

Topic	REMOTE CONTROLLED LIGHT	
Class Description	Students will learn about IR receivers. Students will also learn how remote control devices transmit messages to IR receivers. Using this knowledge, students will create a project to control an RGB LED remotely.	
Class	PRO C275	
Class time	50 mins	
Goal	<ul style="list-style-type: none"> Understanding IR receivers' working principle. Decoding messages received by IR receivers from a remote control device. Understanding how RGB LEDs function. Build a program to light up the RGB LED when a signal is detected. 	
Resources Required	<ul style="list-style-type: none"> Teacher Resources: <ul style="list-style-type: none"> Laptop with internet connectivity Earphones with mic Notebook and pen Smartphone Student Resources: <ul style="list-style-type: none"> Laptop with internet connectivity Earphones with mic Notebook and pen 	
Class structure	Warm-Up Teacher-Led Activity Student-Led Activity Wrap-Up	10 mins 15 mins 15 mins 10 mins
WARM-UP SESSION - 10 mins		
Teacher Action		Student Action

<p>Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today?</p> <p>Following are the WARM-UP session deliverables:</p> <ul style="list-style-type: none"> • Greet the student. • Revision of previous class activities. • Quizzes. 	<p>ESR: Hi, thanks! Yes, I am excited about it!</p> <p>Click on the slide show tab and present the slides</p>
<p style="text-align: center;">WARM-UP QUIZ Click on In-Class Quiz</p>	
<p>Activity Details</p> <p>Following are the session deliverables:</p> <ul style="list-style-type: none"> • Appreciate the student. • Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students. 	
<p style="text-align: center;">TEACHER-LED ACTIVITY 15mins</p>	
<p style="text-align: center;">Teacher Initiates Screen Share</p>	
<ul style="list-style-type: none"> • Understanding IR receivers' working principle. • Decoding messages received by IR receivers from a remote control device. • Understanding how RGB LEDs function. 	
<p style="text-align: center;">Teacher Action</p>	<p style="text-align: center;">Student Action</p>
<p>Do you remember what we did in the previous class?</p> <p>What did we use to store the books for the project?</p>	<p>ESR: We completed the digital reading device project.</p> <p>ESR: We had used an SD card.</p>

<p>Did we learn about some other new component?</p> <p>Great! Do we have any doubts from the previous class?</p> <p><i>If the student has any doubts, clarify the doubts.</i></p>	<p>ESR: Yes. We had used a Slider Switch.</p> <p>ESR: varied.</p>
<p>Have you seen a remote controlled car?</p> <p>How do you make these cars move in a certain direction?</p> <p>Exactly! We use a remote control.</p> <p>Remote control is a device which is used to operate equipment from a distance. What are the other things that we usually operate with remote control?</p> <p>Great! Today, we will try to build a project using a remote control using an Arduino Uno board.</p> <p>But first let's understand how a remote control works.</p>	<p>ESR: Varied.</p> <p>ESR: We can do it with the help of the remote control.</p> <p>ESR: We operate TVs, ACs etc.</p>
<p><u>Working principle of Remote control devices:</u></p> <p>Do you know if someone is in danger, they can send SOS messages by flashing a flashlight?</p>	<p>ESR: Varied!</p>

We can flash the flashlight quickly three times, then slowly three times and then again quickly three times. That's the code for an SOS message. This message uses Morse code which is used to send information by using only dots(.) and dashes(-).

A Remote control also works in this way. It flashes patterns of light which transmits our messages towards our TV or AC.

But we can't see this light emitted by the remote control. It's called infrared light. Infrared light is beyond the visible light range.

When we press different buttons on the remote control, it sends different signals.

Our remote control is the sender and TV is the receiver.

Let's design the circuit first.

For our project, we are going to open wokwi.com and start a new Arduino Uno project.

