Protocol

MTLBuffer

A resource that stores data in a format defined by your app.

iOS 8.0+ | iPadOS 8.0+ | Mac Catalyst 13.1+ | macOS 10.11+ | tvOS | visionOS 1.0+

```
protocol MTLBuffer : MTLResource
```

Mentioned in

- Improving CPU performance by using argument buffers
- Estimating how often a texture region is accessed
- Setting resource storage modes
- Converting a GPU's counter data into a readable format
- Synchronizing a managed resource in macOS

Overview

An `MTLBuffer` instance can be used only with the `MTLDevice` that created it. Don't implement this protocol yourself; instead, use the following `MTLDevice` methods to create `MTLBuffer` instances:

- `makeBuffer(length:options:)` creates a `MTLBuffer` instance with a new storage allocation.
- `makeBuffer(bytes:length:options:)` creates a `MTLBuffer` instance by copying data from an existing storage allocation into a new allocation.
- `makeBuffer(bytesNoCopy:length:options:deallocator:)` creates a `MTLBuffer` instance that reuses an existing storage allocation and does not allocate any new storage.

The Metal framework doesn't know anything about the contents of an `MTLBuffer`, just its size. You define the format of the data in the buffer and ensure that your app and your shaders know how to read and write the data. For example, you might create a struct in your shader that defines the data you want to store in the buffer and its memory layout.

If you create a buffer with a managed resource storage mode (`MTLStorageMode.managed`), you must call `didModifyRange:` to tell Metal to copy any changes to the GPU.

Topics

Creating a texture that shares buffer data

```
func makeTexture(descriptor: MTLTextureDescriptor, offset: Int, bytesPerRow: Int) -> (any MTLTexture)?
```

Creates a texture that shares its storage with the buffer.

Required

Reading the buffer's data on the CPU

```
func contents() -> UnsafeMutableRawPointer
```

Gets the system address of the buffer's storage allocation.

Required

Synchronizing data to the GPU for managed buffers

```
func didModifyRange(Range<Int>)
```

Informs the GPU that the CPU has modified a section of the buffer.

Debugging buffers

```
func addDebugMarker(String, range: Range<Int>)
```

Adds a debug marker string to a specific buffer range.

```
func removeAllDebugMarkers()
```

Removes all debug marker strings from the buffer.

Required

Reading buffer length

```
var length: Int
```

The logical size of the buffer, in bytes.

Required

Creating views of buffers on other GPUs

```
func makeRemoteBufferView(any MTLDevice) -> (any MTLBuffer)?
```

Creates a remote view of the buffer for another GPU in the same peer group.

Required

```
var remoteStorageBuffer: (any MTLBuffer)?
```

The buffer on another GPU that the buffer was created from, if any.

Required

Instance Properties

```
var gpuAddress: MTLGPUAddress
```

Required

```
var sparseBufferTier: MTLBufferSparseTier
```

Required

Instance Methods

```
func makeTensor(descriptor: MTLTensorDescriptor, offset: Int) throws -> any MTLTensor
```

Creates a tensor that shares storage with this buffer.

Required

Relationships

Inherits From

`MTLAllocation`, `MTLResource`, `NSObjectProtocol`