**编码集的整理与理解**

## Unicode编码

Unicode（[统一码](https://baike.baidu.com/item/%E7%BB%9F%E4%B8%80%E7%A0%81)、万国码、单一码）是计算机科学领域里的一项业界标准,包括字符集、编码方案等。Unicode 是为了解决传统的字符编码方案的局限而产生的，它为每种语言中的每个字符设定了统一并且唯一的二进制编码，以满足跨语言、跨平台进行文本转换、处理的要求。1990年开始研发，1994年正式公布。Unicode 是为了解决传统的[字符编码](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6%E7%BC%96%E7%A0%81" \t "_blank)方案的局限而产生的，例如ISO 8859所定义的字符虽然在不同的国家中广泛地使用，可是在不同国家间却经常出现不兼容的情况。很多传统的[编码](https://baike.baidu.com/item/%E7%BC%96%E7%A0%81/80092" \t "_blank)方式都有一个共同的问题，即容许电脑处理双语环境（通常使用[拉丁字母](https://baike.baidu.com/item/%E6%8B%89%E4%B8%81%E5%AD%97%E6%AF%8D" \t "_blank)以及其本地语言），但却无法同时支持多语言环境（指可同时处理多种语言混合的情况）。

在文字处理方面，Unicode为每一个字符而非字形定义唯一的代码（即一个整数）。换句话说，统一码以一种抽象的方式（即数字）来处理字符，并将视觉上的演绎工作（例如字体大小、外观形状、字体形态、文体等）留给其他软件来处理，例如网页浏览器或是文字处理器。

几乎所有电脑系统都支持基本拉丁[字母](https://baike.baidu.com/item/%E5%AD%97%E6%AF%8D)，并各自支持不同的其他编码方式。Unicode为了和它们相互兼容，其首256字符保留给ISO 8859-1所定义的字符，使既有的西欧语系文字的转换不需特别考量；并且把大量相同的字符重复编到不同的字符码中去，使得旧有纷杂的编码方式得以和Unicode编码间互相直接转换，而不会丢失任何信息。举例来说，[全角](https://baike.baidu.com/item/%E5%85%A8%E8%A7%92" \t "_blank)格式区段包含了主要的拉丁字母的全角格式，在中文、日文、以及韩文字形当中，这些字符以全角的方式来呈现，而不以常见的半角形式显示，这对竖排文字和等宽排列文字有重要作用。

在表示一个Unicode的字符时，通常会用“U+”然后紧接着一组十六进制的数字来表示这一个字符。在[基本多文种平面](https://baike.baidu.com/item/%E5%9F%BA%E6%9C%AC%E5%A4%9A%E6%96%87%E7%A7%8D%E5%B9%B3%E9%9D%A2" \t "_blank)（英文为 Basic Multilingual Plane，简写 BMP。它又简称为“零号平面”, plane 0）里的所有字符，要用四位十六进制数（例如U+4AE0，共支持六万多个字符）；在零号平面以外的字符则需要使用五位或六位十六进制数了。旧版的Unicode标准使用相近的标记方法，但却有些微的差异：在Unicode 3.0里使用“U-”然后紧接着八位数，而“U+”则必须随后紧接着四位数。

## 二．GB2312编码

GB2312编码适用于汉字处理、汉字通信等系统之间的信息交换，通行于中国大陆；新加坡等地也采用此编码。中国大陆几乎所有的中文系统和国际化的软件都支持GB 2312。

GB2312基本集共收入汉字6763个和非汉字图形字符682个。整个字符集分成94个区，每区有94个位。每个区位上只有一个字符，因此可用所在的区和位来对汉字进行编码，称为[区位码](https://baike.baidu.com/item/%E5%8C%BA%E4%BD%8D%E7%A0%81" \t "_blank)。把换算成十六进制的区位码加上2020H，就得到[国标码](https://baike.baidu.com/item/%E5%9B%BD%E6%A0%87%E7%A0%81" \t "_blank)。国标码加上8080H，就得到常用的计算机机内码。

信息交换用汉字编码字符集和汉字输入编码之间的关系是，根据不同的汉字输入方法，通过必要的设备向计算机输入汉字的编码，计算机接收之后，先转换成信息交换用汉字编码字符，这时计算机就可以识别并进行处理；汉字输出是先把机内码转成汉字编码，再发送到输出设备。

