Projects

1. Blink
2. Make Some Noise
3. Servo

Blink

```
#define LED_PIN 13
#define BTN_PIN 2
#define BLINK_TIMES 5
#define BLINK_DELAY 500         // smaller number faster

void setup()
{
    Serial.begin(9600);
    pinMode(BTN_PIN, INPUT_PULLUP);
    pinMode(LED_PIN, OUTPUT);
}

void loop()
{
    int sensorVal = digitalRead(BTN_PIN);
    println(sensorVal);
    if (sensorVal == HIGH)
    {
        digitalWrite(LED_PIN, LOW);
    }
    else
    {
        for (int i = 0; i < BLINK_TIMES; i++)
        {
            digitalWrite(LED_PIN, HIGH);
            delay(BLINK_DELAY);
            digitalWrite(LED_PIN, LOW);
            delay(BLINK_DELAY);
        }
    }
}
```

Explanation

- Code starts be defining respective pins of LED and Button; Number of times to blink and delay between OFF and ON state of LED.
- In setup function, that is called once; upon power on of controller, pins are setup to their respective modes. Pull-Up input for button, and Output for LED. `Serial` is initialised with a baud rate of `9600`.
- In loop function, function that runs continously after setup:
    1. state of button is read and stored in `sensorVal`
    2. `sensorVal` is printed to console.

3. if `sensorVal == HIGH`, meaning button is NOT depressed, LED state is set to low, if not already.
4. since `sensorVal` is mutually exclusive (the switch can only be HIGH or LOW), if `sensorVal == HIGH` is untrue, switch was depressed.
    1. turn ON LED, wait for `BLINK_DELAY`
    2. turn OFF LED, wait for `BLINK_DELAY`
    3. repeat `BLINK_TIMES` times.

Make Some Noise

```
#include "pitches.h"

#define LED_PIN 13
#define BZR_PIN 8
#define BTN_PIN 2

int melody[] = {
  NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4
};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
  4, 8, 8, 4, 4, 4, 4, 4
};

void setup() {
  Serial.begin(9600);
  pinMode(BTN_PIN, INPUT_PULLUP);
  pinMode(LED_PIN, OUTPUT);
  pinMode(BZR_PIN, OUTPUT);
}

void loop() {
  int sensorVal = digitalRead(BTN_PIN);
  Serial.println(sensorVal);

  if (sensorVal == LOW)
  {
    for (int i = 0; i < 8; i++)
    {
      // to calculate the note duration, take one second divided by the
note type.
      //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
      int noteDuration = 1000 / noteDurations[i];
      tone(BZR_PIN, melody[i], noteDuration);

      // to distinguish the notes, set a minimum time between them.
      // the note's duration + 30% seems to work well:
      int pauseBetweenNotes = noteDuration * 1.30;
      delay(pauseBetweenNotes);
      // stop the tone playing:
```

```
      noTone(BZR_PIN);
    }
  }
}
```

Explanation

- Code starts be defining respective pins of LED, Button and Buzzer;
- In setup function, that is called once; upon power on of controller, pins are setup to their respective modes. Pull-Up input for button; Output for Buzzer. `Serial` is initialised with a baud rate of `9600`.
- In loop function, function that runs continously after setup:
    1. state of button is read and stored in `sensorVal`
    2. `sensorVal` is printed to console.
    3. if `sensorVal == LOW`, meaning button is depressed:
        1. repeating 8 times,
        2. calculate the `noteDuration` for current note in array
        3. play this buzzer at this pitch for `noteDuration`
        4. calculate the pause timing between notes, wait for this amount of time
        5. stop playing the buzzer

Servo

```
#include <Servo.h>

#define LED_PIN 13
#define BTN_PIN 2
#define SRV_PIN 9

#define MOVEFAST_DELAY 5
#define MOVEFAST_STEPS 5
#define MOVESLOW_DELAY 10
#define MOVESLOW_STEPS 1

Servo myservo;
int pos = 0;

void setup()
{
  myservo.attach(SRV_PIN);

  // move servo to 20 deg (fast)
  for (; pos < 20; pos += MOVEFAST_STEPS)
  {
    myservo.write(pos);
    delay(MOVEFAST_DELAY);
  }
}

void loop()
```

```
{
  // Move Slowly from 20deg -> 150deg -> 20deg

  // Move from 20 degrees to 150 degrees slowly
  for (; pos < 150; pos += MOVESLOW_STEPS)
  {
    myservo.write(pos);
    delay(MOVESLOW_DELAY);
  }

  // Move from 150 degrees to 20 degrees slowly
  for (; pos > 20; pos -= MOVESLOW_STEPS)
  {
    myservo.write(pos);
    delay(MOVESLOW_DELAY);
  }

  // Move Fast from 20deg -> 150deg -> 20deg
  // Move from 20 degrees to 150 degrees quickly
  for (; pos < 150; pos += MOVEFAST_STEPS)
  {
    myservo.write(pos);
    delay(MOVEFAST_DELAY);
  }

  // Move from 150 degrees to 20 degrees quickly
  for (; pos > 20; pos -= MOVEFAST_STEPS)
  {
    myservo.write(pos);
    delay(MOVEFAST_DELAY);
  }
}
```

Explanation

- Code starts be defining Servo pin;
- In setup function, that is called once; upon power on of controller, `Servo` pin is attached to an instance of `servo` using PWM library.
    1. Move servo from 0deg to 20deg quickly
- In loop function, function that runs continously after setup:
    1. Move Servo from 20deg to 150deg slowly
    2. Move Servo from 150deg to 20deg slowly
    3. Move Servo from 20deg to 150deg quickly
    4. Move Servo from 150deg to 20deg quickly