**UNIVERSITATEA „BABEŞ-BOLYAI”**

**FACULTATEA DE MATEMATICǍ ŞI INFORMATICǍ**

**SPECIALIZAREA MATEMATICA-INFORMATICǍ**

**Lucrare de Licența**

**HOW TO TRAIN YOUR CREATURE**

Coordonator științific Student

Lect. Dr. **Suciu Dan Mircea Handaric Rares**

**Cluj-Napoca, Romania**

**2021**

**Abstract**

Every child loves to play, even adults sometimes. In our century most people are increasingly drawn to video games.

In the following chapters I will present an adventure game where the main character captures creatures that are found in his world and fights alongside them in exciting battles against other creatures or even other tamers.

**Cuprins**

Capitolul 1 - Introducere

1.1 Motivul alegerii acestei teme

1.2 Scurta prezentare a temei alese

Capitolul 2 – Platforma de proiectare jocuri video unity

2.1 Unity game engine

2.2 C#

Capitolul 3 – Șabloane de proiectare

Capitolul 4 – Proiectarea si dezvoltarea unui joc video particular

Capitolul 5 – Utilizarea jocului

Capitolul 6 - Concluzii

**Capitolul 1 – Introducere**

Acest capitol prezinta ideile principale ale unui joc si avantajele unui joc de aventura precum si o scurta descriere a temei alese

**1.1 Motivul alegerii acestei teme**

Jocurile reprezintă o formă structurată de joacă, de obicei pentru divertisment sau distracție, iar uneori este folosit ca instrument educațional.

Componentele cheie ale jocurilor sunt obiectivele, regulile, provocările și interacțiunea. Jocurile implică, în general, stimularea mentală sau fizică și, adesea, ambele. Multe jocuri contribuie la dezvoltarea unor abilități practice, servesc ca formă de exercițiu sau îndeplinesc un rol educativ, de simulare sau psihologic.

Oricărui copil ii place sa se joace, chiar si adulților câteodată. În secolul nostru, tot mai mulți copii au acces la calculator sau telefon. Marea majoritate, încep încă de mici sa se joace pe diferite dispozitive (calculator, telefon, ps, xbox, nintendo, etc.).

Chiar dacă mulți oameni nu ar fi de acord, jocurile video sunt de fapt foarte benefice, mai ales dacă alegem jocul potrivit. Dezvoltarea funcțiilor creierului este principalul beneficiu înainte de toate. Bineînțeles, excluzând cazurile în care jucătorii petrec fiecare secundă a zilei cu ochii în calculator. Există câteva beneficii generale cum ar fi îmbunătățirea abilităților de *multitasking*, învingerea dislexiei, ameliorarea stresului si stimularea concentrării.

Numeroase studii au ajuns la concluzia că, o persoană este mai probabil să își îmbunătățească abilitățile de *multitasking* daca se joaca mai multe jocuri. Menținerea țintei pe ecran, răspunsul la lumini, ascultarea comunicațiilor radio și monitorizarea mai multor zone în același timp sunt lucruri necesare, iar acestea duc la o performanță ridicată a abilităților cerebrale. Citirea hărților este conținutul obișnuit al jocurilor de mister și de aventură, care aduce multe beneficii și în viața reală.

Deși dislexia nu afectează inteligența generală a unei persoane, ea afectează prin reducerea capacității creierului de a citi și de a învăța simboluri și, în majoritatea cazurilor, este rezultatul unor probleme de atenție vizuală. Atunci când jucam jocuri de aventură sau de acțiune, ne face să ne concentrăm atenția mai eficient pentru a extrage mai rapid informațiile relevante dintr-un cuvânt scris. Sarcinile fundamentale implică nevoia de a distribui atenția către diferite simțuri, iar prin jocuri, aceasta este îmbunătățită în cadrul multor persoane cu dislexie. Astfel, aceștia învață să își schimbe atenția cu mai multă ușurință. Se considera că mai multe jocuri video antrenează părțile creierului responsabile de atenție și de percepție a mișcării.

