Make Your Own Arduboy Game: Part 3 - Storing Data & Loops

crait (Holmes) #1 June 21, 2019, 6:45am

This is Part 3 in a series on learning how to program your own Arduboy game. If you have skipped the previous parts, please read over **Part 1** and **Part 2**!

Variables

Computers work a lot with calculations and data. To make most video games, you're going to need to be able to store data, like high scores, or player location, or lives remaining. In order to remember data, a computer must set aside some memory to put that data into. Then, the computer can be told to change the data stored there.

Loops

© Guy & Rodd/Distributed by Universal Uclick via CartoonStock.com
guy & rodd
CARTOONSTOCK .com
Search ID: gra061120
FOUR DAYS LATER, IT OCCURRED TO GEOFF THAT HE MAY HAVE BEEN TAKING THE "LATHER, RINSE, REPEAT" INSTRUCTIONS TOO LITERALLY.
11-20

Remember how I said that computers have to be given specific instructions? Ever notice how the back of shampoo bottles say to **1. Lather** , **2. Rinse** , **3. Repeat** ? If a computer were given those instructions, they would be stuck in an infinite loop of lathering, rinsing, and repeating. In programming, having instructions repeat in a loop can be very useful. I'll show you how to use the **loop()** function to do this.

Getting Started

In this program, we’re going to make the Arduboy keep track of a number and display it to you as it is increased.

Let’s do it! Grab the code from the previous tutorial so that we can build off of it:

```
//Jonathan Holmes (crait)
//October 18th, 2016
//Printing Text

#include <Arduboy2.h>
Arduboy2 arduboy;

void setup() {
  // put your setup code here, to run once:
  arduboy.begin();
  arduboy.clear();
  arduboy.print("Holmes is cool!");
  arduboy.display();
}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Initialization

Whenever you create a variable, you have to **initialize** it, which is setting aside memory for the data and giving it a name. You’ve already done this but didn’t realize it. Check out the `Arduboy arduboy;` line. You initialized an object called `arduboy` . Objects are a lil’ more complex than I want to get into during this tutorial, but there are different kinds of variables that you can initialize. Here are some: Integers, booleans, characters, objects, doubles, and many more.

We’ll initialize our number that’s going to increase as an integer. This basically means that it’s a whole number, without fractions. Integers will appear as `int` inside of C++ code.

To initialize a variable, you must put 3 things: The type of variable, the name of the variable, then a semi-colon. Let’s call our variable `counter` . Here’s the code: `int counter;` Put it under the `Arduboy arduboy;` line.

Assignment

Whenever you create a variable, you can give it a value. This is called **assignment** . We don’t know what `counter` 's value is because we never gave it one.

Let’s clean up the code by removing the `arduboy.print();` and `arduboy.display()` functions from **setup()** . Instead, let’s put the assignment there:

```
counter = 0;
```

This line of code is saying that `counter` 's value is now equal to 0. Instead of 0, you could put another number, a mathematical formula, or some other things that I’ll explain below.

Here’s what your code should look like:

```
//Jonathan Holmes (crait)
//October 18th, 2016
//Counter

#include <Arduboy2.h>
Arduboy2 arduboy;

int counter;

void setup() {
  // put your setup code here, to run once:
  arduboy.begin();
  arduboy.clear();
  counter = 0;
}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Incrementing

Okay, our program will be repeating a few simple instructions over and over again. Basically, we’ll change the value of `counter` , then display the value of `counter` , then repeat. To do this, we should add some code into the **loop()** function.

```
counter = counter + 1;
```

This line of code means that you are assigning the value of `counter` to itself plus 1. Or, in other words, `counter` 's value is now equal to the value of `counter` + 1.

Displaying The Variable’s Value

Now that we have the variable increasing, let’s display it! Remember how we used `arduboy.print()` to print some text to the screen? Well, we can use that same function to display numbers to the screen, too. Add `arduboy.print(counter);` .

If you were to run this code as it is, now, then the Arduboy’s screen would fill up with numbers. If we’re going to print something new, we need to be sure to erase what was previously on the screen. We need to add in `arduboy.clear();` at the beginning of **loop()** and `arduboy.display();` at the end to display the updated screen.

