Національний університет “Острозька академія”

Кафедра економіко-математичного моделювання та інформаційних технологій

**Дизайн документ**

**з дисципліни “Розробка ігрових додатків на Unity”**

Студента 3-го курсу

спеціальності КН-31

Демидюка М.Д.

Керівник:

Власюк В.В.

Острог 2020

**Мета**: при виконанні власного проекту, поглиблено розібрати сам рушій Unity, в розробці 2d проектів, та отримати практичні навички для подальшої розробки схожих проектів

**Тема**: Написати 2d проект, в якому присутні анімації, скріпти, своя фізика, і якийсь функціонал для гри.

Гра написана на рушії Unity, грати в неї можуть як дітлахи так і дорослі юзери

Сенс в тому щоб на реакції будувати вежу відповідно до позиції куба в дану секунду і дотримуватись основних рамок платформи

Ігровий процес полягає в тому наскільки довго ви зможете протриматись і на скільки високо зможете побудувати вежу.

Головним героєм виступає куб який ставиться на платформу по кліку користувача

Куб можна ставити від головного куба по вектору на старті у будь-яку з 8 вільних точок.

Основним функціоналом являється те що кожен наступний куб може поставитись на грані куба який був поставлений перед цим

Спеціальні об’єкти тут це сама платформа стартовий куб і потім кожен посілідуючий куб

Загальний опис:

