Arduino Handout

Vocabulary and Syntax:

Common Functions

void ___ () {code}	This is for setting up a function that doesn't return and values.
setup	A special function name that is only ran once that is useful for initializing variables.
loop	Another special function name that repeats indefinitely running all the code contained every time.
digitalRead(#)	Reads the value from a specified digital pin, either HIGH or LOW.
digitalWrite(#,LOW/HIGH)	This sends 0/5 volts to the specified digital pin.
analogRead(#)	This reads data in from a specified pin giving a value between 0 and 1023.
analogWrite(#, value)	This sends 0/5 volts to the specified analog pin.
pinMode(#,INPUT/OUTPUT, INPUT_PULLUP)	This initializes a specified pin for input or output.
Serial.begin(baud rate)	This initializes the serial port and sets the baud rate to communicate with other electronics - typical baud rate 9600.
Serial.print(....)	This prints a statement to the serial port all on one line.
Serial.println(....)	This prints a statement to the serial port with a line ending at the end of the statement.
Serial.available()	This returns the number of characters that have been received from the serial port but not used yet.
Serial.parseInt()	This parses the unused serial characters into usable ints.
random(# -or- min,max)	This generates a random number between 0 and #-1 or between min and max not including max.
map(var,oldmin,oldmax,newmin,newmax)	This scales the var from an old range to a new range.
constrain(var,min,max)	This limits the var to the range provided.
millis()	This function calls the internal time of the processor.
delay(*num*)	The argument inside of delay() is a time value with units of milliseconds.

Conditional Statements

for(var dec; condition; advancement){code}	This loops through a set code until the condition is met doing the advancement every time the loop begins.
if(condition){code_1} else{code_2}	code_1: This does the code if the condition is met. code_2: This can be used after an if statement and this code will
while(condition){code}	This repeats the code while a condition is true.
switch (var) case label_1: //statement break; case label_2: //statement break;	Specify different code that should be executed in various conditions. Compares the value of a variable to the values specified in case statements. When a case statement is found whose value matches that of the variable, the code in that case statement is run.

Libraries

Libraries are recognized as such files ending in (.h) or (.cpp). These files can be inserted into the main Arduino sketch file by simply using:

```
#include<math.h>
```

or

```
#include”math.h”
```

The libraries should be placed at the top of the Arduino Sketch file.

Using Variables

Variables are place holders for short descriptive names of sensors. They can also be used for holding a value(s) of a calculation. Each variable must start with a **Data Type**, which specifies the type of data that shall be used for the variable.

Examples of variables:

int	name_of_variable;	used only for integers
float	name_of_var;	used for numbers with very long decimals
String	name_of_String;	used for declaring a set of characters such as a sentence.

As you can see, there are many different Data Types to use. Here is a list of the types of Data Types that your variables should have.

Arduino Data Types	Value Assigned	Value Ranges
boolean	8 Bit	True or False
byte	8 Bit	0 to 255
char	8 Bit	-127 to 128
unsigned char	8 Bit	0 to 255
word	16 Bit	0 to 65535
unsigned int	16 Bit	0 to 65535
int	16 Bit	-32768 to 32767
long	32 Bit	-2,147,483,648 to 2,147,483,647
float	32 Bit	-3.4028235E38 to 3.4028235E38

Variable Assignments

A variable can also be assigned a particular value or set of characters. This is useful if you plan to use this variable multiple times with the same static value/character(s).

Examples of variables:

int	red_led = 13;	Assignment of a red led to be used on pin 13
float	pi_value = 3.141592653;	used for numbers with very long decimals
String	my_name = “JGarcia”;	used for declaring a set of characters such as a name

Comments

Writing a comment in the code is useful to remember what the piece of code does. It’s good to comment your code that you write so that others can also understand what you have written. You can start a comment by using two forward slash: // or by starting a paragraph comment using

```
// This is a comment and the compiler ignores this section of the code.
```

or

```
/*
This is a comment and the compiler ignores this section of the code.
 */
```