

Wait & While Loops - Looping Code

Required Previous Lessons/Knowledge: Strings, Variables, Commenting, Creating Scripts

Lesson Running Time: Approx 15 - 20 minutes

Optional Handouts: Intro to Lua

Learning Objectives and Outcomes: Use the wait function and properly indented code inside a while true do statement to create a part which changes color every 3 seconds.

Create a Looping Part

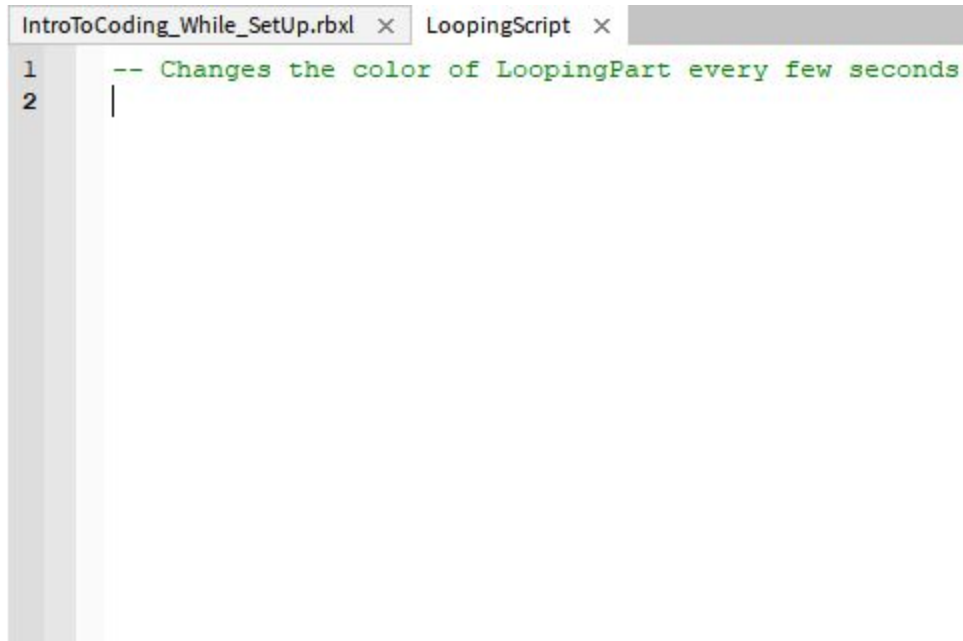
You don't have to be stuck in a world limited to just one color, your parts can be a magical unicorn forest which continue to change colors forever. Don't want a magical unicorn forest? Use the colors of your favorite sports team or superhero!

This next script will loop the code you create over and over so the part will keep changing color forever.



Set up

1. Create a new part, name it something descriptive like *SeahawksPart*, *LoopingPart*, or *RainbowPart*.
2. Create a new script under **ServerScriptService**
3. Name the script *LoopingScript*
4. Delete Hello World
5. Create a comment saying what the code will do in your own words



The screenshot shows the Roblox Studio script editor. At the top, there are two tabs: 'IntroToCoding_While_SetUp.rbxl' and 'LoopingScript'. The 'LoopingScript' tab is active. The script content is as follows:

```
1  -- Changes the color of LoopingPart every few seconds
2  |
```

Create Your Own Variables - Save Yourself Time

Which would you rather type out over and over again as you code all the colors of the rainbow, `game.Workspace.NameOfYourPart.BrickColor` or `loopingPart`? I vote for part.

Remember, variables can be used to store all kinds of things. This includes things you type a lot such as `game.Workspace.NameOfYourPart.BrickColor`

Create a Local Variable

A **local variable** is a variable that only works in the script where it is created. Most of the variables you create will be local variables.

1. At the top of your code, create your local variable by typing `local`, and the local variable's name.
 - Example: `local loopingPart`

pascalCasing

In Roblox lua, variable names use pascalCasing. this means the first letter of a variable is always lowercase. The second word is always capitalized. There should never be spaces in variable names.

2. Tell your variable what information to store, remember that `NameOfYourPart` should be replaced with the actual name of your part.
 - Example: `local loopingPart = game.Workspace.NameOfYourPart.BrickColor`

Now instead of typing in `game.Workspace.NameOfYourPart.BrickColor` all you have to do is type the name of the variable you created; `loopingPart`.

