设置脚本开机启动

前置知识

0：系统停机（关机）模式，系统默认运行级别不能设置为0，否则不能正常启动，一开机就自动关机。
1：单用户模式，root权限，用于系统维护，禁止远程登陆，就像Windows下的安全模式登录。
2：多用户模式，没有NFS网络支持。
3：完整的多用户文本模式，有NFS，登陆后进入控制台命令行模式。
4：系统未使用，保留一般不用，在一些特殊情况下可以用它来做一些事情。例如在笔记本电脑的电池用尽时，可以切换到这个模式来做一些设置。
5：图形化模式，登陆后进入图形GUI模式或GNOME、KDE图形化界面，如X Window系统。
6：重启模式，默认运行级别不能设为6，否则不能正常启动，就会一直开机重启开机重启。

```
pi@raspberrypi:/etc $ ls rc*
rc.local

rc0.d:
K01alsa-utils    K01bluetooth  K01fake-hwclock  K01networking  K01paxctld    K01rpcbind  K01triggerhappy
K01avahi-daemon  K01dhcpcd     K01hwclock.sh    K01nfs-common  K01rng-tools  K01rsyslog  K01udev

rc1.d:
K01alsa-utils    K01bluetooth  K01fake-hwclock  K01paxctld    K01rpcbind  K01triggerhappy
K01avahi-daemon  K01dhcpcd     K01nfs-common    K01rng-tools  K01rsyslog

rc2.d:
S01avahi-daemon      S01cron    S01dphys-swapfile  S01rng-tools  S01ssh
S01bluetooth         S01dbus    S01paxctld         S01rsync      S01sudo
S01console-setup.sh  S01dhcpcd  S01raspi-config    S01rsyslog    S01triggerhappy

rc3.d:
S01avahi-daemon      S01cron    S01dphys-swapfile  S01rng-tools  S01ssh           S90netDetect
S01bluetooth         S01dbus    S01paxctld         S01rsync      S01sudo
S01console-setup.sh  S01dhcpcd  S01raspi-config    S01rsyslog    S01triggerhappy

rc4.d:
S01avahi-daemon      S01cron    S01dphys-swapfile  S01rng-tools  S01ssh
S01bluetooth         S01dbus    S01paxctld         S01rsync      S01sudo
S01console-setup.sh  S01dhcpcd  S01raspi-config    S01rsyslog    S01triggerhappy

rc5.d:
S01avahi-daemon      S01cron    S01dphys-swapfile  S01rng-tools  S01ssh
S01bluetooth         S01dbus    S01paxctld         S01rsync      S01sudo
S01console-setup.sh  S01dhcpcd  S01raspi-config    S01rsyslog    S01triggerhappy

rc6.d:
K01alsa-utils    K01bluetooth  K01fake-hwclock  K01networking  K01paxctld    K01rpcbind  K01triggerhappy
```

c0~6中数字代表运行级别
0代表关机模式
3代表完整的多用户文本模式（常用）
我们一般在init.d目录中放置脚本
并且在rc3.d中设置相应的软引用（start /stop）

实操

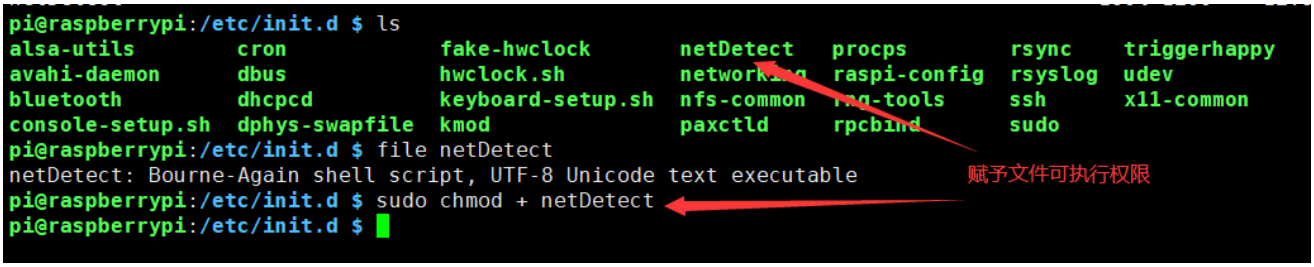
编写脚本

```
#! /bin/bash
#需要开机启动的脚本


#用于检测设备是否联网
function network()
{
    #超时时间
    local timeout=2

    #目标网站
    local target=www.baidu.com
```

```
    #获取响应状态码
    local ret_code=`curl -I -s --connect-timeout ${timeout} ${target} -w %{http_code} | tail -
n1`

    if [ "x$ret_code" = "x200" ]; then
        #网络畅通
        return 1
    else
        #网络不畅通
        return 0
    fi

    return 0
}


# 循环检测设备联网情况
# 联网则启动java -jar xxx.jar程序
# 未联网则一直检测
tryStartJar()
{
while true
do
    network
    if [ $? -eq 0 ];then
        echo "network not ready..." >> /home/pi/error.log
    else
       echo "network ready..." >> /home/pi/error.log
       #echo "break"
       break;
    fi

done
    echo "-------------success------------">/home/pi/info.log
    #后台运行 指定目录下的jar包
    java -jar /home/pi/Final.jar 1>/home/pi/info.log 2>/home/pi/erro.log &
    exec echo "success"
    exit 0
}


## 以下是开机启动的相应命令
## 启动运行    netDetect(脚本名) start
## 终止运行    netDetect(脚本名) stop
## 重     启    netDetect(脚本名) restart
## 确保上述命令运行正常
case "$1" in
        start)
                tryStartJar &
                ;;
        stop)
                echo "===="
                ps aux|grep "netDetect"
                echo "==="
                ps aux | grep "java -jar"
                echo "===="
                #ps aux | grep "netDetect" | awk '{print $2}' | xargs kill -9 2>&1
                ps aux | grep "java -jar"| awk '{print $2}' | xargs kill -9 2>&1
                ;;
        restart)
                $0 stop
```

