C-SHARP LIBRARY FOR UNITY™

Collected and created by René Pol -2021-

Index.

AI follow/Chase the player 2D. *https://youtu.be/0rcFeheNQOw*

Give the gameObject that needs to follow the player this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class AIfollow : MonoBehaviour
{
public float Speed; private
Transform target; public
float StopChase;
void Start()
{
target = GameObject.FindGameObjectWithTag("Player").GetComponent<Transform>();
}
void Update()
{
if (Vector2.Distance(transform.position, target.position) > StopChase)
{
transform.position = Vector2.MoveTowards(transform.position, target.position, Speed * Time.deltaTime);
}
}
}
```

You can set the speed of moving in the inspector's view. The stop chasing distance number can be set in this view to. Make sure the object to follow has the tag Player or change the tag name into the tag you are using.

AI looks at player and shoots when player gets in range 2D.

https://youtu.be/3lBt8dDbRfI

Create the object that needs to look at the Transform player. Give it a Rigidbody2D with 0 gravity and the Z restraint selected. Give it this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class LookatandAttack : MonoBehaviour
{
public GameObject Rockets;
public Transform barrel; public
Transform player; private
Rigidbody2D rb; public float
Ammo;
public float ShootingDistance;



void Start()
{
rb = this.GetComponent<Rigidbody2D>();
}
IEnumerator Reload()
{
yield return new WaitForSeconds(1f);
Ammo = 1;
}
public void shootRocket()
{
if (Ammo > 0)
{
Ammo -= 1;
Instantiate(Rockets, barrel.position, barrel.rotation);
StartCoroutine(Reload()); return;
}
}

void Update()
{
Vector3 direction = player.position - transform.position; float angle
= Mathf.Atan2(direction.y, direction.x) * Mathf.Rad2Deg;
rb.rotation = angle;
if (Vector2.Distance(transform.position, player.position) < ShootingDistance)
{
{
shootRocket();
}
}
else
{
Ammo = 1;
}
}
}
```

Give this Object a child object and call it barrel. You can set the distance to the player where it should start to fire in. Make sure the rocket model is facing right and has a fly to the right direction action to it. Select the barrel as point where the rocket comes from.

AI patroll follow path with path points (options: looping, returning, destroy at end path) 2D

https://youtu.be/8MLkOXjnH5U

Give the game Object a Rigidbody2D and a Box Collider 2D. Make sure the gravity is set to 0. Give the game Object that needs to follow a path this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class Followpath : MonoBehaviour
{
public List<Transform> pathpoints;
public float moveSpeed = 2f;
private int pathsIndex = 0; void
Start()
{
transform.position = pathpoints[pathsIndex].transform.position;
}
void Update()
{
Move();
}
void Move()
{
transform.position = Vector2.MoveTowards(transform.position,
pathpoints[pathsIndex].transform.position, moveSpeed * Time.deltaTime); if
(transform.position == pathpoints[pathsIndex].transform.position)
{
pathsIndex += 1;
}
if (pathsIndex == pathpoints.Count)
{
pathsIndex = 0;
}
```

```
}
}
```

Create an empty game object and call it pathpoint. Duplicate as many as you need and create a path with them. Now set the number of pathpoints and drag them into the inspector's view. Set the moving speed. Make sure the Z-axis on the pathpoints are the same as the game object that has to follow them. Otherwise the object will get stuck at the first point.

To make the object disappear at the end of the path just insert one line in the script.

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class Followpath : MonoBehaviour
{
public List<Transform> pathpoints;
public float moveSpeed = 2f;
private int pathsIndex = 0; void
Start()
{
transform.position = pathpoints[pathsIndex].transform.position;
}
void Update()
{
Move();
}
void Move()
{
transform.position = Vector2.MoveTowards(transform.position,
pathpoints[pathsIndex].transform.position, moveSpeed * Time.deltaTime); if
(transform.position == pathpoints[pathsIndex].transform.position)
{
pathsIndex += 1;
}
```

```
if (pathsIndex == pathpoints.Count)
                                    {
pathsIndex = 0;
Destroy(this.gameObject)
;                                   }
                                    }
                                    }
```

If you want the object to return over the path the same way as it came we need again to insert one new line :

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class Followpath : MonoBehaviour
{
public List<Transform> pathpoints;
public float moveSpeed = 2f; private
int pathsIndex = 0;
```

```
void Start()
{
transform.position = pathpoints[pathsIndex].transform.position;
}
void Update()
{
Move();
}
void Move()
{
transform.position = Vector2.MoveTowards(transform.position,
pathpoints[pathsIndex].transform.position, moveSpeed * Time.deltaTime); if
(transform.position == pathpoints[pathsIndex].transform.position)
{
pathsIndex += 1;
                                    }
if (pathsIndex == pathpoints.Count)
                                    {
pathsIndex = 0;
//Destroy(this.gameObject);///
pathpoints.Reverse()
;                                   }
                                    }
                                    }
```

AI patrolling around colliders 2D

https://youtu.be/7BSkqJnz_8o

Give the gameObject 2 empty child objects for ground detection and wall detection. Give it a rigidbody2D and set it to kinematic. Add this script and drag these detection objects to the inspector window. Set speed and cast rays in the inspector window.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Crawlers : MonoBehaviour
{
private Rigidbody2D rb;
[SerializeField]
public float speed;
public bool hasTurned;
[SerializeField]
private float ZAxisAdd;
public float fallTime;

[Header("Ground Detection")]
public bool groundDetected;
[SerializeField] Transform groundPos;
public float groundCheckSize;
[SerializeField]
public LayerMask whatIsGround;

[Header("Wall Detection")]
public bool wallDetected;
[SerializeField] Transform wallPos;
public float wallCheckSize;

void Start()
{
rb = GetComponent<Rigidbody2D>();
}

void Update()
{
Environment();
}

void FixedUpdate()
{
Movement();
}

void Environment()
{
groundDetected = Physics2D.Raycast(groundPos.position, -transform.up,
groundCheckSize, whatIsGround);
wallDetected = Physics2D.Raycast(wallPos.position, transform.right, wallCheckSize,
whatIsGround);
```

```
if (!groundDetected)
{
if (hasTurned == false)
{
ZAxisAdd -= 90;
transform.eulerAngles = new Vector3(0, 0, ZAxisAdd);
hasTurned = true;
}
fallTime -= Time.deltaTime;
}
if (groundDetected)
{
hasTurned = false;
fallTime = 1;
}
if (wallDetected)
{
if (!hasTurned)
{
ZAxisAdd += 90;
transform.eulerAngles = new Vector3(0, 0, ZAxisAdd);
}
}
if (fallTime == 1)
{
rb.gravityScale = 0;
speed = 3;
}
else if (fallTime <= 0)
{
transform.eulerAngles = new Vector3(0, 0, 0);
ZAxisAdd = 0;
rb.gravityScale = 50;
speed = 0;
}
if (ZAxisAdd <= -360)
{
ZAxisAdd = 0;
}
}

void Movement()
{
rb.velocity = transform.right * speed;
}

private void OnDrawGizmos()
{
Gizmos.DrawLine(groundPos.position, new Vector2(groundPos.position.x,
groundPos.position.y - groundCheckSize));
Gizmos.DrawLine(wallPos.position, new Vector2(wallPos.position.x + wallCheckSize,
wallPos.position.y));
}
}
```

