Documentation Language: Swift

SwiftData / Model()

Macro

Model()

Converts a Swift class into a stored model that’s managed by SwiftData.

iOS 17.0+ | iPadOS 17.0+ | Mac Catalyst 17.0+ | macOS 14.0+ | tvOS 17.0+ | visionOS 1.0+ | watchOS 10.0+ | Swift 5.9+

```
@attached(member, conformances: Observable, Persistent
Model, Sendable, names: named(_$backingData), named(persistentBackingData), named(s
Model, Sendable) macro Model()
```

Mentioned in

Preserving your app’s model data across launches

Overview

Annotate your model classes with the `@Model` macro to make them persistable. At build time, the macro expands to provide conformance to the `PersistentModel` and `Observable` protocols.

```swift
@Model
class RemoteImage {
    var sourceURL: URL
    var data: Data

    init(sourceURL: URL, data: Data = Data()) {
        self.sourceURL = sourceURL
        self.data = data
    }
}
```

For more information about defining models, see Preserving your app's model data across launches.

See Also

Model definition

`macro Attribute(Schema.Attribute.Option..., originalName: String?, hashModifier: String?)`

Specifies the custom behavior that SwiftData applies to the annotated property when managing the owning class.

`macro Unique<T>([PartialKeyPath<T>]...)`

Specifies the key-paths that SwiftData uses to enforce the uniqueness of model instances.

`macro Index<T>([PartialKeyPath<T>]...)`

Specifies the key-paths that SwiftData uses to create one or more binary indices for the associated model.

`macro Index<T>(Schema.Index<T>.Types<T>...)`

Specifies the key-paths that SwiftData uses to create one or more indicies for the associated model, where each index is either binary or R-tree.

{} Defining data relationships with enumerations and model classes

Create relationships for static and dynamic data stored in your app.

```
macro Relationship(Schema.Relationship.Option..., deleteRule: Schema.Relationship.DeleteRule, minimumModelCount: Int?, maximumModelCount: Int?, originalName: String?, inverse: AnyKeyPath?, hashModifier: String?)
```

Specifies the options that SwiftData needs to manage the annotated property as a relationship between two models.

```
macro Transient()
```

Tells SwiftData not to persist the annotated property when managing the owning class.