

TTY Driver and TTY Operations

- [Allocation](#)
 - [TTY Driver Flags](#)
- [Registration](#)
 - [Registering Devices](#)
 - [Linking Devices to Ports](#)
- [TTY Driver Reference](#)
- [TTY Operations Reference](#)

Allocation

The first thing a driver needs to do is to allocate a struct `tty_driver`. This is done by `tty_alloc_driver()` (or `__tty_alloc_driver()`). Next, the newly allocated structure is filled with information. See [TTY Driver Reference](#) at the end of this document on what actually shall be filled in.

The allocation routines expect a number of devices the driver can handle at most and flags. Flags are those starting `TTY_DRIVER_` listed and described in [TTY Driver Flags](#) below.

When the driver is about to be freed, `tty_driver_kref_put()` is called on that. It will decrements the reference count and if it reaches zero, the driver is freed.

For reference, both allocation and deallocation functions are explained here in detail:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\tty\linux-master) (Documentation) (tty) tty_driver.rst, line 28)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/tty/tty_io.c
   :identifiers: __tty_alloc_driver tty_driver_kref_put
```

TTY Driver Flags

Here comes the documentation of flags accepted by `tty_alloc_driver()` (or `__tty_alloc_driver()`):

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\tty\linux-master) (Documentation) (tty) tty_driver.rst, line 37)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: include/linux/tty_driver.h
   :doc: TTY Driver Flags
```

Registration

When a struct `tty_driver` is allocated and filled in, it can be registered using `tty_register_driver()`. It is recommended to pass `TTY_DRIVER_DYNAMIC_DEV` in flags of `tty_alloc_driver()`. If it is not passed, *all* devices are also registered during `tty_register_driver()` and the following paragraph of registering devices can be skipped for such drivers. However, the struct `tty_port` part in [Registering Devices](#) is still relevant there.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\tty\linux-master) (Documentation) (tty) tty_driver.rst, line 52)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/tty/tty_io.c
   :identifiers: tty_register_driver tty_unregister_driver
```

Registering Devices

Every TTY device shall be backed by a struct `tty_port`. Usually, TTY drivers embed `tty_port` into device's private structures. Further details about handling `tty_port` can be found in `:doc:`tty_port``. The driver is also recommended to use `tty_port`'s reference counting by `tty_port_get()` and `tty_port_put()`. The final put is supposed to free the `tty_port` including the device's private struct.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\tty\ (linux-master) (Documentation) (tty) tty_driver.rst, line 58);
[backlink](#)

Unknown interpreted text role "doc".

Unless `TTY_DRIVER_DYNAMIC_DEV` was passed as flags to `tty_alloc_driver()`, TTY driver is supposed to register every device discovered in the system (the latter is preferred). This is performed by `tty_register_device()`. Or by `tty_register_device_attr()` if the driver wants to expose some information through struct attribute_group. Both of them register `index`'th device and upon return, the device can be opened. There are also preferred `tty_port` variants described in [Linking Devices to Ports](#) later. It is up to driver to manage free indices and choosing the right one. The TTY layer only refuses to register more devices than passed to `tty_alloc_driver()`.

When the device is opened, the TTY layer allocates struct `tty_struct` and starts calling operations from `c:member:'tty_driver.ops'`, see [TTY Operations Reference](#).

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\tty\ (linux-master) (Documentation) (tty) tty_driver.rst, line 74);
[backlink](#)

Unknown interpreted text role "c:member".

The registration routines are documented as follows:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\tty\ (linux-master) (Documentation) (tty) tty_driver.rst, line 80)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/tty/tty_io.c
   :identifiers: tty_register_device tty_register_device_attr
               tty_unregister_device
```

Linking Devices to Ports

As stated earlier, every TTY device shall have a struct `tty_port` assigned to it. It must be known to the TTY layer at `c:member:'tty_driver.ops.install()` at latest. There are few helpers to *link* the two. Ideally, the driver uses `tty_port_register_device()` or `tty_port_register_device_attr()` instead of `tty_register_device()` and `tty_register_device_attr()` at the registration time. This way, the driver needs not care about linking later on.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\tty\ (linux-master) (Documentation) (tty) tty_driver.rst, line 88);
[backlink](#)

Unknown interpreted text role "c:member".

If that is not possible, the driver still can link the `tty_port` to a specific index *before* the actual registration by `tty_port_link_device()`. If it still does not fit, `tty_port_install()` can be used from the `c:member:'tty_driver.ops.install'` hook as a last resort. The last one is dedicated mostly for in-memory devices like PTY where `tty_ports` are allocated on demand.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\tty\ (linux-master) (Documentation) (tty) tty_driver.rst, line 95);
[backlink](#)

Unknown interpreted text role "c:member".

The linking routines are documented here:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\tty\ (linux-master) (Documentation) (tty) tty_driver.rst, line 104)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/tty/tty_port.c
   :identifiers: tty_port_link_device tty_port_register_device
               tty_port_register_device_attr
```

TTY Driver Reference

All members of struct `tty_driver` are documented here. The required members are noted at the end. struct `tty_operations` are documented next.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\tty\linux-master) (Documentation) (tty) tty_driver.rst, line 116)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: include/linux/tty_driver.h
   :identifiers: tty_driver
```

TTY Operations Reference

When a TTY is registered, these driver hooks can be invoked by the TTY layer:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\tty\linux-master) (Documentation) (tty) tty_driver.rst, line 126)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: include/linux/tty_driver.h
   :identifiers: tty_operations
```