

# The Adventure of the Nameless Knight

LUCRARE PENTRU OBŢINEREA ATESTATULUI PROFESIONAL ÎN INFORMATICĂ

Elevi: Opariuc Rareș-Ioan și Bărănceanu Vlad-Andrei

Clasa: a XII-a A

Profesor coordonator: Emanuela Cerchez

2020

1. **Introducere**

Industria jocurilor video este în plină dezvaltare. Astfel, apar noi categorii în care acestea pot fi împărțite. Noi am ales să creăm un joc în care se încadrează în categoria de ***platformer***. Personajul sare pe nişte platforme cu scopul de a ajunge la capătul nivelului. Acest proiect se adresează tuturor pasionaţilor de jocuri video, dar şi oamenilor care nu s-au jucat niciodată pe calculator, fiind o metodă de amuzament pentru toate vârstele. Aplicaţia creată prezintă povestea unui cavaler care are ca scop salvarea prinţesei din mâinele unui bandit.

1. Prezentare

Odată ce deschideţi jocul, veţi fi întâmpinaţi de meniul principal. Acesta este format din trei butoane:

* + - **Play:** buton care duce la ecranul de selectare a nivelurilor;
    - **Options:** buton care duce la meniul de selectare a opţiunilor;
    - **Quit**: buton prin care părăseşti aplicaţia.

****

Meniul de selectarea a nivelelor constă în 3 butoane care permit alegerea primului nivel, celui de-al doilea nivel sau întoarcearea la ecranul principal.

Primul nivel are ca scop parcurgerea unei păduri prin intermediul unor platforme și prin evitarea de capcane pentru a ajunge la un bănuţ.

Al doilea nivel constă intr-o serie de camere dintr-un castel pe care eroul trebuie să le parcurgă pentru a ajunge la antagonistul jocului. El trebuie să îl înfrângă pentru a salva prinţesa. Totodată este prezent și un sistem de vieţi care te face să reiei nivelul dacă rămâi fară ele.

**II. Detalii tehnice**

1. Descriere generală

Această aplicație a fost realizată ȋn Unity, utilizând limbajul de programare C#.

1. Personajul principal (Cavalerul fara nume)

Pentru player, am folosit 4 clase:

* public class Teleport : MonoBehaviour (folosit în primul nivel);
* public class Death\_Player : MonoBehaviour (folosit în al doilea nivel);
* public class PlayerController : MonoBehaviour;
* public class Player\_Attack : MonoBehaviour.

2.1. Clasa PlayerController

public class PlayerController : MonoBehaviour

{

private Rigidbody2D rb;

public float speed; //viteza

private float moveInput;

private bool isGrounded; //verificam daca player-ul sare si este in aer sau este pe pamant

public Transform feetPos; //pozitia picioarelor jucatorului

public float checkRadius; //diametrul obiectului

public LayerMask whatisGround; //daca player-ul va calca pe suprafete cu anumite referinte (Enemy, Ground, etc.)

public float jumpForce; //cat de sus sare

private bool isJumping; //daca sare sau nu

private Animator anim; //o variabila prin care putem controla animatiile

//folosim doua variabile: o variabila de tip private care este folosit in cod si cealalta variabila de tip public pe care o putem modifica in aplicatia Unity

private float jumpTimeCounter;

public float jumpTime;

void Start()

{

//initializarea variabilelor

anim = GetComponent<Animator>();

rb = GetComponent<Rigidbody2D>(); //folosim o variabila de tip Rigidbody2D pentru a putea controla pozitia obiectului prin simulari fizice (gravitatia, masa, etc.)

