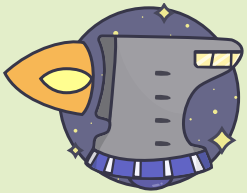
TIS01-X
project::box

The Whimsical Series

Useless Light Meter

Preview

Preview Booklet



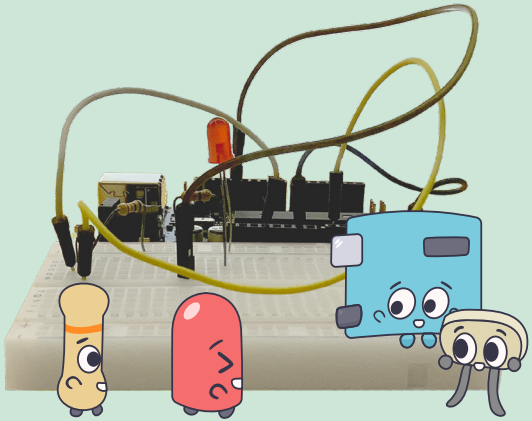
the foundry collective

Useless Light Meter Preview Booklet v1.0, 2021

TIS01-FC

project::box

Recommended For Ages 12 and above



The Whimsical Series

Useless Light Meter

Final Cut

Instruction Booklet

Contents

01

In this section, you will be introduced to this project::box, understand what an Arduino is and the anatomy of the Arduino and Breadboard.

Introduction

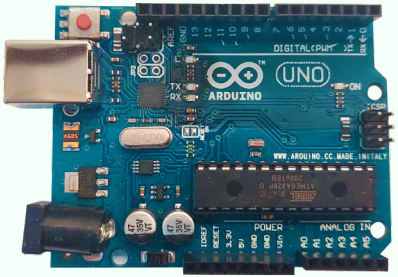
Have you ever wondered how much light there was in your room? Is just lifting your head and looking around too much effort? Well boy is this the device for you. In this project::box, you will build a most wonderful meter that tells you how bright the room is by the glow of a single bulb.

Arduino and Microcontrollers

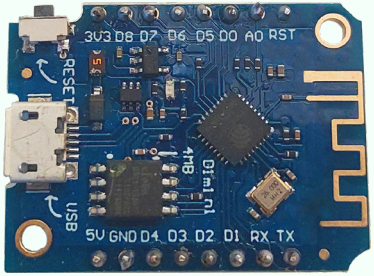
Section 1.1

This kit also happens to be a starter course on Arduino programming, so let us crack on with the basic set up.

Before we continue, let us first get acquainted with how microcontrollers and Arduinos look like.



Arduino Uno

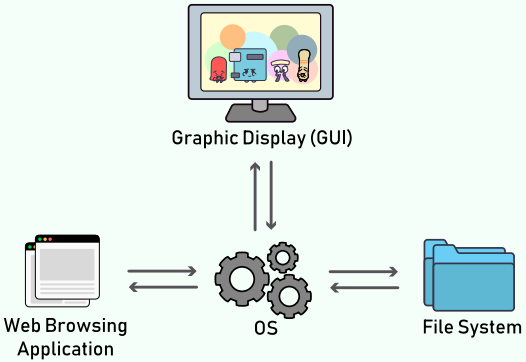


Generic Microcontroller

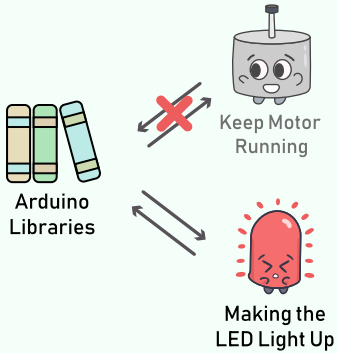
The Arduino is a common brand of microcontrollers. Cheap, powerful and easy to use, it is very popular and is the perfect microcontroller for beginners.

So, what are microcontrollers?

