**Proiect III**

**Tehnologia informatiei**

Pistritu Ionut-Laurentiu

# INTRODUCERE

Limbajele de programare sunt mijloace de comunicare între utilizator şi sistemul de calcul. Prin intermediul lor, omul transmite calculatorului ordinele necesare pentru executarea unei operaţii. Descrierea cu ajutorul unui limbaj de programare a etapelor necesare rezolvării unei anumite probleme se numeşte program. Astfel programul apare ca o succesiune de comenzi transmise calculatorului pe care acesta le execută. Este evident că programul transmis calculatorului pentru a fi executat nu este altceva decât descrierea, în limbajul de programare ales, a algoritmului de rezolvare a problemei respective.

Se ştie că un calculator nu cunoaşte decât un singur limbaj – limbajul calculatorului, numit cod-maşină. Primele generaţii de calculatoare erau programate direct în cod-maşină, lucru deosebit de dificil de realizat. O dată cu apariţia necesităţilor tot mai mari de prelucrare a informaţiilor şi o dată cu progresele înregistrate în domeniul construcţiei calculatoarelor, s-a impus un salt calitativ în ceea ce priveşte programarea calculatoarelor şi anume trecerea de la programarea în cod-maşină la programarea simbolică. În acest fel programatorul se preocupă mai mult de metoda de rezolvare a problemei, fără a se mai preocupa de amănunte legate de calculator. Astfel, pe lângă o sporită comoditate în scrierea programelor, această schimbare a însemnat şi câştigarea unei independenţe a programării faţă de caracteristicile fizice ale calculatorului.

# Prezentarea IDE-ului folosit

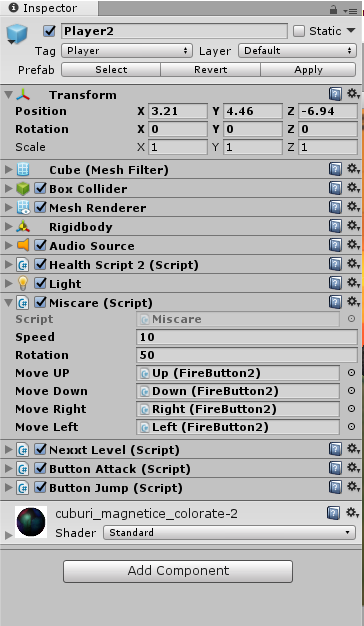
Un mediu integrat de dezvoltare (IDE) este o suită de software care consolidează instrumentele de bază pe care dezvoltatorii trebuie să le scrie și să testeze software-ul. De obicei, un IDE conține un editor de cod, un compilator sau un interpret și un program de depanare pe care dezvoltatorul le accesează printr-o singură interfață grafică de utilizator (GUI). Un IDE poate fi o aplicație independentă sau poate fi inclusă ca parte a uneia sau mai multor aplicații existente și compatibile.

Unity 3D este un engine folosit pentru jocuri 3D, dar cu care se pot creea fara nicio problema si jocurile 2D. Acesta este special pentru faptul ca poate fi folosit de persoane care nu prea au cunostinte de programare, insa si de cei cu mai multa experienta, iar acest lucru se face la un nivel foarte avansat.

Aduce in jur de trei limbaje de programare, precum C#, Boo si Unity Javascript, posibilitatea de a lucra in 3D (aproximativ la fel cu Maya) si poate fi folosit de la Windows sau Mac pana la sistemele de operare folosite pe telefoanele mobile, precum Android sau IPhone.

Cel mai interesant lucru despre Unity este faptul ca pot fi realizate foarte multe tipuri de jocuri, incepand de la cele 2D, pana la jocuri mult mai complicate sau complexe precum shootere, curse de masini, avioane etc. Ei bine, se pare ca acest lucru a facut o multime de persoane sa fie impresionate si sa doreasca a invata cat mai multe lucruri despre acesta. Motiv pentru care exista si o multime de gameri care au cunostintele necesare pentru a oferi suport celor mai putin initiati in domeniu.

Ideea pe care Unity o are la baza este de a oferi gamerilor jocuri online cat mai complexe si accesibile, astfel incat sa se bucure la maxim de ele.

