JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY

**ANIMATION PROJECT**

BOX COLLECTOR

(ARCADE GAME 2D)

PROJECT MEMBERS:-

GARVIT AGRAWAL 14103115

SHUBHAM CHAUHAN 14103175

SANJAY CHOUDHARY 14103183

ISHAN JAIN 14103303

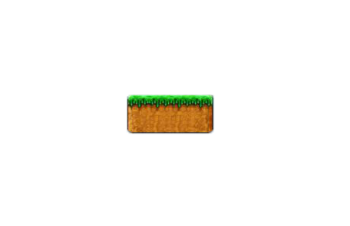
**Project submitted under the guidance of :-**

**Dr. SUMA DAWN**

FEATURES OF THE GAME:-

* In This game player has to collect boxes which are placed at various levels of hills.
* Player jumps on randomly generated moving platform
* The moving range of platforms is also random.
* Games contain 2 levels
* First level camera is moving horizontally
* second level camera is moving vertically
* He scores points on colliding the boxes which are placed at various levels of hills.
* There are three types of color boxes which have exactly one point
* Box are generated randomly at various levels.
* On completion of Level 1, player is direct go to second levels.

**ALL ASSETS,IMAGES OF MATERIALS(SPRITES,TEXTURES)**



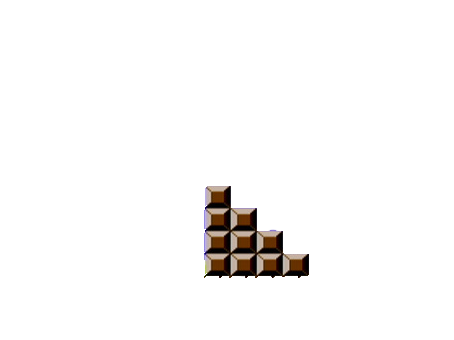


TABLE OF SCRIPTS

|  |  |  |
| --- | --- | --- |
| **Sr. no** | **Script files(Names)** | **Usage** |
| 1. | Camera.cs | 1. Acquires the position of the camera and moves it accordingly. 2. As the camera goes up, the score keeps on increasing with the time span of few seconds. 3. On level Completion, the camera follows the position of player. |
| 2. | Player.cs | 1. It controls the position of the player. Player can move left, right, up using leftArrow, rightarrow and uparrow respectively. 2. It makes the player jump by adding a force in the specific direction. 3. Updates score on collision trigger with diff. type of boxes. 4. Checks if player is on the platform using linecast, only then jumping is allowed. 5. Also, an audio clip is added on jumping and collosion with box |
| 3. | Scroll.cs | 1. Movement of the background along with the player. It sets the speed for the scrolling. 2. Moves according to speed of flying player on level completion. |
| 4. | Camera1.cs | 1. Acquires the position of the camera and moves it accordingly. 2. As the camera goes up, the score keeps on increasing with the time span of few seconds 3. On level Completion, the camera follows the position of player. |
| 5. | Player1.cs | 1. It controls the position of the player. Player can move left, right, up using leftArrow, rightarrow and uparrow respectively. 2. It makes the player jump by adding a force in the specific direction. 3. Updates score on collision trigger with diff. type of boxes. 4. Checks if player is on the platform using linecast, only then jumping is allowed. 5. Also, an audio clip is added on jumping and collosion with box |
| 6. | Gameover.cs | 1. When player collide with spike or jump between the hills 2. Destroys obsolete platform when it goes out of game play. |

