/* code */
Grace

清单式学习 VI 编辑器

制作：手气不错 <i.feelinglucky@gmail.com>
主页：http://www.gracecode.com

摘自：http://www.chinaitpower.com/A/2004-07-31/88248.html
英文原文：http://www.gentoo.org/doc/en/vi-guide.xml

整理日期：2007 年 11 月 28 日

1. 起点

1.1. 介绍

这篇指南要向你介绍 vi 一个强大的可视化编辑器的使用方法。通过一种特殊的循序渐进的清单模式，这篇指南让你在短时间内迅速成为 vi 高手。在这篇指南里面，你将学到如何浏览、编辑文字、使用“插入”模式、复制粘贴以及使用一些重要的 vi 插件（譬如：可视化插件）和多窗口编辑。

如果你不知道，或者对 vi 感到不爽，那么赶快利用这篇指南。让这个 Linux/UNIX 平台最流行、最强大的可视化浏览器为你的工作添油加速吧！

1.2. 关于这篇指南

学习 vi 的最大难点在于 vi 的繁多的指令。为了有效利用 vi，你需要记住其中不少的一些。这可能需要不少时间；而偏偏这篇指南的目的之一就是在于利用尽可能少的时间。所以，从一开始，我们将面对一个挑战：我怎么帮助你在短期内记住大量的命令。

为了在这篇指南的过程中解决这个难题，我们会逐渐一点一点积累起一张关于 vi 的知识清单。这张清单会包含所有重要的 vi 命令。当你完成了这份指南，当你忘记某个 vi 命令的时候，你可以很轻松得从这份清单中查找特定的命令。逐渐地，随着你记住这些命令，你会越来越少依赖这份清单。通过使用“知识清单”技术，你可以前所未有的速度学会使用 vi ！

1.3. 学习过程

在这份指南中，我会使用一些帮助你学习的办法。首先，如同你所期待的，我会介绍一条命令是如何工作的。然后，我会请你在 vi 中练习使用这条命令，之后把这条命令记录在知识清单中，以备日后参考。一一执行这些步骤对于你快速学习 vi 有莫大的帮助。试验并且记录这些命令会帮助你记住它们。

1.4. 介绍 Vim

Vi 有很多不同版本，而我要向你介绍的是一个叫做 *vim* 的版本。vim 非常流行；它有相当多的扩展模块使得 vi 更棒（当我展示一个 vim 独有的命令的时候，我会特别指出）。如果你需要安装 vim，你可以从 www.vim.org 得到它。除了一个增强版本的命令行 vi，vim 还有一个叫做 *gvim* 的、漂亮的、基于 GTK+ 的 GUI 版本。下面是一张我系统中 Gvim 的截图：

