

Mission 3: Remix

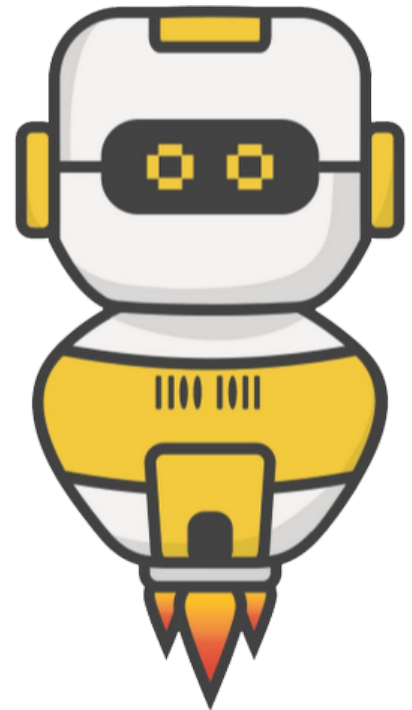
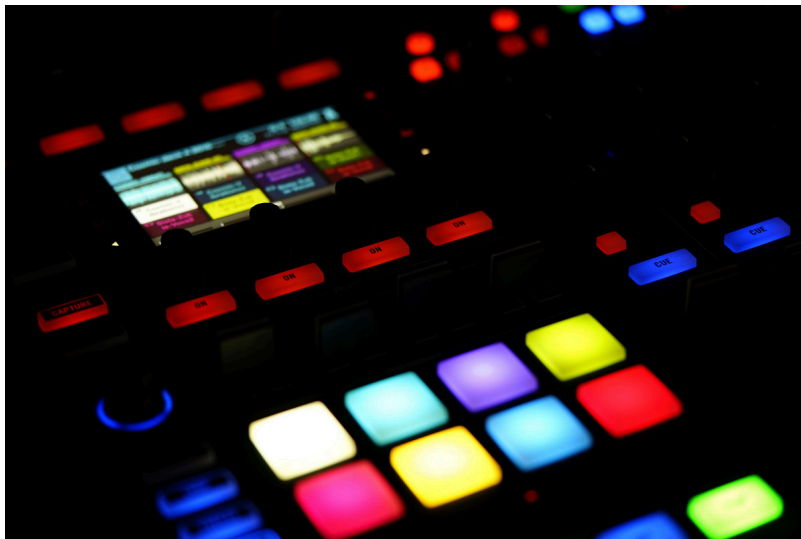
Student Workbook





Hello again!

This assignment will let you be creative and come up with your own program for the CodeX to run.



Go to the Mission 3 Remix Log and fill out the Pre-Remix preparation.



Have you heard of music remixes?

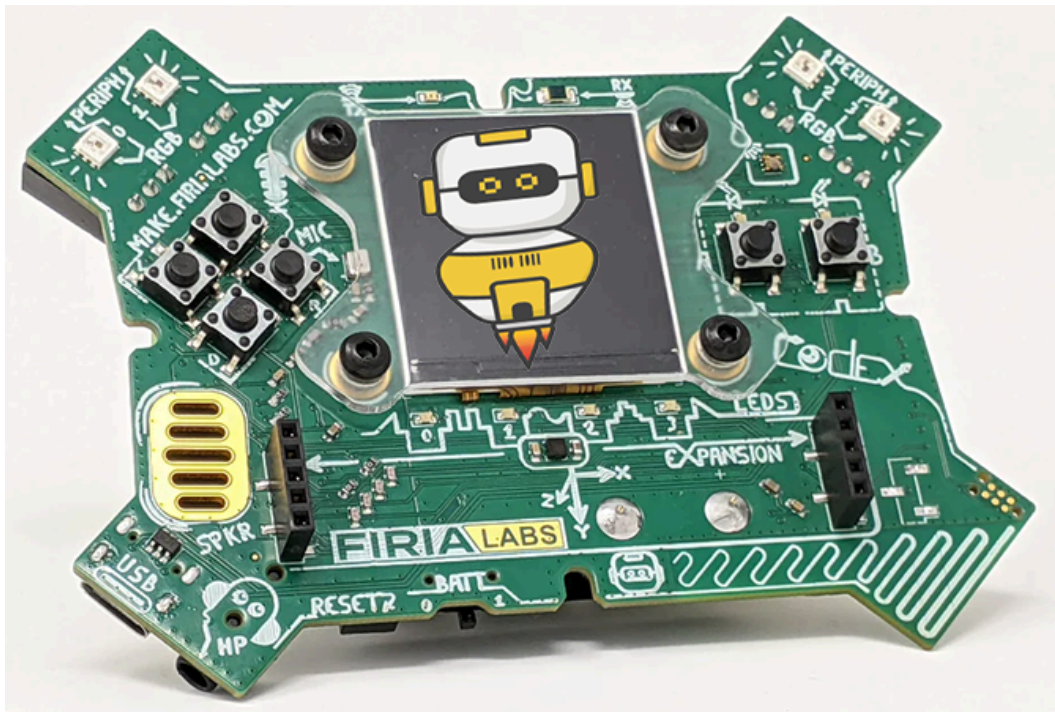
It is when someone takes parts of a song, or several songs, and combines them with some original music to create something original

