CREACIÓ D'UN VIDEOJOC PER A MÒBILS MULTI-TÀCTILS

Miku, el videojoc

Albert Devesa Triscon
2n Batxillerat A
INS Narcís Monturiol
Tutor: Manel Niubó
31/1/2011

Pensa dues vegades abans de començar a programar,
o acabaràs programant dues vegades
abans de començar a pensar.

- Anònim

ÍNDEX

1. INTRODUCCIÓ 3

2. EVOLUCIÓ DE LES VIDEOCONSOLES 4

2.1 PRIMERA GENERACIÓ 4

2.2 SEGONA GENERACIÓ 5

2.3 TERCERA GENERACIÓ 6

2.4 QUARTA GENERACIÓ 6

2.5 CINQUENA GENERACIÓ 7

2.6 SISENA GENERACIÓ 8

2.7 SETENA GENERACIÓ 9

2.8 L'AUGE DE L'IPHONE I ANDROID 10

2.9 XIFRES 11

3. EINES PER CREAR UN JOC 12

3.1 CONEIXEMENTS PREVIS 12

3.2 UNITY 3D, EL MOTOR DEL JOC 12

3.3 DISSENY 3D 14

3.4 JAVASCRIPT COM A LLENGUATGE DE PROGRAMACIÓ 15

4. EL VIDEOJOC, PAS A PAS 17

4.1 DISSENY DELS NIVELLS 17

4.2 LA UNIÓ AMB UNITY 3D 20

4.3 CREACIÓ DEL PROTAGONISTA 22

4.4 PROGRAMACIÓ DELS SCRIPTS 25

5. EXPORTACIÓ 29

6. CONCLUSIONS 31

7. AGRAÏMENTS 32

8. BIBLIOGRAFIA 32

ANNEX A: CODI FONT DEL VIDEOJOC **33**

ANNEX B: TOTA LA INFORMACIÓ DELS NIVELLS **38**

ANNEX C: ESBOSSOS DELS ESCENARIS **42**

1. INTRODUCCIÓ

Fa més d'un lustre que els jocs mouen més diners que, per exemple, la indústria del cinema o de la música. L'allau dels nous videojocs (cada cop més realistes) i l'arribada de videoconsoles amb major nombre de funcionalitats, afegeixen l'entrada de nous segments socials que fins ara romanien com minories dins d'aquest mercat, i que ocupen el nostre temps d'oci i entreteniment.

Algunes d'aquestes videoconsoles han format part de la meva vida i, amb l'experiència adquirida com a jugador, m'agradaria saber com es fa un videojoc i portar-ho a la pràctica. He triat una plataforma mòbil tàctil perquè són les videoconsoles amb més potencial que hi ha actualment i amb les que més experiència he pogut aconseguir.

Començaré investigant com han evolucionat les consoles al llarg del temps i quines predominen actualment. Recercaré quines són les accions, mètodes i programes que han de portar a terme els programadors i dissenyadors per elaborar i publicar un videojoc.

Principalment, extrauré tota la informació d'Internet i solucionaré molts problemes gràcies a l'ajuda dels usuaris dels fòrums d'*Unity* 3D.

No disposo de cap coneixement sobre el disseny en 3D, així que suposarà un gran obstacle per culpa de la seva complexitat. Per l'altra banda, ja he tingut experiència en el camp de la programació i, conseqüentment, la seva interpretació em serà més fàcil i flexible.

Finalment, crearé un videojoc per a mòbils multi-tàctils, sobretot per l'iPhone i l'Android, que em permetrà estudiar la seva estructura i comparar-la amb els videojocs de gran escala.

1.1 OBJECTIUS

- ✓ Estudiar l'evolució de les consoles de videojoc al llarg del temps.
- ✓ Conèixer i aprendre a utilitzar les eines necessàries per a la creació d'un videojoc.
- ✓ Conèixer l'estructura d'un videojoc.
- ✓ Crear un videojoc en 3D.

2. EVOLUCIÓ DE LES VIDEOCONSOLES

Abans de començar s'ha de tenir clar què s'està tractant exactament. Una consola de videojoc o, simplement, videoconsola és un ordinador d'entreteniment interactiu controlat per un dispositiu d'entrada i que produeix una senyal de vídeo que pot ser utilitzada per un dispositiu de sortida (generalment és una televisió o un monitor d'ordinador). Aquests videojocs es presenten en cartutxos, xips, disquets, CD o DVD, tot i que actualment es comencen a distribuir per Internet.

En la industria dels videojocs, les videoconsoles han sigut classificades en generacions. La primera i la segona generació es classifiquen segons els recursos tecnològics en aquella època i la data de llançament. En canvi, de la segona a la sisena generació són classificades segons el nombre de bits[1] que posseeix la consola.

2.1 PRIMERA GENERACIÓ

Els primers jocs van aparèixer el 1950, però aquests utilitzaven pantalles vectorials, no de vídeo. Ralph Baer, el 1972, va llançar al mercat *Magnavox Odyssey*, la primera consola de sobretaula que es va començar a popularitzar gràcies al joc *Pong* de la companyia *Atari*. Aquest esdeveniment va començar a moure l'interès davant d'aquesta nova indústria. Després d'una actualització de la consola Odyssey, la cadena de centres comercials *Sears* va comprar els drets del sistema *Atari Pong* i els van introduir al mercat de consum baix amb el nom de *Sears-Telegames*. Però a causa dels clons de *Pong* i jocs derivats, aquesta compra no va donar el resultat esperat.

Fig. 1. Magnavox Odyssey

Fig. 2. Magnavox Odyssey 2000

Fig. 3. Sears-Telegames

Paral·lelament, *Mattel Auto Race* (1976) i *Electronic Quaterback* (1979) van ser les primeres videoconsoles electròniques portàtils que van sortir al mercat.

[1] El nombre de bits (**bi**nary digi**t**) és un dígit del sistema de numeració binari, 0 ó 1, que determinen l'amplada de bus del processador.

2.2 SEGONA GENERACIÓ

A l'octubre del 1977, l'*Atari 2600* va ser la primera consola llançada al mercat estatunidenc, convertint-se en la primera consola en tenir èxit utilitzant cartutxos intercanviables. *Atari 5200* va ser la seva successora però va acabar adoptant el nom de *CX2600*, el qual es basava en el nombre de catàleg que posseïa. Aquesta consola va ser un gran èxit i *Atari* es va convertir en sinònim de videojocs. La consola venia acompanyada amb dos joysticks[2], un trackball[3] i un cartutx de joc (inicialment *Combat* i subsegüentment *Pac-Man*).

Atari va aconseguir el monopoli, tot i que va tenir dos rivals destacables: *ColecoVision*, amb el doble de colors i *Mattel Intelvission*, que per primera vegada a la història, incloïa una CPU de 16 bits. *Nintendo* i *Sega*, una de les millors companyies de jocs avui en dia, amb *TV-Game 6* i *SG-1000* respectivament, començaven a fer els primers passos a nivell domèstic.

El 1979 va aparèixer la primera videoconsola portàtil, la *MB Microvision*, amb cartutxos intercanviables. En canvi, el 1980, *Nintendo* va treure *Game & Watch*, una consola electrònica portàtil amb 59 jocs per defecte i que no admetia cartutxos.

Fig. 4. Atari 2600

Fig. 5. Atari 5200 (CX2600)

Fig. 6. ColecoVision

Fig. 7. SG-1000

Fig. 8. TV-Game 6

Fig. 9. Game & Watch

[2] Un *joystick* o palanca de control és un dispositiu de control de dos o tres eixos que s'usa a ordinadors, videoconsoles, transbordadors espacial i avions de caça.
[3] Un *trackball* és un dispositiu apuntador compost per una bola incrustada en un receptacle que conté sensors que detecten la rotació de la bola en dos eixos.

2.3 TERCERA GENERACIÓ

A partir de la tercera generació, el món de les consoles va començar a ser abolit pel monopoli japonès. Les consoles d'aquesta generació com *NES* i *Sega Master System* tenien 8 bits. Fins el 1988, *NES*, juntament amb el joc *Super Mario Bros*, dominaven la indústria. *Atari* amb l'esperança de recuperar el domini va treure l'*Atari 7800*, una versió millorada de l'*Atari 5200*, però no ho va aconseguir.

Fig. 40. NES

Fig. 11. Sega Master System

Fig. 12. Atari 7800

2.4 QUARTA GENERACIÓ

El 1983, Nec i Hydon van treure a la llum la consola *PCEngine* (al Japó) o *TurboGrafx* (a la resta del món) que també tenia una CPU de 8 bits però amb un xip gràfic de 16 bits. La consola més rellevant d'aquesta generació era *Sega Mega Drive*, coneguda comercialment com *Sega Genesis*, introduïda el 1988 tot i que *Nintendo* va treure la seva consola de 16 bits anomenada *Super Nintendo* el mateix any. *Neo Geo* va ser la consola més potent d'aquesta generació però era molt cara.

Els xips gràfics afegits als cartutxos destacaven en aquesta generació. Apareixen també conceptes com la multitasca, multimèdia, gràfics vectorials, etc. *Super Nintendo* va ser la consola més venuda amb 49 milions d'unitats al Japó.

