Linux 下查看文件字符编码和转换编码

如果你需要在 Linux 中操作 windows 下的文件，那么你可能会经常遇到文件编码转换的问题。Windows 中默认的文件格式是 GBK(gb2312)，而 Linux 一般都是 UTF-8。下面介绍一下，在 Linux 中如何查看文件的编码及如何进行对文件进行编码转换。

一，查看文件编码：

在 Linux 中查看文件编码可以通过以下几种方式：

1.在 Vim 中可以直接查看文件编码

```
:set fileencoding
```

即可显示文件编码格式。

如果你只是想查看其它编码格式的文件或者想解决用 Vim 查看文件乱码的问题，那么你可以在

~/.vimrc 文件中添加以下内容：

```
set encoding=utf-8
fileencodings=ucs-bom,utf-8,cp936
```

这样，就可以让 vim 自动识别文件编码（可以自动识别 UTF-8或者 GBK 编码的文件），其实就是依照 fileencodings 提供的编码列表尝试，如果没有找到合适的编码，就用 latin-1(ASCII)编码打开。

2. enca (如果你的系统中没有安装这个命令 ,可以用 sudo yum install -y enca 安装)查看文件编码

```
$ enca filename

filename: Universal transformation format 8 bits; UTF-8

CRLF line terminators
```

需要说明一点的是，enca 对某些 GBK 编码的文件识别的不是很好，识别时会出现：

```
Unrecognized encoding
```

二，文件编码转换

1.在 Vim 中直接进行转换文件编码,比如将一个文件转换成 utf-8格式

```
:set fileencoding=utf-8
```

2. iconv 转换，iconv 的命令格式如下：

输入/输出格式规范：

-f, --from-code=名称 原始文本编码

-t, --to-code=名称 输出编码

信息： Www.Svn8.Com

-l, --list 列举所有已知的字符集

输出控制：

-c 从输出中忽略无效的字符

-o, --output=FILE 输出文件 Svn8.Com

-s, --silent 关闭警告

--verbose 打印进度信息

-?, --help 给出该系统求助列表

--usage 给出简要的用法信息

-V, --version 打印程序版本号

例子:

iconv -f utf-8 -t gb2312 aaa.txt >bbb.txt

这个命令读取 aaa.txt 文件，从 utf-8编码转换为 gb2312编码,其输出定向到 bbb.txt 文件。

iconv -f encoding -t encoding inputfile

比如将一个 UTF-8 编码的文件转换成 GBK 编码

iconv -f GBK -t UTF-8 file1 -o file2

3．enconv 转换文件编码

比如要将一个 GBK 编码的文件转换成 UTF-8编码，操作如下

enconv -L zh_CN -x UTF-8 filename

三，文件名编码转换：

从 Linux 往 windows 拷贝文件或者从 windows 往 Linux 拷贝文件，有时会出现中文文件名乱码的情况，出现这种问题的原因是因为，windows 的文件名 中文编码默认为 GBK,而 Linux 中默认文件名编码为 UTF8,由于编码不一致，所以导致了文件名乱码的问题，解决这个问题需要对文件名进行转码。

在 Linux 中专门提供了一种工具 **convmv** 进行文件名编码的转换，可以将文件名从 GBK 转换成 UTF-8编码，或者从 UTF-8转换到 GBK。

首先看一下你的系统上是否安装了 convmv，如果没安装的话用：

yum -y install convmv 安装。

下面看一下 convmv 的具体用法：

例如

convmv -f GBK -t UTF-8 *.mp3

不过这个命令不会直正的转换，你可以看到转换前后的对比。如果要直正的转换要加上参数 --notest

convmv -f GBK -t UTF-8 --notest *.mp3

-f 参数是指出转换前的编码，-t 是转换后的编码。这个千万不要弄错了。不然可能还是乱码哦。还有一个参数很有用。就是 -r 这个表示递归转换当前目录下的所有子目录。

