**ASCII码背景：**

ASCII编码是由美国人发明，美国的字符不超过255个，所以ASCII编码使用了**8bit 即一个字节**来存储字符。八个二进制位就可以组合出256种状态

虽然可以组合出256种状态，但ASCII 码一共只规定了128个字符,也就是只用7位的编码，比如空格SPACE是32（二进制00100000），大写的字母A是65（二进制01000001）。这128个符号（包括32个不能打印出来的控制符号），只占用了一个字节的后面7位，最前面的一位统一规定为0。

**Unicode编码背景：**

汉字的数量远超256个，即便我们将ASCII码得最高位也使用上也只能代表256种组合。所以中国自己发明了一个GB2312编码来表示汉字，一般的汉字使用2个字节.当然，GB2313编码是可以兼容ASCII码的。

然后，日本，韩国等等国家也自己发明了一套编码方法，这时候又出现了一个新的问题。如果一篇文章里面，即有中文，又有日文的话，无论使用中文的编码方法还是使用日文的编码方法都会出现乱码。

随后，Unicode编码便应运而生。Unicode编码对文字的编码进行了统一。Unicode编码表和ASCII编码表一样原理，但是对于世界上各种语言都有对应的Unicode编码，而不像ASCII码表一样只能对应英文。例如下图的Unicode编码表：



例如汉字“辉”，查询Unicode编码表，可以获得Unicode的编码十六进制表示为8f89。

需要注意的是Unicode只规定了符号所对应的十六进制数，但是没有规定如何转为2进制数来表达。

假设，如果我们只采用十六进制按权重转为二进制的话，例如辉将变为1000111110001001，则需要2个字节表示“辉”这个中文符号。相比之下我们从ASCII码的设计中可以看出，英文只需要一个字节表示。如果采用定长表示方法，则会浪费内存空间。

在此条件下，出现了UTF-8，UTF-16, UTF-32等的编码方式去具体规定Unicode码转为最终2进制编码。而UTF-8 就是在互联网上使用最广的一种 Unicode 的实现方式。UTF-8 最大的一个特点，就是它是一种变长的编码方式。它可以使用1~4个字节表示一个符号，根据不同的符号而变化字节长度。

从而也可以推论出，对于同一个汉字”辉“, Unicode码是8f89，当采用不同的编码方式utf8, utf16, utf32转为二进制编码实际进行传输的时候，所得出的二进制编码将不相同。

**字符集(Chracter Set)和编码规范(Encoding)：**

严格来说Unicode称为字符集。表示所能识别的字符以及对应的十六进制编码。

UTF8, UTF16, UTF32称为编码规范，用于将字符集对应的十六进制编码具体转为2进制串。

**常见编码规则：**

Latin1是ISO-8859-1的别名，有些环境下写作Latin-1。ISO-8859-1编码是定长单字节编码，它把ASCII码没有用到过的最前一位都给编上了对应的字符，所以它能表示的字符就更多了. 向下兼容ASCII。ISO-8859-1收录的字符除ASCII收录的字符外，还包括西欧语言、希腊语、泰语、阿拉伯语、希伯来语对应的文字符号.

**编码兼容性：**

utf8，Latin-1，ASCII 三者兼容性

🡪 ASCII与Latin-1之间。所有的ASCII码都是有效的Latin1码，ASCII码编码的内容都可以用latin-1解码出正确的内容， 所有英文字母使用ASCII码和latin-1所得出的编码相同。如果用latin-1只对英文字母内容编码，使用ASCII码也能正确解析。但是如果latin-1对额外字符进行编码，再使用ASCII码解析将出错。

🡪 ASCII与utf8之间。所有的ASCII码都是有效的utf8码，ASCII码编码的内容都可以用utf8解码出正确的内容， 所有英文字母使用ASCII码和utf8所得出的编码相同。如果用utf8只对英文字母内容编码，使用ASCII码也能正确解析。但是如果utf8对额外字符进行编码，再使用ASCII码解析将出错。

🡪Latin-1与utf8之间。如果只是英文字母，那么Latin-1与utf8之间编码相同。但是对于超出ASCII码所能表示的其他字符，Latin-1与UTF8将不相同。所以如果只是英文字符，使用Latin1编码，再utf8解码没问题。或者先使用utf8编码再Latin1解码也没问题，甚至使用ASCII加码，Latin1解码, 或者utf8解码也行。但如果内容含有超出英文字符部分，由于Latin-1与UTF8不同，使用Latin-1编码后必须用Latin-1解码，UTF8解码将会出错。