1. Open **Unity Hub** and create a **new 2D project**.
2. **Create the Maze Layout**:
   * In the **Hierarchy**, right-click → **2D Object** → **Tilemap → Rectangular**.
   * Add a **Grid** and **Tilemap** (for walls and floor).
   * Import **tile sprites** (or create simple square sprites).
   * Use the **Tile Palette** to draw the maze.

Refer this Video for Tile palette

1. **Add Collisions to Walls**:
   * Select **Tilemap** → **Add Component** → **Tilemap Collider 2D**.
   * Add a **Rigidbody 2D** (set to **Kinem** **atic**) to prevent movement.
   * Optionally, use a **Composite Collider 2D** for optimized collision.
2. **Add the Player Sprite**:
   1. Right-click in **Hierarchy** → **2D Object → Sprite**.
   2. Name it **Player** and assign a sprite (or use a simple circle).
   3. Add a **Rigidbody 2D** (set Gravity Scale to **0**).
   4. Add a **Box Collider 2D** for collision.
3. **Write Player Movement Script**:
   1. Create a **C# script** (e.g., PlayerController.cs).
   2. Attach it to the Player GameObject.
   3. Add movement logic using Rigidbody2D.

*using UnityEngine;*

*public class PlayerController : MonoBehaviour*

*{*

*public float moveSpeed = 5f;*

*private Rigidbody2D rb;*

*private Vector2 movement;*

*void Start()*

*{*

*rb = GetComponent<Rigidbody2D>();*

*}*

*void Update()*

*{*

*movement.x = Input.GetAxisRaw("Horizontal");*

*movement.y = Input.GetAxisRaw("Vertical");*

*}*

*void FixedUpdate()*

*{*

*rb.velocity = movement.normalized \* moveSpeed;*

*}*

*}*

1. **Add Enemy Sprite**:

* Right-click **Hierarchy** → **2D Object → Sprite**.
* Name it **Enemy** and assign a sprite.
* Add a **Rigidbody 2D** (set Gravity Scale to **0**).
* Add a **Box Collider 2D**.

1. **Write Enemy Patrol Script**:

* Create a **C# script** (e.g., EnemyPatrol.cs).
* Attach it to the **Enemy** GameObject.
* Make the enemy move between two points.

using UnityEngine;

public class EnemyPatrol : MonoBehaviour

{

public Transform pointA;

public Transform pointB;

public float speed = 2f;

private Vector3 target;

void Start()

{

target = pointB.position;

}

void Update()

{

transform.position = Vector3.MoveTowards(transform.position, target, speed \* Time.deltaTime);

if (Vector3.Distance(transform.position, pointA.position) < 0.1f)

target = pointB.position;

else if (Vector3.Distance(transform.position, pointB.position) < 0.1f)

target = pointA.position;

}

}

* Create **two empty GameObjects** (PointA and PointB) and position them.
* Assign them to the **EnemyPatrol** script in the Inspector.

1. **Add Goal Sprite**:
   * Right-click **Hierarchy** → **2D Object → Sprite**.
   * Name it **Goal** and assign a cup sprite.
   * Add a **Box Collider 2D** (set to **Is Trigger**).
2. **Write Goal Script**:
   * Create a **C# script** (e.g., Goal.cs).
   * Attach it to the **Goal** GameObject.

*using UnityEngine;*

*using UnityEngine.SceneManagement;*

*public class Goal : MonoBehaviour*

*{*

*void OnTriggerEnter2D(Collider2D other)*

*{*

*if (other.CompareTag("Player"))*

*{*

*Debug.Log("You Win!");*

*// Load next level or show victory screen*

*SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);*

*}*

*}*

*}*

 Tag the **Player** as "Player" (Inspector → **Tag** → **Player**).

 This script reloads the level when the player reaches the goal.

1. Handle Enemy Collision
   * Open **EnemyPatrol.cs** and add collision logic:

*void OnTriggerEnter2D(Collider2D other)*

*{*

*if (other.CompareTag("Player"))*

*{*

*Debug.Log("Game Over!");*

*SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);*

*}*

*}*

1. **Set Enemy Collider**:

* Make sure the **Enemy** has a **Box Collider 2D**.
* Set it as **Is Trigger**.

1. Create a **UI Canvas** (Right-click **Hierarchy** → **UI** → **Canvas**).
   * Add **Text** elements:
   1. One for **"You Win!"** (disable it initially).
   2. One for **"Game Over!"** (disable it initially).

Modify the Goal.cs and EnemyPatrol.cs scripts to activate UI elements instead of reloading scenes