[目录 1](#_Toc24084)

[1. GB2312 1](#_Toc21144)

[1.来历： 1](#_Toc13716)

[2.解决的问题 1](#_Toc8692)

[3.字节数 1](#_Toc22533)

[4.描述 1](#_Toc21168)

[5.兼容 1](#_Toc8391)

[2.big5 2](#_Toc32298)

[1.来历 2](#_Toc18744)

[2.产生原因 2](#_Toc20446)

[3.解决的问题 3](#_Toc4140)

[4.描述 3](#_Toc28523)

[5.兼容 3](#_Toc23141)

[3. UTF\_8/16 5](#_Toc963)

[1.来历 5](#_Toc31604)

[2.解决的问题 6](#_Toc24822)

[3.字节数 6](#_Toc4834)

[4.描述 6](#_Toc15669)

[5.兼容 8](#_Toc1960)

[4. Unicode 9](#_Toc28412)

[1.来历 9](#_Toc15404)

[2.解决的问题 10](#_Toc1415)

[3.字节数 10](#_Toc10082)

[4.描述 11](#_Toc22017)

[5.兼容 12](#_Toc9156)

[相互联系： 13](#_Toc11018)

[参考文献： 14](#_Toc2876)

# GB2312

1.来历：GB 2312 或 GB 2312–80 是中华人民共和国国家标准简体中文字符集，全称《信息交换用汉字编码字符集·基本集》，又称GB0，由中国国家标准总局发布，1981年5月1日实施。GB 2312编码通行于中国大陆；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。

2.解决的问题：GB 2312的出现，基本满足了汉字的计算机处理需要，它所收录的已经覆盖中国大陆99.75%的使用频率。但对于人名、古汉语等方面出现的罕用字和繁体字，GB 2312不能处理，因此后来GBK及GB 18030汉字字符集相继出现以解决这些问题。

3.字节数：每个汉字及符号以两个字节来表示。第一个字节称为“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”）。

4.描述：“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。例如“啊”字在大多数程序中，会以两个字节，0xB0（第一个字节）0xA1（第二个字节）储存。（与区位码对比：0xB0=0xA0+16,0xA1=0xA0+1）。

5.兼容：GBK子集与GBK/GB 18030兼容，GB2312.TXT则不兼容。后者基于ftp.unicode.org曾经提供的GB2312.TXT实现，于2011年由官方弃用，2016年9月时已无原文件踪迹。此外还有很多种厂商实现。

截至2015年 (2015-Missing required parameter 1=month!)[[update]](https://zh.wikipedia.org/w/index.php?title=GB_2312&action=edit)，微软.NET使用的是“GBK子集”实现。[ICU](https://zh.wikipedia.org/w/index.php?title=%E4%B8%87%E5%9B%BD%E7%A0%81%E5%9B%BD%E9%99%85%E5%8C%96%E7%BB%84%E4%BB%B6&action=edit&redlink=1)（英语：[International Components for Unicode](https://en.wikipedia.org/wiki/International_Components_for_Unicode" \o "en:International Components for Unicode)）、libiconv-1.14、php-5.6、ActivePerl-5.20、Java 1.7、Python 3.4都使用“GB2312.TXT”实现。Ruby 2.2兼容两者编码，但内部使用“GBK子集”实现。W3C的编码技术指南规定，应将gb2312字节流视为GBK编码，与GB18030一并使用同一解码器解码。

# 2.big5

1.来历：Big5，又称为大五码或五大码，是使用[繁体中文](https://zh.wikipedia.org/wiki/%E7%B9%81%E4%BD%93%E4%B8%AD%E6%96%87" \o "繁体中文)（正体中文）社区中最常用的电脑[汉字](https://zh.wikipedia.org/wiki/%E6%B1%89%E5%AD%97" \o "汉字)[字符集](https://zh.wikipedia.org/wiki/%E5%AD%97%E7%AC%A6%E9%9B%86" \o "字符集)标准，共收录13,060个汉字[[1]](https://zh.wikipedia.org/wiki/%E5%A4%A7%E4%BA%94%E7%A2%BC" \l "cite_note-1)。

中文码分为[内码](https://zh.wikipedia.org/wiki/%E5%85%A7%E7%A2%BC" \o "内码)及[交换码](https://zh.wikipedia.org/wiki/%E4%BA%A4%E6%8F%9B%E7%A2%BC" \o "交换码)两类，Big5属中文内码，知名的中文交换码有[CCCII](https://zh.wikipedia.org/wiki/CCCII" \o "CCCII)、[CNS11643](https://zh.wikipedia.org/wiki/CNS11643" \o "CNS11643)。

Big5虽普及于[台湾](https://zh.wikipedia.org/wiki/%E5%8F%B0%E7%81%A3" \o "台湾)、[香港](https://zh.wikipedia.org/wiki/%E9%A6%99%E6%B8%AF" \o "香港)与[澳门](https://zh.wikipedia.org/wiki/%E6%BE%B3%E9%96%80" \o "澳门)等繁体中文通行区，但长期以来并非当地的国家/地区标准或官方标准，而只是**[业界标准](https://zh.wikipedia.org/wiki/De_facto" \o "De facto)**。[倚天中文系统](https://zh.wikipedia.org/wiki/%E5%80%9A%E5%A4%A9%E4%B8%AD%E6%96%87%E7%B3%BB%E7%B5%B1" \o "倚天中文系统)、[Windows](https://zh.wikipedia.org/wiki/Windows" \o "Windows)繁体中文版等主要系统的字符集都是以Big5为基准，但厂商又各自增加不同的造字与造字区，派生成多种不同版本。

2003年，Big5被收录到CNS11643中文标准交换码的附录当中，获取了较正式的地位。这个最新版本被称为[Big5-2003](https://zh.wikipedia.org/w/index.php?title=Big5-2003&action=edit&redlink=1" \o "Big5-2003（页面不存在）)。

2.产生原因：“大五码”（Big5）是由台湾[财团法人信息产业策进会](https://zh.wikipedia.org/wiki/%E8%B3%87%E8%A8%8A%E5%B7%A5%E6%A5%AD%E7%AD%96%E9%80%B2%E6%9C%83" \o "信息产业策进会)为[五大中文套装软件](https://zh.wikipedia.org/wiki/%E4%BA%94%E5%A4%A7%E4%B8%AD%E6%96%87%E5%A5%97%E8%A3%9D%E8%BB%9F%E9%AB%94" \o "五大中文套装软件)所设计的中文共通内码，在1983年12月完成公告[[2]](https://zh.wikipedia.org/wiki/%E5%A4%A7%E4%BA%94%E7%A2%BC" \l "cite_note-2)[[3]](https://zh.wikipedia.org/wiki/%E5%A4%A7%E4%BA%94%E7%A2%BC" \l "cite_note-3)，隔年3月，信息产业策进会与[台湾](https://zh.wikipedia.org/wiki/%E8%87%BA%E7%81%A3" \o "台湾)13家厂商签定“16位个人电脑套装软件合作开发（BIG-5）项目（[五大中文套装软件](https://zh.wikipedia.org/wiki/%E4%BA%94%E5%A4%A7%E4%B8%AD%E6%96%87%E5%A5%97%E8%A3%9D%E8%BB%9F%E9%AB%94" \o "五大中文套装软件)）”[[4]](https://zh.wikipedia.org/wiki/%E5%A4%A7%E4%BA%94%E7%A2%BC" \l "cite_note-4)，因为此中文内码是为台湾自行制作开发之“五大中文套装软件”所设计的，所以就称为Big5中文内码[[5]](https://zh.wikipedia.org/wiki/%E5%A4%A7%E4%BA%94%E7%A2%BC" \l "cite_note-5)[[6]](https://zh.wikipedia.org/wiki/%E5%A4%A7%E4%BA%94%E7%A2%BC" \l "cite_note-6)[[7]](https://zh.wikipedia.org/wiki/%E5%A4%A7%E4%BA%94%E7%A2%BC" \l "cite_note-7)[[8]](https://zh.wikipedia.org/wiki/%E5%A4%A7%E4%BA%94%E7%A2%BC" \l "cite_note-8)。五大中文套装软件虽然并没有如预期的取代国外的套装软件，但随着采用Big5码的[国乔中文系统](https://zh.wikipedia.org/w/index.php?title=%E5%9C%8B%E5%96%AC%E4%B8%AD%E6%96%87%E7%B3%BB%E7%B5%B1&action=edit&redlink=1" \o "国乔中文系统（页面不存在）)及[倚天中文系统](https://zh.wikipedia.org/wiki/%E5%80%9A%E5%A4%A9%E4%B8%AD%E6%96%87%E7%B3%BB%E7%B5%B1" \o "倚天中文系统)先后在台湾市场获得成功，使得Big5码深远地影响繁体中文电脑[内码](https://zh.wikipedia.org/wiki/%E5%85%A7%E7%A2%BC" \o "内码)，直至今日。“五大码”的英文名称“Big5”后来被人按英文字序译回中文，以致现在有“五大码”和“大五码”两个中文名称。