AI patrolling to the edge of a platforms and changes direction using Raycast

https://youtu.be/mx_XTuLTgQw

Create an empty game object for the object your using and call it groundcheck. Give it an icon and place it next to the object's feet. Now give the parent object this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class PatrolPlatform : MonoBehaviour
{
public float Speed; public bool
MovingRight = true; public
Transform Checkground; void
Update()
{
transform.Translate(Vector2.right * Speed * Time.deltaTime);
RaycastHit2D edgeinfo = Physics2D.Raycast(Checkground.position, Vector2.down, 2f);
if(edgeinfo.collider == false)
{
if(MovingRight == true)
{
transform.eulerAngles = new Vector3(0, -180, 0);
MovingRight = false;
}


else
{
transform.eulerAngles = new Vector3(0, 0, 0);
MovingRight = true;
}
}
}
}
```

Set the speed in the inspector's view. If the raycat of the Checkground transform does not hit anything the game object will turn to the opposite side. Drag the empty game object to the Inspector's view.

AI random patrolling 2D

https://youtu.be/TBr_8J4S2kA

Give the objects that need to patrol this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class RandomPatrol : MonoBehaviour
{
public float Speed; public
float Edgebottom; public
float Edgetop; public
float Edgeleft; public
float Edgeright; Vector2
targetPosition;
void Start()
{
targetPosition = GetRandomPosition();
}
void Update()
{
if ((Vector2)transform.position != targetPosition)
{
transform.position = Vector2.MoveTowards(transform.position, targetPosition, Speed *
Time.deltaTime);
}
else
{
targetPosition = GetRandomPosition();
}
}
Vector2 GetRandomPosition()
{
float randomX = Random.Range(Edgeleft, Edgeright); float
randomY = Random.Range(Edgetop, Edgebottom); return
new Vector2(randomX, randomY);
}
}
```

The object will pick a spot to patroll to withing the given edges (Borders). Once it reaches the spot it will repeat this action.

Airplane movement 2D.

https://youtu.be/8tSO_R0_WHM

Make sure the plane has the tag Player and that the main camera uses the camera follow script from this library. Give the plane a box collider 2D and a Rigidbody2D with its gravity set to zero. Add this script :

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class Airplane : MonoBehaviour
{
Rigidbody2D rb; public float
MoveSpeed; public float
Acceleration; public float
RotationControl; float
MovingY, MovingX = 1;
void Start()
{
rb = GetComponent<Rigidbody2D>();
}
void Update()
{
MovingY = Input.GetAxis("Vertical");
}
private void FixedUpdate()
{
Vector2 Vel = transform.right * (MovingX * Acceleration); rb.AddForce(Vel);
float Dir = Vector2.Dot(rb.velocity, rb.GetRelativeVector(Vector2.right)); if
(Acceleration > 0)
{
if (Dir > 0)
{
rb.rotation += MovingY * RotationControl * (rb.velocity.magnitude / MoveSpeed);
}
```

```
else
{
rb.rotation -= MovingY * RotationControl * (rb.velocity.magnitude / MoveSpeed);
}
}
float thrustForce = Vector2.Dot(rb.velocity, rb.GetRelativeVector(Vector2.down)) * 2.0f;
Vector2 relForce = Vector2.up * thrustForce; rb.AddForce(rb.GetRelativeVector(relForce));
if (rb.velocity.magnitude > MoveSpeed)
{
rb.velocity = rb.velocity.normalized * MoveSpeed;
}
```

```
}
}
```

You can set speed,acceleration and the rotation speed in the Inspector's view.

All about health. Health in numbers,Healthbars, Health pickups with use of triggers 2D

https://youtu.be/N-feJQWDHPQ

You need to create 2 UI elements. Let's begin with the text element. Create a text UI element and call its Health. Give it this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine; using
UnityEngine.UI;
public class Health: MonoBehaviour
{
public static int healthValue = 25; public
Text healthpoints;
void Start()
{
healthpoints = GetComponent<Text>();
}
void Update()
{
healthpoints.text = healthValue.ToString("Health:" + "000");
}
}
```

Give the object that needs to give health this script :

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class Healthpotion : MonoBehaviour
{
void OnTriggerEnter2D(Collider2D other)
{
if (other.gameObject.tag == ("Player"))
{
Health.healthValue += 25;
Destroy(this.gameObject);
```

```
}
}
}
```

Make sure that the script for the UI element holds the line using UnityEngine.UI; otherwise it won't work. The player should have the tag Player and at least one of the two objects (Player and Pickup) needs to have a rigidbody2D.
Next we can use a Healthbar image. Create an image UI element and give it this script :

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine; using
UnityEngine.UI;
public class Healthbar : MonoBehaviour
{
public Image HealthBar; public
float CurrentHealth; public
float Maxhealth = 100f;
void Start()
{
HealthBar = GetComponent<Image>();
}
void Update()
{
HealthBar.fillAmount = CurrentHealth / Maxhealth;
}
}
```

