using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class AddRoom : MonoBehaviour

{

[Header("Walls")]

public GameObject[] walls;

public GameObject door;

[Header("Walls")]

public GameObject portal;

[Header("Enemies")]

public GameObject[] enemyTypes;

public GameObject healthPotion;

public Transform[] enemySpawner;

[HideInInspector] public List<GameObject> enemies;

[HideInInspector] public bool isBossRoom;

[HideInInspector] public bool isBossDefeated;

[HideInInspector] public bool isLastRoom;

private RoomVariants variants;

private bool spawned;

private bool wallsDestroyed;

public RoomVariants RoomVariants

{

get => default;

set

{

}

}

public Player Player

{

get => default;

set

{

}

}

public Enemy Enemy

{

get => default;

set

{

}

}

private void Awake()

{

variants = GameObject.FindGameObjectWithTag("Rooms").GetComponent<RoomVariants>();

}

private void Start()

{

variants.rooms.Add(gameObject);

}

private void OnTriggerEnter2D(Collider2D other)

{

if(other.CompareTag("Player") && !spawned)

{

spawned = true;

if (!isBossRoom && gameObject.name != "MainRoom" && gameObject.name != "MainRoom(Clone)")

{

foreach (Transform spawner in enemySpawner)

{

int rand = Random.Range(0, 10);//до 11

if (rand < 9)

{

GameObject enemyType = enemyTypes[Random.Range(0, enemyTypes.Length)];

GameObject enemy = Instantiate(enemyType, spawner.position, Quaternion.identity) as GameObject;

enemy.transform.SetParent(gameObject.transform);

enemies.Add(enemy);

}

else if (rand == 9)

{

Instantiate(healthPotion, spawner.position, Quaternion.identity);

}

/\*else if (rand == 10)

{

Instantiate(shield, spawner.position, Quaternion.identity);

}\*/

}

}

else if (isBossRoom && gameObject.name != "MainRoom" && gameObject.name != "MainRoom(Clone)")

{

int rand = Random.Range(0, enemySpawner.Length);

GameObject enemy = Instantiate(enemyTypes[0], enemySpawner[rand].position, Quaternion.identity) as GameObject;

enemy.GetComponent<Enemy>().isBoss = true;

enemy.GetComponent<Enemy>().health \*= 2;

enemy.GetComponent<Enemy>().speed = 1;

enemy.GetComponent<Enemy>().damage = 2;

enemy.GetComponent<Enemy>().isBoss = true;

enemy.transform.localScale = new Vector3((float)(gameObject.transform.localScale.x \* 0.5), (float)(gameObject.transform.localScale.y \* 0.5));

enemy.transform.SetParent(gameObject.transform);

enemies.Add(enemy);

}

else

{

DestroyWalls();

}

StartCoroutine(CheckEnemies());

} else if (other.CompareTag("Player") && spawned)

{

foreach(GameObject enemy in enemies)

{

enemy.GetComponent<Enemy>().playerNotInRoom = false;

}

}

}

IEnumerator CheckEnemies()

{

yield return new WaitForSeconds(1f);

yield return new WaitUntil(() => enemies.Count == 0);

if (isBossRoom && isBossDefeated || isLastRoom)

portal.SetActive(true);

DestroyWalls();

}

public void DestroyWalls()

{

foreach(GameObject wall in walls)

{

if(wall != null && wall.transform.childCount != 0)

{

Destroy(wall);

}

}

wallsDestroyed = true;

}

private void OnTriggerStay2D(Collider2D other)

{

if(wallsDestroyed && other.CompareTag("Wall"))

{

Destroy(other.gameObject);

}

if (other.CompareTag("Room"))

{

Destroy(other.gameObject);

Debug.Log("Комната уничтожена");

}

}

private void OnTriggerExit2D(Collider2D other)

{

if (other.CompareTag("Player") && spawned)

{

foreach (GameObject enemy in enemies)

{

enemy.GetComponent<Enemy>().playerNotInRoom = true;

}

}

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Bullet : MonoBehaviour

{

public float speed;

public float lifetime;

public float distance;

public int damage;

public LayerMask whatIsSolid;

public GameObject destroyEffect;

[SerializeField] bool enemyBullet;

public Gun Gun

{

get => default;

set

{

}

}

public Enemy Enemy

{

get => default;

set

{

}

}

private void Start()

{

damage = PlayerPrefs.GetInt("damage");