Big5码的产生，是因为当时[个人电脑](https://zh.wikipedia.org/wiki/%E5%80%8B%E4%BA%BA%E9%9B%BB%E8%85%A6" \o "个人电脑)没有共通的内码，导致厂商推出的中文应用软件无法推广，并且与[IBM 5550](https://zh.wikipedia.org/wiki/IBM_5550" \o "IBM 5550)、[王安码](https://zh.wikipedia.org/w/index.php?title=%E7%8E%8B%E5%AE%89%E7%A2%BC&action=edit&redlink=1" \o "王安码（页面不存在）)等内码，彼此不能兼容；另一方面，台湾当时尚未推出中文编码标准。在这样的时空背景下，为了使台湾早日进入信息时代，所采行的一个项目；同时，这个项目对于以台湾为核心的亚洲繁体汉字圈也产生了久远的影响。

Big5产生前，研发[中文电脑](https://zh.wikipedia.org/wiki/%E4%B8%AD%E6%96%87%E9%9B%BB%E8%85%A6" \o "中文电脑)的[朱邦复](https://zh.wikipedia.org/wiki/%E6%9C%B1%E9%82%A6%E5%BE%A9" \o "朱邦复)认为内码字集应该广纳所有的正异体字，以顾及如户政等应用上的需要，故在当时的内码会议中，建议希望采用他的五万多字的字库。工程师认为虽其技术可行，但是三个字节（超过两个字节）长度的内码却会造成英文显示屏画面映射成中文画面会发生文字无法对齐的问题，因为当时盛行之倚天中文系统画面系以两个字节文字宽度映射成一个中文字图样，英文软件中只要以两个英文字宽度去显示一个中文字，画面就不会乱掉，造成中文系统业者偏爱二个字节长度的内码[[9]](https://zh.wikipedia.org/wiki/%E5%A4%A7%E4%BA%94%E7%A2%BC" \l "cite_note-9)；此外以仓颉输入码压缩成的内码不具排序等功能，因此未被采用。1983年有人诬指朱邦复为[共产党](https://zh.wikipedia.org/wiki/%E5%85%B1%E7%94%A2%E9%BB%A8" \o "共产党)，其研究成果更不可能获采用。[[10]](https://zh.wikipedia.org/wiki/%E5%A4%A7%E4%BA%94%E7%A2%BC" \l "cite_note-10)

3.解决的问题：在Big5码诞生后，大部分台湾的电脑软件都使用了Big5码，加上后来[倚天中文系统](https://zh.wikipedia.org/wiki/%E5%80%9A%E5%A4%A9%E4%B8%AD%E6%96%87%E7%B3%BB%E7%B5%B1" \o "倚天中文系统)的高度普及，使后来的[微软](https://zh.wikipedia.org/wiki/%E5%BE%AE%E8%BD%AF" \o "微软)[Windows 3.x](https://zh.wikipedia.org/wiki/Windows_3.x" \o "Windows 3.x)等亦予以采用。虽然后来台湾还有各种想要取代Big5码，像是倚天中文系统所推行的[倚天码](https://zh.wikipedia.org/wiki/%E5%80%9A%E5%A4%A9%E7%A2%BC" \o "倚天码)、台北市电脑公会所推动的[公会码](https://zh.wikipedia.org/w/index.php?title=%E5%85%AC%E6%9C%83%E7%A2%BC&action=edit&redlink=1" \o "公会码（页面不存在）)等，但是由于Big5字码已沿用多年，因此在习惯不易改变的情况下，始终无法成为主流字码。而台湾后来发展的[国家标准CNS 11643中文标准交换码](https://zh.wikipedia.org/wiki/CNS11643" \o "CNS11643)由于非一般的内码系统，是以交换使用为目的，受先天所限，必须使用至少三个字节来表示一个汉字，所以普及率远远不及Big5码。

在1990年代初期，当[中国大陆](https://zh.wikipedia.org/wiki/%E4%B8%AD%E5%9C%8B%E5%A4%A7%E9%99%B8" \o "中国大陆)的[电子邮件](https://zh.wikipedia.org/wiki/%E9%9B%BB%E9%83%B5" \o "电子邮件)和转码软件还未普遍之时，在[深圳](https://zh.wikipedia.org/wiki/%E6%B7%B1%E5%9C%B3" \o "深圳)的港商和台商公司亦曾经使用Big5系统，以方便与总部的文件交流、以及避免为大陆的办公室再写一套不同内码的系统。使用[简体中文](https://zh.wikipedia.org/wiki/%E7%AE%80%E4%BD%93%E4%B8%AD%E6%96%87" \o "简体中文)的社区，最常用的是[GB 2312](https://zh.wikipedia.org/wiki/GB_2312" \o "GB 2312)、[GBK](https://zh.wikipedia.org/wiki/GBK" \o "GBK)及其后续的[国标码](https://zh.wikipedia.org/wiki/%E5%9B%BD%E5%AE%B6%E6%A0%87%E5%87%86%E4%BB%A3%E7%A0%81" \o "国家标准代码)（[GB 18030](https://zh.wikipedia.org/wiki/GB_18030" \o "GB 18030)）。

除了台湾外，其他使用繁体汉字的地区，如[香港](https://zh.wikipedia.org/wiki/%E9%A6%99%E6%B8%AF" \o "香港)（[香港增补字符集](https://zh.wikipedia.org/wiki/%E9%A6%99%E6%B8%AF%E5%A2%9E%E8%A3%9C%E5%AD%97%E7%AC%A6%E9%9B%86" \o "香港增补字符集)）、[澳门](https://zh.wikipedia.org/wiki/%E6%BE%B3%E9%96%80" \o "澳门)，及使用繁体汉字的海外华人，都曾普遍使用Big5码做为中文内码及交换码。

字节数：Big5码是一套[双字节字符集](https://zh.wikipedia.org/wiki/%E5%8F%8C%E5%AD%97%E8%8A%82%E5%AD%97%E7%AC%A6%E9%9B%86" \o "双字节字符集)，使用了双八码存储方法，以两个字节来安放一个字。第一个字节称为“高位字节”，第二个字节称为“低位字节”。

4.描述：“高位字节”使用了0x81-0xFE，“低位字节”使用了0x40-0x7E，及0xA1-0xFE。在Big5的分区中：

|  |  |
| --- | --- |
| 0x8140-0xA0FE | 保留给用户自定义字符（[造字](https://zh.wikipedia.org/wiki/%E9%80%A0%E5%AD%97" \o "造字)区） |
| 0xA140-0xA3BF | [标点符号](https://zh.wikipedia.org/wiki/%E6%A8%99%E9%BB%9E%E7%AC%A6%E8%99%9F" \o "标点符号)、[希腊字母](https://zh.wikipedia.org/wiki/%E5%B8%8C%E8%85%8A%E5%AD%97%E6%AF%8D" \o "希腊字母)及特殊符号， 包括在0xA259-0xA261，安放了九个[计量用汉字](https://zh.wikipedia.org/wiki/%E8%A8%88%E9%87%8F%E7%94%A8%E6%BC%A2%E5%AD%97" \o "计量用汉字)：兙兛兞兝兡兣嗧瓩糎。 |
| 0xA3C0-0xA3FE | 保留。此区没有开放作造字区用。 |
| 0xA440-0xC67E | [常用汉字](https://zh.wikipedia.org/wiki/%E5%B8%B8%E7%94%A8%E6%BC%A2%E5%AD%97" \o "常用汉字)，先按[笔划](https://zh.wikipedia.org/wiki/%E7%AD%86%E5%8A%83" \o "笔划)再按[部首](https://zh.wikipedia.org/wiki/%E9%83%A8%E9%A6%96" \o "部首)排序。 |
| 0xC6A1-0xC8FE | 保留给用户自定义字符（造字区） |
| 0xC940-0xF9D5 | [次常用汉字](https://zh.wikipedia.org/wiki/%E6%AC%A1%E5%B8%B8%E7%94%A8%E5%AD%97" \o "次常用字)，亦是先按笔划再按部首排序。 |
| 0xF9D6-0xFEFE | 保留给用户自定义字符（造字区） |

值得留意的是，Big5重复收录了两个相同的字：“兀、兀”（0xA461[U+5140]及0xC94A[U+FA0C]）、“嗀、嗀”（0xDCD1[U+55C0]及0xDDFC[U+FA0D]）。此外“十”、“卅”也在符号区又重复了一次，在检索系统中常会造成查询不到字。

5.兼容：倚天Big5延伸[[编辑](https://zh.wikipedia.org/w/index.php?title=%E5%A4%A7%E4%BA%94%E7%A2%BC&action=edit&section=11" \o "编辑小节：倚天Big5延伸)]

