容器创建:

1. 必要参数检查
2. 查找创建容器的image
3. 创建容器的读写层:

layer/layer\_store.go:CreateRWLayer(containerID, parent chainID, opts \*CreateRWLayerOpts)

containerID:容器ID

parent:这里就是镜像的img.RootFS.ChainID

参数选项:包括mount参数，mount初始化函数，存储配置。

1. mount初始化:

* 调用driver创建读写层:

graphdriver/overlay2/overlay.go:CreateReadWrite(id,parent,opts)

id: <**cacheID-init**> 对于linux系统cacheID为一个随机ID，对于windows系统就是容器ID.

1. 获取设置的UID,GID MAP，以映射的UID,GID创建读写层目录,目录权限为0700(不同的graphdriver权限不同)/var/lib/docker/overlayer2/<cacheID-init>
2. 如果配置了限额，则设置限额
3. 创建diff, work,megerd目录和link,,lower文件，在../l中创建符号链接指向diff目录，并将符号链接名称写入link文件。将parent cacheID写入lower文件.

* 将所有layers和读写层mount到cacheID目录下的merged:

graphdriver/overlay2/overlay.go:Get(id,mountLable)

* 调用初始化函数初始化统一层

daemon/initlayer/setup\_unix.go:Setup(initLayer,rootUID,rootGid)

创建如下目录，文件和符号链接

**"/dev/pts"**: **"dir"**,  
**"/dev/shm"**: **"dir"**,  
**"/proc"**: **"dir"**,  
**"/sys"**: **"dir"**,  
**"/.dockerenv"**: **"file"**,  
**"/etc/resolv.conf"**: **"file"**,  
**"/etc/hosts"**: **"file"**,  
**"/etc/hostname"**: **"file"**,  
**"/dev/console"**: **"file"**,  
**"/etc/mtab"**: **"/proc/mounts"**

* 将所有layers再umount

1. 设置主机配置

包括volume配置和tmpfs配置

volume的优先级为:之前配置--🡪volumes-from-🡪volume

1. 设置links

在新容器的/etc/hosts中添加link的容器DNS信息

1. 保存配置到hostconfig.json

/var/lib/docker/<containers-id>/hostconfig.json