Invoke("DestroyBullet", lifetime);

}

// Update is called once per frame

void Update()

{

RaycastHit2D hitInfo = Physics2D.Raycast(transform.position, transform.up, distance, whatIsSolid);

if(hitInfo.collider != null)

{

if(hitInfo.collider.CompareTag("Enemy"))

hitInfo.collider.GetComponent<Enemy>().TakeDamage(damage);

if (hitInfo.collider.CompareTag("Player") && enemyBullet)

hitInfo.collider.GetComponent<Player>().ChangeHealth(-damage);

DestroyBullet();

}

transform.Translate(Vector2.up \* speed \* Time.deltaTime);

}

public void DestroyBullet()

{

Instantiate(destroyEffect, transform.position, Quaternion.identity);

Destroy(gameObject);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class ChangeRoom : MonoBehaviour

{

public Vector3 cameraChangePos;

public Vector3 playerChangePos;

private Camera cam;

public Player Player

{

get => default;

set

{

}

}

private void Start()

{

cam = Camera.main.GetComponent<Camera>();

}

private void OnTriggerEnter2D(Collider2D other)

{

if (other.CompareTag("Player"))

{

other.transform.position += playerChangePos;

cam.transform.position += cameraChangePos;

}

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class DeadEnd : MonoBehaviour

{

public MainMenu MainMenu

{

get => default;

set

{

}

}

private void Update()

{

Time.timeScale = 0f;

}

public void LoadMenu()

{

Time.timeScale = 1f;

SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);

}

public void Restart()

{

Time.timeScale = 1f;

SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Enemy : MonoBehaviour

{

private float stopTime;

public float startStopTime;

private Animator anim;

private Player player;

private AddRoom room;

public GameObject destroyEffect;

private float timeBtwAttack;

public float startTimeBtwAttack;

public float speed;

public int health;

public int damage;

[HideInInspector] public bool playerNotInRoom;

[HideInInspector] public bool isBoss;

private bool stopped;

public Player Player

{

get => default;

set

{

}

}

public AddRoom AddRoom

{

get => default;

set

{

}

}

// Start is called before the first frame update

void Start()

{

anim = GetComponent<Animator>();

player = FindObjectOfType<Player>();

room = GetComponentInParent<AddRoom>();

stopped = false;

}

// Update is called once per frame

void Update()

{

if (!playerNotInRoom)

{

if (stopTime <= 0)

stopped = false;

else

{

stopped = true;

stopTime -= Time.deltaTime;

}

}

else

{

stopped = true;

}

if (room == null)

{

Debug.Log("Room is Null");//Попробуй присваивать комнату волкам при спавне

}

if (health <= 0)

{

room.enemies.Remove(gameObject);

if (isBoss)

{

room.isBossDefeated = true;

player.ChangePoints(25);

} else

{

player.ChangePoints(5);

}

Destroy(gameObject);

}

if (player.transform.position.x > transform.position.x)

transform.eulerAngles = new Vector3(0, 180, 0);

else

transform.eulerAngles = new Vector3(0, 0, 0);

if(!stopped)

transform.position = Vector2.MoveTowards(transform.position, player.transform.position, speed \* Time.deltaTime);

}

public void TakeDamage(int damage)

{

stopTime = startStopTime;

health -= damage;

}

private void OnTriggerStay2D(Collider2D other)

{

if (other.CompareTag("Player"))

{

if (timeBtwAttack <= 0)

{

anim.SetTrigger("attack");

}

else

{

timeBtwAttack -= Time.deltaTime;

}

}

}

public void OnEnemyAttack()

{

Instantiate(destroyEffect, player.transform.position, Quaternion.identity);

player.ChangeHealth(-damage);

timeBtwAttack = startTimeBtwAttack;

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Gun : MonoBehaviour

{

public GameObject bullet;

public Transform shotPoint;

public GunType gunType;

public float startTimeBtwShots;

public float offset;

public enum GunType { Default, Enemy }

private float timeBtwShots;

private float rotZ;

private Vector3 difference;

private Player player;

// Start is called before the first frame update

void Start()

{

player = GameObject.FindGameObjectWithTag("Player").GetComponent<Player>();

}

// Update is called once per frame

void Update()

{

if (gunType == GunType.Default)

{

difference = Camera.main.ScreenToWorldPoint(Input.mousePosition) - transform.position;

rotZ = Mathf.Atan2(difference.y, difference.x) \* Mathf.Rad2Deg;