[倚天中文系统](https://zh.wikipedia.org/wiki/%E5%80%9A%E5%A4%A9%E4%B8%AD%E6%96%87%E7%B3%BB%E7%B5%B1" \o "倚天中文系统)为与[IBM5550码](https://zh.wikipedia.org/w/index.php?title=IBM5550%E7%A2%BC&action=edit&redlink=1" \o "IBM5550码（页面不存在）)兼容，在Big5码添加了以下字符，称为[倚天扩充字集](https://zh.wikipedia.org/wiki/%E5%80%9A%E5%A4%A9%E6%93%B4%E5%85%85%E5%AD%97%E9%9B%86" \o "倚天扩充字集)：

在0xA3C0-0xA3E0，添加了33个[控制字符](https://zh.wikipedia.org/wiki/%E6%8E%A7%E5%88%B6%E5%AD%97%E7%AC%A6" \o "控制字符)图象。

罕用符号区。在0xC6A1-0xC875，添加了圆形1-10、括号1-10、小写[罗马数字](https://zh.wikipedia.org/wiki/%E7%BE%85%E9%A6%AC%E6%95%B8%E5%AD%97" \o "罗马数字)ⅰ、ⅱ、ⅲ……ⅸ、ⅹ等章节符号、一些[部首](https://zh.wikipedia.org/wiki/%E9%83%A8%E9%A6%96" \o "部首)及[笔划](https://zh.wikipedia.org/wiki/%E7%AD%86%E5%8A%83" \o "笔划)结构，[日语](https://zh.wikipedia.org/wiki/%E6%97%A5%E8%AF%AD" \o "日语)[平假名](https://zh.wikipedia.org/wiki/%E5%B9%B3%E5%81%87%E5%90%8D" \o "平假名)、[片假名](https://zh.wikipedia.org/wiki/%E7%89%87%E5%81%87%E5%90%8D" \o "片假名)及[俄语](https://zh.wikipedia.org/wiki/%E4%BF%84%E8%AF%AD" \o "俄语)使用的[西里尔字母](https://zh.wikipedia.org/wiki/%E8%A5%BF%E9%87%8C%E5%B0%94%E5%AD%97%E6%AF%8D" \o "西里尔字母)。

在0xF9D6-0xF9FE，添加了7个[倚天扩充字](https://zh.wikipedia.org/w/index.php?title=%E5%80%9A%E5%A4%A9%E6%93%B4%E5%85%85%E5%AD%97&action=edit&redlink=1" \o "倚天扩充字（页面不存在）)：碁、銹、裏、墻、恒、粧、嫺和34个制表符号和区块组件。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F |
| F9D0 |  |  |  |  |  |  | 碁 | 銹 | 裏 | 墻 | 恒 | 粧 | 嫺 | ╔ | ╦ | ╗ |
| F9E0 | ╠ | ╬ | ╣ | ╚ | ╩ | ╝ | ╒ | ╤ | ╕ | ╞ | ╪ | ╡ | ╘ | ╧ | ╛ | ╓ |
| F9F0 | ╥ | ╖ | ╟ | ╫ | ╢ | ╙ | ╨ | ╜ | ║ | ═ | ╭ | ╮ | ╰ | ╯ | ▓ |  |

这个延伸有时称为Big5-Eten。由于倚天中文系统是[Windows 95](https://zh.wikipedia.org/wiki/Windows_95" \o "Windows 95)推出之前市场占有率最高的中文系统，此延伸是各种非官方延伸当中最重要的一个。

在后期版本的倚天中文系统中，更加入了一些图案和简体中文字，但未被广泛接受。

Code Page 950[[编辑](https://zh.wikipedia.org/w/index.php?title=%E5%A4%A7%E4%BA%94%E7%A2%BC&action=edit&section=12" \o "编辑小节：Code Page 950)]

[Windows](https://zh.wikipedia.org/wiki/Microsoft_Windows" \o "Microsoft Windows)使用的[Code Page 950](https://zh.wikipedia.org/wiki/Code_Page_950" \o "Code Page 950)（系引用IBM Big 5码的编码页号Code Page 950，简称CP950）之中，只添加了上述0xF9D6-0xF9FE的倚天扩充字及表格符号，并没有加入日文假名字母等其他延伸。

在[Windows ME](https://zh.wikipedia.org/wiki/Windows_ME" \o "Windows ME)之中，微软首度在0xA3E1加入了欧元（[€](https://zh.wikipedia.org/wiki/%E2%82%AC" \o "€)）符号，之后所有Windows版本的Code Page 950也都有这个符号。

中国海字集[[编辑](https://zh.wikipedia.org/w/index.php?title=%E5%A4%A7%E4%BA%94%E7%A2%BC&action=edit&section=13" \o "编辑小节：中国海字集)]

“[中国海字集](https://zh.wikipedia.org/wiki/%E4%B8%AD%E5%9C%8B%E6%B5%B7%E5%AD%97%E9%9B%86" \o "中国海字集)”是中国海公司所出品的繁体汉字造字档。它本身虽然是一套商品，但中国海公司很少将之单独贩售，往往是与其他软件一同销售。例如：[中国海字集](https://zh.wikipedia.org/wiki/%E4%B8%AD%E5%9C%8B%E6%B5%B7%E5%AD%97%E9%9B%86" \o "中国海字集)就曾经与《汉书》、《轻松输入法》等一同发售。由于它包括了不少社会上常见的用字、日文假名、和字等，加上曾与[Office 97](https://zh.wikipedia.org/wiki/Microsoft_Office" \o "Microsoft Office)中文版一并发售，所以比起其他官方Big5延伸，更被台湾民众所接受。香港部分BBS网络在[香港增补字符集](https://zh.wikipedia.org/wiki/%E9%A6%99%E6%B8%AF%E5%A2%9E%E8%A3%9C%E5%AD%97%E7%AC%A6%E9%9B%86" \o "香港增补字符集)未出现之前，一度以中国海字集为标准。

日和字集[[编辑](https://zh.wikipedia.org/w/index.php?title=%E5%A4%A7%E4%BA%94%E7%A2%BC&action=edit&section=14" \o "编辑小节：日和字集)]

“日和字集”乃一香港个人开发的造字档，以兼容[香港增补字符集](https://zh.wikipedia.org/wiki/%E9%A6%99%E6%B8%AF%E5%A2%9E%E8%A3%9C%E5%AD%97%E7%AC%A6%E9%9B%86" \o "香港增补字符集)为卖点，为字集中仍没函盖的[日本汉字](https://zh.wikipedia.org/wiki/%E6%97%A5%E6%9C%AC%E6%BC%A2%E5%AD%97" \o "日本汉字)和[日本国字](https://zh.wikipedia.org/wiki/%E6%97%A5%E6%96%87%E6%BC%A2%E5%AD%97" \l ".E5.9C.8B.E5.AD.97" \o "日文汉字)作增补，并附有仓颉、速成等输入法作辅助。

Unicode补完项目[[编辑](https://zh.wikipedia.org/w/index.php?title=%E5%A4%A7%E4%BA%94%E7%A2%BC&action=edit&section=15" \o "编辑小节：Unicode补完项目)]

主条目：[Unicode补完项目](https://zh.wikipedia.org/wiki/Unicode%E8%A3%9C%E5%AE%8C%E8%A8%88%E7%95%AB" \o "Unicode补完项目)

“Unicode补完项目”前称“BIG5 Extension”，通过修改Microsoft Windows及[Mozilla](https://zh.wikipedia.org/wiki/Mozilla" \o "Mozilla)的编码表，从而用户能在网上传递及交换文字。

有鉴于“中国海字集”的成功，“Unicode补完项目”第二版采用了“中国海字集”原有的造字，再加上“中国海字集”所欠的部分简体中文字及香港粤语用字，建成一个能在Big5及Unicode之间转换的编码表；该计划目前已推出了64位测试版。

# UTF\_8/16

1.来历：**UTF-8**（**8-bit Unicode Transformation Format**）是一种针对[Unicode](https://zh.wikipedia.org/wiki/Unicode" \o "Unicode)的可变长度[字符编码](https://zh.wikipedia.org/wiki/%E5%AD%97%E5%85%83%E7%B7%A8%E7%A2%BC" \o "字符编码)，也是一种[前缀码](https://zh.wikipedia.org/wiki/%E5%89%8D%E7%BC%80%E7%A0%81" \o "前缀码)。它可以用来表示Unicode标准中的任何字符，且其编码中的第一个[字节](https://zh.wikipedia.org/wiki/%E5%AD%97%E8%8A%82" \o "字节)仍与[ASCII](https://zh.wikipedia.org/wiki/ASCII" \o "ASCII)兼容，这使得原来处理ASCII字符的[软件](https://zh.wikipedia.org/wiki/%E8%BB%9F%E9%AB%94" \o "软件)无须或只须做少部分修改，即可继续使用。因此，它逐渐成为[电子邮件](https://zh.wikipedia.org/wiki/%E9%9B%BB%E5%AD%90%E9%83%B5%E4%BB%B6" \o "电子邮件)、[网页](https://zh.wikipedia.org/wiki/%E7%B6%B2%E9%A0%81" \o "网页)及其他[存储](https://zh.wikipedia.org/wiki/%E5%84%B2%E5%AD%98%E8%A3%9D%E7%BD%AE" \o "存储设备)或发送文字的应用中，优先采用的编码。