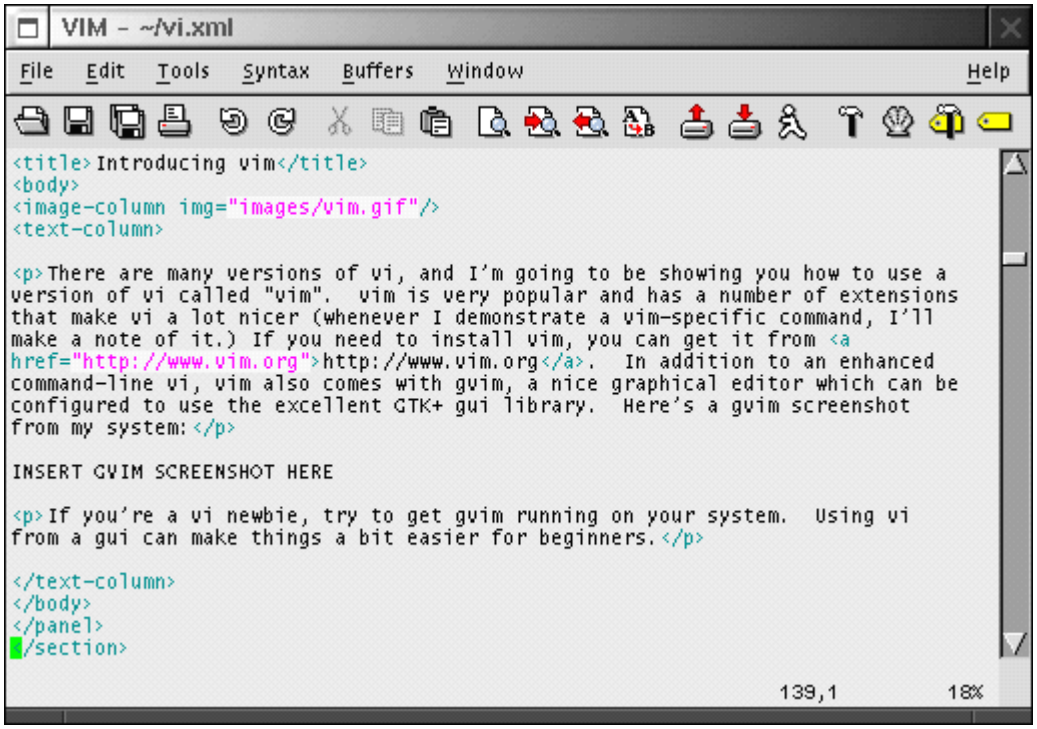

插图 1.1：VIM 截图

如果你是一个 vim 新手，试着在你的系统上运行 gvim，从 GUI 使用可以让新手觉得轻松一点。

2. 第一步

2.1. 打开一个文件

在使用 vi 编辑文件之前，你需要知道如何使用 vi 浏览一个文件。vi 有很多移动命令，而我们会接触它们中的不少。在指南的这部分中，找一个无关紧要的文本文件，然后用 vi 打开它：

代码 2.1: 用 vi 装入一个文件

```
$ vi myfile.txt
```

如果你安装了 vim，输入 "`vim myfile.txt`"。如果你决定使用 gvim，输入 "`gvim myfile.txt`"。myfile.txt 是你系统中的一个文件名。

2.2. 在 vi 中

启动 vi 之后，你应该看到文件的一部分显示在你的显示屏上。恭喜你进入了 vi！不像其它很多编辑器，当 vi 启动的时候会进入一个特别的“*命令模式*”。

这意味着如果你键盘上的 1，vi 不会在当前光标位置输入一个 1，而会将光标向右移动一位。在命令模式，从键盘输入的字符会作为命令传递给 vi，而不是向文件中插入文字。移动命令是所有命令中最基本的一类，让我们来看一看。

3. 来回移动

3.1. 在 vi 中移动，第一部分

在命令模式中，你可以用 **h、j、k** 和 **l** 键将光标分别向**左、下、上**和**右**侧移动。如果你使用的是一个现代版本的 vi，你也可以用方向键达到相同的目的。

h、j、k 和 l 键很有用：当你习惯了它们的时候，你可以不必将你的手从键盘基本键位移开就可以在文件中移动。试着使用 h、j、k 和 l 键（以及方向键）在文件中移动光标。试着用 h 移动到行首。

注意 vi 不允许你用 h 键从行首自动退回到上一行。相似的，你不能用 l 从行末自动转到下一行。

3.2. 在 vi 中移动，第二部分

vi 为跳至行首、尾提供了特殊的快捷键。你可以按 **0（零）跳到行首，$ 跳到行末**。试试看。因为 vi 有这么多的移动命令，我们可以把它作为一个好的“分页”程序（就像 more 或者 less 命令）。把 vi 作为一个分页程序也可以帮助你飞快地学习这些移动命令。

你也可以用 **<CTR>F 和 <CTR>B 向前后一次移动一整页**。新的一些版本（譬如 vim）也支持用 PGUP 和 PGDOWN 键达到相同的目的。

3.3. 按单词移动，第一部分

vi 也允许你按照单词长度来左右移动。要移动到**下一个单词的第一个字母，按 w 键**。**到下一个单词的最后一个字母，按 e 键**。**移动到前一个单词的第一个字母，按 b 键**。试试。

3.4. 按单词移动，第二部分

试过这些移动命令之后，你可能注意到，vi 把 foo-bar-oni 这样的单词作为五个单独的单词！这是因为 vi 默认以*空格或者标点符号*作为单词分界，是以 foo-bar-oni 备分解成五个单词：foo、-、bar、- 和 oni。

有的时候这是你所期望的，有时候不是。幸运的是，vi 能够理解“长单词”的概念。vi 仅仅以空格或换行作为长单词的分界。这意味着虽然 foo-bar-oni 会被当作五个 vi 单词，它也可以作为一个长单词处理。

3.5. 按单词移动，第三部分

你可以使用大写的单词移动命令在长单词之间跳跃移动。**W 跳到下一个长单词的第一个字母，E 跳到最后一个字母，B 移动到前一个单词的第一个字母**。试试看，和普通单词范围的移动进行比较，直到你明白它们的区别。

3.6. 更大范围的移动

在我们把所学过的东西归纳到清单里面之前还有几个命令要看看。你可以**用 (和) 移动到前后两句的开始**。此外，**用 { 或 } 可以跳到当前或者下一个段落的开头**。试试它们。

