GB2312与Unicode、big5、UTF8/16的比较

1. 来历

1、GB2312是一个关键的正式注册的互联网名称字符集的的中国人民共和国，用于简化中国字。GB国际标准简称国家标准，即国家标准。GB2312（1980）已被GBK和GB18030取代，其中包括附加字符，但GB2312仍然被广泛使用。

虽然GB2312占地面积当代使用的字符的99％，历史教科书和许多名字仍然超出范围。GB2312包括6763个中国字符（在两个层面：第一是通过读取布置，所述第二通过自由基然后笔画的数目）与符号和标点，日语，沿着假名，在希腊和西里尔字母，注音，和双字节集的拼音字母带声调的。所有网页的0.7％在2017年4月使用GB2312，比2010年1月的3.5％下降。

有一个类似的字符集称为GB / T 12345，与GB2312密切相关，但传统字符形式替代简化形式，还有一些额外的62个补充字符。GB编码字体通常成对，一个是GB 2312（简体）字符集，另一个用GB / T 12345（传统）字符集。

2、Unicode具有超越传统字符编码的限制的明确目标，例如ISO 8859标准定义的限制，在世界各国广泛使用，但在很大程度上彼此不兼容。许多传统字符编码有一个共同的问题，它们允许双语计算机处理（通常使用拉丁字符和本地脚本），但不是多语言计算机处理（任意脚本的计算机处理彼此混合）。

Unicode在意图中编码底层字符 - 字形和类似字形的单元，而不是这些字符的变体字形（渲染）。在汉字的情况下，这有时会导致将基础字符与其变体字形区分开来的争议（见韩统一）。

在文本处理中，Unicode承担着为每个字符提供唯一的代码点 - 数字，而不是字形。换句话说，Unicode以抽象的方式表示一个字符，并将可视化渲染（大小，形状，字体或样式）留给其他软件，如Web浏览器或文字处理器。然而，这个简单的目标变得复杂，因为Unicode设计师的让步，希望能够更快速地采用Unicode。

前256个代码点与ISO-8859-1的内容相同，从而使现有的西方文本变得微不足道。许多本质上相同的字符在不同的代码点被多次编码，以保留传统编码所使用的区别，因此允许从那些编码转换为Unicode（并返回），而不会丢失任何信息。例如，代码点的“ 全宽形式 ”部分包含与拉丁字母主要部分分离的完整拉丁字母，因为中文，日文和韩文（CJK）字体，这些拉丁字符的渲染与CJK的宽度相同表意文字，而不是宽度的一半。有关其他的例子，请参阅以Unicode字符复制。

根据与经验施乐字符代码标准（的Xccs）自1980年以来，的Unicode日期的起源到1987年，当乔•贝克尔从施乐公司和李•科林斯和马克戴维斯从苹果开始调查创建一个通用字符集的实用性。由Peter Fenwick和Dave Opstad提供的额外投入， Joe Becker在1988年8月发表了一项“国际/多语言文本字符编码系统”草案，暂定为Unicode。他解释说，“他的名字”Unicode“旨在表明一种独特的统一的通用编码。

在本文中，题为Unicode 88，Becker概述了一个16位字符模型：

Unicode旨在解决对可行，可靠的世界文本编码的需求。Unicode可以粗略地描述为“宽身体ASCII”，已经被扩展到16位以包含世界上所有生活语言的字符。在正确设计的设计中，每个字符16位是足够的。

他的原始16位设计是基于这样的假设，即只需要对现代使用中的脚本和字符进行编码.

Unicode比保留过去的古物更为优先，以确保未来的效用。Unicode首先在现代文字中发表的角色（例如1988年在世界上所有报纸和杂志的联合体中），其数量无疑远低于2 14 = 16,384。除了这些现代用途的角色，所有其他人都可能被定义为过时或罕见; 这些是比私人使用注册更好的候选人，而不是拥挤一般有用的Unicodes的公开列表。

1989年初，Unicode工作组扩大到包括康培的Ken Whistler和Mike Kernaghan，RLG的 Karen Smith-Yoshimura和Joan Aliprand ，Sun Microsystems的 Glenn Wright ，以及1990年，来自微软的 Michel Suignard和Asmus Freytag 以及Rick McGowan的NeXT加入了集团。到1990年底，大多数绘制现有字符编码标准的工作已经完成，Unicode的最终审查草案已经准备好了。

