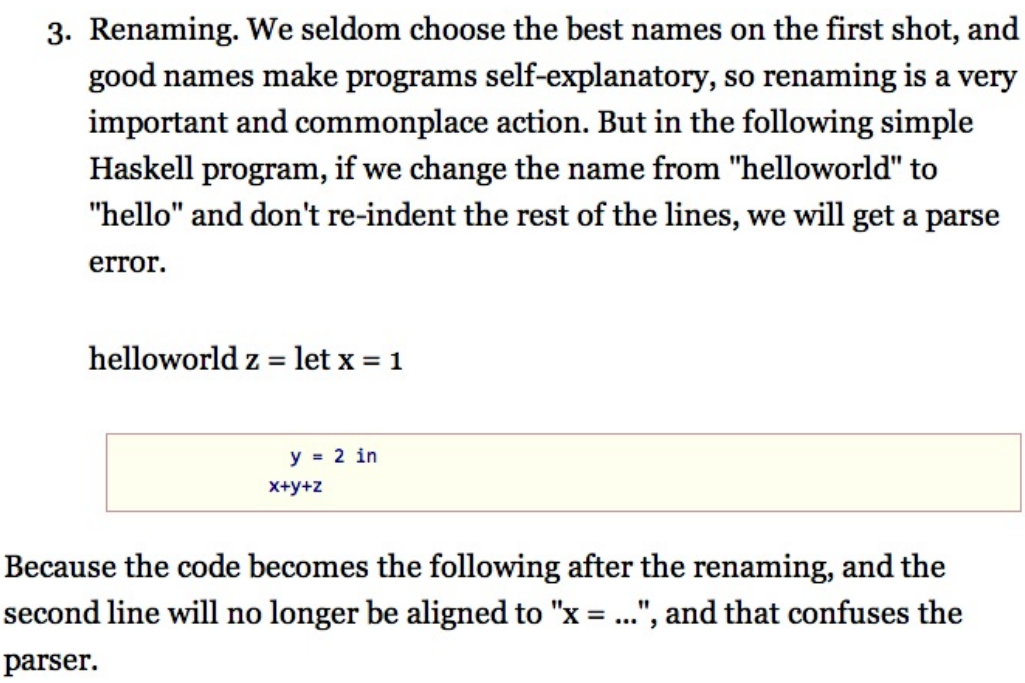
Markdown 的一些问题

把我之前的博文基本上转换成了 markdown 格式。我发现 markdown 虽然在编辑器里看起来比 HTML 清晰一些，但也有一些不足。

这些 markup 语言的格式都有点像我本科的时候给我爸做的一种“标准化试卷标记语言”（因为他是中学英语老师）。当时我写了一个1000来行的 Perl 脚本，可以把这种简单的标记语言转换成美观的 LaTeX 格式文档，并且带有友好的 Tk 图形界面。现在回想起来，我那时候的设计就已经相当先进了。跟我的语言相比，这些 blog 用的 markup 语言真是小巫见大巫了，而且问题多多。有点跑题了，还是回头来看看 markdown 的问题吧。

- Markdown 实际上采用的是类似 Python 和 Haskell 的 layout 语法。

  我已经在一篇英文博文里提到了 layout 语法的多种问题。因为空格的数量决定了文档的结构，这种文档格式相当的“脆弱”。稍微少打一两个空格，就会出现不可预测的结果。这种现象在“itemize”内部的代码块最容易出现。因为每个 item 带来了缩进，所以内部的代码必须比 item 的缩进多4个空格，才能被排到正确的位置。比如我转换博文的时候多次出现以下的情况：

  3. Renaming. We seldom choose the best names on the first shot, and good names make programs self-explanatory, so renaming is a very important and commonplace action. But in the following simple Haskell program, if we change the name from "helloworld" to "hello" and don't re-indent the rest of the lines, we will get a parse error.

     helloworld z = let x = 1

     ```
                 y = 2 in
     x+y+z
     ```

     Because the code becomes the following after the renaming, and the second line will no longer be aligned to "x = ...", and that confuses the parser.

  这里的问题是，代码里的第一行 `helloworld z = let x = 1` 因为缩进不够，被放到了代码块外面。但是为了准确的缩进所耗费的精力，其实比直接打 `<pre>` 这样的 tag 还要多。

- 特殊字符的选择不合理

  markdown 对特殊字符的使用不大合理。我多次发现文档段落整段的变成斜体，就是因为原来的文档里出现了 `x*y` 这样的表达式。在程序员的世界里，“乘法”显然比“强调”更加频繁。把 * 用于标记“强调”，实际上把一个非常有用的字符用在了很不频繁的用途。

- 表达力相当有限

  在很多细节上，markdown 并不能表达我想要的格式。比如它不能正确的插入断行 `<br>`。如果你有两块紧接在一起的代码，但你不想把它们连在一起，markdown 非要给你连在一起…… 于是我就发现自己加入了越来越多的 HTML。

  这在图片的语法上就更加明显，markdown 引入了 `![alt](image url)` 这样的格式，其实比起 HTML 还要难看和不一致。比如现在它仍然无法表达图片的大小，这是相当重要的信息。所以我觉得 markdown 的语法已经显示出了它的弱点，如果它要表达更复杂的信息，就会变得比 HTML 还要难记，难看。所以对于图片，我觉得还不如直接用 HTML 的 `<img>` 。

所以总的感觉是 markdown 引入了太多的“语法”，以至于稍微复杂一点的信息表达起来还不如 HTML 来的直接。现在就这样先凑合着吧。也许过段时间自己设计一个格式。