Проект має назву **Cube Town**

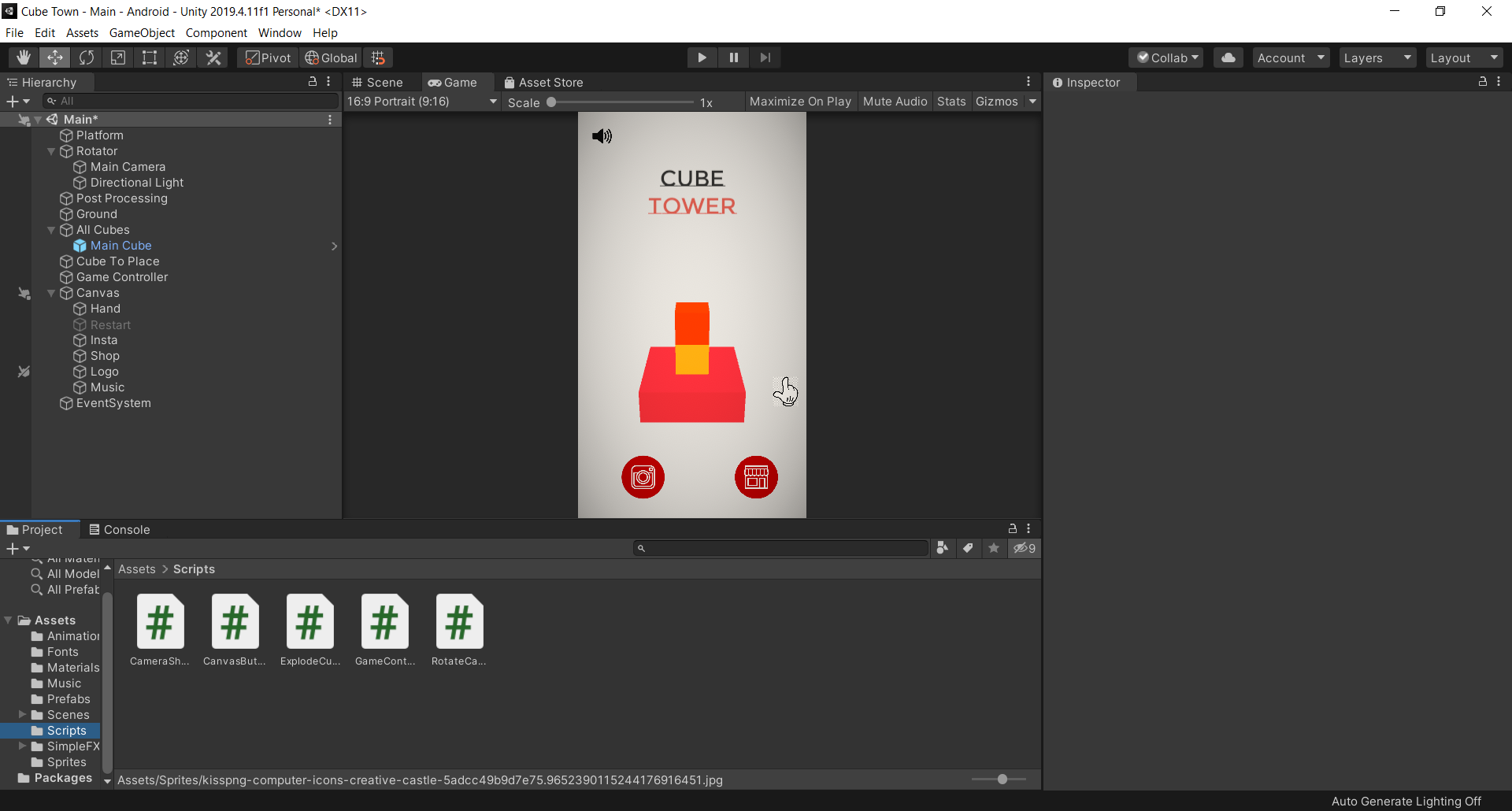
****

Рис 1. Cube Town.

Структура проекту

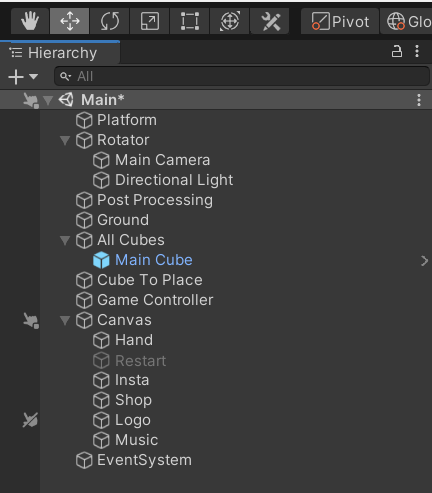


Рис 2. Структура Cube Town.

Папки з усіма файлами:

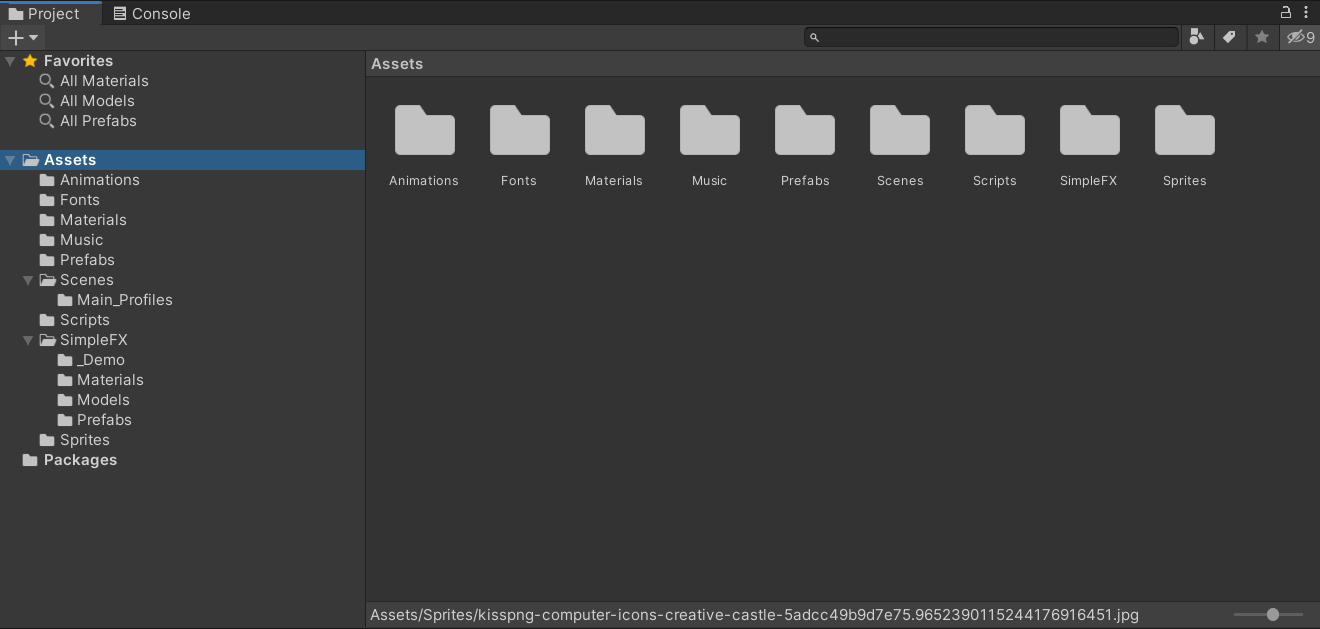


Рис 3. Папки Cube Town.