- You can combine two or more songs as part of a **remix**
- Something new is created by combining parts of songs and adding new music with the parts

Project Remixes

You can do the same thing with your mission projects!

- A new program is created by adding new code and using parts of code from programs you already created
- You can combine parts of two or more programs in a **remix**



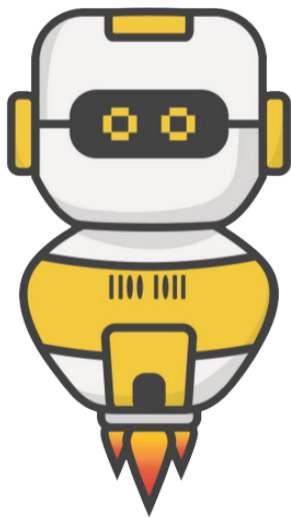
Creating a remix of your projects will let you:

- Use the skills and practice the concepts from the missions
- Be creative
- Improve your understanding of programming
- Collaborate with other students
- Design an original program and write the code all on your own

Step #1: Review projects and concepts

You might want to get started right away ... but don't! To avoid stress and frustration, do some planning first.

- Review your code from Mission 2
- Review your code from Mission 3
- Use the Mission 3 Remix Log to help you record your review



DO THIS:

- Open your project from Mission 2 - Heart1
- Review what the program does
- Review the programming concepts and skills you learned
- Fill out the information in the remix log
- Open your project from Mission 3 - Pixels1
- Review the programming concepts and skills you learned
- Fill out the information in the remix log

```
Heart1 x Pixels1 x
1 from codex import *
2 display.show(pics.MUSIC)
3
```

```
Heart1 x Pixels1 x
1 from codex import *
2 from time import sleep
3
4 delay = 1
5
6 color = RED
7 pixels.set(0, color)
8 pixels.set(1, color)
9 pixels.set(2, color)
10 pixels.set(3, color)
11 sleep(delay)
12
13 color = YELLOW
14 pixels.set(0, color)
15 pixels.set(1, color)
16 pixels.set(2, color)
17 pixels.set(3, color)
18 sleep(delay)
19
```

Step #2: Brainstorm ideas

For this first remix, you will use RGB to display different colors.

- Read the information on the next pages
- Learn about RGB (red, green, blue)
- See how to use RGB to display any color
- Then use RGB in your remix

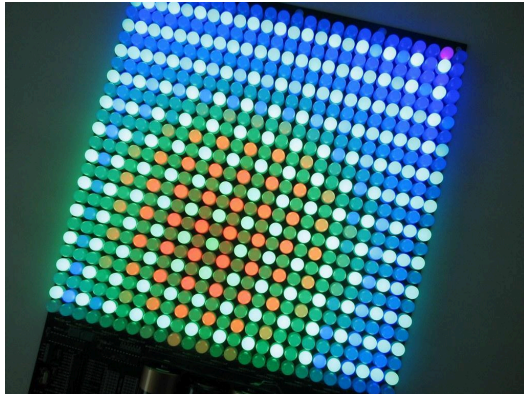
What is RGB?

- RGB stands for Red, Green, Blue.
- Find out more by watching the short video using the link.




