# 文件系统初始化

## vfs\_caches\_init

1. 创建names\_cachep，用于保存文件名
2. dcache\_init()
3. 调用KMEM\_CACHE分配一段内存;
4. alloc\_large\_system\_hash创建地址连续hlist\_head，首地址为dentry\_hashtable;
5. INIT\_HLIST\_BL\_HEAD初始化hlist\_node为null
6. inode\_init()
7. 调用KMEM\_CACHE分配一段内存;
8. alloc\_large\_system\_hash创建地址连续hlist\_head，首地址为inode\_hashtable;
9. INIT\_HLIST\_BL\_HEAD初始化hlist\_node为null;
10. files\_init(mempages)
11. 创建file\_cachep，用于保存文件对象
12. 初始化文件相关全局变量，系统最大文件max\_files系统可以打开的最大文件数sysctl\_nr\_open\_max，每个CPU的最大文件数
13. mnt\_init()
14. 初始化命名空间的读写信号量
15. 创建mnt\_cache，用于保存文件系统的挂载数据对象mount
16. 创建mount\_hashtable、mountpoint\_hashtable;
17. 初始化vfsmount\_lock全局变量，它是一个lglock对象
18. sysfs\_init()注册并挂载sysfs文件系统

* 创建sysfs\_dir\_cachep缓存，用来存放sysfs文件系统的目录项；
* 初始化sysfs\_backing\_dev\_info全局变量，它是一个backing\_dev\_info对象，也就是存储设备信息。
* 注册sysfs，将sysfs\_fs\_type文件类型插入filesystems指向的文件类型链表中
* 在mnt\_cache缓存中创建一个mount结构体
* 创建超级块super\_block，初始化它的各种链表
* sysfs\_root为sysfs的根目录，它是全局的。为sysfs\_root分配inode，inode号为1，首先在inode\_hashtable哈希表中通过超级块以及inode号检索是否存在（初始化时当然还没有存在），若不存在则创建一个inode，并将它加入inode\_hashtable中。
* 在VFS层中创建sysfs的根目录，并将d\_fsdata字段指向sysfs\_root
* 初始化mount结构体，其中mnt\_root字段指向VFS层中sys文件系统的根目录，mnt\_sb字段指向sysfs的超级块，mnt\_mountpoint字段指向VFS层中sys文件系统的根目录，mnt\_parent字段指向自己（因为第一个挂载的文件系统是sys）。
* 对vfsmount\_lock加写锁，将mount通过字段mnt\_instance加入超级块的s\_mounts list\_head中，解锁
* 创建kobject结构体，并设置它的ktype字段为dynamic\_kobj\_ktype
* 设置kobject的name字段为fs
* 根据fs的kobject在sysfs文件系统以及VFS中创建fs目录，其中sysfs文件系统中为sysfs\_dirent结构，VFS中为dentry结构。sysfs中父目录指向sysfs\_root。sysfs中目录结构通过红黑树架设，将新创建的sysfs\_dirent结构通过hash算法遍历红黑树并放在最下层，设置节点颜色

1. kobject\_create\_and\_add("fs", NULL)在sysfs文件系统中创建fs目录，并为fs目录创建kobject结构体
2. init\_rootfs()注册根文件系统。过程同sysfs文件系统注册
3. init\_mount\_tree()安装根文件系统，过程类似sysfs文件系统安装，不同的是会创建mnt\_namespace，并将root文件系统的mount通过mnt\_list字段加入mnt\_namespace list字段指向的list\_head中，设置init\_task进程的nsproxy字段指向的mnt\_ns命名空间为新创建的mnt\_namespace。设置当前进程的当前工作目录、根目录
4. bdev\_cache\_init()
5. chrdev\_init()

## 初始化后数据对象