Totodată, jocurile ajută la îmbunătățirea imaginației. Multe jocuri oferă multe combinații de mișcări sau puzzle-uri care trebuie rezolvate pentru a termina jocul. Rezolvând aceste enigme copiii învață să gândească o problemă în mai multe moduri și astfel li se dezvoltă imaginația și gândirea.

Jocurile pot reduce, de asemenea, nivelul de depresie și de stres și nu numai. Acestea sunt deseori folosite astăzi ca o formă de terapie pentru persoanele care suferă de o gamă largă de probleme de sănătate mintală. S-a descoperit că jocurile video reduc semnificativ nivelul hormonului de stres, cortizol.

Adesea, zonele pe care le putem găsi în jocurile de aventură sunt cele care necesită să luam decizii rapide. Atunci când acest lucru este constant, devine o funcție naturală a creierului nostru. Pentru a rezolva unele probleme și pentru a finaliza nivelurile trebuie să gândim cu câțiva pași înainte și să folosim toată logica posibilă. Aceasta s-a dovedit a fi de mare ajutor pentru copii la vârsta adultă, folosindu-și logica și exactitatea cu abilitatea de a gândi în afara cutiei.

**1.2 Scurta prezentare a temei alese**

În capitolele ce urmează voi prezenta un joc de aventură, în care personajul principal este controlat de jucător. Acesta explorează o lume nouă în care se găsesc creaturi nemaiîntâlnite în lumea reală. Caracterul poate captura aceste creaturi cu ajutorul unor obiecte speciale, iar apoi se poate lupta alături de o echipa de 6 creaturi prinse de acesta in bătălii incitante contra altor creaturi sau chiar contra altor îmblânzitori de creaturi.

Fiecare creatura si atac al său are câte un element. Atacurile pot fi mai eficiente sau mai puțin eficiente in funcție de elementul creaturii care este atacată.

În lupte, doar cate 2 creaturi se pot lupta deodată. Personajul selectează ce atacuri sa folosească creatura sa împotriva celeilalte creaturi ca adversarul sau sa nu mai poată lupta.

Bătălia se va termina când oricare dintre îmblânzitori nu mai are nici o creatura care poate lupta sau in cazul luptei contra unei creaturi sălbatice, lupta se va termina când acea creatură a fost capturată sau când aceasta nu mai poate lupta.

**Capitolul 2 – Platforma de proiectare jocuri video unity**

Capitolul 2 prezinta informații teoretice despre platforma de proiectare jocuri video unity, punând in evidență si o mare parte din avantajele acestei platforme.

**2.1 Unity game engine**

Unity este o multiplatformă pentru jocuri video dezvoltat de Unity Technologies. Acesta a fost anunțat și lansat pentru prima dată în iunie 2005 în cadrul conferinței Worldwide Developers Conference a Apple Inc. ca fiind un program de jocuri exclusiv pentru Mac OS X. De atunci, programul a fost extins treptat pentru a suporta o varietate de platforme (desktop, mobile, console și realitate virtuală). Este in mod special popular pentru dezvoltarea de jocuri pentru dispozitive mobile iOS și Android. Este utilizat pentru jocuri precum Temple Run, Pokemon Go, Subway Surfers, Call of Duty: Mobile, Hearthstone, Angry Birds si multe altele. Este considerat ca fiind ușor de utilizat pentru dezvoltatorii începători și este popular pentru dezvoltarea de jocuri video individuale(Indie Games).

*Engine*-ul poate fi folosit pentru a crea jocuri bidimensionale (2D) și tridimensionale (3D), precum și simulări interactive și alte tipuri de experiențe. *Engine*-ul a fost preluat si de industrii din afara jocurilor video, precum cele din industriile filmul, auto, arhitectură, inginerie și construcții.

**Multiplatformă**

Un prim motiv pentru care Unity a devenit atât de popular este capacitatea sa de a fi implementat pe o mare varietate de platforme, folosind același cod și aceleași *asset*-uri. Este nevoie de două clicuri pentru ca jocul să ruleze pe o altă platformă compatibilă. Unul pentru a schimba platformele, iar celălalt pentru a construi și rula.