}

void FixedUpdate()

{

//functia este folosita pentru a face calcule fizice

moveInput = Input.GetAxisRaw("Horizontal"); //miscarea playerul-ului

rb.velocity = new Vector2(moveInput \* speed, rb.velocity.y); //cand se misca player-ul, pe axa x se mareste in timp ce axa y ramane constanta

}

void Update() //la fiecare frame (imagine), aceasta functie este apelata

{

if (moveInput == 0)

anim.SetBool("isRunning", false);

else

anim.SetBool("isRunning", true);

//cand se misca stanga sau dreapta, imaginea se misca in acelasi timp

if (moveInput < 0)

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

else if (moveInput > 0)

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

isGrounded = Physics2D.OverlapCircle(feetPos.position, checkRadius, whatisGround); //folosim un obiect auxiliar pentru a verifica daca este pe pamant sau nu

if (isGrounded == true && Input.GetKeyDown("space")) //daca apasa pe space doar

{

anim.SetTrigger("takeof");

jumpTimeCounter = jumpTime;

rb.velocity = Vector2.up \* jumpForce;

isJumping = true;

}

if (isGrounded == true)

anim.SetBool("isJump", false);

else

anim.SetBool("isJump", true);

if (Input.GetKey("space") && isJumping == true) //daca tine mai mult timp apasat tasta space, sare mai sus

{

if (jumpTimeCounter > 0)

{

rb.velocity = Vector2.up \* jumpForce;

jumpTimeCounter -= Time.deltaTime;

}

else

isJumping = false;

}

if (Input.GetKeyUp(KeyCode.Space))

isJumping = false;

}

//cand ia banutul, trece la nivelul urmator

private void OnTriggerEnter2D(Collider2D collision)

{

if (collision.gameObject.tag.Equals("Coin"))

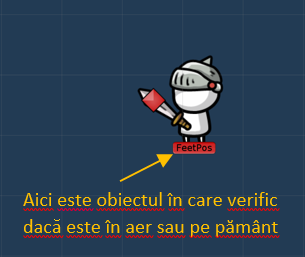
SceneManager.LoadScene("Levels\_menu");

}

}

Clasa are rolul de a mișca player-ul stânga-dreapta și de a sări. Jucătorul se mișcă cu ajutorul tastelor WASD și sare cu ajutorul tastei Spacebar.

Am folosit niște variabile pentru a determina viteza de mișcare (***moveInput***, ***speed***). Ca să știu dacă player-ul este în aer sau este pe pământ, am folosit un obiect auxiliar ***feetPos*** legat la baza picioarelor personajului cu variabilele ***isGrounded*** și ***isJumping***. Variabilele ***jumpForce, JumpTimecounter, JumpTime*** arată cât de sus sare și cât timp este în aer.



2.2 Clasa Player\_Attack

public class Player\_Attack : MonoBehaviour

{

private float timeBtwAttack;

public float startTimeBtwAttack;

float nextAttackTime = 0f;

public float attackRate = 2f;

public Transform attackPos;

public float attackRangeX;

public float attackRangeY;

public LayerMask whatIsEnemy;

public int damage;

public Animator playerAnim;

void Update()

{

if (Time.time >= nextAttackTime)

{

//ataci

if (Input.GetMouseButtonDown(0))

{

playerAnim.SetTrigger("attack");

Collider2D[] enemiesToDamage = Physics2D.OverlapBoxAll(attackPos.position, new Vector2(attackRangeX, attackRangeY), 0, whatIsEnemy); //creeaza un cerc invizibil pt a lovi inamicii

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

enemiesToDamage[i].GetComponent<Enemy>().TakeDamage(damage);

nextAttackTime = Time.time + 1f / attackRate;

}

}

}

void OnDrawGizmosSelected()

{

Gizmos.color = Color.red;

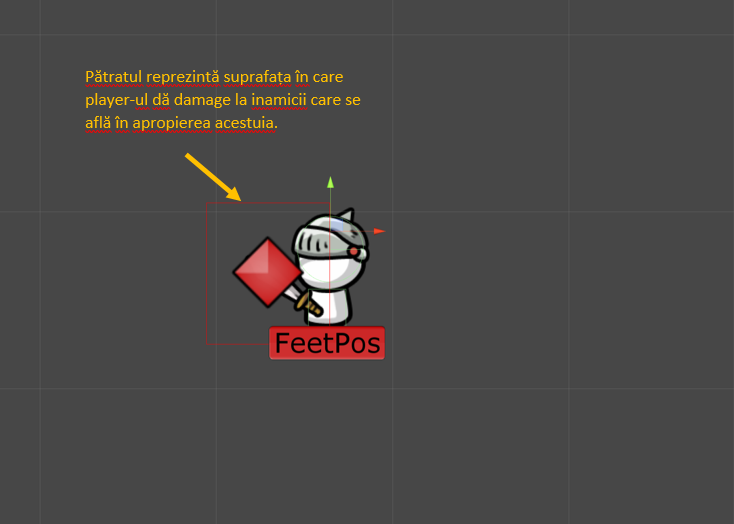
Gizmos.DrawWireCube(attackPos.position, new Vector3(attackRangeX, attackRangeY, 1));

