**UNIVERSITATEA BABEŞ-BOLYAI**

**Facultatea de Matematică si Informatică**

**Informatică Română**

**Lucrare de licenţă**

**Evolutia meseriilor in timp**

**Joc educational 2D**

Absolvent,

Cezara-Teodora **Neagu**

Coordonator ştiinţific,

Lector Universitar Dr. Ioan **Lazar**

**2020**

**Cuprins**

1. Introducere
   1. Motivatia
   2. Istoria jocurilor video
2. Prezentarea subiectului. Istoria meseriilor
   1. Preistoria
   2. Antichitatea

2.3. Evul Mediu

2.4. Renasterea

2.5. Contemporan

1. Studiu de caz
   1. Introducerea termenilor
   2. Erele
   3. Sfarsitul de era
   4. Gameplay
2. Tehnologiile si limbajele folosite
   1. Unity
   2. C#
   3. Json.NET
   4. Adobe Illustrator
3. Dezvoltarea aplicatiei
   1. Arhitectura folosita
   2. Clasele utilizate/create
   3. Folosirea fisierelor json
   4. Setarile jocului
4. Concluzii
5. Bibliografie

1.Introducere

* 1. Motivatia

De-a lungul existentei lor, oamenii au incercat sa isi faca viata din ce in ce mai usoara, cautand diferite solutii, unele fizice, altele spirituale, pentru problemele lor. Toate aceste solutii au putut fi in final puse in practica pe baza unor activitati, care de-a lungul timpului au ajuns sa poarte denumirea de “meserii”.

O mare parte din meseriile practicate in ziua de astazi isi au originile cu milioane de ani in urma, mai exact in preistorie. Domenii precum medicina, constructiile sau agricultura au trecut prin nenumarate etape de dezvoltare pentru a ajunge la forma lor actuala, vizibila in viata de zi cu zi. Multe sacrificii mai mult sau mai putin stiute au fost savarsite pe parcursul dezvoltarii acestora.

Medicina reprezinta probabil domeniul care a lasat in urma cele mai multe victime datorita mijloacelor dure prin care aceasta a fost aplicata. Neavand cunostina de cauza la acele vremuri, au fost necesare foarte multe incercari si interventii esuate asupra oamenilor pana sa se ajunga la o “solutie” de tratare a problemelor potrivita, acestia suferind dureri inimaginabile ca urmare a interventiilor gresite asupra corpurilor lor, de cele mai multe ori avand ca rezultat agravarea situatiei.

Constructiile, un alt domeniu indispensabil in ziua de astazi, au fost de asemenea o provocare pentru civilizatie. Ca urmare a apritiei nevoii de a avea un adapost, primele fiinte umane care au simtit aceasta necesitate au incercat pe baza posibilitatilor de la acea vreme sa puna in aplicare acest lucru. Neavand cunostintele de baza in materie de fizica si matematica pe care se bazeaza acest domeniu la momentul actual, oamenii respectivi in loc sa isi construiasca un adapost, de cele mai multe ori confectionau capcane pentru ei insasi, sfarsind intr-un mod tragic. Pe masura ce tot mai multe incercari au esuat, pe baza acestora s-a imbunatit intreg procesul ajungand intr-un final sa se realizeze diferite constructii care indeplineau cu adevarat rolul pentru care acestea au fost create, nemaipunand in pericol siguranta oamenilor.

Desi poate parea un domeniu cu mai putine riscuri in dezvoltarea acestuia decat cele prezentate anterior, agricultura, nu este nici pe departe. In primul rand, hrana reprezinta o necesitate primara in randul omenirii, aceasta nevoie trebuind a fi indeplinita intr-un mod constant. Aceasta este cauza pentru care agricultura si procurarea hranei au devenit un mare risc asupra oamenilor la vremea respectiva. Nestiind la inceput ce alimente pot fi considerate comestibile, omenirea a incercat de-a lungul vremii aproximativ orice forma de aliment existenta, astfel ajungandu-se la intoxicatii si alte probleme grave ce puneau in pericol sanatatea corpului uman.

Motivul pentru care am ales sa dezvolt aceasta aplicatie este de a aduce la cunostinta oamenilor de orice varsta originile meseriilor principale din cotidian, simuland evolutia acestora intr-un mod interactiv. Nu multi oameni cunosc originea meseriilor din ziua de astazi, evolutia lor si etapele prin care s-a trecut pentru a ajunge la ceea ce astazi consideram o normalitate pentru care nu suntem suficient de recunoscatori. Consider ca fiecare persoana ar trebui sa constientizeze importanta meseriilor din cotidian, dar si cea a oamenilor care au pus bazele lor.

