

Framework

# Spatial

Create and manipulate 3D mathematical primitives.

iOS 16.0+ | iPadOS 16.0+ | Mac Catalyst 16.0+ | macOS 13.0+ | tvOS 16.0+ | visionOS 1.0+ | watchOS 9.0+

## Overview

The Spatial module is a lightweight 3D mathematical library that provides a simple API for working with 3D primitives. Much of its functionality is similar to the 2D geometry support in Core Graphics, but in three dimensions.

## Topics

### Data structures

```
struct Vector3D
```

A three-element vector.

```
struct Vector3DFloat
```

A single-precision structure that defines a three-element vector

```
struct Axis3D
```

Constants that describe an axis.

### 2D primitives

```
struct Angle2D
```

A geometric angle with a value you access in either radians or degrees.

`struct Angle2DFloat`

A single-precision geometric angle whose value you access in either radians or degrees.

## 3D primitives

`struct Point3D`

A point in a 3D coordinate system.

`struct Point3DFloat`

A single-precision structure that contains a point in a three-dimensional coordinate system.

`struct Size3D`

A size that describes width, height, and depth in a 3D coordinate system.

`struct Size3DFloat`

A single-precision structure that contains width, height, and depth values.

`struct Rect3D`

A rectangle in a 3D coordinate system.

`struct Rect3DFloat`

A single-precision structure that contains the location and dimensions of a 3D rectangle.

`struct Rotation3D`

A rotation in three dimensions.

`struct Rotation3DFloat`

A single-precision structure that represents a rotation in three dimensions.

`struct RotationAxis3D`

A 3D rotation axis.

`struct RotationAxis3DFloat`

A 3D axis.

`struct Pose3D`

A structure that contains a 3D position and a 3D rotation.

`struct Pose3DFloat`

A single-precision structure that contains a position and rotation.

`struct ScaledPose3D`

A structure that contains a position, rotation, and scale.

```
struct ScaledPose3DFloat
```

A structure that contains a position, rotation, and scale.

```
struct SphericalCoordinates3D
```

A structure that defines spherical coordinates in radial, inclination, azimuthal order.

```
struct SphericalCoordinates3DFloat
```

A single-precision structure that defines spherical coordinates in radial, inclination, azimuthal order.

```
struct Ray3D
```

A ray in a 3D coordinate system.

```
struct Ray3DFloat
```

A single-precision structure that contains the origin and direction of a 3D ray.

## Affine and projective transforms

```
struct AffineTransform3D
```

A 3D affine transformation matrix.

```
struct AffineTransform3DFloat
```

```
struct ProjectiveTransform3D
```

A 3D projective transformation matrix.

```
struct ProjectiveTransform3DFloat
```

A single-precision 3D projective transformation matrix.

## Converting between coordinate spaces

```
protocol CoordinateSpace3D
```

A type that represents a coordinate space which you can use to convert values to and from other coordinate spaces.

```
protocol CoordinateSpace3DFloat
```

```
protocol CoordinateSpaceValue3D
```

An opaque value which can be resolved to a concrete value in a `CoordinateSpace3D`

```
protocol ProjectiveTransformable3D
```

```
protocol ProjectiveTransformable3DFloat
```

```
struct WorldReferenceCoordinateSpace
```

A coordinate space that represents a world reference point.

## Applying trigonometric functions

```
func cos(Angle2D) -> Double
```

```
func cos(Angle2DFloat) -> Float
```

```
func cosh(Angle2D) -> Double
```

```
func cosh(Angle2DFloat) -> Float
```

```
func sin(Angle2DFloat) -> Float
```

```
func sin(Angle2D) -> Double
```

```
func sinh(Angle2D) -> Double
```

```
func sinh(Angle2DFloat) -> Float
```

```
func tan(Angle2D) -> Double
```

```
func tan(Angle2DFloat) -> Float
```

```
func tanh(Angle2D) -> Double
```

```
func tanh(Angle2DFloat) -> Float
```

## Protocols

```
protocol Primitive3D
```

A set of methods common to Spatial primitives.

```
protocol Rotatable3D
```

A set of methods that defines the interface to rotate Spatial entities.

```
protocol Scalable3D
```

A set of methods that defines the interface to scale Spatial entities.

```
protocol Shearable3D
```

A set of methods that defines the interface to shear Spatial entities.

```
protocol Translatable3D
```

A set of methods that defines the interface to translate Spatial entities.

`protocol Volumetric`

A set of methods for working with Spatial primitives with volume.

`protocol ClampableWithinRectProtocol`

A set of methods that defines the interface for Spatial entities that can be clamped to a volume.

`protocol Primitive3DProtocol`

A set of methods common to Spatial primitives.

`protocol Rotatable3DProtocol`

A set of methods that defines the interface for Spatial entities that can rotate.

`protocol Scalable3DProtocol`

A set of methods that defines the interface for Spatial entities that can scale.

`protocol Shearable3DProtocol`

A set of methods that defines the interface for Spatial entities that can shear.

`protocol SpatialTypeProtocol`

`protocol Transform3DProtocol`

A set of methods that are common to transforms.

`protocol Translatable3DProtocol`

A set of methods that defines the interface for Spatial entities that can translate.

`protocol VolumetricProtocol`

A set of methods for working with Spatial primitives with volume.

## Macros

☰ Macros & Global Variables

## Structures

`struct EulerAnglesFloat`

## Enumerations

enum AxisWithFactorsFloat

The axis of a shear transform.