



Theory and steps for "Create a VM instance using OpenNebula Sunstone GUI over Oracle Windows".

Sure, here are the theory and steps to create a VM instance using OpenNebula Sunstone GUI over Oracle Windows:

Theory:

OpenNebula is an open-source cloud computing platform that enables the management of virtualized data centers. It allows administrators to manage virtual machines (VMs) and other resources such as storage and networking in a unified way.

OpenNebula provides a web-based graphical user interface called Sunstone that allows users to manage their virtual resources easily.

Oracle Windows is a widely used operating system in the enterprise environment. In this tutorial, we will create a VM instance using OpenNebula Sunstone GUI over Oracle Windows. We assume that you have already installed OpenNebula and Oracle V machine.



ShareGPT

Powered by ▲Vercel



Copy



1



0s



0

Steps:

1. Open the Sunstone web interface by typing the IP address of the OpenNebula server into your web browser. The default port for Sunstone is 9869, so the URL should be something like `http://<ip_address>:9869/`.
2. Enter your OpenNebula username and password to log in to Sunstone.
3. Click on the "Virtual Resources" tab in the top menu and select "VM Instances" from the drop-down menu.
4. Click on the green "plus" button to create a new VM instance.
5. In the "General" tab of the VM creation wizard, enter a name for your VM instance and select the operating system you want to install. In this case, select "Windows" as the operating system.
6. In the "Hardware" tab, select the number of CPUs, amount of RAM, and storage capacity for your VM instance.
7. In the "Network" tab, select the network interface you want to use



ShareGPT

Powered by ▲Vercel

for your VM instance.

8. In the "Contextualization" tab, select the contextualization options you want to use. Contextualization allows you to customize your VM instance by providing configuration information, such as hostname, IP address, and user credentials.
9. Click on the "Create" button to create your VM instance.
10. Once your VM instance is created, you can start it by clicking on the "Power" button next to its name in the VM Instances list.

Congratulations! You have successfully created a VM instance using OpenNebula Sunstone GUI over Oracle Windows.



Theory and steps for "Migrate a VM from one host to another using OpenNebula".

Sure, here are the theory and steps to migrate a VM from one host to another using OpenNebula:

Theory:



ShareGPT

Powered by ▲Vercel

OpenNebula is an open-source cloud computing platform that enables the management of virtualized data centers. It allows administrators to manage virtual machines (VMs) and other resources such as storage and networking in a unified way.

OpenNebula provides a feature called Live Migration, which allows administrators to move running VMs from one host to another with minimal downtime.

Live Migration works by transferring the VM's memory and CPU state from the source host to the destination host. Once the transfer is complete, the VM is resumed on the destination host, and any remaining memory pages are copied over the network.

Steps:

1. Log in to the OpenNebula web interface using your credentials.
2. Select the "Virtual Resources" tab and then select "VM Instances" from the drop-down menu.
3. Locate the VM instance that you want to migrate and select it by clicking on its name.

4. Click on the VM instance name in the top center of the page.



ShareGPT

Powered by ▲Vercel

5. In the "Migrate Virtual Machine" dialog box, select the destination host where you want to migrate the VM.

6. Choose the migration mode.

There are two migration modes:

"live" and "shutdown". If you choose "live", the VM will be migrated without any downtime. If you choose "shutdown", the VM will be stopped on the source host, migrated, and then started on the destination host.

7. Click on the "Migrate" button to initiate the migration process.

8. Monitor the migration progress on the "Virtual Machines" page. The migration progress bar shows the percentage of memory pages that have been transferred to the destination host.

9. Once the migration is complete, the VM will be running on the destination host. You can verify this by checking the VM's status in the "Virtual Machines" page.

Congratulations! You have successfully migrated a VM from one host to another using OpenNebula.



ShareGPT

Powered by ▲Vercel



Theory and steps for "Create a virtual network in a VM using OpenNebula".