Fig. 13. PCEngine/TurboGrafx-16

Fig. 14. Sega Mega Drive

Fig. 15. Super Nintedo

Fig. 16. Game Boy

Game Boy de *Nintendo* va ser i encara és una de les consoles portàtils més venudes de la història gràcies a l'arribada del joc *Tetris* i el grandíssim suport de les companyies que treballaven per *NES*. El 1989, per competir amb la *Game Boy*, *Sega* creà la *Game Gear*, la tercera videoconsola portàtil amb color.

2.5 CINQUENA GENERACIÓ

Entorn el 1994, es va deixar d'utilitzar el 2D al posar-se de moda els entorns tridimensionals 3D, aprofitant al màxim el hardware dels equips. Aquesta generació és coneguda com l'era dels 32 bits, tot i que es confon amb la de 64 bits pel nom que li van donar a la consola *Nintendo 64*. També es van enunciar dues grans consoles, la *PlayStation* de *Sony* i la *Sega Saturn*. La primera va tenir un gran èxit en el món dels videojocs que encara perdura.

Bàsicament el mercat estava dominat per tres consoles, *Sega Saturn*, *PlayStation* i *Nintendo*. Aquesta generació també va veure dues versions actualitzades de la *Game Boy*: la *Game Boy Color* i la *Game Boy Light* (aquesta última només estava disponible al Japó). *Apple* també va voler entrar al mercat amb el seu producte *Apple Pippin*, però no va tenir èxit.

Fig. 17. PlayStation

Fig. 18. Sega Saturn

Fig. 19. Apple Pippin

2.6 SISENA GENERACIÓ

Les consoles d'aquesta generació es milloraren gràcies a la introducció d'arquitectures d'ordinadors. Les consoles de sobretaula van substituir els cartutxos per grans capacitats com ara el *DVD*, *GD-ROM*[4] o el *GOD*[5], així els jocs eren més llargs i més atractius visualment. Les noves memòries flash i els disc durs implantats en aquesta generació eren utilitzats per guardar les partides.

Sega va treure la seva nova i última consola anomenada *Dreamcast*, en canvi, *PlayStation 2* de *Sony*, la primera en introduir el reproductor DVD, va continuar tenint el mateix èxit que la *PlayStation*. Totes les consoles *PlayStation*, al llarg de la història, han tingut la seva versió *Slim* pertinent. D'altra banda, la *GameCube* va ser la quarta videoconsola de sobretaula de *Nintendo*. Finalment, l'*Xbox* va ser la última consola en sortir al mercat d'aquesta generació i la primera de *Microsoft*. També utilitza un reproductor *DVD*, amb connexió online i un disc dur integrat.

Les videoconsoles portàtils també anaven evolucionant i *Nintendo* començà a fabricar la successora de la *Game Boy Color*, la *Game Boy Advanced*, amb un processador *ARM*[6] propi de 32 bits. A més, era capaç d'executar els jocs compatibles amb Game Boy (quarta generació) i la *Game Boy Color* (cinquena generació). El 2003, *Nokia*, una de les principals empreses del sector de les telecomunicacions, va treure l'*N-Gage*, el primer telèfon mòbil i consola portàtil a la vegada.

Fig. 20. PlayStation 2

Fig. 21. Xbox

Fig. 22. GameCube

[4] Aquest nou suport òptic va ser implantat per Sega amb l'objectiu de parar la pirateria de software, que era relativament fàcil en les consoles de la generació anterior ja que va coincidir amb la sortida al mercat de les primeres gravadores de CD-ROM. Tot i així aquest nou sistema va ser violat poc temps després del seu llançament.
[5] GOD (GameCube Optical Disc) va ser el format de distribució dels videojocs de Nintendo. Actualment s'empra la Wii Optical Disc per la consola Wii.
[6] ARM (Advanced RISC Machines) són microprocessadors RISC dissenyats per l'empresa Acorn Computers i desenvolupats per Advanced RISC Machines Ltd., una empresa derivada de l'anterior.

2.7 SETENA GENERACIÓ

Aquesta actual generació es caracteritza per la introducció de la tecnologia multi nucli[7] de la CPU. Durant el 2005, *Microsoft* va presentar la nova *Xbox 360*, mentre que el 2006, la *PlayStation 3* va ser presentada per *Sony*. En aquest mateix any, *Nintendo*, amb el desig de recuperar el mercat, presentà la seva nova i actual consola *Wii*, caracteritzada per l'innovador comandament amb la capacitat d'apuntar gràcies a la detecció de l'acceleració dels moviments en tres dimensions. Posteriorment, les versions Slim de la *PlayStation 3* i la *Xbox 360* van ser presentades al llarg del 2009 i el 2010.

Fig. 23. PlayStation 3

Fig. 24. Xbox 360

Fig. 25. Wii

Des del 2005, *Nintendo* i *Sony* han estat competint en el sector de les videoconsoles portàtils. *PlayStation Portable*, o simplement *PSP*, tenia més potència gràfica que la *Nintendo DS* (sigles de *Dual Screen*), però aquesta la superava en ventes. Al llarg del temps s'han anat millorant les consoles. La *PSP* ha tingut fins a quatre versions més, amb noves millores. La *Nintendo DS* també ha tingut el mateix nombre de versions i la última, la quarta, amb tecnologia 3D (efecte tridimensional de la seva pantalla), serà l'encarregada d'estrenar la vuitena generació de consoles quan es llenci al febrer del 2011.

Fig. 26. PSP Go!

Fig. 27. Nintendo 3DS

[7] Combina dos o més processadors independents en el mateix circuit integrat, i busca aconseguir una multitasca real més eficient.

2.8 L'AUGE DE L'IPHONE I ANDROID

Cada cop hi ha més demanda en el sector dels videojocs per mòbils d'última generació. Aquests mòbils, anomenats smartphones[8], cada cop tenen més capacitat per executar jocs com ho faria una de les videoconsoles portàtils esmentades abans.

A finals de 2007, *Apple* va introduir el seu primer smartphone, l'*iPhone*. Fou un dels primers en utilitzar una interfície multi-tàctil[9]. El juliol de 2008, *Apple* va llençar un nou model amb un preu menor i suport 3G. A partir de la versió 2.0 del seu sistema operatiu, es creà l'*App Store* amb aplicacions tant gratuïtes com de pagament, permetent instal·lar aplicacions desenvolupades per tercers. Aquesta botiga d'aplicacions ha tingut un gran èxit arribant fins a les 250.000 aplicacions.

Fig. 28. iPhone 4 executant l'App Store

L'èxit de l'*iPhone* ha refermat certes tendències dins del món dels smartphones com ara l'ús d'interfícies tàctils, l'abandonament dels teclats o la importància de les botigues d'aplicacions.

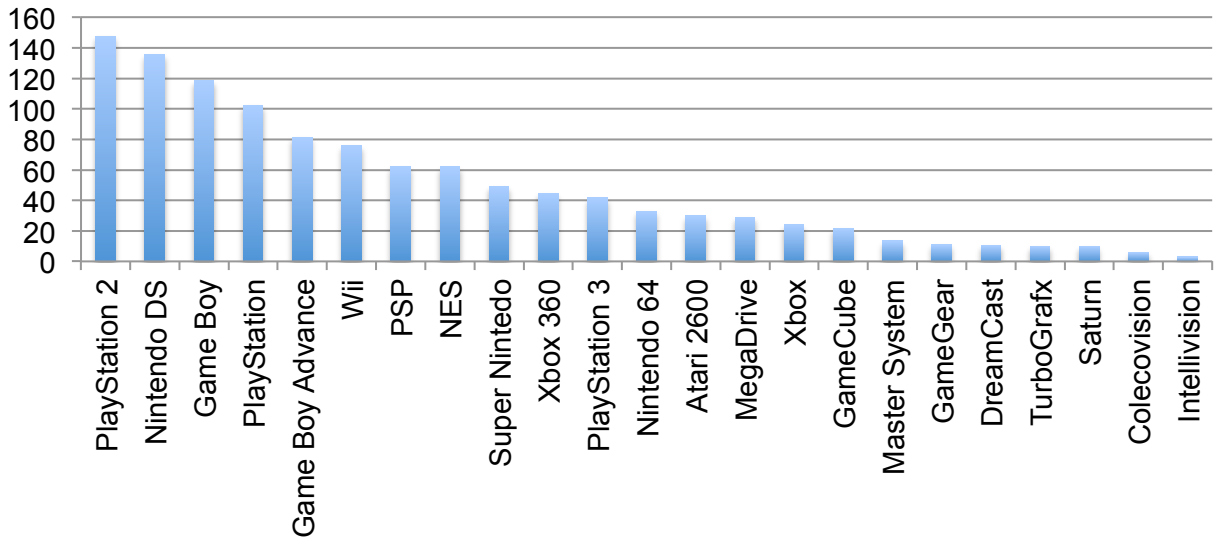
El sistema operatiu *Android*, desenvolupat per *Google*, va aparèixer per a smartphones l'any 2008 per contraatacar l'*iPhone* d'*Apple*. Es tracta d'un sistema operatiu de codi obert basat en Linux i suportat per la Open Handset Alliance (que a part de Google està formada per *Intel*, *HTC*, *Motorola* i *Samsung* entre d'altres). El primer smartphone en utilitzar-l'ho va ser l'*HTC Dream*, anomenat G1 per l'operadora americana *T-Mobile* que el distribuí. El programari inclòs en el sistema operatiu integra els serveis propis de *Google* com ara correu, calendari, navegació, etc. Les aplicacions de tercers estan disponibles a l'*Android Market*, el principal competidor de l'*App Store*.

