man快速入门

这是一个man的使用教程, 同时给出了一个如何寻找帮助的例子.

初识man

你是一只Linux菜鸟. 因为课程实验所迫, 你不得不使用Linux, 不得不使用十分落后的命令行. 实验内容大多数都要在命令行里进行, 面对着一大堆陌生的命令和参数, 这个链接中的饼图完美地表达了你的心情.

不行! 还是得认真做实验, 不然以后连码农都当不上了! 这样的想法鞭策着你, 因为你知道, 就算是码农, 也要有适应新环境和掌握新工具的能力. "还是先去找man吧." 于是你在终端里输入 `man` , 敲了回车. 只见屏幕上输出了一行信息:

> What manual page do you want?

噢, 原来命令行也会说人话! 你明白这句话的意思, `man` 在询问你要查询什么内容. 你能查询什么内容呢? 既然 `man` 会说人话, 还是先多了解 `man` 吧. 为了告诉 `man` 你想更了解ta, 你输入

```
man man
```

敲了回车之后, `man` 把你带到了一个全新的世界. 这时候, 你又看到了一句人话了, 那是 `man` 的独白, ta告诉你, ta的真实身份其实是

> an interface to the on-line reference manuals

接下来, ta忽然说了一大堆你听不懂的话, 似乎是想告诉你ta的使用方法. 可是你还没做好心理准备啊, 于是你无视了这些话.

寻找帮助

很快, 你已经看到"最后一行"了. 难道man的世界就这么狭小? 你仔细一看, "最后一行"里面含有一些信息:

> Manual page man(1) line 1 (press h for help or q to quit)

原来可以通过按 `q` 来离开这个世界啊, 不过你现在并不想这么做, 因为你想多了解 `man` , 以后可能会经常需要 `man` 的帮助. 为了更了解ta, 你按了 `h` .

这时你又被带到了新的世界, 世界的起点是"SUMMARY OF LESS COMMANDS", 你马上知道, 这个世界要告诉你如何使用 `man` , 你十分激动. 于是你往下看, 这句话说"带有'*'标记的命令可以在前面跟一个数, 这时命令的行为在括号里给出". 这是什么意思? 你没看懂, 还是找个带'*'的命令试试吧. 你继续往下看, 看到了两个功能和相应的命令:

- 第一个是展示帮助, 原来除了 `h` 之外, `H` 也可以看到帮助, 而且这里把帮助的命令放在第一个, 也许 `man` 想暗示你, 找到帮助是十分重要的.
- 第二个命令是退出. "哈哈, 知道怎么退出之后, 就不用通过重启来退出一个命令行程序啦", 你心想. 但你现在还是不想退出, 还是再看看其它的吧.

继续往下看, 你看到了用于移动的命令. 果然, 你还是可以在这个世界里面移动的. 第一个用于移动的功能是往下移动一行, 你看到有5种方法可以实现:

```
e  ^E  j  ^N  CR
```

`e` 和 `j` 你看懂了, 就是按 `e` 或者 `j` . 但 `^E` 是什么意思呢? 你尝试找到 `^` 的含义, 但是你没找到, 还是让我告诉你吧. 在上下文和按键有关的时候, `^` 是Linux中的一个传统记号, 它表示 `ctrl+` . 还记得Windows下 `ctrl+c` 代表复制的例子吗? 这里的 `^E` 表示 `ctrl+E` . `CR` 代表回车键, 其实 `CR` 是控制字符(ASCII码小于32的字符)的一个, 这里有一段关于控制字符的问答.

你决定使用 `j` , 因为它像一个向下的箭头, 而且它是右手食指所按下的键. 其实这点和 `vim` 的使用是类似的, 如果你不能理解为什么 `vim` 中使用 `h` , `j` , `k` , `l` 作为方向键, 这里有一个初学者的提问, 事实上, 这是一种touch typing.