Make sure that the script for the UI element holds the line using UnityEngine.UI; otherwise it won't work. Give the object that needs to give health this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class Healthpotion2 : MonoBehaviour
{
private Healthbar HB;
void OnTriggerEnter2D(Collider2D other)
{
```


```
if (other.gameObject.tag == ("Player"))
{
HB.CurrentHealth += 25;
Destroy(this.gameObject);
}
}
void Start()
{
HB = FindObjectOfType<Healthbar>();
}
}
```

The player should have the tag Player and at least one of the two objects (Player and Pickup) needs to have a rigidbody2D.

Camera follows player with use of offset 2D

https://youtu.be/cyZ-hKTzPJA

Give the main camera this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;

public class CameraFollow : MonoBehaviour
{
public float xMin; public float
xMax; public float yMin; public
float yMax; private
GameObject ThePlayer; void
Start()
{
ThePlayer = GameObject.FindGameObjectWithTag("Player");
}
void LateUpdate()
{
float x = Mathf.Clamp(ThePlayer.transform.position.x, xMin, xMax); float
y = Mathf.Clamp(ThePlayer.transform.position.y, yMin, yMax);
gameObject.transform.position = new Vector3(x, y,gameObject.transform.position.z);
}
}
```

Make sure the player has a tag called Player. At start the camera will search for the player and focus on it. You can change the offset settings in the inspector's window so the camera knows where the minimum and maximum edges are while following the player.

Change Camera Background colors with script 2D

https://www.youtube.com/watch?v=Fk8FHK4vg2U

In the example you are able to fade between 2 colors you pick. Make sure the Main camera is dragged into the Inspectors view.

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class CameraColor : MonoBehaviour
{
public Color color1 = Color.yellow;
public Color color2 = Color.blue;
public float duration = 3.0F; public
Camera cam;
void Update()
{
float t = Mathf.PingPong(Time.time, duration) / duration; cam.backgroundColor
= Color.Lerp(color1, color2, t);
}
```

If you want to set a color by using a variable you can use this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine; public class CameraColor :
MonoBehaviour
{
public Color color1 = Color.yellow;
public Color color2 = Color.blue;
public Color color3 = Color.green;
public Camera cam; public int
PickColor;



void Update()
{
if (PickColor == 1)
{
cam.backgroundColor = color1;
}
if (PickColor == 2)
{
cam.backgroundColor = color2;
}
if (PickColor == 3)
{
cam.backgroundColor = color3;
}
```

```
}
}
```

Countdown timer with speed settings

https://youtu.be/fPg_zJ-nnIA

Create an UI text element and call it Countdown timer. Give it this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine; using
UnityEngine.UI;
public class Countdown : MonoBehaviour
{
public float CountdownTime;
public float Countsetting = 1f;
public bool RunFaster = false;
public Text time; void Start()
{
time = GetComponent<Text>();
}
void Update()
{
Countsetting -= Time.deltaTime;
if(Countsetting <= 0 && CountdownTime >0)
{
CountdownTime -= 1; if
(RunFaster == false)
{
Countsetting = 1f;
}
```

```
if (RunFaster == true)
{
Countsetting = 0.5f;
}
}
time.text = CountdownTime.ToString("00");
}
}
```

Set the start time in the inspectors view. When the bool runfaster is true it takes 1 of the counter each 1 second, If not it will take 1 of the counter each half second. You can use different speeds

and instead of a Bool variable you could use an int with different speed setting numbers. Drag the UI text to the inspector's view.

Doodle jumping (Continuously jumping on collisions) 2D

https://youtu.be/OPsoy-05quE

Make sure the platform has a tag Platform and a box collider 2D. Give the jumping object a boxcollider near its feet and a rigidbody2D with the Z constraint selected. Give the jumping object this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class DoodleJump : MonoBehaviour
{
public float JumpForce = 500;
public float movement = 0f; public
float movementSpeed = 10f;
private Rigidbody2D rb;
void Start()
{
rb = GetComponent<Rigidbody2D>();
}
public void OnCollisionEnter2D(Collision2D other)
{
if (other.gameObject.tag == ("Platform") && rb.velocity.y <= 0)
{
rb.AddForce(Vector2.up * JumpForce);
}
}
void Update()
{
movement = Input.GetAxis("Horizontal") *movementSpeed;
}
```

```
void FixedUpdate()
{
Vector2 velocity = rb.velocity; velocity.x
= movement;
rb.velocity = velocity;
}
}
```

You can set the movement speed and the jump force in the inspector's view. In order to jump through a platform you have to give it a platform effector2D and make sure that the box used by effector is selected.

Endless bouncing ball

https://youtu.be/UYCnGQnDlSc

The ball has a circle collider2D and a Rigidbody2D with gravity set to 0 and the restraint z box selected. The ball also gets a 2D Physic material called bouncy. It has bouncing on 1 (Continously) and friction to 0 (No friction). Then give the object this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class BouncingBall : MonoBehaviour
{
Rigidbody2D rb; Vector3
LastVelocity; private float
speed = 300;
void Start()
{
rb = GetComponent<Rigidbody2D>();
rb.AddForce(new Vector2(20 * Time.deltaTime * speed, 20 * Time.deltaTime *speed));
}
public void OnCollisionEnter2D(Collision2D collision)
{
var speed = LastVelocity.magnitude;
var direction = Vector3.Reflect(LastVelocity.normalized,
collision.contacts[0].normal); rb.velocity = direction *
Mathf.Max(speed, 0f);
}
void Update()
{
LastVelocity = rb.velocity;
}
}
```

Every time the ball collides it uses Reflect to bounce off(ricochet) with real physics.

Fade in and out game Objects

https://youtu.be/sAJmA7eckKo

Place a gameObject and give it this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class Fading : MonoBehaviour
{
public bool fadeoutin = true; public
float fadeSpeed; IEnumerator
FadeInObject()
{
while (this.GetComponent<Renderer>().material.color.a < 1) { Color objectColor =
this.GetComponent<Renderer>().material.color; float fadeAmount =
objectColor.a + (fadeSpeed * Time.deltaTime); objectColor = new
Color(objectColor.r, objectColor.g, objectColor.b, fadeAmount);
this.GetComponent<Renderer>().material.color = objectColor; yield return null;
}
}
IEnumerator FadeOutObject()
{
while (this.GetComponent<Renderer>().material.color.a > 0)
{
Color objectColor = this.GetComponent<Renderer>().material.color; float
fadeAmount = objectColor.a - (fadeSpeed * Time.deltaTime); objectColor = new
Color(objectColor.r, objectColor.g, objectColor.b, fadeAmount);
this.GetComponent<Renderer>().material.color = objectColor; yield return null;
}
}

public void Update()
{
if (Input.GetKeyDown(KeyCode.Q) && fadeoutin == true)
{
fadeoutin = false;
StartCoroutine(FadeOutObject());
}
if (Input.GetKeyDown(KeyCode.W) && fadeoutin == false)
{
fadeoutin = true;
StartCoroutine(FadeInObject());
}
}
}
```