产生的原因：1992年初，为创建良好的[字节串编码系统](https://zh.wikipedia.org/w/index.php?title=%E4%BD%8D%E5%85%83%E7%B5%84%E4%B8%B2%E7%B7%A8%E7%A2%BC%E7%B3%BB%E7%B5%B1&action=edit&redlink=1" \o "字节串编码系统（页面不存在）)以供多[字节](https://zh.wikipedia.org/wiki/%E5%AD%97%E8%8A%82" \o "字节)字符集使用，开始了一个正式的研究。[ISO/IEC 10646](https://zh.wikipedia.org/wiki/%E9%80%9A%E7%94%A8%E5%AD%97%E7%AC%A6%E9%9B%86" \o "通用字符集)的初稿中有一个非必须的附录，名为UTF。当中包含了一个供32[比特](https://zh.wikipedia.org/wiki/%E4%BD%8D%E5%85%83" \o "比特)的[字符](https://zh.wikipedia.org/wiki/UCS" \o "UCS)使用的字节串编码系统。这个编码方式的性能并不令人满意，但它提出了将0-127的范围保留给ASCII以兼容旧系统的概念。

1992年7月，[X/Open](https://zh.wikipedia.org/wiki/X/Open" \o "X/Open)委员会XoJIG开始寻求一个较佳的编码系统。[Unix系统实验室](https://zh.wikipedia.org/wiki/Unix%E7%B3%BB%E7%BB%9F%E5%AE%9E%E9%AA%8C%E5%AE%A4" \o "Unix系统实验室)（USL）的Dave Prosser为此提出了一个编码系统的建议。它具备可更快速实现的特性，并引入一项新的改进。其中，7[比特](https://zh.wikipedia.org/wiki/%E4%BD%8D%E5%85%83" \o "比特)的[ASCII](https://zh.wikipedia.org/wiki/ASCII" \o "ASCII)符号只代表原来的意思，所有多字节序列则会包含第8[比特](https://zh.wikipedia.org/wiki/%E4%BD%8D%E5%85%83" \o "比特)的符号，也就是所谓的[最高有效比特](https://zh.wikipedia.org/wiki/%E6%9C%80%E9%AB%98%E6%9C%89%E6%95%88%E4%BD%8D%E5%85%83" \o "最高有效比特)。

1992年8月，这个建议由[IBM](https://zh.wikipedia.org/wiki/IBM" \o "IBM)[X/Open](https://zh.wikipedia.org/wiki/X/Open" \o "X/Open)的代表流传到一些感兴趣的团体。与此同时，[贝尔实验室](https://zh.wikipedia.org/wiki/%E8%B2%9D%E7%88%BE%E5%AF%A6%E9%A9%97%E5%AE%A4" \o "贝尔实验室)[九号项目](https://zh.wikipedia.org/wiki/%E8%B2%9D%E7%88%BE%E5%AF%A6%E9%A9%97%E5%AE%A4%E4%B9%9D%E8%99%9F%E8%A8%88%E7%95%AB" \o "贝尔实验室九号项目)[操作系统](https://zh.wikipedia.org/wiki/%E4%BD%9C%E6%A5%AD%E7%B3%BB%E7%B5%B1" \o "操作系统)工作小组的[肯·汤普逊](https://zh.wikipedia.org/wiki/%E8%82%AF%C2%B7%E6%B1%A4%E6%99%AE%E9%80%8A" \o "肯·汤普逊)对这编码系统作出重大的修改，让编码可以自我同步，使得不必从字符串的开首读取，也能找出字符间的分界。1992年9月2日，[肯·汤普逊](https://zh.wikipedia.org/wiki/%E8%82%AF%C2%B7%E6%B1%A4%E6%99%AE%E9%80%8A" \o "肯·汤普逊)和[罗勃·派克](https://zh.wikipedia.org/wiki/%E7%BE%85%E5%8B%83%C2%B7%E6%B4%BE%E5%85%8B" \o "罗勃·派克)一起在[美国](https://zh.wikipedia.org/wiki/%E7%BE%8E%E5%9C%8B" \o "美国)[新泽西州](https://zh.wikipedia.org/wiki/%E6%96%B0%E6%BE%A4%E8%A5%BF%E5%B7%9E" \o "新泽西州)一架餐车的餐桌垫上描绘出此设计的要点。接下来的日子，Pike及汤普逊将它实现，并将这编码系统完全应用在[九号项目](https://zh.wikipedia.org/wiki/%E8%B2%9D%E7%88%BE%E5%AF%A6%E9%A9%97%E5%AE%A4%E4%B9%9D%E8%99%9F%E8%A8%88%E7%95%AB" \o "贝尔实验室九号项目)当中，及后他将有关成果回馈X/Open。

1993年1月25-29日的在[圣地牙哥](https://zh.wikipedia.org/wiki/%E8%81%96%E5%9C%B0%E7%89%99%E5%93%A5" \o "圣地牙哥)举行的[USENIX](https://zh.wikipedia.org/wiki/USENIX" \o "USENIX)会议首次正式介绍UTF-8。

自1996年起，[微软](https://zh.wikipedia.org/wiki/%E5%BE%AE%E8%BB%9F" \o "微软)的[CAB](https://zh.wikipedia.org/wiki/CAB" \o "CAB)（MS Cabinet）规格在UTF-8标准正式落实前就明确容许在任何地方使用UTF-8编码系统。但有关的编码器实际上从来没有实现这方面的规格。

2.解决的问题：UTF-8使用一至六个[字节](https://zh.wikipedia.org/wiki/%E5%AD%97%E8%8A%82" \o "字节)为每个字符编码（尽管如此，2003年11月UTF-8被RFC 3629重新规范，只能使用原来Unicode定义的区域，U+0000到U+10FFFF，也就是说最多四个字节）：

1. 128个US-ASCII字符只需一个字节编码（Unicode范围由U+0000至U+007F）。
2. 带有[附加符号](https://zh.wikipedia.org/wiki/%E9%99%84%E5%8A%A0%E7%AC%A6%E5%8F%B7" \o "附加符号)的[拉丁文](https://zh.wikipedia.org/wiki/%E6%8B%89%E4%B8%81%E6%96%87" \o "拉丁文)、[希腊文](https://zh.wikipedia.org/wiki/%E5%B8%8C%E8%87%98%E6%96%87" \o "希腊文)、[西里尔字母](https://zh.wikipedia.org/wiki/%E8%A5%BF%E9%87%8C%E7%88%BE%E5%AD%97%E6%AF%8D" \o "西里尔字母)、[亚美尼亚语](https://zh.wikipedia.org/wiki/%E4%BA%9E%E7%BE%8E%E5%B0%BC%E4%BA%9E%E8%AA%9E" \o "亚美尼亚语)、[希伯来文](https://zh.wikipedia.org/wiki/%E5%B8%8C%E4%BC%AF%E4%BE%86%E6%96%87" \o "希伯来文)、[阿拉伯文](https://zh.wikipedia.org/wiki/%E9%98%BF%E6%8B%89%E4%BC%AF%E6%96%87" \o "阿拉伯文)、[叙利亚文](https://zh.wikipedia.org/wiki/%E5%8F%99%E5%88%A9%E4%BA%9A%E6%96%87" \o "叙利亚文)及[它拿字母](https://zh.wikipedia.org/wiki/%E5%AE%83%E6%8B%BF%E5%AD%97%E6%AF%8D" \o "它拿字母)则需要两个字节编码（Unicode范围由U+0080至U+07FF）。
3. 其他[基本多文种平面](https://zh.wikipedia.org/wiki/%E5%9F%BA%E6%9C%AC%E5%A4%9A%E6%96%87%E7%A8%AE%E5%B9%B3%E9%9D%A2" \o "基本多文种平面)（BMP）中的字符（这包含了大部分常用字，如大部分的汉字）使用三个字节编码（Unicode范围由U+0800至U+FFFF）。
4. 其他极少使用的Unicode [辅助平面](https://zh.wikipedia.org/wiki/%E8%BC%94%E5%8A%A9%E5%B9%B3%E9%9D%A2" \o "辅助平面)的字符使用四至六字节编码（Unicode范围由U+10000至U+1FFFFF使用四字节，Unicode范围由U+200000至U+3FFFFFF使用五字节，Unicode范围由U+4000000至U+7FFFFFFF使用六字节）。

