if(Input.GetKeyDown(KeyCode.C))

{

Launch();

}

void Update()

{

//remember ! inverse the test, so if broken is true !broken will be false and return won’t be executed.

if(!broken)

{

return;

}

//Public because we want to call it from elsewhere like the projectile script

public void Fix()

{

broken = false;

rigidbody2D.simulated = false;

}

RUBY CONTROLLER:

﻿public class RubyController : MonoBehaviour

{

public float speed = 3.0f;

public int maxHealth = 5;

public GameObject projectilePrefab;

public int health { get { return currentHealth; }}

int currentHealth;

public float timeInvincible = 2.0f;

bool isInvincible;

float invincibleTimer;

Rigidbody2D rigidbody2d;

float horizontal;

float vertical;

Animator animator;

Vector2 lookDirection = new Vector2(1,0);

// Start is called before the first frame update

void Start()

{

rigidbody2d = GetComponent<Rigidbody2D>();

animator = GetComponent<Animator>();

currentHealth = maxHealth;

}

// Update is called once per frame

void Update()

{

horizontal = Input.GetAxis("Horizontal");

vertical = Input.GetAxis("Vertical");

Vector2 move = new Vector2(horizontal, vertical);

if(!Mathf.Approximately(move.x, 0.0f) || !Mathf.Approximately(move.y, 0.0f))

{

lookDirection.Set(move.x, move.y);

lookDirection.Normalize();

}

animator.SetFloat("Look X", lookDirection.x);

animator.SetFloat("Look Y", lookDirection.y);

animator.SetFloat("Speed", move.magnitude);

if (isInvincible)

{

invincibleTimer -= Time.deltaTime;

if (invincibleTimer < 0)

isInvincible = false;

}

if(Input.GetKeyDown(KeyCode.C))

{

Launch();

}

}

void FixedUpdate()

{

Vector2 position = rigidbody2d.position;

position.x = position.x + speed \* horizontal \* Time.deltaTime;

position.y = position.y + speed \* vertical \* Time.deltaTime;

rigidbody2d.MovePosition(position);

}

public void ChangeHealth(int amount)

{

if (amount < 0)

{

if (isInvincible)

return;

isInvincible = true;

invincibleTimer = timeInvincible;

}

currentHealth = Mathf.Clamp(currentHealth + amount, 0, maxHealth);

Debug.Log(currentHealth + "/" + maxHealth);

}

void Launch()

{

GameObject projectileObject = Instantiate(projectilePrefab, rigidbody2d.position + Vector2.up \* 0.5f, Quaternion.identity);

Projectile projectile = projectileObject.GetComponent<Projectile>();

projectile.Launch(lookDirection, 300);

animator.SetTrigger("Launch");

}

}