你按下了 `j`, 发现画面上的信息向下滚动了一行. 你看到了 `*`, 想起了 `*` 标记的命令可以在前面跟一个数. 于是你试着输入 `10j`, 发现画面向下滚动了10行, 你第一次感觉到在这个"丑陋"的世界中也有比GUI方便的地方. 你继续阅读帮助, 并且尝试每一个命令. 于是你掌握了如何通过移动来探索 `man` 所在的世界.

继续往下翻, 你看到了用于搜索的命令. 你十分感动, 因为使用关键字可以快速定位到你关心的内容. 帮助的内容告诉你, 通过按 `/` 激活前向搜索模式, 然后输入关键字(可以使用正则表达式), 按下回车就可以看到匹配的内容了. 帮助中还列出了后向搜索, 跳到下一匹配处等功能. 于是你掌握了如何使用搜索.

探索man

你一边阅读帮助, 一边尝试新的命令, 就这样探索着这个陌生的世界. 你虽然记不住这么多命令, 但你知道你可以随时来查看帮助. 掌握了一些基本的命令之后, 你按 `q` 离开了帮助, 回到了 `man` 的世界. 现在你可以自由探索 `man` 的世界了. 你向下翻, 跳过了看不懂的 `SYNOPSIS` 小节, 在 `DESCRIPTION` 小节看到了人话, 于是你阅读这些人话. 在这里, 你看到整个manual分成9大类, 每个manual page都属于其中的某一类; 你看到了一个manual page主要包含以下的小节:

- NAME - 命令名
- SYNOPSIS - 使用方法大纲
- CONFIGURATION - 配置
- DESCRIPTION - 功能说明
- OPTIONS - 可选参数说明
- EXIT STATUS - 退出状态, 这是一个返回给父进程的值
- RETURN VALUE - 返回值
- ERRORS - 可能出现的错误类型
- ENVIRONMENT - 环境变量
- FILES - 相关配置文件
- VERSIONS - 版本
- CONFORMING TO - 符合的规范
- NOTES - 使用注意事项
- BUGS - 已经发现的bug
- EXAMPLE - 一些例子
- AUTHORS - 作者
- SEE ALSO - 功能或操作对象相近的其它命令 你还看到了对 `SYNOPSIS` 小节中记号的解释, 现在你可以回过头来看 `SYNOPSIS` 的内容了. 但为了弄明白每个参数的含义, 你需要查看 `OPTIONS` 小节中的内容.

你想起了搜索的功能, 为了弄清楚参数 `-k` 的含义, 你输入 `/-k`, 按下回车, 并通过 `n` 跳过了那些 `OPTIONS` 小节之外的 `-k`, 最后大约在第254行找到了 `-k` 的解释: 通过关键字来搜索相关功能的manual page. 在 `EXAMPLES` 小节中有一个使用 `-k` 的例子:

```
man -k printf
```

你阅读这个例子的解释: 搜索和 `printf` 相关的manual page. 你还是不太明白这是什么意思, 于是你退出 `man`, 在命令行中输入

```
man -k printf
```

并运行, 发现输出了很多和 `printf` 相关的命令或库函数, 括号里面的数字代表相应的条目属于manual的哪一个大类. 例如 `printf (1)` 是一个shell命令, 而 `printf (3)` 是一个库函数. 要访问库函数 `printf` 的manual page, 你需要在命令行中输入

```
man 3 printf
```

当你想做一件事的而不知道用什么命令的时候, `man` 的 `-k` 参数可以用来列出候选的命令, 然后再通过查看这些命令的manual page来学习怎么使用它们.

接下来, 你又开始学习 `man` 的其它功能...

开始旅程

到这里, 你应该掌握 `man` 的用法了. 你应该经常来拜访ta, 因为在很多时候, ta总能给你提供可靠的帮助.

在这个励志的故事中, 你学会了:

- 阅读程序输出的提示和错误信息
- 通过搜索来定位你关心的内容
- 动手实践是认识新事物的最好方法
- 独立寻找帮助, 而不是一有问题就问班上的大神

于是, 你就这样带着 `man` 踏上了Linux之旅...