}

else if (gunType == GunType.Enemy)

{

difference = player.transform.position - transform.position;

rotZ = Mathf.Atan2(difference.y, difference.x) \* Mathf.Rad2Deg;

}

transform.rotation = Quaternion.Euler(0f, 0f, rotZ + offset);

if (timeBtwShots <= 0)

{

if (Input.GetMouseButton(0) || gunType == GunType.Enemy)

Shoot();

} else

{

timeBtwShots -= Time.deltaTime;

}

}

public void Shoot()

{

Instantiate(bullet, shotPoint.position, shotPoint.rotation);

timeBtwShots = startTimeBtwShots;

}

public Player Player

{

get => default;

set

{

}

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class InputComponent : MonoBehaviour

{

[SerializeField] private Text \_buttonText; // текст кнопки

[SerializeField] private string \_defaultKeyName; // ключ/имя вызова

[SerializeField] private KeyCode \_defaultKeyCode; // клавиша ключа

public KeyCode keyCode { get; set; }

public bool isUp;

public bool isDown;

public bool isLeft;

public bool isRight;

private IEnumerator coroutine;

private string tmpKey;

public Text buttonText

{

get { return \_buttonText; }

}

public string defaultKeyName

{

get { return \_defaultKeyName; }

}

public KeyCode defaultKeyCode

{

get { return \_defaultKeyCode; }

}

public void ButtonSetKey() // событие кнопки, для перехода в режим ожидания

{

tmpKey = \_buttonText.text;

\_buttonText.text = "???";

coroutine = Wait();

StartCoroutine(coroutine);

}

// ждем, когда игрок нажмет какую-нибудь клавишу, для привязки

// если будет нажата клавиша 'Escape', то отмена

IEnumerator Wait()

{

while (true)

{

yield return null;

if (Input.GetKeyDown(KeyCode.Escape))

{

\_buttonText.text = tmpKey;

StopCoroutine(coroutine);

}

foreach (KeyCode k in KeyCode.GetValues(typeof(KeyCode)))

{

if (Input.GetKeyDown(k) && !Input.GetKeyDown(KeyCode.Escape))

{

keyCode = k;

if (isUp) PlayerPrefs.SetInt("up", (int)k);

if (isDown) PlayerPrefs.SetInt("down", (int)k);

if (isLeft) PlayerPrefs.SetInt("left", (int)k);

if (isRight) PlayerPrefs.SetInt("right", (int)k);

PlayerPrefs.SetInt("OptionsChanged", 1);

\_buttonText.text = k.ToString();

StopCoroutine(coroutine);

}

}

}

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine.SceneManagement;

using UnityEngine;

public class MainMenu : MonoBehaviour

{

public RoomSpawner RoomSpawner

{

get => default;

set

{

}

}

public void PlayGame()

{

SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);

}

public void ExitGame()

{

Debug.Log("Exit");

Application.Quit();

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class options : MonoBehaviour

{

public GameObject upBtn;

public GameObject downBtn;

public GameObject leftBtn;

public GameObject rightBtn;

public GameObject resetButton;

private void Start()

{

upBtn.GetComponentInChildren<Text>().text = ((KeyCode)PlayerPrefs.GetInt("up")).ToString();

downBtn.GetComponentInChildren<Text>().text = ((KeyCode)PlayerPrefs.GetInt("down")).ToString();

leftBtn.GetComponentInChildren<Text>().text = ((KeyCode)PlayerPrefs.GetInt("left")).ToString();

rightBtn.GetComponentInChildren<Text>().text = ((KeyCode)PlayerPrefs.GetInt("right")).ToString();

}

private void Update()

{

if (PlayerPrefs.GetInt("OptionsChanged") == 1)

{

resetButton.SetActive(true);

} else

{

resetButton.SetActive(false);

}

}

public void resetOptions()

{

PlayerPrefs.SetInt("up", PlayerPrefs.GetInt("defUp"));

upBtn.GetComponentInChildren<Text>().text = "W";

PlayerPrefs.SetInt("down", PlayerPrefs.GetInt("defDown"));

downBtn.GetComponentInChildren<Text>().text = "S";

PlayerPrefs.SetInt("left", PlayerPrefs.GetInt("defLeft"));

leftBtn.GetComponentInChildren<Text>().text = "A";

