SwiftData / Attribute(_:originalName:hashModifier:)

Macro

Attribute(_:originalName:hashModifier:)

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

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+

```swift
@attached(peer)
macro Attribute(
    _ options: Schema.Attribute.Option...,
    originalName: String? = nil,
    hashModifier: String? = nil
)
```

Parameters

`options`

A list of options to apply to the attached property to customize its behavior. For possible values, see `Schema.Attribute.Option`.

`originalName`

The previous name of the attribute, if it’s different to the one in the current schema version. The default value is `nil`.

`hashModifier`

A unique hash value that represents the most recent version of the attached property. The default value is `nil`.

Mentioned in

- Fetching and filtering time-based model changes
- Preserving your app's model data across launches

Overview

The framework's default behavior for managing a model class's stored properties is suitable for most use cases. However, if you need to alter the persistence behavior of a particular property, annotate it with the `@Attribute` macro. For example, you may want to avoid conflicts in your model data by specifying that an attribute's value is unique across all instances of that model.

```
@Model
class RemoteImage {
    @Attribute(.unique) var sourceURL: URL
    var data: Data

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

See Also

Model definition

`macro Model()`

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

`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.