Fig. 29. HTC Desire amb Android 2.2

[8] Un smartphone, a cops anomenat telèfon intel·ligent, és un ordinador de butxaca que integra les funcions de telèfon mòbil, organitzador personal i sovint altres funcions de connectivitat mòbil.
[9] El multi-tàctil (multi-touch) es composa d'una pantalla tàctil o d'un teclat digital tàctil que reconeix punts de contacte simultanis i múltiples així com d'un programari que interpreta aquests contactes simultanis.

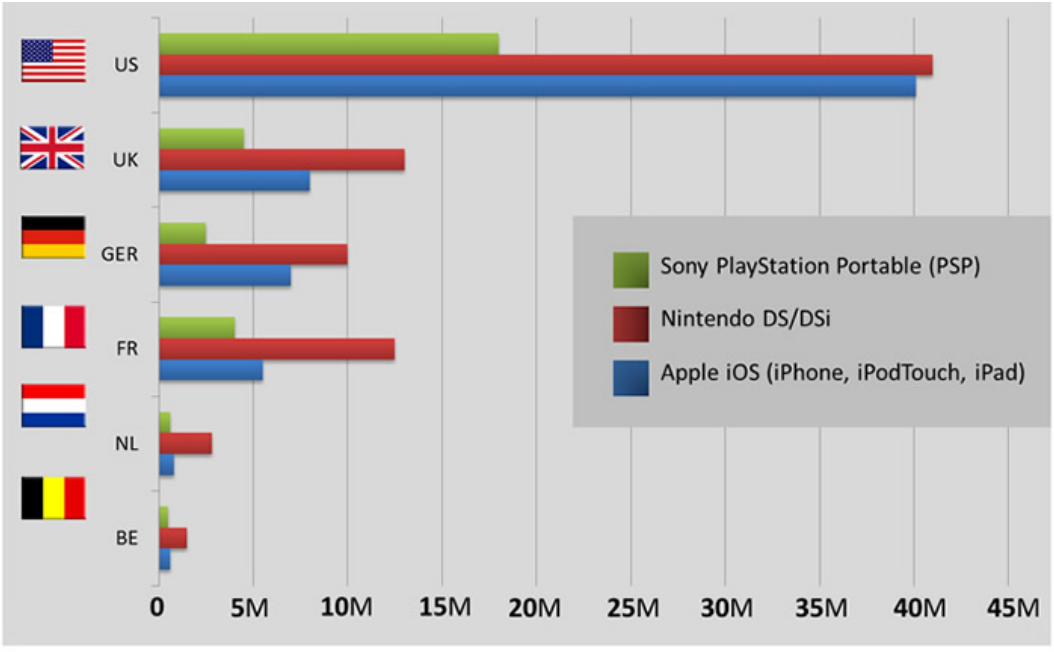
2.9 XIFRES

Aquest apartat va destinat a les dades obtingudes del món dels videojocs i les consoles al llarg del temps fins a l'actualitat. La següent estadistíca mostra les consoles més venudes (unitats en milions) durant els últims 25 anys (actualitzat el mes gener del 2011):



La indústria dels videojocs és un mercat que mou més diners que la indústria del cinema i i va creixent any rere any. Per aquest motiu, multinacionals com *Sony* o *Microsoft* han invertit milers de milions de dòlars en el desenvolupament de les seves noves consoles, conscients que si les coses els surten bé s'amortitzaran els diners gastats.

El següent gràfic, realitzat per *Newzoo* © durant el 2010, mostra el nombre de persones entre 10 anys i superior que juga en unes específiques consoles portàtils. Des de l'arribada de l'*App Store*, el nombre de jocs destinats a iOS no ha parat de crèixer i el 22 de gener del 2011 va arribar a les deu mil milions de baixades.



3. EINES PER CREAR UN JOC

Són moltes les eines que es poden utilitzar per l'elaboració d'un joc. Entre aquestes, he utilitzat les eines més flexibles i fàcils d'utilitzar, sobretot en l'àmbit 3D. També he intentat crear un joc fàcil, simple i apte per a tothom. De les característiques que m'ofereix l'exportació d'Unity 3D, m'he centrat en la plataforma mòbil.

3.1 CONEIXEMENTS PREVIS

A banda de tenir coneixements de programació, s'ha de tenir agilitat per moure's entre programes, adaptar-se al que se't demana i saber els límits d'aquest. S'ha de tenir una ment clara a l'hora d'escriure programació, sinó et guanyes problemes i mal de caps innecessaris.

Per crear un joc en la plataforma d'iOS[10], Apple ofereix el seu programa XCode, el qual es basa amb Objective-C, llenguatge de programació amb llicència GPL pel compilador GCC. Aquest llenguatge deriva del llenguatge C, orientat a objectes. En el meu cas he triat el programa Unity per la simplicitat que em dóna. Unity s'ocupa de tots els aspectes que requereix la creació d'un joc, àdhuc transforma el llenguatge JavaScript, C# o Boo a Objective-C per ser compilat i executat a l'iOS. També s'ocupa de l'apartat de la física, tot i que es necessita un mínim de coneixements per resoldre problemes que puguin sorgir durant el desenvolupament del joc.

3.2 UNITY 3D, EL MOTOR DEL JOC

Unity 3D és el programa per crear els jocs 3D de vídeo o un altre contingut interactiu com visualitzacions arquitectòniques o animacions 3D en temps real. Integra quasi totes les necessitats indispensables per crear el videojoc.

Funciona tant en Windows com en Mac OS X, i pot produir jocs per a diferents plataformes: Windows, Mac, Wii, iPad, iPhone i, ara també, per PlayStation 3 i Xbox 360. També pot importar aquests jocs a navegadors gràcies al seu plug-in[11], compatible amb Internet Explorer, Mozilla Firefox, Safari, Netscape, Opera, Google Chrome i Camino. S'ha de tenir en compte que un joc creat a Unity 3D no sempre s'executarà bé a totes les plataformes. Per exemple, un joc creat per PlayStation 3 pot ser exportat a l'iPhone però no ho farà amb la mateixa fluïdesa que a la PlayStation 3, ja que aquesta posseeix un hardware més potent, igual que l'Xbox 360.

[10] iOS, antigament conegut com iPhoneOS, és el sistema operatiu de l'iPhone, l'iPod Touch i l'iPad d'Apple.
[11] Un plug-in, també conegut com a extensió o addon, és una aplicació informàtica que interactua amb una altra aplicació per aportar-li una funció o utilitat addicional.

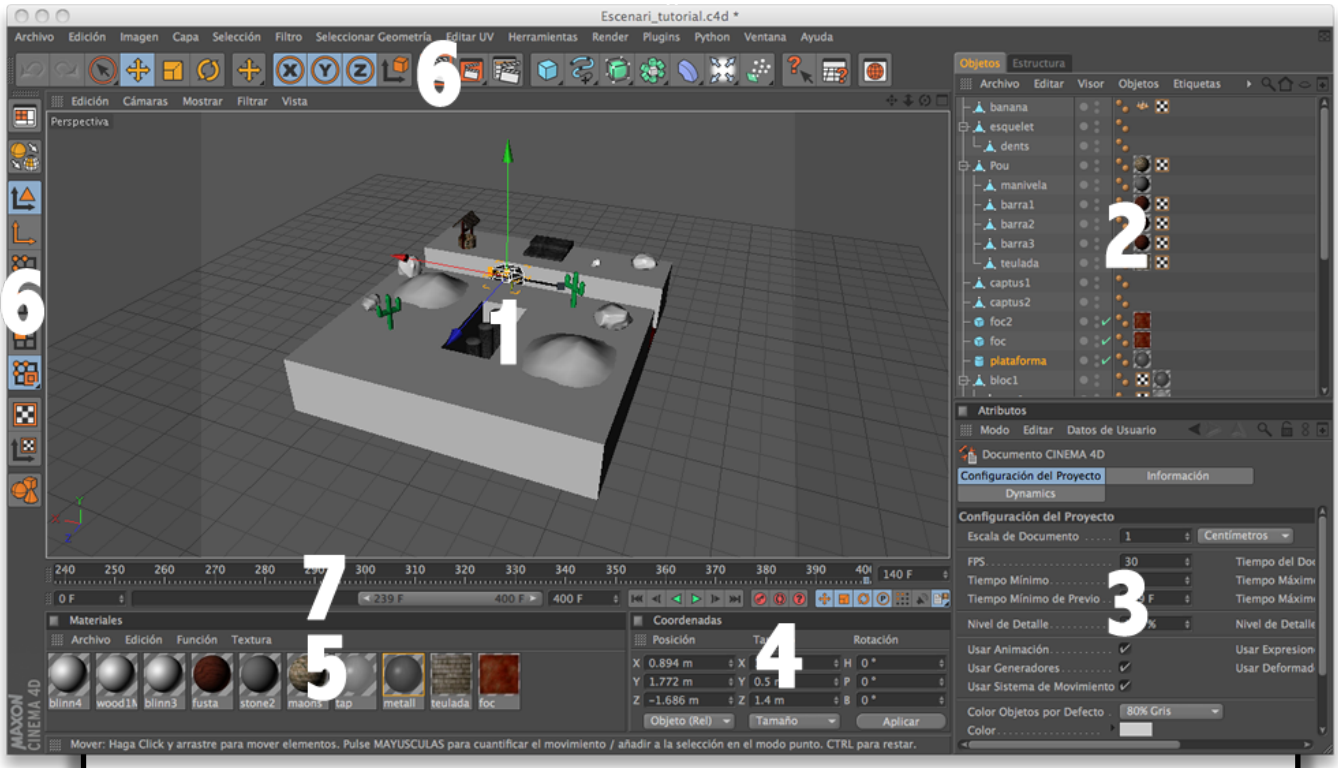
1. **Escenari:** finestra on es visualitza l'escenari del joc en 3D. Des d'aquesta finestra es pot editar, moure i rotar qualsevol objecte.
2. **Projecte:** llistat de tots els fitxers que hi ha en tot el joc.
3. **Jerarquia:** fitxers que només correspon al nivell.
4. **Inspector:** propietats de cada objecte del joc.
5. **Joc:** com es veu el joc realment un cop sigui exportat a la plataforma. També informa dels recursos que s'empren en aquell precís moment.