4. 退出

我们已经说了基本的移动命令，但是还有一些命令是你需要知道的。**输入 ":q" 可以退出 vi**。如果这样不能退出，可能是因为你修改过这个文件。要让 vi **退出而不保存修改，输入 ":q!"**。这样你应该退出到了命令提示符。

在 vi 中，以 : 开头的命令被称为 *ex 模式命令*。这是因为 vi 内置了一个叫做 ex 的非可视化的编辑器。我们可以像使用 sed 一样地使用 ex 进行按行编辑文本。

不仅如此，它也可以用来退出 vi，就像我们刚刚看见的。当你在命令模式**按下 "Q" 键，就可以进入 ex 模式**。这时，你会看见一个提示符 :；如果你按下回车键，整个屏幕会向上滚动。要回到熟悉的 vi 模式，只要输入 vi 然后回车就可以了。

5. 知识清单

5.1. 清单的开头

我们已经提到了很多命令；现在到了把它们加入清单的时候了。你可以用一张 A4 或者 US 信纸来制作这份清单，因为我们会在里面写很多内容！这是一份我的清单，里面包含了上面提到的所有命令。尽可能参考我的格式，这样我们可以把所有东西都集中在一张纸上。

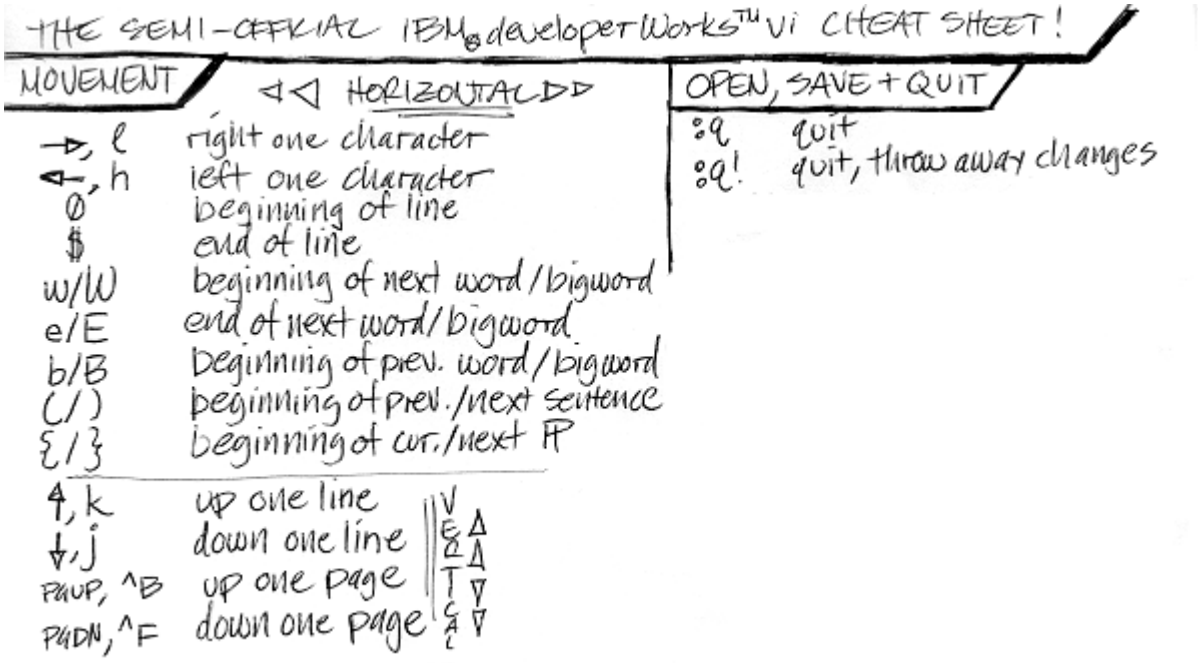

插图 5.1：知识清单

5.2. 繁复的 vi (Miscellaneous vi)

让我们继续我们快速的命令行学习过程。在命令模式你可以**用 G 跳至特定行**。要跳到文件的第一行，输入 1G。注意 G 是大写字母。

如果你要搜索某个特定字符串模式在文件中的位置，输入 "/<regexp>" 然后回车。用一个正则表达式来替换你要搜索的模式。如果你不会使用正则表达式，不用烦恼，直接输入 "/foo" 会移动到下一个 "foo" 出现的地方。唯一需要注意的是，如果你需要寻找字面上的 "^、.、$" 或者 "\"，你需要在它们之前加上一个反斜线 "\"。譬如，"/foo\.gif" 会寻找 "foo.gif" 的下一个出现位置。

要继续向下寻找，按 "n"。要继续向上寻找，输入 "N"。还是一样，用你自己的 vi 编辑器试验这些命令。你也可以用 "//" 来重复上一次的搜索。

6. 保存和编辑

6.1. “保存”和“另存为...”