Microcontrollers are cheaper computers that are created for simpler tasks. Just like computers, we can program microcontrollers to do almost anything within their limits. Read sensors? Easy. Control robots? No problem. Just don't use it to control your gardening system while maintaining your bathwater temperature – you will either get a rude shock as boiling water comes raining down on you or a drowned garden full of dead plants. Sadly, just like most of us, the microcontroller can't do multiple things at once. This is due to the absence of the operating system (OS) within the microcontroller.



Computers



Arduino

The OS in computers is what you would communicate with to get your hardware to do things. It is a layer of abstraction on top of the hardware that makes it easier for the user to get the computer to do things. All you need to do is let the OS know that you need it to do something (e.g., opening your web browser on the GUI tells the OS to start running the web browser and display it on the screen). Since the OS handles all the running of these processes, it can also coordinate them and allow them to run together at the same time or multitask. However, Arduino Libraries lets you directly instruct your hardware components to do things without passing through an additional layer. That would mean that Arduinos are missing the multitasking software in the OS needed – only one process can be running at a time. If the Arduino is making the LED light up, it cannot also keep the motor running at the same time.

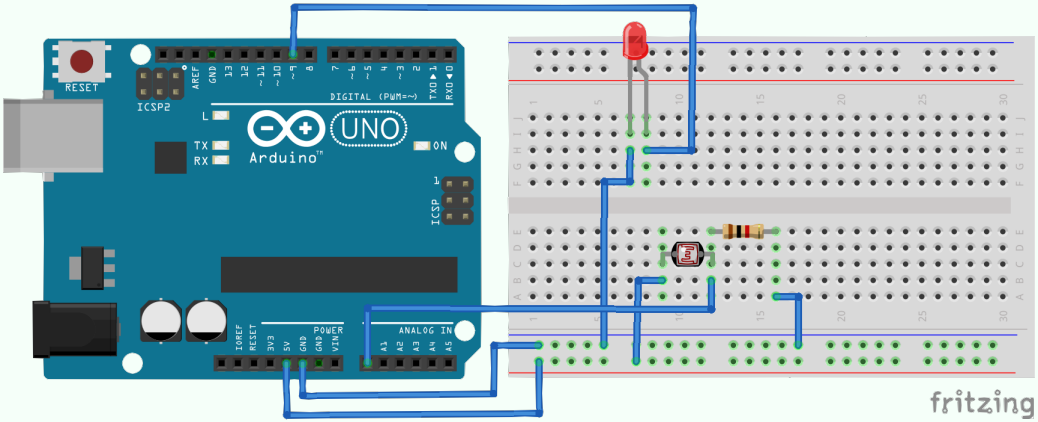
Pages 3 to 33 have been left out

Useless Light Meter

Now that you have done your "Hello World" introduction to the world of Arduino Programming and built a rather useful Light Meter, you no doubt probably have many ideas of the things that you want to build that would improve all our lives – an Autonomous Vacuum Cleaner anyone? Unfortunately, it is about time in this booklet that we kept our promise and teach you what you came here to make in the first place – a Useless Light Meter. To recap, we are building a light meter that will make adjustments to how brightly a LED glows according to how bright or dim the surroundings are.

Wire it Up!

Section 6.1



Change your setup according to the diagram above.

06

In this section, you will finally create the Useless Light Meter, You will also learn how to write analog values and be introduced to Pulse-Wave Modulation (PWM).

Sketch and Run

Section 6.2

```
const int sensorPin = A0;
const int ledPin = 9;
void setup () {
    pinMode(ledPin, OUTPUT);
}

void loop () {
    int val = analogRead(sensorPin);
    analogWrite(ledPin, val);
}
```

Copy the sketch above to your text editor (either manually or directly through your foundry::kits account) and upload it to your Arduino. Watch how the brightness of the LED now reflects the brightness of the room.

