Class

GCVirtualController

A software emulation of a real controller that you configure specifically for your game.

iOS 15.0+ | iPadOS 15.0+ | Mac Catalyst 15.0+

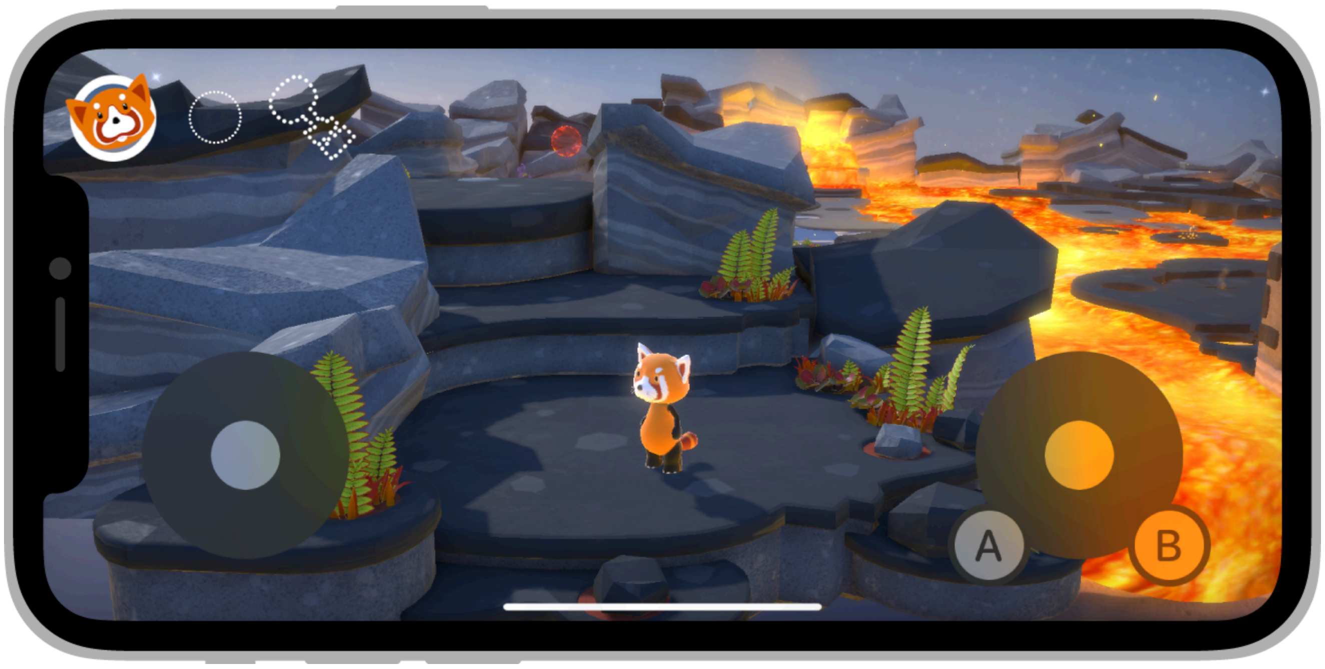
```
class GCVirtualController
```

Mentioned in

Adding touch controls to games that support game controllers in iOS

Overview

Use a virtual controller to display software controls that you can customize over your game. You create a virtual controller from a configuration where you choose the input elements to display. You can even customize the images for the elements. When you connect the controller to the device, users interact with it similarly to a real controller.



To add a virtual controller to your game, create a `GCVirtualController.Configuration` object containing the elements you want to appear in the controller. Then create the virtual controller by passing the configuration to the `init(configuration:)` method. Use the `connect(replyHandler:)` method to display the virtual controller on the screen.

To customize an element in the virtual controller, pass a new `GCVirtualController.ElementConfiguration` object for the element to the `updateConfiguration(forElement:configuration:)` method.

You process input from a virtual controller similarly to a real controller. Use the `controller` property to get the underlying `GCController` object. You can either poll the elements of the controller object or set the element's handlers to get callbacks when their input values change.

Topics

Creating virtual controllers

`init(configuration: GCVirtualController.Configuration)`

Creates a new virtual controller using the configuration you specify.

`class Configuration`

The configuration of a virtual controller.

Customizing the elements

```
func updateConfiguration(forElement: String, configuration: (GCVirtual
Controller.ElementConfiguration) -> GCVirtualController.Element
Configuration)
```

Changes the configuration for one of the virtual controller's input elements.

```
class ElementConfiguration
```

The properties of a virtual controller's element that you can customize.

Accessing the elements

```
var controller: GCController?
```

The underlying controller object that you use to access input elements.

Connecting and displaying virtual controllers

```
func connect(replyHandler: (((any Error)?) -> Void)?)
```

Connects the virtual controller to the device and displays it on the screen.

```
func disconnect()
```

Disconnects the virtual controller from the device and removes it from the screen.

Presenting a custom interface

```
func setPosition(CGPoint, forDirectionPadElement: String)
```

Changes the value of a directional pad element in the virtual controller.

```
func setValue(CGFloat, forButtonElement: String)
```

Changes the value of a button element in the virtual controller.

Relationships

Inherits From

`NSObject`

Conforms To

```
CVarArg
CustomDebugStringConvertible
CustomStringConvertible
Equatable
Hashable
NSObjectProtocol
```

See Also

Touch controller

Adding touch controls to games that support game controllers in iOS

Use touch input and virtual controllers to make your game available to players without controllers.