在Unicode协会在加利福尼亚州注册成立于1991年1月3日，和1991年10月，Unicode标准的第一册出版。第二卷涵盖汉字表意，1992年6月出版。

在1996年，Unicode 2.0中实现了代理角色机制，因此Unicode不再局限于16位。这将Unicode代码空间增加到超过一百万个代码点，这允许编码许多历史性脚本（例如，埃及象形文字）以及数千个未被使用或过时的字符，这些字符未被预期为需要编码。不是原来用于Unicode的字符很少使用汉字或汉字，其中许多是个人和地名的一部分，使得它们很少被使用，但比原始的Unicode体系结构更加重要。

来自1992年的Microsoft TrueType规范版本1.0 在命名表中使用了“ Apple Unicode”而不是“ Unicode”来代替Platform ID。

3、ASCII无法支持大字符集，例如用于中文，日文和韩文，导致政府和行业找到创造性的解决方案，使其语言能够在计算机上呈现。各种特设和通常专有的输入法导致了开发标准系统的努力。因此，Big5编码是由台湾信息产业研究所于1984 年定义的。“Big5”这个名称，是由台湾最大的五家IT企业之一的合作组成的宏伟宏碁Acer（宏碁）; 神达（神通）; 佳佳（ZERO ONE Technology）（零壹或01tech）; 佳佳 和第一国际电脑（FIC）（大众）。

Big5在台湾和全球范围内迅速普及，通过采用几种商业软件包，特别是E-TEN中文DOS输入系统（ETen中文系统），将中文字符集用于传统汉字。在中华民国政府宣布中文作为他们在20世纪80年代的标准，因为它是，届时，事实上电脑上使用传统的中国标准。

4、1992年7月，X / Open委员会XJJIG正在寻找更好的编码。Unix系统实验室的 Dave Prosser 提交了一个具有更快实现特性的提案，并提出了7位ASCII字符只能表示自身的改进; 所有多字节序列将仅包括设置高位的字节。文件系统安全UCS转换格式（FSS-UTF）的名称和此提案的大部分文本后来保留在最终规范中。

1992年8月，IBM X / Open代表向有关方面分发了此提案。贝尔实验室计划9 操作系统小组的肯•汤普森（Ken Thompson）进行的修改使得它比以前的提议稍微不那么有效，但最重要的是允许它进行自同步 让读者从任何地方开始，立即检测字节序列边界。它也放弃了偏见的使用，而是添加了只允许最短的编码的规则; 紧凑性的额外损失相对来说并不重要，但读者现在必须注意无效的编码，以避免可靠性，特别是安全问题。汤普森的设计于1992年9月2日在Rob Pike的新泽西餐厅放置了一个场地。在接下来的日子里，派克和汤普森实施了它，并更新了计划9，使用它，然后将其成功传回给X / Open，它将其接受为FSS-UTF的规范。

UTF-8于1993年1月25日至29日在圣地亚哥的USENIX会议上首次正式提交。

2003年11月，UTF-8被RFC 3629限制，以匹配UTF-16字符编码的约束：明确禁止与高字节和低字符序列的3％以上的高和低代用字符相对应的代码点，并结束在U + 10FFFF处删除了超过48％的四字节序列以及所有五字节和六字节序列。

Google报道说，2008年，UTF-8（标有“Unicode”）成为HTML文件中最常见的编码方式。

在20世纪80年代后期，工作开始为“通用字符集”（UCS）开发统一的编码，这将通过一个协调的系统替代以前的语言编码。目标是包括来自世界各国语言的所有必需字符，以及来自科学，数学和音乐等技术领域的符号。原来的想法是用需要每字符2个字节的2 16 = 65,536个值替换需要每个字符1个字节的典型256字符编码。两组并行工作，IEEE和Unicode联盟，后者主要代表计算机设备制造商。两组试图同步他们的角色分配，以便开发编码将是相互兼容的。早期的2字节编码通常称为“Unicode”，但现在称为“UCS-2”。UCS-2与UTF-16不同，它是一个恒定长度编码，只能编码BMP的字符，它被许多程序支持。