Understanding the Sketch

Section 6.3

In this sketch, every run of the `loop` function would first read the analog value from the Light Dependent Resistor on Line 8, and then pass that analog value along to the LED in Line 9 using the built-in function `analogWrite`. While we have explained how `analogRead` works, you might be wondering – since an Arduino can only communicate in binary and cannot just switch its signal on "halfway", how does the Arduino transmit these analog signals and allow `analogWrite` to work?

How Does analogWrite Work – Pulse Wave Modulation

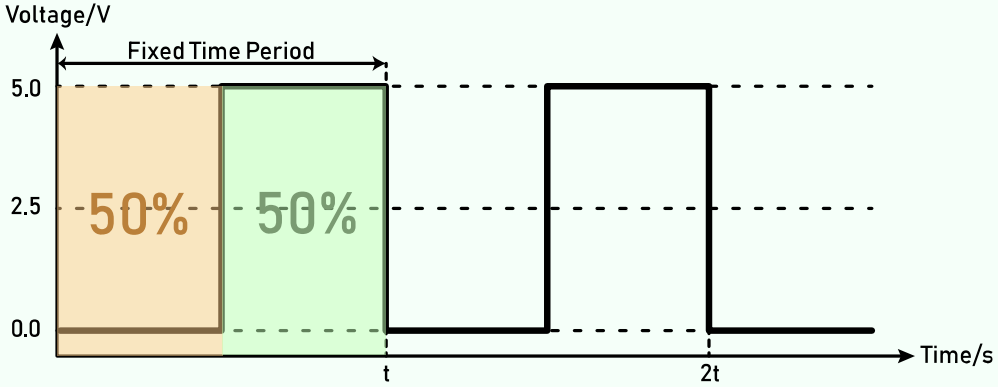
Section 6.4

In order to answer that, let's first look at this modified version of the loop() function of the Hello World (Section 4) sketch.

```
digitalWrite(ledPin, HIGH);
delay(delay_1);
digitalWrite(ledPin, LOW);
delay(delay_2);
```

We have swapped out "`500`" in each `delay()` with the integer variables `delay_1` and `delay_2`. What do you think would happen if both `delay_1` and `delay_2` decreases close to 0? Well, as you lower both `delay_1` and `delay_2` together, you will notice that the LED begins to flicker before dimming when the flicker becomes too fast for your eye to keep up.

Furthermore, if you were to change either `delay_1` and `delay_2`, you can vary the duration the LED is on (by changing `delay_1`) relative to the duration it is off (by changing `delay_2`). The longer the LED is on relative to being off, the brighter it appears. This kind of control is known as *Pulse-Width Modulation* or *PWM*.



In Arduinos, Pulse Wave Modulation is done on voltages since we are dealing with electricity. The graph above, known as a *duty cycle*, shows how we can use PWM to achieve an average output voltage of 2.5V. The electrical signal is alternated such that 50% of the time during a fixed time period the output signal would be at 5V. Since the output is only at 5V only 50% and 0V the other 50% of the time, the overall average output voltage would just be (5V + 0V) / 2 = 2.5 V. If you want to learn more about PWM and its applications, fret not as these will be covered more extensively in future project::boxes.

This kind of control is how the Arduino approximates an analog output. Luckily, we do not have to implement PWM ourselves as the Arduino has a built-in function, `analogWrite`, that handles all the heavy lifting for us. Just like how the `analogRead` function returns an output between 0 and 1023 inclusive, the `analogWrite` function accepts a value from 0 to 1023,inclusive to write to a selected pin. However, not all pins support PWM. The pins that support PWM are marked on the Arduino board with the little ~ symbol.