In this example 2 IE numerators are created for fade in and out. Depending on the bool variable setting it gets activated. The bool Can be changed by using the keys Q (for fade out) and W (for fade in). You can use these different ways. You can set the fade speed in the inspector's view.

Finding all gameObjects with the same tag and destroy them

https://youtu.be/cjMP679cY54

Place a few game Objects with the name tag Enemy. Create a new empty game object and call it EnemyDestroyer. Give it this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class EnemyDestroyer : MonoBehaviour
{
GameObject[] Enemies; void
Start()
{
Enemies = GameObject.FindGameObjectsWithTag("Enemy");
}
void Update()
{
if(Input.GetKeyDown(KeyCode.Space))
{
foreach (GameObject A in Enemies)
{
Destroy(A.gameObject);
}
}
}
}
```

We define a group that holds all tags with the name Enemy and call it Enemies. At start the group will check for all its members (The objects with the tag Enemy) For every object in this group we define a name (In this example A but it can be anything) and destroy all these A game objects with the press of the space bar.
This comes in handy when you need all enemies removed if you cleared a level.

Flappy bird movement with rotation 2D

https://youtu.be/GlFd8f6DCP8

Give the Flappy bird game Object a Circle collider 2D and a Rigidbody2D with the gravity set to 1. Give the game Object this script :

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class FlappyMovement : MonoBehaviour
{

public float velocity = 4; private
Rigidbody2D rb;
void Start()
{
```

```
rb = GetComponent<Rigidbody2D>();
}
void Update()
{
if (Input.GetMouseButtonDown(0))
{
rb.velocity = Vector2.up * velocity;
}
transform.eulerAngles = new Vector3(0,0,rb.velocity.y);
}
}
```

Every time the player uses the mouse button the bird will go up a little. You can use key or touch function as well for the flying.

Grid based movement with LayerMask used as borders 2D

https://youtu.be/oZCqdhTYjQQ

Place the game Object on a grid. Add a child game Object and call it movepoint. Give the grid moving gameObject this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class Gridmovement : MonoBehaviour
{

public float moveSpeed;
public float Horizontalgrid;
public float Verticalgrid; public
Transform movePoint;
public LayerMask StopsMovement; void
Start()
{
movePoint.parent = null;
}
void Update()
{
transform.position = Vector3.MoveTowards(transform.position, movePoint.position,
moveSpeed * Time.deltaTime);
if (Vector3.Distance(transform.position, movePoint.position) <= .05f)
{
if (Mathf.Abs(Input.GetAxisRaw("Horizontal")) == 1f)
{
if (!Physics2D.OverlapCircle(movePoint.position + new
Vector3(Input.GetAxisRaw("Horizontal"), 0f), .2f, StopsMovement))
{
movePoint.position += new Vector3(Horizontalgrid * Input.GetAxisRaw("Horizontal"),0f);
}
```

```
}
else
if (Mathf.Abs(Input.GetAxisRaw("Vertical")) == 1f)
{
if (!Physics2D.OverlapCircle(movePoint.position + new Vector3(0f, 0.9f *
Input.GetAxisRaw("Vertical"), 0f), .2f, StopsMovement))
{
movePoint.position += new Vector3(0f, Verticalgrid * Input.GetAxisRaw("Vertical"),
0f);
}
}
}
}
}
```

Drag the movepoint into the inspector's view and set the needed grid sizes and moving speed. The movement can be stopped if you place an object at the border with a layer that is called with StopsMovement. On release key the game object will always end movement in a grid. If you want diagonal movement you only need to remove the word else in the script.

Growing and Shrinking gameObjects using localscale and pivot settings (Resize and scaling)2D

https://youtu.be/dHC-w5A-aYA

Make sure that the pivot of the game object is placed at the point where it should grow or shrink from so it holds its position. Place an object that needs to grow or shrink and give it this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class ShrinkGrow : MonoBehaviour
{
public float changeSpeed = 1;
public float MaxupSize; public
float MindownSize;
Vector3 temp;
void Update()
{
if (Input.GetKey(KeyCode.UpArrow) && temp.x < MaxupSize)
{
temp = transform.localScale; temp.x += 1f *
changeSpeed * Time.deltaTime; temp.y += 1f *
changeSpeed * Time.deltaTime;
transform.localScale = temp;
}
if (Input.GetKey(KeyCode.DownArrow) && temp.x > MindownSize)
{
temp = transform.localScale; temp.x -= 1f *
changeSpeed * Time.deltaTime; temp.y -= 1f *
```

```
changeSpeed * Time.deltaTime;
transform.localScale = temp;
}
}
}
```

You can set the maximum and minimum size of the game object in the inspector's view. This example scales at a key press but you can use it anyway you want. You can change the changing speed into your own likings.

Horizontal Parallax Scrolling 2D

https://www.youtube.com/watch?v=5cteIYhpoCI

Create an object sprite to scroll and make sure it is seamless. Place the same object as a child right next to it. Take the x-position of this child and give it as a negative to the parent object. Give this script to the parallax scroll parent object.