早在这个过程中它变得越来越清楚的是2个16字符是不够的，[ 引证需要 ]和IEEE引入较大的31比特的空间和将要求每个字符的4个字节的编码（UCS-4）。这被Unicode联盟所抵制，因为每个字符的4个字节浪费了大量的磁盘空间和内存，并且因为一些制造商已经大量投入了2字节每字符的技术。在UTF-16编码方案被开发作为一种妥协来解决这个僵局的Unicode标准的2.0版本在1996年7月[3] ，并在完全指定RFC 2781在2000年出版IETF。

在UTF-16中，使用两个 16位代码单元编码大于或等于2 16的代码点。标准组织选择可用的最大块可用未分配的16位代码点作为这些代码单元。与UTF-8不同，它们没有提供编码这些代码点的方法。

UTF-16在国际标准ISO / IEC 10646和Unicode标准的最新版本中规定。“UCS-2现在应该被认为是过时的，它不再是指10646或Unicode标准中的编码形式。” ]没有计划扩展UTF-16以支持更多的代码点，或代码替换为代理，因为分配代码点将违反Unicode稳定策略wrt general\_category和/或替代代码点。

1. 产生的原因

GB2312:

在处理GB2312的程序中，EUC-CN通常用作字符编码（即外部存储），从而保持与ASCII的兼容性。两个字节用于表示ASCII中未找到的每个字符。第一个字节的值为0xA1-0xF7（161-247），第二个字节的值为0xA1-0xFE（161-254）。由于所有这些范围都超出ASCII，如UTF-8，当使用EUC-CN时，可以检查一个字节是否是多字节构造的一部分，但是如果字节是第一个或最后一个字节，则不能。

与UTF-8相比，GB2312（无论是本机还是在EUC-CN中编码）都具有更高的存储效率：而UTF-8每个CJK表意文字使用三个字节[a]，GB2312只使用两个。然而，GB2312并没有涵盖像Unicode那样多的表意文字。和GB2312是较少的能量效率，它需要除法由94是比较昂贵的，而UTF-8使用一个便宜的变速操作。

要将代码点映射到字节，将158（0x98）添加到代码点的行号以形成高字节，并添加158列代码点以形成低字节。行号是代码点整数除以94，该列的代码点模数为 94。

例如，如果您的GB2312代码点为4566（“外”，表示外部），则高字节将为4566/94 + 158 = 206 = 0xCE，低字节将来自4566％94 + 158 = 212 = 0xD4。所以，完整的编码是0xCED4 = 52948。

Unicode:

根据与经验施乐字符代码标准（的Xccs）自1980年以来，的Unicode日期的起源到1987年，当乔•贝克尔从施乐公司和李•科林斯和马克戴维斯从苹果开始调查创建一个通用字符集的实用性。由Peter Fenwick和Dave Opstad提供的额外投入，Joe Becker在1988年8月发表了一项“国际/多语言文本字符编码系统”草案，暂定为Unicode。他解释说，“他的名字”Unicode“旨在表明一种独特的统一的通用编码。

Big5:

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的高度普及，使后来的微软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、台湾地区中文电脑公会所推动的公会码等，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。而台湾以后发展的“国家标准中文交换码”（CNS 11643）由于先天所限，必须使用3字节来表示一个汉字，与现行英语软件欠缺相容，所以普及率远远不及Big5码。

“五大码”（Big5)是在1984年由台湾13家厂商与台湾地区财团法人信息工业策进会为五大中文套装软件（宏碁、神通、佳佳、零壹、大众）所设计的中文内码，所以就称为Big5中文内码，虽然五大套装软件并没有成功，但Big5码却深远地影响中文电脑内码，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，是因为当时台湾不同厂商各自推出不同的编码，如IBM 5550、王安码等，彼此不能兼容；另一方面，台湾当时尚未推出官方的汉字编码，而中国内地所推行的GB 2312编码，亦未有收录繁体字。同时，这个计划对于以台湾为核心的亚洲汉字圈也产生了久远的影响。

UTF8/16:

UTF-8是一种字符编码，可以使用一到四个8位字节对Unicode中的所有1,112,064 []个有效代码点进行编码,[]编码由Unicode标准定义，最初由Ken Thompson和Rob Pike设计。[]该名称源自Unicode（或通用编码字符集）转换格式 - 8位。