convmv -f 源编码 -t 新编码 [选项] 文件名

常用参数：

-r 递归处理子文件夹

–notest 真正进行操作，请注意在默认情况下是不对文件进行真实操作的，而只是试验。

–list 显示所有支持的编码

–unescap 可以做一下转义，比如把%20变成空格

比如我们有一个 utf8编码的文件名，转换成 GBK 编码，命令如下：

convmv -f UTF-8 -t GBK –notest utf8编码的文件名

这样转换以后”utf8编码的文件名”会被转换成 GBK 编码（只是文件名编码的转换，文件内容不会发生变化）

四，vim 编码方式的设置

和所有的流行文本编辑器一样，Vim 可以很好的编辑各种字符编码的文件，这当然包括 UCS-2、UTF-8 等流行的 Unicode 编码方式。然而不幸的是，和很多来自 Linux 世界的软件一样，这需要你自己动手设置。Vim 有四个跟字符编码方式有关的选项，encoding、fileencoding、fileencodings、termencoding（这些选项可能的取值请参考 Vim 在线帮助 :help encoding-names），它们的意义如下：

* encoding: Vim 内部使用的字符编码方式，包括 Vim 的 buffer（缓冲区）、菜单文本、消息文本等。默认是根据你的 locale 选择.用

户手册上建议只在 .vimrc 中改变它的值，事实上似乎也只有在.vimrc 中改变它的值才有意义。你可以用另外一种编码来编辑和保存文件，如你的 vim 的 encoding 为 utf-8,所编辑的文件采用 cp936 编码,vim 会 自动将读入的文件转成 utf-8(vim 的能读懂的方式)，而当你写入文件时,又会自动转回成 cp936（文件的保存编码).

* fileencoding: Vim 中当前编辑的文件的字符编码方式，Vim 保存文件时也会将文件保存为这种字符编码方式（不管是否新文件都如此)。

* fileencodings: Vim 自动探测 fileencoding 的顺序列表，启动时会按照它所列出的字符编码方式逐一探测即将打开的文件的字符编码方式，并且将 fileencoding 设置为最终探测到的字符编码方式。因此最好将 Unicode 编码方式放到这个列表的最前面，将拉丁语系编码方式 latin1 放到最后面。

* termencoding: Vim 所工作的终端（或者 Windows 的 Console 窗口）的字符编码方式。如果 vim 所在的 term 与 vim 编码相同，则无需设置。如其不然，你可以用 vim 的 termencoding 选项将自动转换成 term 的编码.这个选项在 Windows 下对我们常用的 GUI 模式的 gVim 无效，而对 Console 模式的 Vim 而言就是 Windows 控制台的代码页，并且通常我们不需要改变它。

五，Vim 的多字符编码工作方式

1. Vim 启动，根据 .vimrc 中设置的 encoding 的值来设置 buffer、菜单文本、消息文的字符编码方式。

2．读取需要编辑的文件，根据 fileencodings 中列出的字符编码方式逐一探测该文件编码方式。并设置 fileencoding 为探测到的，看起来是正确的（注1）字符编码方式。

3．对比 fileencoding 和 encoding 的值，若不同则调用 iconv 将文件内容转换为 encoding 所描述的字符编码方式，并且把转换后的内容放到为此文件开辟的 buffer 里，此时我们就可以开始编辑这个文件了。注意，完成这一步动作需要调用外部的 iconv.dll(注2)，你需要保证这个文件存在于 $VIMRUNTIME 或者其他列在 PATH 环境变量中的目录里。

4．编辑完成后保存文件时，再次对比 fileencoding 和 encoding 的值。若不同，再次调用 iconv 将即将保存的 buffer 中的文本转换为 fileencoding 所描述的字符编码方式，并保存到指定的文件中。同样，这需要调用 iconv.dll 由于 Unicode 能够包含几乎所有的语言的字符，而且 Unicode 的 UTF-8 编码方式又是非常具有性价比的编码方式（空间消耗比 UCS-2 小），因此建议 encoding 的值设置为 utf-8。这么做的另一个理由是 encoding 设置为 utf-8 时，Vim 自动探测文件的编码方式会更准确（或许这个理由才是主要的　　。我们在中文 Windows 里编辑的文件，为了兼顾与其他软件的兼容性，文件编码还是设置为 GB2312/GBK 比较合适，因此 fileencoding 建议设置为 chinese（chinese 是个别名，在 Unix 里表示 gb2312，在 Windows 里表示 cp936，也就是 GBK 的代码页）。