基础入门—JAVA字符集详解（一） PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/145/2021_2022__E5_9F_BA_E7_A1_80_E5_85_A5_E9_c104_145064.htm 1. 概述 本文主要包括以下几个方面：编码基本知识，java，系统软件，url，工具软件等。 在下面的描述中，将以"中文"两个字为例，经查表可以知道其GB2312编码是"d6d0 cec4"，Unicode编码为"4e2d 6587"，UTF编码就是"e4b8ad e69687"。注意，这两个字没有iso8859-1编码，但可以用iso8859-1编码来"表示"。 2. 编码基本知识 最早的编码是iso8859-1，和ascii编码相似。但为了方便表示各种各样的语言，逐渐出现了很多标准编码，重要的有如下几个。 2.1. iso8859-1 属于单字节编码，最多能表示的字符范围是0-255，应用于英文系列。比如，字母a的编码为0x61=97。 很明显，iso8859-1编码表示的字符范围很窄，无法表示中文字符。但是，由于是单字节编码，和计算机最基础的表示单位一致，所以很多时候，仍旧使用iso8859-1编码来表示。而且在很多协议上，默认使用该编码。比如，虽然"中文"两个字不存在iso8859-1编码，以gb2312编码为例，应该是"d6d0 cec4"两个字符，使用iso8859-1编码的时候则将它拆开为4个字节来表示："d6 d0 ce c4"（事实上，在进行存储的时候，也是以字节为单位处理的）。而如果是UTF编码，则是6个字节"e4 b8 ad e6 96 87"。很明显，这种表示方法还需要以另一种编码为基础。 2.2. GB2312/GBK 这就是汉子的国标码，专门用来表示汉字，是双字节编码，而英文字母和iso8859-1一致（兼容iso8859-1编码）。其中gbk编码能够用来同时表示繁体字和简体字，而gb2312只能表示简体字，gbk是兼容gb2312编

码的。 2.3. unicode 这是最统一的编码，可以用来表示所有语言的字符，而且是定长双字节（也有四字节的）编码，包括英文字母在内。所以可以说它是不兼容iso8859-1编码的，也不兼容任何编码。不过，相对于iso8859-1编码来说，uniocode编码只是在前面增加了一个0字节，比如字母a为"00 61"。 需要说明的是，定长编码便于计算机处理（注意GB2312/GBK不是定长编码），而unicode又可以用来表示所有字符，所以在很多软件内部是使用unicode编码来处理的，比如java。 2.4. UTF 考虑到unicode编码不兼容iso8859-1编码，而且容易占用更多的空间：因为对于英文字母，unicode也需要两个字节来表示。所以unicode不便于传输和存储。因此而产生了utf编码，utf编码兼容iso8859-1编码，同时也可以用来表示所有语言的字符，不过，utf编码是不定长编码，每一个字符的长度从1-6个字节不等。另外，utf编码自带简单的校验功能。一般来讲，英文字母都是用一个字节表示，而汉字使用三个字节。 注意，虽然说utf是为了使用更少的空间而使用的，但那只是相对于unicode编码来说，如果已经知道是汉字，则使用GB2312/GBK无疑是最节省的。不过另一方面，值得说明的是，虽然utf编码对汉字使用3个字节，但即使对于汉字网页，utf编码也会比unicode编码节省，因为网页中包含了很多的英文字符。 3. java对字符的处理 在java应用软件中，会有多处涉及到字符集编码，有些地方需要进行正确的设置，有些地方需要进行一定程度的处理。 3.1. getBytes(charset) 这是java字符串处理的一个标准函数，其作用是将字符串所表示的字符按照charset编码，并以字节方式表示。注意字符串在java内存中总是按unicode编码存储的。比如"中文"，正常情况下（即

没有错误的时候）存储为"4e2d 6587"，如果charset为"gbk"，则被编码为"d6d0 cec4"，然后返回字节"d6 d0 ce c4"。如果charset为"utf8"则最后是"e4 b8 ad e6 96 87"。如果是"iso8859-1"，则由于无法编码，最后返回 "3f 3f"（两个问号）。 3.2. new String(charset) 这是java字符串处理的另一个标准函数，和上一个函数的作用相反，将字节数组按照charset编码进行组合识别，最后转换为unicode存储。参考上述getBytes的例子，"gbk"和"utf8"都可以得出正确的结果"4e2d 6587"，但iso8859-1最后变成了"003f 003f"（两个问号）。 因为utf8可以用来表示/编码所有字符，所以new String(str.getBytes("utf8"), "utf8") === str，即完全可逆。 3.3. setCharacterEncoding() 该函数用来设置http请求或者相应的编码。 对于request，是指提交内容的编码，指定后可以通过getParameter()则直接获得正确的字符串，如果不指定，则默认使用iso8859-1编码，需要进一步处理。参见下述"表单输入"。值得注意的是在执行setCharacterEncoding()之前，不能执行任何getParameter()。java doc上说明：This method must be called prior to reading request parameters or reading input using getReader()。而且，该指定只对POST方法有效，对GET方法无效。分析原因，应该是在执行第一个getParameter()的时候，java将会按照编码分析所有的提交内容，而后续的getParameter()不再进行分析，所以setCharacterEncoding()无效。而对于GET方法提交表单是，提交的内容在URL中，一开始就已经按照编码分析所有的提交内容，setCharacterEncoding()自然就无效。 对于response，则是指定输出内容的编码，同时，该设置会传递给浏览器，告诉浏览器输出内容所采用的编码。 3.4. 处理过程 下面分析

两个有代表性的例子，说明java对编码有关问题的处理方法。
3.4.1. 表单输入 User input *(gbk:d6d0 cec4) browser *(gbk:d6d0 cec4) web server iso8859-1(00d6 00d 000ce 00c4) class，需要在class中进行处理：getbytes("iso8859-1")为d6 d0 ce c4，new String("gbk")为d6d0 cec4，内存中以unicode编码则为4e2d 6587。l 用户输入的编码方式和页面指定的编码有关，也和用户的操作系统有关，所以是不确定的，上例以gbk为例。
从browser到web server，可以在表单中指定提交内容时使用的字符集，否则会使用页面指定的编码。而如果在url中直接用?的方式输入参数，则其编码往往是操作系统本身的编码，因为这时和页面无关。上述仍旧以gbk编码为例。 Web server接收到的是字节流，默认时（getParameter）会以iso8859-1编码处理之，结果是不正确的，所以需要进行处理。但如果预先设置了编码（通过request. setCharacterEncoding ()），则能够直接获取到正确的结果。 在页面中指定编码是个好习惯，否则可能失去控制，无法指定正确的编码。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com