它被设计为与ASCII的向后兼容性。较低数值的代码点往往更频繁地出现，使用较少的字节进行编码。Unicode的前128个字符与ASCII对应一对一，使用与ASCII相同的二进制值的单个八位字节进行编码，因此有效的ASCII文本也是有效的UTF-8编码Unicode。由于在将非ASCII码代码编码为UTF-8时不会出现ASCII字节，因此UTF-8在大多数以特殊方式解释某些ASCII字符的编程和文档语言中是安全的，例如文件名中的“ / ” \ “的转义序列，和printf中的 ”％“ 。

显示了Google记录的2001年至2012年网络上主要编码的使用情况，其中UTF-8在2008年超过了所有其他人，并在2012年接近了50％的网页。

请注意，ASCII只包含网页如果它们被限制为ASCII字符，则使用任何声明的标题。

自2009年以来，UTF-8一直是万维网的主要字符编码，截至2017年9月，UTF-8占所有网页的89.8％。（下一个最受欢迎的多字节编码，Shift JIS和GB 2312分别为0.9％和0.7％）。所述的因特网邮件联盟（IMC）建议所有的电子邮件程序能够显示和使用UTF-8创建邮件，和W3C推荐UTF-8作为默认编码在XML和HTML。

1. 用来解决什么问题

GB2312是一个关键的正式注册的互联网名称字符集的的中国人民共和国，用于简化中国字。GB国际标准简称国家标准，即国家标准。GB2312（1980）已被GBK和GB18030取代，其中包括附加字符，但GB2312仍然被广泛使用。

Unicode旨在解决对可行，可靠的世界文本编码的需求。Unicode可以粗略地描述为“宽身体ASCII”，已经被扩展到16位以包含世界上所有生活语言的字符。在正确设计的设计中，每个字符16位是足够的。

他的原始16位设计是基于这样的假设，即只需要对现代使用中的脚本和字符进行编码：

Unicode比保留过去的古物更为优先，以确保未来的效用。Unicode首先在现代文字中发表的角色（例如1988年在世界上所有报纸和杂志的联合体中），其数量无疑远低于2 14 = 16,384。除了这些现代用途的角色，所有其他人都可能被定义为过时或罕见; 这些是比私人使用注册更好的候选人，而不是拥挤一般有用的Unicodes的公开列表。1989年初，Unicode工作组扩大到包括康培的Ken Whistler和Mike Kernaghan，RLG的 Karen Smith-Yoshimura和Joan Aliprand ，Sun Microsystems的 Glenn Wright ，以及1990年，来自微软的 Michel Suignard和Asmus Freytag 以及Rick McGowan的NeXT加入了集团。到1990年底，大多数绘制现有字符编码标准的工作已经完成，Unicode的最终审查草案已经准备好了。

在Unicode协会在加利福尼亚州注册成立于1991年1月3日，[4]和1991年10月，Unicode标准的第一册出版。第二卷涵盖汉字表意，1992年6月出版。

在1996年，Unicode 2.0中实现了代理角色机制，因此Unicode不再局限于16位。这将Unicode代码空间增加到超过一百万个代码点，这允许编码许多历史性脚本（例如，埃及象形文字）以及数千个未被使用或过时的字符，这些字符未被预期为需要编码。不是原来用于Unicode的字符很少使用汉字或汉字，其中许多是个人和地名的一部分，使得它们很少被使用，但比原始的Unicode体系结构更加重要。

来自1992年的Microsoft TrueType规范版本1.0 在命名表中使用了“ Apple Unicode”而不是“ Unicode”来代替Platform ID。

UTF-16是用于OS API文本中的Microsoft Windows 2000 / XP / 2003 / Vista中 / 7 / 8 / CE。较旧的Windows NT系统（Windows 2000之前）只支持UCS-2。在Windows XP中，U + FFFF以上的任何代码点都包含在用于欧洲语言的Windows中提供的任何字体中。的文件和网络数据往往是UTF-16，UTF-8的混合物，和遗留字节编码。

IBM iSeries系统指定用于UCS-2字符编码的代码页 CCSID 13488，用于UTF-16编码的CCSID 1200，用于UTF-8编码的CCSID 1208。