Miku es va començar a desenvolupar amb Unity iPhone 1.7, ja que Unity estava separat per plataformes. Amb la nova versió 3, s'han fusionat totes les plataformes per formar Unity 3D. Aquest és capaç d'importar models 3D creats per 3ds Max, Maya, Blender, Cheetah3D i Cinema 4D. Utilitza diferents motors gràfics, Direct3D en el cas de Windows, OpenGL a Mac i també a Windows, OpenGL ES únicament per iOS i APIs[12] propietàries com ara Wii.

[12] Una API (Interfície de Programació d'Aplicacions) és el conjunt de funcions i procediments que ofereix certes biblioteques per a ser utilitzat per un altre programari com una capa d'abstracció.

3.3 DISSENY 3D

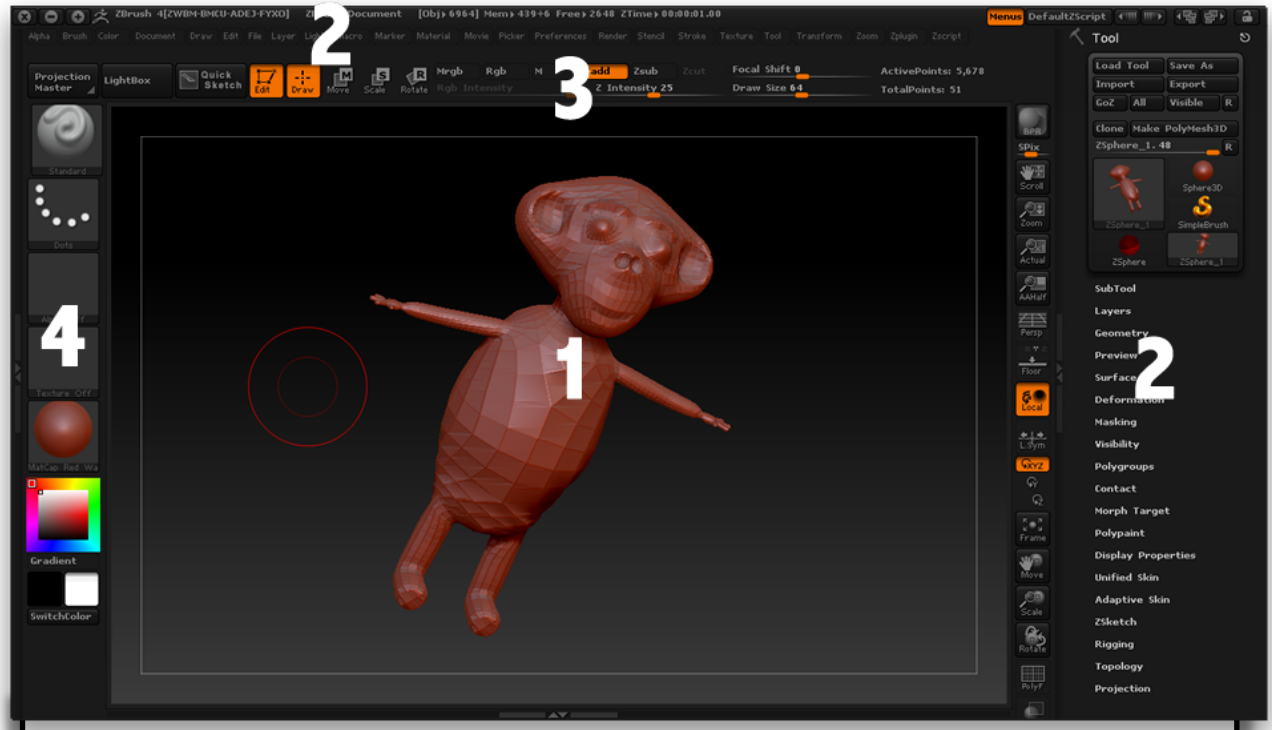
Per fer tot el disseny 3D del joc, s'han utilitzat dos programes: Cinema 4D i ZBrush. El primer és un programa de creació de gràfics i animació 3D desenvolupat per *Commodore Amiga* per la companyia alemanya *Maxon*, i portat posteriorment a les plataformes Windows, Linux i Macintosh. Permet el modelatge de primitives, splines[13] i polígons, texturització i animació. Aquest programa s'encarregarà dels escenaris 3D.



1. **Escenari:** finestra on es visualitza els objectes en 3D. Des d'aquesta finestra es pot editar, moure i rotar qualsevol objecte.
2. **Objectes:** llistat de tots els objectes 3D.
3. **Atributs:** finestra on es mostren les propietats i atributs dels objectes.
4. **Coordenades:** posició, mida i rotació de l'objecte seleccionat.
5. **Materials:** llista de les textures dels objectes.
6. **Eines:** totes les eines que ofereix Cinema 4D per editar els objectes. També es poden seleccionar des del menú de dalt de tot.
7. **Animació:** permet animar els objectes i visualitzar el seu comportament a cada fotograma.

[13] Un spline és una corba definida en porcions mitjançant polinomis.

En canvi, ZBrush em permet crear el protagonista del joc gràcies a l'original plantejament del seu procés creatiu per a personatges. L'utilitzen companyies de gran escala com ara *Electronic Arts* o també per a pel·lícules, com *El senyor dels Anells*.



1. **Escenari:** finestra on es visualitza l'objecte o personatge que s'estigui creant.
2. **Eines:** totes les eines que ofereix el programa. A la part de la dreta podem trobar les eines més utilitzades i a la part superior les podem trobar totes.
3. **Opcions:** podem regular l'eina seleccionada quan és utilitzada.
4. **Atributs:** podem seleccionar els materials, els seus colors, els punters i els efectes especials per aplicar-los al personatge o objecte.

3.4 JAVASCRIPT COM A LLENGUATGE DE PROGRAMACIÓ

Com he expressat abans, el joc està realitzat amb el llenguatge JavaScript, llenguatge script basat en el concepte de prototip, implementat originàriament per *Netscape Communications Corporation*, i que va derivar en l'estàndard *ECMAScript*. És conegut sobretot pel seu ús en pàgines web, però també s'utilitza en altres aplicacions. Malgrat el seu nom, JavaScript no deriva del llenguatge de programació Java, però tots dos comparteixen una sintaxi similar inspirada en el llenguatge C. El nom *JavaScript* és una marca registrada per *Sun Microsystems*.

D'entre tots els programes, he triat *Unitron* per la seva senzillesa i la seva integració amb Unity.

