Protocol

MTLLibrary

A collection of Metal shader functions.

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

```
protocol MTLLibrary : NSObjectProtocol, Sendable
```

Mentioned in

- Building a shader library by precompiling source files
- Logging shader debug messages

Overview

An `MTLLibrary` instance contains Metal shading language source code compiled during an app's build process or at runtime from a text string.

Don't implement this protocol yourself; instead, use the library creation methods provided by the `MTLDevice` protocol. To create an `MTLLibrary` from a precompiled Metal library binary, call one of these `MTLDevice` methods:

- `makeDefaultLibrary()`
- `makeLibrary(filepath:)`
- `makeLibrary(data:)`

To create an `MTLLibrary` by compiling source code at runtime, call one of these `MTLDevice` methods:

- `makeLibrary(source:options:completionHandler:)`

- makeLibrary(source:options:)

Topics

Querying basic library attributes

`var installName: String?`

The installation name for a dynamic library.

Required

`var type: MTLLibraryType`

The library's basic type.

Required

Querying library contents

`var functionNames: [String]`

The names of all public functions in the library.

Required

Creating shader function instances

`func makeFunction(name: String) -> (any MTLFunction)?`

Creates an instance that represents a shader function in the library.

Required

`func makeFunction(name: String, constantValues: MTLFunctionConstantValues, completionHandler: ((any MTLFunction)?, (any Error)?) -> Void)`

Asynchronously creates a specialized shader function.

Required

`func makeFunction(name: String, constantValues: MTLFunctionConstantValues) throws -> any MTLFunction`

Synchronously creates a specialized shader function.

Required

`func makeFunction(descriptor: MTLFunctionDescriptor, completionHandler: ((any MTLFunction)?, (any Error)?) -> Void)`

Asynchronously creates an object representing a shader function, using the specified descriptor.

Required

```
func makeFunction(descriptor: MTLFunctionDescriptor) throws -> any MTLFunction
```

Synchronously creates an object representing a shader function, using the specified descriptor.

Required

Creating intersection function instances

```
func makeIntersectionFunction(descriptor: MTLIntersectionFunctionDescriptor, completionHandler: ((any MTLFunction)?, (any Error)?) -> Void)
```

Asynchronously creates an object representing a ray-tracing intersection function, using the specified descriptor.

Required

```
func makeIntersectionFunction(descriptor: MTLIntersectionFunctionDescriptor) throws -> any MTLFunction
```

Synchronously creates an object representing a ray-tracing intersection function, using the specified descriptor.

Required

Identifying the library

```
var device: any MTLDevice
```

The Metal device object that created the library.

Required

```
var label: String?
```

A string that identifies the library.

Required

Instance Methods

```
func reflection(functionName: String) -> MTLFunctionReflection?
```

Retrieves reflection information for a function in the library.

Required

Relationships

Inherits From

`NSObjectProtocol`, `Sendable`, `SendableMetatype`

See Also

Shader library management

`protocol` `MTLDynamicLibrary`

A dynamically linkable representation of compiled shader code for a specific Metal device object.

`protocol` `MTLBinaryArchive`

A container for pipeline state descriptors and their associated compiled shader code.

`class` `MTLCompileOptions`

Compilation settings for a Metal shader library.

`enum` `MTLLibraryType`

A set of options for Metal library types.

`enum` `MTLLanguageVersion`

Metal shading language versions.

`enum` `MTLCompileSymbolVisibility`

`enum` `MTLLibraryOptimizationLevel`

The optimization options for the Metal compiler.