

[Metal](#) / MTL4CommandQueue

Protocol


MTL4CommandQueue

An abstraction representing a command queue that you use commit and synchronize command buffers and to perform other GPU operations.

iOS 26.0+ | iPadOS 26.0+ | Mac Catalyst 26.0+ | macOS 26.0+ | tvOS 26.0+ | visionOS 26.0+

```
protocol MTL4CommandQueue : NSObjectProtocol, Sendable
```

Mentioned in

 Understanding the Metal 4 core API

Topics

Instance Properties

```
var device: any MTLDevice
```

Returns the GPU device that the command queue belongs to.

Required

```
var label: String?
```

Obtains this queue's optional label for debugging purposes.

Required

Instance Methods

```
func addResidencySet(any MTLResidencySet)
```

Applies a residency set to a queue, which Metal applies to the queue's command buffers as you commit them.

Required

```
func addResidencySets([any MTLResidencySet])
```

Applies multiple residency sets to a queue, which Metal applies to the queue's command buffers as you commit them.

```
func commit([any MTL4CommandBuffer], options: MTL4CommitOptions?)
```

Enqueues an array of command buffer instances for execution with a set of options.

```
func copyMappings(sourceBuffer: any MTLBuffer, destinationBuffer: any MTLBuffer, operations: [MTL4CopySparseBufferMappingOperation])
```

Copies multiple offsets within a source placement sparse buffer to a destination placement sparse buffer.

```
func copyMappings(sourceTexture: any MTLTexture, destinationTexture: any MTLTexture, operations: [MTL4CopySparseTextureMappingOperation])
```

Copies multiple regions within a source placement sparse texture to a destination placement sparse texture.

```
func removeResidencySet(any MTLResidencySet)
```

Removes a residency set from a command queue's list, which means Metal doesn't apply it to the queue's command buffers as you commit them.

Required

```
func removeResidencySets([any MTLResidencySet])
```

Removes multiple residency sets from a command queue's list, which means Metal doesn't apply them to the queue's command buffers as you commit them.

```
func signalDrawable(any MTLDrawable)
```

Schedules a signal operation on the command queue to indicate when rendering to a Metal drawable is complete.

Required

```
func signalEvent(any MTLEvent, value: UInt64)
```

Schedules an operation to signal a GPU event with a specific value after all GPU work prior to this point is complete.

Required

```
func updateMappings(buffer: any MTLBuffer, heap: (any MTLHeap)?, operations: [MTL4UpdateSparseBufferMappingOperation])
```

Updates multiple regions within a placement sparse buffer to alias specific tiles from a Metal heap.

```
func updateMappings(texture: any MTLTexture, heap: (any MTLHeap)?,
operations: [MTL4UpdateSparseTextureMappingOperation])
```

Updates multiple regions within a placement sparse texture to alias specific tiles of a Metal heap.

```
func waitForDrawable(any MTLDrawable)
```

Schedules a wait operation on the command queue to ensure the display is no longer using a specific Metal drawable.

Required

```
func waitForEvent(any MTLEvent, value: UInt64)
```

Schedules an operation to wait for a GPU event of a specific value before continuing to execute any future GPU work.

Required

Relationships

Inherits From

NSObjectProtocol, Sendable, SendableMetatype

See Also

Submitting work to a GPU with Metal 4

```
class MTL4CommandQueueDescriptor
```

Groups together parameters for the creation of a new command queue.

```
struct MTL4CommandQueueError
```

```
enum Code
```

Enumeration of kinds of errors that committing an array of command buffers instances can produce.

`let MTL4CommandQueueErrorDomain: String`

`protocol MTL4CommandBuffer`

Records a sequence of GPU commands.

`class MTL4CommandBufferOptions`

Options to configure a command buffer before encoding work into it.

`protocol MTL4CommandEncoder`

An encoder that writes GPU commands into a command buffer.

`struct MTL4RenderEncoderOptions`

Custom render pass options you specify at encoder creation time.

`protocol MTL4ArgumentTable`

Provides a mechanism to manage and provide resource bindings for buffers, textures, sampler states and other Metal resources.

`class MTL4ArgumentTableDescriptor`

Groups parameters for the creation of a Metal argument table.

`protocol MTL4CommandAllocator`

Manages the memory backing the encoding of GPU commands into command buffers.

`class MTL4CommandAllocatorDescriptor`

Groups together parameters for creating a command allocator.

`class MTL4CommitOptions`

Represents options to configure a commit operation on a command queue.

`protocol MTL4CommitFeedback`

Describes an object containing debug information from Metal to your app after completing a workload.

`typealias MTL4CommitFeedbackHandler`

Defines the block signature for a callback Metal invokes to provide your app feedback after completing a workload.