works closely with the Unicode Consortium to ensure that ISO/IEC 10646 remains consistent with Unicode. Because Unicode and UCS are so similar, I'll use the two terms interchangeably.

Q&A

Unicode was originally limited to 65,536 characters (the number of characters that can be represented using 16 bits). That limit was later found to be insufficient; Unicode currently has over 100,000 characters. (For the most recent version, visit *www.unicode.org*.) The first 65,536 characters of Unicode—which include the most frequently used characters—are known as the ***Basic Multilingual Plane (BMP)***.

Encodings of Unicode

Unicode assigns a unique number (known as a ***code point***) to each character. There are a number of ways to represent these code points using bytes; I'll mention two of the simpler techniques. One of these encodings uses wide characters; the other uses multibyte characters.

UCS-2 is a wide-character encoding in which each Unicode code point is stored as two bytes. UCS-2 can represent all the characters in the Basic Multilingual Plane (those with code points between 0000 and FFFF in hexadecimal), but it is unable to represent Unicode characters that don't belong to the BMP.

A popular alternative is the *8-bit UCS Transformation Format (UTF-8)*, which uses multibyte characters. UTF-8 was devised by Ken Thompson and his Bell Labs colleague Rob Pike in 1992. (Yes, that's the same Ken Thompson who designed the B language, the predecessor of C.) UTF-8 has the useful property that ASCII characters look identical in UTF-8: each character is one byte and has the same binary encoding. Thus, software designed to read UTF-8 data can also handle ASCII data with no change. For these reasons, UTF-8 is widely used on the Internet for text-based applications such as web pages and email.

In UTF-8, each code point requires between one and four bytes. UTF-8 is organized so that the most commonly used characters require fewer bytes, as shown in Table 25.7.

Table 25.7 UTF-8 Encoding

Code Point Range (Hexadecimal)	*UTF-8 Byte Sequence (Binary)*
000000-00007F	0xxxxxxx
000080-0007FF	110xxxxx 10xxxxxx
000800-00FFFF	1110xxxx 10xxxxxx 10xxxxxx
010000-10FFFF	11110xxx 10xxxxxx 10xxxxxx 10xxxxxx

UTF-8 takes the bits in the code point value, divides them into groups (represented by the x's in Table 25.7), and assigns each group to a different byte. The simplest case is a code point in the range 0–7F (an ASCII character), which is represented by a 0 followed by the seven bits in the original number.

A code point in the range 80–7FF (which includes all the Latin-1 characters) would have its bits split into groups of five bits and six bits. The five-bit group is