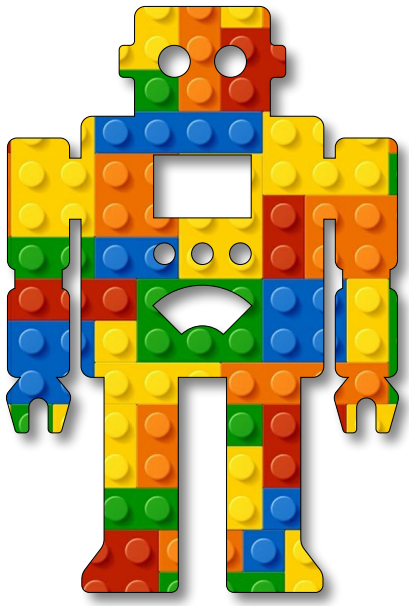
Intelligent Machines
Module B
Unit #5
A button



Module B Unit #5 p5.js button

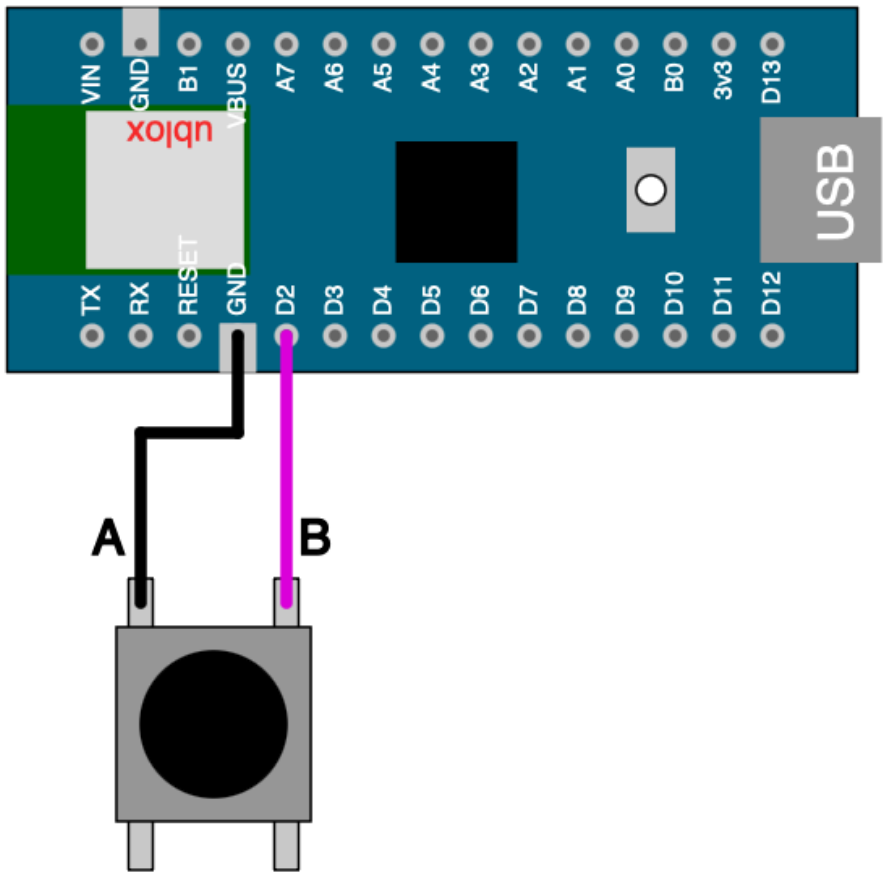
Introduction to p5.js and a button

Introduction to p5.js and a button

As we have covered in a previous unit, a button is connected to pin D2. Here we are going to send data from the Arduino to the sketch. Here we will simply change the colour from black to white by pressing the button.

A reminder of the circuit diagram, on the same side of the button, one pin goes to ground and the other to digital pin D2 (next to it).

Figure 1: Circuit diagram



Sketch B5.1 button

❗ The Arduino sketch

We have added the button to digital pin 2. We called the variable buttonPin and used the pull-up resistor. We read that pin and print it to the serial monitor. We used that to control the colour of the circle.

Arduino sketch

```
const int buttonPin = 2;
int val;

void setup()
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT_PULLUP);
}

void loop()
{
  val = digitalRead(buttonPin);
  Serial.println(val);
}
```

🗒️ Notes

I have highlighted the main changes.