我们提到过，你可以用 **ex 命令 :q 退出 vi**。如果你要**保存你做的改变，输入 :w**。如果你要保存你的更改后的文本到另外一个文件，**输入 :w filename.txt 会把文本保存为 filename.txt**。如果你想**保存后退出，输入 :x 或者 :wq**。

在 vim （以及其它一些高级的编辑器，例如 elvis）:w，你可以同时打开几个缓存。要**在一个新窗口中打开文件，输入 :sp filename.txt**。filename.txt 会在一个分割开的新窗口中打开以供编辑。**要在窗口之间切换，输入 <CTR>w<CTR>w**（按两次 control-w）。:q、:q!、:w 和 :x 中的任一个都只作用于当前活动的窗口。

6.2. 简单的编辑

现在是我们学习一些简单编辑命令的时候了。我们将要提到的命令被称为“简单”是因为这些命令只需要你呆在命令模式。更复杂的编辑命令会让你自动进入*插入模式*：一个允许你从键盘输入文本的模式。我们会在一会儿提到这些内容。

现在，试着移过一些字符然后反复按 x 键。你会看见 x 会删除当前光标位置的字符。现在移动到你打开的文件的一个段落中间的某处然后点 J（大写）。你会发现 J 命令让 vi 把下一行接在这一行的结尾。现在，把光标移过一段文字然后点 r，再输入一个新的字母，你会看见原来的内容被替换了。最后，移动到文件的任一位置然后输入 dd。你会发现 **dd 会删除当前一行文字**。

6.3. 替换和删除

你可以点 "." 来重复任何一个编辑命令。如果你试试，你会看见 "dd..." 会删除四行文字，而 "J....." 会连接四行。和往常一样，vi 提供了非常顺手的快捷方式。

要删除文本，你可以用 "d" 命令结合任何的移动命令。例如，"dw" 会删除从当前光标位置到下一个单词开始处的内容；**d) 会删除到下一个句子结束的地方**；**d} 会删除这个段落的剩余内容**。试着用 "d" 命令和其它编辑命令直到你熟悉了它们。

6.4. 撤销！

我们正在试验删除命令，而这正是一个学习撤销改变的好时候。按 u 键，传统的 vi 会仅仅允许你撤销最后一次操作。然而，新版本的 vi 譬如 vim 允许你用重复的 u 命令撤销你对文件作出的更改。把 d 和 u 命令一起试试。

