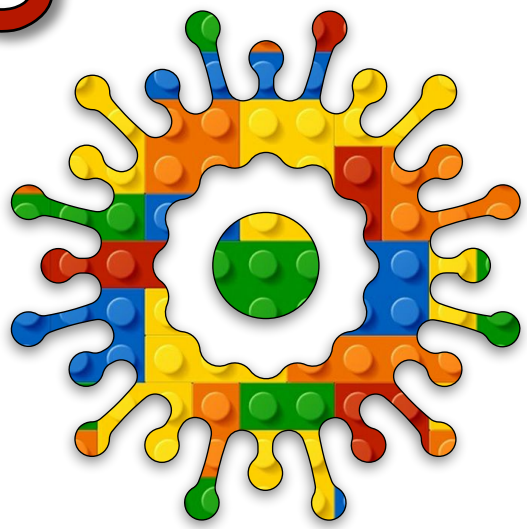
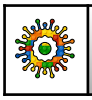
Internet of Things

Module A

Unit #5

boolean blinking





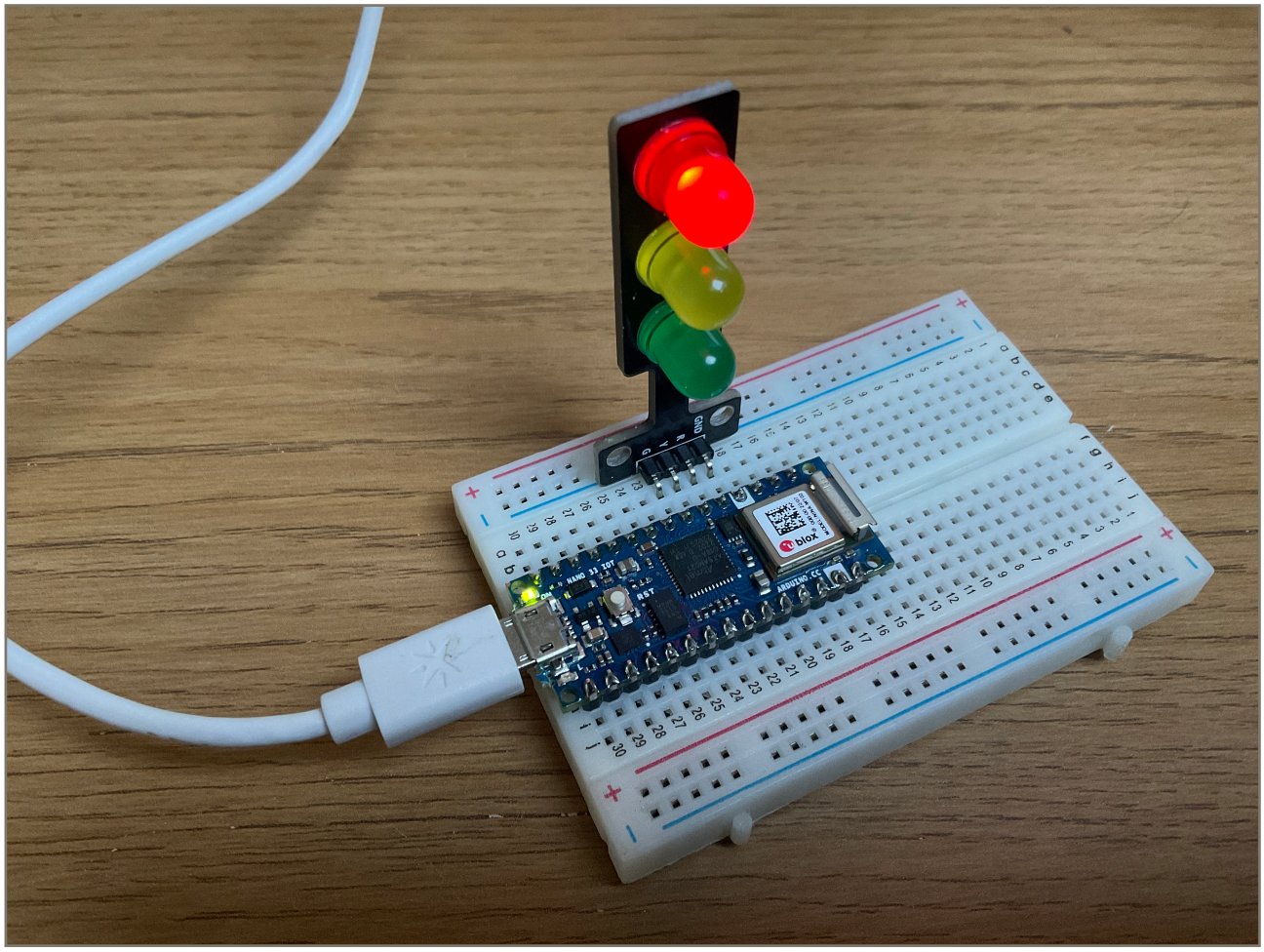
Contents

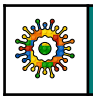
Module A Unit #5 boolean

Introduction to boolean

Boolean is a form of logic which can be described as gates; the most common are AND gates and OR gates. There are more of them, and they form the basics of computing. They are also used widely in higher-level programming. Two symbols are used: `&&` for AND and `||` for OR (called pipes). Trying to explain it makes it seem quite confusing, and yet it is literally the simplest logic. The best way is to use it, and you will see how simple and powerful it is. The `if()` statement can be similar to the `while()` loop: if something is true or a condition is met, then do something. To put it into words, if (`x < 100 && y > 50`) means if `x` is less than `100` and `y` is greater than `50`.

Keep the same setup





Sketch A5.1 full brightness

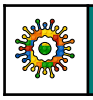
With the LED, we can fade the LED using the `analogWrite()` function. Not all pins support this; those that do have this symbol ~ next to them. This indicates a pulse width modulation (PWM) for that pin. It means we can assign a value to the pin, so if we give pin D2 a value of 255, that means fully bright (100%). If we assign a value of 0 (0%), then it is completely off. Any numbers in between those two values will give a brightness between the two. The following sketch will demonstrate this.

```
void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  analogWrite(2, 255);
}
```

🛠️ Code Explanation

`analogWrite(2, 255);`	This writes a value (255 in this instance) to pin 2. Although it is a digital pin it is treating it as an analog output.



Sketch A5.2 zero brightness

Now give it the minimum value 0. This should be completely off.

```
void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  analogWrite(2, 0);
}
```

🛠️ Code Explanation

`analogWrite(2, 0);`	This writes a value (0 in this instance) to pin D2. Although it is a digital pin it is treating it as an analog output.

Sketch A5.3 something in between

This is somewhere in the lower end of the two extremes, 25. It is a bit hard to see the difference, so we will do an incremental fade-in in the following sketches.

```
void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  analogWrite(2, 25);
}
```

Sketch A5.4 a slight delay

The delay is small, and it is the amount of time (in milliseconds) that it shines for that intensity of brightness.

