

基础3-mknod命令

mknod命令用于创建Linux中的字符设备文件和块设备文件。

语法

mknod(选项)(参数)

选项

-Z: 设置安全的上下文;
-m: 设置权限模式;
[-help](#): 显示帮助信息;
--version: 显示版本信息。

参数

- 文件名: 要创建的设备文件名;
- 类型: 指定要创建的设备文件的类型;
- 主设备号: 指定设备文件的主设备号;
- 次设备号: 指定设备文件的次设备号。

实例

```
ls -la /dev/ttyUSB*  
crw-rw-- 1 root dialout 188, 0 2008-02-13 18:32 /dev/ttyUSB0  
mknod /dev/ttyUSB32 c 188 32
```

扩展知识

Linux的设备管理是和文件系统紧密结合的, 各种设备都以文件的形式存放在/dev目录下, 称为设备文件。应用程序可以打开、关闭和读写这些设备文件, 完成对设备的操作, 就像操作普通的数据文件一样。

为了管理这些设备, 系统为设备编了号, 每个设备号又分为主设备号和次设备号。主设备号用来区分不同种类的设备, 而次设备号用来区分同一类型的多个设备。对于常用设备, Linux有约定俗成的编号, 如硬盘的主设备号是3。

Linux为所有的设备文件都提供了统一的操作函数接口, 方法是使用数据结构struct file_operations。这个数据结构中包括许多操作函数的指针, 如open()、close()、[read\(\)](#)和[write\(\)](#)等, 但由于外设的种类较多, 操作方式各不相同。Struct file_operations结构体中的成员为一系列的接口函数, 如用于读/写的read/write函数和用于控制的ioctl等。

打开一个文件就是调用这个文件file_operations中的open操作。不同类型的文件有不同的file_operations成员函数, 如普通的磁盘数据文件, 接口函数完成磁盘数据块读写操作; 而对于各种设备文件, 则最终调用各自驱动程序中的I/O函数进行具体设备的操作。这样, 应用程序根本不必考虑操作的是设备还是普通文件, 可一律当作文件处理, 具有非常清晰统一的I/O接口。所以file_operations是文件层次的I/O接口。