**Editorul unity**

Editorul Unity este compus din mai multe subferestre. Cele mai frecvent utilizate sunt*: Browser-ul* de proiect, Inspector, Vizualizarea jocului, Vizualizarea scenei și Ierarhia.

Browser-ul de proiect:

O imagine care conține text, monitor, interior, captură de ecran

Descriere generată automat

Figura 2.1 - Captura cu browser-ul de proiect al editorului unity

*Browser*-ul de proiect este fereastra care conține toate *asset*-urile care au fost importate în Unity și care sunt disponibile pentru utilizare. Aspectul este aproape identic cu cel al Finder-ului din Mac OS X și Explorer în Windows. Acest lucru îi face pe programatori să se simtă mai familiarizați și confortabil, oferindu-le o interfață pe care o folosesc deja zilnic.

Inspectorul:

O imagine care conține text, monitor, interior, captură de ecran

Descriere generată automat

Figura 2.2 - Captura cu browser-ul de proiect al editorului unity

Fereastra Inspector este locul în care sunt vizualizate si modificate detaliile fiecărui obiect de joc. Aici programatorul poate modifica valorile pentru a obține cea mai buna senzație pentru jocului. Inspectorul afișează toate componentele care sunt atașate unui obiect (Script-uri*, Physics, Colliders, Sound*). Acesta este, de asemenea, locul unde pot fi alocate sau modificate variabilele expuse din scripturi.

Vizualizarea jocului:

O imagine care conține text, monitor, interior, electronice

Descriere generată automat

Figura 2.3 - Captura cu browser-ul de proiect al editorului unity

Vizualizarea jocului oferă utilizatorului un *preview* a modului în care va arăta jocul atunci când îl va crea. Aceasta include date complete de intrare și permite programatorului să își testeze modificările fără a fi nevoit să aștepte ca proiectul său să fie să fie compilat și implementat pe platforma vizată.

Vizualizarea Scenei:

O imagine care conține text, interior, monitor

Descriere generată automat

Figura 2.4 - Captura cu browser-ul de proiect al editorului unity

Vizualizarea scenei este locul în care este construit jocul. Programatorul poate trage și plasa asset-uri din vizualizarea proiectului. Controalele familiare ale mânerelor 3D și ale grilelor permit utilizatorului să își plaseze obiectele în poziția perfectă, până la pixel.

Ierarhia:

O imagine care conține text, interior

Descriere generată automat

Figura 2.5 - Captura cu browser-ul de proiect al editorului unity

Fereastra Ierarhie conține o listă a tuturor obiectelor care se află în scena curentă. Această listă este actualizată automat atunci când un obiect este adăugat în scenă. În această fereastră, prin tragerea unui obiect deasupra altui obiect, programatorul poate atribui părinți sau copii.

**Limbaje de Script**

Unity oferă dezvoltatorilor săi trei moduri de a-și programa aplicația încorporată în IDE: UnityScript, C# și Boo. Fiecare dintre aceste scripturi limbaje ar trebui să permită dezvoltatorilor să implementeze același conținut indiferent de preferințele lor în materie de limbaj. În realitate, acest lucru se dovedește a nu fi 100% precis. Aceste limbaje rulează pe Mono și au timpi de compilare rapizi. Mono este o versiune *open-source* a programului .NET al Microsoft care permite compatibilitatea încrucișată pe diferite platforme.

O caracteristică de *scripting* care este unică pentru Unity este reprezentată de proprietățile vizuale. Proprietățile vizuale permit ca variabilele publice din script-uri să fie modificate direct în editor fără a atinge codul. Acest lucru este util deoarece permite reglarea rapidă în joc pentru a obține exact valoarea potrivită. Acest instrument permite, de asemenea, proiectanților de jocuri, care nu sunt familiarizați cu codul, să modifice aspectul proiectului.