6.5. 升级你的知识清单

到了更新你的知识清单的时候了！添加上我们刚刚说过的那些命令，你的知识清单应该看起来是这个样子：

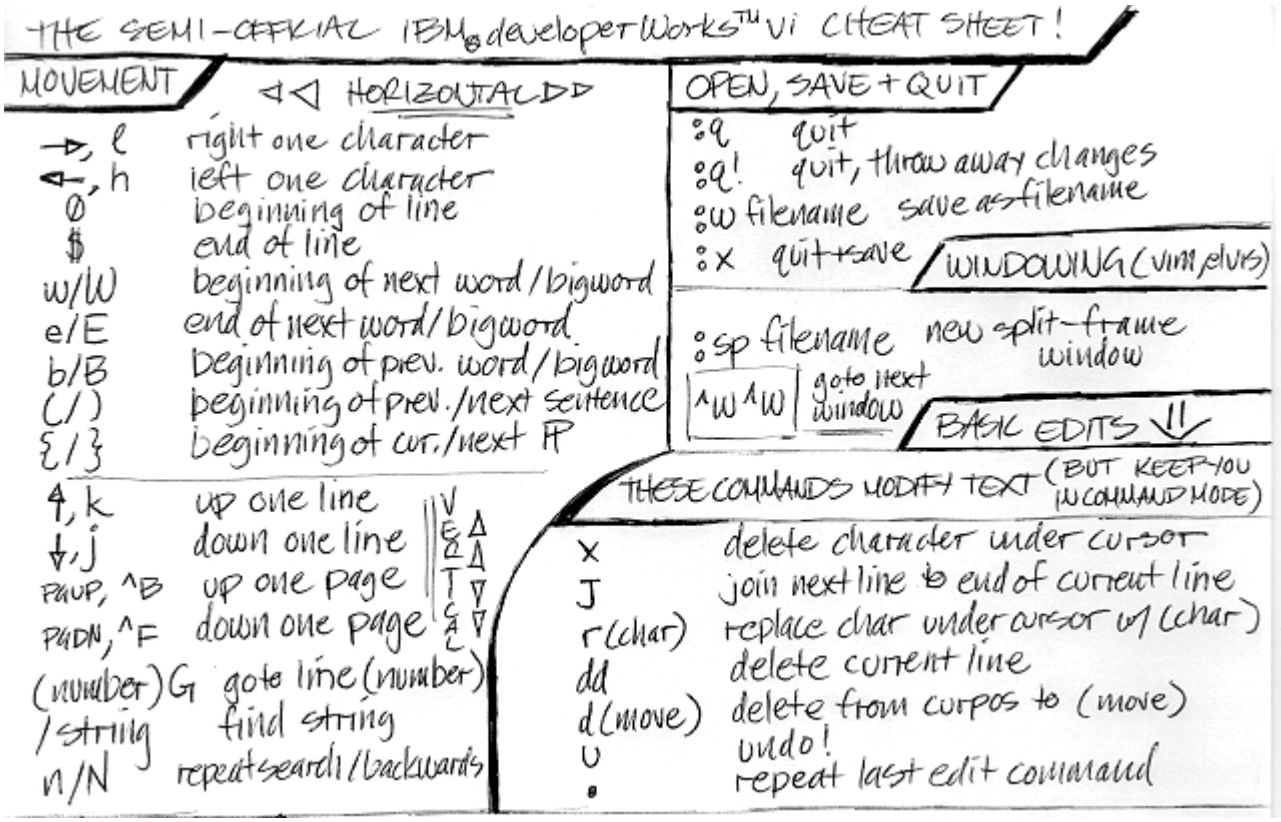

插图 6.5：加入了编辑命令的知识清单

7. 插入模式

到目前为止，我们知道了如何在 vi 中移动光标，进行读写文件，还有一些基本的编辑操作。但是，我还没有向你们展示如何输入自由格式文本！这是我故意的，因为 vi 的插入模式在开始的时候会显得有些复杂。不过，一旦你熟悉了插入模式，它的复杂度（以及可用性）会成为非常有用的东西。

在 vi 的插入模式，你将可以直接将文字输入到屏幕，就像其它很多可视化编辑器一样。当你完成了你输入的内容，可以**用 Esc 键退回到命令模式。i 或 a 可以让你进入插入模式**。如果**按 i，你的文本会插入在当前字符之前；而 a 会让你的文本插入在当前字符之后**。记住，完成输入之后，用 Esc 键回到命令

模式。

7.1. 插入模式的优点

去试试 a 和 i 命令。按其中任意一个，输入一些文字，然后按退出键回到命令模式。按 a 或 i 之后试着按回车键看看会发生什么。试着用方向键和 Del 键感觉一下插入模式怎么工作。利用方向键和 Del 键，你可以不必退出到命令模式而完成相当多的编辑工作。

7.2. 插入选项

这里有其它一些方便的进入插入模式的方法。**按 A（大写）从当前一行的末尾开始插入内容**而不管你当前的位置。相似的，**I（大写）让你在当前行的开始插入。按 o 会在当前一行的下方新建一个空行以供插入**，而 **O 会在上方建立新行**。要**用新的空白行替代当前整行文字，输入 cc**。要**替代从当前位置到当前行的结束，输入 c$**。要**替换从当前位置到行的开始，输入 c0**。

除了上述提到的功能，上面的所有命令都会让你进入 vi 的插入模式。输入完成之后，按 Esc 键回到命令模式。

7.3. 改变文本

我们在前面尝试过 c 修改命令，例如 cc、c0 和 c$。cc 是修改命令的一个特殊形式，类似 dd。c0 和 c$ 命令是组合使用修改命令和移动命令的示例。从这个角度来看，c 和 d 的工作方式相似，除了一点：它让你停留在插入模式。这样，你可以输入用来*替代*删除部分的文本。试着把一些移动命令和 c 结合起来在你的文件中试试看例如：cw、ce、c()。

8. 复合命令

当你开始使用复合（“combo”）命令（类似 d{ 和 cw 之类）的时候，vi 才真正显示它的强大。不仅是这些命令，我们还可以用一个数字与移动命令组合，例如 **3w 会让 vi 向右侧跳过三个单词**。12b、4j 是其它一些数字和移动命令形成的组合命令。

vi，除了支持（数字）（移动命令）组合之外，还允许 d 和 c 与一个数字或者移动命令组合。**d3w 会删除下面三个单词，d2j 会删除当前和下面两行**，等等。试试这些 c 和 d 的组合命令，看看 vi 能让你的文本编辑变得如何简单轻松。当这些命令变得顺手的时候，你将可以以一种难以置信的速度编辑文件。

