Make Your Own Arduboy Game: Part 2 - Printing Text

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

This is Part 2 in a series on learning how to program your own Arduboy game. If you have yet to install the Arduino IDE and Arduboy Library, **please go back to Part 1**.

What Is Programming?

Because of how hacking and programming are portrayed in movies and stuff, many people have no idea how programming actually works. Basically, programs are a big list of instructions for the computer to follow. Computers do exactly what they are told. They have no opinions nor intuition, so computers can't just decide or guess what to do next. You have to be very specific.

Different programming languages have different ways that they're written and different kinds of instructions. These tutorials will go over C++ basics and you'll learn about those kinds of instructions.

Creating A New Sketch

Using the **Arduino IDE**, go to **File** > **New** . This will create a new window with a new **sketch** . A *sketch* is a file with your code in it. Sometimes you'll hear it called " *source code* ".

Reviewing Our Code

In that new window, you should see the following code:

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

Whenever you write code, you're going to put a lot of it inside of braces, which look like { and } . That is to split up different lines of code that occur at different times. You'll learn more specifics about what those different times are later on. All you need to worry about is that you have some code that will be put into the **setup()** section and the **loop()** section. These are called **functions** , but we don't need to know much about them right now. All we need to know is that whatever code that's inside of the **setup()** function will be run once whenever the Arduboy turns on and whatever code is inside the **loop()** function will be run later on.

Comments

The next thing I want to bring to your attention are the **comments** . These are lines of code that are completely ignored. They're great for adding in notes so that you can read your code easier. I'm going to use them a lot, so I thought we should start with them.

It’s very typical to use comments to explain the purpose of functions, variables, and at the beginning of your code to explain the purpose of your program/game.

Let’s go ahead and add some comments to the top of the program! At the top, add a few new lines of text that starts with two slashes, then your name, date, and the title of this program.

Your code should now look like this:

```
//Jonathan Holmes (crait)
//October 18th, 2016
//Printing Text
void setup() {
  // put your setup code here, to run once:
}
void loop() {
  // put your main code here, to run repeatedly:
}
```

Those lines of text should turn grey.

Getting Arduboy-Specific Instructions

Okay, that was simple enough, but we need to start adding in our instructions, now. To add in a lot of Arduboy-specific instructions, we need to import the Arduboy Library into our code. Make a new line of code after your comments and add `#include <Arduboy2.h>` . This line of code says that you want to grab a lot of instructions from the *Arduboy2.h* file. There’s nothing more about that that you need to know.

If you recall, I mentioned in Part 1 that there is an old library called ‘Arduboy’ which was superseded by a new library called ‘Arduboy2’. This tutorial will concentrate on the newest library only.

Make a new line after that, and add `Arduboy2 arduboy;` . Don’t worry about this too much. Like I said, this is part of getting Arduboy-Specific instructions.

Our code should look like this:

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

#include <Arduboy2.h>
Arduboy2 arduboy;

void setup() {
  // put your setup code here, to run once:

}
```

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

}
```

The setup() Function

Sweet! I’m glad you got this far. Add the following lines of code inside of the braces of the **setup()** function:

```
  arduboy.begin();
  arduboy.clear();
  arduboy.print("Hello");
  arduboy.display();
```

Notice that each line of code ends in a semi-colon **;** . This is because all instructions in C++ need to end in one. Otherwise, the computer will think it’s something else that I’ll explain later.

Next, notice that all of these lines of code include `arduboy.` at the beginning. This is because our instructions are stored inside of the **arduboy** library.

Last thing I want you to look at is the parenthesis, **(** and **)** . All functions have these. Some include things inside of them and some do not.

The `arduboy.begin();` function pretty much tells the Arduboy to turn on properly. Once that is done, the next instruction is followed.

`arduboy.clear();` tells the Arduboy to erase everything on the screen. Technically, it is not needed here as the `arduboy.begin()` function actually clears the screen already.

`arduboy.print("Hello");` tells the Arduboy to write some text to the screen. Notice that we actually have something inside of the **(** and **)** this time. To tell the Arduboy *what* we want to print, we need to it inside of the parenthesis. Any time you’re working with text, you have to put quotation marks around the text, so we gotta do that, too. You can actually put whatever you want inside of those quotations. Maybe play around with it and put your own name into it. 😃

Last, we have to tell the Arduboy to actually refresh the screen and show you what you just printed to it. That’s what the `arduboy.display();` does.

You’re done!

That’s it! You’ve programmed your first program for the Arduboy! This is what the finished code should look like:

```
//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:

}
```

Installing

Like we did in **the previous tutorial**, connect your Arduboy with a USB data cable, select the correct port and board, then click the → button to transfer it to your Arduboy!