# Player

## Distance From Character

A screenshot of a computer

Description automatically generated

Link: <https://create.roblox.com/docs/reference/engine/classes/Player#DistanceFromCharacter>

## Getting Player Ping

A screenshot of a computer

Description automatically generated

Link: <https://create.roblox.com/docs/reference/engine/classes/Player#GetNetworkPing>

## Checking if the player loaded

The HasAppearanceLoaded Player function returns whether or not the appearance of the player's Player.Character has loaded.

A screenshot of a computer program

Description automatically generated

Link: <https://create.roblox.com/docs/reference/engine/classes/Player#HasAppearanceLoaded>

## Getting rank/role in a group

Takes in parameter of group id and returns a number between 0 (non-member) and 255 (group owner). Make sure to use this on the client because it does not update if a player leaves a group for example.

A screenshot of a computer program

Description automatically generated

Link: <https://create.roblox.com/docs/reference/engine/classes/Player#GetRankInGroup>

A screenshot of a computer program

Description automatically generated

Link: <https://create.roblox.com/docs/reference/engine/classes/Player#GetRoleInGroup>

## Checking if 2 players are friends

Takes in the 2nd player as parameter and returns a Boolean if they are friends or not.

A screen shot of a computer program

Description automatically generated

Link: <https://create.roblox.com/docs/reference/engine/classes/Player#IsFriendsWith>

# Get Parts in a PartA screenshot of a computer program Description automatically generated

Link: <https://create.roblox.com/docs/reference/engine/classes/WorldRoot#GetPartsInPart>

# Context Action Service (Player Inputs)

A context is simply a condition during which a player may perform some action. Some examples include holding a Tool, being seated in a car, or standing near a door. Whatever the case may be, it is up to your LocalScripts to call BindAction when the context is entered and UnbindAction when the context is left.

## Example:

A screen shot of a computer program

Description automatically generated

Link: <https://create.roblox.com/docs/reference/engine/classes/ContextActionService>

# Variadic Functions

There are two main uses for Variadic Functions:

1. To run a function with an x number of parameters. – *See: “Simple Use”*
2. To pass multiple parameters to other functions. – *See: “Argument Forwarding”*

### Example:

Simple Use:

A screen shot of a computer code

Description automatically generated

Argument Forwarding:A computer screen with white text

Description automatically generated

# Storing Functions in Variables and Tables

Functions being "first-class values" means that in Lua, functions are treated as regular values. They can be stored in variables and tables, passed as arguments to other functions, and returned as results from functions.

### Examples:

Functions Stored in Tables:

A screen shot of a computer program

Description automatically generated

Functions Stored in Variables:

A black background with white text

Description automatically generated

# Sorting Tables

You can use the table.sort() function to sort tables either alphabetically, reverse alphabetically, numerically or reverse numerically. It takes two parameters, the table to sort and the function that determines the sort order.

In the sort order function, using the operator “<” makes it alphabetical/numerical, using the opposite, “>” makes it reverse alphabetical/numerical. The parameters “a” and “b” are solely variables for the two indexes being compared. Returning the function returns the table in the order that is desired.

### Examples:

Sorting an array of names alphabetically.

A screen shot of a computer program

Description automatically generated

Sorting the names in a dictionary alphabetically

A computer screen shot of a program

Description automatically generated

# Protecting Functions

We can use protected calls also known as *“pcall”* to protect functions so that specific code runs if an error happens. In Roblox, from my current observation, pcall takes an anonymous function as the sole parameter. It returns as true if there are no errors and false if there are errors. IF there is an error, the error message is also returned. With this in mind, we can store the pcall function in two variables.

1. The resulting boolean from the pcall
2. The error message if there is one

### Example:

Printing the result of the pcall

A computer screen shot of text

Description automatically generated

From the example above, when i = 0, an error is passed onto the pcall and the error function would run. After that function is run, the rest of the code is halted. If i = 1 then the error would be skipped meaning the code is running as it should and the result is true/successful meaning that function would run AND the rest of the code below the script would continue.

# Coroutines

The basics of coroutines is that essentially, they allow multi-threading. However, there are facilities that can be used such as returning values in a coroutine.

Printing the coroutine.resume, prints the yield result.

Printing the coroutine.yield, prints the parameters in the resume

### Example:

Printing the resumeA screen shot of a computer code

Description automatically generated

Printing the yield

A computer screen shot of a code

Description automatically generated

Essentially the first resume is needed as a formality to reach the subsequent yield. The second resume then converts the reached yield into its parameters which instead of having to be printed, can be used directly as parameters to another function.

# Using Self

The use of self is “syntactical sugar” because it’s mainly used for a table/argument referencing itself, as in the name. It’s essentially not necessary to use but the function is there so why not.

## Examples:

Basic UseA computer screen shot of a black background

Description automatically generatedComplex UseA computer screen shot of a black screen

Description automatically generated

# Tables:

### table.find()

Link: <https://create.roblox.com/docs/reference/engine/libraries/table#find>

A screenshot of a computer program

Description automatically generated

### table.unpack()

Link: <https://create.roblox.com/docs/reference/engine/libraries/table#unpack>

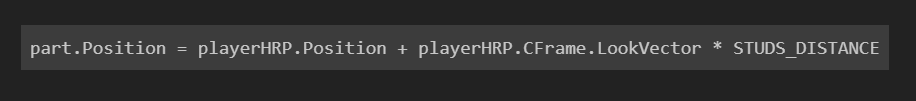
A screenshot of a computer program

Description automatically generated

# CFrames:

### Player look direction with LookVector

Link: <https://devforum.roblox.com/t/how-can-i-get-the-direction-a-player-is-facing/603867>



# Time Passed with os.clock()

You can get the amount of time passed with os.clock(). It gives the number of seconds at the certain point of time you call it. Subtract 2nd time by 1st time to get seconds passed.

## Example:

A screen shot of a computer program

Description automatically generated

Printing this would result in:

A black background with white text

Description automatically generated