1. Components:

- 1 x **Arduino Uno board**
- 1 x **IR Receiver**
- 1 x **IR Remote**

2. **Let's do connections:**

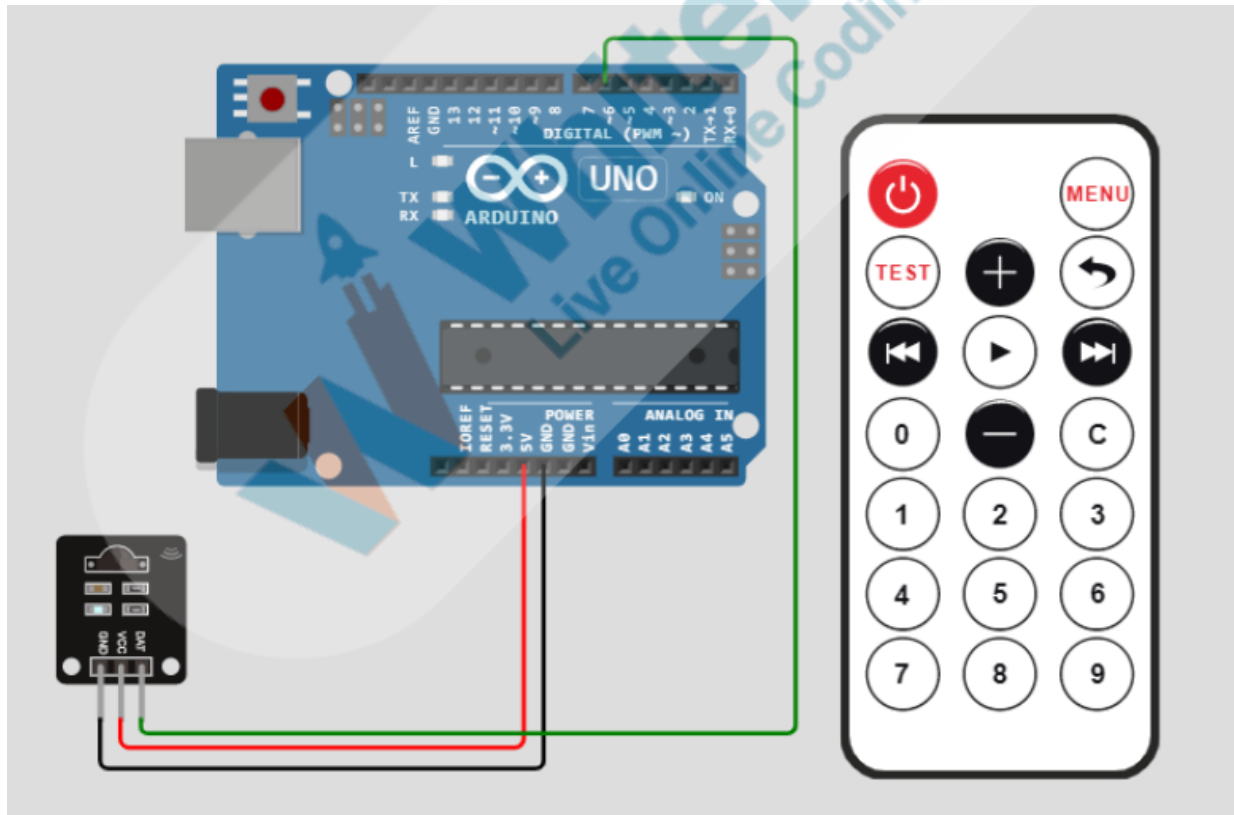
The circuit of this project consists of an Arduino Uno board, an **IR Receiver** and **IR Remote**.

We will connect the Arduino Uno board and IR Receiver. We don't need to connect the IR Remote.

IR Receiver	Arduino Uno PIN
VCC	5V
GND	GND.2
DAT	D6

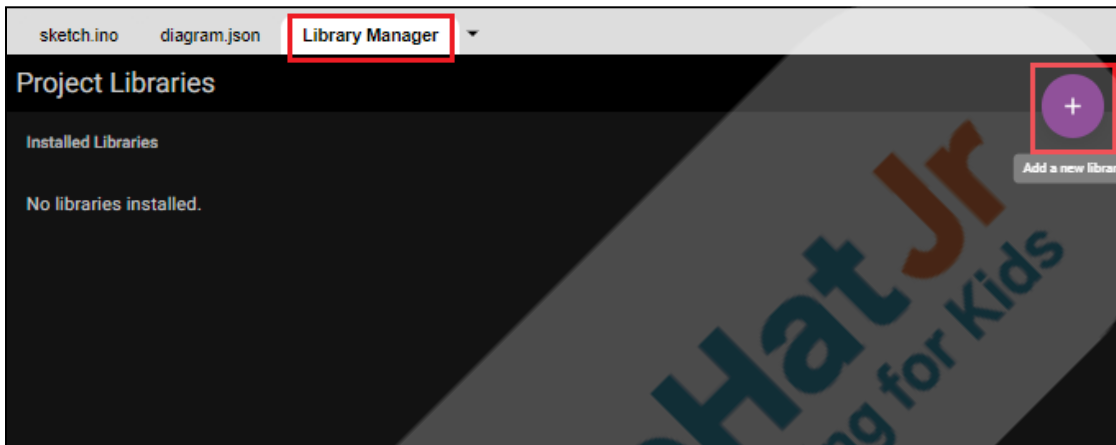
*Note: Wire color can be changed by **clicking over it** and selecting the color, or via **diagram.json** file. Go to the **diagram.json** wire and change the color of the wire. Any design changes or color changes can be done via the **diagram.json** file. Keep the track of the component and change the design settings.*

Reference:



Now it's time to write the code.