}

}

Clasa ajută player-ul de a ataca inamicii și de a folosi un atac la un anumit timp. Pentru a ataca, apeși pe M1 (click stânga pe mouse).

Variabilele ***timeBtwAttack***, ***startTimeBtwAttack***, ***nextAttackTime*** inițializează durata unui atac și când va fi următorul atac, iar variabila ***attackRate*** ajustează numărul de atacuri în acea perioadă de timp.

La fiecare atac al player-ului, apare un obiect în fața lui cu lungimea ***attackRangeX*** și lățimea ***attackRangeY*** în care, dacă un inamic se lovește de acesta, el va pierde din viață până când va fi eliminat.

2.3 Clasa Death\_Player

public class Death\_Player : MonoBehaviour

{

//folosit la nivelul 2

public static float health = 3;

public Transform respawnPoint;

public GameObject bloodEffect;

public Transform teleportTarget\_1;

public Transform teleportTarget\_2;

public Transform teleportTarget\_3;

public Transform teleportTarget\_4;

public Camera Cam;

public Transform teleportCam\_1;

public Transform teleportCam\_2;

public Transform teleportCam\_3;

public Transform teleportCam\_4;

public GameObject Player;

//cand trece la urmatoarea camera, se schimba pozitia player-ului si pozitia camerei prin care vede jucatorul

private void OnCollisionEnter2D(Collision2D col)

{

if (col.gameObject.tag.Equals("nxt\_lvl\_1\_2"))

{

Player.transform.position = teleportTarget\_2.transform.position;

Cam.transform.position = teleportCam\_2.transform.position;

}

if (col.gameObject.tag.Equals("nxt\_lvl\_2\_3"))

{

Player.transform.position = teleportTarget\_3.transform.position;

Cam.transform.position = teleportCam\_3.transform.position;

}

if (col.gameObject.tag.Equals("nxt\_lvl\_3\_4"))

{

Player.transform.position = teleportTarget\_4.transform.position;

Cam.transform.position = teleportCam\_4.transform.position;

}

if (col.gameObject.tag.Equals("Obstacle"))

{

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

Level\_GM.keys = 0;

}

}

public void TakeDamage\_player(int damage)

{

health -= damage;

if (health <= 0)

{

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

health = 3;

Level\_GM.keys = 0;

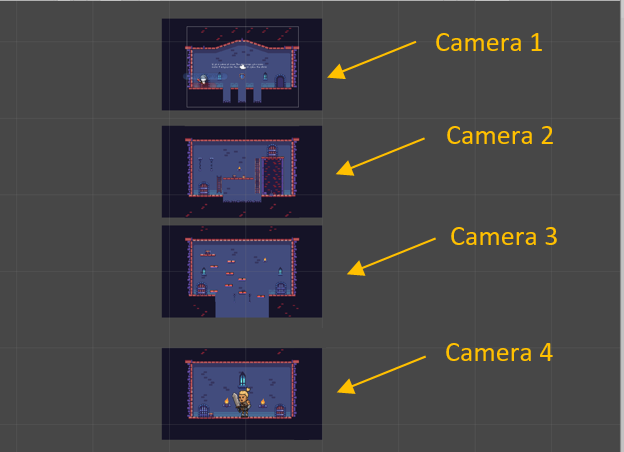
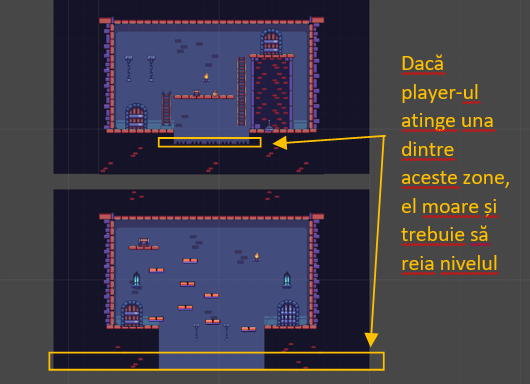
}