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class Scrollingground : MonoBehaviour
{
public float MovingSpeed = 1f;
public float offset; private
Vector2 StartPosition; private
float newXposition;
void Start()
{
StartPosition = transform.position;
}
void Update()
{
newXposition = Mathf.Repeat(Time.time * -MovingSpeed, offset); transform.position
= StartPosition + Vector2.right * newXposition;
}
}
```

Set speed and the offset (X-position of the child object) in the inspector's window.

Look at mouse cursor and fire bullets into that direction 3D

https://youtu.be/fd4H6yNZlJ4

Give the gameObject that needs to look (Face all time) the mouse cursor. Drag a prefab made bullet into the inspector's view. Give the gameObject a child empty gameObject and call it Barrel. Drag this barrel into the Inspector's view. This barrel is the point where the bullet will be instantiated.

Give the follow mouse gameobject this script :

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;

public class Lookatmouse : MonoBehaviour
{
public GameObject Bullet;
public Transform Barrel;

void Update()
{
if (Input.GetMouseButtonDown(0))
{
Instantiate(Bullet, Barrel.position, Barrel.rotation);
}
Ray mouseRay = Camera.main.ScreenPointToRay(Input.mousePosition);
float midPoint = (transform.position - Camera.main.transform.position).magnitude * 0.5f;
transform.LookAt(mouseRay.origin + mouseRay.direction * midPoint);
}
}
```

A simple bullet script for the bullet gameObjects :

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;

public class Bullet : MonoBehaviour
{
public float bulletSpeed ;

void Update()
{
transform.Translate(Vector3.forward * bulletSpeed * Time.deltaTime);
}
}
```

Moving automatically to any given tag (Homing missile target)

https://youtu.be/Xe-hJM9U_6k

Place an object as target give it the tag name Enemy. Give the game object that needs to move to this enemy this script :

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class HomingMissile : MonoBehaviour
{
private GameObject target;
public float speed = 5f; public
float rotatingSpeed = 200f;
private Rigidbody2D rb; private
GameObject target;
void Start()
{
target = GameObject.FindGameObjectWithTag("Enemy"); rb
= GetComponent<Rigidbody2D>();
}
void FixedUpdate()
{
Vector2 point2Target = (Vector2)transform.position -
(Vector2)target.transform.position; point2Target.Normalize();
float value = Vector3.Cross(point2Target, transform.up).z; rb.angularVelocity
= rotatingSpeed * value;
rb.velocity = transform.up * speed;
}
void OnCollisionEnter2D()
{
Destroy(gameObject);
}
}
```

You can change the settings of Rotation speed and movement speed in the inspector's view.
The rocket has a box collider 2D and a Rigidbody2D while the target has only a box collider 2D The rocket will always go to the given game object with the right tag name.

Moving automatically to any given Vector2

https://youtu.be/9oVybaehlr0

Place an object where the game object with this script should move to and take its position on the x and y -axis. Give the game object that needs to move to this point this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class MoveToTarget : MonoBehaviour
{
public Vector2 TargetLocation; public
bool AbleToMove = false; public float
Speed = 2f;
public float Release = 3f;
IEnumerator Moving()
{
yield return new WaitForSeconds(Release);
AbleToMove = true;
}
void Start()
{
StartCoroutine(Moving());
}
void Update()
{
if(AbleToMove == true)
{
transform.position = Vector2.MoveTowards(transform.position, TargetLocation,Speed*Time.deltaTime); if
(transform.position.x == TargetLocation.x && transform.position.y == TargetLocation.y)
{
Destroy(this.gameObject);
}
}
}
}
```

Set the target location in the inspector's view and set the moving speed to your likings.

Moving Platforms Horizontal and Vertical 2D

https://youtu.be/8X3FAd5i-i8

Place 2 platform game objects and give them this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class MovingPlatform : MonoBehaviour
```

```
{
public int PlatformType;
public bool UpDown = true;
public bool LeftRight = true;
public float SpeedUpDown;
public float SpeedLeftRight;
public float up; public float
down; public float left;
public float right; private
Vector3 StartPos;
void Start()
{
StartPos = transform.position;
}
void Update()
{
if (PlatformType == 2 && transform.position.x < StartPos.x + right && LeftRight == true)
{
transform.Translate(Vector2.right * SpeedLeftRight * Time.deltaTime); if
(transform.position.x > StartPos.x + right)
LeftRight = false;
}
if (PlatformType == 2 && transform.position.x > StartPos.x - left && LeftRight == false)
{
transform.Translate(Vector2.left * SpeedLeftRight * Time.deltaTime); if
(transform.position.x < StartPos.x - left)
LeftRight = true;
}
if (PlatformType == 1 && transform.position.y > StartPos.y - down && UpDown == false)
{
```



```
transform.Translate(Vector2.down * SpeedUpDown * Time.deltaTime); if
(transform.position.y < StartPos.y - down)
UpDown = true;
}
if (PlatformType == 1 && transform.position.y < StartPos.y+up && UpDown == true)
{
transform.Translate(Vector2.up * SpeedUpDown * Time.deltaTime);
if (transform.position.y > StartPos.y + up)
UpDown = false;
}
}
}
```

Settings can be changed in the Inspector's view. Platform type 1 is for a up and down going platform. Platform type 2 is for going left and right platform. Set their speed and how far in distance the platform should move before it will reverse its movement.

Orbit a gameObject around another gameObject 2D.

https://youtu.be/BtYXIt8NIUE

You give the objects that needs to orbit this script. As soon as it appears it will look for the game Object with the tag Player and it will use that as the center point. It will stay with this gameObject wherever it goes until you turn it off.