TIS01-AC

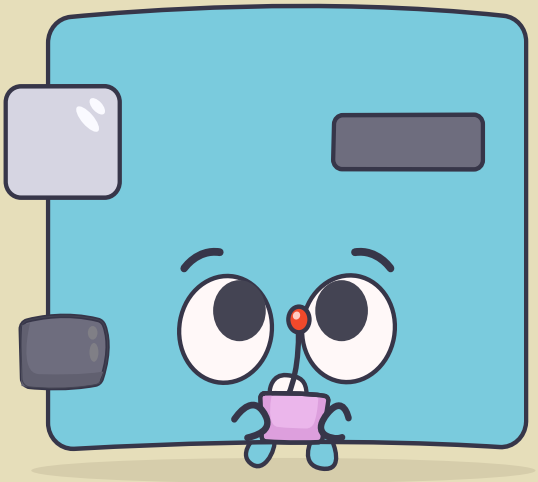
project::box

Recommended For Ages 10 and above

The Whimsical Series

Useless Light Meter

Ardy's Cut



Instruction Booklet

Contents

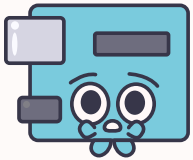
Disclaimer

During the course of this booklet, a thought might come across your mind: "I ordered a Box that should contain instructions and guidance that will teach me how to code and make cool stuff with the Arduino, not a storybook!". Rest assured, we will get to that part in every Project Box. The stories are only meant to make learning fun, interesting and easy. The code, concepts and everything else are still explained in the same fun and enriching whiteboard monologues that you will surely be familiar with.

All names, characters, places, events and businesses appearing in this instruction booklet are fictitious. While our characters' looks and main functions are based loosely on the many electrical components covered here, other aspects such as their names, personalities, actions and events they participate in are purely fictitious. All resemblances to real people, places, events and businesses are purely coincidental. No electrical components – or plants – were harmed in the making of this booklet.

Now that we have that out of the way, let us begin with our most absurd yet educational journey into the world of an Arduino and his friends and create the most wonderfully useless light meter ever.

Key Characters of Micropolis

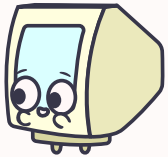


Ardy
Arduino

Larry
LED

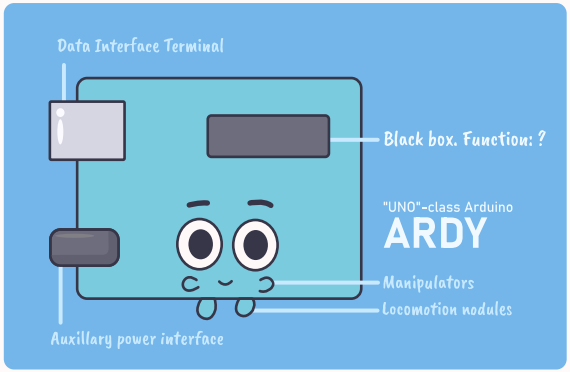
Dale
Light Dependent Resistor (LDR)

