 小写罗马字母作为序号（隐式）
.... 大写字母作为序号（隐式）
..... 大写罗马字母作为序号（隐式）
```

1 显式语法在每个条目的开头指定序号的类型和起始大小。
4 隐式语法不指定序号的类型和大小，而将这些交给AsciiDoc来决定。
2 3 每当序号类型改变时，条目就会自动缩进到更低的一级。

显式效果如下：

例 5.5. 有序列表效果

1.阿拉伯数字作为序号（显式）

a.小写字母作为序号（显式）

b.小写字母作为序号（显式）

A.大写字母作为序号（显式）

i.小写罗马字母作为序号（显式）

I.大写罗马字母作为序号（显式）

1.阿拉伯数字作为序号（隐式）

a.小写字母作为序号（隐式）

b.小写字母作为序号（隐式）

i.小写罗马字母作为序号（隐式）

A.大写字母作为序号（隐式）

I.大写罗马字母作为序号（隐式）

5.1.3. 标签列表

标签列表的每一个条目都以用户指定的标签开头，有点像HTML里面的dt标签。

例 5.6. 标签列表代码

```
定义1:: 1
  “定义1”是……
  定义1.1:::
    “定义1.1”是……
    定义1.1.1::::
      “定义1.1.1”是……
定义2::
  “定义2”是……
  定义2.2;; 2
    “定义2.2”是……
```

1 2 可以使用两个，三个，四个“:”或两个“;”进行标签列表条目的标记。

效果如下：

例 5.7. 标签列表效果

定义1
“定义1”是……

定义1.1
“定义1.1”是……

定义1.1.1
“定义1.1.1”是……

定义2
“定义2”是……

定义2.2
“定义2.2”是……

5.1.4. Callout列表

Callout我还真不知道怎么翻译比较好，呵呵。不过这个东东在分析代码的时候非常有用！你几乎可以在任何用DocBook写的技术书籍里面看到它的身影。我们来看个例子：

例 5.8. Callout列表代码

```
  ......................
  - 第一级 <1>
  * 第二级 <2>
  ** 第三级
  *** 第四级
  **** 第五级
  ***** 第六级 <3>
  ......................

  <1> 最高一级用“-”表示。
  <2> 接下来用一个或多个“*”表示。
  <3> 最多支持5个“*”，也就是6级嵌套。
```

效果请参考例 5.3 “无序列表的多级嵌套”。

5.2. 特殊段落

我们在第 3.2 节 “段落”中已经介绍过普通的段落了，AsciiDoc还支持其他特殊的段落，如引用段落，提示段落，警告段落等等。

5.2.1. literal段落

被标记为literal段落的文字会被原封不动的显式出来，AsciiDoc不会对里面的特殊标记进行解析。语法很简单，只需将整段文字缩进一格即可：

例 5.9. literal段落语法

```
 这行文字不会被解析为标题
 ========================
```

显示效果如下：

例 5.10. literal段落效果

```
这行文字不会被解析为标题
========================
```

5.2.2. 引用段落

你可以使用引用段落将你所引用的内容凸显出来，语法如下：

例 5.11. 引用段落代码

```
[quote, GNU, from The Free Software Definition] 1
“Free software” is a matter of liberty, not price. To understand the concept, you should
 think of “free” as in “free speech,” not as in “free beer.”
```

1 第一个quote指定引用的类型，当你引用诗句的时候这里可以填写verse;第二个填的是作者;第三个填的是来源。

例 5.12. 引用段落效果

> “Free software” is a matter of liberty, not price. To understand the concept, you should think of “free” as in “free speech,” not as in “free beer.”
>
> — GNU from The Free Software Definition

5.2.3. 警示段落

写书的时候经常需要在某个地方打上一段显眼的文字以提醒读者注意某些重要的内容，这是可以将这段文字设为警示段落，请看下面的例子：

例 5.13. 警示段落代码

```
WARNING: 师兄出没，注意！

TIP: 防火防盗防师兄。

CAUTION: 小心湿滑。

IMPORTANT: 明天要按时交周报告了！

NOTE: 我还是单身哦。
```

例 5.14. 警示段落效果

警告

师兄出没，注意！

提示

防火防盗防师兄。

小心

小心湿滑。

重要

明天要按时交周报告了！

注意

我还是单身哦。

你也可以使用属性元素来标记:

```
[WARNING]
师兄出没，注意！

[TIP]
防火防盗防师兄。
```

5.3. 有界块元素

顾名思义，有界块元素就是使用特殊符号作为分界线所画出的某一块用于表示特殊内容的区域。AsciiDoc支持各种各样的块结构，善用这些元素能使得写出来的文档可读性更强，更生动。

5.3.1. 示例块

例 5.15. 示例块代码

```
.示例标题
============================
这是一个示例
============================
```

示例块效果如下：

例 5.16. 示例标题

这是一个示例

5.3.2. 注释块

例 5.17. 注释块代码

```
//////////////////////////
注释内容会被AsciiDoc忽略
//////////////////////////
```

5.3.3. 引用块

引用块的效果与例 5.12 “引用段落效果”的效果一样。它适用于多段文字的引用。

例 5.18. 引用块代码