PlayerPrefs.SetInt("right", PlayerPrefs.GetInt("defRight"));

rightBtn.GetComponentInChildren<Text>().text = "D";

PlayerPrefs.SetInt("OptionsChanged", 0);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class PauseMenu : MonoBehaviour

{

public static bool isPaused = false;

public GameObject pauseMenuUI;

public MainMenu MainMenu

{

get => default;

set

{

}

}

// Update is called once per frame

void Update()

{

if (Input.GetKeyUp(KeyCode.Escape))

if (isPaused)

Resume();

else

Pause();

}

public void Resume()

{

pauseMenuUI.SetActive(false);

Time.timeScale = 1f;

isPaused = false;

}

public void Restart()

{

SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);

}

public void LoadMenu()

{

SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);

}

void Pause()

{

pauseMenuUI.SetActive(true);

Time.timeScale = 0f;

isPaused = true;

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

using UnityEngine.UI;

public class Player : MonoBehaviour

{

private CircleCollider2D controller;

public float speed;

public int health;

public int armor = 0;

public int maxHealth;

public int upPoints = 0;

public Text healthDisplay;

public Text level;

public Text floor;

private Rigidbody2D rb;

private Vector2 moveInput;

private Vector2 moveVelocity;

private Animator anim;

private bool facingRight = true;

private bool keyButtonPushed;

private int levelNum;

private int floorNum;

public GameObject mainRoom;

public GameObject Room;

public GameObject deadScreen;

public GameObject HUD;

[Header("Key")]

public GameObject keyIcon;

[Header("Controls")]

public KeyCode up;

public KeyCode down;

public KeyCode left;

public KeyCode right;

private bool isControlsChanged = false;

public RoomVariants RoomVariants

{

get => default;

set

{

}

}

public PauseMenu PauseMenu

{

get => default;

set

{

}

}

public DeadEnd DeadEnd

{

get => default;

set

{

}

}

/\*public GameObject portalIcon;\*/

// Start is called before the first frame update

void Start()

{

if (PlayerPrefs.GetInt("maxHealth") <= maxHealth)

{

PlayerPrefs.SetInt("maxHealth", maxHealth);

}

health = maxHealth;

controller = GetComponent<CircleCollider2D>();

if (PlayerPrefs.GetInt("OptionsChanged") == 0)

isControlsChanged = false;

else

isControlsChanged = true;

if (PlayerPrefs.GetInt("armor") == PlayerPrefs.GetInt("defArmor"))

armor = PlayerPrefs.GetInt("defArmor");

else

armor = PlayerPrefs.GetInt("armor");

upPoints = PlayerPrefs.GetInt("points") != 0 || PlayerPrefs.GetInt("points") != null ? PlayerPrefs.GetInt("points") : 0;

PlayerPrefs.SetInt("defDamage", 1);

if (PlayerPrefs.GetInt("damage") <= PlayerPrefs.GetInt("defDamage"))

PlayerPrefs.SetInt("damage", PlayerPrefs.GetInt("defDamage"));

PlayerPrefs.SetInt("defUp", (int)KeyCode.W);

PlayerPrefs.SetInt("defDown", (int)KeyCode.S);

PlayerPrefs.SetInt("defLeft", (int)KeyCode.A);

PlayerPrefs.SetInt("defRight", (int)KeyCode.D);

if (!isControlsChanged)

{

PlayerPrefs.SetInt("up", PlayerPrefs.GetInt("defUp"));

PlayerPrefs.SetInt("down", PlayerPrefs.GetInt("defDown"));

PlayerPrefs.SetInt("left", PlayerPrefs.GetInt("defLeft"));

PlayerPrefs.SetInt("right", PlayerPrefs.GetInt("defRight"));

}

up = (KeyCode)PlayerPrefs.GetInt("up");

down = (KeyCode)PlayerPrefs.GetInt("down");

left = (KeyCode)PlayerPrefs.GetInt("left");

right = (KeyCode)PlayerPrefs.GetInt("right");

rb = GetComponent<Rigidbody2D>();

anim = GetComponent<Animator>();

levelNum = int.Parse(level.text);

floorNum = int.Parse(floor.text);

Time.timeScale = 1f;

}

// Update is called once per frame

void Update()

{

healthDisplay.text = health.ToString();

if (controller.isActiveAndEnabled)

{

int hor;

int ver;

if (Input.GetKey(up))

{

hor = 1;

