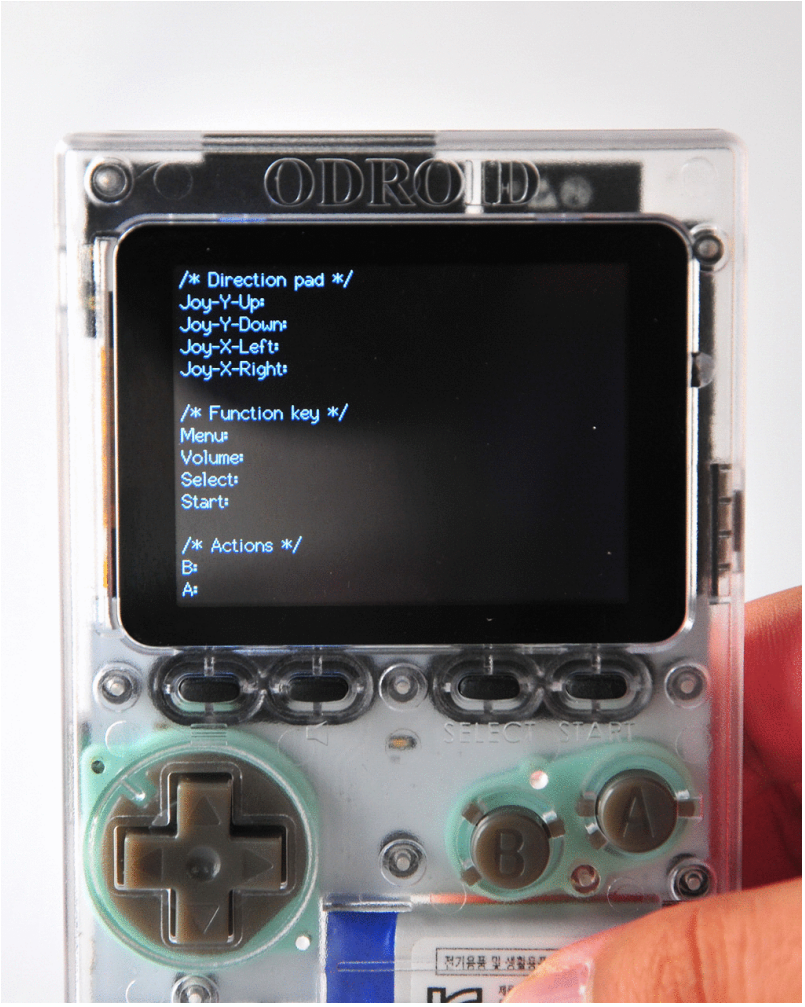
Arduino for ODROID-GO - Buttons

- Make sure that you've followed these guides:
  - Getting started with Arduino
  - Arduino for ODROID-GO - Hello World

- Refer to the Arduino official documents. This provides useful **common functions** with great instructions.
  - https://www.arduino.cc/reference/en/
- Refer to the ESP32 official programming guide. Most of **ESP32 specific functions** are introduced here.
  - https://esp-idf.readthedocs.io/en/v3.0/

We will learn how to read the status of the buttons with Arduino.
A simple output will be shown on the LCD.

Get the status of the buttons



ODROID-GO has 10 buttons available.

- 4 for direction pad.
- 2 for an action.
- and 4 functional buttons.

Thanks to the GO library, we can get the status of the buttons easily.

First of all, initialize the LCD with the **GO.begin()** function.
Set text size to **2** because the default text size is too small to read.

```
#include <odroid_go.h>

void setup() {
  // put your setup code here, to run once:
  GO.begin();
  GO.lcd.setTextSize(2);
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

Add an independent function to display the status, and name it **displayButtons()**.
This function has 2 functions:

- **GO.lcd.clear()** to clear any LCD content previously shown.
- **GO.lcd.setCursor()** to set the start point to print a string. The two parameters (0, 0) means top, left.

```
#include <odroid_go.h>

void setup() {
  // put your setup code here, to run once:
  GO.begin();
  GO.lcd.setTextSize(2);
}

void displayButtons() {
  GO.lcd.clear();
  GO.lcd.setCursor(, );
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

Fill **displayButtons()** out with the code below.

It displays whether any of the buttons are pressed or not. When a button is pressed, then a **“Pressed”** string appears beside the element that corresponds to the button.

All of the buttons ODROID-GO has are available as an instance of the **Button** class.
The **Button** class has some helpful functions to let us know the button's status.
Thus, we use **isAxisPressed()** and **isPressed()** functions to know if a button is currently pressed or not.

If the button is pressed, that functions returns **not 0**.

- **isAxisPressed()** function is only for the direction pad. If the button is pressed, it returns **1 or 2** to distinguish the direction.
- **isPressed()** function is for the other buttons. If the button is pressed, it returns **1**.

Lastly, we have to fill the **loop()** function out to show the status on the LCD.
There are three functions to add:

- **GO.update()** to update the buttons' states so that the **isPressed()** functions works well.
- **displayButtons()** to add result strings to the LCD.
- **delay(1000)** to prevent the LCD from blinking too fast when **GO.lcd.clean()** function acts.

```
#include <odroid_go.h>

void setup() {
  // put your setup code here, to run once:
  GO.begin();
  GO.lcd.setTextSize(2);
}
```

```
void displayButtons() {
  GO.lcd.clear();
  GO.lcd.setCursor(, );

  GO.lcd.println("/* Direction pad */");
  GO.lcd.printf("Joy-Y-Up: %s \n", (GO.JOY_Y.isAxisPressed() == 2) ?
"Pressed" : " ");
  GO.lcd.printf("Joy-Y-Down: %s \n", (GO.JOY_Y.isAxisPressed() == 1) ?
"Pressed" : " ");
  GO.lcd.printf("Joy-X-Left: %s \n", (GO.JOY_X.isAxisPressed() == 2) ?
"Pressed" : " ");
  GO.lcd.printf("Joy-X-Right: %s \n", (GO.JOY_X.isAxisPressed() == 1) ?
"Pressed" : " ");
  GO.lcd.println("");
  GO.lcd.println("/* Function key */");
  GO.lcd.printf("Menu: %s \n", (GO.BtnMenu.isPressed() == 1) ? "Pressed" : "
");
  GO.lcd.printf("Volume: %s \n", (GO.BtnVolume.isPressed() == 1) ? "Pressed"
: " ");
  GO.lcd.printf("Select: %s \n", (GO.BtnSelect.isPressed() == 1) ? "Pressed"
: " ");
  GO.lcd.printf("Start: %s \n", (GO.BtnStart.isPressed() == 1) ? "Pressed" :
" ");
  GO.lcd.println("");
  GO.lcd.println("/* Actions */");
  GO.lcd.printf("B: %s \n", (GO.BtnB.isPressed() == 1) ? "Pressed" : " ");
  GO.lcd.printf("A: %s \n", (GO.BtnA.isPressed() == 1) ? "Pressed" : " ");
}

void loop() {
  // put your main code here, to run repeatedly:
  GO.update();
  displayButtons();
  delay(1000);
}
```

Press **CTRL-U** to compile and upload the sketch.

Then, press any button to show **“Pressed”** string besides that button.
As you noticed, you need to keep pressing it until the LCD is updated due to the **delay(1000)** function.

A completed example

The complete example is available as follows:

Click the **Files → Examples → ODROID-GO → Buttons** menu to import and press **CTRL-U** to compile/upload.