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class Bonusspinner : MonoBehaviour
{
public float xSpread; public float
ySpread; private GameObject
centerPoint; public float
rotSpeed; public bool
rotateClockWise;
float timer = 0;
private void Start()
{
centerPoint = GameObject.FindWithTag("Player");
}
private void Update()
{
timer += Time.deltaTime * rotSpeed;
Rotate();
}
void Rotate()
{
if (rotateClockWise)
{
float x = -Mathf.Cos(timer) * xSpread; float
z = Mathf.Sin(timer) * ySpread; Vector3
pos = new Vector3(x, z, 0f);
transform.position = pos + centerPoint.transform.position;
}
else
{
float x = Mathf.Cos(timer) * xSpread; float
z = Mathf.Sin(timer) * ySpread; Vector3
pos = new Vector3(x, z, 0f);
transform.position = pos + centerPoint.transform.position;
}
}
}
```

In the inspector's view you can set the size of the radius (X and Y axis) as well as the rotation speed. This way you can make it fit the style of your game. You can choose the rotation direction as well.

Rolling dice using Images and sound 2D

https://youtu.be/wZqCQgexdPE

Place the basic image for the dice. Give the game Object this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;

public class DiceRoll : MonoBehaviour
{
public Sprite DiceCover, Side1, Side2, Side3, Side4, Side5, Side6;
public int ShowDiceSide = 0; public float RollingTime = 5f;
public bool RunTimer = false; public AudioClip Ding; public
AudioClip Diceoutcome;
IEnumerator RollDice()
{
while (RollingTime > 0 && RunTimer == true)
{
ShowDiceSide = Random.Range(1, 7);
AudioSource.PlayClipAtPoint(Ding, new Vector3(0, 0, -10));
yield return new WaitForSeconds(0.2f);
ShowDiceSide = Random.Range(1, 7);
AudioSource.PlayClipAtPoint(Ding, new Vector3(0, 0, -10));
yield return new WaitForSeconds(0.2f);
ShowDiceSide = Random.Range(1, 7);
AudioSource.PlayClipAtPoint(Ding, new Vector3(0, 0, -10));
yield return new WaitForSeconds(0.2f);
ShowDiceSide = Random.Range(1, 7);
AudioSource.PlayClipAtPoint(Ding, new Vector3(0, 0, -10));
yield return new WaitForSeconds(0.2f);
ShowDiceSide = Random.Range(1, 7);
AudioSource.PlayClipAtPoint(Ding, new Vector3(0, 0, -10));
yield return new WaitForSeconds(0.2f);
ShowDiceSide = Random.Range(1, 7);
AudioSource.PlayClipAtPoint(Ding, new Vector3(0, 0, -10));
yield return new WaitForSeconds(0.2f);
ShowDiceSide = Random.Range(1, 7);
AudioSource.PlayClipAtPoint(Ding, new Vector3(0, 0, -10)); yield
return new WaitForSeconds(0.2f);
}
}

void Update()
{
if (Input.GetKeyDown(KeyCode.Space) && RunTimer == false)
{
RunTimer = true;
StartCoroutine(RollDice());
}
```

```
if(RunTimer == true)
{
RollingTime-= Time.deltaTime;
if(RollingTime < 1)
{
AudioSource.PlayClipAtPoint(Diceoutcome, new Vector3(0, 0, -10));
StopCoroutine(RollDice());
RunTimer = false;
RollingTime = 5;
}
}
if(ShowDiceSide == 0)
{
this.GetComponent<SpriteRenderer>().sprite = DiceCover;
}
if (ShowDiceSide == 1)
{
this.GetComponent<SpriteRenderer>().sprite = Side1;
}
if (ShowDiceSide == 2)
{
this.GetComponent<SpriteRenderer>().sprite = Side2;
}
if (ShowDiceSide == 3)
{
this.GetComponent<SpriteRenderer>().sprite = Side3;
}
if (ShowDiceSide == 4)
{
this.GetComponent<SpriteRenderer>().sprite = Side4;
}
if (ShowDiceSide == 5)
{
this.GetComponent<SpriteRenderer>().sprite = Side5;
}
if (ShowDiceSide == 6)
{
this.GetComponent<SpriteRenderer>().sprite = Side6;
}
}
}
```

Drag the images and sounds you use to the inspectors view. Change the roll time to your own likings. The roll is activated by the press of a key (Spacebar). You can use the click of a mouse or touch if you want.

Rotating a Character in the Direction of Movement 3D

https://youtu.be/mztheOfd9YU

Place a character and give it this script.

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;

public class Movement : MonoBehaviour
{

public float Speed = 4f;
public float rotationSpeed;

void Update()
{
float xDirection = Input.GetAxis("Horizontal"); float
zDirection = Input.GetAxis("Vertical");
Vector3 moveDirection = new Vector3(xDirection, 0.0f, zDirection);
moveDirection.Normalize();
transform.Translate(moveDirection *Speed*Time.deltaTime,Space.World);
if(moveDirection !=Vector3.zero)
{
///   Quaternion toRotation = Quaternion.LookRotation(moveDirection, Vector3.up);
///  transform.rotation = Quaternion.RotateTowards(transform.rotation, toRotation, rotationSpeed
*Time.deltaTime);
transform.forward = moveDirection;
}
}
}
```

If you want the character to vissually rotate just delete the transform.forward line and uncomment the 2 green lines. Set rotationSpeed to 700 in the inspector's view to test.

Rotating gameObjects with mouse 3D

https://youtu.be/IFwpbCTzjEA

Create a 3D cube and give it this script. When you click with the mouse on the cube you can make it rotate left right up and down. Place it at the center of a child object and make the cube invisible. Now you can rotate any model that has the cube as parent.

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class Viewmodel : MonoBehaviour
{
Vector3 mPrevPos = Vector3.zero;
Vector3 mPosDelta = Vector3.zero; void
Update()
{
if(Input.GetMouseButton(0))
{
mPosDelta = Input.mousePosition - mPrevPos;
transform.Rotate(transform.up, Vector3.Dot(mPosDelta,Camera.main.transform.right),
Space.World);
transform.Rotate(Camera.main.transform.right,Vector3.Dot(mPosDelta,Camera.main.tra
nsform.up), Space.World);
}
mPrevPos = Input.mousePosition;
}
}
```

Rotating smoothly between 2 given angles 2D/3D

https://youtu.be/tBFUr3DumYs

You can use any axis you want so it can be use for 2D and 3D Projects. The game object will rotate between the 2 given angles on the press of the space bar. Set speed in the Inspector's view.