```
void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  analogWrite(2, 25);
  delay(10);
}
```

Notes

This makes no odds at the moment.

Sketch A5.5 gradually brighter

Instead of a fixed value, we want to increment it by 1 every 10 milliseconds.

```
void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  for (int i = 0; i < 256; i++)
  {
    analogWrite(2, i);
    delay(10);
  }
}
```

🗒️ Notes

You should see it start off dim and gradually getting brighter until maximum brightness, and then restart all over again.

🛠️ Code Explanation

`analogWrite(2, i);`	This increments to brightness from 0 through to 255 incrementing the value of i by 1 each iteration of the loop

Sketch A5.6 and then gradually fade

Now to fade back down again. Notice the `i--` at the end of the decreasing fade. What you should have now is the LED getting rapidly brighter and then quickly fading and then brighter repeatedly.

```
void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  for (int i = 0; i < 256; i++)
  {
    analogWrite(2, i);
    delay(5);
  }
  for (int i = 255; i > 0; i--)
  {
    analogWrite(2, i);
    delay(5);
  }
}
```

Notes

There is quite a bit of unnecessary repetition. Can you think of a way to combine the two? Don't worry if you can't, just have a think about it.

Sketch A5.7 new sketch

❗ Start a new sketch with the basics below.
Another way to do this is to use a boolean operator with an if() statement. So start with a basic sketch putting the LED on maximum brightness (255).

```
void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  analogWrite(2, 255);
}
```

Notes

LED full brightness.

Sketch A5.8 brightness

Now adding a variable for brightness. Nothing too dramatic here.

```
int brightness = 255;

void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  analogWrite(2, brightness);
}
```

Notes

You should have full brightness again.

Boolean Operators

This is all to do with logic, AND, OR, NOT.

🛠 Code Explanation

!	A logical NOT, i.e. a condition where something is not true can be written != (not equal to)
&&	a logical AND, i.e. a condition where two things have to be true
||	A logical OR, i.e. a condition where either one of two conditions is true

Sketch A5.9 the value of fade

We want to fade it up and down. So the initial brightness we will change to `0` so that when it reaches `255` the brightness decreases and when it is `0` it increases. We are using a boolean operator that is called pipes `||` which means one `OR` the other has to be true.

