Difference between /dev/console, /dev/tty, and /dev/tty0

Last modified: September 3, 2022

by baeldung (https://www.baeldung.com/linux/author/baeldung)

Files (https://www.baeldung.com/linux/category/files)

If you have a few years of experience in the Linux ecosystem, and you're interested in sharing that experience with the community, have a look at our **Contribution Guidelines** (/linux/contribution-guidelines).

1. Introduction

In this tutorial, we'll first explore the two *tty* (https://man7.org/linux/man-pages/man4/tty.4.html) drivers present in a Linux system; */dev/tty* and */dev/console*. Then we'll discover the different roles of */dev/console*, */dev/tty0* and */dev/tty*.

2. */dev/console*

/dev/console is known as the system console. Historically this was the terminal attached directly to the Linux computer. Now it provides an option to connect a serial terminal to a Linux computer.

Let's have a look at */dev/console*:

```
$ ls -lah /dev/console
crw------- 1 root root 5, 1 Jun 15 15:25 /dev/console
$ file /dev/console
/dev/console: character special (5/1)
```

As we can see, it's a file that lives in the folder */dev*. **This represents a character device with an identifier consisting of a major number of 5 and a minor number of 1.**

Looking at the kernel (https://www.kernel.org/doc/html/v4.11/admin-guide/devices.html?highlight=terminal#virtual-consoles-and-the-console-device) documentation, it states that */dev/console* used to be a symbolic link. As a result, */dev/console* is pointing to another device.

Let's see what */dev/console* points to:

```
$ cat /sys/devices/virtual/tty/console/active
tty0
```

Hmm, does this mean that *tty0* and */dev/console* are the same thing? Let's investigate further.

If we look at the documentation for the Linux Serial Console (https://www.kernel.org/doc/html/v4.14/admin-guide/serial-console.html) it states that we can change what */dev/console* points to.

Let's try and do that:

Note: This example requires a GRUB bootloader.

```
$ cat /etc/default/grub | head -n 10
# If you change this file, run 'update-grub' afterwards to update
# /boot/grub/grub.cfg.
# For full documentation of the options in this file, see:
#   info -f grub -n 'Simple configuration'

GRUB_DEFAULT=0
GRUB_TIMEOUT=0
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
GRUB_CMDLINE_LINUX="ignore_loglevel"
```

We have to modify the line in */etc/default/grub* with GRUB_CMD_LINE_LINUX="ignore_loglevel" to:

GRUB_CMD_LINE_LINUX="console=/dev/tty2 ignore_loglevel"

Let's update the grub config and restart:

```
$ sudo update-grub
$ sudo reboot
```

Now let's have a look at where */dev/console* is pointing:

```
$ cat /sys/devices/virtual/tty/console/active
tty2
```

We can see that */dev/console* has changed.

So now that we know we can change what */dev/console* points to. What's the purpose of this device?

It seems that */dev/console* provides the kernel a guaranteed place to output its log messages, especially during boot time. Also, the device used when switching to single-user mode is */dev/console*.

Let's give this a go by entering into single-user mode:

```
$ init 1
```

We notice that the graphics have disappeared. We're in single-user text mode!

Let's see which *tty* we are on:

```
$ tty
tty2
```

Which is the same as */dev/console*.

As we've discovered. */dev/console* can be pointed to a variety of devices (https://github.com/torvalds/linux/blob/8f02f363f76f99f08117336cfac7f24c76b25be3/Documentation/admin-guide/kernel-parameters.txt#L627). **Kernel messages are logged to */dev/console* where a login prompt can be launched**.

3. */dev/tty0*

/dev/tty0 points to the active *tty* terminal. Let's explore further with some examples.

To access a *tty3* lets press CTRL + ALT + F3. We can confirm that we're using *tty3* by typing in the command *tty*:

```
$ tty
/dev/tty3
```

Let's see if we can write to the console using echo:

```
$ echo "hello world" > /dev/tty3
hello world
```

Okay, so far we've seen that we can change to a *tty N* by pressing CTRL + ALT + FN. Note however that there is not F0 key on the keyboard but there is a *tty0*.

Let's try and find a bit more about tty0:

```
$ cat /sys/devices/virtual/tty/tty0/active
tty3
```

Seems like *tty0* is pointing to *tty3*. Okay, let's try and switch to a separate *tty*. Let's say *tty4*. A quick shortcut would be to press CTRL + ALT + Right Arrow. This would take us from *tty3* to *tty4*.

```
$ tty
/dev/tty4
$ cat /sys/devices/virtual/tty/tty0/active
tty4
```

We can see that *tty0* is the current active console.

4. */dev/tty*

In this section, we'll explore **how */dev/tty* is different from */dev/tty0*. */dev/tty* is an alias to the terminal of the active process, regardless of the type of terminal**. For instance, this can be the terminal associated with *bash* or *sshd*.

According to the POSIX standard (https://pubs.opengroup.org/onlinepubs/009695399/basedefs/xbd_chap10.html), */dev/tty* is considered to be the terminal associated with a process

group whereas */dev/tty0* is associated with a virtual terminal.

Let's see what that means in practice.

First, let's ssh into the local machine and check our *tty*:

```
$ ssh localhost
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.4.0-37-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 updates can be installed immediately.
0 of these updates are security updates.

Last login: Tue Jun 16 10:59:08 2020 from 127.0.0.1
$ tty
/dev/pts/0
```

This is a *pts*, which is a different type of terminal. Now let's check whether printing "hello world" to */dev/tty0* affects our ssh session:

Note that only root can write to */dev/tty0*, therefore, **we should switch to root**:

```
$ sudo -i
$ echo "hello world" > /dev/tty0
$ # Nothing here was expecting hello world.
$ exit # let's leave root.
```

The message, "hello world" is not printed. If we check the active *tty* however, we should be able to find that "hello world" is printed there.

Let's try using */dev/tty*:

```
$ echo "hello world" > /dev/tty
hello world
```

The message, "hello world" is printed! This is because ***/dev/tty* represents the terminal which is associated with the ssh session**. Regardless of what terminal is in use, */dev/tty* is a quick way to access the current terminal.

5. Conclusion

In this tutorial, we've looked at */dev/console* and its role. We then looked at *tty* and *tty0* and seen the difference between the two. */dev/tty* is the current active *tty* and */dev/tty0* is a terminal associated with the current process.

If you have a few years of experience in the Linux ecosystem, and you're interested in sharing that experience with the community, have a look at our **Contribution Guidelines** (/linux/contribution-guidelines).

Comments are closed on this article!