1. First, we need to go to the **Library Manager** and **Add a new library** called **IRremote**. Once we add the Library, a new file named **libraries.txt** will automatically be created.



2. Now, go to **sketch.ino**. We need to add the header file here.

```
#include <IRremote.h>
```

3. Now, add a variable which holds the pin number to which the **IR receiver** is connected.

```
const byte recv_pin=6;
```

4. Let's make an **object** of the receiver class.

```
IRrecv receiver(recv_pin);
```

5. Now, let's write the following code inside the **setup()** function.

- we need to enable/start the IR receiver.

```
receiver.enableIRIn();
```

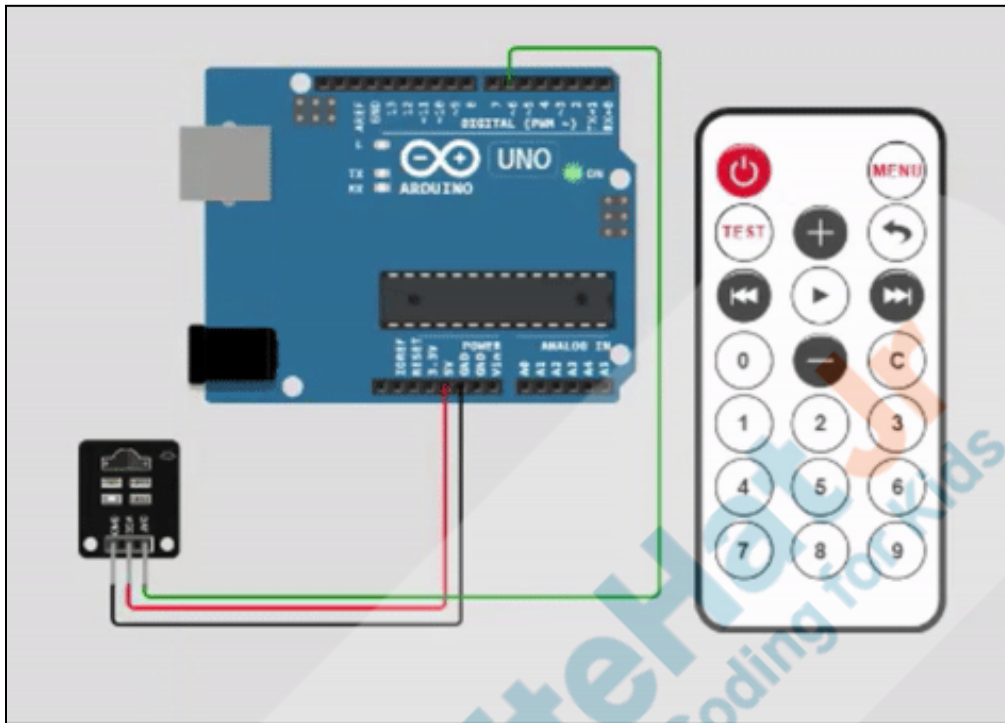
- To test if the remote is receiving from the IR sensor or not, we will use the `blink13()` function. There is an in-built LED connected to the 13th pin which will glow up when the IR receiver receives a message –

```
receiver.blink13(true);
```

Your code should look like this -

```
1  #include <IRremote.h>
2
3  const byte recv_pin=6;
4
5  IRrecv receiver(recv_pin);
6
7  void setup() {
8
9      Serial.begin(9600);
10
11     receiver.enableIRIn();
12
13     receiver.blink13(true);
14
15 }
16
17 void loop() {
18
19 }
```

Let's observe the output-



[Click here](#) to view the output video.

We can see that the red light beside the 13th pin blinks when we press any button on the remote control.

Now, our Arduino Uno board is successfully receiving messages from the remote control.

Now, there are two things we need to understand about the IR signals-

1. IR Interference:

- Basically, every living being emits some infrared light. There are many other sources of infrared rays like- the sun, fire, animals, human beings. The IR receiver might pick those signals up which we don't want.

- Solution:

To prevent this problem, IR signals from a remote control are modulated. To modulate means to adjust a property of a signal in a certain way.