对上述提及的第四种字符而言，UTF-8使用四至六个字节来编码似乎太耗费资源了。但UTF-8对所有常用的字符都可以用三个字节表示，而且它的另一种选择，[UTF-16编码](https://zh.wikipedia.org/wiki/UTF-16" \o "UTF-16)，对前述的第四种字符同样需要四个字节来编码，所以要决定UTF-8或UTF-16哪种编码比较有效率，还要视所使用的字符的分布范围而定。不过，如果使用一些传统的压缩系统，比如[DEFLATE](https://zh.wikipedia.org/wiki/DEFLATE" \o "DEFLATE)，则这些不同编码系统间的的差异就变得微不足道了。若顾及传统压缩算法在压缩较短文字上的效果不大，可以考虑使用[Unicode标准压缩格式](https://zh.wikipedia.org/w/index.php?title=Unicode%E6%A8%99%E6%BA%96%E5%A3%93%E7%B8%AE%E6%A0%BC%E5%BC%8F&action=edit&redlink=1" \o "Unicode标准压缩格式（页面不存在）)（SCSU）。

[互联网工程工作小组](https://zh.wikipedia.org/wiki/%E7%B6%B2%E9%9A%9B%E7%B6%B2%E8%B7%AF%E5%B7%A5%E7%A8%8B%E5%B7%A5%E4%BD%9C%E5%B0%8F%E7%B5%84" \o "互联网工程工作小组)（IETF）要求所有[互联网](https://zh.wikipedia.org/wiki/%E7%B6%B2%E9%9A%9B%E7%B6%B2%E8%B7%AF" \o "互联网)[协议](https://zh.wikipedia.org/wiki/%E7%BD%91%E7%BB%9C%E5%8D%8F%E8%AE%AE" \o "网络协议)都必须支持UTF-8编码[[1]](https://zh.wikipedia.org/wiki/UTF-8" \l "cite_note-1)。[互联网邮件联盟](https://zh.wikipedia.org/w/index.php?title=%E4%BA%92%E8%81%AF%E7%B6%B2%E9%83%B5%E4%BB%B6%E8%81%AF%E7%9B%9F&action=edit&redlink=1" \o "互联网邮件联盟（页面不存在）)（IMC）建议所有电子邮件软件都支持UTF-8编码。[[1]](https://web.archive.org/web/20071026103104/http://www.imc.org/mail-i18n.html)

3.字节数：UTF-8是UNICODE的一种变长度的编码表达方式《一般UNICODE为双位元组（指UCS2）》，它由[肯·汤普逊](https://zh.wikipedia.org/wiki/%E8%82%AF%C2%B7%E6%B1%A4%E6%99%AE%E9%80%8A" \o "肯·汤普逊)（Ken Thompson）于1992年创建，现在已经标准化为RFC 3629。UTF-8就是以8位为单元对UCS进行编码，而UTF-8不使用大尾序和小尾序的形式，每个使用UTF-8存储的字符，除了第一个字节外，其余字节的头两个比特都是以"10"开始，使文字处理器能够较快地找出每个字符的开始位置。

4.描述：目前有好几份关于UTF-8详细规格的文件，但这些文件在定义上有些许的不同：

* [RFC 3629](https://tools.ietf.org/html/rfc3629) / STD 63（2003），这份文件制定了UTF-8是标准的[互联网](https://zh.wikipedia.org/wiki/%E7%B6%B2%E9%9A%9B%E7%B6%B2%E8%B7%AF" \o "互联网)[协议](https://zh.wikipedia.org/wiki/%E7%BD%91%E7%BB%9C%E5%8D%8F%E8%AE%AE" \o "网络协议)元素
* *第四版，The Unicode Standard*，§3.9－§3.10（2003）
* ISO/IEC 10646-1:2000附加文件D（2000）

它们取代了以下那些被淘汰的定义：

* ISO/IEC 10646-1:1993修正案2／附加文件R（1996）
* *第二版，The Unicode Standard*，附录A（1996）
* [RFC 2044](https://tools.ietf.org/html/rfc2044)（1996）
* [RFC 2279](https://tools.ietf.org/html/rfc2279)（1998）
* *第三版，The Unicode Standard*，§2.3（2000）及勘误表#1：UTF-8 Shortest Form（2000）
* *Unicode Standard附加文件#27: Unicode 3.1*（2001）

事实上，所有定义的基本原理都是相同的，它们之间最主要的不同是支持的字符范围及无效输入的处理方法。

Unicode[字符](https://zh.wikipedia.org/wiki/UCS" \o "UCS)的[比特](https://zh.wikipedia.org/wiki/%E4%BD%8D%E5%85%83" \o "比特)被分区为数个部分，并分配到UTF-8的[字节](https://zh.wikipedia.org/wiki/%E5%AD%97%E8%8A%82" \o "字节)串中较低的[比特](https://zh.wikipedia.org/wiki/%E4%BD%8D%E5%85%83" \o "比特)的位置。在U+0080的以下[字符](https://zh.wikipedia.org/wiki/UCS" \o "UCS)都使用内含其字符的单[字节](https://zh.wikipedia.org/wiki/%E5%AD%97%E8%8A%82" \o "字节)编码。这些编码正好对应7比特的ASCII字符。在其他情况，有可能需要多达4个字符组来表示一个字符。这些多字节的[最高有效比特](https://zh.wikipedia.org/wiki/%E6%9C%80%E9%AB%98%E6%9C%89%E6%95%88%E4%BD%8D%E5%85%83" \o "最高有效比特)会设置成1，以防止与7比特的ASCII字符混淆，并保持标准的字节主导字符串运作顺利。

|  |  |  |  |
| --- | --- | --- | --- |
| **代码范围 [十六进制](https://zh.wikipedia.org/wiki/%E5%8D%81%E5%85%AD%E9%80%B2%E5%88%B6" \o "十六进制)** | **标量值（scalar value） [二进制](https://zh.wikipedia.org/wiki/%E4%BA%8C%E9%80%B2%E5%88%B6" \o "二进制)** | **UTF-8 [二进制](https://zh.wikipedia.org/wiki/%E4%BA%8C%E9%80%B2%E5%88%B6" \o "二进制)／[十六进制](https://zh.wikipedia.org/wiki/%E5%8D%81%E5%85%AD%E9%80%B2%E5%88%B6" \o "十六进制)** | **注释** |
| 000000 - 00007F 128个代码 | 00000000 00000000 0zzzzzzz | 0zzzzzzz（00-7F） | ASCII字符范围，字节由零开始 |
| 七个z | 七个z |
| 000080 - 0007FF 1920个代码 | 00000000 00000yyy yyzzzzzz | 110yyyyy（C0-DF) 10zzzzzz(80-BF） | 第一个字节由110开始，接着的字节由10开始 |
| 三个y；二个y；六个z | 五个y；六个z |
| 000800 - 00D7FF 00E000 - 00FFFF 61440个代码 [[Note 1]](https://zh.wikipedia.org/wiki/UTF-8" \l "endnote_D800Note_1) | 00000000 xxxxyyyy yyzzzzzz | 1110xxxx(E0-EF) 10yyyyyy 10zzzzzz | 第一个字节由1110开始，接着的字节由10开始 |
| 四个x；四个y；二个y；六个z | 四个x；六个y；六个z |
| 010000 - 10FFFF 1048576个代码 | 000wwwxx xxxxyyyy yyzzzzzz | 11110www(F0-F7) 10xxxxxx 10yyyyyy 10zzzzzz | 将由11110开始，接着的字节由10开始 |
| 三个w；二个x；四个x；四个y；二个y；六个z | 三个w；六个x；六个y；六个z |

**[Note 1](https://zh.wikipedia.org/wiki/UTF-8" \l "ref_D800Note_1)** Unicode在范围D800-DFFF中不存在任何字符，[基本多文种平面](https://zh.wikipedia.org/wiki/%E5%9F%BA%E6%9C%AC%E5%A4%9A%E6%96%87%E7%A8%AE%E5%B9%B3%E9%9D%A2" \o "基本多文种平面)中约定了这个范围用于UTF-16扩展标识[辅助平面](https://zh.wikipedia.org/wiki/%E8%BE%85%E5%8A%A9%E5%B9%B3%E9%9D%A2" \o "辅助平面)（两个UTF-16表示一个[辅助平面](https://zh.wikipedia.org/wiki/%E8%BE%85%E5%8A%A9%E5%B9%B3%E9%9D%A2" \o "辅助平面)字符）。当然，任何编码都是可以被转换到这个范围，但在unicode中他们并不代表任何合法的值。

例如，希伯来语字母aleph（א）的Unicode代码是U+05D0，按照以下方法改成UTF-8：

* 它属于U+0080到U+07FF区域，这个表说明它使用双字节，*110*yyyyy *10*zzzzzz.
* [十六进制](https://zh.wikipedia.org/wiki/%E5%8D%81%E5%85%AD%E8%BF%9B%E5%88%B6" \o "十六进制)的0x05D0换算成[二进制](https://zh.wikipedia.org/wiki/%E4%BA%8C%E8%BF%9B%E5%88%B6" \o "二进制)就是101-1101-0000.
* 这11位数按顺序放入"y"部分和"z"部分：110**10111** 10**010000**.
* 最后结果就是双字节，用十六进制写起来就是0xD7 0x90，这就是这个字符aleph（א）的UTF-8编码。