Sure, here are the theory and steps to create a virtual network in a VM using OpenNebula:

Theory:

OpenNebula is an open-source cloud computing platform that enables the management of virtualized data centers. It allows administrators to manage virtual machines (VMs) and other resources such as storage and networking in a unified way.

OpenNebula provides a feature called Virtual Networks, which allows administrators to create and manage virtual networks that connect VMs and other virtual resources.

A virtual network in OpenNebula is a logical representation of a network segment that can be used to connect VMs. Each virtual network has its own IP address range, and VMs can be assigned IP addresses from the network's address range.

Steps:



ShareGPT

Powered by ▲Vercel

1. Log in to the OpenNebula web interface using your credentials.
2. Select the "Virtual Resources" tab and then select "Virtual Networks" from the drop-down menu.
3. Click on the "Create" button to create a new virtual network.
4. In the "General" tab of the virtual network creation wizard, enter a name for your virtual network.
5. In the "Address Ranges" tab, enter the IP address range for your virtual network. You can also configure other options such as the gateway address and DNS servers.
6. In the "Context" tab, you can configure context information that will be passed to the VMs that are connected to the virtual network. This can include hostname, IP address, and other configuration options.
7. Click on the "Create" button to create your virtual network.
8. Now, you can connect your VM to the virtual network by editing the VM's net VM that the virtual network from the



ShareGPT

Powered by ▲Vercel

"Virtual Machines" page, and then click on the "Network Interfaces" tab.

9. Click on the "Add" button to add a new network interface to the VM.

10. In the "Network" field, select the virtual network that you created in step 3.

11. Configure other options for the network interface such as the MAC address and IP address.

12. Click on the "Save" button to save the network interface settings.

13. Start the VM to allow it to connect to the virtual network.

Congratulations! You have successfully created a virtual network in a VM using OpenNebula.



Theory and steps for "Creat a VM instance in Google Cloud Platform and provide the related code".

Sure, here are the theory and steps to create a VM instance in Google Cloud Platform (GCP).

Theory:



ShareGPT

Powered by ▲Vercel

Google Cloud Platform (GCP) is a cloud computing platform that provides various cloud services, including compute, storage, and networking. GCP allows users to create virtual machines (VMs) in the cloud, which can be used to run applications and services.

To create a VM instance in GCP, you need to first create a project, select a region, choose a machine type, and configure networking and other options.

Steps:

1. Log in to the GCP Console using your Google account credentials.
2. Create a new project or select an existing project in the GCP Console.
3. Select the project and navigate to the "Compute Engine" section.
4. Click on the "Create Instance" button to create a new VM instance.
5. In the "Create a new instance" dialog box, configure the following settings:

- Name: Enter



ShareGPT

Powered by ▲Vercel

- Region and zone: Select the region and zone where you want to deploy your VM instance.
- Machine type: Select the machine type that you want to use for your VM instance. You can choose from various options, such as CPU, RAM, and storage.
- Boot disk: Select the boot disk image that you want to use for your VM instance. You can choose from various operating systems and versions.
- Networking: Configure networking options for your VM instance, such as IP address, firewall rules, and network tags.
- Management: Configure management options for your VM instance, such as startup scripts and metadata.
 1. Once you have configured all the settings, click on the "Create" button to create your VM instance.

Related code:

Here's an example code snippet that creates a VM instance in GCP using the Google Cloud SDK command-line interface:

sql Copy code

```
gcloud compute instances create e
  --zone us-central1-a \
  --machine-type n1-standard-1
  --image-family debian-10 \
  --ima
  --boot
  --tags http-server,https-serv
```



ShareGPT

Powered by ▲Vercel

```
--metadata startup-script="#!  
sudo apt-get update  
sudo apt-get install apac  
sudo service apache2 star  
"
```

This code creates a VM instance named "example-instance" in the "us-central1-a" zone with a machine type of "n1-standard-1". It uses the Debian 10 boot disk image, with a boot disk size of 10GB. It also tags the instance with "http-server" and "https-server" tags and sets a startup script that installs and starts the Apache web



Theory and steps for "Create a HELLO WORLD web application in Google Application Engine using Eclipse".

