DIGITAL BOOK - I

What is our GOAL for this CLASS?

In this class, we learned how to interface an SD card with the Arduino. Using the interface, we read the data from the files which were present in the SD card.

What did we ACHIEVE in the class TODAY?

- Learned about types of memory in an Aurdino board.
- Created an interface from the Aurdino to an SD card.
- Wrote code to read the data stored in text files in the SD card.
- Displayed the text read.

Which CONCEPTS/ CODING BLOCKS did we cover today?

- Concepts : Types of memory in an Aurino board, pinout of the micro SD card.
- Coding blocks : SD.h library, detect card(), openfile(), read file, display data.
  open(), begin(), available(), close(), read()

How did we DO the activities?

1. Open the wokwi simulator and create a new Arduino project.
2. Drag out the following components in the workspace.
   - **Arduino board X 1.**
   - **Micro SD card X 1.**

   Here, **SD** stands for **“Secured Digital”** card**.**
3. The **Arduino** has **3** types of memory.
   a) **Flash memory or the program memory :** The space where the sketch or

program is stored.

b) **Static Random access memory (SRAM):** It's the space where the Arduino creates and manipulates variables.

c) **EEPROM :** It's the long term memory of Arduino, which is used to hold information for a very long duration.

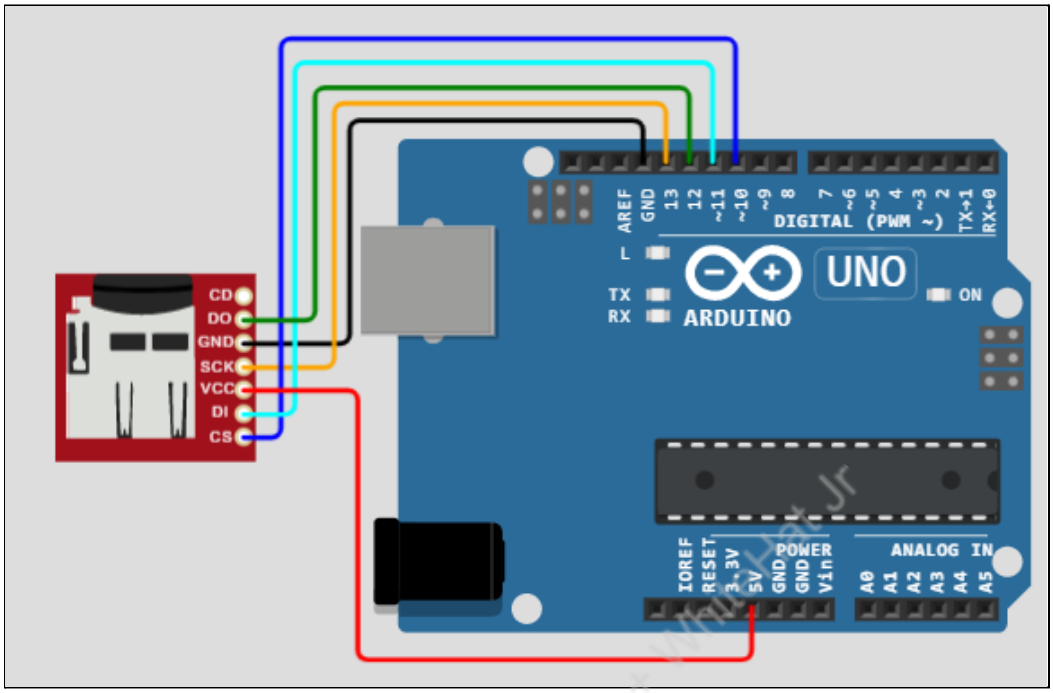
d) The arduino has 32 kb (flash memory) + 2 kb (SRAM) + 1kb (EEPROM) = 34 kb memory.