所以开始的128个字符（US-ASCII）只需一字节，接下来的1920个字符需要双字节编码，包括带[附加符号](https://zh.wikipedia.org/wiki/%E9%99%84%E5%8A%A0%E7%AC%A6%E5%8F%B7" \o "附加符号)的[拉丁字母](https://zh.wikipedia.org/wiki/%E6%8B%89%E4%B8%81%E5%AD%97%E6%AF%8D" \o "拉丁字母)，[希腊字母](https://zh.wikipedia.org/wiki/%E5%B8%8C%E8%85%8A%E5%AD%97%E6%AF%8D" \o "希腊字母)，[西里尔字母](https://zh.wikipedia.org/wiki/%E8%A5%BF%E9%87%8C%E5%B0%94%E5%AD%97%E6%AF%8D" \o "西里尔字母)，[科普特语](https://zh.wikipedia.org/wiki/%E7%A7%91%E6%99%AE%E7%89%B9%E8%AF%AD" \o "科普特语)字母，[亚美尼亚语](https://zh.wikipedia.org/wiki/%E4%BA%9A%E7%BE%8E%E5%B0%BC%E4%BA%9A%E8%AF%AD" \o "亚美尼亚语)字母，[希伯来文](https://zh.wikipedia.org/wiki/%E5%B8%8C%E4%BC%AF%E6%9D%A5%E6%96%87" \o "希伯来文)字母和[阿拉伯字母](https://zh.wikipedia.org/wiki/%E9%98%BF%E6%8B%89%E4%BC%AF%E5%AD%97%E6%AF%8D" \o "阿拉伯字母)的字符。[基本多文种平面](https://zh.wikipedia.org/wiki/%E5%9F%BA%E6%9C%AC%E5%A4%9A%E6%96%87%E7%A8%AE%E5%B9%B3%E9%9D%A2" \o "基本多文种平面)中其余的字符使用三个字节，剩余字符使用四个字节。

根据这种方式可以处理更大数量的字符。原来的规范允许长达6字节的序列，可以覆盖到31位（[通用字符集](https://zh.wikipedia.org/wiki/%E9%80%9A%E7%94%A8%E5%AD%97%E7%AC%A6%E9%9B%86" \o "通用字符集)原来的极限）。尽管如此，2003年11月UTF-8被RFC 3629重新规范，只能使用原来Unicode定义的区域，U+0000到U+10FFFF。根据这些规范，以下字节值将无法出现在合法UTF-8序列中：

|  |  |  |
| --- | --- | --- |
| **编码（[二进制](https://zh.wikipedia.org/wiki/%E4%BA%8C%E8%BF%9B%E5%88%B6" \o "二进制)）** | **编码（[十六进制](https://zh.wikipedia.org/wiki/%E5%8D%81%E5%85%AD%E8%BF%9B%E5%88%B6" \o "十六进制)）** | **注释** |
| 1100000x | C0, C1 | 过长编码：双字节序列的头字节，但码点 <= 127 |
| 1111111x | FE, FF | 无法达到：7或8字节序列的头字节 |
| 111110xx 1111110x | F8, F9, FA, FB, FC, FD | 被RFC 3629规范：5或6字节序列的头字节 |
| 11110101 1111011x | F5, F6, F7 | 被RFC 3629规范：码点超过10FFFF的头字节 |

5.兼容：UCS字符U+0000到U+007F（ASCII）被编码为字节0x00到0x7F（ASCII兼容），这也意味着只包含7位ASCII字符的文件在ASCII和UTF-8两种编码方式下是一样的。

* 所有>U+007F的UCS字符被编码为一个多个字节的串，每个字节都有标记位集。因此，ASCII字节（0x00-0x7F）不可能作为任何其他字符的一部分。
* 表示非ASCII字符的多字节串的第一个字节总是在0xC0到0xFD的范围里，并指出这个字符包含多少个字节。多字节串的其余字节都在0x80到0xBF范围里，这使得重新同步非常容易，并使编码无国界，且很少受丢失字节的影响。
* 可以编入所有可能的231个UCS代码
* UTF-8编码字符理论上可以最多到6个字节长，然而16位BMP字符最多只用到3字节长。
* Bigendian UCS-4字节串的排列顺序是预定的。
* 字节0xFE和0xFF在UTF-8编码中从未用到，同时，UTF-8以字节为编码单元，它的字节顺序在所有系统中都是一様的，没有字节序的问题，也因此它实际上并不需要[BOM](https://zh.wikipedia.org/wiki/%E4%BD%8D%E5%85%83%E7%B5%84%E9%A0%86%E5%BA%8F%E8%A8%98%E8%99%9F" \o "字节顺序记号)。
* 与UTF-16或其他Unicode编码相比，对于不支持Unicode和XML的系统，UTF-8更不容易造成问题。

# Unicode

1.来历：**Unicode**（[中文](https://zh.wikipedia.org/wiki/%E4%B8%AD%E6%96%87" \o "中文)：万国码、国际码、统一码、单一码）是[计算机科学](https://zh.wikipedia.org/wiki/%E9%9B%BB%E8%85%A6%E7%A7%91%E5%AD%B8" \o "计算机科学)领域里的一项业界标准。它对世界上大部分的[文字系统](https://zh.wikipedia.org/wiki/%E6%96%87%E5%AD%97%E7%B3%BB%E7%B5%B1" \o "文字系统)进行了整理、编码，使得电脑可以用更为简单的方式来呈现和处理文字。

Unicode伴随着[通用字符集](https://zh.wikipedia.org/wiki/%E9%80%9A%E7%94%A8%E5%AD%97%E7%AC%A6%E9%9B%86" \o "通用字符集)的标准而发展，同时也以书本的形式[[1]](https://zh.wikipedia.org/wiki/Unicode" \l "cite_note-1)对外发表。Unicode至今仍在不断增修，每个新版本都加入更多新的字符。目前最新的版本为2017年6月20日公布的10.0.0[[2]](https://zh.wikipedia.org/wiki/Unicode" \l "cite_note-Unicode10.0-2)，已经收录超过十万个[字符](https://zh.wikipedia.org/wiki/%E5%AD%97%E7%AC%A6" \o "字符)（第十万个字符在2005年获采纳）。Unicode涵盖的数据除了视觉上的字形、编码方法、标准的[字符编码](https://zh.wikipedia.org/wiki/%E5%AD%97%E7%AC%A6%E7%BC%96%E7%A0%81" \o "字符编码)外，还包含了字符特性，如大小写字母。

Unicode发展由非营利机构[统一码联盟](https://zh.wikipedia.org/wiki/%E7%B5%B1%E4%B8%80%E7%A2%BC%E8%81%AF%E7%9B%9F" \o "统一码联盟)负责，该机构致力于让Unicode方案取代既有的字符编码方案。因为既有的方案往往空间非常有限，亦不适用于[多语](https://zh.wikipedia.org/wiki/%E5%A4%9A%E8%AA%9E" \o "多语)环境。

Unicode备受认可，并广泛地应用于电脑软件的[国际化与本地化](https://zh.wikipedia.org/wiki/%E5%9C%8B%E9%9A%9B%E5%8C%96%E8%88%87%E6%9C%AC%E5%9C%B0%E5%8C%96" \o "国际化与本地化)过程。有很多新科技，如[可扩展置标语言](https://zh.wikipedia.org/wiki/%E5%8F%AF%E6%89%A9%E5%B1%95%E7%BD%AE%E6%A0%87%E8%AF%AD%E8%A8%80" \o "可扩展置标语言)(Extensible Markup Language，简称：XML)、[Java编程语言](https://zh.wikipedia.org/wiki/Java" \o "Java)以及现代的[操作系统](https://zh.wikipedia.org/wiki/%E4%BD%9C%E6%A5%AD%E7%B3%BB%E7%B5%B1" \o "操作系统)，都采用Unicode编码。

产生原因：Unicode是为了解决传统的[字符编码](https://zh.wikipedia.org/wiki/%E5%AD%97%E5%85%83%E7%B7%A8%E7%A2%BC" \o "字符编码)方案的局限而产生的，例如[ISO 8859-1](https://zh.wikipedia.org/wiki/ISO/IEC_8859" \o "ISO/IEC 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://zh.wikipedia.org/wiki/%E6%8B%89%E4%B8%81%E5%AD%97%E6%AF%8D" \o "拉丁字母)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）。

2.解决的问题：Unicode编码包含了不同写法的字，如“ɑ／a”、“強／强”、“戶／户／戸”。然而在[汉字](https://zh.wikipedia.org/wiki/%E6%B1%89%E5%AD%97" \o "汉字)方面引起了一字多形的认定争议（详见[中日韩统一表意文字](https://zh.wikipedia.org/wiki/%E4%B8%AD%E6%97%A5%E9%9F%93%E7%B5%B1%E4%B8%80%E8%A1%A8%E6%84%8F%E6%96%87%E5%AD%97" \o "中日韩统一表意文字)主题）。