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

}

}

Această clasă este folosită în nivelul 2 și ajută la avansarea acestuia prin camere si la moartea acestuia. Când ajunge la o ușă deschisă, el trece la următoarea cameră până la Boss-ul final. Daca HP (viața) ajunge la 0 sau atinge țepii, acesta moare si reia nivelul (prima imagine).

Toate aceste camere sunt puse în aceeași scenă și când trece la următoarea cameră, el este teleportat in următoarea cameră (a doua imagine).

2.4. Clasa Teleport

public class Teleport : MonoBehaviour

{

//folosit in primul nivel, cand player-ul moare, isi da respawn la locul de inceput

public GameObject Portal;

public GameObject Player;

public void OnTriggerEnter2D(Collider2D other)

{

if(other.gameObject.tag=="Player")

{

StartCoroutine(Teleportare());

}

}

IEnumerator Teleportare()

{

yield return new WaitForSeconds(0);

Player.transform.position = new Vector2(Portal.transform.position.x, Portal.transform.position.y);

}

}

Această clasă este folosită în nivelul 1 și teleportează player-ul la poziția de început dacă player-ul cade sau atinge țepii.

1. Boss-ul final și clasa Enemy

Pentru Boss-ul final, am folosit clasa Enemy. Când reușești să elimini Boss-ul, primești o cheie care te va ajuta să salvezi prințesa.

public class Enemy : MonoBehaviour

{

public int health;

public GameObject bloodEffect;

public Animator anim;

public float attackRange;

public int damage;

private float lastAttackTime;

public float attackDelay;

public Transform target;

public GameObject chest;

private void Start()

{

chest.gameObject.GetComponent<BoxCollider2D>().enabled = false;

chest.gameObject.GetComponent<SpriteRenderer>().enabled = false;

}

void Update()

{

float distanceToPlayer = Vector3.Distance(transform.position, target.position);

if (distanceToPlayer < attackRange)

{

if (Time.time > lastAttackTime + attackDelay)

{

anim.SetTrigger("Attack");

target.SendMessage("TakeDamage\_player", damage);

lastAttackTime = Time.time;

}

}

if (health <= 0)

{

anim.SetBool("isDead", true);

this.enabled = false;

this.gameObject.GetComponent<BoxCollider2D>().enabled = false;

chest.gameObject.GetComponent<BoxCollider2D>().enabled = true;

chest.gameObject.GetComponent<SpriteRenderer>().enabled = true;

}

}

public void TakeDamage(int damage)

{

anim.SetTrigger("Hit");

health -= damage;

Debug.Log("damage taken");

}

}

Boss-ul funcționează astfel: când player-ul se apropie destul de mult de el, dă un atac și se pregătește pentru următorul. Variabila ***attackRange*** arată distanța minimă sigură dintre player si Boss, iar variabila ***target*** ghidează Boss-ul pe cine să atace (player-ul). ***lastAttackTime*** și ***attackDelay*** inițializează primul atac și îl pregătește pe următorul.

1. Nivelul2

În scena **Nivelul2**,fiecare cameră are ceva special. Jucătorul trebuie să gasească cheile ce sunt în cufere pentru a salva prințesa.

public class chestInteraction1 : MonoBehaviour

Fiecare cufăr are această clasă. Când player-ul se apropie de cufăr, trebuie să apese pe tasta E ca să deschidă cufărul și să ia cheia.

public class Ladder : MonoBehaviour

În camera a doua, player-ul trebuie să urce (tasta W) și să coboare (tasta S) pe o scară. El când apasă pe una dintre aceste taste, axa Y este modificată și astfel avansează în nivel.