**Publicații si tutoriale spre Unity**

Înainte de Unity 3.0, tutorialele și publicațiile pentru Unity erau rare și greu de găsit, Unify Community Wiki fiind unul dintre foarte puținele locuri de unde se puteau obține informații. Acest lucru s-a schimbat atunci când Will Goldstone[1] a publicat cartea Unity Game Development. Această carte a fost prima carte publicată care a permis utilizatorilor să învețe Unity urmând tutorialele oferite. Înainte de aceasta, dezvoltatorii trebuiau să învețe prin încercări și printr-o mulțime de cercetări pe internet. Odată cu creșterea popularității lui Unity, a apărut o nevoie mai mare de publicații și tutoriale pentru a satisface comunitatea în creștere. De asemenea, faptul că produsul de bază Unity a devenit gratuit a crescut interesul pentru publicațiile legate de Unity.

În data de 10 noiembrie 2010, Unity Technologies a anunțat și a lansat Unity Asset Store. Această piață este construită chiar în IDE, astfel încât dezvoltatorii pot naviga și achiziționa *asset*-uri pentru aplicația lor fără a mai să părăsească proiectul lor. De asemenea, dezvoltatorii individuali pot posta *asset*-uri și pachete. Acest lucru oferă comunității Unity o metodă cu totul nouă de a-și monetiza abilitățile și talentele. Unity Technologies păstrează 30% din venituri, iar designerul de active primește restul de 70%. *Asset*-urile de top produc suficienți bani pentru ca o persoană să trăiască confortabil. Mulți dezvoltatori folosesc, de asemenea, Asset Store pentru a-și finanța proiectele lor mai avantajoase în paralel. În prezent, există mii de pachete de active disponibile în magazin, iar altele sunt adăugate în fiecare zi.

**„Conducta” *asset*-urilor**

Conducta *asset*-urilor Unity este una din cele mai bune din industrie în ceea ce privește ușurința de utilizare și compatibilitatea cu *software*-ul actual de dezvoltare a activelor. Pentru a importa un *assset* în Unity, utilizatorul trebuie pur și simplu să îl tragă în fereastra proiectului și Unity se ocupă de toate celelalte. Atunci când este editat un *asset*, Unity actualizează automat *assetul* cu cele mai recente modificări și le distribuie în întregul proiect.

**De ce C# in unity?**

Cel mai simplu răspuns ar fi ca C# poate fi utilizat și în afara Unity-ului pentru a programa. C# este cel mai popular dintre cele 3 limbaje folosite in Unity si nativ pentru .Net. Învățarea C# va fi utilă și în viitor. JavaScript în Unity nu este javascript-ul real care se folosește si în afara Unity-ului, iar Boo nu este foarte cunoscut.

Totodata, C# se execută mai rapid și are unele caracteristici de limbaj care oferă avantaje semnificative față de UnityScript, care probabil sunt mai greu de înțeles la început, dar care sunt extrem de utile. Majoritatea *asset*-urilor de pe magazin sunt dezvoltate folosind C# sau oferă atât JS, cât și C#.

Dezvoltarea cu C# are ca avantaj oferit de Visual Studio. Acesta oferă o funcție de completare a codului mai bună decât dezvoltarea Mono .

Nu în ultimul rând, toată documentația și toate tutorialele pentru Unity sunt scrise în prezent în C# și UnityScript, iar dintre cele trei limbaje disponibile pentru Unity, doar exemplele în C# sunt oferite în documentația .NET. Astfel, C# este și mai accesibil în materie de învățare.

**Utilizări în afara jocului**

Recent, dezvoltatorii au început să realizeze că *game engine*-urile pot fi folosite cu succes pentru aplicații care nu sunt jocuri. De exemplu, arhitecții pot realiza cu ușurință prototipuri de idei, artiștii pot crea instalații artistice interactive sau cercetătorii le pot utiliza pentru vizualizarea datelor. Totuși, invers nu se aplică. Este practic imposibil să creezi un joc decent folosind un program de proiectare asistată de calculator (CAD).

**Capitolul 3 – Șabloane de proiectare**

In acest capitol se vor prezenta șabloanele de proiectare care s-au folosit pentru realizarea acestui proiect.