Let's say when a singer is singing, they can modulate their voices to sing louder/ quieter. They can also modulate their voices to be sharper / deeper.

Similarly, to make the IR signals (from remote control) stand out among all the interference, the signal is modulated at a certain frequency (38KHz is the most common carrier frequency).

2. How do we know which button is pressed?

Each button has a unique code so that we can identify each button separately.

Similar to the SOS signal, each data is encoded here as well.

To make the program understand the signal we need to decode it. For that we will add few lines of code in the **loop()** method-

- decode() - method looks for IR signals, and if received, it decodes them. We will use this function to write a conditional statement.

```
void loop(){  
  if (receiver.decode()){  
  }  
}
```

- Now, we want to store the decoded IR response in a variable and print it on the console.

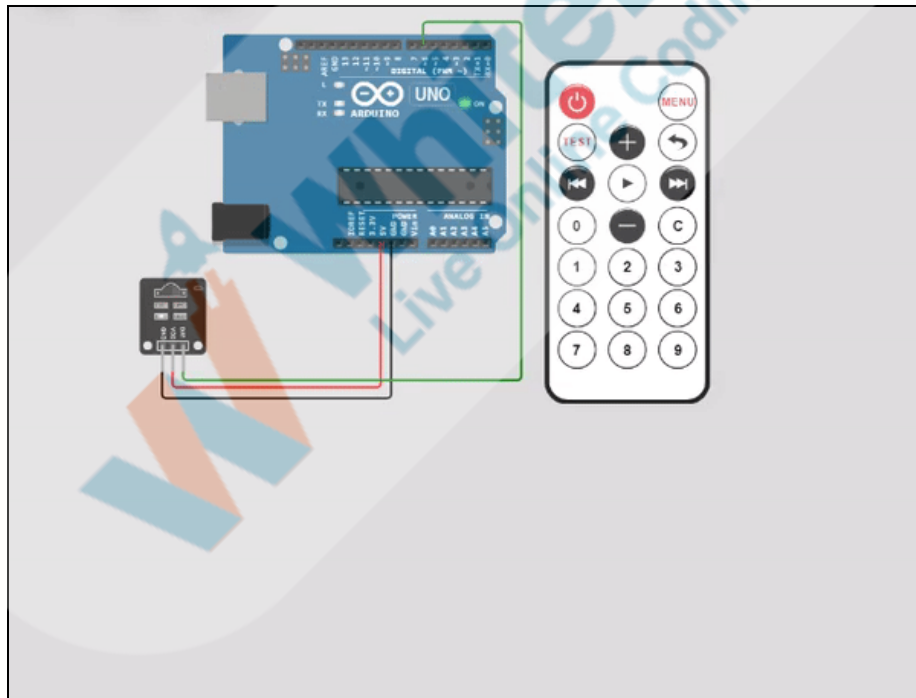
```
if (receiver.decode()){  
  
  int response = receiver.decodedIRData.command;  
  Serial.println(response);  
}
```

- Once a signal is received, we want the IR receiver to resume with the next signal. So , we write -

receiver.resume();
- Add a **delay()** method outside the if condition to make the simulator wait a little in between and improve it's working.

```
void loop(){  
  
  if (receiver.decode()){  
  
    int response = receiver.decodedIRData.command;  
    Serial.println(response);  
  
    receiver.resume();  
  }  
  
  delay(10);  
}
```

Let's look at the output-



[Click here](#) to view the output video.

In the output, we can observe there are certain codes for each key on the remote. e.g.

Key 1 has the code - 48,

Key 2 has the code - 24,

Key 3 has the code - 122 etc.

Now, what do we want to control with this remote?

ESR: Varied.

We will work on it. But today we are going to control an **RGB LED** with our remote control and you are going to build it.