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;

public class OpenClosePanel : MonoBehaviour
{

Quaternion targetAngle_90 = Quaternion.Euler(90, 0, 0);
Quaternion targetAngle_0 = Quaternion.Euler(0, 0, 0); public
Quaternion currentAngle;
public float Speed;

void Start()
{currentAngle = targetAngle_0;
}

void Update()
{
```

```
if (Input.GetKeyDown(KeyCode.Space))
{
ChangeCurrentAngle();
}
}
this.transform.rotation = Quaternion.Slerp(this.transform.rotation, currentAngle, Speed*Time.deltaTime);
}

void ChangeCurrentAngle()
{
if (currentAngle.eulerAngles.x == targetAngle_0.eulerAngles.x)
{
currentAngle = targetAngle_90;
}
else
{
currentAngle = targetAngle_0;
}
}
}
```

Set gameObject as Child and use a tag to set the parent gameObject

https://www.youtube.com/watch?v=8BwqCL4HUDI

In the example you can see how this script is used in a break out kind of style. At start the ball will find the bat and attaches itself to it. On space it will release itself while R will attach it againg. Make sure the object it needs to attach to has a tag called Bat.

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class FindParent : MonoBehaviour
{
private GameObject Bat;
void Start()
{
Bat = GameObject.FindGameObjectWithTag("Bat"); transform.position
= new Vector3(Bat.transform.position.x,
Bat.transform.position.y+0.7f, Bat.transform.position.z);
transform.SetParent(Bat.transform);
}
```

```
void Update()
{
if(Input.GetKeyDown(KeyCode.Space))
{
transform.SetParent(null);
}
if (Input.GetKeyDown(KeyCode.R))
{
Bat = GameObject.FindGameObjectWithTag("Bat"); transform.position
= new Vector3(Bat.transform.position.x, Bat.transform.position.y +
0.7f, Bat.transform.position.z); transform.SetParent(Bat.transform);
}
}
}
```

You can use this script anyway you like it by changing the parent and its tag or create an automatic attaching on impact (Collision or trigger).

Shuffle an array filled with gameObjects

https://youtu.be/-DCqdsxdigA

As example an array with 20 GameObject is used.

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class ShuffleArray : MonoBehaviour
{
public GameObject [] MCards = new GameObject[20]; void
reshuffle(GameObject[] MCards)
{
for (int t = 0; t < MCards.Length; t++)
{
GameObject tmp = MCards[t]; int r =
Random.Range(t, MCards.Length);
MCards[t] = MCards[r];
MCards[r] = tmp;
}
```


```
}
```

Every time you need the array to be shuffled call the reshuffle function with : reshuffle (MCards); This script is easy to use for memory and or card games.

Spawn objects random without overlapping

https://youtu.be/OQrdzWt-gxE

First we create an empty game Object we call ObjectSpawner and give it this script :

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class StarSpawner : MonoBehaviour
{
public GameObject spawnPrefab;
public float CountD;
void Update()
{
CountD -= Time.deltaTime;
if (CountD <= 0)
{
SpawnRandomSpot();
CountD = 2f;
}
}
```

```
public void SpawnRandomSpot()
{
int ranX = Random.Range(-8, 8); int
ranY = Random.Range(-6, 6);
Vector3 randomPosition = new Vector3(ranX, ranY, 0f);
Instantiate(spawnPrefab, randomPosition, Quaternion.identity);
}
}
```

You can set the spawn time in the inspector's view as well as the game Object it needs to spawn.
Get the coordinates for the X and Y limits from the inspector's view.
In order to let the spawned Objects not overlap we need these objects to have this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
```

```
public class Star : MonoBehaviour
{
public float scanRadius = 3f;
public LayerMask filterMask;
private StarSpawner SP; private
Collider2D checkCollider;
void Awake()
{
SP = FindObjectOfType<StarSpawner>();
}
void Update()
{
checkCollider = Physics2D.OverlapCircle(transform.position, scanRadius, filterMask);
if (checkCollider != null && checkCollider.transform != transform)
{
Destroy(checkCollider.gameObject);
SP.CountD = 0;
}
}
```

```
protected void OnDrawGizmos()
{
Gizmos.color = Color.red;
Gizmos.DrawWireSphere(transform.position, scanRadius);
}
}
```

Create a LayerMask and select it in this gameObject. You can set the overlap radius in the Inspector's view. When it overlaps it gets removed and the timer is set to 0 so a new object is created at another spot immediately. You can name it what you want and use any object you need to be spawned.

Super Mario like movement with double jumping and flipping the gameObject 2D

https://youtu.be/YLfQfCAQu9E

This movement script will give you an excellent start to create your own 2D platformer game. Give your character a Rigidbody2D and a Box collider 2D. Give the game Object this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class MarioMovement : MonoBehaviour
{
public float moveSpeed; public
bool FacingRight = true; public
float JumpForce; public bool
OnGround; public Transform
groundCheck; public
LayerMask groundlayer; public
```

```
bool DoubleJump; private float
xInput, yInput; Rigidbody2D rb;
Vector3 characterScale; float
characterScaleX;
void Start()
{
rb = GetComponent<Rigidbody2D>(); characterScale
= transform.localScale;
characterScaleX = characterScale.x;
}
void FixedUpdate()
{
if (xInput > 0)
{
FacingRight = true;
}
if (xInput < 0)
{
FacingRight = false;
}


xInput = Input.GetAxisRaw("Horizontal"); //xInput
= Input.GetAxis("Horizontal");//
transform.Translate(xInput * moveSpeed, 0, 0);
PlatformMovement();
transform.localScale = characterScale;
OnGround = Physics2D.OverlapCircle(groundCheck.position, 0.2f, groundlayer);
}
void Update()
{
if (Input.GetKeyDown(KeyCode.Space))
{
if (OnGround == true)
{
Jumping();
DoubleJump = true;
}
else if(DoubleJump)
{
JumpForce = JumpForce / 1.5f;
Jumping();
DoubleJump = false;
JumpForce = JumpForce * 1.5f;
}
}
}
void Jumping()
{
rb.velocity = Vector2.up * JumpForce;
}
```

```
void PlatformMovement()
{
if (FacingRight == true)
{
characterScale.x = characterScaleX;
}
if (FacingRight == false)
{
characterScale.x = -characterScaleX;
}
rb.velocity = new Vector2(moveSpeed * xInput, rb.velocity.y);
}
}
```

When you're not using Raw in the input line the movement gets a more sliding look to it. When raw is
used stopping movement happens immediately on key release. The second jump is less high then the first by dividing the original jumpforce. After the second jump the jumpforce is restored. Flipping the character uses the localScale. You might have noticed people use FlipX in the script however when your character has a child object, this child object will not flip with. So using localScale is better. The bool FacingRight determines when the gameObject should be flipped.