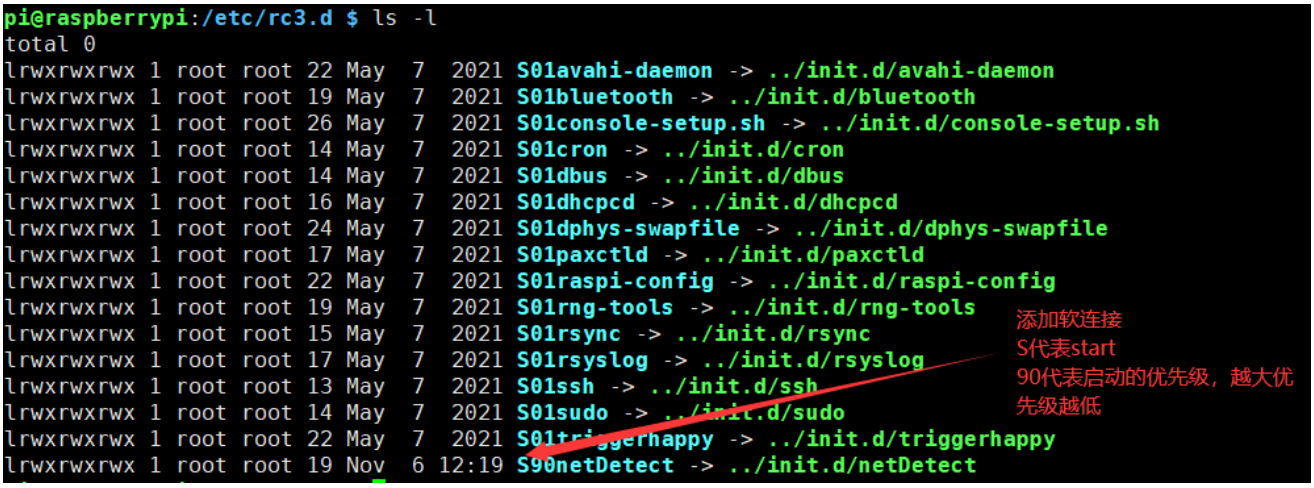
```
                $0 start
                ;;
        *)
                echo "Usage: $0 {start|stop|restart}"
                exit 1
                ;;
esac
```

脚本放置在/etc/init.d目录下



添加软链接

```
#设置开机启动 rc3开机模式
ln -s /etc/init.d/netDetect /etc/rc3.d/S90netDetect
#设置关机关闭 rc0关机模式
ln -s /etc/init.d/netDetect /etc/rc0.d/K90netDetect
##可以发现在启动脚本前面都加了 "K数字"，或者 "S数字"
##其中 K 表示 Kill 某个程序，S 表示 Start 某个程序
##后面紧跟着的数字，表示启动/停止某个程序的顺序，数字越小的越先启动
```



java获取本机IP（对外）

```
import java.net.Inet4Address;
import java.net.InetAddress;
import java.net.NetworkInterface;

public static String getIpAddress() {
        try {
            Enumeration<NetworkInterface> allNetInterfaces =
NetworkInterface.getNetworkInterfaces();
            InetAddress ip = null;
            while (allNetInterfaces.hasMoreElements()) {
                NetworkInterface netInterface = (NetworkInterface)
allNetInterfaces.nextElement();
                if (netInterface.isLoopback() || netInterface.isVirtual() ||
!netInterface.isUp()) {
```

```
                    continue;
                } else {
                    Enumeration<InetAddress> addresses = netInterface.getInetAddresses();
                    while (addresses.hasMoreElements()) {
                        ip = addresses.nextElement();
                        if (ip != null && ip instanceof Inet4Address) {
                            return ip.getHostAddress();
                        }
                    }
                }
            }
        } catch (Exception e) {
            System.err.println("IP地址获取失败" + e.toString());
        }
        return "";
    }
```