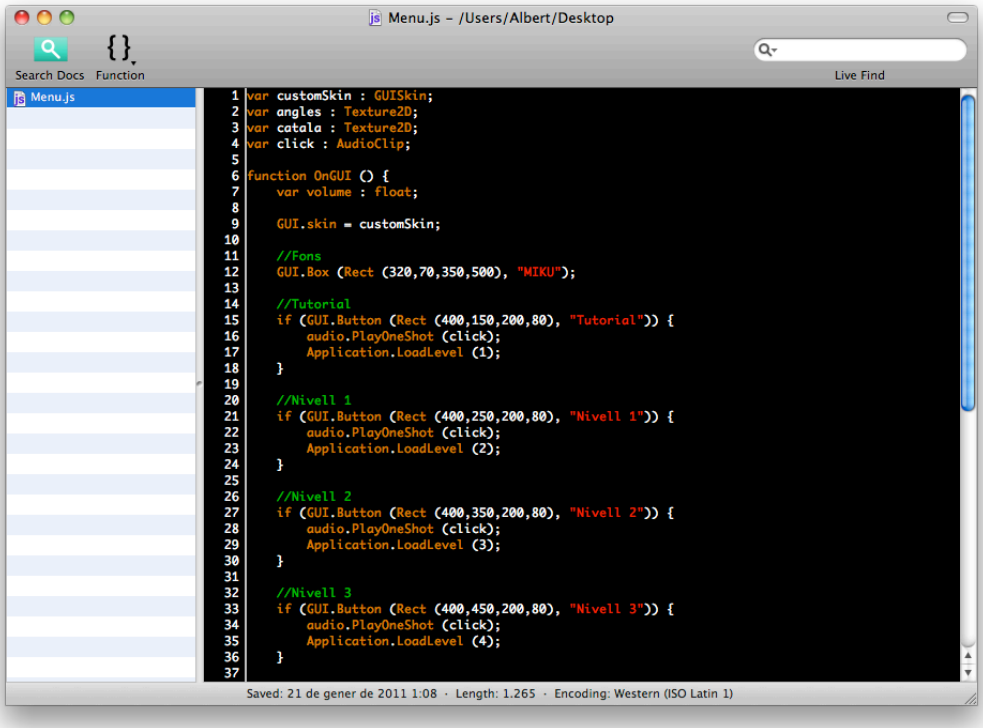


Fig. 30. Captura d'Unitron

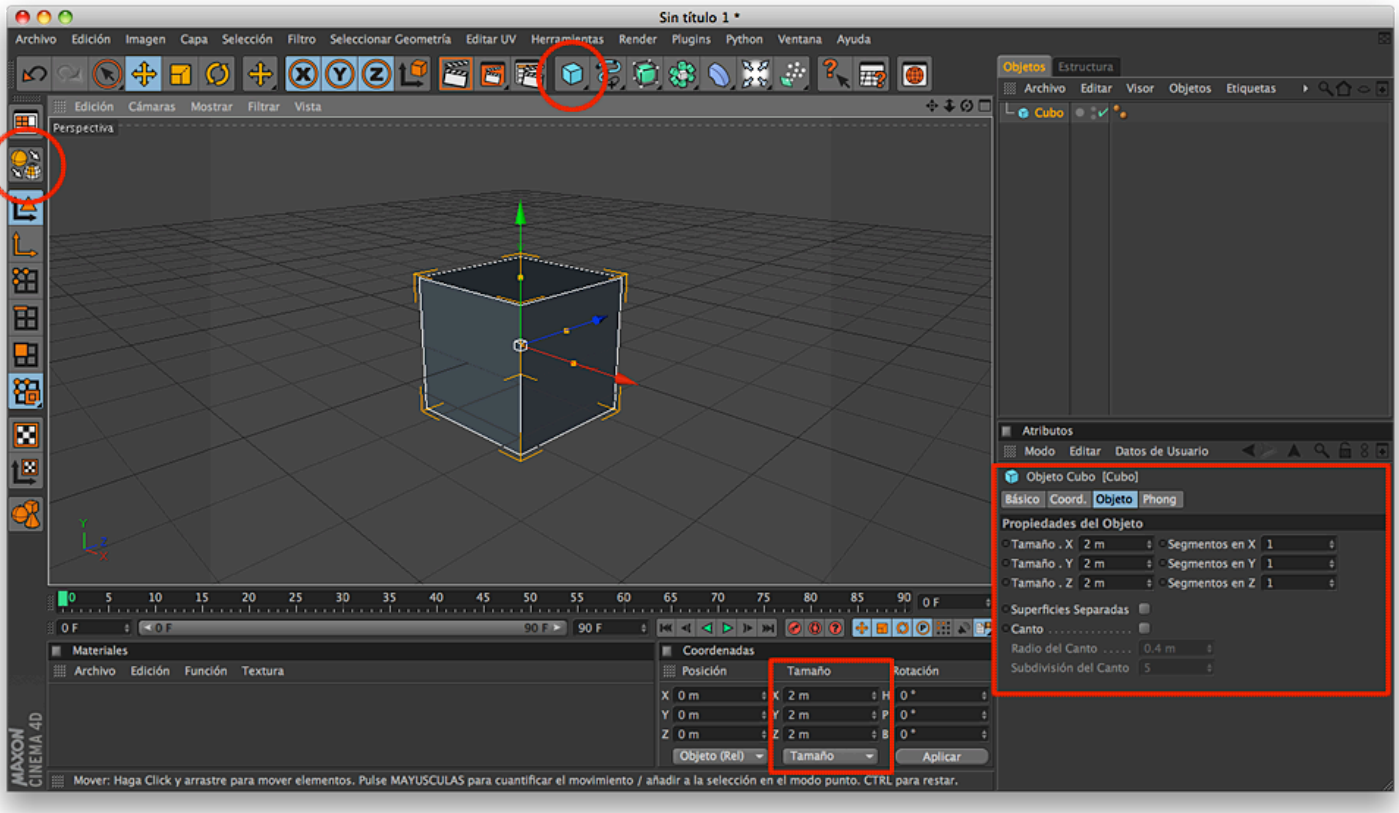
4. EL VIDEOJOC, PAS A PAS

En els següents apartats aniré explicant els conceptes fonamentals per a la realització del joc i els passos que he anat seguint fins a la seva publicació. La millor manera de dissenyar i programar un joc és planificar des del principi tota la mecànica, els objectius i les funcionalitats, sinó seria molt difícil que fos acabat amb èxit.

4.1 DISSENY DELS NIVELLS

Després d'haver plantejat i dibuixat a mà el nivell en una plantilla[14] podem procedir a la seva creació en 3D amb el programa esmentat a l'apartat anterior: Cinema 4D.

Primer de tot creem un objecte preparat per Cinema 4D que serà la base, en aquest cas un quadrat. El modifiquem i li canviem les mides. Especifiquem el nombre de segments i el fem editable. Com més segments tingui més petits seran els talls i més precisió tindrem, però el joc tardarà més en processar. En la següent captura de pantalla he ressaltat els llocs que hem hagut d'anar prement.



[14] Totes les plantilles amb els nivells dibuixats del joc es poden veure a l'Annex C.

Tot seguit comencem a crear els obstacles que prèviament hem dibuixat [15]. El primer obstacle pel jugador serà saltar en tres blocs sense caure al foc. Per a la seva creació seleccionem el nombre de quadrats i extrudim el terreny amb l'eina *Extrude*. Després creem un cilindre, el redimensionem, el posicionem i el fem editable. Per estalviar-nos fer la mateixa feina, dupliquem dues vegades el mateix cilindre i els alineem (hem de tenir en compte la potència de salt del personatge).

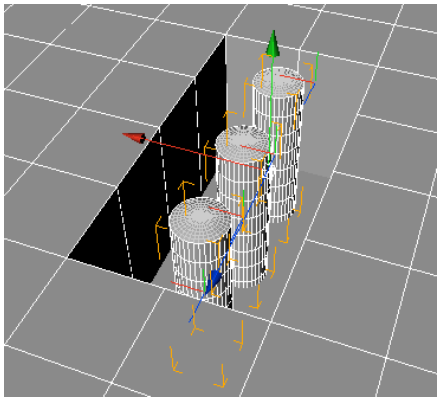


Fig. 31. Creació de tres blocs

El proper obstacle serà saltar sobre una plataforma mòbil sense caure al foc. En aquest cas triem l'opció de crear un *tanc* i li canviem el nombre de segments de rotació per tal de convertir-lo en un hexaedre. Un cop redimensionat i situat a la seva posició, l'animem. Ens col·loquem al primer fotograma i li afegim un fotograma clau per guardar la seva posició inicial. Movent només l'eix Z (o l'eix Y depenent de com estigui orientat l'escenari), el desplacem uns quants metres a l'esquerra i ens situem al fotograma 200 (el nombre de fotogrames és proporcional al temps que triga en completar el moviment). Guardem la posició afegint un fotograma clau, ens situem al fotograma 400 i tornem a posicionar l'hexaedre a la posició inicial. En el moment de prémer el fotograma clau, Cinema 4D ens crearà una línia que segueix la trajectòria de la plataforma (fig. 32).

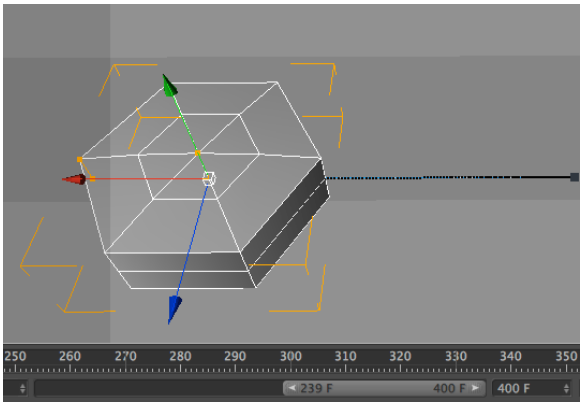


Fig. 32. Creant animacions

Seguint els passos anteriors ja tindrem fets dos obstacles. Com que aquest escenari es tracta del *Nivell 1* cal que no sigui gaire complicat perquè hem d'acostumar al jugador els moviments i límits del personatge. En el cas d'aquest escenari he triat un ambient *Wild West* i per simular-ho he afegit dos cactus, un pou, unes quantes pedres i dos turons.