“Sănătatea este o comoară pe care puţini ştiu să o preţuiască, deşi aproape toţi se nasc cu ea.” Hipocrate

1.2. Istoria jocurilor video

\*\*\*INCA CEVA INTRODUCERE\*\*\*

Pentru a descrie ideea de joc video, propun intai o descriere in amanunt a termenului ‘joc’.

Jocurile reprezinta o activitate recreationala ce au aparut cu mii de ani in urma (primul joc observat fiind “Jocul Regal al lui Ur”, aparut in secolul 3.000 i.Hr.), acestea fiind si in prezent o activitate practicata de multi dintre noi, in diferite forme. Elementele principale care definesc un joc sunt un obiectiv, un set de reguli si o realitate preconitaza. Jocurile video reprezinta un subset al acestei activitati, ele reusind sa -\_\_-

“Există o controversă în ceea ce privește care este primul joc video. Deși termenul de video se aplică unei imagini generate electronic pe un monitor, jocuri video sunt considerate a fi pe mai multe platforme.

Cel mai vechi joc electronic interactiv a fost creat de Thomas T. Goldsmith Jr. și Estle Ray Mann pe un tub catodic. Patentul invenției a fost înregistrat la 25 ianuarie 1947. Jocul era asemănător cu un radar din al doilea război mondial.”

"Today the term "video game" has completely shed its purely technical definition and encompasses a wider range of technology. While still rather ill-defined, the term "video game" now generally encompasses any game played on hardware built with electronic logic circuits that incorporates an element of interactivity and outputs the results of the player's actions to a display.”

2. Prezentarea subiectului. Istoria meseriilor

2.1. Preistoria

2.1.1. Inceputul

„Primele rase ale lumii [...] au călătorit fără efort pe cer și l-au întâlnit pe Dumnezeu față în față. Traversând un pod care unea cerul și pământul, au ajuns la jaluzele superioare. Dar, printr-o oarecare eroare misterioasă, această legătură a fost perturbată, podul rupt. Încetul cu încetul, după acest eveniment zguduitor din lumea primordială, ființele simțitoare și-au pierdut înțelepciunea inițială, astfel încât ceea ce era accesibil pentru toată lumea este atins astăzi doar de către șamani.”

4. Tehnologiile si limbajele folosite

4.1. Unity

Unity este un motor grafic aparut in anul 2005 ce are la baza cod sursa in limbaj C++.

Programul este folosit pentru a crea jocuri atat 3D, cat si 2D si ofera suport pentru majoritatea platformelor si sistemelor de operare, acestea fiind doua din motivele pentru care 50% din totalul de jocuri dezvoltate la momentul de fata au fost create folosind acest program. Alte motive tin de usurinta folosirii interfetei si multitudinea de tutoriale aflate la dispozitia oricarui incepator.

Obiectul fundamental in crearea de jocuri cu Unity este GameObject-ul, acesta fiind elementul de baza al camerei, caracterelor, decorului etc. Functionalitatea acestui obiect este data de o ‘componenta’, atasata la acesta. Exista componente predefinite, insa acestea pot fi create si de catre utilizator cu ajutorul scriputilor pentru a-i putea indeplini exact cerintele. Majoritatea programatorilor care folosesc acest motor grafic aleg sa creeze scripturile in C#, insa Unity ofera suport si pentru alte limbaje, cum ar fi JavaScript.

Din multitudinea de jocuri cunoscute la nivel international create cu acest motor grafic se pot mentiona Cuphead, Pokemon GO, Hearthstone.

( <https://unity.com/our-company> )

(<https://docs.unity3d.com/Manual/> )

4.2. C#

C# (C-Sharp) este un limbaj de programare orientat pe obiecte, dezolvat de catre Microsoft si lansat publicului in anul 2000. Limbajul este asemanator sintactic cu C++ si Java, fiind usor de invatat datorita nivelul ridicat de expresivitate. Comparativ cu C++, C# aduce in plus enumeratii, expresii lambda, acces direct la memorie si multe altele pentru a usura rezolvarea problemelor in asa fel incat atentia sa cada asupra crearii de noutate.

