谈谈Windows程序中的字符编码 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/220/2021_2022__E8_B0_88_E8_B0_88Wind_c67_220589.htm 写这篇文章的起因是这么一个问题：我们在使用和安装Windows程序时，有时会看到以“ 2052 ”、“ 1033 ”这些数字为名的文件夹，这些数字似乎和字符集有关，但它们究竟是什么意思呢？ 研究这个问题的同时，又会遇到其它问题。我们会谈到Windows的内部架构、Win32 API的A/W函数、Locale、ANSI代码页、与字符编码有关的编译参数、MBCS和Unicode程序、资源和乱码等，一起经历这段琐碎细节为主，间或乐趣点缀的旅程。 0 Where is Win32 API Windows程序有用户态和核心态的说法。在32位地址空间中，0x80000000以下属于用户态，0x80000000以上属于核心态。所有硬件管理都在核心态。用户态程序的不能直接使用核心态的任何代码。所谓核心态其实只是CPU的一种保护模式。在x86 CPU上，用户态处于ring 3，核心态处于ring 0。 从用户态进入核心态的最常用的方法是在寄存器eax填一个功能码，然后执行int 2e。这有点像DOS时代的DOS和BIOS系统调用。在NT架构中这种机制被称作system service。 在核心态提供system service的有两个家伙：ntoskrnl.exe和win32k.sys。ntoskrnl.exe是Windows的大脑，它的上层被称为Executive，下层被称作Kernel。Win32k.sys提供与显示有关的system service。 在用户态一侧，有一个重要的角色叫作ntdll.dll，大多数system service都是它调用的。它封装这些system service，然后提供一个API接口。这个接口被称作native API。 native API的用户是各个子系统（subsystem），包括Win32子系统

、OS/2子系统、POSIX子系统。各个子系统为Win32、OS2、POSIX程序提供了运行平台。 ntdll.dll由于提供了平台无关的API接口，所以被看作是NT系统的原生接口，由之得到了“ native API ”的匪号。其实它的主要工作是将调用传递到核心态。 Win32、OS/2、POSIX，听起来很庞大。其实真正做好的只有Win32子系统。OS2、POSIX都是Console UI，即只有字符界面。提供OS/2子系统，只因为在1988年，NT的主要设计目标就是与OS/2兼容，后来由于Windows 3.0卖得很好，所以设计目标被变更为与Windows兼容。提供POSIX子系统，是为了应付美国政府的一个编号为FIPS 151-2的标准。 Win32子系统的管理员是一个叫作csrss.exe的弟兄，它的全名是：Client/Server Run-Time Subsystem。它刚上任时，本来要分管所有的子系统，但后来POSIX和OS/2都被分别处理了，所以只管了一个Win32。即使这样也很了不起，所有的Win32程序的进程、线程们都要向它登记。 不过Win32程序用得最多的还是Win32子系统的DLL们，最核心的DLL包括：kernel32.dll、User32.dll、Gdi32.dll、Advapi32.dll。这些DLL包装了ntdll.dll的native API。其中Gdi32.dll比较特殊，它与核心态的win32k.sys直接保持联系，以提高NT系统的图形处理能力。Win32子系统的DLL们提供的接口函数在MSDN文档中被详细介绍，它们就是Win32 API。 