Sure, here are the theory and steps to create a "Hello World" web application in Google App Engine using Eclipse:

Theory:

Google App Engine is a platform as a service (PaaS) offering from Google Cloud Platform that allows you to build and deploy web applications in a fully managed, serverless environment. App Engine



ShareGPT

Powered by ▲Vercel

supports multiple programming languages, including Java, Python, PHP, and Go.

Eclipse is an integrated development environment (IDE) that provides tools and features to develop and debug applications in various programming languages. Eclipse has a plugin for Google Cloud Platform that provides integration with App Engine, allowing developers to create and deploy applications to App Engine directly from within Eclipse.

Steps:

1. Download and install Eclipse if you haven't already done so. You can download Eclipse from the Eclipse Foundation website.
2. Install the Google Cloud Tools for Eclipse plugin. To do this, open Eclipse and select Help > Eclipse Marketplace from the menu bar. Search for "Google Cloud Tools for Eclipse" and click on the "Install" button.
3. Create a new App Engine project in Eclipse. To do this, select File > New > Other from the menu bar. Expand t
for Eclips
Engine S



ShareGPT

Powered by ▲Vercel

Click on the "Next" button and follow the prompts to create a new project.

4. Create a new servlet in the project.

To do this, right-click on the project in the Package Explorer view and select New > Servlet from the context menu. Enter a name for the servlet and click on the "Finish" button.

5. Modify the servlet code to output "Hello World". Replace the contents of the doGet() method in the servlet with the following code:

go Copy code

```
response.setContentType("text/pla  
response.getWriter().println("Hel
```

1. Run the application locally to test it. Right-click on the project in the Package Explorer view and select Run As > Web Application from the context menu. Eclipse will start a local development server and deploy the application to it.

2. Deploy the application to App Engine. To do this, right-click on the project in the Package Explorer and select Deploy to



ShareGPT

Powered by ▲Vercel

context menu. Follow the prompts to deploy the application.

3. Visit the application in a web browser. To do this, open a web browser and enter the URL for the deployed application. The URL will be in the format "https://[project-id].appspot.com".

Congratulations! You have successfully created a "Hello World" web application in Google App Engine using Eclipse.



Theory and steps for "Write a program using Tinkercad to glow RGB LED with delay of 2 sec".

Sure, here are the theory and steps to write a program using Tinkercad to glow an RGB LED with a delay of 2 seconds:

Theory:

Tinkercad is an online platform for creating and simulating virtual circuits and electronic projects. It provides a drag-and-drop interface for designing circuits, as w



ShareGPT

Powered by ▲Vercel

programming microcontrollers such as the Arduino.

An RGB LED is a type of LED that contains three separate LEDs, one for each primary color (red, green, and blue). By adjusting the brightness of each of the three LEDs, it is possible to produce a wide range of colors.

A delay in programming is a pause or waiting period in the execution of a program. It is often used to control the timing of events or to create time-based effects.

Steps:

1. Open Tinkercad and create a new circuit. Add an RGB LED to the circuit by selecting it from the components library and dragging it onto the workspace.
2. Connect the anode (positive) pin of the RGB LED to pin 11 of the Arduino board, and connect the cathode (negative) pin to ground.
3. Open the Code Editor by clicking on the "Code" button in the top menu bar.
4. Write the following code to

declare t
and set t



ShareGPT

Powered by ▲Vercel

arduino Copy code

```
int redPin = 11;
int greenPin = 10;
int bluePin = 9;

void setup()
{
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
}
```

1. Write the following code to turn on the red LED for 2 seconds, then turn it off and wait for 2 seconds before repeating the process with the green and blue LEDs:

scss Copy code

```
void loop()
{
  digitalWrite(redPin, HIGH); //
  delay(2000);                //
  digitalWrite(redPin, LOW);   //
  delay(2000);                //

  digitalWrite(greenPin, HIGH); /
  delay(2000);                /
  digitalWrite(greenPin, LOW);  /
  delay(2000);                /

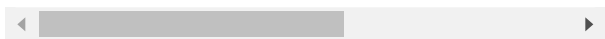
  digitalWrite(bluePin, HIGH); //
  delay(2000);                //
  digitalWrite(bluePin, LOW);   //
  delay(2000);                //
}
```