** **

# Prezentarea limbajului de programare folosit

**Scurt istoric**

Lansat publicului in iunie 2000 si oficial in primavara anului 2002, C# este un limbaj de programare care combina facilitati testate de-a lungul timpului cu inovatii de ultim moment. Creatorii acestui limbaj au fost o echipa de la firma Microsoft condusa de Anders Hejlsberg. Desi limbajul este creat de Microsoft, acesta nu este destinat doar platformelor Microsoft. Compilatoare C# exista si pentru alte sisteme precum Linux sau Macintosh. Creat ca instrument de dezvoltare pentru arhitectura .NET, limbajul ofera o modalitate facila si eficienta de a scrie programe pentru sistemul Windows, internet, componente software etc. C# deriva din doua dintre cele mai de succes limbaje de programare: C si C++. De asemenea, limbajul este o “ruda” apropiata a limbajului Java. Pentru o mai buna intelegere a limbajului C# este interesant de remarcat care este natura relatiilor acestuia cu celelalte trei limbaje mentionate mai sus. Pentru aceasta, vom plasa mai intai limbajul C# in contextul istoric determinat de cele trei limbaje. Limbajul C. Programarea structurata Limbajul C a fost inventat de catre Dennis Ritchie in anii ‘70 pe un calculator pe care rula sistemul de operare UNIX. Limbajul C s-a dezvoltat in urma revolutiei programarii structurate din anii ’60. Inainte de programarea structurata, programele erau greu de scris si de inteles din cauza logicii. O masa incalcita de salturi, apelui si reveniri, greu de urmarit era cunoscuta sub numele de cod spaghetti. Datorita sintaxei sale concise si usor de utilizat, in anii ‘80, limbajul C a devenit cel mai raspandit limbaj structurat. Limbajul C are insa limitele sale. Una dintre acestea o reprezinta incapacitatea de a lucra cu programe mari. Limbajul C ridica o bariera atunci cand programul atinge o anumita dimensiune. Acest prag depinde de program, instrumentele folosite, programator, dar este posibil sa se situeze in jurul a 5000 de linii de cod.

# Platforma pe care va rula aplicatia

**Android** este o platformă software și un sistem de operare pentru dispozitive și telefoane mobile bazată pe nucleul Linux, dezvoltată inițial de compania Google, iar mai târziu de consorțiul comercial Open Handset Alliance. Android permite dezvoltatorilor să scrie cod gestionat în limbajul Java, controlând dispozitivul prin intermediul bibliotecilor Java dezvoltate de Google.[3] Aplicațiile scrise în C și în alte limbaje pot fi compilate în cod mașină ARM și executate, dar acest model de dezvoltare nu este sprijinit oficial de către Google.

Lansarea platformei Android la 5 noiembrie 2007 a fost anunțată prin fondarea Open Handset Alliance, un consorțiu de 48 de companii de hardware, software și de telecomunicații, consacrat dezvoltării de standarde deschise pentru dispozitive mobile.  Google a lansat cea mai mare parte a codului Android sub licența Apache, o licență de tip free-software și open source.

**Istoric**

În iulie 2005 Google a achiziționat Android, Inc, o mică companie de tip startup cu sediul în Palo Alto, California, SUA. Cofondatorii companiei Android, care au continuat să muncească la Google, au fost Andy Rubin (cofondator al Danger), Rich Miner(cofondator al Wildfire Communications, Inc), Nick Sears (fost vicepreședinte al T-Mobile) și Chris White (unul dintre primii ingineri ai WebTV). La acea dată se cunoștea foarte puțin despre Android, Inc., doar că făceau software pentru telefoane mobile.Aceasta a cauzat zvonuri că Google ar plănui să intre pe piața telefoniei mobile, deși era neclar ce funcție ar putea îndeplini în această piață.