8.1. 更新知识清单

又到了更新知识清单的时间。它现在看起来应该是这个样子：

插图 8.1：加入了复合命令的知识清单

8.2. 提高生产力的特色

到目前为止，我们已经学会了移动光标、保存和退出、进行简单的编辑和删除操作，以及使用插入模式。利用知识清单上面的内容，你几乎可以用 vi 完成所有的任务。

不过，vi 还有许多更强大的命令。在这个章节中，你将学会如何剪切、复制和粘贴，搜索和替换，以

及使用“自动缩进”等特点。这些命令让 vi 变得更有趣而有用。

8.3. 可视化模式

使用剪贴的最方便的办法是使用可视化模式，一个加入现代版本 vi（譬如 vim 和 elvis）的特殊模式。你可以把可视化模式想象成“高亮文本”模式。当文本被反白，我们可以复制或者删除它，然后粘贴。如果你在使用 gvim，你可以轻松地按住鼠标左键拖动来达到反白一段文字的效果：

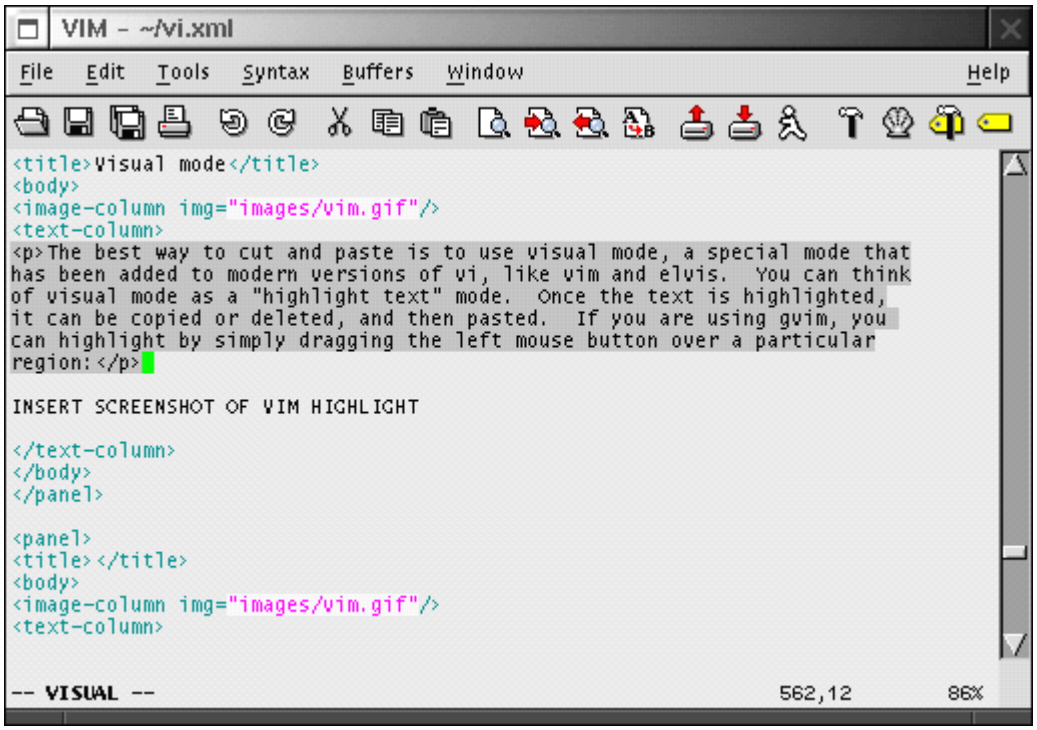

插图 8.2：VIM 与反白文本

除此之外，你也可以通过按 **v** （如果你通过控制台调用 vi，这可能是你唯一的选择）。然后，用移动命令（尤其是方向键）控制光标，你可以反白一段文字。反白之后，我们就可以复制或者剪切这段文字了。

如果要复制这段文字，按 **y**；要剪切，点 **d**。你会就此回到命令模式。现在，移动到你想要粘贴刚刚复制/剪切的内容的地方，然后按 **P** 在光标后插入内容，或者 **p** 将内容插入在光标之前。瞧，剪贴复制，就是这么轻松！

