Smart Clock-I

What is our GOAL for this CLASS?

In this class, we learned how to calculate time using the RTC module.

What did we ACHIEVE in the class TODAY?

- Calculated time internally with Arduino
- Understood the need for the RTC (Real-time clock) module
- Learned how to use the RTC module with Arduino

Which CONCEPTS/ CODING BLOCKS did we cover today?

- **Concepts**: RTC, Fetch date and time from Arduino, Transmitting data, Receiving data, Infinite loops, Sequencing of code, OOP
- **Coding blocks**: Serial.begin(), Serial.print(), Serial.available(), delay(), Serial.readString(), Serial.setTimeout(), pinMode(), digitalWrite(), String keyword, toInt(), rtc.now(), DateTime, hour(), minute(), second(), year(), month(), day()

How did we DO the activities?

1. **Learned about RTC Module:**

   RTC is a Real-Time Clock.
   A real-time clock is a clock that keeps track of the current time and that can be used to program actions at a certain time.
   The chip maintains seconds, minutes, hours, days, dates, months, and year information. The date at the end of the month is automatically adjusted for months with fewer than 31 days, including corrections for leap years (valid up to 2100).
   The clock operates in either the 24-hour or 12-hour format with an AM/PM indicator.

   The **DS1307** incorporates a battery input and maintains accurate timekeeping when the main power to the device is interrupted.

   Below is the diagram of RTC.

   Pin Description:
   SCL is the clock input for the I2C interface and is used to synchronize data movement on the serial interface.

   SDA is the data input/output for the I2C serial interface.

   VCC/5V pin supplies power for the module. It can be anywhere between 3.3V to 5.5V.

   GND is a ground pin.

2. **Learned how to calculate time internally with Arduino:**

   Open the wokwi simulator and create a new Arduino Uno project.

   a. Use variables to store the minutes, hours, and seconds.

```
byte seconds = 0;

byte minutes = 0;
```

```
byte hours = 0;
```

b. Ask users to enter the time. It should be done at the very beginning and once. So we wrote in **setup()**.

```
//manual setting time in the beginning
Serial.print("Enter the hours passed : ");
while (!Serial.available());
hours = Serial.readString().toInt();
Serial.println(hours);

Serial.print("Enter the minutes passed : ");
while (!Serial.available());
minutes = Serial.readString().toInt();
Serial.println(minutes);
```

c. Pass the value 9600 to the speed parameter. This tells the Arduino to get ready to exchange messages with the Serial Monitor at a data rate of 9600 bits per second. That's 9600 binary ones or zeros per second and is commonly called a baud rate.

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

d. Create a function to calculate time. Every 60 seconds makes a minute and so on. To do so and validate the time, define the function.

```
void time_check(){
if (seconds >= 60){
  seconds = 0;
  minutes++;
}
if (minutes >= 60){
  minutes = 0;
  hours++;
}
```

```
if (hours >= 24){
  hours = 0;
}
}
```

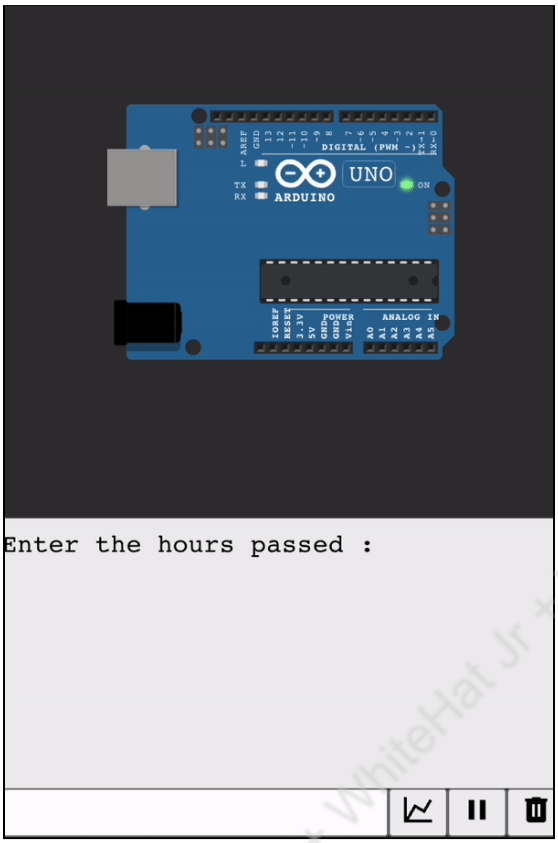
e. Call this function in the **loop()** to keep validating the time after updating.

```
time_check();
```

f. Next, to keep updating time after every second, delay it for a second and print the current time to check on the console. Then, increase the seconds variable by 1 unit.

```
delay(1000);
String time = "Time : " + String(hours) + ":" + String(minutes) +
              ":" + String(seconds);

Serial.println(time);
seconds++;
```

Reference output:



3. **Learned how to connect and use RTC Module with Arduino**

First, we created the circuit diagram.