1. Анімації присутні для Тексту,(Назви проекту на старті гри на головному екрані), та для руки курсора який вказує на те, яким чином почати гру.
2. Зі шрифтів був використаний тільки один MontserratSubrayada-Regular, для Назви проекту.
3. Були використані такі матеріали для створення платформи, головного куба, куба який буде ставитись, та матеріал для поверхні землі.

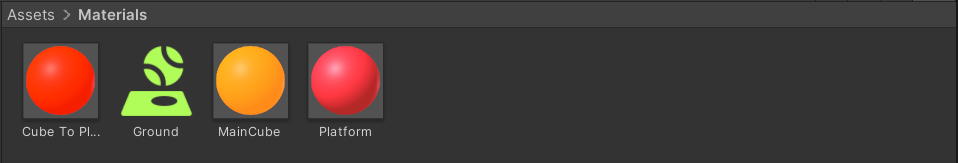


Рис 4. Матеріали Cube Town.

1. Музику додав для фону гри, для кнопок, ставлення нового куба, і коли все розпадається також є звук.

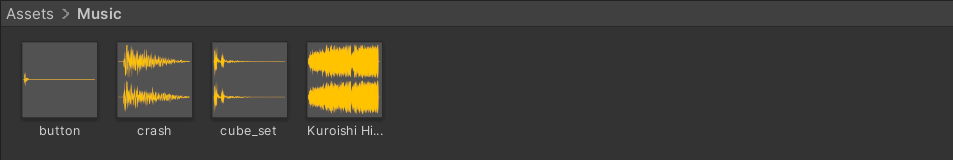


Рис 5. Музика Cube Town.

1. Скріпти для гри:

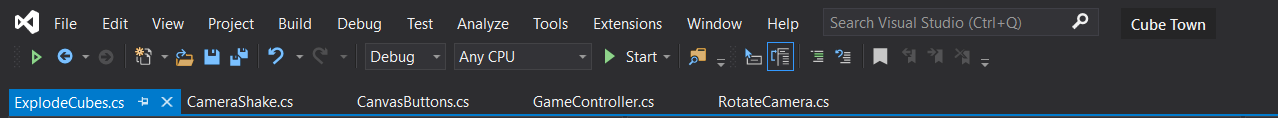


Рис 6. Скріпти Cube Town.

* 1. CameraShake –відповідає за трясіння камери в момент ставлення блоку;

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class NewBehaviourScript : MonoBehaviour  {  private Transform camTransform;  private float shakeDur = 1f, shakeAmount = 0.04f, decreaseFactor = 1.5f;  private Vector3 originPos;  private void Start()  {  camTransform = GetComponent<Transform>();  originPos = camTransform.localPosition;  }  private void Update()  {  if(shakeDur > 0)  {  camTransform.localPosition = originPos + Random.insideUnitSphere \* shakeAmount;  shakeDur -= Time.deltaTime \* decreaseFactor;  }  else  {  shakeDur = 0;  camTransform.localPosition = originPos;  }  }  } |

* 1. CanvasButtons –відповідає за функціонал натискання кнопок;

|  |
| --- |
| using UnityEngine;  using UnityEngine.UI;  using UnityEngine.SceneManagement;  public class CanvasButtons : MonoBehaviour  {  public Sprite musicOn, musicOff;  /\* private void Start()  {  if (PlayerPrefs.GetString("music") == "No" && gameObject.name == "Music");  GetComponent<Image>().sprite = musicOff;  }  \*/  public void RestartGame()  {  if (PlayerPrefs.GetString("music") == "No")  GetComponent<AudioSource>().Play();  SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);  }  public void LoadInstagram()  {  if (PlayerPrefs.GetString("music") == "No")  GetComponent<AudioSource>().Play();  Application.OpenURL("https://www.instagram.com/mishademydiuk/");  }  public void MusicWork()  { //музика на нуль  if(PlayerPrefs.GetString("music") == "No")  {  GetComponent<AudioSource>().Play();  PlayerPrefs.SetString("music","Yes");  GetComponent<Image>().sprite = musicOn;  }      else  {  PlayerPrefs.SetString("music", "No");  GetComponent<Image>().sprite = musicOff;  }  }  } |