} else if (Input.GetKey(down))

{

hor = -1;

} else

{

hor = 0;

}

if (Input.GetKey(right))

{

ver = 1;

}

else if (Input.GetKey(left))

{

ver = -1;

}

else

{

ver = 0;

}

moveInput = new Vector2(ver, hor);

moveVelocity = moveInput.normalized \* speed;

if(moveInput.x == 0)

anim.SetBool("isRunning", false);

else

anim.SetBool("isRunning", true);

if (!facingRight && moveInput.x>0)

Flip();

else if(facingRight && moveInput.x<0)

Flip();

}

if (health <= 0)

{

deadScreen.SetActive(true);

HUD.SetActive(false);

}

}

private void FixedUpdate()

{

rb.MovePosition(rb.position + moveVelocity \* Time.fixedDeltaTime);

}

private void Flip()

{

facingRight = !facingRight;

Vector3 Scaler = transform.localScale;

Scaler.x \*= -1;

transform.localScale = Scaler;

}

public void ChangeHealth(int healthValue)

{

if (healthValue < 0)

{

if (healthValue \* -1 - armor <= 0)

health += 0;

else

health += (healthValue + armor);

Debug.Log("Отнимаем здоровье: " + healthValue);

}

else

{

health += healthValue;

Debug.Log("Добавляем здоровье: " + healthValue);

}

//health += healthValue + (healthValue < 0 ? (healthValue >= armor ? armor : healthValue) : 0);

if (health > maxHealth)

health = maxHealth;

}

public void ChangePoints(int points)

{

upPoints += points;

PlayerPrefs.SetInt("points", upPoints);

}

private void OnTriggerEnter2D(Collider2D other)

{

if (other.CompareTag("Key"))

{

keyIcon.SetActive(true);

Destroy(other.gameObject);

}

if (other.CompareTag("Heal"))

{

if (health < maxHealth)

{

ChangeHealth(3);

Destroy(other.gameObject);

}

}

}

public void onKeyButtonDown()

{

keyButtonPushed = !keyButtonPushed;

}

/\*public void onPortalBurronDown()

{

portalButtonPushed = !portalButtonPushed;

}\*/

private void OnTriggerStay2D(Collider2D other)

{

if (other.CompareTag("Door") && keyButtonPushed && keyIcon.activeInHierarchy)

{

keyIcon.SetActive(false);

other.gameObject.SetActive(false);

keyButtonPushed = false;

}

if (other.CompareTag("Portal"))

{

if (int.Parse(floor.text) < 3)

{

floorNum++;

floor.text = floorNum.ToString();

}

else

{

levelNum++;

level.text = levelNum.ToString();

floorNum = 1;

floor.text = floorNum.ToString();

}

if (level.text != "3")

{

GameObject[] rooms = GameObject.FindGameObjectsWithTag("Room");

foreach(var room in rooms)

{

Destroy(room);

}

Instantiate(Room, Camera.main.GetComponent<Camera>().transform.position, Quaternion.identity);

Instantiate(mainRoom, Camera.main.GetComponent<Camera>().transform.position, Quaternion.identity);

gameObject.transform.position = Camera.main.GetComponent<Camera>().transform.position;

}

}

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class RoomSpawner : MonoBehaviour

{

public Direction direction;

public enum Direction

{

Top,

Bottom,

Left,

Right,

None

}

private RoomVariants variants;

private int rand;

private bool spawned = false;

private float waitTime = 3f;

private void Start()

{

variants = GameObject.FindGameObjectWithTag("Rooms").GetComponent<RoomVariants>();

Invoke("Spawn", 0.2f);

Destroy(gameObject, waitTime);

}

public void Spawn()

{

if (!spawned)

{

if(direction == Direction.Top)

{

rand = Random.Range(0, variants.topRooms.Length);

Instantiate(variants.topRooms[rand], transform.position, variants.topRooms[rand].transform.rotation);

} else if (direction == Direction.Bottom)

{

rand = Random.Range(0, variants.bottomRooms.Length);

Instantiate(variants.bottomRooms[rand], transform.position, variants.bottomRooms[rand].transform.rotation);

}else if (direction == Direction.Left)

{

rand = Random.Range(0, variants.leftRooms.Length);

Instantiate(variants.leftRooms[rand], transform.position, variants.leftRooms[rand].transform.rotation);

}else if (direction == Direction.Right)