[Watch this video](#), starting at 0:45 and ending at 2:35

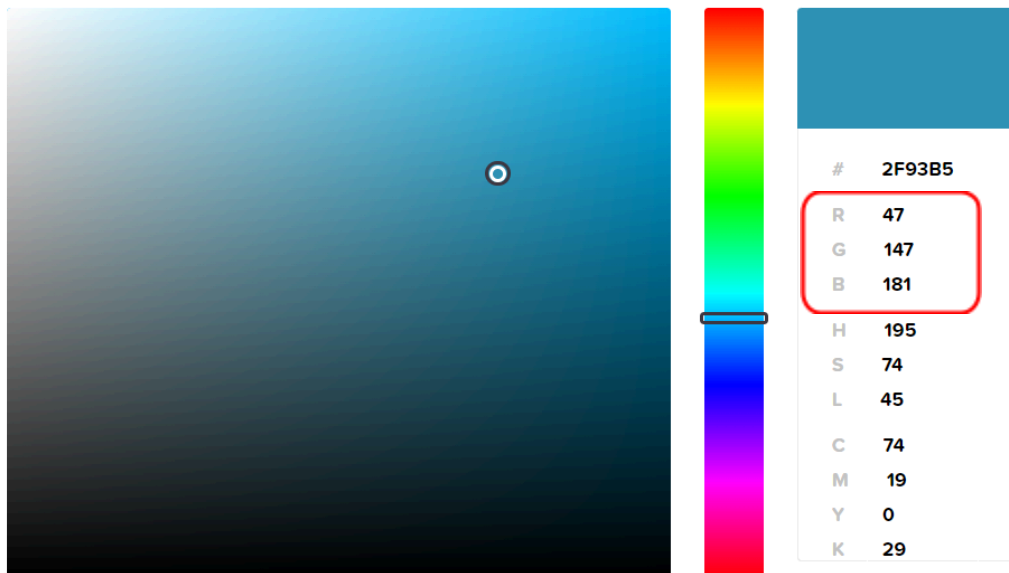
Getting RGB values



The video mentions “triplets” of numbers. Each number in the triplet represents a value in RGB.

 = (47, 147, 181)

- The first number is the amount of red
- The second number is the amount of green
- The third number is the amount of blue



You can use online software to select a color and find the RGB colors. Online color picker: <https://htmlcolorcodes.com/color-picker/>

In the example above, the RGB is (47, 147, 181)

- In Python, the triplet is called a “tuple”

Using RGB values

Set your own colors by changing the color value to a tuple instead of a built-in color:

```
color = (47, 147, 181)
```

```
delay = 1
color = (47, 147, 181)
pixels.set(0, color)
pixels.set(1, color)
pixels.set(2, color)
pixels.set(3, color)
```

You can also assign each pixel their own color by using the tuple in the pixels.set() function

```
sleep(delay)
pixels.set(0, (219, 31, 58))
pixels.set(1, (236, 213, 80))
pixels.set(2, (15, 42, 163))
pixels.set(3, (231, 61, 238))
```


OPTIONAL:

Challenge -- Random RGB values

If you want to try something new, generate random numbers for R, G and B and see what color happens!

Everytime you run the code, or add the code multiple times, you should get a different color.

```
from codex import *
from time import sleep
from random import randrange

delay = 1
red = randrange(256)
green = randrange(256)
blue = randrange(256)
color = (red, green, blue)
pixels.set(0, color)
pixels.set(1, color)
pixels.set(2, color)
pixels.set(3, color)
sleep(delay)
```

Step #2: Brainstorm ideas

Read through remix suggestions on the next pages

- Use a suggestion for your remix, OR
- Use your creativity to come up with your own idea for a project
- How can you use RGB colors in your remix project?
- Decide with your partner what project you will do



Mild Remix

Select an RGB color for all four pixels **and** display an image. After a short sleep(delay), select a different RGB color and image. Repeat as many times as you want. Refer to “Using RGB values” for help.



[Video of Mild Remix](#)



Medium Remix

Light all pixels with an RGB color. Then turn off the pixels and display an image. Then clear the screen and light the pixels with another RGB color. Repeat as many times as you want.



[Video of Medium Remix](#)

Hint: To clear the pixels, use `display.clear()` and to clear the screen use `pixels.set(#, BLACK)`



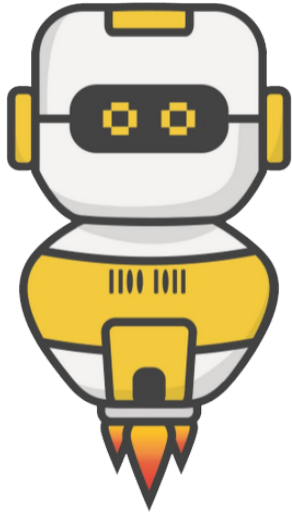
Spicy Remix

Use one of the ideas from Mild or Medium, but instead of using an RGB color that you chose, use random colors. Use the Page 8 Challenge for help.