4. The pinout of the micro SD card module is as follows:
   - **CD (Card detect) :** This pin should be connected to **ground** if there is no SD card in the module, but in case of wokwi simulator, there is always a card within the module, so you don't need to worry about this pin.
   - **DO/MISO (Data out / Master in slave out) :** The **SD card** or the **slave** device uses this pin to **send** data towards the **Arduino** or the **master** device.
   - **GND (Ground) :** This pin is used to provide **0 volts** to our **SD card**.
   - **SCK (Serial clock line) :** This pin is used to generate **clock signals** so that the communication or the data exchange between the **Arduino (master)** and the **SD card (slave)** is **synchronous**.
   - **VCC (Power line) :** This pin is used to provide **5 volts** to the **SD card**.
   - **DI/MOSI (Data input / Master out slave in) :** The **SD card (slave)** uses this pin to **receive** data from the **Arduino (master)**.
   - **CS (Chip select / Slave select) :** This pin is used to decide, with which **slave** the **master** will communicate. (In case, there are multiple slaves)

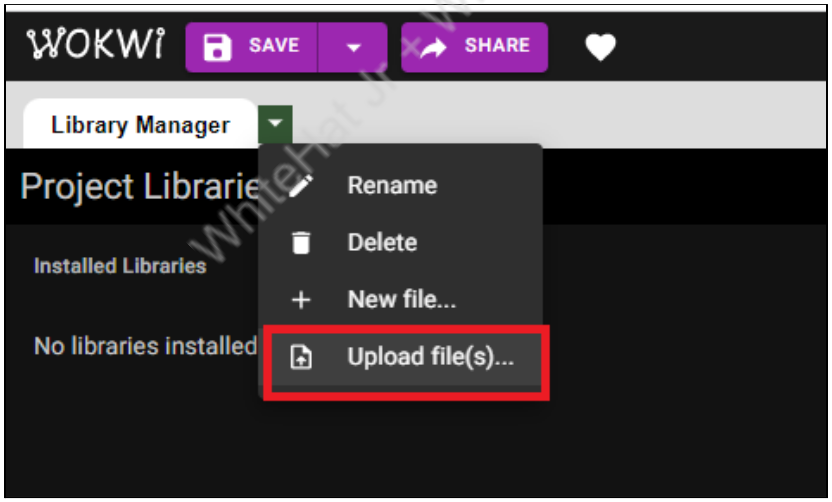
5. Make the following connections :

SD card	Arduino
CD **(Card detect)**	-
DO **(Data output)**	12
GND **(Ground)**	GND **(0 volts)**
SCK **(Serial clock line)**	13
VCC **(Power pin)**	5V **(5 Volts)**
DI **(Data input)**	11
CS **(Chip select)**	10

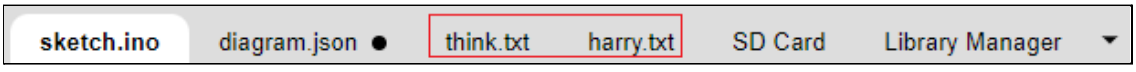
Connection Diagram for reference:



6. Open the link and download the files.
7. Upload the downloaded files into your project by clicking on the **Upload file(s)** button.



8. The files will be visible as shown:

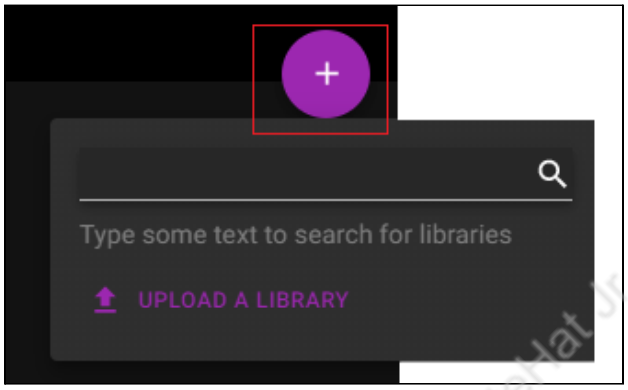


9. Setup the communication between the SD card and the arduino board by adding the SD library and writing the following code:

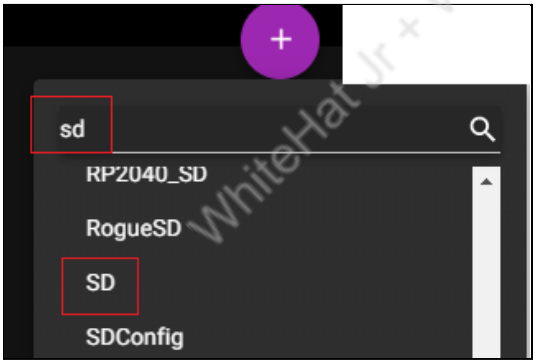
a. Import SD.h into using the Library Manager