{

rand = Random.Range(0, variants.rightsRooms.Length);

Instantiate(variants.rightsRooms[rand], transform.position, variants.rightsRooms[rand].transform.rotation);

}

spawned = true;

}

}

private void OnTriggerStay2D(Collider2D other)

{

if(other.CompareTag("RoomPoint") && other.GetComponent<RoomSpawner>().spawned)

{

Destroy(gameObject);

}

}

public AddRoom AddRoom

{

get => default;

set

{

}

}

public RoomVariants RoomVariants

{

get => default;

set

{

}

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class RoomVariants : MonoBehaviour

{

public GameObject[] topRooms;

public GameObject[] bottomRooms;

public GameObject[] leftRooms;

public GameObject[] rightsRooms;

public GameObject key;

private Player player;

[HideInInspector] public List<GameObject> rooms;

private void Start()

{

StartCoroutine(RandomSpawner());

player = GameObject.FindGameObjectWithTag("Player").GetComponent<Player>();

}

IEnumerator RandomSpawner()

{

yield return new WaitForSeconds(5f);

AddRoom lastRoom = rooms[rooms.Count - 1].GetComponent<AddRoom>();

lastRoom.isLastRoom = true;

if (int.Parse(player.floor.text) == 3)

lastRoom.isBossRoom = true;

int rand = Random.Range(0, rooms.Count - 2);

Instantiate(key, rooms[rand].transform.position, Quaternion.identity);

lastRoom.door.SetActive(true);

lastRoom.DestroyWalls();

Destroy(gameObject);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

using UnityEngine.UI;

public class Upgrade : MonoBehaviour

{

public GameObject[] first;

public GameObject[] second;

public GameObject[] third;

public GameObject[] fourth;

public Text points;

private bool[] HP\_Up;

private bool[] Armor\_Up;

private bool[] DMG\_Up;

public Upgrade Upgrade1

{

get => default;

set

{

}

}

// Start is called before the first frame update

void Start()

{

Time.timeScale = 1f;

PlayerPrefs.SetInt("HP\_1", PlayerPrefs.GetInt("maxHealth") == 10 ? 1 : PlayerPrefs.GetInt("HP\_1"));

PlayerPrefs.SetInt("HP\_2", PlayerPrefs.GetInt("maxHealth") <= 13 ? 1 : PlayerPrefs.GetInt("HP\_2"));

PlayerPrefs.SetInt("HP\_3", PlayerPrefs.GetInt("maxHealth") <= 16 ? 1 : PlayerPrefs.GetInt("HP\_3"));

PlayerPrefs.SetInt("HP\_4", PlayerPrefs.GetInt("maxHealth") <= 19 ? 1 : PlayerPrefs.GetInt("HP\_4"));

PlayerPrefs.SetInt("ARM\_1", PlayerPrefs.GetInt("armor") == 0 ? 1 : PlayerPrefs.GetInt("ARM\_1"));

PlayerPrefs.SetInt("ARM\_2", PlayerPrefs.GetInt("armor") <= 1 ? 1 : PlayerPrefs.GetInt("ARM\_2"));

PlayerPrefs.SetInt("ARM\_3", PlayerPrefs.GetInt("armor") <= 2 ? 1 : PlayerPrefs.GetInt("ARM\_3"));

PlayerPrefs.SetInt("ARM\_4", PlayerPrefs.GetInt("armor") <= 3 ? 1 : PlayerPrefs.GetInt("ARM\_4"));

PlayerPrefs.SetInt("DMG\_1", PlayerPrefs.GetInt("damage") == 1 ? 1 : PlayerPrefs.GetInt("DMG\_1"));

PlayerPrefs.SetInt("DMG\_2", PlayerPrefs.GetInt("damage") <= 2 ? 1 : PlayerPrefs.GetInt("DMG\_2"));

PlayerPrefs.SetInt("DMG\_3", PlayerPrefs.GetInt("damage") <= 3 ? 1 : PlayerPrefs.GetInt("DMG\_3"));

PlayerPrefs.SetInt("DMG\_4", PlayerPrefs.GetInt("damage") <= 4 ? 1 : PlayerPrefs.GetInt("DMG\_4"));

HP\_Up = new bool[]

{

PlayerPrefs.GetInt("HP\_1") == 0 ? false : (PlayerPrefs.GetInt("HP\_1") == 1 ? true : false),

