﻿using UnityEngine;

using System.Collections;

public class TargetDamage : MonoBehaviour {

public int hitPoints = 2; // The amount of damage our target can take

public Sprite damagedSprite; // The reference to our "damaged" sprite

public float damageImpactSpeed; // The speed threshold of colliding objects before the target takes damage

private int currentHitPoints; // The current amount of health our target has taken

private float damageImpactSpeedSqr; // The square value of Damage Impact Speed, for efficient calculation

private SpriteRenderer spriteRenderer; // The reference to this GameObject's sprite renderer

void Start () {

// Get the SpriteRenderer component for the GameObject's Rigidbody

spriteRenderer = GetComponent <SpriteRenderer> ();

// Initialize the Hit Points

currentHitPoints = hitPoints;

// Calculate the Damage Impact Speed Squared from the Damage Impact Speed

damageImpactSpeedSqr = damageImpactSpeed \* damageImpactSpeed;

}

void OnCollisionEnter2D (Collision2D collision) {

// Check the colliding object's tag, and if it is not "Damager", exit this function

if (collision.collider.tag != "Damager")

return;

// Check the colliding object's velocity's Square Magnitude, and if it is less than the threshold, exit this function

if (collision.relativeVelocity.sqrMagnitude < damageImpactSpeedSqr)

return;

// We have taken damage, so change the sprite to the damaged sprite

spriteRenderer.sprite = damagedSprite;

// Decriment the Current Health of the target

currentHitPoints--;

// If the Current Health is less than or equal to zero, call the Kill() function

if(currentHitPoints <= 0)

Kill ();

}

void Kill () {

// As the particle system is attached to this GameObject, when Killed, switch off all of the visible behaviours...

spriteRenderer.enabled = false;

GetComponent<Collider2D>().enabled = false;

GetComponent<Rigidbody2D>().isKinematic = true;

// ... and Play the particle system

GetComponent<ParticleSystem>().Play();

}

}