COLLAPSE SIDEBAR
Filter Classes
TYPE INDEX PAGES
CLASSES
ENUMS
DATA TYPES
LUA DOCUMENTS
GROUPED | CLASS TREE
ADORNMENTS
ANIMATIONS
AVATAR
BODY MOVERS
BUILDING
CONSTRAINTS
DATA
DEBUGGING
EFFECTS
GAMEPLAY
GUI
INPUT
JOINTS
LIGHTING
LOCALIZATION
MESHES
MONETIZATION
PATHFINDING
PLUGINS
POST PROCESSING
SCRIPTING
SETTINGS
SOCIAL
SOUNDS
UTILITY
VALUES

BindToRenderStep

Function of: RunService

Description:

The `BindToRenderStep` function binds a custom function to be called at a specific time during the render step. There are three main arguments for BindToRenderStep: `name`, `priority`, and `what function to call`.

As it is linked to the client's rendering process, BindToRenderStep can only be called on the client.

Name

The `name` parameter is a label for the binding, and can be used with `RunService:UnbindFromRenderStep` if the binding is no longer needed.

```
local RunService = game:GetService("RunService")

local function functionToBind() end

-- Bind the function above to the binding named "tempBinding"
RunService:BindToRenderStep("tempBinding", 1, functionToBind)
-- Unbind the function bound to "tempBinding"
RunService:UnbindFromRenderStep("tempBinding")
```

Priority

The `priority` of the binding is an integer, and determines when during the render step to call the custom function. The lower this number, the sooner the custom function will be called. If two bindings have the same priority the Roblox engine will randomly pick one to run first. The default Roblox control scripts run with these specific priorities:

- Player Input: 100
- Camera Controls: 200
  For convenience, the `RenderPriority` enum can be used to determine the integer value to set a binding. For example, to make a binding right before the default camera update, simply subtract 1 from the camera priority level.

Note: When using Enum.RenderPriority, remember to use ***InlineCode.Value*** at the end of the desired enum. BindToRenderStep will not work if just the enum on its own is used.

```
local RunService = game:GetService("RunService")

local function beforeCamera(delta)
    -- Code in here will run before the default Roblox camera script
end

RunService:BindToRenderStep("Before camera", Enum.RenderPriority.Camera.Value - 1, beforeCamera)
```

Custom Function and Delta Time

The last argument of BindToRenderStep is the `custom function` to call. This function will be passed one parameter called deltaTime. ***DeltaTime*** shows how much time passed between the beginning of the previous render step and the beginning of the current render step.

Note: All rendering updates will wait until the code in the render step finishes. Make sure that any code called by BindToRenderStep runs quickly and efficiently. If code in BindToRenderStep takes too long, then the game visuals will be choppy.

Parameters

Name	Type	Default	Description
name	string		The **name** parameter is a label for the binding, and can be used with `RunService.Unbind` if the binding is no longer needed
priority	int		The *priority* of the binding is an integer, and determines when during the render step to call the custom function. The lower this number, the sooner the custom function will be called. If two bindings have the same priority the Roblox engine will randomly pick one to run first. The default Roblox control scripts run with these specific priorities: • Player Input: 100 • Camera Controls: 200 For convenience, the "'RenderPriority"' enum can be used to determine the integer value to set a binding. For example, to make a binding right before the default camera update, simply subtract 1 from the camera priority level.
function	Function		The custom function being bound

Returns

Return Type	Summary
void	None.

Code Samples

RunService Custom Function

This example shows how to bind a simple function to the render step. All this function does is print how much time passed between the last render step and the current one. Note that this code will need to be in a `LocalScript` to run.

RUNSERVICE CUSTOM FUNCTION — EXPAND

```
-- Make variables for Roblox services
local RunService = game:GetService("RunService")

-- Function that will be bound to the render step
local function checkDelta(deltaTime)
    -- Print the time since the last render step
    print("Time since last render step:", deltaTime)
end

-- Bind the function
RunService:BindToRenderStep("Check delta", Enum.RenderPriority.Fir
```

COPY CODE

Bind and Unbind a Function

This example uses the `RunService` to bind and unbind a function named `printHello`. First, we bind the function to the RenderStep so that fires every *step*. Then, after we wait 5 seconds (`wait(5)`), we unbind the function.

Please note that we take caution to surround the function unbind in a pcall to prevent the code from breaking due to an error being thrown if the function name passed does not match the name of an already bound function. While we know that the function used in this example is bound when we try to unbind it, doing this is good coding practice.

BIND AND UNBIND A FUNCTION — EXPAND

```
local RunService = game:GetService("RunService")

-- Step 1: Declare the function and a name
local name = "Print Hello"
function printHello()
    print("Hello")
end

-- Step 3: Bind the function
RunService:BindToRenderStep(name, Enum.RenderPriority.First.Value,

-- Step 3: Unbind the function
local success, message = pcall(function() RunService:UnbindFromRen
if success then
    print("Success: Function unbound!")
else
```

COPY CODE

Frame Moving in Circle

This code sample moves a GuiObject in a circle within its parent object using RunService's BindToRenderStep. It defines a parametric equation in a function to help with positioning the GuiObject.

To try this code out, put a ScreenGui in the StarterGui. Inside the ScreenGui, insert a Frame with a LocalScript. Paste this code into the LocalScript, then play the game. Watch the Frame travel counterclockwise within.

FRAME MOVING IN CIRCLE — EXPAND

```
local RunService = game:GetService("RunService")

-- How fast the frame ought to move
local SPEED = 2

local frame = script.Parent
frame.AnchorPoint = Vector2.new(.5, .5)

-- A simple parametric equation of a circle
-- centered at (0.5, 0.5) with radius (0.5)
local function circle(t)
    return .5 + math.cos(t) * .5,
           .5 + math.sin(t) * .5
end

-- Keep track of the current time
```

COPY CODE