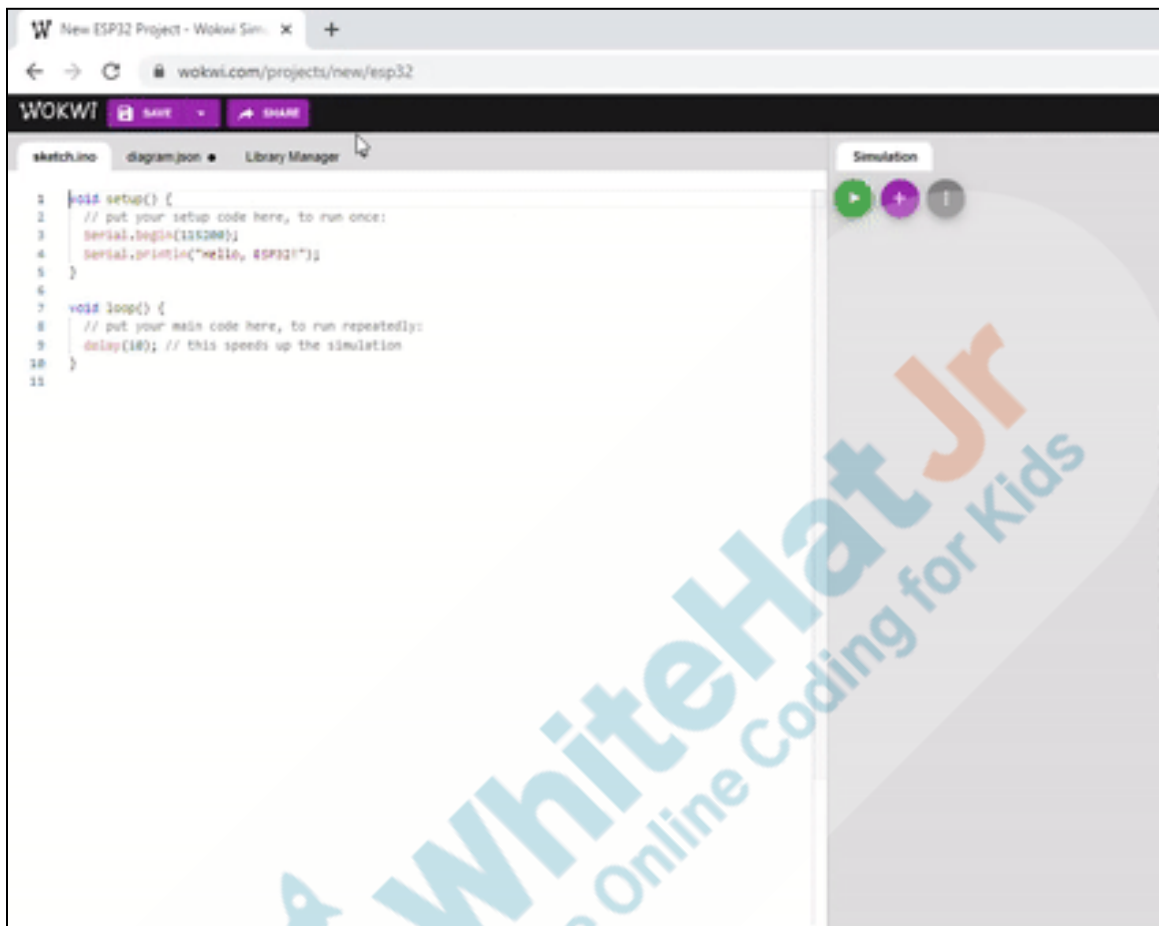
Before you start with your task. Let's understand **RGB LEDs** first.



You can imagine an **RGB LED** as three different red, green and blue LEDs. A normal Led has two legs- a cathode and an anode. An RGB LED has four legs, which are-

- Common pin (COM) - By default, the common pin is the anode (positive). You can change it by setting the "common" attribute to "cathode". You can imagine the common pin as all the cathode legs/ anode legs of the three LEDs tied together.
- Red LED (R) - if the COM pin is "anode", R would be the "cathode" of the Red LED and vice versa.
- Green LED (G) - if the COM pin is "anode", G would be the "cathode" of the Green

LED and vice versa. <ul style="list-style-type: none"> Blue LED (B) - if the COM pin is “anode”, B would be the “cathode” of the Blue LED and vice versa. 	
As you understood the basics. You will build a project to build a light system controlled remotely.	
Student Stops Screen Share	
Can you solve it? Let's try. I will guide you through it.	ESR: Yes, sure!
STUDENT-LED ACTIVITY- 15 mins	
<ul style="list-style-type: none"> Ask the student to press the ESC key to come back to the panel. Guide the student to start Screen Share. The teacher gets into Full Screen. 	
Student Initiates Screen Share	
<div style="text-align: center;"><u>ACTIVITY</u></div> <ul style="list-style-type: none"> Build a program to light up the RGB LED when a signal is detected. 	
Teacher Action	Student Action
<i>Teacher guides the student to download boilerPlate code from Student activity 2.</i>	Student opens wokwi simulator.
<i>Unzip the downloaded file and upload it in your wokwi project folder.</i>	



Let's try to create the circuit diagram first.

Step - 1: Add the following components-

- 1x RGB LED
- 3x Resistors

Step - 2: Let's change the COM pin of the RGB LED to "cathode".

