Protocol

MTL4ComputeCommandEncoder

Encodes a compute pass and other memory operations into a command buffer.

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

```
protocol MTL4ComputeCommandEncoder : MTL4CommandEncoder
```

Mentioned in

Understanding the Metal 4 core API

Overview

Use instances of this abstraction to encode a compute pass into `MTL4CommandBuffer` instances, as well as commands that copy and modify the underlying memory of various Metal resources, and commands that build or refit acceleration structures.

Topics

Instance Methods

```
func build(destinationAccelerationStructure: any MTLAcceleration
Structure, descriptor: MTL4AccelerationStructureDescriptor, scratch
Buffer: MTL4BufferRange)
```

Encodes an acceleration structure build into the command buffer.

Required

```
func copy(sourceAccelerationStructure: any MTLAccelerationStructure, destinationAccelerationStructure: any MTLAccelerationStructure)
```

Encodes an acceleration structure copy operation into the command buffer.

Required

```
func copy(sourceBuffer: any MTLBuffer, sourceOffset: Int, destinationBuffer: any MTLBuffer, destinationOffset: Int, size: Int)
```

Encodes a command that copies data from a buffer instance into another.

Required

```
func copy(sourceBuffer: any MTLBuffer, sourceOffset: Int, sourceBytesPerRow: Int, sourceBytesPerImage: Int, sourceSize: MTLSize, destinationTexture: any MTLTexture, destinationSlice: Int, destinationLevel: Int, destinationOrigin: MTLOrigin, options: MTLBlitOption)
```

Encodes a command to copy image data from a buffer into a texture with options for special texture formats.

```
func copy(sourceTensor: any MTLTensor, sourceOrigin: MTLTensorExtents, sourceDimensions: MTLTensorExtents, destinationTensor: any MTLTensor, destinationOrigin: MTLTensorExtents, destinationDimensions: MTLTensorExtents)
```

Encodes a command to copy data from a tensor instance into another.

Required

```
func copy(sourceTexture: any MTLTexture, destinationTexture: any MTLTexture)
```

Encodes a command that copies data from a texture to another.

Required

```
func copy(sourceTexture: any MTLTexture, sourceSlice: Int, sourceLevel: Int, destinationTexture: any MTLTexture, destinationSlice: Int, destinationLevel: Int, sliceCount: Int, levelCount: Int)
```

Encodes a command that copies slices of a texture to slices of another texture.

Required

```
func copy(sourceTexture: any MTLTexture, sourceSlice: Int, sourceLevel: Int, sourceOrigin: MTLOrigin, sourceSize: MTLSize, destinationBuffer: any MTLBuffer, destinationOffset: Int, destinationBytesPerRow: Int, destinationBytesPerImage: Int, options: MTLBlitOption)
```

Encodes a command that copies image data from a slice of a texture instance to a buffer, with options for special texture formats.

```
func copy(sourceTexture: any MTLTexture, sourceSlice: Int, sourceLevel:
Int, sourceOrigin: MTLOrigin, sourceSize: MTLSize, destinationTexture:
any MTLTexture, destinationSlice: Int, destinationLevel: Int,
destinationOrigin: MTLOrigin)
```

Encodes a command that copies image data from a slice of a texture into a slice of another texture.

Required

```
func copyAndCompact(sourceAccelerationStructure: any MTLAcceleration
Structure, destinationAccelerationStructure: any MTLAcceleration
Structure)
```

Encodes a command to copy and compact an acceleration structure.

Required

```
func copyCommands(sourceBuffer: any MTLIndirectCommandBuffer, source
Range: Range<Int>, destinationBuffer: any MTLIndirectCommandBuffer,
destinationIndex: Int)
```

Encodes a command that copies commands from one indirect command buffer into another.

```
func dispatchThreadgroups(indirectBuffer: MTLGPUAddress, threadsPer
Threadgroup: MTLSize)
```

Encodes a compute dispatch command with a grid that aligns to threadgroup boundaries, using an indirect buffer for arguments.

Required

```
func dispatchThreadgroups(threadgroupsPerGrid: MTLSize, threadsPer
Threadgroup: MTLSize)
```

Encodes a compute dispatch command with a grid that aligns to threadgroup boundaries.