在文字处理方面，统一码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。

目前，几乎所有电脑系统都支持基本拉丁字母，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://zh.wikipedia.org/wiki/%E5%85%A8%E5%BD%A2" \o "全角)格式区块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://zh.wikipedia.org/wiki/%E5%9F%BA%E6%9C%AC%E5%A4%9A%E6%96%87%E7%A8%AE%E5%B9%B3%E9%9D%A2" \o ")（英文：Basic Multilingual Plane，简写BMP。又称为“零号平面”、plane 0）里的所有字符，要用四个数字（即两个char,16bit ,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个或六个数字。旧版的Unicode标准使用相近的标记方法，但却有些微小差异：在Unicode 3.0里使用“U-”然后紧接着八个数字，而“U+”则必须随后紧接着四个数字。

3.字节数：统一码的编码方式与[ISO 10646](https://zh.wikipedia.org/wiki/ISO_10646" \o "ISO 10646)的[通用字符集](https://zh.wikipedia.org/wiki/%E9%80%9A%E7%94%A8%E5%AD%97%E7%AC%A6%E9%9B%86" \o "通用字符集)概念相对应。目前实际应用的统一码版本对应于[UCS-2](https://zh.wikipedia.org/wiki/UCS-2" \o "UCS-2)，使用16[位](https://zh.wikipedia.org/wiki/%E4%BD%8D%E5%85%83" \o "比特)的编码空间。也就是每个字符占用2个[字节](https://zh.wikipedia.org/wiki/%E5%AD%97%E8%8A%82" \o "字节)。这样理论上一共最多可以表示216（即65536）个字符。基本满足各种语言的使用。实际上当前版本的统一码并未完全使用这16位编码，而是保留了大量空间以作为特殊使用或将来扩展。

上述16位统一码字符构成[基本多文种平面](https://zh.wikipedia.org/wiki/%E5%9F%BA%E6%9C%AC%E5%A4%9A%E6%96%87%E7%A8%AE%E5%B9%B3%E9%9D%A2" \o "基本多文种平面)。最新（但未实际广泛使用）的统一码版本定义了16个[辅助平面](https://zh.wikipedia.org/wiki/%E8%BE%85%E5%8A%A9%E5%B9%B3%E9%9D%A2" \o "辅助平面)，两者合起来至少需要占据21位的编码空间，比3字节略少。但事实上辅助平面字符仍然占用4字节编码空间，与[UCS-4](https://zh.wikipedia.org/wiki/UCS-4" \o "UCS-4)保持一致。未来版本会扩充到ISO 10646-1实现级别3，即涵盖UCS-4的所有字符。UCS-4是一个更大的尚未填充完全的31位字符集，加上恒为0的首位，共需占据32位，即4字节。理论上最多能表示231个字符，完全可以涵盖一切语言所用的符号。

基本多文种平面的字符的编码为*U+hhhh*，其中每个*h*代表一个[十六进制](https://zh.wikipedia.org/wiki/%E5%8D%81%E5%85%AD%E8%BF%9B%E5%88%B6" \o "十六进制)数字，与UCS-2编码完全相同。而其对应的4字节UCS-4编码后两个字节一致，前两个字节则所有位均为0。

4.描述：Unicode的实现方式不同于编码方式。一个字符的Unicode编码是确定的。但是在实际传输过程中，由于不同[系统平台](https://zh.wikipedia.org/wiki/%E7%B3%BB%E7%BB%9F%E5%B9%B3%E5%8F%B0" \o "系统平台)的设计不一定一致，以及出于节省空间的目的，对Unicode编码的实现方式有所不同。Unicode的实现方式称为**Unicode转换格式**（Unicode Transformation Format，简称为UTF）

例如，如果一个仅包含基本7位[ASCII](https://zh.wikipedia.org/wiki/ASCII" \o "ASCII)字符的Unicode文件，如果每个字符都使用2字节的原Unicode编码传输，其第一字节的8位始终为0。这就造成了比较大的浪费。对于这种情况，可以使用UTF-8编码，这是一种变长编码，它将基本7位ASCII字符仍用7位编码表示，占用一个字节（首位补0）。而遇到与其他Unicode字符混合的情况，将按一定算法转换，每个字符使用1-3个字节编码，并利用首位为0或1进行识别。这样对以7位ASCII字符为主的西文文档就大幅节省了编码长度（具体方案参见*[UTF-8](https://zh.wikipedia.org/wiki/UTF-8" \o "UTF-8)*）。类似的，对未来会出现的需要4个字节的辅助平面字符和其他UCS-4扩充字符，2字节编码的[UTF-16](https://zh.wikipedia.org/wiki/UTF-16" \o "UTF-16)也需要通过一定的算法进行转换。

再如，如果直接使用与Unicode编码一致（仅限于BMP字符）的UTF-16编码，由于每个字符占用了两个字节，在[麦金塔电脑](https://zh.wikipedia.org/wiki/%E9%BA%A5%E9%87%91%E5%A1%94%E9%9B%BB%E8%85%A6" \o "麦金塔电脑)（[Mac](https://zh.wikipedia.org/wiki/Mac" \o "Mac)）机和[个人电脑](https://zh.wikipedia.org/wiki/%E5%80%8B%E4%BA%BA%E9%9B%BB%E8%85%A6" \o "个人电脑)上，对字节顺序的理解是不一致的。这时同一字节流可能会被解释为不同内容，如某字符为十六进制编码4E59，按两个字节拆分为4E和59，在Mac上读取时是从低字节开始，那么在Mac OS会认为此4E59编码为594E，找到的字符为“奎”，而在Windows上从高字节开始读取，则编码为U+4E59的字符为“乙”。就是说在Windows下以UTF-16编码保存一个字符“乙”，在Mac OS环境下打开会显示成“奎”。此类情况说明UTF-16的编码顺序若不加以人为定义就可能发生混淆，于是在UTF-16编码实现方式中使用了[大端序](https://zh.wikipedia.org/wiki/%E5%AD%97%E8%8A%82%E5%BA%8F" \o "字节序)（Big-Endian，简写为UTF-16 BE）、[小端序](https://zh.wikipedia.org/wiki/%E5%AD%97%E8%8A%82%E5%BA%8F" \o "字节序)（Little-Endian，简写为UTF-16 LE）的概念，以及可附加的[字节顺序记号](https://zh.wikipedia.org/wiki/%E4%BD%8D%E5%85%83%E7%B5%84%E9%A0%86%E5%BA%8F%E8%A8%98%E8%99%9F" \o "字节顺序记号)解决方案，目前在PC机上的Windows系统和Linux系统对于UTF-16编码默认使用UTF-16 LE。（具体方案参见*[UTF-16](https://zh.wikipedia.org/wiki/UTF-16" \o "UTF-16)*）

此外Unicode的实现方式还包括[UTF-7](https://zh.wikipedia.org/wiki/UTF-7" \o "UTF-7)、[Punycode](https://zh.wikipedia.org/wiki/Punycode" \o "Punycode)、[CESU-8](https://zh.wikipedia.org/w/index.php?title=CESU-8&action=edit&redlink=1" \o "CESU-8（页面不存在）)、[SCSU](https://zh.wikipedia.org/w/index.php?title=SCSU&action=edit&redlink=1" \o "SCSU（页面不存在）)、[UTF-32](https://zh.wikipedia.org/wiki/UTF-32" \o "UTF-32)、[GB18030](https://zh.wikipedia.org/wiki/GB18030" \o "GB18030)等，这些实现方式有些仅在一定的国家和地区使用，有些则属于未来的规划方式。目前通用的实现方式是UTF-16小端序（LE）、UTF-16大端序（BE）和UTF-8。在微软公司[Windows XP](https://zh.wikipedia.org/wiki/Windows_XP" \o "Windows XP)附带的[记事本](https://zh.wikipedia.org/wiki/%E8%AE%B0%E4%BA%8B%E6%9C%AC" \o "记事本)（Notepad）中，“另存为”对话框可以选择的四种编码方式除去非Unicode编码的[ANSI](https://zh.wikipedia.org/wiki/ANSI" \o "ANSI)（对于英文系统即[ASCII](https://zh.wikipedia.org/wiki/ASCII" \o "ASCII)编码，中文系统则为[GB2312](https://zh.wikipedia.org/wiki/GB2312" \o "GB2312)或[Big5](https://zh.wikipedia.org/wiki/Big5" \o "Big5)编码）外，其余三种为“Unicode”（对应UTF-16 LE）、“Unicode big endian”（对应UTF-16 BE）和“UTF-8”。