La Google, echipa condusă de Rubin a dezvoltat un sistem de operare pentru dispozitive mobile bazat pe Linux, pe care l-au prezentat producatorilor de telefoane mobile și operatorilor de rețele de telefonie mobilă, cu perspectiva de a asigura un sistem flexibil, upgradabil. Google a raportat că a aliniat deja o serie de parteneri producători de componente hardware și software la noul concept, și a semnalat operatorilor de rețele de telefonie mobilă că era deschis la diferite grade de cooperare din partea acestora. Mai multe speculații că Google ar fi putut intra pe piața telefoniei mobile au apărut în decembrie 2006.Rapoarte de la BBC și Wall Street Journal au remarcat faptul că Google își dorea căutarea web și aplicațiile sale pe telefoane mobile și că lucra din greu către acest țel. Presa și siturile de știri au publicat curând zvonuri că Google ar dezvolta un dispozitiv mobil marca Google. A urmat și mai multă speculație, susținând că în timp ce Google definea specificațiile tehnice, ar fi demonstrat prototipuri producătorilor de telefoane mobile și operatorilor de rețea. S-a raportat că până la 30 de telefoane prototip operau deja pe piață.

**Ce este un JAR?**

Formatul JAR (Java Archive) permite impachetarea mai multor fisiere intr-o singura arhiva. Un fisier JAR va contine in mod tipic fisiere .class si alte fisiere auxiliare (de exemplu imagini, fisiere audio etc.).  
Formatul JAR a fost introdus de catre SUN incepand cu versiunea 1.1 a JDK (Java Development Kit). O data cu versiunea 1.2, si formatul JAR a cunoscut unele imbunatatiri.  
          Iata cateva dintre caracteristicile fisierelor JAR:

Securitate. Continutul unui fisier JAR poate fi semnat digital, astfel incat numai persoanele care recunosc semnatura pot beneficia de privilegiile de securitate. Voi reveni pe parcursul articolului.

Scaderea timpului de download. Daca toate fisierele unui aplet (inclusiv auxiliare) sunt adunate intr-un singur fisier, atunci acesta poate fi descarcat de catre browser intr-o singura transactie HTTP (se stie ca pentru fiecare nou fisier pe care browserul trebuie sa-l aduca de pe server, se realizeaza o noua conexiune TCP).

Compresie. Formatul JAR permite comprimarea fisierelor (cu formatul ZIP).

"Sigilarea" pachetelor (numai versiunea 1.2). Pachetele dintr-un fisier JAR pot fi "sigilate". Acest lucru inseamna ca toate fisierele definite in pachetul respectiv trebuie sa se gaseasca in acelasi fisier JAR pentru ca toate sa fie interpretate cu aceeasi versiune de Java.

Informatii privind vesriunea pachetului. Un fisier JAR poate pastra date despre fisierele pe care le contine, de exemplu versiunea, producatorul, etc.

Portabilitate. Aceasta caracteristica se aplica limbajului Java in general, cu tot ce este legat de el, prin urmare si mecanismul acesta de arhivare este o parte standard a platformei Java API.

          Dupa aceasta introducere, vom vedea in continuare cum se creeaza si cum se folosesc fisierele JAR si alte fisiere importante continute in JAR (fisierul manifest). Apoi voi prezenta cateva elemente legate de semnarea si autentificarea fisierelor JAR.  
          Fisierele JAR sunt arhive de tip ZIP. Deci ele pastreaza caracteristicile formatului ZIP, si anume: compresia datelor fara pierdere de informatii, arhivare, decompresie.  
          Pentru a realiza sarcinile unui fisier JAR trebuie sa folosim unealta furnizata ca parte a JDK: Java Archive Tool. De fapt aceasta unealta este folosita prin comanda jar si vom vedea in continuare cum o vom folosi.

**Nota:**Indiferent de mediul de dezvoltare Java pe care il folositi (Symantec Visual Cafe, Borland JBuilder, Oracle JDeveloper sau oricare altul), acesta are ca motor de compilare si interpretare pachetul JDK de la Sun. Aceste medii ofera in plus interfete prietenoase menite sa ajute programatorul, eventual ofera in plus o suma de alte clase care pot fi folosite, ofera unelte vizuale. Dar toate acestea folosesc de fapt uneltele standard oferite de Sun. Ele se afla de obicei in directorul java/bin, aflat in directorul home al programului respectiv instalat.  
          Revenind la tema articolului de fata, desigur ca toate aceste medii de dezvoltare java au optiuni de creare a fisierelor JAR. Dar acestea apeleaza de fapt utilitarul jar din JDK. De aceea voi prezenta aici modul de folosire a acestui utilitar.