**TABLE OF ANIMATIONS AND THEIR ANIMATION CONTROLLERS**

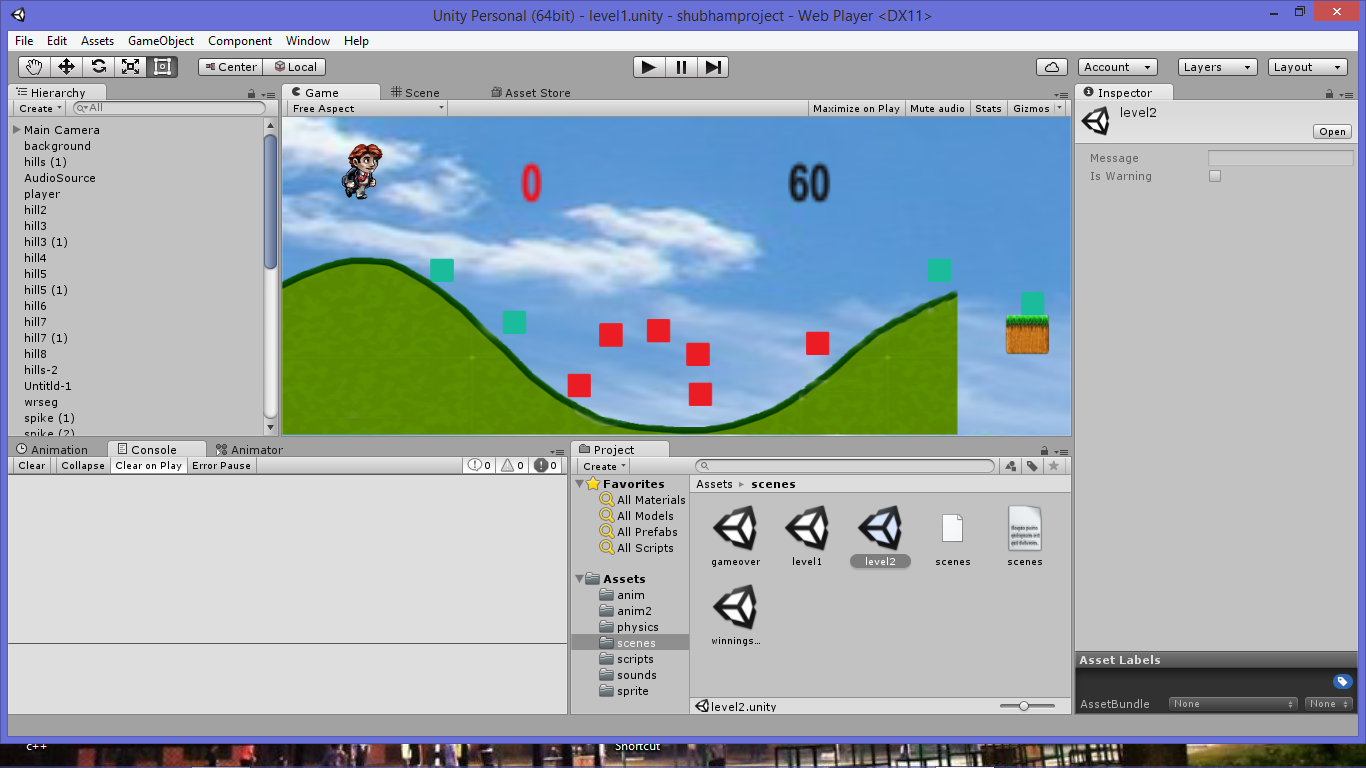
|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No | Animator Controller | Animation Names | Used for |
| 1. | Animcontroller | Initial animation | Player flips in lateral direction.  Trigger: landed |
| Jump animation | This animation works when the player jumps. its face design varies while in jumping by flipping sprite laterally.  Trigger: Jumped |