🛠️ Code Explanation

`const int buttonPin = 2;`	Attaching the button to pin 2
`pinMode(buttonPin, INPUT_PULLUP);`	Using the internal resistor
`val = digitalRead(buttonPin);`	Reading the value of the button

Sketch B5.2 the port.js sketch

❗ A few changes to the `port.js` sketch

We have to add a `serialEvent()` function that will catch the data. We push the data to a variable called `circleColour`.

port.js

```
const serial = new p5.WebSerial()
let portButton
let inData
let circleColour

function navigation()
{
  if (!navigator.serial)
  {
    alert ("WebSerial is not supported in this browser. Try Chrome")
  }
  serial.getPorts()
  serial.on("noport", makePortButton)
  serial.on("portavailable", openPort)
  serial.on("requesterror", portError)
  serial.on("data", serialEvent)
  serial.on("close", makePortButton)
}

function makePortButton()
{
  portButton = createButton("choose port")
  portButton.position(10, 10)
  portButton.mousePressed(choosePort)
}

function choosePort()
{
  if (portButton) portButton.show()
  serial.requestPort()
}
```

```
function openPort()
{
  serial.open().then(initiateSerial)
  function initiateSerial()
  {
    console.log("port open")
    if (portButton) portButton.hide()
  }
}

function portError(err)
{
  alert("Serial port error: " + err)
}

function portConnect()
{
  console.log("port connected")
  serial.getPorts()
}

function portDisconnect()
{
  serial.close()
  console.log("port disconnected")
}

function closePort()
{
  serial.close()
}

function serialEvent()
{
  inData = serial.readStringUntil('\n')
  if (inData)
  {
```

```
    circleColour = 255 - inData * 255
  }
}
```

🗒️ Notes

This is necessary because we are sending data to the `p5.js` sketch, whereas before we were sending data from the `p5.js` sketch.

🛠️ Code Explanation

`let circleColour`	A variable to hold the colour of the circle
`serial.on("data", serialEvent)`	Call the serialEvent() function
`inData = serial.readStringUntil('\n')`	Keeps reading until there is a new line /n
`if (inData)`	If it detects any data
`circleColour = 255 - inData * 255`	The inData value will be either 1 or 0

Sketch B5.3 the sketch.js

❗ In `sketch.js`

All we are going to do with the data is fill a circle with the final colour value.

sketch.js

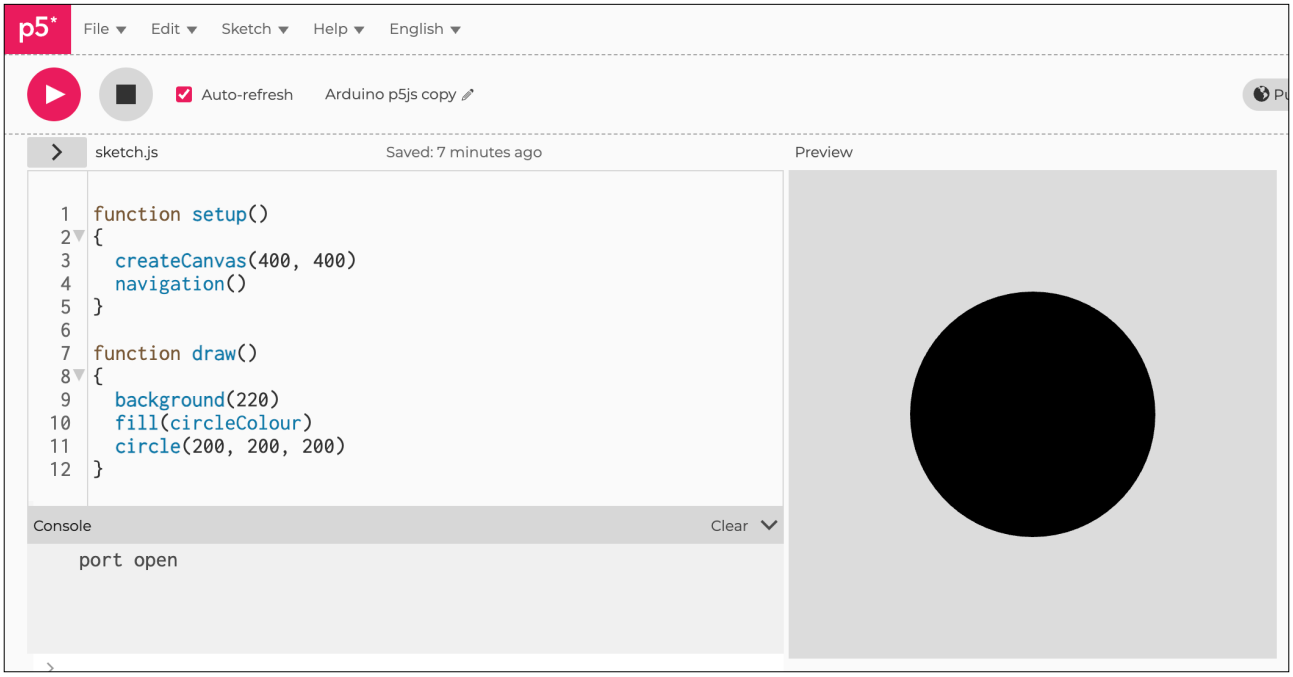
```
function setup()
{
  createCanvas(400, 400)
  navigation()
}

function draw()
{
  background(220)
  fill(circleColour)
  circle(200, 200, 200)
}
```

🗒️ Notes

When you press the button, the circle goes from `black` to `white`.

Figure B5.3


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Arduino p5js copy
sketch.js
Saved: 7 minutes ago
Preview
1 function setup()
2 {
3 createCanvas(400, 400)
4 navigation()
5 }
6
7 function draw()
8 {
9 background(220)
10 fill(circleColour)
11 circle(200, 200, 200)
12 }
Console
Clear
port open