Click on Library Manager

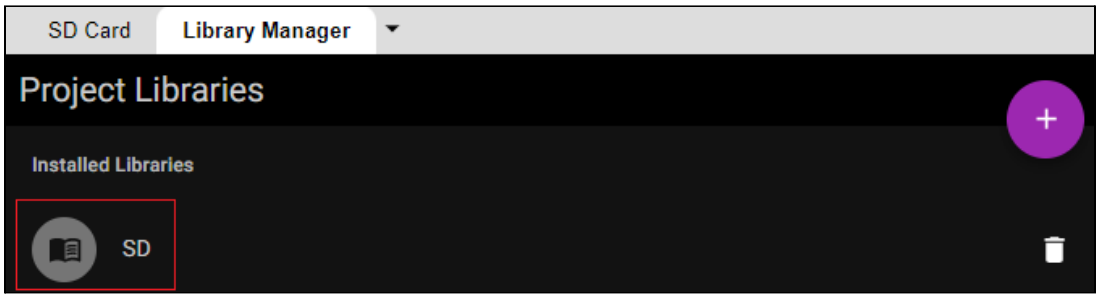
Click on the plus button on the right



Type SD and scroll down until you see SD. Click on SD



Verify that the file has been added when SD is seen under installed libraries



b. In **sketch.ino**, inside the **setup()** function, import the SD card module by writing:

```
#include <SD.h>
```

c. Create file object with global scope by writing

```
File file;
```

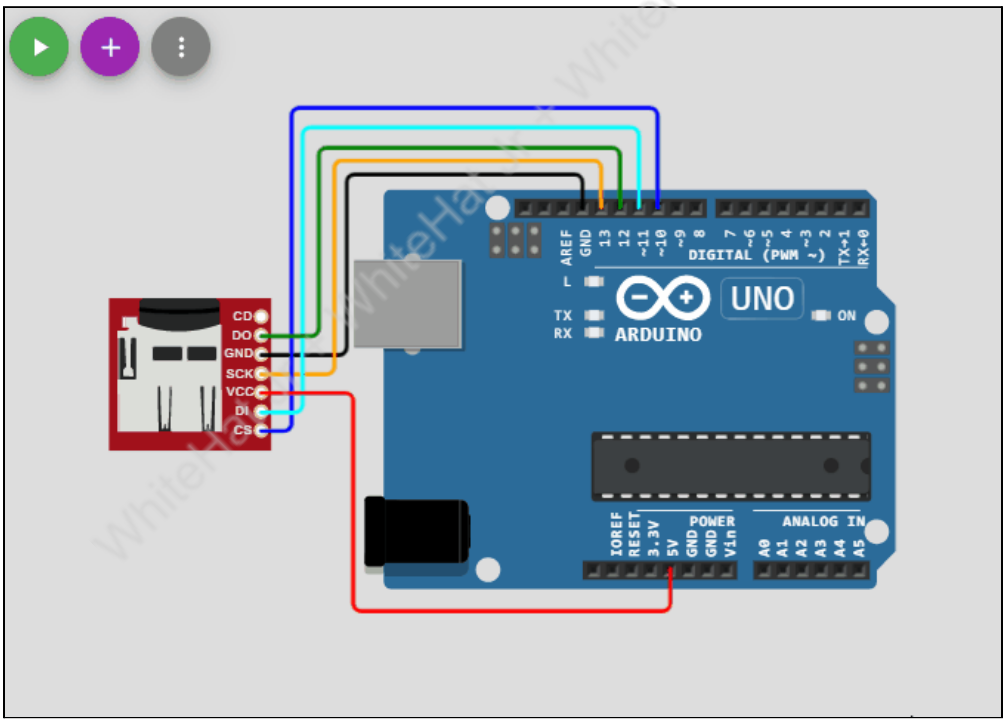
d. To start the communication write

```
Serial.begin(9600);
```

e. Verify that the SD card is detected by writing the if condition below

```
if (!SD.begin())Serial.println("SD card not detected");
else Serial.println("SD card detected");
```

OUTPUT:



10. Now that the files are ready, write code to
    a. open the file,
    b. read the contents of the file
    c. print them on the screen.