**Fabrica / Gruparea obiectelor**

Modelul de proiectare Fabrica este legat de crearea de noi instanțe sau obiecte dintr-un plan. În Unity, folosim mai ales prefababricate, deoarece acestea sunt deja configurate cu toate componentele pe care am dorit să le aibă un obiect. Așadar, din punct de vedere al codului, acest model de proiectare necesită doar implementarea metodei GetNewInstance() de către noi.

Un exemplu excelent poate fi o instanțiere a gloanțelor sau a dușmanilor din prefabricate. Dar există și anumite limitări ale acestui model. Crearea unor cantități mari de obiecte care sunt distruse în continuu poate duce la un joc instabil sau chiar la blocarea jocului din cauza fragmentării memoriei. Pentru a preveni acest lucru, se folosește o tehnică de optimizare numită gruparea obiectelor. Acest lucru este esențial atunci când se creează un joc cu o mulțime de gloanțe, inamici sau obstacole.

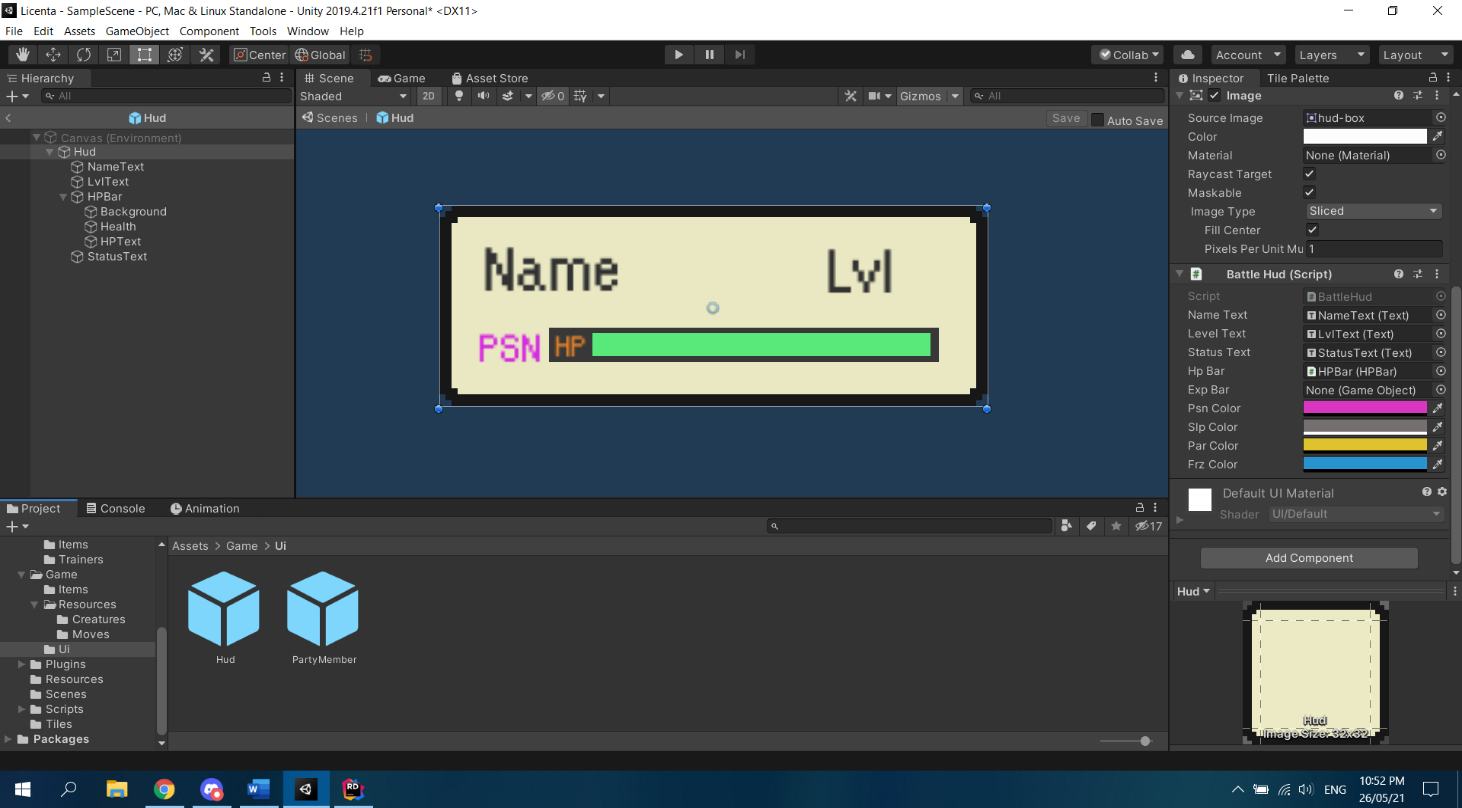


Figura 3.1 - Captura cu browser-ul de proiect al editorului unity

O imagine care conține text, monitor, captură de ecran, computer

Descriere generată automat

Figura 3.2 - Captura cu browser-ul de proiect al editorului unity

Am folosit Fabricarea pentru a inițializa componentele de acelaș fel precum bara de viața a fiecărei creaturi care se afla in lupta sau a fiecărei creaturi disponibile.

**Mașina cu stări**

Mașina cu stări este unul dintre cele mai utilizate modele de proiectare în dezvoltarea de jocuri. Este util pentru o varietate de scopuri, cum ar fi AI, animații, controlere de joc, logica jocului, dialoguri, *cutscenes* și multe altele.

După cum sugerează și numele, este o mașină cu stări diferite. Iar prin mașină înțelegem un script care va păstra aceste stări.

Cel mai simplu exemplu ar fi un simplu AI care va patrula o anumită zonă de pe o hartă. Acest AI ar putea avea 2 stări generale, cum ar fi starea de așteptare și starea de mișcare.

Un alt exemplu ar putea fi o mașină cu stări pentru un întreg joc, care ar putea avea stări precum starea de meniu, starea de joc, starea de terminare joc (la fel cum am folosit si in acest joc).