Fiind un limbaj orient pe obiecte, C# accepta conceptele de incapsulare, mostenire si polimorfism.

(<https://docs.microsoft.com/en-us/dotnet/csharp/> )

4.3. Json .NET

JSON (JavaScript Object Notation) este un format de reprezentare si interschimb de date intre aplicatii informatie. Este un format text, inteligibil pentru oameni, utilizat pentru reprezentarea obiectelor și a altor structuri de date și este folosit în special pentru a transmite date structurate prin rețea, procesul purtând numele de serializare.

4.4. Adobe Illustrator

Adobe Illustrator este un editor grafic vectorial creat si distribut de Adobe Inc in anul 1987.

5. Dezvoltarea aplicatiei

5.2. Clasele utilizate/create

5.2.1. Pachetul Model

5.2.1.1. GameData

Pentru a putea gestiona datele ce tin de instanta unui joc (numarul de mancare, resurse, populatie, era curenta, job-ul selectat, nivelul de energie) clasa contine atributele necesare, fiind statice pentru a asigura ca fiecare cerere a unui atribut, la un moment dat, returneaza acelasi rezultat. Pe langa atribute, clase contine si metode statice, cu nume sugestive: saveGame() salveaza datele jocului curent in PlayerPref (clasa predefinita de Unity ce stocheaza si acceseaza preferintele jucatorului intre sesiunile de joc), loadGame() le incarca din PlayerPrefs, newGame() le reseteaza la cele initiale, iar restoreEnergy() restabileste nivelul de energie la cel maxim, functia fiind apelata la fiecare sfarsit de era.

5.2.1.2. Chromozome

Clasa Chromozome contine atribute ce reprezinta capacitatile populatiei, fiind reprezentate ca numere reale intre 0 si 1, unde 0 inseamna ca acea capacitate nu este deloc dezvoltata, iar 1 inseamna ca este dezolvatata la maxim. Aceasta instanta este instantiata la fiecare inceput de joc cu numere aleatoare, **intre 0 si 0.2**, pentru a incepe in mod realist cu un procent pe care l-ar fi avut primii oameni.

Pe langa folosirea unei instante a clasei pentru a defini capacitatile populatiei, exista si instante ce vor modifica cromozomul populatiei, aceste instante fiind atribute ale clasei ActionButtonData prezentate mai jos.

5.2.1.3 ActionButtonData

Pentru fiecare meserie, utilizatorul poate selecta dintr-o lista actiuni ce vor trebui indeplinite de catre muncitorul respectiv. Aceste date sunt salvate in fisiere json cu nume ales strategic pentru o mai buna abastractizare. De exemplu, pentru saman, path-ul fisierului este “Era1/shaman”, acesta fiind format din numele erei curente impreuna cu numele job-ului selectat. In acest fisier se afla o lista de obiecte de tip ActionButtonData (clasa regasita in pachetul “model” al solutiei), insa pentru a intelege structura butonului de actiune, pun la dispozitie un exemplu de astfel de buton, asa cum este prezentat in interfata cu utilizatorul:



Astfel, un obiect de tip ActionButtonData contine urmatoarele atribute:

- atribute observate de catre utilizator

-text:string, numele actiunii, in exemplu : “Construieste corturi”

-subtext:string, cerintele necesare pentru a putea indeplini actiunea, in exemplu: “Ai nevoie de 10 mancare, 20 resurse”

-atribute ce tin de decizia de a executa actiunea:

-requested\_ability:string, abilitatea solicitata pentru indeplinirea actiunii

-requested\_ability\_percentage:double, procentul minim necesar corespunzator abilitatii declarate anterior

-atribute ce modifica datele jocului, daca actiunea poate fi indeplinita:

-affected\_food:int, daca numarul este pozitiv, mancarea curenta va creste, insa daca este negativ, atunci va folosi

-affected\_resources:int, numarul de resurse ce afecteaza numarul curent de resurse(asemanator ca la mancare)

-affected\_people:int, numarul de oameni in plus/in minus

- affected\_chromozome:Chromozome

-atributele folosite daca actiunea nu poate fi indeplinita:

-bad\_message:string, in cazul in care procentul minim nu este atins, se afiseaza acest mesaj intr-un panel de alerta

-lose\_food:int

-lose\_resources:int

-lose\_people:int

-lose\_affected\_chromozome:Chromozome