PlayerPrefs.GetInt("HP\_2") == 0 ? false : (PlayerPrefs.GetInt("HP\_2") == 1 ? true : false),

PlayerPrefs.GetInt("HP\_3") == 0 ? false : (PlayerPrefs.GetInt("HP\_3") == 1 ? true : false),

PlayerPrefs.GetInt("HP\_4") == 0 ? false : (PlayerPrefs.GetInt("HP\_4") == 1 ? true : false)

};

Armor\_Up = new bool[]

{

PlayerPrefs.GetInt("ARM\_1") == 0 ? false : (PlayerPrefs.GetInt("ARM\_1") == 1 ? true : false),

PlayerPrefs.GetInt("ARM\_2") == 0 ? false : (PlayerPrefs.GetInt("ARM\_2") == 1 ? true : false),

PlayerPrefs.GetInt("ARM\_3") == 0 ? false : (PlayerPrefs.GetInt("ARM\_3") == 1 ? true : false),

PlayerPrefs.GetInt("ARM\_4") == 0 ? false : (PlayerPrefs.GetInt("ARM\_4") == 1 ? true : false)

};

DMG\_Up = new bool[]

{

PlayerPrefs.GetInt("DMG\_1") == 0 ? false : (PlayerPrefs.GetInt("DMG\_1") == 1 ? true : false),

PlayerPrefs.GetInt("DMG\_2") == 0 ? false : (PlayerPrefs.GetInt("DMG\_2") == 1 ? true : false),

PlayerPrefs.GetInt("DMG\_3") == 0 ? false : (PlayerPrefs.GetInt("DMG\_3") == 1 ? true : false),

PlayerPrefs.GetInt("DMG\_4") == 0 ? false : (PlayerPrefs.GetInt("DMG\_4") == 1 ? true : false)

};

points.text = "Очки: " + PlayerPrefs.GetInt("points");

first[0].SetActive(HP\_Up[0]);

first[1].SetActive(Armor\_Up[0]);

first[2].SetActive(DMG\_Up[0]);

second[0].SetActive(HP\_Up[1]);

second[1].SetActive(Armor\_Up[1]);

second[2].SetActive(DMG\_Up[1]);

third[0].SetActive(HP\_Up[2]);

third[1].SetActive(Armor\_Up[2]);

third[2].SetActive(DMG\_Up[2]);

fourth[0].SetActive(HP\_Up[3]);

fourth[1].SetActive(Armor\_Up[3]);

fourth[2].SetActive(DMG\_Up[3]);

if (PlayerPrefs.GetInt("points") >= 100)

for (int i = 0; i < first.Length; i++)

{

first[i].SetActive(true);

}

else

for (int i = 0; i < first.Length; i++)

{

first[i].SetActive(false);

}

if (PlayerPrefs.GetInt("points") >= 200)

for (int i = 0; i < second.Length; i++)

{

second[i].SetActive(true);

}

else

for (int i = 0; i < second.Length; i++)

{

second[i].SetActive(false);

}

if (PlayerPrefs.GetInt("points") >= 300)

for (int i = 0; i < third.Length; i++)

{

third[i].SetActive(true);

}

else

for (int i = 0; i < third.Length; i++)

{

third[i].SetActive(false);

}

if (PlayerPrefs.GetInt("points") >= 350)

for (int i = 0; i < fourth.Length; i++)

{

fourth[i].SetActive(true);

}

else

for (int i = 0; i < fourth.Length; i++)

{

fourth[i].SetActive(false);

}

}

private void Update()

{

points.text = "Очки: " + PlayerPrefs.GetInt("points");

if (PlayerPrefs.GetInt("points") >= 100)

{

for (int i = 0; i < first.Length; i++)

{

first[i].SetActive(true);

}

first[0].SetActive(HP\_Up[0]);

first[1].SetActive(Armor\_Up[0]);

first[2].SetActive(DMG\_Up[0]);

}

else

for (int i = 0; i < first.Length; i++)

{

first[i].SetActive(false);

}

if (PlayerPrefs.GetInt("points") >= 200)

{

for (int i = 0; i < second.Length; i++)

{

second[i].SetActive(true);

}

second[0].SetActive(HP\_Up[1]);

second[1].SetActive(Armor\_Up[1]);

second[2].SetActive(DMG\_Up[1]);

}

else

for (int i = 0; i < second.Length; i++)

{

second[i].SetActive(false);