[Video of Spicy \(Mild\) Remix](#)

[Video of Spicy \(Medium\) Remix](#)



DO THIS:

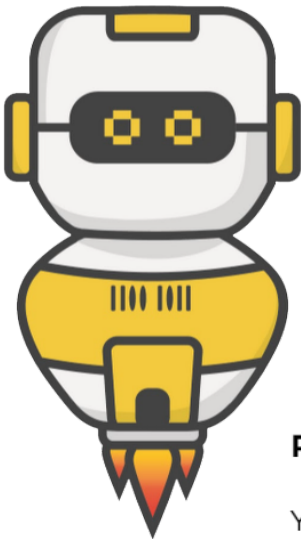
- Decide with your partner what project you will do
- Fill out the information in the Mission 3 Remix Log for **Step #2**

Remix Step 2: Describe what your remix project will do:

Step #3: Make a plan

Now that you have an idea for your remix, you need a plan.

- What variables will you need?
- What colors will you use?
- What images will you display?



DO THIS:

- Fill out the information in the Mission 3 Remix Log for **Step #3**


Remix Step 3: Plan your code. What variables will you use in the project?

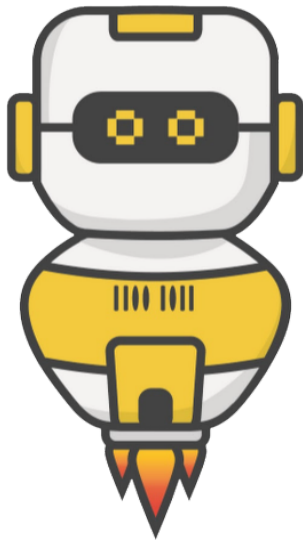
You do not need to fill out every line if you don't need that many variables.

| Variable Name | What it will be used for |
|---------------|--------------------------|
| | |
| | |

| Pixel colors to use: | Images to display: |
|----------------------|--------------------|
| | |

Step #4: Code your project

- **IMPORTANT:** In CodeSpace, go to the sandbox:  It is above the toolbox in the lower left corner.
- You can leave **Heart1** and **Pixels1** open (Use them as a guide)



DO THIS:

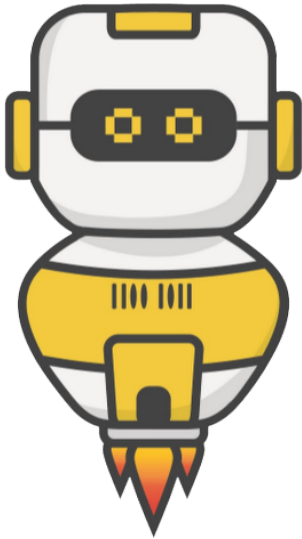
- Start with a new file and give it a descriptive name (**Remix3**)
- Import your modules
- Define your variables
- Write your code, testing frequently

Reminders!

- Don't try to write all the code at one time
- Mistakes happen, so find them early
- Type just a few lines of code and then run the program
- If there is an error, fix it before continuing
- Use the debugger and your other programs for help

Step #5: Documentation

You should always make your code readable and easy to follow.

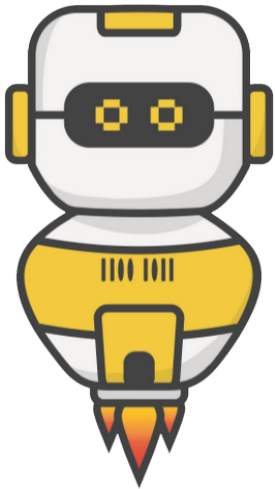


DO THIS:

- Add blank lines where needed to divide sections of code
- Add a comment at the top with your name and the name of the program
- Add a few more comments to sections of your code that explain what they do

Step #5: Get feedback

Getting feedback and reflecting on your code can help you make the program even better.



DO THIS:

- Show your code to another student
- Have him/her fill out the feedback form on your Mission 3 Remix Log
- Give yourself some feedback
- Is there something you want to change or improve or add?
- Fill out the feedback form on your Mission 3 Remix Log

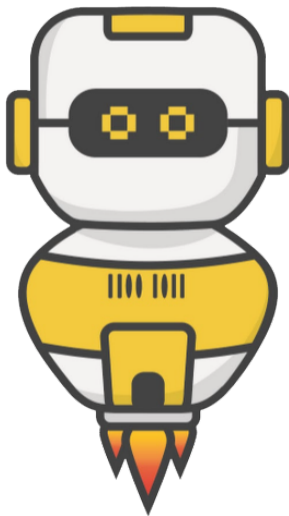
Modify your code to make your project even better

Congratulations!

Now you have your own remix!
Great job! Share your project with
your friends.

By completing this remix you have:

- learned more about programming
- used skills and concepts from the missions
- been thinking!
- and problem solving
- and much more!



DO THIS:

- Complete the Mission 3 Remix Log
- Don't forget to clear your CodeX by running your **Clear** program