```
[quote, 啄木鸟社区, 我们的奋起宣言]
_____________________________________ 1

每日至少抽一刻钟,解答邮件列表中初学者的问题,

每周至少抽两小时,整理新学知识将体验发表/分享出去,

通过Blog/Wiki/邮件列表/个人网站……

每旬至少抽四个小时, 来翻译自个儿喜爱的自由软件的文档,

每月至少抽八小时, 快乐的编程, 推进自个儿的项目,

每年至少参加一次, 自由软件的活动, 传播自由软件思想,

发展一名“自由人”……

只要我们每个人都坚持下去……

10年！就足以改变中国软件的整体风貌！
_____________________________________
```

1 这里所使用的字符是“_”而不是“-”。

例 5.19. 引用块效果

> 每日至少抽一刻钟,解答邮件列表中初学者的问题,
>
> 每周至少抽两小时,整理新学知识将体验发表/分享出去,
>
> 通过Blog/Wiki/邮件列表/个人网站……
>
> 每旬至少抽四个小时，来翻译自个儿喜爱的自由软件的文档,
>
> 每月至少抽八小时，快乐的编程，推进自个儿的项目,
>
> 每年至少参加一次，自由软件的活动，传播自由软件思想,
>
> 发展一名“自由人”……
>
> 只要我们每个人都坚持下去……
>
> 10年！就足以改变中国软件的整体风貌！
>
> — 啄木鸟社区 我们的奋起宣言

5.4. 块标题

块标题用于给块元素添加标题，例如段落、有界块元素、列表、表格和图片等等。

例 5.20. 块标题代码

```
.块标题示例 1
- 条目1 2
- 条目2
```

1 标题以“.”打头。
2 块元素要紧跟它的标题。

例 5.21. 块标题效果

块标题示例

- 条目1
- 条目2

5.5. 表格

例 5.22. 表格代码

```
.表格示例 1
[width="80%",options="header"] 2
|========================================== 3
| 发行版          | 特性  4
| Ubuntu          | 最好的用户体验，最广的用户群
| Fedora          | 最潮的发行版，新技术的实验田
| Arch            | 轻量，灵活，精简
| Debian          | 最自由，最通用的发行版
| Gentoo          | 高定制性，性能优化的极限
|========================================== 5
```

1 表格的标题，可以参看第 5.4 节 “块标题”。
2 这里可以设置表格的属性，header表示第一行为表格标题，你也可以设置footer。
3 5 表格以“=”开始和结束，注意每一行前面都要加上“|”。
4 表格的每一行就是一行文字，每一列用“|”隔开。

上面代码的显示效果如下：

例 5.23. 表格效果

表 5.1. 表格示例

发行版	特性
Ubuntu	最好的用户体验，最广的用户群
Fedora	最潮的发行版，新技术的实验田
Arch	轻量，灵活，精简
Debian	最自由，最通用的发行版
Gentoo	高定制性，性能优化的极限

5.6. 图片的插入

图片有两种插入方式，一种是内联（inline）模式，另一种是块（block）模式。内联模式用于在文字中间插入图片，而块模式则会另起一行将图片显示出来，并独占它的横向空间。

例 5.24. 图片代码

```
内联模式的“图 image:./img/gnu-linux.jpg[height=20] 片”可以显示在文字的中间。 1

块模式的效果：

image::./img/meditate.jpg[title="Levitating GNU",scaledwidth="50%",align="center"]
```

1 注意图片插入命令两边要有空格。

渲染效果如下：

例 5.25. 图片效果

内联模式的“图  片”可以显示在文字的中间。

块模式的效果：

图 5.1. Levitating GNU

第 6 章 提高效率的小技巧

6.1. 文档的组织

通常写书时，不建议将所有内容写在一个文件里面。而是将书以章节为单位进行划分。

将每个章节分别写到不同的文件中至少有如下好处：

- 你不用为了查看某一章节的改动而重新编译整本书。
- 方便查看，你只需要到相应的文件中找你要找的内容，而不需要检索整本书。
- 方便编辑，修改相应章节时只需编辑相应的文件。
- 方便版本管理，每一章节有自己的commit历史。

你可以通过以下方式在AsciiDoc中插入其他AsciiDoc文档：

例 6.1. AsciiDoc文档的插入

```
include::chapter1.txt[]
```

这样，chapter1.txt的内容就被插入到了当前位置。接下来，我们只需将每个章节写到不同的文件中，然后在主文档中include各个章节即可。

为了使得每个章节能够单独编译，我通常会将章节的标题定为0级section（关于section请参考“一本书的骨架”的注释）。但是这样直接include的时候一本书就会出现多个0级section的定义，导致语法错误。所以我们需要使用下面的办法来将所include的内容的标题上调一级：

例 6.2. leveloffset的设定

```
:leveloffset: 1 1

include::chapter1.txt[]

:leveloffset: 0 2
```

1 将leveloffset设为1后，后面出现的section都会上调一级，于是0级标题就变成1级标题（也就是章节标题）了。你也可以根据需要将它设为2、3或4。

2 如果后面还有内容的话记得将leveloffset设回0。

附录 A. 附录示例

附录里面也是可以定义子附录的。

A.1. 附录子目录

当然你还可以继续深入下去。

参考文献目录

- [asciidoc_guide] Stuart Rackham. AsciiDoc User Guide.
- [pdf_backend] Stuart Rackham. AsciiDoc User Guide - 5.4. PDF generation. http://www.methods.co.nz/asciidoc/userguide.html#_pdf_generation
- [walsh-muellner] Norman Walsh & Leonard Muellner. DocBook - The Definitive Guide. O'Reilly & Associates. 1999.
- [bob-stayton] Bob Stayton. DocBook XSL: The Complete Guide. Sagehill Enterprises. 2007
- [docbook_chinese] 如何用docbook生成中文的pdf. http://dev.sei.pku.edu.cn/trac/pkuas/wiki/如何用docbook生成中文的pdf
- [fop_faq] 利用HTML生成PDF之问题整理. http://salever.iteye.com/blog/860611
- [fop_chinese] fop中文配置. http://wangxc.iteye.com/blog/598912
- [asciidoc_chinese] asciidoc中文PDF输出. http://dram.me/blog/2011/03/09/export-chinese-pdf-from-asciidoc.html

索引

符号

S

版权