**STATE DIAGRAM**

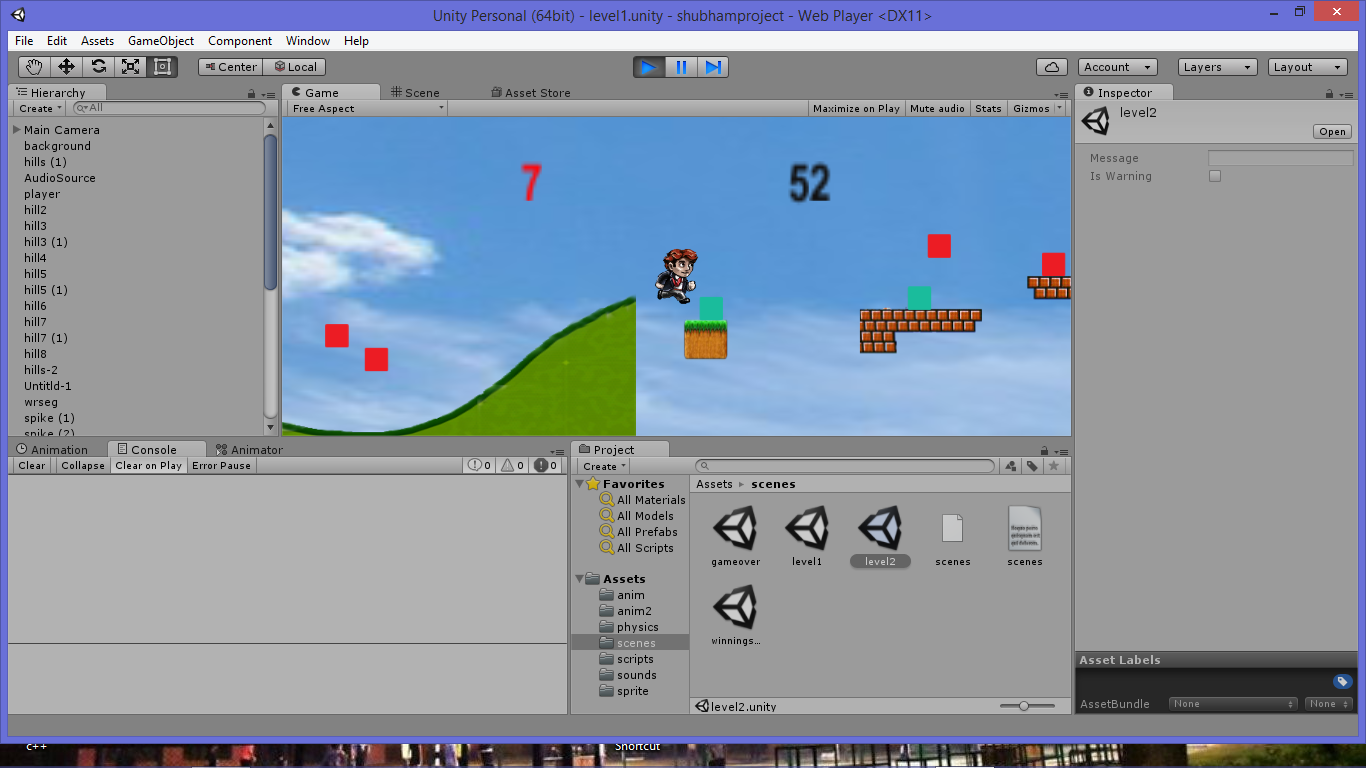
Jump

Initial

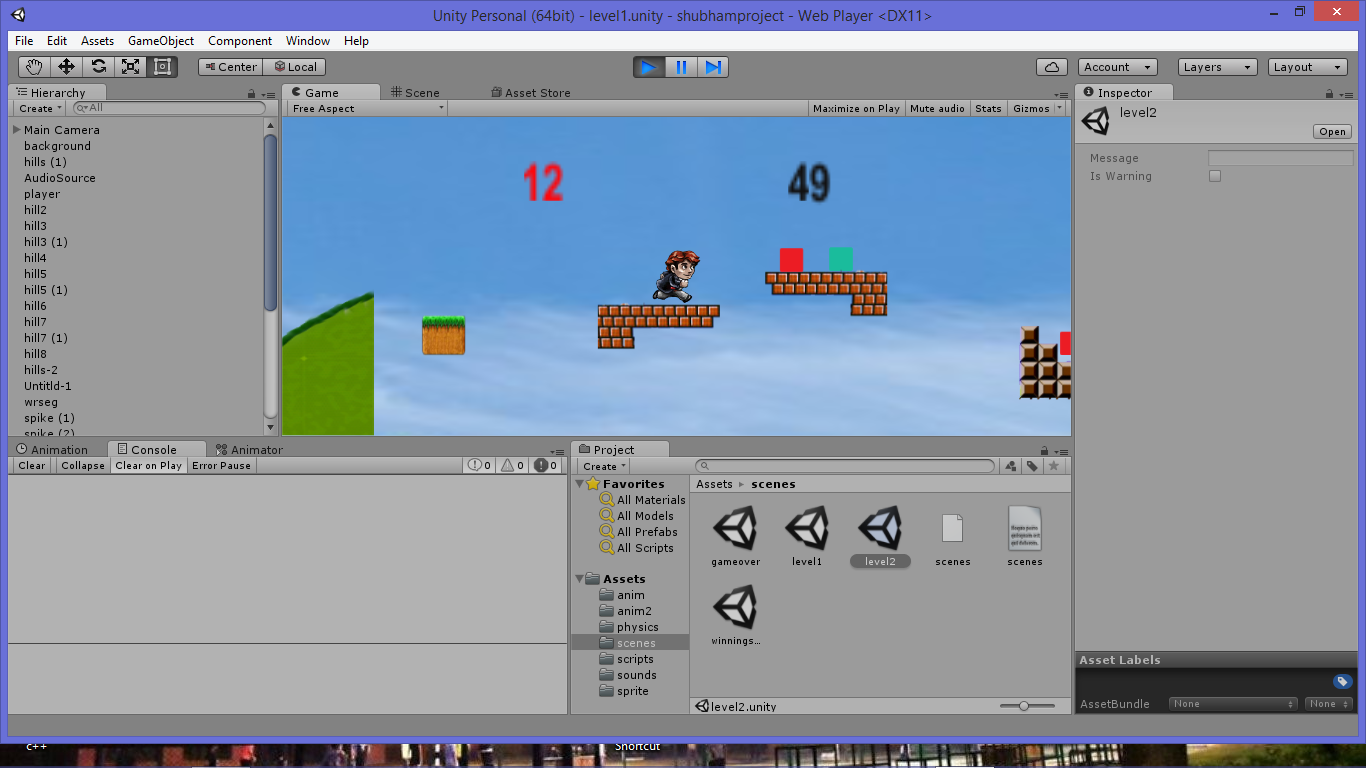