Un cop creat tots els objectes, hem d'afegir les textures. Aquestes poden ser extretes d'Internet perquè portaria molta feina i esforç fer cada fotografia de cada component en concret. De fet, existeixen programes dedicats a la realització i editatge de textures.

[15] L'explicació pas a pas del joc es basa en el primer nivell. Imatges dels escenaris en 3D dels altres nivells es poden veure a l'Annex B.

Per aplicar les textures als objectes, creem un material nou, li afegim la fotografia determinada per a cada objecte i després només farà falta que l'arrosseguem sobre l'objecte desitjat. Si volem editar el material només haurem d'anar a l'editor de materials i allà tindrem una llarga llista d'opcions per a modificar (fig. 33). Com que de vegades no disposem d'una fotografia amb prou resolució, a l'hora d'aplicar-la sobre l'objecte no quedarà bé. Per solucionar-ho només caldrà que li afegim repeticions i el mateix programa s'encarregà de dividir la textura.

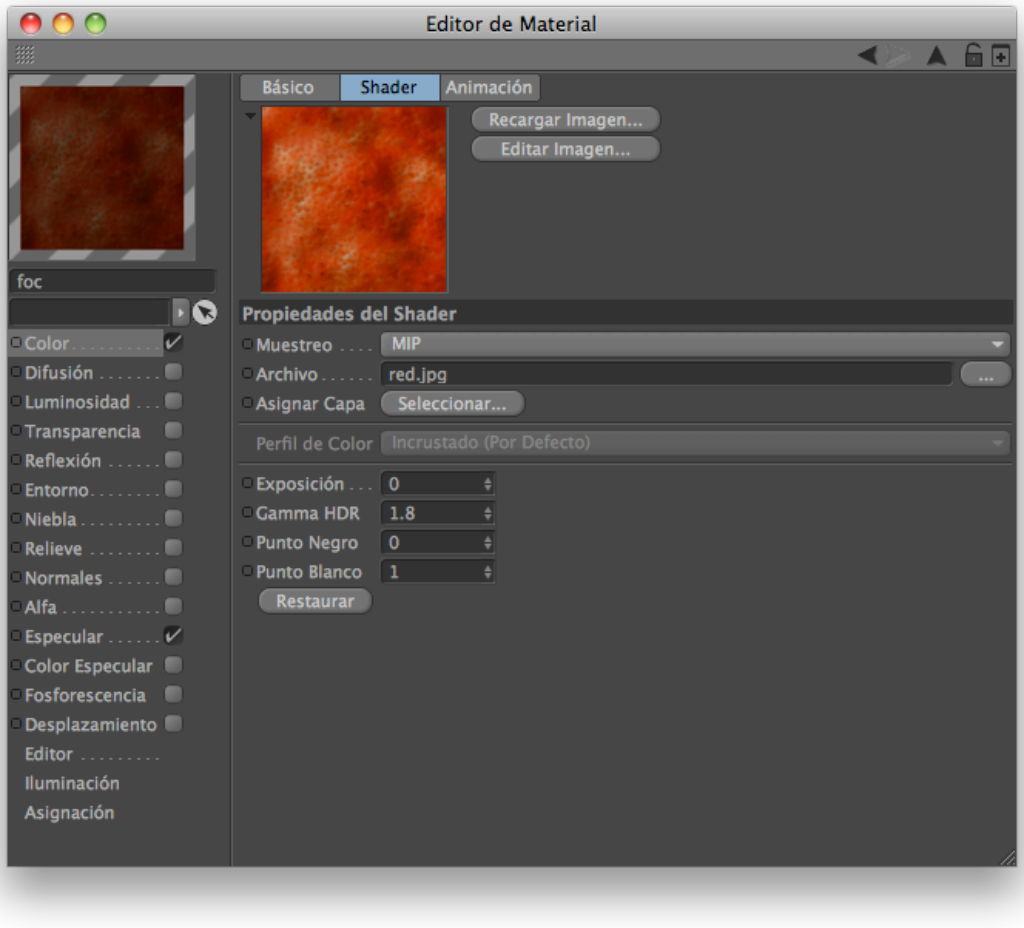


Fig. 33. Editor de materials

Un cop tinguem l'escenari 3D complet només l'haurem d'exportar. Veurem que hi ha diferents extensions d'exportació. Les més utilitzades són *.obj* (la més internacional) i *.fbx*. En aquest cas he triat la segona ja que em permet exportar les animacions. També seran exportades les textures de cada material.

Un cop tinguem l'escenari exportat només l'hem d'arrossegar dins del llistat dels arxius a Unity o desar-lo dins la carpeta del joc.

4.2 LA UNIÓ AMB UNITY 3D

Un cop a Unity, quan seleccionem l'escenari en 3D que hem realitzat amb Cinema 4D, se'ns mostrarà uns seguit d'opcions a la finestra de l'Inspector. Allà podem configurar la nostra configuració més adient pel nostre nivell. Les opcions seleccionades seran (fig. 34): *Scale Factor* a 0.1, redimensiona l'escenari; *Generate Colliders* seleccionat, genera col·lisionadors als objectes perquè el personatge pugui xocar contra ells; Animation Wrap Mode a Loop, així la plataforma mòbil segueix la trajectòria infinitament fins que es completi el nivell sinó només faria una trajectòria. Totes les altres opcions les deixem per defecte.

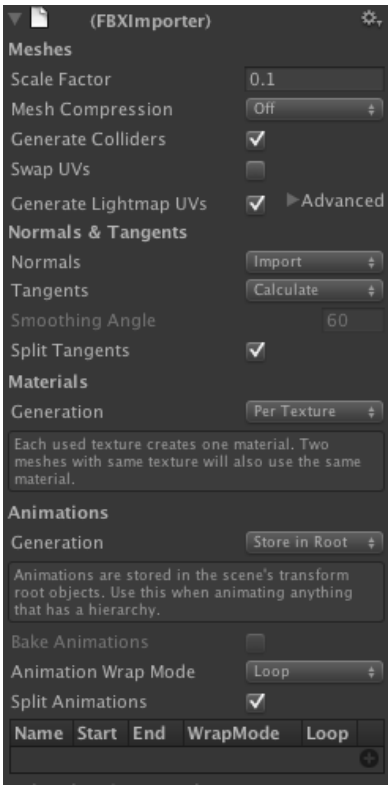


Fig. 34. Inspector

Per controlar el personatge, Unity ens ofereix un seguit de controls ja configurats i adaptats per l'iOS. Cada plataforma té les seves configuracions més adients. Pel primer nivell he triat la configuració més clàssica per un dispositiu mòbil tàctil, moviment en primera persona mitjançant dos *joysticks* virtuals. Només cal obrir l'arxiu *FirstPersonSetup* [16] i automàticament se'ns crearà un nou escenari amb els arxius necessaris.

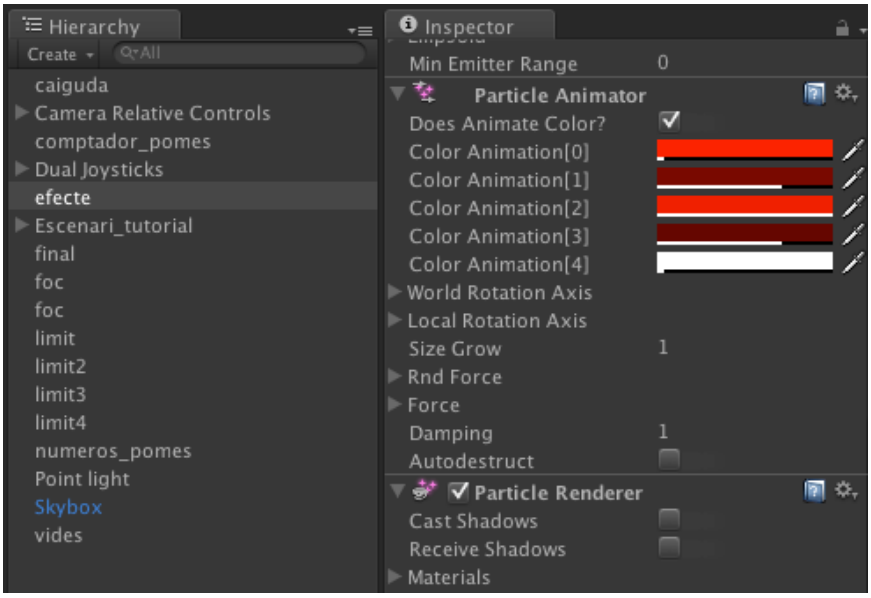


Fig. 35. Tots els fitxers que corresponen al *Nivell 1*

[16] El codi de programació de tot el joc està disponible a l'Annex A.

