1. Update the Game Manager Script for adding value to good hits and perfect hits.

|  |
| --- |
| using System.Collections; using System.Collections.Generic; using UnityEngine; using UnityEngine.UI;  public class GameManager : MonoBehaviour {  //this class is used to control when the music starts playing  //keeps tracks of the effects of when a note is hit   public AudioSource theMusic;  public bool startPlaying;  public BeatScroller theBS;  //used to only have one instance of the object  public static GameManager instance;   //keeps track of your score  public int currentScore;  //default value per note  public int scorePerNote = 100;  //value of a note with a good hit  public int scorePerGoodNote = 125;  //value of a note with a perfect hit  public int scorePerPerfectNote = 150;   public Text scoreText;  public Text multiText;   //the current multiplier to add to score, ex. 1, 2, 3  public int currentMultiplier;  //keeps track on how many notes you have to the next multiplier threshold  public int multiplierTracker;  //the # of notes you need to hit the threshold, need 4 notes to get to 2, 8 notes to get to 3 etc.  public int[] multiplierThresholds;   // Start is called before the first frame update  void Start()  {  instance = this;  scoreText.text = "Score: 0";  currentMultiplier = 1;  }   // Update is called once per frame  void Update()  {  //when the game is booted up, the music and game should not start yet  if (!startPlaying)  {  //if a button is pushed, start the beatscroller and music  if (Input.anyKeyDown)  {  startPlaying = true;  theBS.hasStarted = true;   theMusic.Play();  }  }  }   //used to see if you hit a note  public void NoteHit()  {  //Debug.Log("Hit On Time");   //prevents the counters from going beyond the index of the array  if (currentMultiplier - 1 < multiplierThresholds.Length) {   //increment the multiplier track  multiplierTracker++;  //if the multiplierTracker now equals the multiplierThreshold, reset the count and increment the currentMultiplier   if (multiplierThresholds[currentMultiplier - 1] <= multiplierTracker)  {  multiplierTracker = 0;  currentMultiplier++;  }  }  multiText.text = "Multiplier: x" + currentMultiplier;   //currentScore += scorePerNote \* currentMultiplier;  scoreText.text = "Score: " + currentScore;  }   //Is called in the Note Object Update Function to check the accuracy when a note can be pressed  public void NormalHit()  {  currentScore += scorePerNote \* currentMultiplier;  NoteHit();  }   public void GoodHit()  {  currentScore += scorePerGoodNote \* currentMultiplier;  NoteHit();  }   public void PerfectHit()  {  currentScore += scorePerPerfectNote \* currentMultiplier;  NoteHit();  }   //used to see if you missed a note  public void NoteMissed()  {  Debug.Log("Missed Note");   //when you miss, the multiplier resets  currentMultiplier = 1;  multiplierTracker = 0;  multiText.text = "Multiplier: x" + currentMultiplier;  } } |

1. Go to Assets > Rhythm Game Tutorial > Graphics and drag out the four effect sprites out to the scene.
2. Rotate the four effects sprites by 13.18
3. To create particles, go to GameObject > Effects > Particle System.
4. Change the Shape to Circle, Emissions > Rate over Time to 0
5. Click the plus sign button under Emission
6. Change the Rotation to 0, 0, 0
7. Change the Duration (at the top setting under Particle System) to 1
8. Change the Start Lifetime to 1
9. Change the Size over Lifetime to a downward curve
10. Drag a Particle System under an effect
11. Change the color to be the same as the text color of the effect sprite
12. Change the Start Speed 1
13. Change the Shape > Radius to 0.25
14. Change the Emission > Bursts > Count to 10
15. Select all four effect sprites and change the Order in Layer to 1 to be always above the buttons
16. Uncheck looping for the Particle System
17. Duplicate the Particle System for effect sprite and change their color.
18. Create a new script call EffectObject

|  |
| --- |
| using System.Collections; using System.Collections.Generic; using UnityEngine;  public class EffectObject : MonoBehaviour {  //how long the effect last for  public float lifetime = 1f;   // Start is called before the first frame update  void Start()  {    }   // Update is called once per frame  void Update()  {  //After it activates for 1 second, destroy it  Destroy(gameObject, lifetime);  } } |

1. Attach the EffectObject script to every effect sprite
2. Go to Asset and create a new folder call Prefabs
3. Create prefabs for the modified effect sprites
4. Delete the effect sprites from the scene
5. The NoteObject script is updated for bringing effect sprites to the game when a note it hit/missed

|  |
| --- |
| using System.Collections; using System.Collections.Generic; using UnityEngine;  public class NoteObject : MonoBehaviour {  //A script for the arrow/note objects detecting whether the note can be pressed based on position   public bool canBePressed;  public KeyCode keyToPress;   //variables use to bring the effect in the world when a note is hit or missed  public GameObject hitEffect, goodEffect, perfectEffect, missEffect;   // Use this for initialization  void Start()  {   }   // Update is called once per frame  void Update()  {  if (Input.GetKeyDown(keyToPress))  {  if (canBePressed)  {  gameObject.SetActive(false);  Destroy(this.gameObject);  //if you hit the key when it canBePressed, print you hit a note  //GameManager.instance.NoteHit();    if(Mathf.Abs(transform.position.y) > 0.25)  {  Debug.Log("Hit");  GameManager.instance.NormalHit();  //make a new effect object at the position of the arrow and with the rotation of the effect sprite  Instantiate(hitEffect, transform.position, hitEffect.transform.rotation);  }  else if (Mathf.Abs(transform.position.y) > 0.05f)  {  Debug.Log("Good");  GameManager.instance.GoodHit();  Instantiate(goodEffect, transform.position, goodEffect.transform.rotation);   }  else  {  Debug.Log("Perfect");  GameManager.instance.PerfectHit();  Instantiate(perfectEffect, transform.position, perfectEffect.transform.rotation);   }  }  }  }   //if the an arrow is colliding, then canBePressed is true  private void OnTriggerEnter2D(Collider2D other)  {  if (other.tag == "Activator")  {  canBePressed = true;  }  }   private void OnTriggerExit2D(Collider2D other)  {  //Debug.Log(gameObject.transform.position);   if (other.tag == "Activator")  {  canBePressed = false;   float positionY = gameObject.transform.position.y;   //when the note passed the buttons  if (positionY < -0.419)  {  GameManager.instance.NoteMissed();  Instantiate(missEffect, transform.position, missEffect.transform.rotation);  Destroy(this.gameObject);    }  }  } } |

1. Select all notes and in the new effect fields of the note object, drag the prefabs to the corresponding fields