- Go to **diagram.json** → find **wokwi-rgb-led** in parts → change the **common** attribute as **cathode**.

```

h.ino  diagram.json  libraries.txt  Library Manager
{
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-arduino-uno", "id": "uno",
    { "type": "wokwi-ir-receiver", "id": "ir1",
    { "type": "wokwi-ir-remote", "id": "remote1"
    {
      "type": "wokwi-rgb-led",
      "id": "rgb1",
      "top": -304.95,
      "left": 154.01,
      "attrs": { "common": "cathode" }
    },
    {
      "type": "wokwi-resistor",

```

Step - 3: Let's change the resistors value. By default it is 1000 ohm, which is very high. Let's change it to 220 ohm. (ohm is the unit with which resistance is measured).

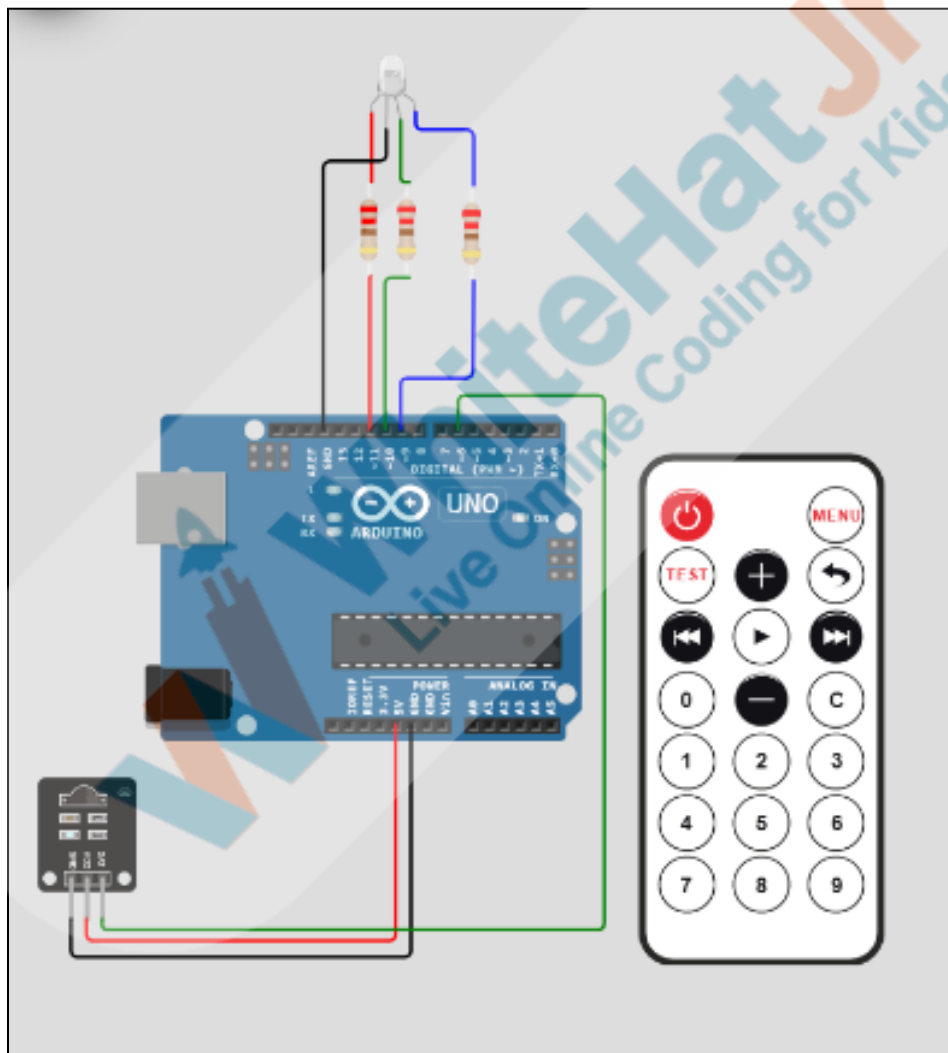
- Go to **diagram.json** → find **wokwi-resistor** in parts → change the **value** attribute to **220**.
- Do it for all three resistors.

Step - 4: Let's connect the **RGB LED** as per the instructions given below:

RGB LED	Arduino Uno PIN
COM	GND.1
R	Connect it to pin 11 through a resistor
G	Connect it to pin 10 through a resistor
B	Connect it to pin 9 through a resistor

*Note: Wire color can be changed by **clicking over it** and selecting the color, or via **diagram.json** file. Go to the **diagram.json** wire and change the color of the wire. Any design changes or color changes can be done via the **diagram.json** file. Keep the track of the component and change the design settings.*

Reference circuit:



Let's work on the code now.

