

---

[TabletopKit](#) / [TabletopInteraction](#) / TabletopInteraction.Delegate

Protocol

# TabletopInteraction.Delegate

A protocol for objects that manage the entire flow of players interacting with equipment.

visionOS 2.0+

protocol Delegate

---

## Overview

Implement the `update(interaction:)` method to take an appropriate action depending on the equipment and the phase of the interaction. For example, turn a card face up if a player flips it over or toss a die when a gesture ends.

---

## Topics

### Taking actions

```
func update(interaction: TabletopInteraction)

Required
```

### Starting a toss

```
func onTossStart(interaction: TabletopInteraction, outcomes: [TabletopInteraction.TossOutcome])
```

Implement `onTossStart(interaction:outcomes)` to be notified that the toss has just started simulating and to receive the outcome of the simulation. If the provided outcome is set to the equipment via actions, the equipment will retain the final state of the simulation even after the simulation ended.

**Required** Default implementation provided.

## Accepting interactions

```
func shouldAcceptDirectInteraction(initialValue: TabletopInteraction.  
Value, handoffValue: TabletopInteraction.Value?) -> TabletopInteraction  
.NewDirectInteractionIntent
```

Implement `shouldAcceptDirectInteraction(initialValue:handoffValue)` to provide the constants and initial configuration for the new direct interaction, and to decide if the new interaction should be accepted or rejected. If the function is not implemented, the default implementation will be used, which will call into the more generic `shouldAcceptInteraction(initialValue:handoffValue)`.

**Required** Default implementation provided.

```
func shouldAcceptIndirectInteraction(initialValue: TabletopInteraction.  
Value, handoffValue: TabletopInteraction.Value?) -> TabletopInteraction  
.NewIndirectInteractionIntent
```

Implement `shouldAcceptIndirectInteraction(initialValue:handoffValue)` to provide the constants and initial configuration for the new indirect interaction, and to decide if the new interaction should be accepted or rejected. If the function is not implemented, the default implementation will be used, which will call into the more generic `shouldAcceptInteraction(initialValue:handoffValue)`.

**Required** Default implementation provided.

```
func shouldAcceptInteraction(initialValue: TabletopInteraction.Value,  
handoffValue: TabletopInteraction.Value?) -> TabletopInteraction.New  
InteractionIntent
```

Implement `shouldAcceptInteraction(initialValue:handoffValue)` to provide the initial configuration for new interactions, and to decide if a new interaction should be accepted or rejected. If this function is not implemented, the default implementation will be used which will reject all handoff interactions and will accept all other interactions providing the default configuration.

**Required** Default implementation provided.

---

## See Also

## Performing actions

`struct TossOutcome`

An object representing the final outcome of tossing one equipment, as it appears at the end of its simulation.

`func addAction(some TabletopAction)`

Submit an action tied to this interaction. If the interaction gets canceled, all the associated actions will be automatically rolled back.

`func addAction(some CustomAction)`

Submit a custom action tied to this interaction. If the interaction gets canceled, all the associated actions will be automatically rolled back.

`func addActions(some Sequence<any TabletopAction>)`

Submit a collection of actions tied to this interaction. If the interaction gets canceled, all the associated actions will be automatically rolled back.

`func toss(equipmentID: EquipmentIdentifier, as: TossableRepresentation, linearVelocity: Vector3D?, angularVelocity: Vector3D?)`

Begins a simulation of a toss of the equipment with the specified parameters. Equipment that begins a toss in the same TabletopInteraction may interact with each other as well as the game's boundary.

`func end()`

Ends the current interaction.

`func cancel()`

Cancels the current interaction. All actions added to the interaction will also be cancelled.