Protocol

Equipment

A protocol for equipment that players directly interact with in a game.

visionOS 2.0+

```
protocol Equipment : Identifiable where Self.ID == EquipmentIdentifier
```

Overview

To represent equipment in your game, such as cards, pieces, and dice, following these steps:

- Create a structure that conforms to this protocol.
- Declare the `initialState` property as either `BaseEquipmentState`, `DieState`, or `CardState`, depending on the type of data you want TabletopKit to sync between players. For more complex data, use `RawValueState`.
- Declare the `id` property as a `EquipmentIdentifier` structure.
- Implement an initializer that sets the `id` and `initialState` properties.

Optionally, implement the `layoutChildren(for:visualState:)` method for equipment that represents groups, and the `restingOrientation(state:)` method to provide a custom resting orientation.

Topics

Gettting the initial state of the equipment

```
var initialState: Self.State
```

Required

```
associatedtype State : EquipmentState
```

Required

Displaying the equipment

```
func layoutChildren(for: TableSnapshot, visualState: TableVisualState)
-> any EquipmentLayout
```

This function provides the layout of the direct children of this equipment and is called whenever the snapshot changes. Override it to provide a custom layout. The output of this function is considered to be only a function of its inputs. Reaching out to data outside what is provided might result in undefined behavior.

Required Default implementation provided.

```
func restingOrientation(state: Self.State) -> Rotation3D
```

The resting orientation of the equipment given the current State.

Required Default implementations provided.

Relationships

Inherits From

`Identifiable`

Inherited By

`EntityEquipment`

See Also

Equipment

{} Implementing playing card overlap and physical characteristics

Add interactive card game behavior for a pile of playing cards with physically realistic stacking and overlapping.

`struct EquipmentCollection`

A collection of equipment whose state can be inspected and modified.

`protocol EntityEquipment`

A protocol for equipment in a game that you render using RealityKit.

`struct EquipmentIdentifier`

A unique identifier for equipment.

`protocol EquipmentState`

A protocol for the equipment data that TabletopKit syncs between players.

`struct EquipmentStateCollection`

A collection of equipment states that can be inspected and modified.

`struct BaseEquipmentState`

A state for equipment that contains no equipment-specific data.

`protocol CustomEquipmentState`

A specialized protocol for the equipment state that allows to accommodate custom data that TabletopKit syncs between players.

`protocol MutableEquipmentState`

A protocol for equipment data that TabletopKit syncs between players, and that can be mutated.

`struct CardState`

A state for cards that contains face up and down information.

`struct DieState`

A state for dice that contains the current value.

`struct RawValueState`

A state for equipment that contains a game-specific value.

`enum ControllingSeats`

The seats that can manipulate or interact with the equipment.