在使用GB2312的程序中，通常采用EUC储存方法，以便兼容于ASCII。浏览器编码表上的“GB2312”，通常都是指“EUC-CN”表示法。每个汉字及符号以两个字节来表示。第一个[字节](https://baike.baidu.com/item/%E5%AD%97%E8%8A%82)称为“高位字节”（也称“区字节）”，第二个字节称为“低位字节”（也称“位字节”）。“高位字节”使用了0xA1-0xF7(把01-87区的区号加上0xA0)，“低位字节”使用了0xA1-0xFE(把01-94加上 0xA0)。 由于一级汉字从16区起始，汉字区的“高位字节”的范围是0xB0-0xF7，“低位字节”的范围是0xA1-0xFE，占用的码位是 72\*94=6768。其中有5个空位是D7FA-D7FE。

## 三．Big5编码

大五码（Big5），是通行于台湾、香港地区的一个繁体字编码方案。地区标准号为：CNS11643，这就是人们讲的Big-5码。VimIM在Vim环境中，可以直接键入十进制或十六进制Big5码。既不需要启动输入法，也不需要码表。

大五码是由资策会于1984年策划制定，拥有13053个中文字、408个字符以及33个控制字元的字集，是我国早期中文电脑的业界标准，也是中文社群最常用的电子汉字字集标准。而后随着电脑扩充需要，业界各操作系统开发商推出了不同版本的大五码，为统一标准，经济部标准检验局在2003年委托财团法人中国数位化技术推广基金会修改了大五码编码字元表，重整为Big5-2003版本。

Big5编码的使用方法为：首先安装BIG5码字体,安装方法：打开“控制面版”，找到“字体”打开，将需要的字体文件复制到字体文件夹中即可，安装完毕最好重新启动电脑。其它字体安装方法同上。（华康、金梅、超世纪等繁体字库都属于此类大五码字体，字体扩展名为TTF、或TTC）。其次，在word里把需要的汉字输入工作区内，然后将文字全选，选择“工具”菜单中的“语言”，内有“中文简繁转换”选择“简体中文转换为繁体中文”点确定，如果word中没有这个项目，说明你还没有安装，请首先找office安装光盘安装此项目。然后，如果你想在photoshop中使用，请在word中把要用的文字用上述步骤转换简体中文为繁体中文后，复制文字段，打开 photoshop，新建图像，点击文字输入工具，注意：先选择好字体（通常BIG5码字体名称都显示为字母和乱码，在字体列表最下边），然后用“粘贴” 命令，所要的字就出来了。一定注意，先选择字体再粘贴。最后，其他的繁体字库（BIG5码）98里也许不可以用，但在2000或xp、2003中按同样的方法也可以使用。

## 四．UTF-8/16

UTF-8（8-bit Unicode Transformation Format）是一种针对Unicode的可变长度字符编码，又称万国码。由Ken Thompson于1992年创建。现在已经标准化为RFC 3629。UTF-8用1到6个字节编码Unicode字符。用在网页上可以统一页面显示中文简体繁体及其它语言（如英文，日文，韩文）。

在所有[字符集](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6%E9%9B%86" \t "_blank)中，最知名的可能要数被称为[ASCII](https://baike.baidu.com/item/ASCII)的7位字符集了。它是[美国标准信息交换代码](https://baike.baidu.com/item/%E7%BE%8E%E5%9B%BD%E6%A0%87%E5%87%86%E4%BF%A1%E6%81%AF%E4%BA%A4%E6%8D%A2%E4%BB%A3%E7%A0%81" \t "_blank)（American Standard Code for Information Interchange）的缩写, 为美国英语通信所设计。它由128个[字符](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6" \t "_blank)组成，包括大小写字母、数字0-9、标点符号、非[打印](https://baike.baidu.com/item/%E6%89%93%E5%8D%B0" \t "_blank)字符（换行符、[制表符](https://baike.baidu.com/item/%E5%88%B6%E8%A1%A8%E7%AC%A6" \t "_blank)等4个）以及[控制字符](https://baike.baidu.com/item/%E6%8E%A7%E5%88%B6%E5%AD%97%E7%AC%A6" \t "_blank)（退格、响铃等）组成。

