ioctl MEDIA_REQUEST_IOC_QUEUE

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\mediactl\(linux-master) (Documentation) (userspace-api) (media) (mediactl)media-request-ioc-queue.rst`, line 2)

Unknown directive type "c:namespace".

```
.. c:namespace:: MC
```

Name

MEDIA_REQUEST_IOC_QUEUE - Queue a request

Synopsis

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\mediactl\(linux-master) (Documentation) (userspace-api) (media) (mediactl)media-request-ioc-queue.rst`, line 18)

Unknown directive type "c:macro".

```
.. c:macro:: MEDIA_REQUEST_IOC_QUEUE
```

`int ioctl(int request_fd, MEDIA_REQUEST_IOC_QUEUE)`

Arguments

`request_fd`

File descriptor returned by :ref:`MEDIA_IOC_REQUEST_ALLOC`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\mediactl\(linux-master) (Documentation) (userspace-api) (media) (mediactl)media-request-ioc-queue.rst`, line 26); *backlink*

Unknown interpreted text role "ref".

Description

If the media device supports :ref:`requests <media-request-api>`, then this request ioctl can be used to queue a previously allocated request.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\mediactl\(linux-master) (Documentation) (userspace-api) (media) (mediactl)media-request-ioc-queue.rst`, line 31); *backlink*

Unknown interpreted text role "ref".

If the request was successfully queued, then the file descriptor can be :ref:`polled <request-func-poll>` to wait for the request to complete.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\mediactl\(linux-master) (Documentation) (userspace-api) (media) (mediactl)media-request-ioc-queue.rst`, line 34); *backlink*

Unknown interpreted text role "ref".

If the request was already queued before, then `EBUSY` is returned. Other errors can be returned if the contents of the request contained invalid or inconsistent data, see the next section for a list of common error codes. On error both the request and driver state are unchanged.

Once a request is queued, then the driver is required to gracefully handle errors that occur when the request is applied to the hardware. The exception is the EIO error which signals a fatal error that requires the application to stop streaming to reset the hardware state.

It is not allowed to mix queuing requests with queuing buffers directly (without a request). EBUSY will be returned if the first buffer was queued directly and you next try to queue a request, or vice versa.

A request must contain at least one buffer, otherwise this ioctl will return an ENOENT error.

Return Value

On success 0 is returned, on error -1 and the errno variable is set appropriately. The generic error codes are described at the :ref:`Generic Error Codes <gen-errors>` chapter.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\mediactl\(linux-master)(Documentation)(userspace-api)(media)(mediactl)media-request-ioc-queue.rst, line 57); *backlink*

Unknown interpreted text role "ref".

EBUSY

The request was already queued or the application queued the first buffer directly, but later attempted to use a request. It is not permitted to mix the two APIs.

ENOENT

The request did not contain any buffers. All requests are required to have at least one buffer. This can also be returned if some required configuration is missing in the request.

ENOMEM

Out of memory when allocating internal data structures for this request.

EINVAL

The request has invalid data.

EIO

The hardware is in a bad state. To recover, the application needs to stop streaming to reset the hardware state and then try to restart streaming.