11. Open the file **harry.txt** by writing. The first parameter is the file name and the second parameter mentions that the mode of opening the file is read mode.

```
file = SD.open("harry.txt",FILE_READ);
```

12. Count the characters in the file by writing

```
Serial.println("Total characters");
Serial.println(file.size());
```

OUTPUT:

```
Total characters : 29324
```

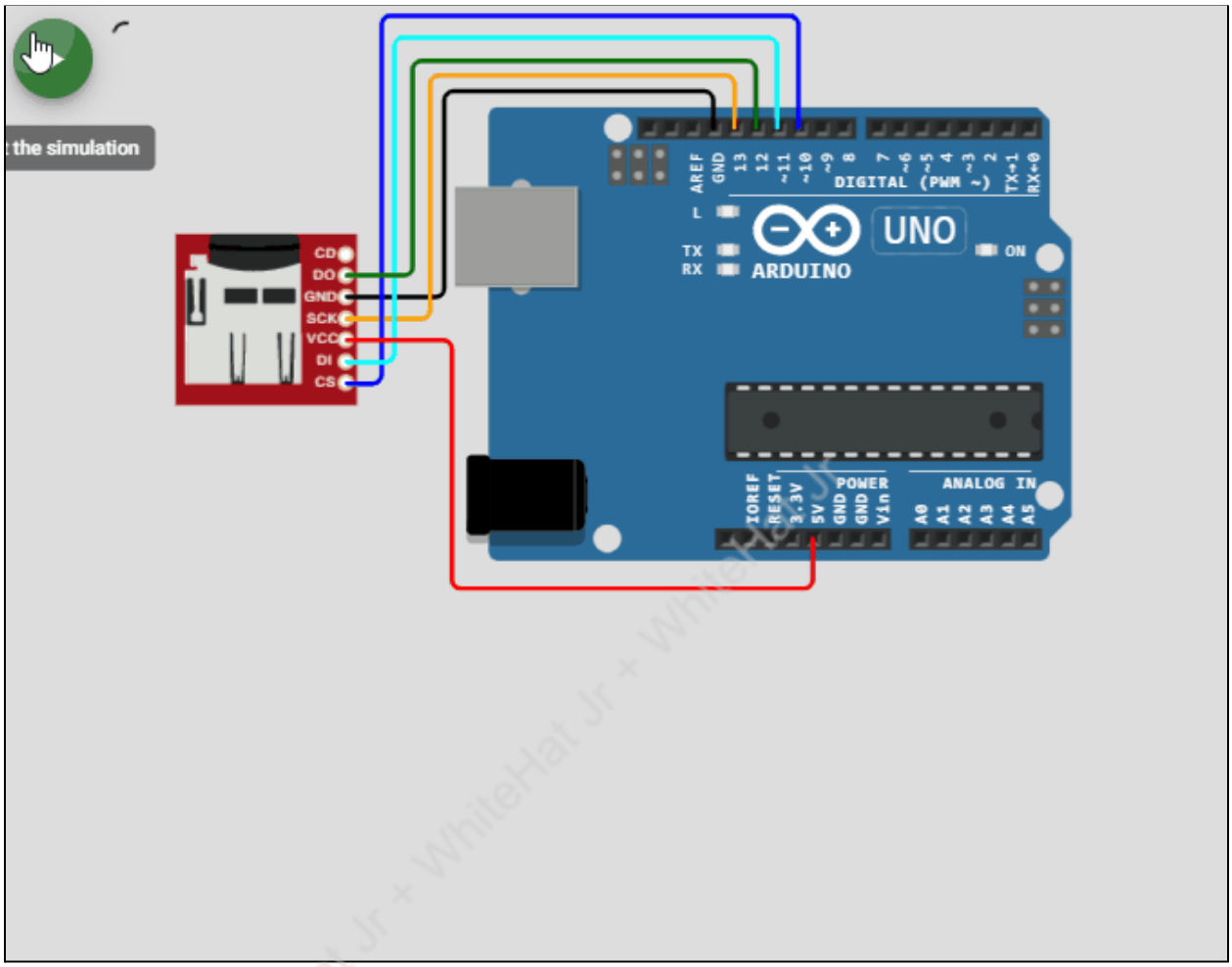
13. Read the contents of the file by doing the following steps:

- verify that the file object is valid.
- If the file object is valid, print “HARRY POTTER : ” on the serial terminal.
- Wait for 2 seconds.
- Check continuously whether there are characters available to read in the file or not.
- Read one byte or one character at a time.
- Print that character over the serial monitor.
- Once all the characters are extracted from the file and printed, the **loop** will **break**, and we will print an empty new line.
- **Close** the file.
- Print **“File is read successfully”**.

CODE:

```
if (file){
   Serial.println(message);
   delay(2000);
   while (file.available()){
      char data = file.read();
      Serial.print(data);
   }
   Serial.println();
   file.close();
}
Serial.println("File read successfully");
```

OUTPUT:



Click here to view the reference video.