Code Example

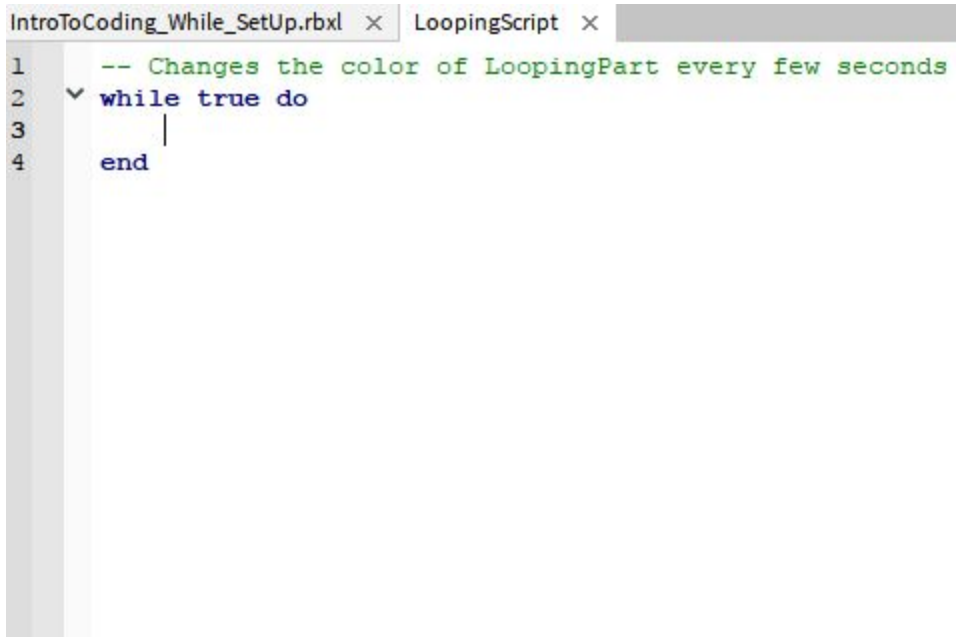
```
-- Changes the color of Part every few seconds  
  
-- Set up a new variable to store  
game.Workspace.NameOfYourPart.BrickColor  
local loopingPart = game.Workspace.PracticePart.BrickColor
```

Looping Code - while true do

While loops will run over and over and over and over unless something tells them to stop. There are different types of loops, you're going to use `while true do`. All you have to is sandwich the portion of code you want within a while true do loop and your magical part will be powered forever.

Create the while true do Loop

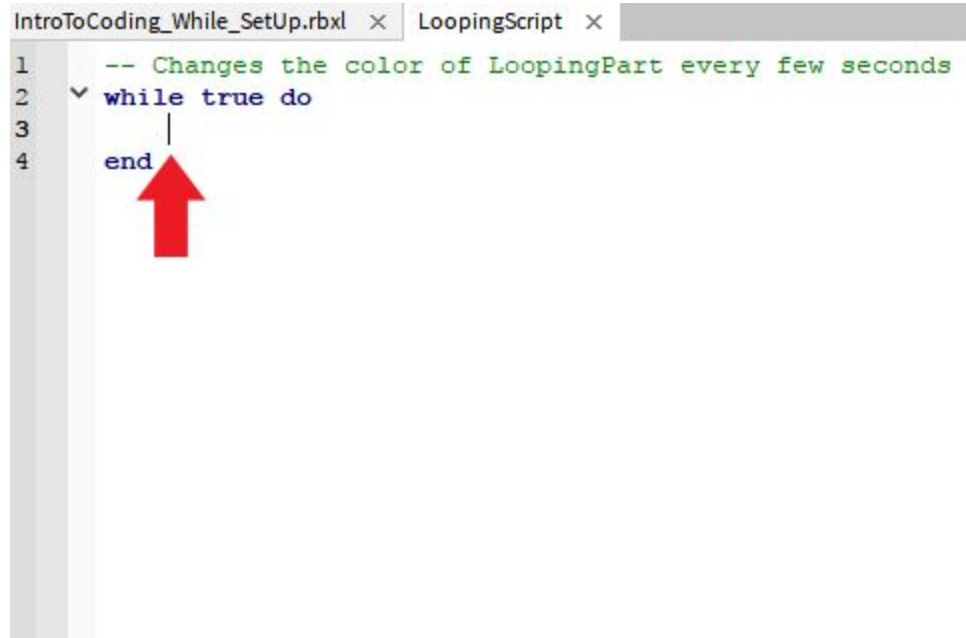
1. At the top of the code, write a new comment saying the code will loop forever
2. On the next line type `while true do`
3. Press **Enter** on the keyboard. The word `end` should autocomplete for you



```
IntroToCoding_While_SetUp.rbxl x LoopingScript x
1  -- Changes the color of LoopingPart every few seconds
2  while true do
3      |
4  end
```