Rick
Resistor



Corey
Computer

A Glimpse into the World of an Arduino

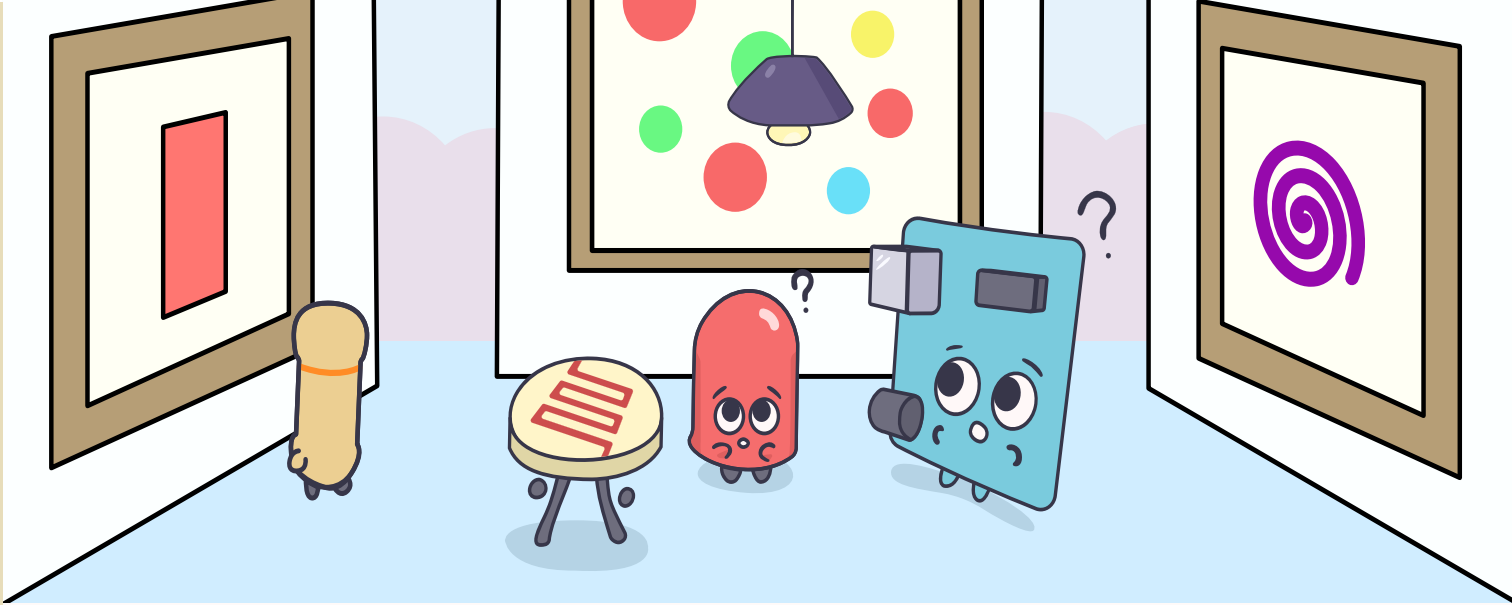


No one really understands black boxes. There's something about that black emptiness that seems to draws us in – surely it hides great wisdom! After all, everyone describes everything they don't understand as black boxes that just work– complicated technology, weird people, the Universe.

Ardy might be one of the few who comes close to understanding the power of black boxes. After all, being an Arduino, he was born with one stuck on his body. If it were red, orange or green, passersby might have thrown him looks of pity as it truly did resemble a strange tumour. But this was a black box. And everyone looked on in awe instead.

Honestly, the black box didn't make Ardy much wiser than others – many townspeople also had the "brain" (or a microchip) in it. It just happened that Ardy's was in a black box. Regardless, Ardy still tried his best to live up to expectations. Philosophers are smart, right? They seemed to spend their whole lives asking questions. Guess the two are connected somehow. Perfect.

In this section, you will understand what an Arduino is and be introduced to this project box.



One weekend, Ardy found himself at an art gallery with his friends in front of the centrepiece of the entire showcase – a piece of modern art bought recently for an absurd amount of money. Larry and Dale seemed drawn by something they described as the “balance of colours”, but all Ardy could see was a few blots of paint. He never really understood modern art anyways – a few streaks of paint were just a few streaks of paint! Not wanting to sound dumb by engaging in that conversation, he frantically scanned his surroundings and finally decided on a “profound” question to ask.

“Have you ever wondered how much light there was in this room?”

Larry blinked once, then twice, and started, “You can just look at the lamp up-”. “Shhh!” If Dale had not stopped Larry, he was definitely going to embarrass himself debating something like that with someone as smart as Ardy. “Well...”

And so the three friends launched into a heated discussion with as much direction as the art around them, and in the process, this project::box was born...

Pages 4 to 30 have been left out

06

In this section, you will finally create the Useless Light Meter, learn how to write analog values and be introduced to Pulse-Wave Modulation (PWM).