The `fadeValue` is an integer variable that is how much it will fade by in steps of `5` in this case.

```
int brightness = 0;
int fadeValue = 5;

void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  analogWrite(2, brightness);
  brightness = brightness + fadeValue;
  if (brightness == 0 || brightness == 255)
  {
    fadeValue = -fadeValue;
  }
  delay(10);
}
```

🗒️ Notes

It will pulse because it toggles the value of fadeValue between 5 and -5 depending on whether it is starting from 0 or 255; just follow the logic.

🛠️ Code Explanation

`if (brightness == 0 || brightness == 255)`	If brightness is equal to 0 OR 255 then change fadeValue from positive to negative or vice versa

Sketch A5.10 another newish sketch

New sketch again.
Sometimes it is critical that we don't use the function `delay()` simply because the whole programme (the Arduino) stops, and you might want other things to happen while it stops. There is a way of doing the blink sketch without using the `delay()` function. Starting with a very basic sketch and returning to pin `D2`, the red LED.

```
void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  digitalWrite(2, HIGH);
}
```

Notes

The red LED should shine brightly (HIGH).

Sketch A5.11 the state of the LED

Instead of using HIGH and LOW for on and off, we will give these a variable name so that we can change the state of each. We will call this variable ledState and set it to LOW initially.

```
int ledState = LOW;


void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  digitalWrite(2, ledState);
}
```

Notes

The red LED should be off completely, (LOW).

Challenge

You can check it is working by changing the ledState to HIGH and uploading it again.

Sketch A5.12 during the interval

We want the LED to blink on/off for one-second intervals, so we have a variable called `interval`, and we will make it a constant (ie fixed) value of `1000`. `const` is short for constant and means it can never be changed accidentally.

```
int ledState = LOW;
const int interval = 1000;


void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  digitalWrite(2, ledState);
}
```

Notes

This changes nothing.

Code Explanation

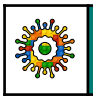
`const int interval = 1000;`	const is short for constant and cannot be changed

Data types

These are the types of data you may well come across and need to use, a brief description is given

🛠️ Code Explanation

`const`	Short for constant eg pin number
`int`	Short for integer eg 3
`float`	Has a decimal point eg 3.76
`long`	Can use much bigger numbers
`unsigned`	Can only be a positive number
`char`	A type of string (text)



Sketch A5.13 every millisecond counts

The Arduino starts counting milliseconds as soon as the sketch starts running. We can use that to check how much time has elapsed since a particular moment in time. So we need to get the number of milliseconds that have elapsed since a particular point in time.

This number is going to get very big very quickly, and also we don't want it to become negative for any reason. So the data type we use is called `long`, and we can make sure it is always positive by defining it as unsigned.

Our variable will be called `currentMillis`.

```
int ledState = LOW;
const int interval = 1000;

void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  unsigned long currentMillis = millis();
  digitalWrite(2, ledState);
}
```

🛠️ Code Explanation

`millis();`	It returns the number of milliseconds since the programme was started, a bit like a running clock

Sketch A5.14 previously

We also need another similar variable to hold the previous number of milliseconds. We will call this `previousMillis`.

```
int ledState = LOW;
const int interval = 1000;
unsigned long previousMillis = 0;

void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  unsigned long currentMillis = millis();
  digitalWrite(2, ledState);
}
```

Notes

Still nothing new to see. We have two variables to compare, `previousMillis` and `currentMillis`. We can do maths with them to work out any delay; it is like a stopwatch with a lap function.

Sketch A5.15 current interval

We need an `if()` statement to check the difference and reset the `previousMillis`.

```
int ledState = LOW;
const int interval = 1000;
unsigned long previousMillis = 0;

void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  unsigned long currentMillis = millis();
  if (currentMillis - previousMillis >= interval)
  {
    previousMillis = currentMillis;
  }
  digitalWrite(2, ledState);
}
```

Notes

Nothing new to see here but we are getting there.

Sketch A5.16 blink without delay

And now to change the state depending on whether it is LOW to become HIGH or HIGH to become LOW. It toggles between them. That is why we need to keep track of the state of the LED; if it is off, then we want it on; if on, then off, simple.

```
int ledState = LOW;
const int interval = 1000;
unsigned long previousMillis = 0;

void setup()
{
  pinMode(2, OUTPUT);
}

void loop()
{
  unsigned long currentMillis = millis();
  if (currentMillis - previousMillis >= interval)
  {
    previousMillis = currentMillis;
    if (ledState == LOW)
    {
      ledState = HIGH;
    }
    else
    {
      ledState = LOW;
    }
  }
  digitalWrite(2, ledState);
}
```

🗒️ Notes

Your LED should blink every second.

🌻 Challenge

Change the `interval` to a much smaller number, e.g. `100`; it should blink much faster.

🛠️ Code Explanation

`if ... else`	Another way of using an if statement where there is more than one condition. In words: if this isn't true then do something else.