14. Observe that our **harry.txt** file has **29324 characters.**
If the **think.txt** file is taken into consideration, it will have approximately a similar number of characters. So in total close to **60000 characters,** and the arduino has a memory of **34 kb (~34816 characters),** which means that it's not possible to **hard code** or **store** all this data in the **program memory** of the Arduino. An external storage is always helpful.

Also, let's comment out the section of code which prints the file data. After we have written the new function to read the file, we can remove this section of code.

```
void setup() {
  Serial.begin(9600);
  SD.begin();

  next.setDebounceTime(50);
  prev.setDebounceTime(50);
  select_book.setDebounceTime(50);
  open_book.setDebounceTime(50);

  /*file = SD.open("harry.txt", FILE_READ);
  Serial.print("Total characters :");
  Serial.println(file.size());

  if (file) {
    Serial.println("HARRY POTTER : ");
    delay(2000);
    while (file.available()) {
      char data = file.read();
      Serial.print(data);
    }
    Serial.println();
    file.close();
  }
  Serial.println("File read successfully");*/
}
```

15. Let's add another functionality as well. As we have two books, let's integrate the buttons in the circuit so that we can choose the books.

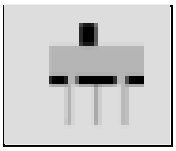
    a. Add two push buttons and a slider switch in the circuit.
    b. Connect them as per the following table.

Push Button (btn.1)	Arduino Uno board
1.l	2
2.r	GND

Push Button (btn.2)	Arduino Uno board
1.l	3
2.r	GND

    c. Integrate a **slide switch** component. This will be used to open or close a

book. This component almost works like a light bulb switch.



It has three pins - **Left terminal, ground** and **right terminal.**

d. Connect the **slide switch** component

Slide Switch	Arduino Uno board
sw1 :1	4
sw2 :2	GND
sw3 :3	5

16. Code:

a. Let's initiate a new variable named **book_num** to 0.

```
int book_num = 0;
int book_state = 0;
```

b. Include the **<ezButton.h>** header file

```
#include <ezButton.h>
```

c. Initiate all 4 buttons.

```
ezButton next(2);
ezButton prev(3);
ezButton select_book(4);
ezButton open_book(5);
```

d. In the **setup()** method, call the **setDebounceTime()** method for all the buttons.

```
next.setDebounceTime(50);
prev.setDebounceTime(50);
select_book.setDebounceTime(50);
open_book.setDebounceTime(50);
```

e. In the **loop()** method,

```
void loop(){

  next.loop();
  prev.loop();
  select_book.loop();
  open_book.loop();

}
```

f. If the **next** button is pressed, the **book_num** variable will increase and if the **prev** button is pressed the **book_num** variable will decrease.

This should be written in the **loop()** method.

```
if (next.isPressed()){
  book_num++;
}
else if (prev.isPressed()){
 book_num--;
}
```

g. As we have two books only, use the **constrain()** method so that the **book_num** cannot go beyond 0 and 1.

book_num = constrain(**book_num**, 0, 1);

h. Now, let's write a **check_book()** method, which will assign the **book_name** according to the selected **book_num**.

```
void check_book(){

    if (book_num)
      Serial.print("Think and Grow Rich");
    else
      Serial.print("Harry Potter");

}
```

i. Call **check_book()** method in the **loop()** method.

j. Now, when we run the code we can see that the name of the selected book is being printed multiple times. This is happening because it is running for every **loop(),** but we do not want it. We want it to run only when the book is changed.

Let's resolve this problem.

Define a variable named **prev_book_num** to -1.

```
int prev_book_num = -1;
```

k. In the **check_book()** method, add an if condition which compares the **prev_book_num** with the **book_num** variable, if they are not equal to each other then only print the book name.

Also, set **prev_book_num** equals to **book_num** whenever this condition is met.

```
void check_book(){
  if (prev_book_num  !=  book_num){
    prev_book_num = book_num;
    lcd.clear();
    if (book_num)
      Serial.println("Think and Grow Rich");
    else
      Serial.println("Harry Potter");
  }
}
```

l. Now, we will use the **slide switch** to open the selected book. According to our

definition, the **select_book** button on the **slide switch** will make the **button_state** to 0.

The **open_book** button on the **slide switch** will change the **button_state** to 1.