UTF-16由高通BREW操作系统使用; 在.NET环境; 和Qt跨平台图形小工具包。

1. 有多少个字节

UTF-8, 都是由 1~4 字节组成的， 至于是多少字节， 是根据第一个字节的内容判断的。

UNICODE 是一个超集， 内包含 UTF-8， UTF-16， UTF-32， UTF-8 最少 1 字节， UTF-16最少 2 字节， 如此类推。 全英文环境上，UTF-8 的优势是字节少， 相对传送效率高。

在处理GB2312的程序中，EUC-CN通常用作字符编码（即外部存储），从而保持与ASCII的兼容性。两个字节用于表示ASCII中未找到的每个字符。第一个字节的值为0xA1-0xF7（161-247），第二个字节的值为0xA1-0xFE（161-254）。由于所有这些范围都超出ASCII，如UTF-8，当使用EUC-CN时，可以检查一个字节是否是多字节构造的一部分，但是如果字节是第一个或最后一个字节，则不能。

与UTF-8相比，GB2312（无论是本机还是在EUC-CN中编码）都具有更高的存储效率：而UTF-8每个CJK表意文字使用三个字节[a]，GB2312只使用两个。然而，GB2312并没有涵盖像Unicode那样多的表意文字。和GB2312是较少的能量效率，它需要除法由94是比较昂贵的，而UTF-8使用一个便宜的变速操作。

BIG5码是双字节编码方案，其中第一个字节的值在OXAO-OXFE之间，第二个字节在OX40-OX7E和OXA1-OXFE之间。

BIG5收录13461个汉字和符号，包括：

符号408个，编码位置A140-A3BE

常用字5401个，编码位置A440-C67E，包括台湾教育部颁布的《常用国字标准字体表》的全部汉字4808个，台湾教科书常用字587个，异体字6个。

次常用字7652个，编码位置C940-F9D5,包括台湾教育部颁布的《次常用国字标准字体表》的全部汉字6341个，《罕用国字标准字体表》中使用频率较高的字1311个。

1. 兼容语言

在处理GB2312的程序中，EUC-CN通常用作字符编码（即外部存储），从而保持与ASCII的兼容性。两个字节用于表示ASCII中未找到的每个字符。第一个字节的值为0xA1-0xF7（161-247），第二个字节的值为0xA1-0xFE（161-254）。由于所有这些范围都超出ASCII，如UTF-8，当使用EUC-CN时，可以检查一个字节是否是多字节构造的一部分，但是如果字节是第一个或最后一个字节，则不能。

与UTF-8相比，GB2312（无论是本机还是在EUC-CN中编码）都具有更高的存储效率：而UTF-8每个CJK表意文字使用三个字节[a]，GB2312只使用两个。然而，GB2312并没有涵盖像Unicode那样多的表意文字。和GB2312是较少的能量效率，它需要除法由94是比较昂贵的，而UTF-8使用一个便宜的变速操作。

要将代码点映射到字节，将158（0x98）添加到代码点的行号以形成高字节，并添加158列代码点以形成低字节。行号是代码点整数除以94，该列的代码点模数为 94。

UNICODE字符集

作用：为世界650种语言进行统一编码，兼容ISO-8859-1。

位数：UNICODE字符集有多个编码方式，分别是UTF-8，UTF-16和UTF-32

BIG5字符集

作用：统一繁体字编码。

位数：使用2个字节表示，表示13053个汉字。

范围：高字节从A1到F9，低字节从40到7E，A1到FE

GB2312字符集

作用：国家简体中文字符集，兼容ASCII。

位数：使用2个字节表示，能表示7445个符号，包括6763个汉字，几乎覆盖所有高频率汉字。

范围：高字节从A1到F7, 低字节从A1到FE。将高字节和低字节分别加上0XA0即可得到编码。

1. 它们之间能否相互转换

• UTF-8可以对任何Unicode字符进行编码。可以正确显示不同脚本中的文件，而无需选择正确的代码页或字体。例如中文和阿拉伯语可以被支持（在同一个文本中），没有插入特殊的代码或手动设置来切换编码。

• UTF-8是自同步的：通过在任一方向扫描明确的位模式，可以轻松识别字符边界。如果字节由于错误或损坏而丢失，则可以始终找到下一个有效字符并恢复处理。如果需要缩短字符串以适应指定的字段，则可以轻松找到以前的有效字符。许多多字节编码更难以重新同步。