public class LeverInteraction1 : MonoBehaviour

Tot în a doua camera, player-ul trebuie să apese o manetă cu ajutorul tastei E și să deschidă ușa pentru a continua jocul.

public class Platform\_Change : MonoBehaviour

public class Platform\_interaction : MonoBehaviour

În al treilea nivel, player-ul trebuie să sare pe platforme pentru a recupera cheia. Când atinge fiecare platformă, ea se distruge după un anumit timp. Prima clasă anunță player-ul că platforma va fi distrusă, iar a două clasă distruge platforma.

public class Ending : MonoBehaviour

Când ajungi la finalul nivelului 2 și ai cele 3 chei, primesti finalul. Dacă nu ai destule chei, atunci jocul te anunță că trebuie să dai restart și să încerci să le obții a doua oară pe toate. Acest lucru este gestionat de clasa de mai sus.

1. UI (User interface)

În **Nivelul2,** am introdus un sistem de vieți și un sistem de chei ca să anunțe player-ul câte vieți și chei are.

public class Level\_GM : MonoBehaviour

public class number\_keys : MonoBehaviour

Prima clasă inițializează numărul de chei cu 0, iar a doua clasă updatează numărul de chei după ce le colectează jucătorul.

public class Number\_lives : MonoBehaviour

Această clasă inițializează numărul de vieți și anunță jucătorul câte vieți mai are. Dacă ajunge la 0, player-ul va muri și nivelul își va da restart.

* 1. Main Menu și Pause Menu

Când jucătorul intră în menu, trebuie să apese pe butonul de **PLAY** ca să ajungă în meniul de nivele, să apese pe butonul de **OPTIONS** ca să modifice volumul jocului sau să apese pe butonul de **QUIT** ca să iesă din joc. Acest lucru este gestionat de clasa:

public class MainMenu : MonoBehaviour

Când jucătorul vrea să se oprească din jucat pentru a face o mică pauză, apasă pe tasta Escape. Aici, poate ieși din nivel și să ajungă la meniul principal (butonul **QUIT**), să dea restart la nivel (butonul **RESTART**) sau să se întoarcă la nivel (butonul **RESUME**). Acest menu de pauză este gestionat de clasa:

public class Pause\_Menu : MonoBehaviour

1. Muzica

Jucătorul trece prin nivele și aude muzică. La fiecare nivel și la meniul principal, am creat un obiect în care am atașat melodia și clasele:

public class VolumeMenu : MonoBehaviour

public class VolumeChangeValue : MonoBehaviour

La meniul principal și meniul de pauză, am pus o bara de volum și am atașat clasa aceasta. Astfel, poți modifica volumul muzicii, mișcând cercul de pe bara de volum.

1. Alte clase

În primul nivel, camera urmărește player-ul și se oprește când ajunge la marginea nivelului. Acest lucru este controlat de clasa:

public class camerafollow : MonoBehaviour

După meniul principal, este meniul de nivele de unde poți accesa Nivelul1/2 sau să te întorci la meniul principal. Acest lucru este administrat de clasa:

public class Levels : MonoBehaviour

III. Cerințe tehnice

Pentru a rula aplicația, este nevoie de un calculator cu minim Windows 7.

1. Bibliografie

<https://docs.unity3d.com/Manual/index.html>

<https://stackoverflow.com/>

**Cuprins**

1. Introducere…………………………………………………………………….2
2. Prezentare………………………………………………………………...2
3. Detalii tehnice………………………………………………………………....4
4. Descriere generala………………………………………………………...4
5. Personajul principal………………………………………………………..4
   1. Clasa PlayerController……………………………………………..5
   2. Clasa Player\_Attack………………………………………………..7
   3. Clasa Death\_Player………………………………………………...8
   4. Clasa Teleport……………………………………………………..10
6. Boss-ul final și clasa Enemy…………………………………………....11
7. Nivelul2…………………………………………………………………...12
8. UI (User interface)……………………………………………………….13
   1. Main Menu și Pause Menu………………………………………..13
9. Muzica…………………………………………………………………....14
10. Alte clase…………………………………………………………………14
11. Cerințe tehnice…………………………………………………………........15
12. Bibliografie…………………………………………………………………..15