1. Initiate 3 const to store the pin values to which R,G,B pins are connected to.

```
const byte r_pin=11;  
const byte g_pin=10;  
const byte b_pin=9;
```

2. The RGB LED receives output from the controller. So, the next task would be to define the pin mode as OUTPUT.

- a. Go to **setup()** method and define pin mode for R, G, B pins.

```
pinMode(r_pin, OUTPUT);  
pinMode(g_pin, OUTPUT);  
pinMode(b_pin, OUTPUT);
```

3. We use `analogWrite(pin, value)` function to light up RGB LEDs.

To light up the LED with red color-

```
analogWrite(r_pin,255);
```

To light up the LED with blue color-

```
analogWrite(b_pin,255);
```

To light up the LED with green color-

```
analogWrite(g_pin,255);
```

4. Now, if we want a compound color (e.g. Yellow , purple, cyan etc.), we need to write the **analogWrite()** function for all three pins.

Let's write the **lit_led()** function which we will use whenever we want to light up the LED. We will pass three values i.e. r, g, b through this function -

```
53  
54 void lit_led(int r,int g, int b){  
55  
56     analogWrite(r_pin,r);  
57     analogWrite(g_pin,g);  
58     analogWrite(b_pin,b);  
59  
60 }  
61
```

5. Now, what should we do?

ESR: We should light up the **RGB LED** with different colors for different buttons on the remote control.

Great! We will light up the **RGB LED** with different colors for different buttons on the remote control.

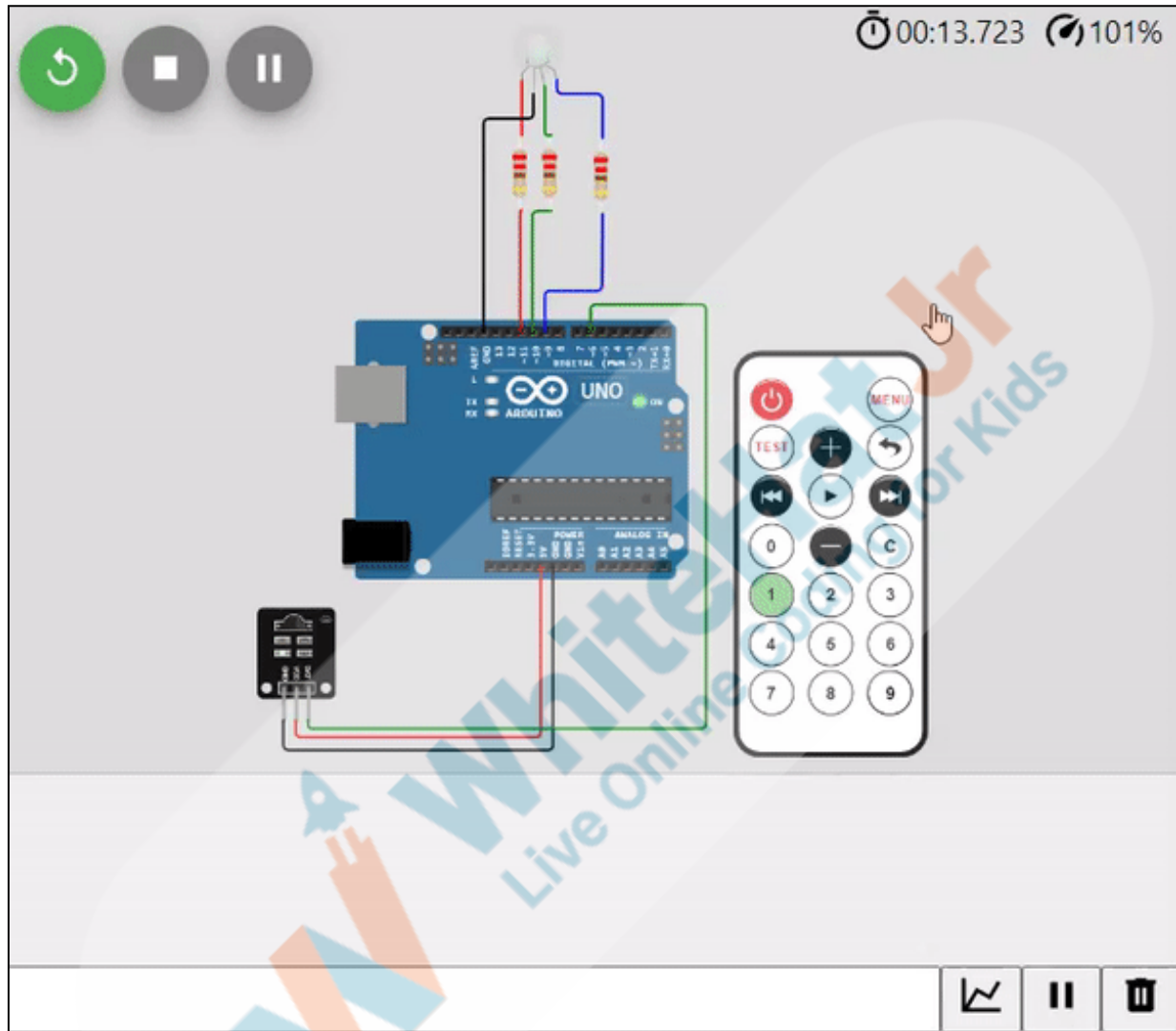
We will write a function for that. We will name the function as **action()** and we will pass an argument through this function. This argument will hold the decoded value of the button pressed on the remote control -

```
35 void action(int num){
36
37     if(num== 48)
38     | lit_led(255,0,0);
39     else if(num == 24)
40     | lit_led(0,255,0);
41     else if(num == 122)
42     | lit_led(0,0,255);
43     else if(num == 16)
44     | lit_led(255,255,0);
45     else if(num == 56)
46     | lit_led(0,255,255);
47     else if(num == 90)
48     | lit_led(255,0,255);
49     else
50     | lit_led(255,255,255);
51
52 }
```