**STAGES OF GAME**



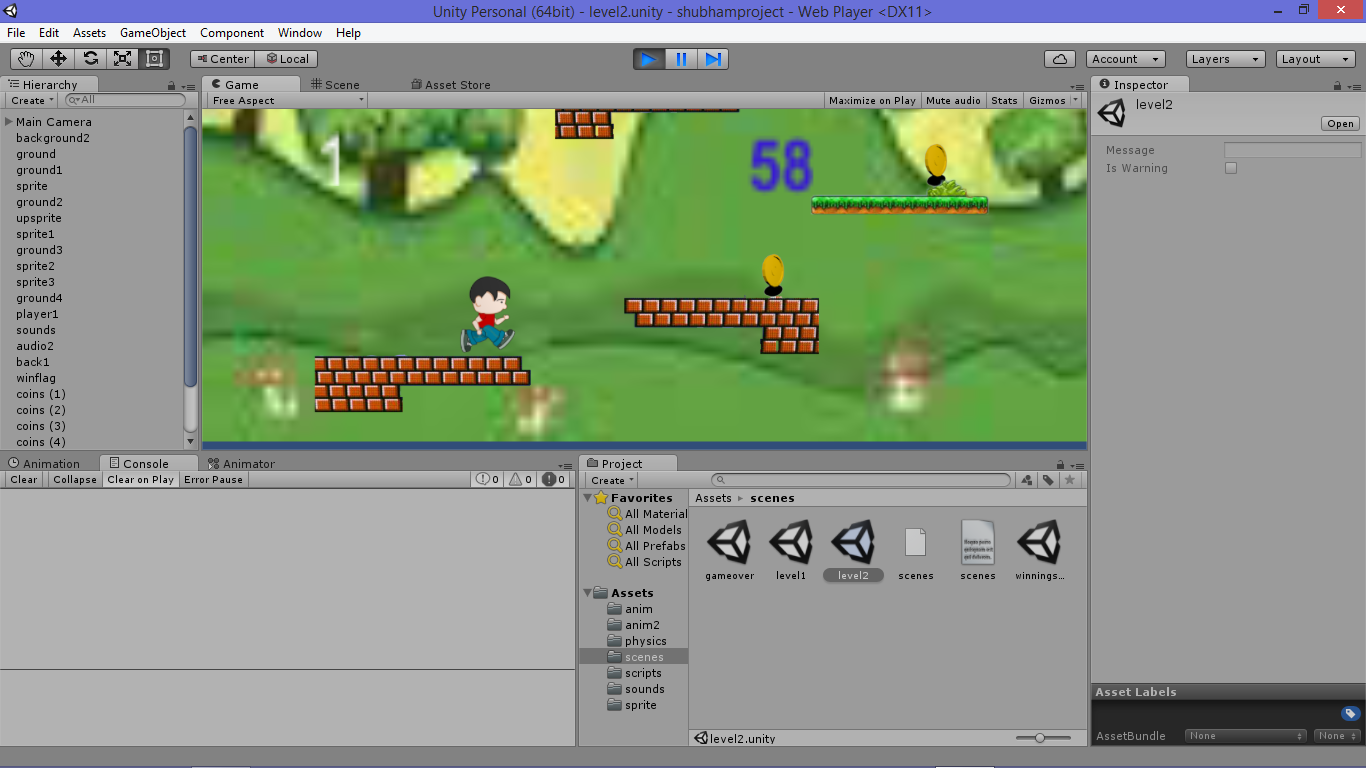
Game Starts.