A partir d'ara només cal afegir tots els objectes necessaris:

- *Skybox*: és un quadre que simula un cel, ja que les seves parets són imatges d'aquest.
- *Point Light*: es tracta d'un punt de llum que pots regular la seva intensitat, el color, la resolució... i, conseqüentment, provoca les ombres dels objectes.
- *Particle System*: efecte especial que, amb una llarga llista d'opcions, pots donar-li infinitats de formes com ara la simulació del foc.
- *GUI Texture (Graphic User Interface)*: elements gràfics que permeten interactuar de forma molt més intuïtiva i són impresos a la pantalla per sobre del joc. També és possible fer el mateix amb textos (*GUI Text*).

Tots els fitxers que són importats i reconeguts a Unity tenen propietats que poden ser canviades a l'Inspector. Per exemple, un cop el personatge és importat dintre Unity ens permet dividir les seves animacions per *frames* (fotogrames). Simplement seleccionem el personatge i, dins l'inspector, anem a la secció *Animations*. Allà separem per *frames* les animacions que hem aplicat al personatge (fig. 36). Des del fotograma 2 al 60 hem definit l'animació *walk* mentre que *idle* serà l'animació quan no estigui en moviment. La creació de les animacions està explicat en el següent apartat.

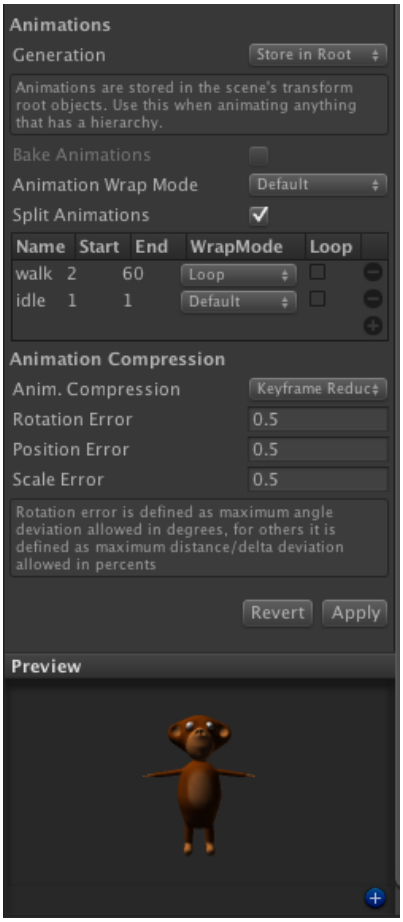


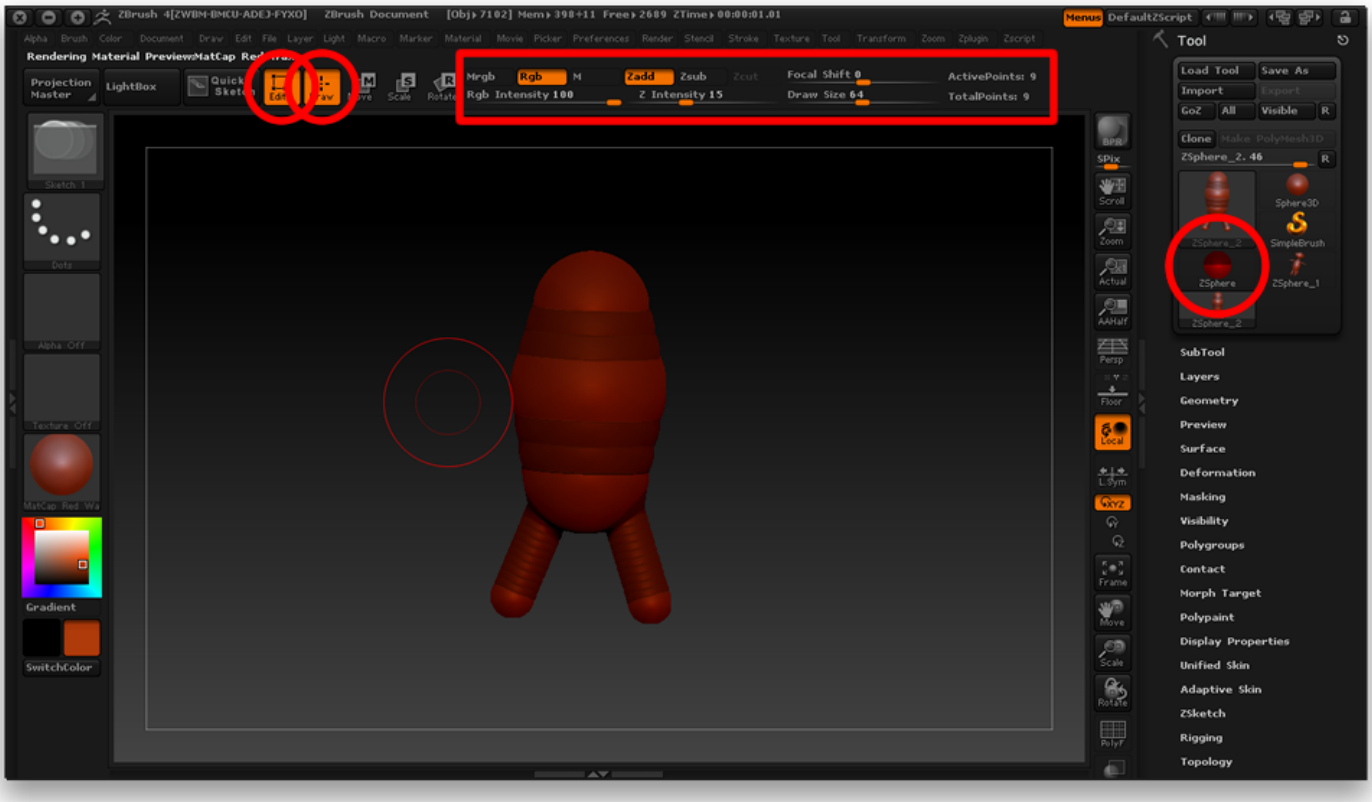
Fig. 36. Animacions del personatge

Tot i així, no n'hi haurà prou per tenir un personatge totalment operatiu per jugar, faltarà l'ús de la programació per indicar-li al programa quan utilitzar aquestes animacions. Podem trobar les instruccions d'aquest procediment a l'apartat de programació dels scripts.

4.3 CREACIÓ DEL PROTAGONISTA

Un cop realitzat i situat l'escenari on es mourà el personatge, és hora de fer el protagonista del joc. Jo he decidit fer el personatge abans de programar el joc, però es pot fer al revés ja que el resultat serà el mateix.

Primer de tot elegim l'eina *ZSphere* prement sobre *Tool*. Creem una bola de diàmetre considerable en el mode *Draw* (en el cas que no el tinguéssim seleccionat) i a continuació premem *Edit* perquè sigui editable. Un cop fet això, només cal anar muntant boles sobre altres boles més petites o més grans depenent de la secció del cos. La captura següent marca les parts on hem hagut de clicar.



Un cop tinguem la figura feta, premem la tecla A i el mateix programa convertirà la figura a una figura geomètrica (fig. 37). En el cas que volguéssim afegir algun detall com ara els orificis nassals o de les orelles, cal desmarcar el mode de dibuix, prement alhora la tecla *Alt*, i clicar sobre la part que volem foradar. Ara només falta exportar-lo en format *.obj* per ser tractat a Cinema 4D.

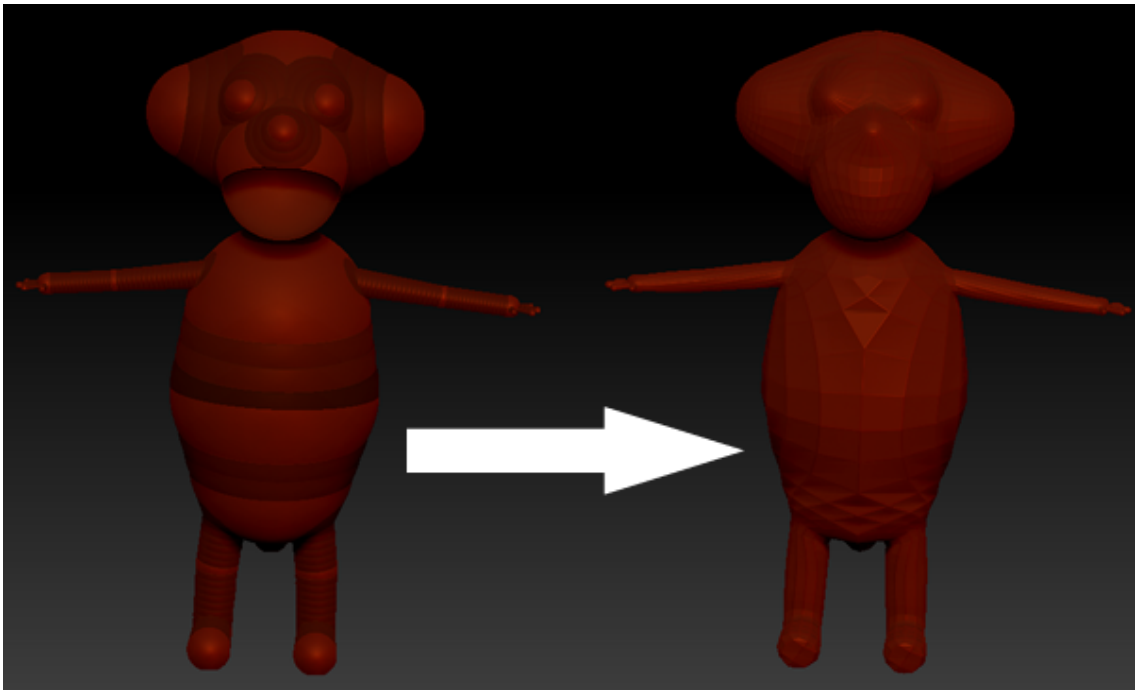


Fig. 37. Conversió a figura geomètrica

Amb Cinema 4D obert, és hora d'afegir-li articulacions per poder crear les animacions.