O imagine care conține text, captură de ecran, computer, electronice

Descriere generată automat

Figura 3.3 - Captura cu browser-ul de proiect al editorului unity

O imagine care conține text

Descriere generată automat

Figura 3.4 - Captura cu browser-ul de proiect al editorului unity

O imagine care conține text

Descriere generată automat

Figura 3.5 - Captura cu browser-ul de proiect al editorului unity

Iar fiecare stare are codul sau propriu de efectuat atunci cand starea apelata.

**Observatorul**

Observatorul este un model de proiectare foarte util în ingineria software, utilizat pentru a transmite automat, notificări între obiecte, la orice schimbare de stare. Este important și în dezvoltarea de jocuri, deoarece de cele mai multe ori trebuie să informăm anumite obiecte despre apariția unui eveniment și să inițiem acțiuni legate de aceste obiecte.

Există două tipuri de obiecte în acest model:

1. Observabilul - acesta este obiectul care invocă efectiv acțiunea inițială. Acesta are o listă de observatori care trebuie să fie notificați atunci când are loc evenimentul.

2. Observator - aceste obiecte sunt pregătite să acționeze la notificarea observabilului.

*OnClickListners* poate fi luat ca un bun exemplu al acestui scenariu. Atunci când utilizatorul face clic pe un buton (observabil), toți ascultătorii (observatorii) atașați acestuia sunt notificați. Apoi, în urma acestei notificări, aceștia inițiază separat metodele *onClick* legate de eveniment.

Spre exemplu, am invocat curentSelection pentru a primi elementul care este selectat din meniuO imagine care conține text

Descriere generată automat:

Figura 3.6 - Captura cu browser-ul de proiect al editorului unity

Sau am folosit acțiunea *onMoveOver* pentru a retine când jucătorul și-a terminat acțiunea de mers:

O imagine care conține text

Descriere generată automat

Figura 3.7 - Captura cu browser-ul de proiect al editorului unity

**Capitolul 4**

**Proiectarea si dezvoltarea unui joc video particular**

Capitolul 4 prezinta modul de dezvoltare a jocului folosind platforma Unity si instrumentele corespunzatoare