6. Let's call the **action()** method inside the **loop()** method. Also, pass the **response** variable through the function -

```
void loop() {
    if(receiver.decode()){
        int response=receiver.decodedIRData.command;
        action(response);
        Serial.println(response);
        receiver.resume();
    }
}
```

Reference Output:



[Click here](#) to view the reference video.

Great, we have now completed our project with PIR sensor and NeoPixel Ring.

Teacher Guides Student to Stop Screen Share

WRAP-UP SESSION - 05 mins

Activity details

Following are the WRAP-UP session deliverables:

- Appreciate the student.
- Revise the current class activities.
- Discuss the quizzes.

WRAP-UP QUIZ

Click on In-Class Quiz




Activity Details

Following are the session deliverables:

- Explain the facts and trivia
- Next class challenge
- Project for the day
- Additional Activity (Optional)

FEEDBACK

- **Appreciate and compliment the student for trying to learn a difficult concept.**
- **Get to know how they are feeling after the session.**
- **Review and check their understanding.**

Teacher Action	Student Action
<p>You get “hats-off” for your excellent work!</p> <p>In the next class, we will learn about a new television component and add it to the circuit. We will also learn to display images and text on it.</p>	<p><i>Make sure you have given at least 2 hats-off during the class for:</i></p> <div> <div>Creatively Solved Activities  +10</div> <div>Great Question  +10</div> <div>Strong Concentration  +10</div> </div>

PROJECT OVERVIEW DISCUSSION

Refer the document below in Activity Links Sections

Teacher Clicks

✕ End Class

ADDITIONAL ACTIVITIES

(Optional)

Additional Activities

ACTIVITY LINKS

Activity Name	Description	Links
Teacher Activity 1	Simulator	https://wokwi.com/
Teacher Activity 2	IR receiver documentation	https://docs.wokwi.com/parts/wokwi-ir-receiver
Teacher Activity 3	IR remote documentation	https://docs.wokwi.com/parts/wokwi-ir-remote
Teacher Reference 1	Reference Code	https://github.com/procodingclass/P-RO-C275-Reference-Code
Teacher Reference 2	Project	https://s3-whjr-curriculum-uploads.whjr.online/08d7ab66-ada3-4d71-9cd9-e38ba6755bd7.pdf
Teacher Reference 3	Project Solution	https://github.com/procodingclass/P-RO-C275-Project-Solution
Teacher Reference 4	In-Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/1c09991a-0fbc-4d52-bcf3-ed453ce208ce.pdf
Student Activity 1	Simulator	https://wokwi.com/

Student Activity 2	Student Activity Boilerplate code	https://github.com/procodingclass/PRO-C275-Student-Boilerplate
Student Activity 3	IR receiver documentation	https://docs.wokwi.com/parts/wokwi-ir-receiver
Student Activity 4	IR remote documentation	https://docs.wokwi.com/parts/wokwi-ir-remote