ShareGPT

Powered by ▲Vercel


```
delay(2000);  
}  
//
```



1. Click on the "Compile" button in the top menu bar to compile the code. If there are any errors, fix them and compile again.
2. Click on the "Start Simulation" button in the top menu bar to simulate the circuit and run the program. The RGB LED should glow with each color for 2 seconds before changing to the next color.

Congratulations! You have



Theory and steps for "Write a program using Tinkercad to interface push button with LED, so that LED blinks when button is pressed".

Sure, here are the theory and steps to write a program using Tinkercad to interface a push button with an LED, so that the LED blinks when the button is pressed:

Theory:

Tinkercad is a web-based platform for creating and simulating electronic circuits and electronic projects. It provides a



ShareGPT

Powered by ▲Vercel

drag-and-drop interface for designing circuits, as well as a code editor for programming microcontrollers such as the Arduino.

A push button is a type of switch that is activated by pressing it. It has two states: pressed (closed) and unpressed (open).

An LED (Light Emitting Diode) is a type of diode that emits light when a current flows through it. It has two terminals: the anode (positive) and the cathode (negative).

Steps:

1. Open Tinkercad and create a new circuit. Add a push button and an LED to the circuit by selecting them from the components library and dragging them onto the workspace.
2. Connect one leg of the push button to a digital pin on the Arduino board (e.g. pin 2), and connect the other leg to ground. Connect the anode (positive) leg of the LED to a digital pin on the Arduino board (e.g. pin 3), and connect the cathode (negative) leg to ground through a resistor.



ShareGPT

Powered by ▲Vercel

3. Open the Code Editor by clicking on the "Code" button in the top menu bar.

4. Write the following code to declare the pins for the push button and the LED and set them to input and output mode, respectively:

arduino Copy code

```
const int buttonPin = 2;
const int ledPin = 3;

void setup() {
  pinMode(buttonPin, INPUT);
  pinMode(ledPin, OUTPUT);
}
```

1. Write the following code to check if the button is pressed. If it is, turn on the LED for a short period of time (e.g. 500 milliseconds) and then turn it off again:

scss Copy code

```
void loop() {
  if (digitalRead(buttonPin) == H
    digitalWrite(ledPin, HIGH);
    delay(500);
    digitalWrite(ledPin, LOW);
  }
}
```



ShareGPT

Powered by ▲Vercel

1. Click on the "Compile" button in the top menu bar to compile the code. If there are any errors, fix them and compile again.
2. Click on the "Start Simulation" button in the top menu bar to simulate the circuit and run the program. The LED should blink when the button is pressed.

Congratulations! You have successfully written a program using Tinkercad to interface a push button with an LED, so that the LED blinks

1 2 3 4 5 6 7 8 9 10 11 12



Theory and steps for "Write a program using Tinkercad to interface buzzer with Arduino board to buzz on/off with delay of 1 second".

Sure, here are the theory and steps to write a program using Tinkercad to interface a buzzer with an Arduino board to buzz on/off with a delay of 1 second:

Theory:

Tinkercad is an online platform for creating and simulating electronic circuits and electronic components. It is a drag-and-drop



ShareGPT

Powered by ▲Vercel

circuits, as well as a code editor for programming microcontrollers such as the Arduino.

A buzzer is an electronic device that produces sound when a current flows through it. It is often used in electronic projects to provide audible feedback to the user.

The Arduino board is a microcontroller board that allows you to control electronic components and devices through programming.