* 1. ExplodeCubes –відповідає за умови розрухи кубової піраміди;

|  |
| --- |
| using UnityEngine;  using System.Collections;  using System.Collections.Generic;  using System;  using UnityEngine.SceneManagement;  public class ExplodeCubes : MonoBehaviour  {  public GameObject restartButton, explosion;  private bool \_collisionSet;// дає змогу виконувати код один раз навіть якщ ще чомусь додати тег Cube    private void OnCollisionEnter(Collision collision)  {  if(collision.gameObject.tag == "Cube" && !\_collisionSet)  {  for (int i = collision.transform.childCount - 1; i >= 0; i--)  {  Transform child = collision.transform.GetChild(i);  child.gameObject.AddComponent<Rigidbody>();  child.gameObject.GetComponent<Rigidbody>().AddExplosionForce(70f, Vector3.up, 5f);// розмах розкидування кубів  child.SetParent(null);  }  restartButton.SetActive(true);  Camera.main.transform.localPosition -= new Vector3(0, 0, 3f);  //Camera.main.gameObject.AddComponent<CameraShake>();  Instantiate(explosion, new Vector3(collision.contacts[0].point.x, collision.contacts[0].point.y, collision.contacts[0].point.z), Quaternion.identity);  if (PlayerPrefs.GetString("music") == "No")  GetComponent<AudioSource>().Play();  Destroy(collision.gameObject);  \_collisionSet = true;  }    }  } |

1.4 GameController –описуєтся основні умови при яких гра може продовжуватись, та описуються рамки того, як можна ставити куби;