Finally, The Answer

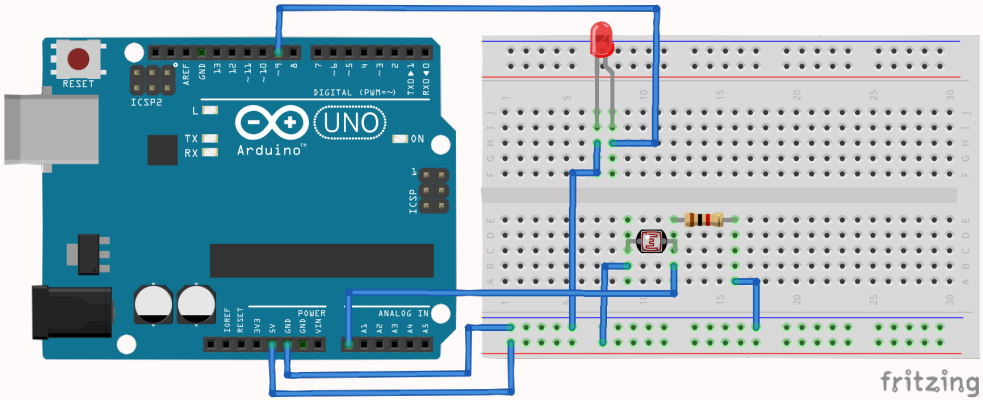
Now that Ardy and his friends (and us) can measure light, they can finally create the answer to the question, “Have you ever wondered how much light there was in this room?”.

Wire it Up!

Section 6.1

After somemore dicussion, they came up with what is now known as the "Useless Light Meter". In Ardy's world, this was more like a noisy crowd than an actual meter. Dale would tell Ardy how bright the surrounding light is through the Light Meter, and Ardy would tell the Larry how brightly he should shine based on the Light Meter. All three would talk over each other, and everyone could tell how bright the room is by looking at Larry – how convenient!

Wire up your Arduino with the actual circuit interpretation of the Useless Light Meter below.



Sketch and Run

Section 6.2

```
const int sensorPin = A0;
const int ledPin = 9;
void setup () {
    pinMode(ledPin, OUTPUT);
}

void loop () {
    int val = analogRead(inputPin);
    analogWrite(ledPin, val);
}
```

Copy the sketch above to your text editor and upload it to your Arduino. Watch how the brightness of the LED now reflects the brightness of the room.

Understanding the Sketch

Section 6.3

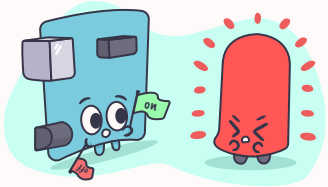
Before we continue, let's first look at this modified version of the loop() function of the First Contact (Section 4) sketch.

```
digitalWrite(ledPin, HIGH);
delay(delay_1);
digitalWrite(ledPin, LOW);
delay(delay_2);
```

We have swapped out "500" in each delay() with the integer variables delay_1 and delay_2. What do you think would happen if both delay_1 and delay_2 decreases close to 0? Well, as you lower both delay_1 and delay_2 together, you will notice that the LED begins to flicker before dimming when the flicker becomes too fast for your eye to keep up.

Furthermore, if you were to change either delay_1 and delay_2, you can vary the duration the LED is on (by changing delay_1) relative to the duration it is off (by changing delay_2). The longer the LED is on relative to being off, the brighter it appears. This kind of control is known as Pulse-Width Modulation or PWM. PWM and its applications will be covered more extensively in future Project Boxes.

This is how the Arduino approximates an analog output. Fortunately, the Arduino has a built-in function that does all the heavy lifting for us: analogWrite, which as you can guess from the name is something like the opposite of the analogRead function; just as the analogRead function returns an output between 0 and 1023, inclusive, the analogWrite function accepts a value from 0 to 1023, inclusive, to write to a selected pin. Not all pins support this, though. Look for the ones marked with the little ~.



From there, it is just a simple matter of taking what we read from analogRead and plugging it into analogWrite to get an LED whose brightness is in direct relation to the level of light outside.