Have you ever used a type writer? Whenever you type letters, the cursor moves over. The `arduboy.print()` function works similarly. Every time you use the `arduboy.print()` function, it moves the **cursor** over. So we need to reset the cursor to the top of the screen with `arduboy.setCursor(0, 0);` . I will explain this more in a later tutorial, but just throw that at the top of the **loop()** .

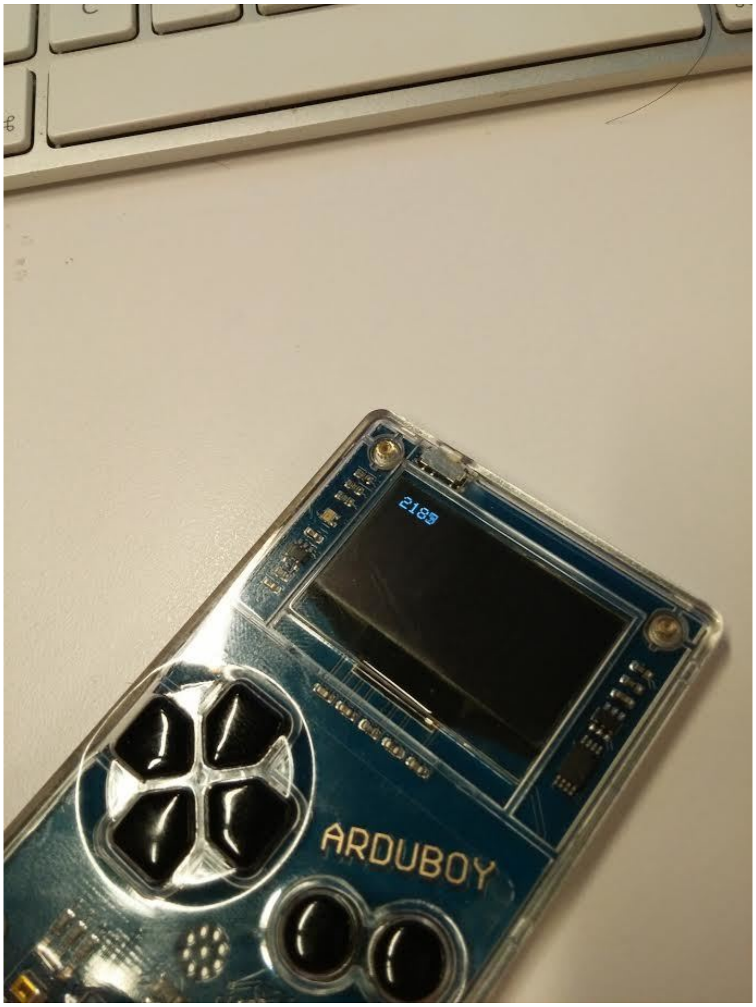
The Completed Code

I’ve added in some comments, but your code should look like the following:

```
//Initialize our counter variable
int counter;
//The setup() function runs once when you turn your Arduboy on
void setup() {
  //Start the Arduboy properly and display the Arduboy logo
  arduboy.begin();
  //Get rid of the Arduboy logo and clear the screen
  arduboy.clear();
  //Assign our counter variable to be equal to 0
  counter = 0;
}
//The loop() function repeats forever after setup() is done
void loop() {
  //Clear whatever is printed on the screen
  arduboy.clear();
  //Move the cursor back to the top-left of the screen
  arduboy.setCursor(0, 0);
  //Increase counter's value by 1
  counter = counter + 1;
  //Print out the value of counter
  arduboy.print(counter);
  //Refresh the screen to show whatever's printed to it
  arduboy.display();
}
```

Running The Code

Connect your Arduboy to your computer and upload this code to it. Your Arduboy should start counting up!



What’s Next?

In the next tutorial, we’ll be learning about testing the values of variables, booleans, and pressing buttons! **Click here to go to Part 4!**

Credits

I wrote this tutorial in order to give back to the programming community that taught me to get into it about 10 years ago. If you’d like to follow me on Twitter, please do so at **http://www.twitter.com/crait** . I’d greatly appreciate that. 😄

8 Likes

Make Your Own Arduboy Game: Part 6 - Graphics!

Make Your Own Arduboy Game: Part 5 - Your First Game!

Make Your Own Arduboy Game: Part 7 - Make Pong From Scratch!

Make Your Own Arduboy Game: Part 9 - Mapping DinoSmasher

Make Your Own Arduboy Game: Part 2 - Printing Text