在进入下面一小节内容之前，多试试剪切复制粘贴操作。

8.4. 替换文本

我们用 ex 模式进行文本替换。如果你要替换当前行出现的第一个匹配模式，输入

```
:s/<regexp>/<replacement>/
```

然后按回车。这里 `<regexp>` 是你要匹配的模式，而 `<replacement>` 是你用来替换的字符串。要替换当前行的所有匹配处，输入

```
:s/<regexp>/<replacement>/g
```

然后回车。要替换文件中所有匹配位置（这通常是你需要的），输入

```
:%s/<regexp>/<replacement>/g
```

如果你希望在全局替换过程中，vi 在每次替换之前进行提示，输入

```
:%s/<regexp>/<replacement>/gc
```

（c 表示 confirm 确认）然后回车。

8.5. 缩排

vi 支持自动缩排；当你编辑程序源码的时候可能需要这个功能。多数现代版本的 vi（例如 vim）会在你打开一个源码文件（例如 .c 文件）自动启用自动缩排功能。

当自动缩排启动之后，你可以用 **<CTR>d（control-d）**向左移动一个缩排级别，或者 **<CTR>t（control-t）**向右移动一个缩排级别。

如果自动缩排模式没有自动启用，你可以用 ex 模式命令

```
:set autoindent
```

来启动它。你还可以用

```
:set tabstop
```

命令告诉 vi 你习惯的 tab 宽度；`:set tabstop=4` 是一个相当流行的设置。

8.6. 我们的最终版知识清单

哦也，我们完成了我们的 vi 指南！加入了所有的高级编辑功能之后，这份清单应该看着像这个样子：

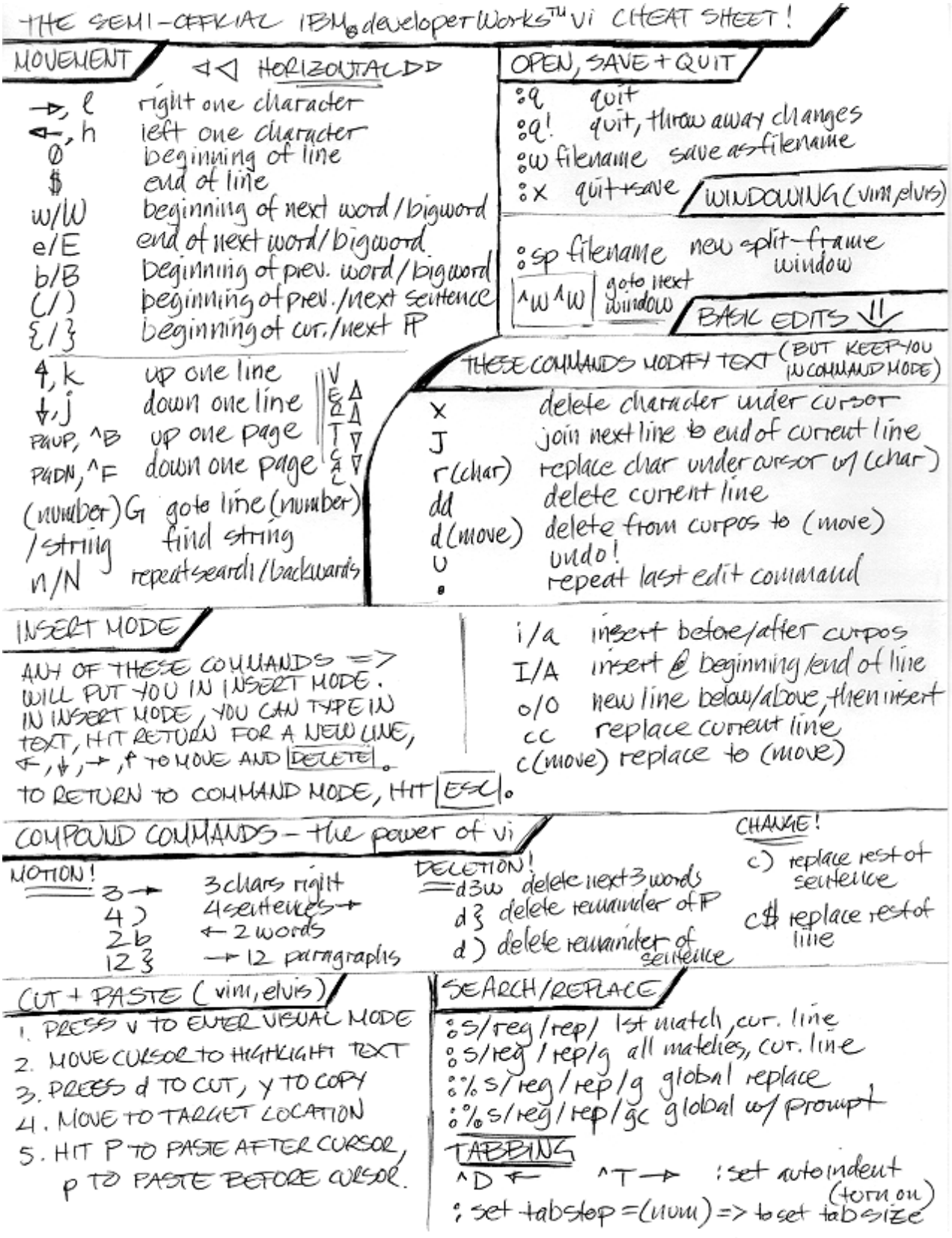
THE SEMI-OFFICIAL IBM developerWorks™ Vi CHEAT SHEET!
MOVEMENT
◁◁ HORIZONTAL ▷▷
→, l right one character
←, h left one character
0 beginning of line
$ end of line
w/W beginning of next word/bigword
e/E end of next word/bigword
b/B beginning of prev. word/bigword
(/) beginning of prev./next sentence
{/} beginning of cur./next ¶
↑, k up one line
↓, j down one line
PAUP, ^B up one page
PADN, ^F down one page
(number)G goto line (number)
/string find string
n/N repeat search fwd/backwards
▽ VERTICAL △
OPEN, SAVE + QUIT
:q quit
:q! quit, throw away changes
:w filename save as filename
:x quit + save
WINDOWING (vim/elvis)
:sp filename new split-frame window
^W ^W goto next window
BASIC EDITS
THESE COMMANDS MODIFY TEXT (BUT KEEP YOU IN COMMAND MODE)
x delete character under cursor
J join next line to end of current line
r(char) replace char under cursor w/ (char)