Indenting Code

You might have noticed that not only did the editor auto-add `end` to your script, it also indented the next line of your code. Indenting the code within the while loop makes it easier to read



Add Code to the Loop

Everything between `while true do` and `end` will loop forever. Or until you have a power outage. Whichever. To create an ever-changing rainbow part, add code within the `while true do` loop that will change the color of your part.

- Between `while true do` and `end`, type the code which will change the brick to the first color you want it to turn.
 - Type: `loopingPart.BrickColor = BrickColor.new("First color name")`

Code Example

```
-- Changes the color of Part every few seconds

-- Variables
local loopingPart = game.Workspace.PracticePart

-- Looping Code
while true do
    -- Type code to loop here
    loopingPart.BrickColor = BrickColor.new("Light reddish violet")
```

```
end
```

Scope

In order for code to run in the loop, it has to be between the `while true do` and before `end`. Having your code in the right place so that it runs correctly is referred to as **scope**.

Think of it as a code sandwich. The top of the sandwich is `while true do`, the bottom is `end`. Your code is the delicious peanut butter and jelly. You have to keep all the tasty peanut butter and jelly between the two pieces of bread if you want to eat it.

```
-- Changes the color of Part every few seconds

-- Variables
local loopingPart = game.Workspace.NameOfYourPart.BrickColor

-- Looping Code
while true do
    -- This is within the scope of the while true do loop
    loopingPart.BrickColor = BrickColor.new("Color name")
end

-- This is outside of the loop
```

wait()

The code as it is will instantly change the brick to a new color. To make the script wait before running the next line of code, use the `wait()` function. **Functions** are just pre-programmed chunks of code. Rather than having to type all that code out again, people create functions that can be called on by name to run that code for them. Hello World used a `print()` function.

Don't Crash Your Computer

All while loops need to have a `wait()` somewhere within the loop to prevent the code looping so fast your computer crashes.

Code a wait() Function

To have the program wait three seconds before going on to the next color:

These documents are licensed by Roblox Corporation under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](#). Roblox, Powering Imagination, and Robux are trademarks of Roblox Corporation, registered in the United States and other countries.

- On the line after the first BrickColor changing code, type `wait(3)`

Code Example

```
-- Changes the color of part every few seconds

-- Variables
local loopingPart = game.Workspace.NameOfYourPart.BrickColor

-- Looping Code
while true do

    loopingPart.BrickColor = BrickColor.new("Color name")
    --Will have the program wait 3 seconds before moving on
    wait(3)

end
```

Number Type Variables

`wait()` takes a number variable type instead of a string variable. **Number variable types** are used to track amounts of things rather than checking for a specific combination of characters such as a name. Number variables *do not* use quotation marks.

Adding More Colors

1. Choose an amount of time between one and ten seconds you would like the brick to stay the color you chose.
2. Add a `wait()` after the code which changes the BrickColor property.
3. Add a new line of code with a new color.
4. Test your code.
5. Continue adding colors and wait statements until you have as many colors as you want.

Finished Code Example

```
-- Changes the color of part every few seconds

-- Variables
local loopingPart = game.Workspace.NameOfYourPart.BrickColor

-- Looping Code
while true do

    loopingPart.BrickColor = BrickColor.new("First color name")
    wait(3)
    loopingPart.BrickColor = BrickColor.new("Second color name")

end
```

```
wait(3)
loopingPart.BrickColor = BrickColor.new("Third color name")
wait(3)

end
```

Troubleshoot Your Code

- **Problem: Part turns grey**
 - Check that all strings used match the name as it is shown in the color picker
- **Problem: Colors are skipped**
 - Place a wait() after each color statement, including the last statement.
- **Common troubleshooting tips**
 - Don't use quotation marks for number values
 - Triple-check that NameOfYourPart has been replaced with the exact name of your part
 - Check that capitalization matches
 - Colors chosen must be from the color picker