Lesson 2

The Pico Button Activity Worksheet

**The Pico Button Worksheet**

In this activity you will wire the breadboard to the Pico to create a ground rail and a voltage rail. Once this is complete you will go on to wire a button to the Pico and program it to input keypresses and respond by lighting up an LED.

**You will need the following equipment:**

* Raspberry Pi Pico microcontroller
* Micro USB cable
* A breadboard
* A small button
* 4 x M2M jumper wires

Your first task is to fit the Pico into the breadboard as shown in the diagram below. **Note** how the USB port of the Pico is at the top end of the breadboard.

A circuit board with wires and wires

Description automatically generated

In the image above, the button is connected to **pin 14**, and the opposite side of the button is connected to the 3v pin via the power rail on the breadboard. **Remember how the legs that face each other are connected internally**

**Now you’re ready to enter the starter code.**

**From the machine Library import Pin** and from time import sleep

Initialise a variable and set it to a Pin object and set the pin properties values to Pin 14, Pin.IN as an input device and Pin.PULL\_DOWN. The pull down sets the output property to be low so there is no voltage running through the pin.



A white background with black text

Description automatically generated

**Now** create an infinite loop to read the value of the pin**.**

A screenshot of a computer code

Description automatically generated



Now **save and execute** the program. 

If you have **no errors** in your code, well done, **move on to answer the questions below.**

*If you have errors reading the error messages, can you find the line number of the error message? Check your code carefully. Ask a friend to help you spot the errors.*

**What happens when you execute the program?**



**What happens when you press the button?**



**Can you explain the results?**





**Now, let’s output a message only when the button is pressed**

We have seen that because the **Pin** is assigned the value **PULL\_DOWN** it's always off and a button press creates the circuit turning the pin on.

Knowing this means you can now write code to check to see if the pin is receiving a voltage or not and respond accordingly.

To do this you can use a **selection statement**, (if statement) to check if the button has been pressed and output a message in response. Amend your code to the following:

A white background with black text

Description automatically generated



**Save and execute the code**

**Experiment with the button press and the sleep value. This is the debounce** if the value is too large you will miss key presses. If it's too short you will get false readings as the computer executes the code faster than the pin voltage is reset. The sweet spot is usually around 0.1 or 0.2

**Challenge activity 1:**

Can you amend the code to turn on the onboard LED when you press the button and turn it off after a short sleep? To do this you will need to combine the code from the last lesson with what you have learnt this lesson.

**Challenge 2:**

Can you add a second button on Pin 17?

Amend your code so that one button turns on the LED and the second button turns off the LED.

**Plenary:** What is meant by the term debounce?