**Crearea fisierelor JAR**

Formatul de baza pentru crearea unui fisier JAR este urmatorul : jar cf fisier\_jar fisiere\_input Optiunile si argumentele care pot fi folosite in aceasta comanda sunt :

* c = create - creeaza un fisier JAR;
* f = file - indica faptul ca output-ul comenzii se va duce intr-un fisier, si nu la stdout, mai precis in fisierul fisier\_jar ;
* fisier\_jar - numele fisierului jar care va resulta si care va avea numele fisier\_jar.jar ;
* fisiere\_input - este o lista de fisiere care vor fi plasate in fisierul .jar. Lista va fi separata prin spatii. Simbolul \* poate fi folosit pentru a indica un sablon de mai multe fisiere. Daca unul dintre fisierele-input este nume de director, atunci toate fisierele continute in acest director vor fi adaugate arhivei.

Comanda de mai sus va genera un fisier. jar comprimat si il va plasa in directorul curent. Comanda va genera si un fisier manifest, despre care voi vorbi mai tarziu in acest articol. Urmatoarele optiuni mai pot fi adaugate comenzii de creare a unui fisier .jar :

* v = verbose - afiseaza pe ecran informatii referitoare la fisierele adaugate (numele, rata de comprimare etc) pe masura ce fisierul JAR este construit.
* 0 = zero - se foloseste in cazul in care nu se doreste ca fisierul .jar sa fie comprimat. De exemplu un fisier necomprimat va mari viteza de executie a fisierelor continute in arhiva pentru ca decomprimarea lor nu mai este necesara (operatie consumatoare de timp). Bineinteles ca timpul de download va fi mai mare in acest caz, pentru ca dimensiunea fiiserului .jar va fi mai mare. Dar de exemplu in cazul folosirii unui fisier .jar in Intranet, unde timpul suplimentar de download nu este semnificativ mai mare, aceasta optiune poate fi o solutie buna pentru cresterea vitezei de executie.
* -C = change directory (numai versiunea 1.2) - schimba directorul in timpul executiei comenzii. Astfel, structura de directoare nu mai este pastrata, toate fisierele fiind plasate pe primul nivel, fara nici o structura de directoare.

Comanda urmatoare :

jar cvf joc.jar joc.class img

creeaza fisierul joc.jar din clasa joc.class si toate fisierele continute in directorul img. Comanda afiseaza informatii pe ecran pe parcursul crearii fisierului, si anume ceva de genul:

adding:joc.class (in=2145)

(out=1755) (deflated 36%)

adding:img/ (in=0)

(out=0) (stored 0%)

adding:img/sfera.gif (in=648)

(out=632) (deflated 2%)

adding:img/bila.gif (in=314)

(out=310) (deflated 1%)