Steps:

1. Open Tinkercad and create a new circuit. Add an Arduino board and a buzzer to the circuit by selecting them from the components library and dragging them onto the workspace.
2. Connect one leg of the buzzer to a digital pin on the Arduino board (e.g. pin 2), and connect the other leg to ground.
3. Open the Code Editor by clicking on the "Code" button in the top menu bar.
4. Write the following code to

declare t

set it to c



ShareGPT

Powered by ▲Vercel

arduino Copy code

```
const int buzzerPin = 2;

void setup() {
  pinMode(buzzerPin, OUTPUT);
}
```

1. Write the following code to turn the buzzer on and off with a delay of 1 second between each state:

scss Copy code

```
void loop() {
  digitalWrite(buzzerPin, HIGH);
  delay(1000);
  digitalWrite(buzzerPin, LOW);
  delay(1000);
}
```

1. Click on the "Compile" button in the top menu bar to compile the code. If there are any errors, fix them and compile again.
2. Click on the "Start Simulation" button in the top menu bar to simulate the circuit and run the program. The buzzer should buzz on and off with a delay of 1 second between each state.

Congratulations
successfully
Tinkercad to



ShareGPT

Powered by ▲Vercel

an Arduino board to buzz on/off with



Theory and steps for "Write a program using Tinkercad to interface LED and buzzer with Arduino board so that the buzzer is put on whenever LED is on, the buzzer is put off whenever LED is off".

Sure, here are the theory and steps to write a program using Tinkercad to interface an LED and a buzzer with an Arduino board so that the buzzer is turned on whenever the LED is on, and the buzzer is turned off whenever the LED is off:

Theory:

Tinkercad is an online platform for creating and simulating virtual circuits and electronic projects. It provides a drag-and-drop interface for designing circuits, as well as a code editor for programming microcontrollers such as the Arduino.

An LED is an electronic component that emits light when a current flows through it. It is often used in electronic projects to provide visual feedback to



ShareGPT

Powered by ▲Vercel

A buzzer is an electronic device that produces sound when a current flows through it. It is often used in electronic projects to provide audible feedback to the user.

The Arduino board is a microcontroller board that allows you to control electronic components and devices through programming.

Steps:

1. Open Tinkercad and create a new circuit. Add an Arduino board, an LED, and a buzzer to the circuit by selecting them from the components library and dragging them onto the workspace.
2. Connect the positive leg of the LED to a digital pin on the Arduino board (e.g. pin 2), and connect the negative leg to ground. Connect one leg of the buzzer to another digital pin on the Arduino board (e.g. pin 3), and connect the other leg to ground.
3. Open the Code Editor by clicking on the "Code" button in the top menu bar.

4. Write the following code to control the buzzer:

```
const int buzzerPin = 3;  
void setup() {  
  pinMode(buzzerPin, OUTPUT);  
}  
void loop() {  
  digitalWrite(buzzerPin, HIGH);  
  delay(1000);  
  digitalWrite(buzzerPin, LOW);  
  delay(1000);  
}
```



ShareGPT

Powered by ▲Vercel

the buzzer and set them to output mode:

arduino Copy code

```
const int ledPin = 2;
const int buzzerPin = 3;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
}
```

1. Write the following code to turn the LED and the buzzer on and off:

scss Copy code

```
void loop() {
  digitalWrite(ledPin, HIGH);
  digitalWrite(buzzerPin, HIGH);
  delay(1000);
  digitalWrite(ledPin, LOW);
  digitalWrite(buzzerPin, LOW);
  delay(1000);
}
```

1. Click on the "Compile" button in the top menu bar to compile the code. If there are any errors, fix them and compile again.

2. Click on the "Start Simulation" button in the top menu bar to simulate program should turn on and on together.



ShareGPT

Powered by ▲Vercel

Congratulations! You have successfully written a program using Tinkercad to interface an LED and a buzzer with an Arduino board so that the buzzer is turned on whenever the



Theory and steps for "Configure a MQTT broker to communicate with your clients and create one connection to access the broker using Paho client".