目前辅助平面的工作主要集中在第二和第三平面的[中日韩统一表意文字](https://zh.wikipedia.org/wiki/%E4%B8%AD%E6%97%A5%E9%9F%A9%E7%BB%9F%E4%B8%80%E8%A1%A8%E6%84%8F%E6%96%87%E5%AD%97" \o "中日韩统一表意文字)中，因此包括[GBK](https://zh.wikipedia.org/wiki/GBK" \o "GBK)、[GB18030](https://zh.wikipedia.org/wiki/GB18030" \o "GB18030)、[Big5](https://zh.wikipedia.org/wiki/Big5" \o "Big5)等[简体中文](https://zh.wikipedia.org/wiki/%E7%AE%80%E4%BD%93%E4%B8%AD%E6%96%87" \o "简体中文)、[繁体中文](https://zh.wikipedia.org/wiki/%E7%B9%81%E4%BD%93%E4%B8%AD%E6%96%87" \o "繁体中文)、[日文](https://zh.wikipedia.org/wiki/%E6%97%A5%E6%96%87" \o ")、[韩文](https://zh.wikipedia.org/wiki/%E9%9F%A9%E6%96%87" \o "韩文)以及越南[喃字](https://zh.wikipedia.org/wiki/%E5%96%83%E5%AD%97" \o "喃字)的各种编码与Unicode的协调性被重点关注。考虑到Unicode最终要涵盖所有的字符。从某种意义而言，这些编码方式也可视作Unicode的出现于其之前的*既成事实*的实现方式，如同[ASCII](https://zh.wikipedia.org/wiki/ASCII" \o "ASCII)及其扩展[Latin-1](https://zh.wikipedia.org/wiki/Latin-1" \o "Latin-1)一样，后两者的字符在16位Unicode编码空间中的编码第一字节各位全为0，第二字节编码与原编码完全一致。但上述东亚语言编码与Unicode编码的对应关系要复杂得多。

5.兼容：[XML](https://zh.wikipedia.org/wiki/XML" \o "XML)及其子集[XHTML](https://zh.wikipedia.org/wiki/XHTML" \o "XHTML)采用[UTF-8](https://zh.wikipedia.org/wiki/UTF-8" \o "UTF-8)作为[标准字集](https://zh.wikipedia.org/w/index.php?title=%E6%A0%87%E5%87%86%E5%AD%97%E9%9B%86&action=edit&redlink=1" \o "标准字集（页面不存在）)，理论上我们可以在各种支持XML标准的浏览器上显示任何地区文字的网页，只要电脑本身安装有合适的字体即可。可以利用&#nnn;的格式显示特定的字符。nnn代表该字符的[十进制](https://zh.wikipedia.org/wiki/%E5%8D%81%E8%BF%9B%E5%88%B6" \o "十进制)Unicode代码。如果采用十六进制代码，在编码之前加上x字符即可。但部分旧版本的浏览器可能无法识别十六进制代码。

过去电脑编码的8位标准，使每个国家都只按国家使用的字符而编定各自的编码系统；而对于部分字符系统比较复杂的语言，如[越南语](https://zh.wikipedia.org/wiki/%E8%B6%8A%E5%8D%97%E8%AF%AD" \o "越南语)，又或者东亚国家的大型字符集，都不能在8位的环境下正常显示。

只是最近才有在文本中对十六进制的支持，那么旧版本的浏览器显示那些字符或许可能有问题-大概首先会遇到的一个问题只是在对于大于8位Unicode字符的显示。解决这个问题的普遍做法仍然是将其中的十六进制码转换成一个十进制码（例如：♠用&#9824;代替&#x2660;）。

也有一些字符集标准将一些常用的标志存放在字符内码外面，那么你可能使用象—这样的文本标志来表示一个长划（—）的情况，即使它的字符内码已经被使用，这些标准也不包含那个字符。

然而部分由于Unicode版本发展原因，很多浏览器只能显示UCS-2完整字符集，也即现在使用的Unicode版本中的一个小子集。下表可以检验您的浏览器如何显示各种Unicode代码：

|  |  |  |
| --- | --- | --- |
| **代码** | **字符标准名称（英语）** | **在浏览器上的显示** |
| &#x0041; | 大写[拉丁字母](https://zh.wikipedia.org/wiki/%E6%8B%89%E4%B8%81%E5%AD%97%E6%AF%8D" \o "拉丁字母)“A” | A |
| &#x00DF; | 小写拉丁字母“Sharp S” | ß |
| &#x00FE; | 小写拉丁字母“Thorn” | þ |
| &#x0394; | 大写[希腊字母](https://zh.wikipedia.org/wiki/%E5%B8%8C%E8%85%8A%E5%AD%97%E6%AF%8D" \o "希腊字母)“Delta” | Δ |
| &#x0419; | 大写[斯拉夫字母](https://zh.wikipedia.org/wiki/%E6%96%AF%E6%8B%89%E5%A4%AB%E5%AD%97%E6%AF%8D" \o "斯拉夫字母)“Short I” | Й |
| &#x05E7; | [希伯来字母](https://zh.wikipedia.org/wiki/%E5%B8%8C%E4%BC%AF%E6%9D%A5%E5%AD%97%E6%AF%8D" \o "希伯来字母)“Qof” | ק |
| &#x0645; | [阿拉伯字母](https://zh.wikipedia.org/wiki/%E9%98%BF%E6%8B%89%E4%BC%AF%E5%AD%97%E6%AF%8D" \o "阿拉伯字母)“Meem” | م |
| &#x0E57; | [泰文](https://zh.wikipedia.org/wiki/%E6%B3%B0%E8%AF%AD" \o "泰语)[数字](https://zh.wikipedia.org/wiki/%E6%95%B0%E5%AD%97" \o "数字)7 | ๗ |
| &#x1250; | [埃塞俄比亚音节文字](https://zh.wikipedia.org/wiki/%E9%98%BF%E5%A7%86%E5%93%88%E6%8B%89%E8%AF%AD" \o "阿姆哈拉语)“Qha” | ቐ |
| &#x3042; | 日语[平假名](https://zh.wikipedia.org/wiki/%E5%B9%B3%E5%81%87%E5%90%8D" \o "平假名)“A” | あ |
| &#x30A2; | 日语[片假名](https://zh.wikipedia.org/wiki/%E7%89%87%E5%81%87%E5%90%8D" \o "片假名)“A” | ア |
| &#x53F6; | [简体](https://zh.wikipedia.org/wiki/%E7%AE%80%E4%BD%93%E4%B8%AD%E6%96%87" \o "简体中文)[汉字](https://zh.wikipedia.org/wiki/%E6%B1%89%E5%AD%97" \o "汉字)“叶” | 叶 |
| &#x8449; | [繁体](https://zh.wikipedia.org/wiki/%E7%B9%81%E4%BD%93%E4%B8%AD%E6%96%87" \o "繁体中文)汉字“葉” | 葉 |
| &#xC5FD; | [韩国音节文字](https://zh.wikipedia.org/wiki/%E8%B0%9A%E6%96%87" \o "谚文)“Yeop” | 엽 |

一些多语言支持的网页浏览器，比如[微软](https://zh.wikipedia.org/wiki/%E5%BE%AE%E8%BD%AF" \o "微软)[Windows](https://zh.wikipedia.org/wiki/Windows" \o "Windows)系统的[Internet Explorer](https://zh.wikipedia.org/wiki/Internet_Explorer" \o "Internet Explorer) 5.5及以上版本，以及跨平台的浏览器[Mozilla](https://zh.wikipedia.org/wiki/Mozilla" \o "Mozilla)／[Netscape](https://zh.wikipedia.org/wiki/Netscape" \o "Netscape) 6，可以在安装时根据需要动态地使用相应的字符集，预先安装了合适的语言包，就可以同时显示页面上的各种Unicode字符。[Internet Explorer](https://zh.wikipedia.org/wiki/Internet_Explorer" \o "Internet Explorer) 5.5还提出用户可以在需要新字体时，即装即用。另外的浏览器如[Netscape Navigator](https://zh.wikipedia.org/wiki/Netscape_Navigator" \o "Netscape Navigator) 4.77，则只能显示跟页面编码相应字符集中的文字。当你使用后一种浏览器时，你不大可能预先安装所有的字体，即使有了字体，浏览器也不一定能将这些字体完全应用起来。可能遇到的情况是，这种浏览器只能够显示部分文字，因为它们是按照标准进行编码，尽管理论上在兼容的系统中，只要有了相应的[Code2000](https://zh.wikipedia.org/wiki/Code2000" \o "Code2000)字体，就可以正确显示。一种变通的办法，是将某些少见的字符，通过“名称实体引用”的方式来使用。

# 相互联系：

GB2312是在ASCLL上发展过来的，是为了储存汉字和一些常见的字符。

Big5是为了制定一个统一的储存繁体字标准而制定的。

GB2312和BIG5都是为了储存对应的字型而建立的，两者相辅相成。

UTF-8/16是为了建立统一的互联网编码标准而建立的。

Unicode是为了统一各国国家文化而建立的一个字符集。

# 参考文献：

https://zh.wikipedia.org/wiki/GB\_2312

https://zh.wikipedia.org/wiki/%E5%A4%A7%E4%BA%94%E7%A2%BC

https://web.archive.org/web/20041212084615/http://www.haiyan.com/steelk/navigator/ref/big5/b5index.htm

https://zh.wikipedia.org/wiki/UTF-8

https://zh.wikipedia.org/wiki/Unicode