**Modifying Text-File Locales**  
Sometimes it’s necessary to access textual data that originated on a system that used one  
encoding but processed the data with a program that doesn’t support that encoding.  
For instance, your preferred text editor might support UTF-8 but not ISO-8859. If you  
deal exclusively with English text files in ASCII, this isn’t a problem; but if you receive an  
ISO-8859-1 text file with a few non-Roman characters, such as characters with umlauts,  
your editor might display those characters strangely.  
To overcome this problem, the iconv utility converts between character sets. Its syntax  
is as follows:  
iconv -f encoding [-t encoding] [inputfile]...

-f: fromcode

-t: tocode  
The -f and -t options specify the source and destination encodings. (You can obtain  
a list of encodings by typing **iconv --list**.) If you omit the target encoding, iconv uses  
your current locale for guidance. The program sends output to standard output, so if you  
want to store the data in a file, you must redirect it:  
$ **iconv -f iso-8859-1 -t UTF-8 umlautfile.txt > umlautfile-utf8.txt**

One of your coworkers in the Asia-Pacific office sent you a file containing instructions on how to set up the new application he is working on. When you look at the file in your editor, it’s a mess. You run **file** on the message and see:

message: Big-endian UTF-16 Unicode text

How can you decode this file?

**iconv -f UTF-16 -t ASCII message**