MQTT (Message Queuing Telemetry Transport) is a lightweight messaging protocol that is used for IoT (Internet of Things) devices to communicate with each other and with servers. A MQTT broker is a server that acts as a central hub for all MQTT messages and is responsible for routing messages between clients.

Here are the steps to configure a MQTT broker to communicate with your clients and create one connection to access the broker using Paho client:

Theory:

There are various MQTT brokers available such as RabbitMQ, a we will use M



ShareGPT

Powered by ▲Vercel

Paho client is an open-source MQTT client library that is used to connect to MQTT brokers and send and receive messages. It is available in various programming languages such as Python, Java, C, and so on.

Steps:

1. Install Mosquitto MQTT broker by following the installation instructions from their official website.
2. Once the installation is complete, start the Mosquitto broker service by running the following command on the terminal:

sql Copy code

```
sudo service mosquitto start
```

1. Verify that the Mosquitto broker service is running by running the following command on the terminal:

lua Copy code

```
sudo service mosquitto status
```

1. Install Paho MQTT client library for Python by running the following command on the terminal:

Copy code

```
pip install
```



ShareGPT

Powered by ▲Vercel

1. Open a new Python file and import the Paho MQTT client library by adding the following line of code:

python Copy code

```
import paho.mqtt.client as mqtt
```

1. Define a callback function that will be triggered when the client connects to the MQTT broker:

python Copy code

```
def on_connect(client, userdata,
               print("Connected with result
```

1. Create a new MQTT client object and set the callback function:

python Copy code

```
client = mqtt.Client()
client.on_connect = on_connect
```

1. Connect the client to the MQTT broker by calling the connect() method:


python Copy code

```
client.connect("localhost", 1883,
```

Note: Replace "localhost" with the IP address of the machine running the MQTT broker if you are connecting to the broker from



ShareGPT

Powered by  Vercel

1. Start the client loop to listen for incoming messages by calling the `loop_start()` method:

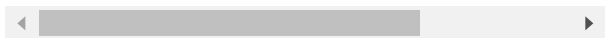
python Copy code

```
client.loop_start()
```

1. Publish a message to the MQTT broker by calling the `publish()` method:

python Copy code

```
client.publish("test/topic", "Hel
```



Note: "test/topic" is the topic to which the message will be published. You can replace it with any topic name of your choice.

1. Subscribe to a topic on the MQTT broker by calling the `subscribe()` method:

python Copy code

```
client.subscribe("test/topic")
```


1. Define a callback function that will be triggered when a message is received on the subscribed topic:

python Copy code

```
def on_message(client, userdata, msg):  
    print(msg.topic+" "+str(msg.payload))
```



ShareGPT

Powered by  Vercel

1. Set the callback function for the client by adding the following line of code:

python Copy code

```
client.on_message = on_message
```

1. Disconnect the client from the MQTT broker by calling the disconnect() method:

python Copy code

```
client.disconnect()
```

1. Run the Python program and verify that the message is published to the MQTT broker and received by the subscribed client.

Congratulations! You have successfully configured a MQTT broker to communicate with your clients and created one connection to



Theory and steps for "Write a program using Tinkercad to interface distance sensor and buzzer with Arduino board so that the buzzer is put on whenever distance sensor is on".

In this project, we will interface a distance sensor and a buzzer with an Arduino board so that the buzzer is put on whenever the distance sensor detects an object. In Tinkercad, ar



ShareGPT

Powered by ▲Vercel

platform, to simulate the circuit and write the program. Here are the steps to follow:

Theory:

The distance sensor (HC-SR04) sends out ultrasonic waves and measures the time it takes for the waves to bounce back from an object. The Arduino board then calculates the distance based on the time taken and triggers the buzzer if the distance is below a certain threshold.