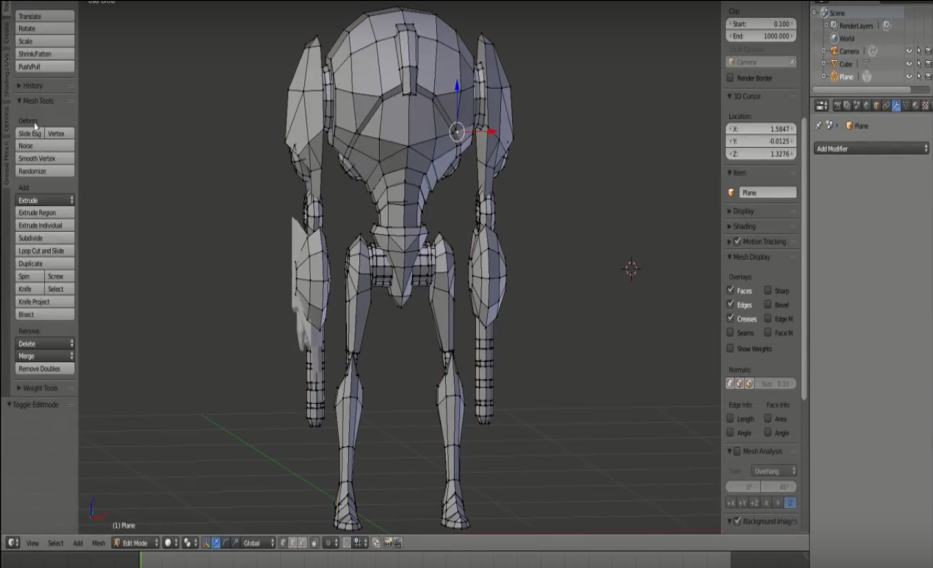
**Fisierele APK**

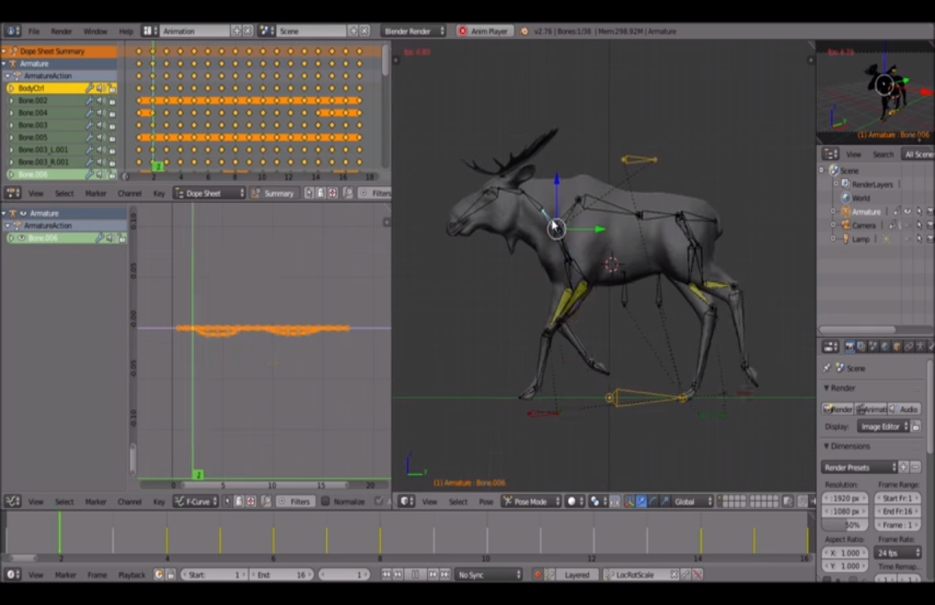
Cum Windows-ul folosește fișierele .exe pentru a instala un program, Android folosește aceste APK-uri. APK este un format de fișier folosit pentru a instala aplicații pe sistemele Android.

Poți instala fișiere APK pentru a avea acces la noi versiuni de aplicații Android, cu mult înainte ca versiunile oficiale să fie lansate. Aceste APK-uri de Android îți permit să vezi și să testezi noile funcții ale aplicației pe care în mod normal tu nu le vei primi pe telefonul tău. De asemenea, mai sunt acele softuri, aplicații, care sunt restricționate regional. Fișierele APK de asemenea îți permit instalarea aplicațiilor care nu se găsesc în PlayStore și care de altfel sunt foarte multe la număr.