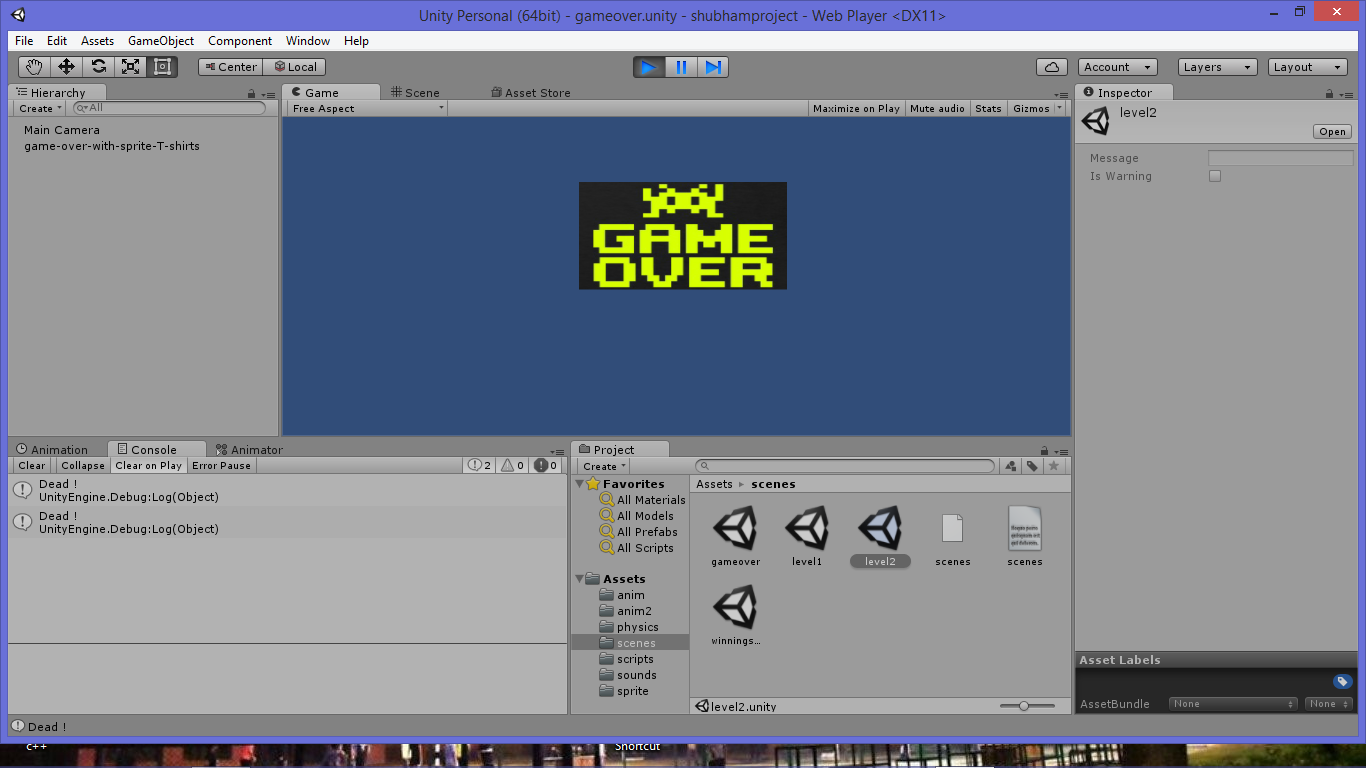
Gameplay1



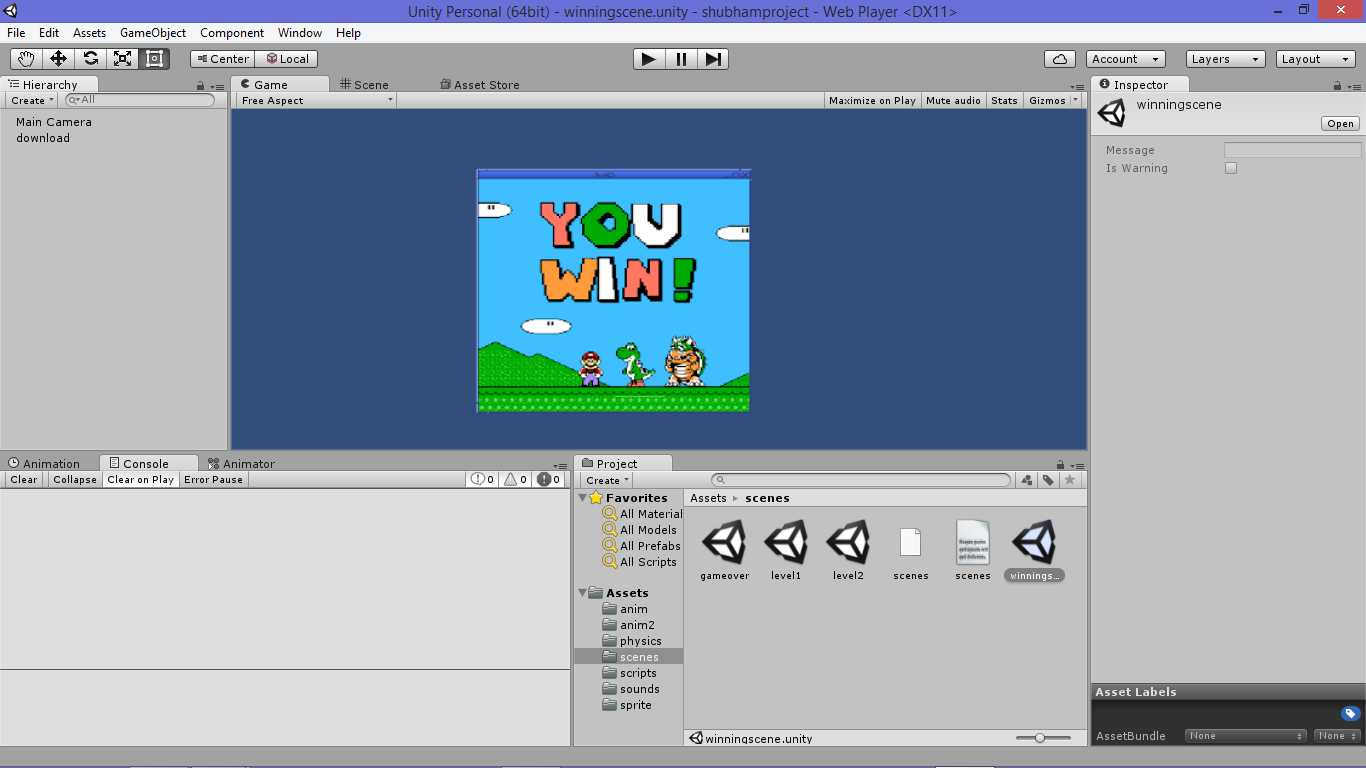
Gameplay2



Gameover screen



Winning:



**Number of levels:- 2**

**TEST PARAMETERS:**

1. Lagging of game (Whether game lags after certain time)

Result: No observable lag after an hour of gameplay.

1. Animation Control (Smoothness of change in animation)

Result: Animation change not so smooth.

1. Consistency of Colors

Result: No discolororation in 1 hour of gameplay.

1. Cpu power required

Result: No major suckup of power by the game.

1. Randomization of Objects

Result: Randomisation as expected.

1. Firing of triggers on collision

Result: a) Smooth firing on collision with fruits.

b) Minor inconsistency on health reduction on jumping on red platform.

1. Calculation of score

Result: Score is calculated as expected.

**GAME OPTIMISATION:** The platforms are destroyed as soon as they leave gamescreen. 2D sprites consumes least memory.

**References used for the game:-**

* [www.unity3d.com](http://www.unity3d.com)- study material and scripts
* [www.youtube.com](http://www.youtube.com) – referred to for video lessons
* [www.google.com](http://www.google.com) - sprites of kung-fu panda
* ansers.unity3d.com

