User Defined Functions

The ‘void setup()’ and ‘void loop()’ functions are predestined to be in every Arduino sketch. They do not accept any variables, and they do not have an output. They simply make the Arduino code understandable by the IDE, but functions can do so much more! If you write your own functions, they can do things and make the code more readable overall. Writing your own functions can also make it so you only need to write the code for repetitive tasks once. Also, functions are used extensively with Object Oriented Programming and Arduino Libraries, as you will see in future lessons.

Cautionary advice going forward:

You have advanced to the next-level assignments. From this point on, you will need to pay extra-close attention to the details of the code examples to get everything done correctly.

Example Sketch 1: Naming and Calling a Function

Below is an example of a very simple implementation of a custom function called ‘void myFunction()’ that does very little. Mostly, this is just to show the basics of how to make a simple function and call (implement) it into a sketch’s ‘void setup()’ or ‘void loop()’ sections.

```
void setup() {  // the setup function runs once
  Serial.begin(9600); // open serial communication
}
void loop() {   // the setup function runs repeatedly
  myFunction(); // call the function to run here
  delay(500);   // wait a half second
}
void myFunction (){ // function only runs when called
  Serial.println("Function Called!");
  delay(500);   // wait a half second
}
```

Explain the Serial Monitor behavior of the above sketch with written sentences:

Describe the output of this program on a serial monitor. (what text and time intervals?)

Answer here

How is this output achieved by the program? (Explain the logic of the program)

Answer Here

Example Sketch 2 : Function Inputs

A custom function can be very useful, and they can respond to inputs, and change their behavior based on what is input. In fact, you have been using these all along. ‘Serial.println()’ ‘Serial.begin(9600)’ and ‘delay(2000)’are functions [of a library] that take inputs and change their behavior in response. The only tricky piece to using inputs, is that the variable needs to have the input called with the same (or smaller) *type* of variable that is declared in the function.

```
void setup() {  // the setup function runs once
  Serial.begin(9600);
}
int x = 0;      // create a global variable called 'x'
void loop() {   // the setup function runs repeatedly
  tattle(x);    // call the function with input 'x'
  delay(2000);  // wait 2 seconds
  x = x + 1;    // increase the value of x by 1
}
void tattle (int L){ // function only runs when called
  Serial.print(L);   //serial print the input
  Serial.println(" was input to custom function.");
}
```

Explain the Serial Monitor behavior of the above sketch with written sentences:

Describe the output of this program on a serial monitor. (what text and time intervals?)

Answer here

How is this output achieved by the program? (Explain the logic of the program)

Answer Here

Example Sketch 3 : Function Returns

A custom function can be very useful, and they can respond to inputs, change their behavior based on what is input, and return things back to the program after being called. The 'void' of 'void setup()' and 'void loop()' both mean that those functions return nothing [a void value]. But if a function will actually return a value, its type needs to be declared as the function is named. See below for a few examples. The variables in the function do not all need to be of that type, but the function's type is for the 'return' value, and is returned to the place the function is called.

```
void setup() {  // the setup function runs once
  Serial.begin(9600);
}
int x = 2;      // create a global variable called 'x'
int y = 3;      // create a global variable called 'y'
void loop() {   // the setup function runs repeatedly
  Serial.print(add(x,y));
  Serial.print(" is the sum");
  Serial.print(multiply(x,y));
  Serial.print(" is the product");
  int f = add(x,y);
  Serial.print(multiply(f,y));
  Serial.print(" is the new product");
  while(true){
  }
}
int add (int a, int b){ // function returns type 'int'
  int c = a + b;
  return c;
}
long multiply (int a, float b){ // function is type 'long'
  long c = a * b;
  return c;
}
```

Explain the Serial Monitor behavior of the above sketch with written sentences:

Describe the output of this program on a serial monitor. (what text and time intervals?)

Answer here

How is this output achieved by the program? (Explain the logic of the program)

Answer Here

Arduino’s Own Explanations

Function Reference Page << a written explanation of Function Declaration on Arduino
The one link above has many examples on the page. A close reading will help you...

Create your own Functions

You don't know if you understand it, until you can create it from nothing...
Using what you have seen above, in the example sketches and from Arduino’s reference materials. Create your own sketch that implements a custom Function in some way. You have plenty of creative license in this goal, but you must make a sketch that successfully uses a custom Function. A screenshot of your sketch and output is half of your response to these. A written explanation of the logic is also required to prove that you understand what you are doing with this work. Primary tasks are required for all students. Secondary tasks are required to get a top grade on the assignment.

- Primary tasks:
    - Successfully implement a custom function

Screenshot & Explanation here

    - Successfully implement a custom function that is called more than once

Screenshot & Explanation here

    - Successfully implement a custom function that takes an input

Screenshot & Explanation here

    - Successfully implement a custom function that returns a value

Screenshot & Explanation here

- Moderate understanding secondary tasks:
    - Implement a custom function that also calls another custom function

Screenshot & Explanation here

    - Implement a custom function that takes more than one input

Screenshot & Explanation here

- Advanced understanding secondary tasks:
    - Implement a few custom functions, so you can overload a function

Screenshot & Explanation here