**V. Blender**

**Blender 3D** este un program [software liber](https://ro.wikipedia.org/wiki/Software_liber) de grafică 3D. El poate fi folosit pentru crearea modelelor 3D, [mapare UV](https://ro.wikipedia.org/wiki/Mapare_UV), texturare, rigging (animarea prin folosirea "oaselor"), simularea apei, animatie, randare, particule și alte simulări computerizate, editare non-lineară, compositing și crearea aplicațiilor interactive 3D. Blender e disponibil pentru câteva [sisteme de operare](https://ro.wikipedia.org/wiki/Sisteme_de_operare), inclusiv [FreeBSD](https://ro.wikipedia.org/wiki/FreeBSD), [IRIX](https://ro.wikipedia.org/wiki/IRIX), [GNU/Linux](https://ro.wikipedia.org/wiki/GNU/Linux), [Microsoft Windows](https://ro.wikipedia.org/wiki/Microsoft_Windows), [Mac OS X](https://ro.wikipedia.org/wiki/Mac_OS_X), [Solaris](https://ro.wikipedia.org/w/index.php?title=Solaris_Operating_Environment&action=edit&redlink=1), [SkyOS](https://ro.wikipedia.org/w/index.php?title=SkyOS&action=edit&redlink=1), și [MorphOS](https://ro.wikipedia.org/w/index.php?title=MorphOS&action=edit&redlink=1). Blender are o varietate de funcții similare ca scop și complexitate ca alte programe 3D ca [Softimage|XSI](https://ro.wikipedia.org/w/index.php?title=Softimage_XSI&action=edit&redlink=1), [Cinema 4D](https://ro.wikipedia.org/w/index.php?title=Cinema_4D&action=edit&redlink=1), [3ds Max](https://ro.wikipedia.org/w/index.php?title=3ds_Max&action=edit&redlink=1), [Lightwave](https://ro.wikipedia.org/wiki/Lightwave) și [Maya](https://ro.wikipedia.org/wiki/Maya_(software)). Aceste funcții includ unelte pentru simulări avansate ca *rigid body*, *fluid*, *cloth* și *softbody dynamics*, unelte puternice pentru animarea caracterelor, materiale bazate pe *node*-uri și scripturi [Python](https://ro.wikipedia.org/wiki/Python).





Blender are o mărime relativ mică la instalare și rulează pe câteva platforme populare. Cu toate că deseori e distribuit fără documentație sau fără scene exemplificatoare complicate, programul abundă în funcționalități caracteristice software-urilor de modelare de top. Printre capabilitățile sale se numără:

* Suportă o varietate de forme geometrice primitive, inclusiv structuri [poligonale](https://ro.wikipedia.org/w/index.php?title=Poligonale&action=edit&redlink=1), modelare [subdivision surface](https://ro.wikipedia.org/w/index.php?title=Subdivision_surface&action=edit&redlink=1) rapidă, [curbe Bezier](https://ro.wikipedia.org/w/index.php?title=Bezier_curve&action=edit&redlink=1) și [suprafețe NURBS](https://ro.wikipedia.org/w/index.php?title=Nonuniform_rational_B-spline&action=edit&redlink=1), [metaballs](https://ro.wikipedia.org/w/index.php?title=Metaballs&action=edit&redlink=1), sculptarea poligoanelor, și [fonturi Typeface](https://ro.wikipedia.org/wiki/Font) vectoriale.
* Capabilități versatile de randare internă și integrare cu raytracer-ul [open source](https://ro.wikipedia.org/wiki/Open_source) [YafRay](https://ro.wikipedia.org/wiki/YafRay).
* Unelte de animație, inclusiv [cinematica inversă](https://ro.wikipedia.org/w/index.php?title=Inverse_kinematic_animation&action=edit&redlink=1), deformări de armatură (scheletale) și cu ajutorul laticelor, shape keys, keyframes, timeline, animație non-lineară, constraints, vertex weighting, dinamica softbody, inclusiv detectarea coliziunilor dintre structurile obiectelor, dinamica fluidelor, dinamica hard body, păr bazat pe sistemul de particule, și un sistem de particule cu funcție de detecție a coliziunilor.
* Scripting [Python](https://ro.wikipedia.org/w/index.php?title=Python_programming_language&action=edit&redlink=1) pentru crearea de noi unelte sau de prototipuri de unelte, game scripting logic, sau automatizarea anumitor sarcini.
* Editare video non-lineară de bază și capabilități de compoziție.
* [Game Blender](https://ro.wikipedia.org/w/index.php?title=Game_Blender&action=edit&redlink=1), un sub-proiect, oferă funcții de interactivitate cum ar fi detectarea coliziunilor, dynamics engine, și logică programabilă. Totodată permite și crearea de aplicații autonome ce pot rula în timp real, de la vizualizare arhitecturală până la programarea de jocuri video.
* În Blender, un Obiect (care reprezintă o entitate care interacționează cu mediul înconjurător) și Datele lui "personale" (forma/funcția propriu-zisă a obiectului) sunt clare și distincte. Relațiile Obiect-date sunt linkabile dinamic tot timpul, facilitând procese de modelare rapide care constituie funcții unice ale programului Blender.
* Un sistem intern de [fișiere](https://ro.wikipedia.org/wiki/Fi%C8%99ier) care permite "împachetarea" mai multor scene într-un singur fișier (denumit fișier ".blend").
* Toate fișierele ".blend" ale programului sunt compatibile între toate platformele, precum și forward și backward cu alte versiuni ale programului, și pot fi utilizate ca bibliotecă de unde poți lua content gata creat.
* Fișierele ".blend" pot fi automat salvate ca instanțe periodice de către program, fiind astfel mai probabilă supraviețuirea fișierului în eventualitatea unei căderi a programului.
* Toate scenele, obiectele, materialele, texturile, sunetele, imaginile, efectele de post-producție a unei întregi animații pot fi stocate într-un singur fișier ".blend".
* Diversele configurațiile ale interfeței sunt memorate în fișierul ".blend", astfel încât ceea ce salvezi e ceea ce obții atunci când îl încarci. Acest fișier poate fi stocat ca "user default" (preferințe implicite ale utilizatorului), așa încât configurația ecranului, cât și toate obiectele stocate în el, sunt utilizate de fiecare dată când pornești programul Blender.

