Printable Version (PDF) of this Blog

http://freemind.pluskid.org/misc/printable-version-pdf-of-this-blog

posted on Free Mind on June 16, 2012
generated with pandoc on December 3, 2015
category: MISC

tags: LaTeX, Tools

从现在开始这个 blog 上的文章将提供可打印的版本下载，如果文章页面的右上角有 PDF 图标出现，那么点击该图标那里的链接就可以下载到文章的 PDF 版本，例如本文就是有 PDF 版本可供下载的。PDF 和 Web 版本各有优缺点这是不争的事实，我就不在这里多比较了。

PDF 版本并不是直接通过 HTML 版本转换得来的（否则还不如直接在浏览器里打印为 PDF 文件得了）。实际上我考虑过各种方案，最后总结了一下自己发 blog 的需求：好像我从新建这个 blog 以来发表 blog 的频率还是比较高的，而且各种类型的文章也都有，这实际上是有组织有预谋的 `XD`，通过尝试各种类型的文章，我找出了自己发 blog 的时候最需要的一些功能：包括数学公式、页内的公式和图片编号引用、参考文献引用、代码高亮、图片（包括左右 float 的图片以及正常的带 caption 的图片和几张图片并排显示的那种 thumbnail 形式等），还有就是一些基本的格式如 blockquote、超链接、强调、列表之类的。

除了代码高亮之外其他的都是 TeX 比较擅长的，但是我又不能直接用 TeX 来写 blog ，一来那样太麻烦，二来从 TeX 转换到 HTML 并没有什么特别好用的工具，ConTeXt 倒是可以转换为 xml 文档，不过似乎没有什么可定制性，要把 xml 弄成最终合式的 HTML 也许也够呛。所以最后我没有选择这种方案。另一方面，HTML 和主流的各种标记语言像 markdown、textile、reStructured Text 等几乎都没有页内公式编号和引用功能，虽然图片是支持的，但是一般也是最简单的显示单张图片，多张图片位置摆放、caption 以及浮动等功能都不好弄。我之前的解决方案是使用 textile 和 Mastache template 混合，自己手工来处理像公式、复杂图片摆放和页内引用等功能。

后来想想虽然我的有些需求看起来比较复杂一点，但是总的功能需求还是很精简的，反正也要混合几种工具来处理，不如自己山寨一个统一的工具，只要实现我那几种功能就好了，然后同时提供 HTML 和 TeX 的渲染输出，就很方便了。于是就有了TeX Markup Language (TeXML) 这个东西，花了一天多的时间，在将 parser 实现出来以及基本的语法和可扩展接口弄好之后，我突然决定不干了……`=.=bb`。因为这似乎是个无底洞，自己山寨这些东西始终是不好的，如果哪天出了什么问题之类的还要自己费力去 fix……总而言之这个东西很快地诞生然后又很快的废弃了。不过项目的页面还留着的，至少可以作为使用 Ruby 的 Treetop 库来写一个 parser 的例子还是可以作为参考的。

结果呢，我最终的解决方案是使用 pandoc，这货是一个各种文档格式之间互转的神器，支持各种标记语言，HTML 和 LaTeX 这些复杂语言，甚至还有微软的 docx 格式（实际上是 xml）和 OpenOffice 的 ODT

等。更神奇的是这家伙居然是 Haskell 写的，真想不到有一天我居然会实实在在用一个 Haskell 程序，想起了每周和 MSTC 的大家在 YQ B004 读《Real World Haskell》的~~痛苦~~欢乐情形，最后我承诺走之前给会给大家做的关于 Monad 的 presentation 也最终未能实现，不知道这件事以后会不会有一个结局呢？`:D`

回到本文的正题，pandoc 提供了一种扩展的 markdown 语法，可以支持公式和参考文献等功能，当然最关键的还是它能很好地同时转换为 HTML 和 TeX。并且这里有一个选项是转换为 ConTeXt 格式。这个东西我很早就听过，但是一直没有机会真正尝试过，这下子突然激起了我的兴趣（莫非我最近非常无聊？`=.=bb`），于是就做了一些尝试。

同样是基于 TeX 的，ConTeXt 和 LaTeX 相比应该各有侧重点吧，LaTeX 的卖点应该是用户只要关注内容就好了，例如，写一篇论文什么的，只要按照章、节等填好内容，剩下的就能自动排版为一个漂亮的文档；而 ConTeXt 应该是更加侧重于排版的功能，提供了各种高级的排版控制功能。ConTeXt 和 LuaTeX 接合起来似乎可控制性和易操作性都会相对强一些。我倒是没有考究过历史，不知道是不是 ConTeXt 比 LaTeX 起源要晚一些，反正 ConTeXt 整体看起来要清爽一些，不像 LaTeX 那么巨无霸的样子各种功能都有 N 多宏包来实现而各个宏包有时候又互相冲突。当然也有可能是 ConTeXt 本身用户量也相对小的缘故吧。

使用起来也有它的方便之处，例如图文混排的功能直接就能实现，而在 LaTeX 里图片、子图、图文混排等的宏包可以说有无数个（有一本专门讲 LaTeX 图片排版的书），而且互相之间还有和其他一些宏包之间时常会有一些冲突。另外 ConTeXt 也是原生支持 unicode 以及中文排版断行、还有 TTF、OTF 等字体，不过倒是现在 LaTeX 使用 XeLaTeX 也可以很好地处理这些。然后 ConTeXt 还直接支持 svg 图，LaTeX 里就不行。

不过最终有一个很严重的问题困扰我：就是公式。ConTeXt 当然也能排版出来很漂亮的公式，但是稍微复杂一点的公式就必须使用 plain TeX 以上的更高级的语法，于是就和 LaTeX 的语法出现了分歧。两者相比 LaTeX 在公式排版方面有很多宏包可以用，并且最难能可贵的是有很好的文档可以参考。ConTeXt 的语法设计更加 clean 一些，而且许多功能都已经内置了，但是没有文档比较痛苦。最关键的还是 ConTeXt 的语法 MathJaX 是不认的：这样一来我为文档写的公式就无法直接用 MathJaX 来渲染了。一种解决方案是像以前那样用图片来显示公式，用 ConTeXt 把公式片断编译成图片，但是这个方法颇麻烦；另一个方法就是使用 ConTeXt 导出 xml 的功能，它会把公式变成 MathML，MathJaX 是支持显示 MathML 的，不过同样这个要自己做好多工作。所以最终我还是放弃 ConTeXt 了。

不过对 ConTeXt 的印象还是蛮好的，以后碰到合适的场合大概还是会再尝试吧。大概比较大的问题就是像其他的小众开源软件那样，相关的文档啊教程啊之类的相对比较难找吧。哦，对了，之所以难找还有另一个重要的原因，也是我认为的 ConTeXt 最大的缺陷：和 Google Go 类似

的，这个名字太大众化了！想像一下我要在 Google 里搜索 context 会得到些什么结果？

最终的解决方案是 pandoc 分别到 LaTeX 和 HTML 的转换，为了处理页内引用和复杂图片排版，以及 HTML 和 TeX 不同的语法高亮支持，仍然使用 Mastache 的 template 扩展。另外 pandoc 虽然支持 bibtex 的引用，但是不是很好用，我也通过自己的扩展来实现。再就是非常强迫症地花了不少时间来给 TeX 文档做了一个格式主题。最终结果就是这样了，把之前的几篇文章也做了格式上的转换，最终 HTML 渲染的结果没有太大的变化，TeX 得到的 PDF 也挺漂亮，自己还算满意了。荒废了好几天的正事，得赶紧再回到正轨了！