}

if (PlayerPrefs.GetInt("points") >= 300)

{

for (int i = 0; i < third.Length; i++)

{

third[i].SetActive(true);

}

third[0].SetActive(HP\_Up[2]);

third[1].SetActive(Armor\_Up[2]);

third[2].SetActive(DMG\_Up[2]);

}

else

for (int i = 0; i < third.Length; i++)

{

third[i].SetActive(false);

}

if (PlayerPrefs.GetInt("points") >= 350)

{

for (int i = 0; i < fourth.Length; i++)

{

fourth[i].SetActive(true);

}

fourth[0].SetActive(HP\_Up[3]);

fourth[1].SetActive(Armor\_Up[3]);

fourth[2].SetActive(DMG\_Up[3]);

}

else

for (int i = 0; i < fourth.Length; i++)

{

fourth[i].SetActive(false);

}

}

public void addHP\_1()

{

PlayerPrefs.SetInt("HP\_1", 0);

PlayerPrefs.SetInt("maxHealth", PlayerPrefs.GetInt("maxHealth") + 3);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 100);

/\*first[0].SetActive(false);\*/

}

public void addHP\_2()

{

PlayerPrefs.SetInt("HP\_2", 0);

PlayerPrefs.SetInt("maxHealth", PlayerPrefs.GetInt("maxHealth") + 3);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 200);

}

public void addHP\_3()

{

PlayerPrefs.SetInt("HP\_3", 0);

PlayerPrefs.SetInt("maxHealth", PlayerPrefs.GetInt("maxHealth") + 3);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 300);

}

public void addHP\_4()

{

PlayerPrefs.SetInt("HP\_4", 0);

PlayerPrefs.SetInt("maxHealth", PlayerPrefs.GetInt("maxHealth") + 3);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 350);

}

public void addARM\_1()

{

PlayerPrefs.SetInt("ARM\_1", 0);

PlayerPrefs.SetInt("armor", PlayerPrefs.GetInt("armor") + 1);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 100);

}

public void addARM\_2()

{

PlayerPrefs.SetInt("ARM\_2", 0);

PlayerPrefs.SetInt("armor", PlayerPrefs.GetInt("armor") + 1);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 200);

}

public void addARM\_3()

{

PlayerPrefs.SetInt("ARM\_3", 0);

PlayerPrefs.SetInt("armor", PlayerPrefs.GetInt("armor") + 1);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 300);

}

public void addARM\_4()

{

PlayerPrefs.SetInt("ARM\_4", 0);

PlayerPrefs.SetInt("armor", PlayerPrefs.GetInt("armor") + 1);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 350);

}

public void addDMG\_1()

{

PlayerPrefs.SetInt("DMG\_1", 0);

PlayerPrefs.SetInt("damage", PlayerPrefs.GetInt("damage") + 1);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 100);

}

public void addDMG\_2()

{

PlayerPrefs.SetInt("DMG\_2", 0);

PlayerPrefs.SetInt("damage", PlayerPrefs.GetInt("damage") + 1);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 200);

}

public void addDMG\_3()

{

PlayerPrefs.SetInt("DMG\_3", 0);

PlayerPrefs.SetInt("damage", PlayerPrefs.GetInt("damage") + 1);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 300);

}

public void addDMG\_4()

{

PlayerPrefs.SetInt("DMG\_4", 0);

PlayerPrefs.SetInt("damage", PlayerPrefs.GetInt("damage") + 1);

PlayerPrefs.SetInt("points", PlayerPrefs.GetInt("points") - 350);

}

public void NextBtn()

{

SceneManager.LoadScene(0);

}

}

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Wall : MonoBehaviour

{

public GameObject block;

public Wall Wall1

{

get => default;

set

{

}

}

private void OnTriggerEnter2D(Collider2D other)

{

if (other.CompareTag("Block"))

{

Debug.Log("Block behind the Wall!");

Instantiate(block, transform.GetChild(0).position, Quaternion.identity);

Instantiate(block, transform.GetChild(1).position, Quaternion.identity);

Destroy(gameObject);

}

}

private void OnTriggerStay2D(Collider2D other)

{

if (other.CompareTag("Block"))

{

Debug.Log("Block behind the Wall!");

Instantiate(block, transform.GetChild(0).position, Quaternion.identity);

Instantiate(block, transform.GetChild(1).position, Quaternion.identity);

Destroy(gameObject);

}

}

}