Arduino review

Before we do more coding today we're going to review some key points about Processing, the language we're coding in now.

Let's look at this sample program:

```
//Hays

void setup() {
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}

void loop() {
  Serial.println("Let's blink the light!");
  for (int i = 10; i < 150; i += 20)
  {
    digitalWrite(13, HIGH);
    delay(i);
    digitalWrite(13, LOW);
    delay(i);
    Serial.println(i);
  }
  delay(3000);
}
```

(continued on next page)

First: remember, all Arduino programs have at least the following two functions: setup() and loop():

```
//Hays

void setup() {
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}

void loop() {
  Serial.println("Let's blink the light!");
  for (int i = 10; i < 150; i += 20)
  {
    digitalWrite(13, HIGH);
    delay(i);
    digitalWrite(13, LOW);
    delay(i);
    Serial.println(i);
  }
  delay(3000);
}
```

Sometimes you put code above the void setup() line, but usually only when you are creating a variable that you're going to use throughout the program.

(continued on next page)

Block of code are contained within curly brackets or braces. Every open brace comes with a close brace:

```
//Hays

void setup() {
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}

void loop() {
  Serial.println("Let's blink the light!");
  for (int i = 10; i < 150; i += 20)
  {
    digitalWrite(13, HIGH);
    delay(i);
    digitalWrite(13, LOW);
    delay(i);
    Serial.println(i);
  }
  delay(3000);
}
```

(Continued on next page)

Next, let's remember that if you want to print something to the Serial Monitor you have to have the following line in your setup area:

```
void setup() {
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}
```

The things you put into the void setup() area are things like Serial.begin(9600) or pinMode(13,OUTPUT).

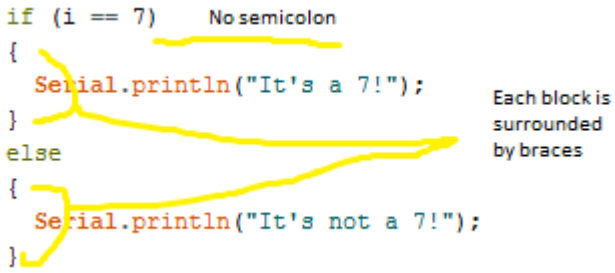
(Continued on next page)

Remember that when you want to do an if statement, there are some key parts to keep track of:

```
if (i == 7)
{
  Serial.println("It's a 7!");
}
else
{
  Serial.println("It's not a 7!");
}
```

The first line of an if statement does NOT have a semicolon.

Each block of code is surrounded by braces.



If you're checking to see if something is equal to something, use double equals (==) like we did in Python. If you want to assign a value to a variable use one equals sign.

You do not have to have an else in your if statement if you don't need it:

```
if (i == 7)
{
  Serial.println("It's a 7!");
}
```

(Continued on next page)

Let's say part 1 of a project was to blink the light with a for loop. If part 2 is to then print a line about tigers, and we want part 2 to happen AFTER part 1, where would we add it?

```
//Hays
6
void setup() {
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}
5
void loop() {  4
  Serial.println("Let's blink the light!");
  for (int i = 10; i < 150; i += 20)
  {  3
    digitalWrite(13, HIGH);
    delay(i);
    digitalWrite(13, LOW);
    delay(i);
    Serial.println(i);
  }  2
  delay(3000);
}
1
```

(Answer on next page)

Answer: We'd click where the 2 is, type Enter, and then add our new part above the delay(3000); line.

For loops

Let's dissect a for loop. A for loop has two main parts: the declaration line and then a block of code that runs underneath it:

```
for(int i = 0; i<20; i++)      //declaration line
{
  Serial.println(i);        //block of code, contained within { braces }
}
```

The declaration line has three parts:

First, we set up the variable: we often use i as the for loop variable, but you can use any variable. We say "int i = 0" if we want the variable to start at 0. If we wanted it to start at 255, we'd say "int i = 255". Then we put a semicolon

```
for(int i = 0;
```

Next, we have the condition part. The for loop will run as long as this statement is true. So in my example above, I said i<20, which means the loop will run as long as i is less than 20. It starts out as 0, so we know the loop will run at least once. If we were going to go down from a big number to a small number, we'd might reverse things and say

```
for(int i = 20; i>0
```

This also would be true at the start; if i is 20 to begin with, then it would be greater than zero, so your loop would work. If you said for(int i = 0; i>20 your loop would do nothing because i starts at 0 and it is not ever greater than 20, so the whole for loop would be skipped.

Now comes the change part. The last part changes the for loop variable so that the loop will end someday. Often we say i++, which increases the variable by one each time. We can also use i+=2 or i+=10 to increase i by 2 or 10 each time. You can use i-- to reduce i by 1 each time, or i-=10 to reduce i by 10 each time. Here are all of the parts together:

```
for(int i = 0; i<20; i++)
```

Finally, you need a pair of braces with code inside of them. If you don't have code inside braces then the for loop won't work.

```
{
  Serial.println(i);
}
```

Also let's remember that we don't put a semicolon on the end of a for loop line:

```
for (int i = 10; i < 150; i += 20)
```

(Continued on next page)

Printing

I said it Thursday, and I'll say it again: if you want to print more than one thing, you have to use more than one print call.

Let's say you're trying to print a line with a variable i and then a statement about i like this:

```
8 is even
```

You CANNOT say

```
Serial.println(i "is even);
or Serial.println(i+"is even);
or Serial.println(I,"is even);
```

That is because you cannot print more than one thing in a print line. To what we want, you would do this:

```
Serial.print(i);
Serial.println(" is even");
```

Notice that I did a print without the ln for the first line, but then used an ln on the second line. That's so that the number and the text print on the same line.