Blender deține un sistem de fișiere intern ce permite introducerea mai multor scene într-un singur fișier (numit fișier ***.blend***).



* Toate fișierele ".blend" sunt compatibile cu celelalte versiuni ale programului și sunt cross-platform. Ele pot fi folosite ca o librărie pentru a împrumuta o textură sau un obiect.
* Programul Blender poate salva automat lucrul pentru a nu pierde datele în cazul unui crash.
* Toate scenele, obiectele, materialele, texturile, sunetele, imaginile, efectele pentru o întreagă animație sunt stocate într-un singur fișier ".blend", așa că ceea ce salvezi este ceea primești când îl încarci. Fișierul stochează și preferințele utilizatorului, cum ar fi aranjamentul ferestrelor, așa că la deschiderea unui fișier programul arată la fel ca la salvarea lui.

Într-un fișier ".blend" sunt stocate obiectele și relațiile dintre ele într-un mod greu de descifrat, făcând dificilă convertirea acestora dintr-un program extern, dar există multe convertoare integrate în Blender ce accesează datele obiectelor direct, prin [API](https://ro.wikipedia.org/wiki/Application_programming_interface), facilitând exportul către alte programe.

Blender organizează datele ca diferite tipuri de "*data block*-uri", ca *Objects*, *Meshes*, *Lamps*, *Scenes*, *Materials*, *Images* și altele. În Blender un obiect este alcătuit din mai multe "data block-uri", de exemplu un cub are cel puțin un datablock pentru Object și Mesh și, de obicei, are și un material. Acest lucru permite, de exemplu, duplicarea obiectelor dar păstrarea unui singur datablock la care sunt legate toate copiile obiectului inițial, permițând și modificarea tuturor obiectelor în același timp. Relațiile dintre data-blockuri pot fi schimbate și manual. Datablock-urile pot fi luate și din alte fișiere .blend, permițând păstrarea acestora ca biblioteci pentru a fi accesate ulterior.