**APPENDIX:1**

ALL CODES

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No | Script Files(Names) | Codes | Usage |
| 1. | camera.cs | using UnityEngine;  using System.Collections;  public class camera : MonoBehaviour {  [SerializeField]  private float xMax;  [SerializeField]  private float yMax;  [SerializeField]  private float xMin;  [SerializeField]  private float yMin;  private Transform target;  void Start () {  target = GameObject.Find ("player").transform;  }    // Update is called once per frame  void Update () {  transform.position = new Vector3(Mathf.Clamp (target.position.x, xMin, xMax), Mathf.Clamp (target.position.y, yMin, yMax),transform.position.z);}}  using UnityEngine;  using System.Collections;  using UnityEngine.SceneManagement;  public class gameover : MonoBehaviour {  // Use this for initialization  void Start () {  }  // Update is called once per frame  void OnGUI () {  GUI.contentColor = Color.red;  if (GUI.Button (new Rect (Screen.width / 2-50, Screen.height / 2+10, 80, 40), "Retry ?"))  SceneManager.LoadScene (2);  if(GUI.Button(new Rect(Screen.width/2-50,Screen.height/2+50,80,40),"Quit "))  Application.Quit();  }  } | 1. Acquires the position of the camera and moves it accordingly. 2. As the camera goes up, the score keeps on increasing with the time span of few |
| 2. | Player.cs | using UnityEngine;  using System.Collections;  public class player : MonoBehaviour {  public AudioSource audioSource;  public AudioClip Jump;  public AudioClip Coin;  public AudioClip Win;  public AudioClip gamesound;  public float speed;  public float jump;  float moveVelocity;  bool grounded ;  public int coins;  public float timeRemaining=60;  public TextMesh score;  public TextMesh time;  void Update () {  timeRemaining -= Time.deltaTime;  time.text = (int)timeRemaining+"";  if (timeRemaining <= 0) {  Application.LoadLevel (2);      }  if (Input.GetKey (KeyCode.RightArrow)) {  transform.localScale = new Vector2 (14, 14);  transform.Translate (Vector2.right \* 30f \* Time.deltaTime);  }  if (Input.GetKey (KeyCode.LeftArrow)) {  transform.localScale = new Vector2 (-14, 14);  transform.Translate (Vector2.right \* -30f \* Time.deltaTime);  }  if (Input.GetKey (KeyCode.UpArrow) || Input.GetKeyDown (KeyCode.W)) {  if (grounded) {  GetComponent<Rigidbody2D> ().velocity = new Vector2 (GetComponent<Rigidbody2D> ().velocity.x, jump);  audioSource.clip = Jump;  audioSource.Play ();  }  }  }  void OnTriggerExit2D(){  grounded = false;  }  void OnCollisionStay2D(Collision2D coll) {  if (coll.gameObject.tag == "level2") {  //coll.gameObject.SendMessage("Enter in new level");  Application.LoadLevel(1);    }  }  void OnTriggerEnter2D(Collider2D other)  {  if(other.tag=="Coin")  {  coins++;  score.text=coins + "";  Destroy(other.gameObject);  audioSource.clip=Coin;  audioSource.Play();  }  else if(other.tag== "spikes" )  {  Debug.Log("Dead !");  Application.LoadLevel(2);  }  else if(other.tag== "gameover" )  {    Application.LoadLevel(2);  }  else  {  if (grounded == true)  grounded = false;  else  grounded = true;  }}} | 1. It controls the position of the player. Player can move left, right, up using ‘A’, ‘D’ and ‘Space’ respectively. 2. It makes the player jump by adding a force in the specific direction. 3. Updates score on collision trigger with diff. kind of food items. 4. Decreases health on collision with red platform. 5. Checks if player is on the platform using linecast, only then jumping is allowed.   6.Also, an audio clip is added on jumping and eating of food. |
|  |  |  |  |
| 3 | gameover.cs | using UnityEngine;  using System.Collections;  using UnityEngine.SceneManagement;  public class gameover : MonoBehaviour {  // Use this for initialization  void Start () {  }  // Update is called once per frame  void OnGUI () {  GUI.contentColor = Color.red;  if (GUI.Button (new Rect (Screen.width / 2-50, Screen.height / 2+10, 80, 40), "Retry ?"))  SceneManager.LoadScene (2);  if(GUI.Button(new Rect(Screen.width/2-50,Screen.height/2+50,80,40),"Quit "))  Application.Quit();  }  } | 1. If the player falls, it tracks the lives and re-spawns accordingly to the middle of screen. 2. If health falls to zero it decreases life and replenishes health. 3. Displays Game Over when all lives are lost. 4. Restarts the game on players choice. |