Steps:

1. Open Tinkercad and create a new circuit.
2. Search for "Arduino UNO" and add it to the workspace.
3. Add a "Breadboard" to the workspace and connect it to the Arduino board.
4. Add the distance sensor to the breadboard.
5. Connect the VCC pin of the distance sensor to the 5V pin of the Arduino board.
6. Connect distance the Ardu



ShareGPT

Powered by ▲Vercel

7. Connect the TRIG pin of the distance sensor to pin 10 of the Arduino board.
8. Connect the ECHO pin of the distance sensor to pin 11 of the Arduino board.
9. Add the buzzer to the breadboard.
10. Connect the positive (+) pin of the buzzer to pin 12 of the Arduino board.
11. Connect the negative (-) pin of the buzzer to the GND pin of the Arduino board.
12. Write the following program in the Arduino IDE or in the Tinkercad code editor:

c++ Copy code

```
const int trigPin = 10;
const int echoPin = 11;
const int buzzerPin = 12;

void setup() {
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(buzzerPin, OUTPUT);
}
```

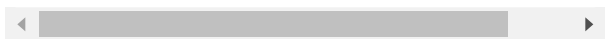
```
void loop() {
  long du
```



ShareGPT

Powered by ▲Vercel


```
digitalWrite(trigPin, LOW);  
delayMicroseconds(2);  
digitalWrite(trigPin, HIGH);  
delayMicroseconds(10);  
digitalWrite(trigPin, LOW);  
duration = pulseIn(echoPin, HIGH);  
distance = duration / 58.2;  
Serial.println(distance);  
if (distance <= 20) {  
    digitalWrite(buzzerPin, HIGH);  
} else {  
    digitalWrite(buzzerPin, LOW);  
}  
delay(200);  
}
```




1. Verify and upload the program to the Arduino board.
2. Run the simulation and test the circuit by moving an object close to the distance sensor. The buzzer should buzz if the distance is below 20cm.

Congratulations! You have successfully written a program using Tinkercad to interface a distance sensor and a buzzer with an Arduino board so that the buzzer is put on



ShareGPT

Powered by  Vercel



Write a program using Tinkercad to interface IR sensor and buzzer with Arduino board.

In this project, we will interface an IR sensor and a buzzer with an Arduino board. The buzzer will buzz whenever the IR sensor detects an object. We will use Tinkercad, an online simulation platform, to simulate the circuit and write the program. Here are the steps to follow:

Theory:

The IR sensor detects infrared radiation emitted by an object. When an object is detected, the sensor sends a signal to the Arduino board, which then triggers the buzzer.

Steps:

1. Open Tinkercad and create a new circuit.
2. Search for "Arduino UNO" and add it to the workspace.
3. Add a "Breadboard" to the workspace and connect it to the Arduino board.

4. Add the
breadboard...



ShareGPT

Powered by ▲Vercel

5. Connect the VCC pin of the IR sensor to the 5V pin of the Arduino board.
6. Connect the GND pin of the IR sensor to the GND pin of the Arduino board.
7. Connect the OUT pin of the IR sensor to pin 10 of the Arduino board.
8. Add the buzzer to the breadboard.
9. Connect the positive (+) pin of the buzzer to pin 11 of the Arduino board.
10. Connect the negative (-) pin of the buzzer to the GND pin of the Arduino board.
11. Write the following program in the Arduino IDE or in the Tinkercad code editor:

c++ Copy code

```
const int irSensorPin = 10;
const int buzzerPin = 11;

void setup() {
  Serial.begin(9600);
  pinMode(irSensorPin, INPUT);
  pinMode(buzzerPin, OUTPUT);
}

void loop() {
```




ShareGPT

Powered by ▲Vercel

```
int sensorValue = digitalRead(i
Serial.println(sensorValue);
if (sensorValue == HIGH) {
    digitalWrite(buzzerPin, HIGH)
} else {
    digitalWrite(buzzerPin, LOW);
}
delay(100);
}
```

1. Verify and upload the program to the Arduino board.
2. Run the simulation and test the circuit by moving an object close to the IR sensor. The buzzer should buzz when an object is detected.

Congratulations! You have successfully written a program using Tinkercad to interface an IR sensor and a buzzer with an Arduino board.

**ShareGPT**Powered by  Vercel