dd delete current line
d(move) delete from curpos to (move)
u undo!
. repeat last edit command
INSERT MODE
ANY OF THESE COMMANDS => WILL PUT YOU IN INSERT MODE. IN INSERT MODE, YOU CAN TYPE IN TEXT, HIT RETURN FOR A NEW LINE, ←, ↓, →, ↑ TO MOVE AND DELETE. TO RETURN TO COMMAND MODE, HIT ESC.
i/a insert before/after curpos
I/A insert @ beginning/end of line
o/O new line below/above, then insert
cc replace current line
c(move) replace to (move)
COMPOUND COMMANDS – the power of vi
MOTION!
3→ 3 chars right
4) 4 sentences →
2b ← 2 words
12} → 12 paragraphs
DELETION!
d3w delete next 3 words
d} delete remainder of ¶
d) delete remainder of sentence
CHANGE!
c) replace rest of sentence
c$ replace rest of line
CUT + PASTE (vim, elvis)
1. PRESS v TO ENTER VISUAL MODE
2. MOVE CURSOR TO HIGHLIGHT TEXT
3. PRESS d TO CUT, y TO COPY
4. MOVE TO TARGET LOCATION
5. HIT P TO PASTE AFTER CURSOR, p TO PASTE BEFORE CURSOR.
SEARCH/REPLACE
:s/reg/rep/ 1st match, cur. line
:s/reg/rep/g all matches, cur. line
:%s/reg/rep/g global replace
:%s/reg/rep/gc global w/ prompt
TABBING
^D ← ^T → :set autoindent (turn on)
:set tabstop=(num) => to set tabsize

插图 8.6：最终版知识清单

把这份清单放在手边，然后开始用 vi 编辑文档、写 email。当你需要的时候参考这份清单；你会发现在这一周之内，你会记住几乎所有这些命令，而 vi 的生产力会让你飞翔天际！

如果你像把 vi 作为你的默认编辑器，修改你的 `/etc/rc.conf` ，设置 vi 成默认编辑器（设置 EDITOR 为你选择的编辑器）：

```
#EDITOR="/bin/nano"
EDITOR="/usr/bin/vim"
#EDITOR="/usr/bin/emacs"
```

9. 参考资料

这里有一些你可能认为有助于继续学习 vi 的资料：

- vi 爱好者主页，关于 vi 的一切的好地方
- vim 主页 是你寻找 vim 相关信息的地方
- 如果你在找一本好的过时的书，《Learning the vi Editor, 6th Edition》会是一个不错的选择，覆盖了不少 vi 和 vi 变种的信息。

10. 关于这份文档

这份原始文档作为 Westtech Information Services 的所有，最初发表在 IBM developerWorks。这一份是更新过的文档，包含了很多 Gentoo Linux 文档团队的改进。