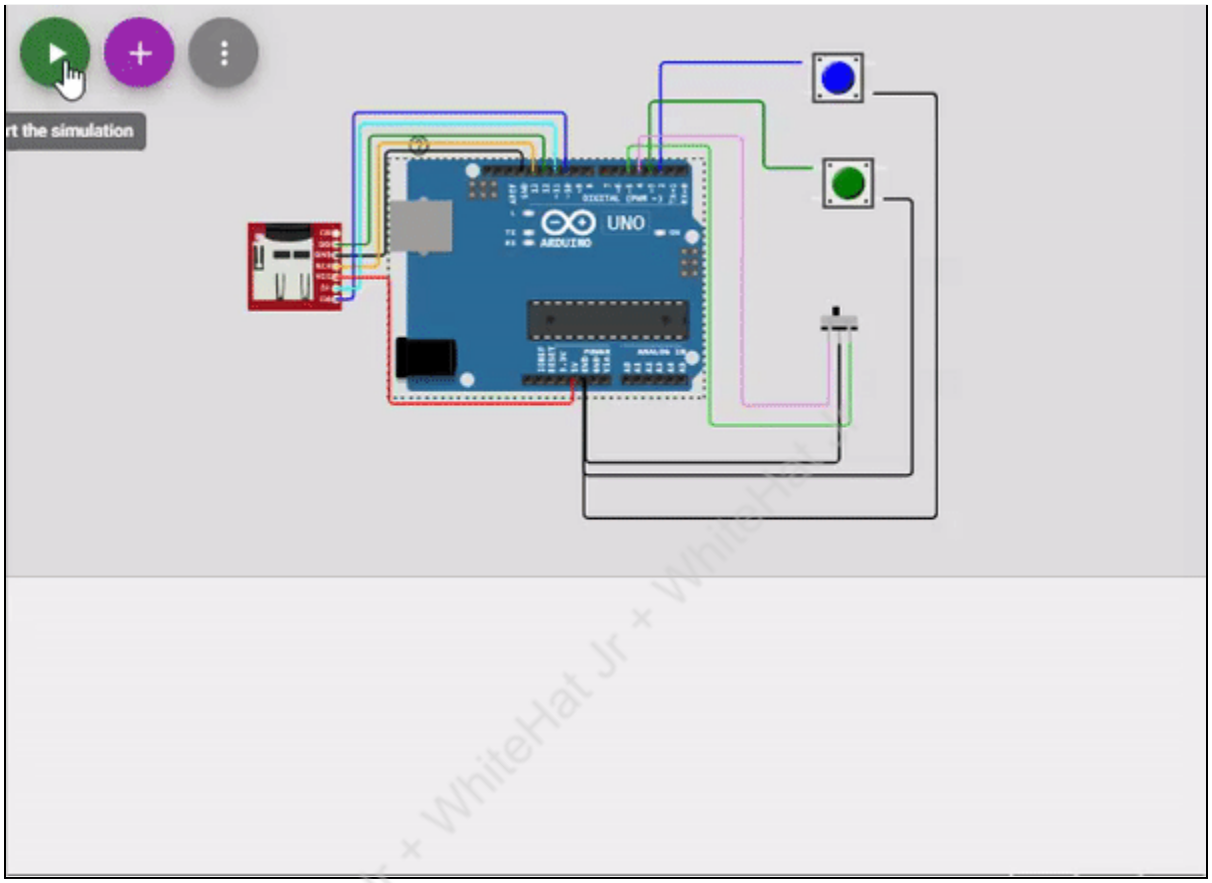
```
if (next.isPressed()) {
  book_num++;
}
else if (prev.isPressed()) {
  book_num--;
}
else if (select_book.isPressed()) {
  book_state = 0;
  prev_book_num = -1;
}
else if (open_book.isPressed())
  book_state = 1;
```

m. Now, let's define the **read_book()** method and print the book only when the **book_state** is 1.

```
void read_book() {
  if (book_state) { //  if book is opened
    if (book_num)
      file = SD.open("think.txt", FILE_READ);
    else
      file = SD.open("harry.txt", FILE_READ);

    String display_text = "";
    if (file) {
      while (file.available()) {
        char data = file.read();
        Serial.print(data);
      }
    }
  }
  file.close();
}
```

Reference output:



Click here to view the reference output video.

What's NEXT?

In the **next class**, we will complete the digital book project.