Required

```
func dispatchThreads(indirectBuffer: MTLGPUAddress)
```

Encodes a compute dispatch command with an arbitrarily sized grid, using an indirect buffer for arguments.

Required

```
func dispatchThreads(threadsPerGrid: MTLSize, threadsPerThreadgroup:
MTLSize)
```

Encodes a compute dispatch command using an arbitrarily-sized grid.

Required

```
func executeCommands(buffer: any MTLIndirectCommandBuffer, indirect
Buffer: MTLGPUAddress)
```

Encodes an instruction to execute commands from an indirect command buffer, using an indirect buffer for arguments.

Required

`func executeCommands(buffer: any MTLIndirectCommandBuffer, range: Range<Int>)`

Encodes a command to execute commands from an indirect command buffer.

`func fill(buffer: any MTLBuffer, range: Range<Int>, value: UInt8)`

Encodes a command that fills a buffer with a constant value for each byte.

`func generateMipmaps(texture: any MTLTexture)`

Encodes a command that generates mipmaps for a texture instance from the base mipmap level up to the highest mipmap level.

Required

`func optimizeCommands(buffer: any MTLIndirectCommandBuffer, range: Range<Int>)`

Encode a command to attempt to improve the performance of a range of commands within an indirect command buffer.

`func optimizeContents(forCPUAccess: any MTLTexture)`

Encodes a command that modifies the contents of a texture to improve the performance of CPU accesses to its contents.

Required

`func optimizeContents(forCPUAccess: any MTLTexture, slice: Int, level: Int)`

Encodes a command that modifies the contents of a texture to improve the performance of CPU accesses to its contents in a specific region.

Required

`func optimizeContents(forGPUAccess: any MTLTexture)`

Encodes a command that modifies the contents of a texture to improve the performance of GPU accesses to its contents.

Required

`func optimizeContents(forGPUAccess: any MTLTexture, slice: Int, level: Int)`

Encodes a command that modifies the contents of a texture instance to improve the performance of GPU accesses to its contents in a specific region.

Required

`func refit(sourceAccelerationStructure: any MTLAccelerationStructure, descriptor: MTL4AccelerationStructureDescriptor, destinationAccelerationStructure: (any MTLAccelerationStructure)?, scratchBuffer: MTL4BufferRange, options: MTLAccelerationStructureRefitOptions)`

Encodes an acceleration structure refit operation into the command buffer, providing additional options.

Required

`func resetCommands(buffer: any MTLIndirectCommandBuffer, range: Range<Int>)`

Encodes a command that resets a range of commands in an indirect command buffer.

`func setArgumentTable((any MTL4ArgumentTable)?)`

Sets an argument table for the compute shader stage of this pipeline.

Required

`func setComputePipelineState(any MTLComputePipelineState)`

Configures this encoder with a compute pipeline state that applies to your subsequent dispatch commands.

Required

`func setImageblockSize(width: Int, height: Int)`

Specifies the size, in pixels, of imageblock data in tile memory.

Required

`func setThreadgroupMemoryLength(Int, index: Int)`

Configures the size of a threadgroup memory buffer for a threadgroup argument in the compute shader function.

Required

`func stages() -> MTLStages`

Queries a bitmask representing the shader stages on which commands currently present in this command encoder operate.

Required

`func writeCompactedSize(sourceAccelerationStructure: any MTLAccelerationStructure, destinationBuffer: MTL4BufferRange)`

Encodes a command to compute the size an acceleration structure can compact into, writing the result into a buffer.

Required

`func writeTimestamp(granularity: MTL4TimestampGranularity, counterHeap: any MTL4CounterHeap, index: Int)`

Writes a GPU timestamp into a heap.

Required

Relationships

Inherits From

`MTL4CommandEncoder`, `NSObjectProtocol`

See Also

Encoding a compute pass

Creating threads and threadgroups

Learn how Metal organizes compute-processing workloads.

Calculating threadgroup and grid sizes

Calculate the optimum sizes for threadgroups and grids when dispatching compute-processing workloads.

`protocol MTLComputeCommandEncoder`

An interface for dispatching commands to encode in a compute pass.