JAVA中的字符与代码点（1） PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/354/2021_2022_JAVA_E4_B8_AD_E7_9A_84_c104_354068.htm 摘要 本文介绍 Java 平台支持增补字符的方式。增补字符是 Unicode 标准中代码点超出 U FFFF 的字符，因此它们无法在 Java 编程语言中描述为单个的 16 位实体（例如char数据类型）。这些字符一般极少用，但是，有些会在诸如中文或日文人名中用到，因此，在东亚国家，政府应用程序通常会要求支持这些字符。 Java 平台目前正在改进，以便支持对增补字符的处理，这种改进对现有的应用程序影响微乎其微。新的低层 API 在需要时能够使用单个的字符运行。不过，大多数文本处理 API 均使用字符序列，例如String类或字符数组。现在，这些均解释为 UTF-16 序列，而且，这些 API 实现已转变为正确地处理增补字符。这些改进已融入 Java 2 平台 5.0 版，标准版 （J2SE）。 除详细解释这些改进之外，本文同时为应用程序开发人员确定和实现必要的更改提供指导，以支持整个 Unicode 字符集的使用。背景 Unicode 最初设计是作为一种固定宽度的 16 位字符编码。在 Java 编程语言中，基本数据类型char初衷是通过提供一种简单的、能够包含任何字符的数据类型来充分利用这种设计的优点。不过，现在看来，16 位编码的所有 65，536 个字符并不能完全表示全世界所有正在使用或曾经使用的字符。于是，Unicode 标准已扩展到包含多达 1，112，064 个字符。那些超出原来的 16 位限制的字符被称作增补字符。Unicode 标准 2.0 版是第一个包含启用增补字符设计的版本，但是，直到 3.1 版才收入第一批增补字符集。由于 J2SE 的 5.0 版必须支

持 Unicode 标准 4.0 版，因此它必须支持增补字符。 对增补字符的支持也可能会成为东亚市场的一个普遍商业要求。政府应用程序会需要这些增补字符，以正确表示一些包含罕见中文字符的姓名。出版应用程序可能会需要这些增补字符，以表示所有的古代字符和变体字符。中国政府要求支持 GB18030（一种对整个 Unicode 字符集进行编码的字符编码标准），因此，如果是 Unicode 3.1 版或更新版本，则将包括增补字符。台湾标准 CNS-11643 包含的许多字符在 Unicode 3.1 中列为增补字符。香港政府定义了一种针对粤语的字符集，其中的一些字符是 Unicode 中的增补字符。最后，日本的一些供应商正计划利用增补字符空间中大量的专用空间收入 50，000 多个日文汉字字符变体，以便从其专有系统迁移至基于 Java 平台的解决方案。 因此，Java 平台不仅需要支持增补字符，而且必须使应用程序能够方便地做到这一点。由于增补字符打破了 Java 编程语言的基础设计构想，而且可能要求对编程模型进行根本性的修改，因此，Java Community Process 召集了一个专家组，以期找到一个适当的解决方案。该小组被称为 JSR-204 专家组，使用Unicode 增补字符支持的 Java 技术规范请求的编号。从技术上来说，该专家组的决定仅适用于 J2SE 平台，但是由于 Java 2 平台企业版 （J2EE） 处于 J2SE 平台的最上层，因此它可以直接受益，我们期望 Java 2 平台袖珍版 （J2ME） 的配置也采用相同的设计方法。 不过，在了解 JSR-204 专家组确定的解决方案之前，我们需要先理解一些术语。 代码点、字符编码方案、UTF-16：这些是指什么？不幸的是，引入增补字符使字符模型变得更加复杂了。在过去，我们可以简单地说“ 字符 ”，在一个基于 Unicode 的环

境（例如 Java 平台）中，假定字符有 16 位，而现在我们需要更多的术语。我们会尽量介绍得相对简单一些 如需了解所有详细的讨论信息，您可以阅读Unicode 标准第 2 章或 Unicode 技术报告 17“ 字符编码模型 ”。Unicode 专业人士可略过所有介绍直接参阅本部分中的最后定义。 字符是抽象的最小文本单位。它没有固定的形状（可能是一个字形），而且没有值。“ A ”是一个字符，“ ”（德国、法国和许多其他欧洲国家通用货币的标志）也是一个字符。 字符集是字符的集合。例如，汉字字符是中国人最先发明的字符，在中文、日文、韩文和越南文的书写中使用。 编码字符集是一个字符集，它为每一个字符分配一个唯一数字。Unicode 标准的核心是一个编码字符集，字母“ A ”的编码为 004116 和字符“ ”的编码为20AC16.Unicode 标准始终使用十六进制数字，而且在书写时在前面加上前缀“ U ”，所以“ A ”的编码书写为“ U 0041 ”。 代码点是指可用于编码字符集的数字。编码字符集定义一个有效的代码点范围，但是并不一定将字符分配给所有这些代码点。有效的 Unicode 代码点范围是 U 0000 至 U 10FFFF.Unicode 4.0 将字符分配给一百多万个代码点中的 96，382 代码点。 增补字符是代码点在 U 10000 至 U 10FFFF 范围之间的字符，也就是那些使用原始的 Unicode 的 16 位设计无法表示的字符。从 U 0000 至 U FFFF 之间的字符集有时候被称为基本多语言面 （BMP）。因此，每一个 Unicode 字符要么属于 BMP，要么属于增补字符。 字符编码方案是从一个或多个编码字符集到一个或多个固定宽度代码单元序列的映射。最常用的代码单元是字节，但是 16 位或 32 位整数也可用于内部处理。UTF-32、UTF-16 和 UTF-8 是 Unicode 标准的编

码字符集的字符编码方案。 UTF-32 即将每一个 Unicode 代码点表示为相同值的 32 位整数。很明显，它是内部处理最方便的表达方式，但是，如果作为一般字符串表达方式，则要消耗更多的内存。 UTF-16 使用一个或两个未分配的 16 位代码单元的序列对 Unicode 代码点进行编码。值 U 0000 至 U FFFF 编码为一个相同值的 16 位单元。增补字符编码为两个代码单元，第一个单元来自于高代理范围（U D800 至 U DBFF），第二个单元来自于低代理范围（U DC00 至 U DFFF）。这在概念上可能看起来类似于多字节编码，但是其中有一个重要区别：值 U D800 至 U DFFF 保留用于 UTF-16；没有这些值分配字符作为代码点。这意味着，对于一个字符串中的每个单独的代码单元，软件可以识别是否该代码单元表示某个单单元字符，或者是否该代码单元是某个双单元字符的第一个或第二单元。这相当于某些传统的多字节字符编码来说是一个显著的改进，在传统的多字节字符编码中，字节值 0x41 既可能表示字母“ A ”，也可能是一个双字节字符的第二个字节。 UTF-8 使用一至四个字节的序列对编码 Unicode 代码点进行编码。U 0000 至 U 007F 使用一个字节编码，U 0080 至 U 07FF 使用两个字节，U 0800 至 U FFFF 使用三个字节，而 U 10000 至 U 10FFFF 使用四个字节。UTF-8 设计原理为：字节值 0x00 至 0x7F 始终表示代码点 U 0000 至 U 007F（Basic Latin 字符子集，它对应 ASCII 字符集）。这些字节值永远不会表示其他代码点，这一特性使 UTF-8 可以很方便地在软件中将特殊的含义赋予某些 ASCII 字符。 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