Top view movement 2D Vehicle with engine pitching sound

https://youtu.be/OZVwltMWZts

Give the car a 2DBoxcollider and a Rigidbody2D. Set its gravity to 0. Add an audio source and drag the engine sound to use to the inspector's view. Select the loop box.

```
 using System.Collections; using
System.Collections.Generic;
using UnityEngine;

public class Car5th : MonoBehaviour
```

```
{ public float rotateSpeed = -
130; public float moveSpeed = 5;

private AudioSource audioSource;
public int startingPitch = 0; public
int timeToDecrease = 1; private
Rigidbody2D rb;

private void Awake()
{ audioSource =
GetComponent<AudioSource>();
}
void Start()
{ rb =
GetComponent<Rigidbody2D>();
audioSource.Play();
}
void Update()
{
if (rb.velocity.magnitude > 0.2)
{
rb.angularVelocity = Input.GetAxis("Horizontal") * rotateSpeed; rotateSpeed
= -130;
} else
{
if(rb.velocity.magnitude < 0.2) rotateSpeed
= 0;
}
```

```
rb.velocity = transform.up * Input.GetAxis("Vertical") * moveSpeed; if
(rb.velocity.magnitude >= 0.2 && audioSource.pitch < 5)
{
audioSource.pitch += 8 * Time.deltaTime * startingPitch / timeToDecrease;
} else
{
if (audioSource.pitch > 1)
{
audioSource.pitch -= 10 * Time.deltaTime * startingPitch / timeToDecrease;
}
}
}
}
```

You can change the moving and rotation speed in the inspector's view as well as the starting pitch and decreasing pitch.

Top view movement with 8 directions 2D

https://youtu.be/2l3YBSvf1qI

Give the object that needs to walk around this script:

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine; public class MovePlayer :
MonoBehaviour
{
public float MovingSpeed;
private float Xmove;
private float Ymove; void
Update()
{
Xmove = Input.GetAxis("Horizontal");
Ymove = Input.GetAxis("Vertical");
Vector3 moveDir = new Vector3(Xmove, Ymove).normalized; transform.position
+= moveDir * MovingSpeed * Time.deltaTime;
}
}
```

You can set the speed of moving in the inspector's view. With the use of normalized, the character will move the same speed diagonal as it does horizontal and vertical. Without it the magnitude will be higher when moving diagonal so the object will move faster. (Moving diagonal uses 2 moves (keys) at same time).

Top view movement with 8 directions using velocity (Rigidbody2D)

https://youtu.be/yC2KQPMPk-U

Give the character sprite a Rigidbody2D but set it to Kinematic.

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;
public class PlayerControler : MonoBehaviour
{
public float Speed; private
Vector2 moveVelocity; private
Rigidbody2D rb;
void Start()
{
rb = GetComponent<Rigidbody2D>();
}
void Update()
{
```

```
Vector2 moveInput = new Vector2(Input.GetAxis("Horizontal"),
Input.GetAxis("Vertical"));
moveVelocity = moveInput.normalized * Speed;
}
void FixedUpdate()
{
rb.MovePosition(rb.position + moveVelocity * Time.fixedDeltaTime);
}
}
```

This script will give your character smooth movement. Normalized is used so the speed will not be faster when moving diagonal.
When you want a more direct movement (No key's pressed is a direct stop instead of slowing down first) all you need to do is add the word Raw to the GetAxis lines.

Vector2 moveInput = new Vector2(Input.GetAxisRaw("Horizontal"),Input.GetAxisRaw("Vertical"));

Using your own Custom Cursor / Crosshair 2D/3D

https://youtu.be/o2wVdLeLMYs

```
using System.Collections; using
System.Collections.Generic; using
UnityEngine;

public class CrosshairPL1 : MonoBehaviour
{
public Texture2D OwnCursor;

void Start()
{
Cursor.SetCursor(OwnCursor, Vector2.zero, CursorMode.ForceSoftware);
}

void Update()
{
Vector2 cursorPos = Camera.main.ScreenToWorldPoint(Input.mousePosition);
transform.position = cursorPos;
}
}
```

Wave movement 2D

https://youtu.be/FRBQk4sDtMc

Give this script to a Game Object that has to move left and right in a wave pattern. You can set the frequency,speed and magnitude. The

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Wavemovement : MonoBehaviour
{


[SerializeField]
float moveSpeed = 5f;
[SerializeField]
float frequency = 20f;
[SerializeField]
float magnitude = 0.5f;
bool facingRight = true;
Vector3 pos, localScale;

void Start()
{
pos = transform.position;
localScale = transform.localScale;
}


void Update()
{
CheckWhereToFace();
if (facingRight)
MoveRight();
else
MoveLeft();
}

void CheckWhereToFace()
{
if (pos.x < -7f)
facingRight = true;
else if (pos.x > 7f)
facingRight = false;
if (((facingRight) && (localScale.x < 0)) || ((!facingRight) && (localScale.x > 0)))
localScale.x *= -1;
transform.localScale = localScale;
}

void MoveRight()
{
pos += transform.right * Time.deltaTime * moveSpeed;
transform.position = pos + transform.up * Mathf.Sin(Time.time * frequency) *magnitude;
}
```

```
void MoveLeft()
{
pos -= transform.right * Time.deltaTime * moveSpeed;
transform.position = pos + transform.up * Mathf.Sin(Time.time * frequency) *magnitude;
}
}
```