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using System;  using UnityEngine;  using UnityEngine.EventSystems;  public class GameController : MonoBehaviour  {  private CubePos nowCube = new CubePos(0,1,0);  public float cubeChangePlaceSpeed = 0.5f;  public Transform cubeToPlace;  public GameObject cubeToCreate, allCubes, vfx;  private float camMoveToYPosition, camMoveSpeed = 2f;  public GameObject[] canvasStartPage;  private Rigidbody allCubesRb;  public Color[] bgcolors;  private Color toCameraColor;  private bool isLose, firstCube;  private List<Vector3> allCubePositions = new List<Vector3>  {  new Vector3(0,0,0),  new Vector3(1,0,0),  new Vector3(-1,0,0),  new Vector3(0,1,0),  new Vector3(0,0,1),  new Vector3(0,0,-1),  new Vector3(1,0,1),  new Vector3(-1,0,-1),  new Vector3(-1,0,1),  new Vector3(1,0,-1),  };  private int prevCountMaxHorzontal;  private Transform mainCam;  private Coroutine showCubePlace;    private void Start()  {  toCameraColor = Camera.main.backgroundColor;  mainCam = Camera.main.transform;  camMoveToYPosition = 5.9f + nowCube.y - 1f;  allCubesRb = allCubes.GetComponent<Rigidbody>();  showCubePlace = StartCoroutine(ShowCubePlace());  }    private void Update()  {  if((Input.GetMouseButtonDown(0) || Input.touchCount >0) && ( cubeToPlace != null) && allCubes != null && !EventSystem.current.IsPointerOverGameObject())  {  #if !UNITY\_EDITOR  if (Input.GetTouch(0).phase != TouchPhase.Began)  return;  #endif  if (!firstCube)  {  firstCube = true;  foreach (GameObject obj in canvasStartPage)  Destroy(obj);  }  GameObject newCube = Instantiate(  cubeToCreate,  cubeToPlace.position,  Quaternion.identity) as GameObject;  newCube.transform.SetParent(allCubes.transform);  nowCube.setVector(cubeToPlace.position);  allCubePositions.Add(nowCube.GetVector());  if (PlayerPrefs.GetString("music") == "No")  GetComponent<AudioSource>().Play();  Instantiate(vfx, newCube.transform.position, Quaternion.identity);  allCubesRb.isKinematic = true;  allCubesRb.isKinematic = false;  SpawnPositions();  MoveCameraChangeBg();    }  if (!isLose && allCubesRb.velocity.magnitude > 0.1f)  {  Destroy(cubeToPlace.gameObject);  isLose = true;  StopCoroutine(showCubePlace);  }  mainCam.localPosition = Vector3.MoveTowards(mainCam.localPosition,  new Vector3(mainCam.localPosition.x, camMoveToYPosition, mainCam.localPosition.z),  camMoveSpeed \* Time.deltaTime);  if (Camera.main.backgroundColor != toCameraColor)  Camera.main.backgroundColor = Color.Lerp(Camera.main.backgroundColor, toCameraColor, Time.deltaTime / 1.5f);  }  IEnumerator ShowCubePlace()  {  while (true)  {  SpawnPositions();  yield return new WaitForSeconds(cubeChangePlaceSpeed);    }  }  private void SpawnPositions()  {  List<Vector3> positions = new List<Vector3>();  if (IsPositionEmpty(new Vector3(nowCube.x + 1, nowCube.y, nowCube.z))  && nowCube.x + 1 != cubeToPlace.position.x)  {  positions.Add(new Vector3(nowCube.x + 1, nowCube.y, nowCube.z));  }  if (IsPositionEmpty(new Vector3(nowCube.x - 1, nowCube.y, nowCube.z))  && nowCube.x - 1 != cubeToPlace.position.x)  {  positions.Add(new Vector3(nowCube.x - 1, nowCube.y, nowCube.z));  }  if (IsPositionEmpty(new Vector3(nowCube.x, nowCube.y + 1, nowCube.z))  && nowCube.y + 1 != cubeToPlace.position.y)  {  positions.Add(new Vector3(nowCube.x, nowCube.y + 1, nowCube.z));  }  if (IsPositionEmpty(new Vector3(nowCube.x, nowCube.y - 1, nowCube.z))  && nowCube.y - 1 != cubeToPlace.position.y)  {  positions.Add(new Vector3(nowCube.x, nowCube.y - 1, nowCube.z));  }  if (IsPositionEmpty(new Vector3(nowCube.x, nowCube.y, nowCube.z + 1))  && nowCube.z + 1 != cubeToPlace.position.z)  {  positions.Add(new Vector3(nowCube.x, nowCube.y, nowCube.z + 1));  }  if (IsPositionEmpty(new Vector3(nowCube.x, nowCube.y, nowCube.z - 1))  && nowCube.z - 1 != cubeToPlace.position.z)  {  positions.Add(new Vector3(nowCube.x, nowCube.y, nowCube.z - 1));  }  if (positions.Count > 1)  cubeToPlace.position = positions[UnityEngine.Random.Range(0, positions.Count)];  else if (positions.Count == 0)  isLose = true;  else  cubeToPlace.position = positions[0];  }  private bool IsPositionEmpty(Vector3 targetPos)  {  if(targetPos.y == 0)  {  return false;  }  foreach(Vector3 pos in allCubePositions)  {  if (pos.x == targetPos.x && pos.y == targetPos.y && pos.z == targetPos.z)  return false;  }  return true;  }  private void MoveCameraChangeBg()  {  int maxX = 0, maxY = 0, maxZ = 0, maxHor;  foreach (Vector3 pos in allCubePositions)  {  if (Mathf.Abs(Convert.ToInt32(pos.x)) > maxX)  maxX = Convert.ToInt32(pos.x);  if (Convert.ToInt32(pos.y) > maxY)  maxY = Convert.ToInt32(pos.y);  if (Mathf.Abs(Convert.ToInt32(pos.z)) > maxZ)  maxZ = Convert.ToInt32(pos.z);  }  camMoveToYPosition = 5.9f + nowCube.y - 1f;  maxHor = maxX > maxZ ? maxX : maxZ;  if(maxHor % 3 == 0 && prevCountMaxHorzontal !=maxHor )  {  mainCam.localPosition -= new Vector3(0,0,2.5f);  prevCountMaxHorzontal = maxHor;  }  if(maxY >= 7)  toCameraColor = bgcolors[2];  else if (maxY >= 5)  toCameraColor = bgcolors[1];  else if (maxY >= 2)  toCameraColor = bgcolors[0];  }  }  struct CubePos  {  public int x, y, z;    public CubePos(int x, int y, int z)  {  this.x = x;  this.y = y;  this.z = z;  }  public Vector3 GetVector()  {  return new Vector3(x, y, z);  }  public void setVector(Vector3 pos)  {  x = Convert.ToInt32(pos.x);  y = Convert.ToInt32(pos.y);  z = Convert.ToInt32(pos.z);  }  } |

1.5 RotateCamera –відповідає за те щоб камера крутилась довкола платформи;

|  |
| --- |
| using UnityEngine;  public class RotateCamera : MonoBehaviour  {  public float speed = 5f;  private Transform \_rotator;  private void Start()  {  \_rotator = GetComponent<Transform>();  }  private void Update()  {  \_rotator.Rotate(0,speed \* Time.deltaTime, 0);  }  } |

1. Спрайти для вигляду кнопок та курсора;
2. Довстановив SimpleFX prefabs –для ігрових ефектів
3. Проблеми з якими зіткнувся:

Під час виконання проекту зіткнувся з великою кількість іноді простих, а іноді не дужу простих проблем, які потребували іноді 3-4 години, щоб зрозуміти в чому саме проблеми і як її вирішити.