• 任何面向字节的 字符串搜索算法都可以与UTF-8数据一起使用，因为一个字符的字节序列不能发生在其他地方。一些较旧的可变长度编码（如Shift JIS）没有这个属性，因此使得字符串匹配算法相当复杂。在Shift JIS中，字符的结尾字节和下一个字符的第一个字节可能看起来像另一个合法字符，这在UTF-8中是不可能发生的。

• 使用简单的位操作高效编码。UTF-8不需要较慢的数学运算，如乘法或除法（与过时的UTF-1编码不同）。

• UTF-8将比为特定脚本设计的多字节编码占用更多的空间。东亚传统编码通常使用每个字符两个字节，而在UTF-8中每个字符需要三个字节。

• 字节编码和UTF-8由程序中的字节数组表示，当从字节编码转换为UTF-8时，通常无需对函数进行任何操作。UTF-16由16位字阵列表示，并转换为UTF-16，同时保持与现有基于ASCII的程序（例如Windows完成）的兼容性，需要将每个 API和数据结构复制一个字符串，一个版本接受字节字符串和另一个接受UTF-16的版本。

• 如果在U + 0080以下的代码点比U + 0800..U + FFFF范围内的代码点多，UTF-8编码的文本将小于UTF-16编码的文本。所有现代欧洲语言都是如此。

• 大多数富文本格式（包括HTML）包含大量ASCII字符，以便进行格式化，因此与UTF-16相比，大小通常会显着降低，即使语言主要使用UTF中的3字节长字符-8。

• 大多数通信（例如HTML和IP）和存储（例如对于Unix）是针对字节流而设计的。UTF-16字符串必须为每个代码单元使用一对字节：

• 这两个字节的顺序成为一个问题，必须在UTF-16协议中指定，例如使用字节顺序标记。

• 如果从UTF-16中缺少奇数个字节，则字符串的其余部分将是无意义的文本。UTF-8中缺少的任何字节仍然允许从缺少字节后的下一个字符开始准确地恢复文本。

• 字符U + 0800到U + FFFF在UTF-8中使用三个字节，但在UTF-16中只有两个字节。结果，如果有更多的这些字符比ASCII字符更多，那么（例如）中文，日语或梵文的文本将占用更多UTF-8空间。数据主要由纯散文组成，但上下文使用ASCII空格，数字和标点符号的程度则减少了这一点。

UTF-16（16 位 Unicode转换格式）是一种字符编码，能够对Unicode的所有1,112,064个有效代码点进行编码。编码是可变长度的，因为代码点用一个或两个16位代码单元进行编码。（也可参见比较UTF-8，-16和-32 的Unicode编码比较）

UTF-16 一旦发现16位不足以满足Unicode的用户社区，就从早期的固定宽度的16位编码（称为UCS-2（针对2字节通用字符集）开发）。

内容 。

参考文献

1. Hannas，William C.（1997）。亚洲的正义困境。夏威夷大学出版社。页。264。

2. “使用字符编码的网站的历史趋势”。W3Techs。

3. “GB / T 12345”

4. 17×2 16 = 1,114,112个代码点减去2,048个技术无效代理代码点

5. Unicode标准中的一组八位被称为八位字节。

6. 电子邮件主题：UTF-8历史，来自：“Rob”指挥官派克“，日期：2003年4月30日，星期三...，UTF-8设计在我眼前，在一个placemat在一个新泽西州的餐厅一个晚上在九月或一九九二年......所以那天晚上，肯写了包装和拆包代码，我开始撕裂到C和图形库。第二天所有的代码都完成了...

7. 派克，罗布; 汤普森，肯（1993）。“Hello World orΚαλημέρακόσμε或こんにちは世界”。1993年冬季会议论文集（PDF）。

8. “第二章一般结构”。Unicode标准（6.0版）。Mountain View，加州，美国：Unicode联盟。ISBN 978-1-936213-01-6。

9. 戴维斯，马克（2010-01-28）。“Unicode接近网路的50％”。谷歌官方博客。Google 。检索2010-12-05。

10. “使用字符编码的历史趋势” 。检索2017-09-09。

11. “UTF-8使用统计”。BuiltWith 。检索2011-03-28。

12. ^ “在互联网邮件中使用国际字符”。互联网邮件联盟。1998-08-01 。检索2007-11-08。

13. “指定文件的字符编码”，HTML5，万维网联盟，2014-06-17 ，检索2014-07-30