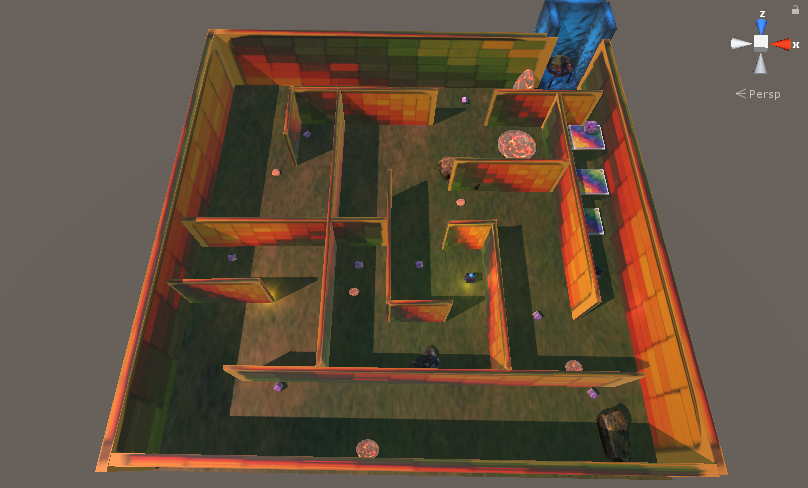
**VI. Povestire**

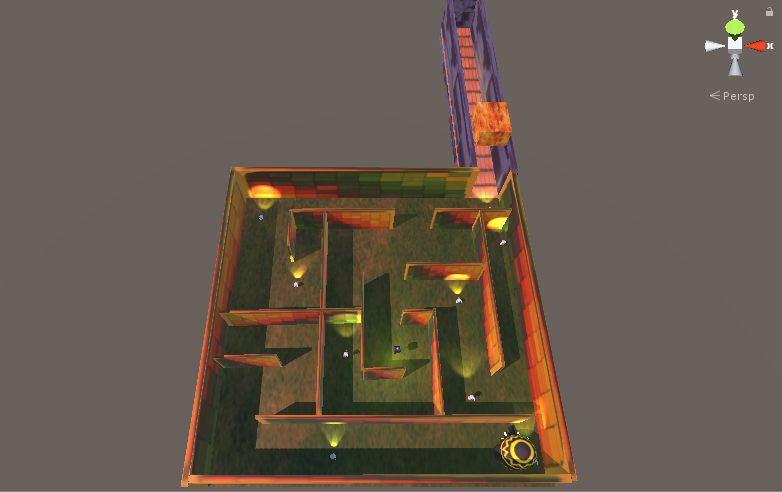
Lucrarea de fata prezinta realizarea unui proiect grafic, mai exact a jocului intitulat “Ratacit in labirint”. Fiind o oportunitate de a invata mai bine sa utilizez practic informatiile acumulate pana acum.

Aplicatia proiectata si dezvoltata in cadrul acestui proiect este dedicata pasionatilor de gaming, dar poate fi folosita si de cei care doresc sa isi testeze abilitatile de concentrare pentru a gasi drumul corect pentru a ajunge la capatul labirintului.

Povestea incepe cu un personaj al carui tata a cazut sub vraja unei vrajitoare avand nevoie de apa cea magica pentru a-si pastra restul zilelor. Baiatul sau trebuie sa ajunga la templul din adancul padurii amazoniene unde se spune ca aceasta apa vitala se gaseste. Templul se afla in interiorul unui vulcan. Aventura personajului incepe din satul sau unde primeste de la fieraria din sat cea mai buna spada care se crede ca ar avea puteri supranaturale. Astfel va porni in cautarea templului prin jungla.

Dupa fiecare nivel complexitatea labirintului va creste. La nivelul 1 vor fi obstacole gen roci pe care le va putea evita prin saritura si maimute veninind pe liane pe care le va evita sarind sau lovindule cu sabia. La nivelul 2 se va apropria de vulcan iar noile capcane vor fi: roci vulcanice evitate prin saritura si portiuni de drum lipsa sau tepi pe care le va evita doar prin saritura sau alegerea unui alt traseu.

 Nivelul 1

 Nivelul 2

**Gameloop**

*while( user doesn't exit )*

*check for user input*

*run AI*

*move enemies (run physics)*

*resolve collisions*

*draw graphics*

*play sounds*

*end while*

**BIBLIOGRAFIE:**

* 1. [**www.e-scoala.ro/informatica/indexarhive.html**](http://www.e-scoala.ro/informatica/indexarhive.html)
  2. [**http://searchsoftwarequality.techtarget.com/definition/integrated-development-environment**](http://searchsoftwarequality.techtarget.com/definition/integrated-development-environment)
  3. [**www.animatic.ro/2011/05/unity-3d-un-tool-pentru-jocuri-3d-si-nu-numai/**](http://www.animatic.ro/2011/05/unity-3d-un-tool-pentru-jocuri-3d-si-nu-numai/)
  4. [**www.androidu.ro/despre-android-ce-este-android-totul-despre-android/**](http://www.androidu.ro/despre-android-ce-este-android-totul-despre-android/)
  5. [**www.link-academy.com/ce-inseamna-programare-java-la-ce-se-foloseste**](http://www.link-academy.com/ce-inseamna-programare-java-la-ce-se-foloseste)
  6. **https://ro.wikipedia.org/wiki/Blender**