**文件编码与字符编码**

首先明确一点，文件不存在什么编码(归根结底文件都是二进制文件，用ue打开可以看到都是一个个的16进制数)，只有文件中的字符才可以说编码。

**编码与解码过程**

字符通过某种编码组织起来存到文件里面，计算机通过这种编码解析解析文件，根据解析出来的文字绘制图片显示到显示设备中，这样我们就看到了文字。

**常见编码介绍**

**ansi编码**

最初的计算机是又8个晶体管，通过晶体管的开合与排列可以表示数种状态，所以一个字节就定义为8bit，而一个bit只能有0，1开关的表示，2的8次方是256，所以最初的计算机只能表示256种状态。

人们定义了前32个为状态字符比如翻页，换行，发出（嘟）的声音等，后来人们为了用计算机存储文字，又把空格，英文字母，数字等加进了进来，总共使用了127个，这时候大家把这个存储方案叫做ANSI的[ASCII编码](http://www.dreamdu.com/xhtml/ascii/)[American Standard Code for Information Interchange，美国信息互换标准代码](http://www.dreamdu.com/xhtml/ascii/)，这个表存储英文已经没有 问题了，但是127个里并不包含其它欧洲国际的文字，人们又继续扩展ASCII表的内容，加入了一些字符，与一些画表格的符号，直接扩展到255个。

**GB2312与GBK**

国人发现只使用ASCII表根本无法表示汉字！怎么办？没有什么能难道我们！于是我们发明了GB2312编码，此编码完全忽略了ASCII表中 127位后面的内容，127位前面的内容保留，如果两个字节同时大于127(7F)就认为这两个字节表示一个汉字，同时像标点、字母也都重新使用两个字节 定义了一遍，这就是我们经常说的 全角，这种方案可以表示6000种文字。

但是中国的文字太复杂6000个字也不够用，人们开始扩展GB2312，规定只要一个字节大于127，这个字节和后面的字节组合起来就代表一个汉字，这种编码成为GBK，于是又增加了20000多个汉字！

现在明白[meta](http://www.dreamdu.com/xhtml/tag_meta/)的[编码信息](http://www.dreamdu.com/xhtml/charset/)里为什么有GB2312与GBK了吧？：）

<meta http-equiv="content-type" content="text/html; charset=GB2312" /> <meta http-equiv="content-type" content="text/html; charset=GBK" />

这样很多国家都开始定义自己的编码了，日本，韩国等。甚至连中国的台湾省都定义了一种编码 BIG5。所以在当时一个程序要想适应多国语言简直要把人郁闷死。

如果搞过windows编程的人应该知道，win里面有多字节字符集MBCS（multi-byte character set）的说法，而且MBCS包含两种字符类型，单字节字符SBCS（single-byte characters set）和双字节字符（double-byte characters set）DBCS。我们的GBK与GB2312都是DBCS。所以我们在编程时经常遇到一个中文字符等于两个英文字符的事情。BIG5与日本韩国的编码也 都属于DBCS。

这下清楚了吧，根本没什么ansi文件或gb2312文件，文件打开时会根据操作系统的编码方式（就是安装在操作系统中的编码解析方式）来尝试打开文件，如果安装了中文编码，就把ansi文件当作中文打开，如果日文编码，就当作日文打开。

**UNICODE与UTF-8**

ISO最后提出了UNICODE(Universal Multiple-Octet Coded Character Set，简称 UCS)编码来解决所有的问题。

UNICODE编码方式规定使用两个字节(16位)表示表示一个字符，算算2的16次方是多少？原来ANSI规定的都扩充为2字节，并且把所有已知的语言都编码进UNICODE。UNICODE可以表示65536个字符。

这下多国语言程序开始高兴了，使用UNICODE全部搞定！于是微软重现编辑windows内核，完全使用UNICODE编码，搞过win编程的人应该都知道，以A或W结尾的函数，还有灵巧的\_T宏吧？

虽然UNICODE有很多优点，但是缺点也不少，我先总结我知道的两点：

**1，狂占空间**，以前一个字节可以表示，现在却要用两个字节了，网络上有80%属于英文字符，这下网络几乎要扩大一倍！

所以又有人研究出来了UTF-8(Unicode Translation Format - 8)编码，UNICODE转换格式，对于常用字符使用单字节，汉字等使用双字节。8代表每次在网络上传输8位，如果是UTF-16就是每次传输16位。搞 过网络编程的朋友应该知道，字节序（就是字节的排放顺序）分为两种，主机字节序与网络字节序，就是大头（俗称）在前，小头在前的问题，在网络上面传输的流 的字节序很可能是不一致的，于是需要使用一种方法通知接收端，传输流的字节序。有人发明了一种简单的方法，在每个流的开始加上FFEF或EFFF，分别主 机与网络字节序，我们可以使用记事本保存一个UNICODE文件，再使用ue打开看看（HEX方式打开）。所以有时候网页传到网上，在网页最开始的地方会 出现一个字符，这个有时候很令人费解。

用记事本新建立两个文件存为UNICODE与UNICODE big endian模式，输入梦之都，保存再用ue打开。

**UNICODE**

FF FE A6 68 4B 4E FD 90

**UNICODE big endian**

FE FF 68 A6 4E 4B 90 FD (观察，没两个字节和上面的对比)

**2，UNICODE与GBK等两字节编码完全不兼容，**无法找到一种简单的方式转换(只能使用查找表的方式)

这点我们可以使用记事本新建立两个文件一个ansi的文件，另一个utf8的文件，分别写入梦之都 ，保存。使用ue的hex模式打开我们会看到。

**UNICODE**

FF FE A6 68 4B 4E FD 90

**ansi**

C3 CE D6 AE B6 BC

**猜编码**

在windows系统中打开文件时，是使用了猜的方式选择解析文件内容编码系统，如果文件开头使用了FEFF或FFFE，win系统认为 UNICODE编码，否则为ANSI编码，如果是ANSI编码继续分析，如果一个字节大于127，就证明这个字节与后面的字节组成了一个汉字。

所以windows中文系统下，如果ansi文件，那么就会用gb2312方式转换，如果是日文系统，就会使用日文方式转变，但是绝对不能说ansi文件里面有中文字符就是gb2312！一个gb2312占用两个字节。而utf8 win系统在前面加了几个字节以示区别。

通过这种方式分析时会产生一个很著名的问题，如果用记事本输入“联通”保存，再打开，发现“联通”两个字没了！为什么没有了，大家可以自己分析一下。有人说这就是联通竞争不过移动的原因。

总结：

不同的国家和地区制定了不同的标准，由此产生了 GB2312, BIG5, JIS 等各自的编码标准。这些使用 2 个[字节](http://baike.baidu.com/view/60408.htm)来代表一个字符的各种汉字延伸编码方式，称为 ANSI 编码。在简体中文系统下，ANSI 编码代表 GB2312 编码，在日文[操作系统](http://baike.baidu.com/view/880.htm)下，ANSI 编码代表 JIS 编码。 不同 ANSI 编码之间互不兼容，当信息在国际间交流时，无法将属于两种语言的文字，存储在同一段 ANSI 编码的文本中。 当然对于ANSI编码而言，0x00~0x7F之间的[字符](http://baike.baidu.com/view/263416.htm)，依旧是1个[字节](http://baike.baidu.com/view/60408.htm)代表1个字符。这一点是ASNI编码与Unicode编码之间最大也最明显的区别。