Step 1: Select the components:

- 1 x **Arduino Uno**
- 1 x **DS1307 RTC**

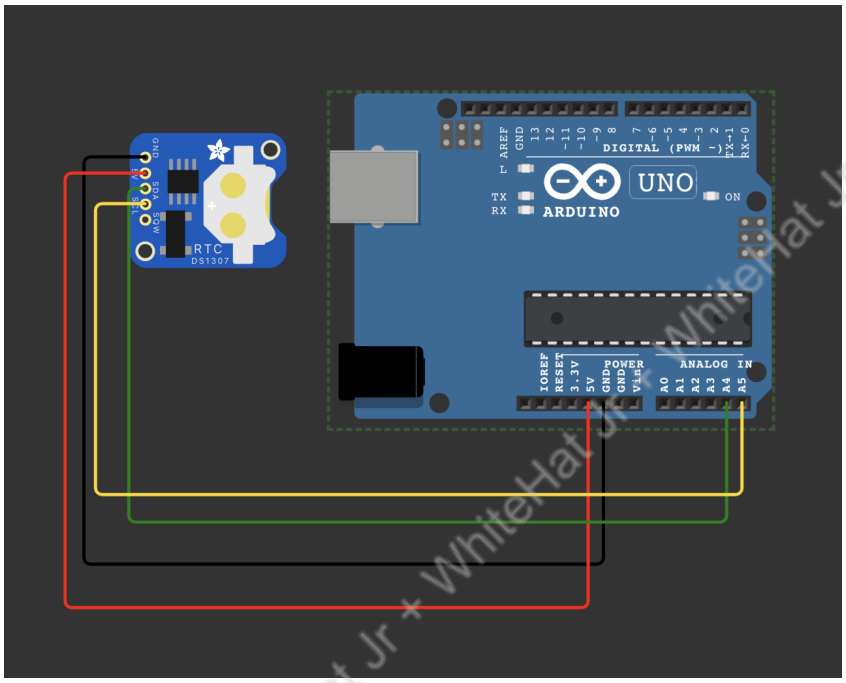
Step 2: Made the connections:

The circuit of this project consists of an **Arduino** Controller and a **DS1307 RTC**.

DS1307 RTC	Arduino PIN
VCC	5V
GND	GND

SDA	A4
SCL	A5

Reference image:



1. Added the library

```
#include <RTClib.h>
```

2. Created an object to work with the RTC module.

```
RTC_DS1307 rtc;
```

3. Checked if it is initialized or not.

```
void setup(){
 Serial.begin(9600);
  if (!rtc.begin()){
```

```
  Serial.println("RTC not initialized");
  while(true);
}
Serial.println("RTC found");
}
```

4. Created the variables to store the date and time.

```
String rtc_date = "";
String rtc_time = "";
```

5. Created a function to get date.

```
String get_date(DateTime current){

 int year = current.year();
 int month = current.month();
 int day = current.day();

 String current_date = "Date : " + String(day) + "/" + String(month) +
           "/" + String(year);
  return current_date;
}
```

6. Similarly, we created a function to get time.

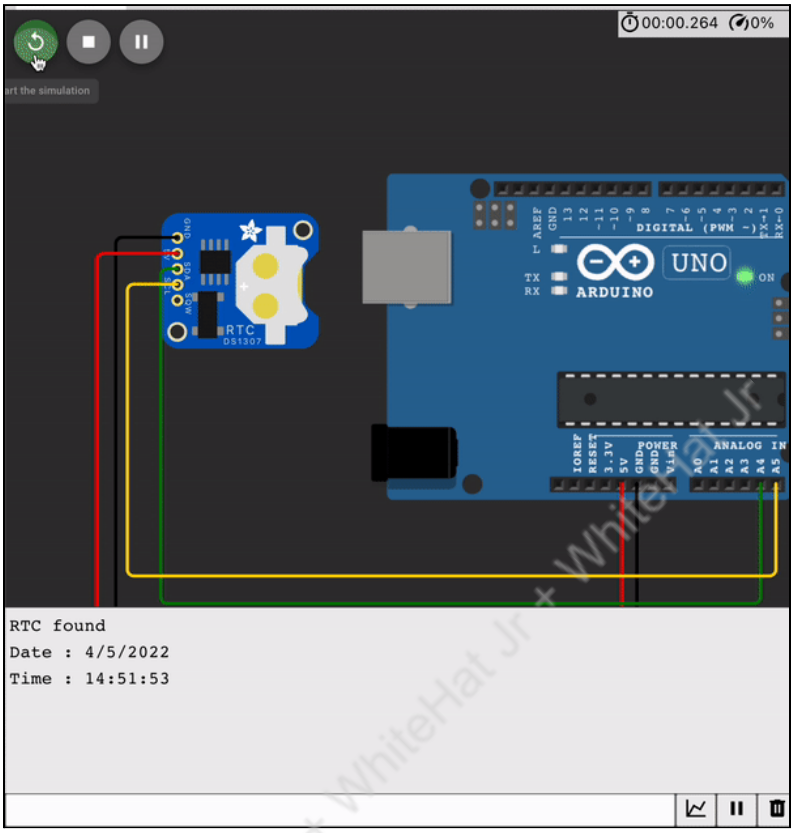
```
String get_time(DateTime current){

 int hour = current.hour();
 int minute = current.minute();
 int second = current.second();
```

```
String current_time = "Time : " + String(hour) + ":" + String(minute) +
                      ":" + String(second);


return current_time;
}
```

7. Called these functions in a loop to get date and time and save them in variables.
   rtc.now() returns the current date and time together. To extract them separately, we used hour(), minute(), second(), day(), month(), year() instructions above.

```
void loop(){
DateTime dt = rtc.now();
rtc_date = get_date(dt);
rtc_time = get_time(dt);

//  for better working of simulator
delay(10);
}
```

Reference output:



What's NEXT?

In the **next class**, we will learn about encoders and interrupts.

Expand Your Knowledge

Read more about the RTC here.