但是，由于他是针对英语设计的，当处理带有音调标号（形如汉语的拼音）的亚洲文字时就会出现问题。因此，创建出了一些包括255个字符的由ASCII扩展的字符集。其中有一种通常被称为IBM字符集，它把值为128-255之间的字符用于画图和画线，以及一些特殊的欧洲字符。另一种8位字符集是[ISO 8859-1](https://baike.baidu.com/item/ISO%208859-1" \t "_blank)Latin 1，也简称为ISOLatin-1。它把位于128-255之间的[字符](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6" \t "_blank)用于拉丁字母表中特殊语言字符的编码，也因此而得名。欧洲语言不是地球上的唯一语言，因此亚洲和非洲语言并不能被8位[字符集](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6%E9%9B%86" \t "_blank)所支持。仅汉语字母表（或pictograms）就有80000以上个[字符](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6" \t "_blank)。但是把汉语、日语和越南语的一些相似的字符结合起来，在不同的语言里，使不同的字符代表不同的字，这样只用2个[字节](https://baike.baidu.com/item/%E5%AD%97%E8%8A%82" \t "_blank)就可以编码地球上几乎所有地区的文字。因此，创建了[UNICODE](https://baike.baidu.com/item/UNICODE)编码。它通过增加一个高字节对ISO Latin-1字符集进行扩展，当这些高字节位为0时，低字节就是ISO Latin-1字符。UNICODE支持欧洲、非洲、中东、亚洲（包括统一标准的东亚象形汉字和韩国表音文字）。但是，UNICODE并没有提供对诸如Braille,[Cherokee](https://baike.baidu.com/item/Cherokee/61025), Ethiopic, Khmer, Mongolian, Hmong, Tai Lu, Tai Mau文字的支持。同时它也不支持如Ahom, Akkadian, Aramaic, [Babylonian](https://baike.baidu.com/item/Babylonian" \t "_blank)Cuneiform, Balti, Brahmi, Etruscan, Hittite, Javanese, Numidian, Old Persian Cuneiform, Syrian之类的古老文字。

事实证明，对可以用[ASCII](https://baike.baidu.com/item/ASCII)表示的[字符](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6" \t "_blank)使用UNICODE并不高效，因为UNICODE比ASCII占用大一倍的空间，而对ASCII来说高[字节](https://baike.baidu.com/item/%E5%AD%97%E8%8A%82" \t "_blank)的0对他毫无用处。为了解决这个问题，就出现了一些中间格式的[字符集](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6%E9%9B%86" \t "_blank)，他们被称为通用转换格式，即[UTF](https://baike.baidu.com/item/UTF)（Unicode Transformation Format）。常见的UTF格式有：UTF-7, UTF-7.5, UTF-8,[UTF-16](https://baike.baidu.com/item/UTF-16), 以及 [UTF-32](https://baike.baidu.com/item/UTF-32)。

UTF-8编码规则：如果只有一个字节则其最高二进制位为0；如果是多字节，其第一个字节从最高位开始，连续的二进制位值为1的个数决定了其编码的字节数，其余各字节均以10开头。Unicode转换为UTF-8时，可以将Unicode二进制从低位往高位取出二进制数字，每次取6位，如上述的二进制就可以分别取出为如下示例所示的格式，前面按格式填补，不足8位用0填补。

UTF-8编码的优点是可以通过屏蔽位和[移位](https://baike.baidu.com/item/%E7%A7%BB%E4%BD%8D)操作快速读写。字符串比较时strcmp()和[wcscmp](https://baike.baidu.com/item/wcscmp)()的返回结果相同，因此使排序变得更加容易。[字节](https://baike.baidu.com/item/%E5%AD%97%E8%8A%82)FF和FE在UTF-8编码中永远不会出现，因此他们可以用来表明UTF-16或UTF-32文本（见BOM） UTF-8 是[字节顺序](https://baike.baidu.com/item/%E5%AD%97%E8%8A%82%E9%A1%BA%E5%BA%8F)无关的。它的字节顺序在所有系统中都是一样的，因此它实际上并不需要BOM。

UTF-8编码的缺点是你无法从UNICODE[字符](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6)数判断出UTF-8文本的[字节](https://baike.baidu.com/item/%E5%AD%97%E8%8A%82)数，因为UTF-8是一种[变长编码](https://baike.baidu.com/item/%E5%8F%98%E9%95%BF%E7%BC%96%E7%A0%81)它需要用2个字节编码那些用[扩展ASCII](https://baike.baidu.com/item/%E6%89%A9%E5%B1%95ASCII)[字符集](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6%E9%9B%86)只需1个字节的字符 ISO Latin-1 是UNICODE的[子集](https://baike.baidu.com/item/%E5%AD%90%E9%9B%86)，但不是UTF-8的子集 8位字符的UTF-8编码会被email网关过滤，因为internet信息最初设计为7位ASCII码。因此产生了UTF-7编码。 UTF-8 在它的表示中使用值100xxxxx的几率超过50%， 而现存的实现如ISO 2022， 4873， 6429， 和8859系统，会把它错认为是C1 控制码。因此产生了UTF-7.5编码。