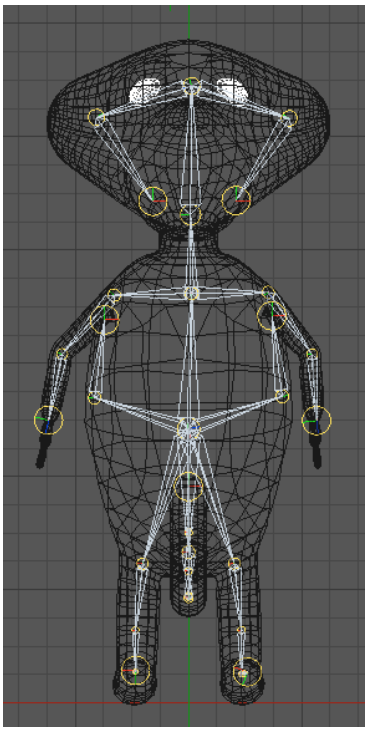
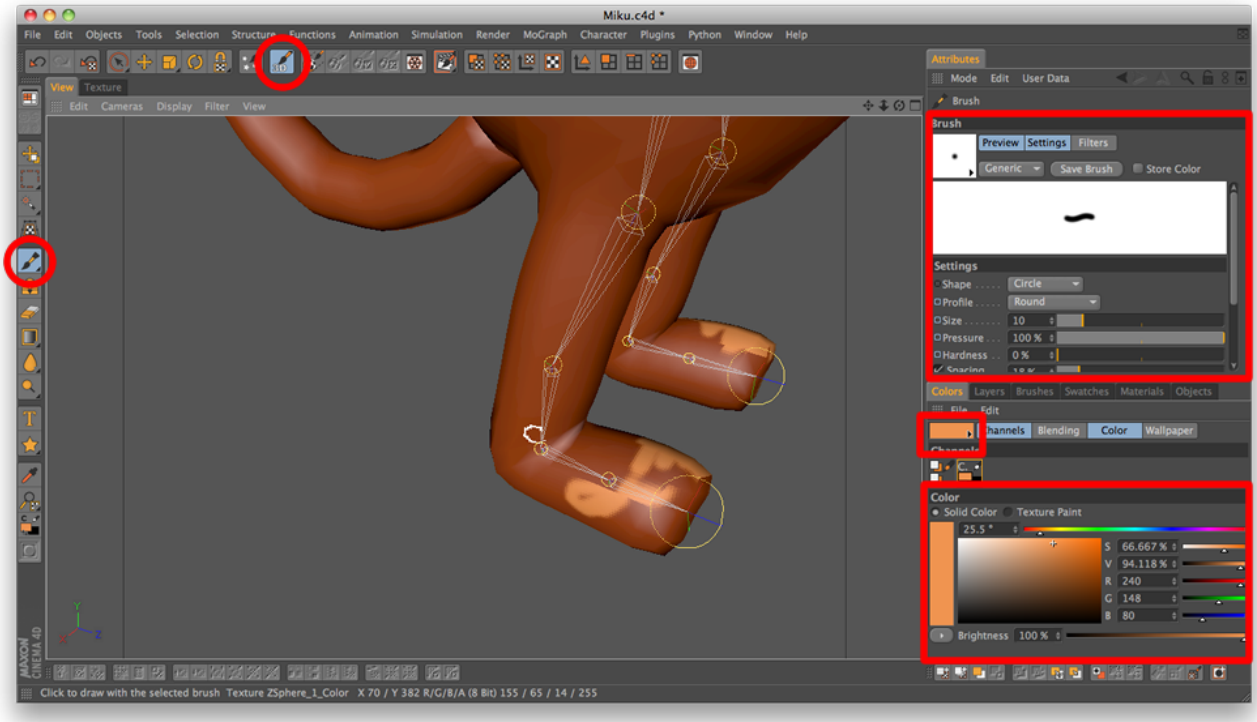


Fig. 38. Vista frontal amb les articulacions

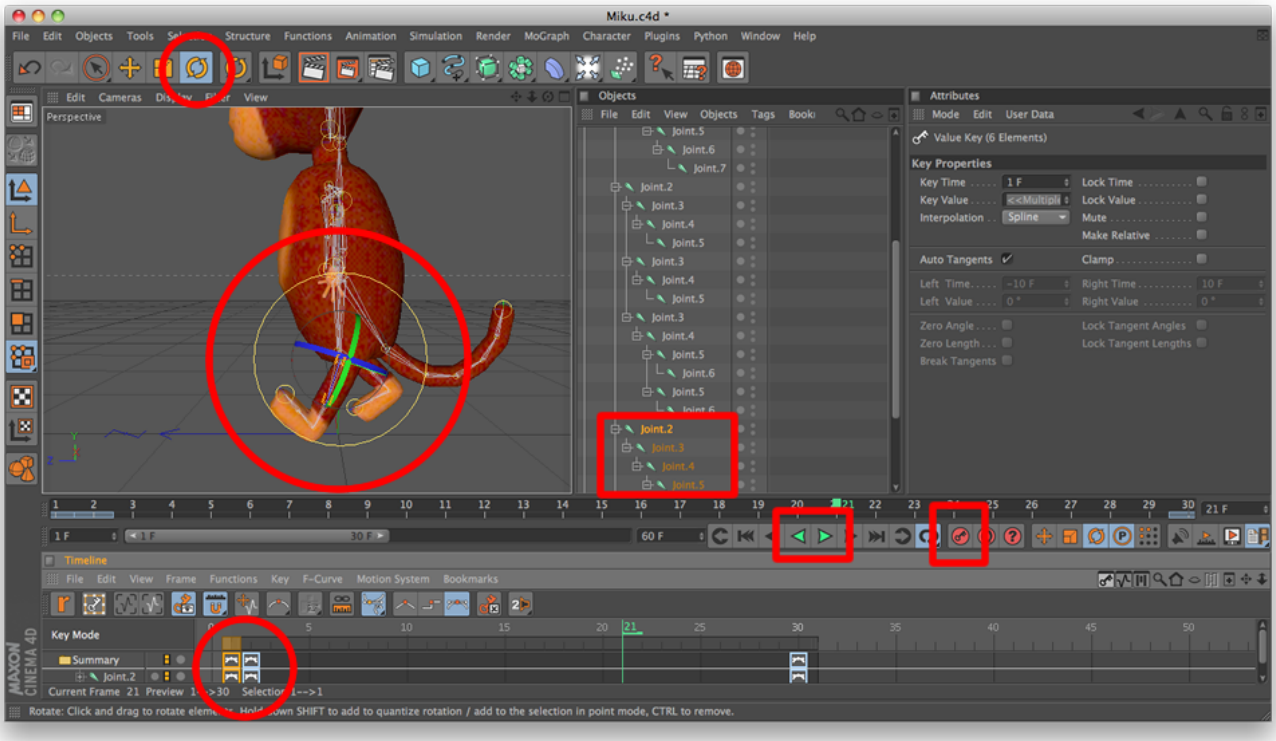
Desplaguem el menú *Character* i seleccionem *Joint Tool*. Aquesta eina permet crear, moure i eliminar les articulacions del personatge. Canviem a vista frontal i li afegim les articulacions des dels peus fins al cap (fig. 38). Com més articulacions tingui, més real serà el moviment però més difícil serà controlar-lo.

Quan creiem que ja tenim suficients articulacions, les alineem amb l'eina *Align* i tot seguit les seleccionem totes i apliquem l'opció *Blind*. Aquest unirà les articulacions amb el cos 3D.

Cinema 4D ofereix un servei anomenat *BodyPaint 3D* que ens servirà per donar-li color. Primer hem de seguir un assistent que ens guiarà i preguntarà quina textura o objecte volem pintar. Per defecte s'executarà el pinzell i a partir d'aquí només haurem d'anar controlant el gruix i el color que volem aplicar.



Per acabar-ho de completar, anem al panell d'animacions i ens situem al primer fotograma. Elegim l'articulació de la cama dreta i la girem 60º i guardem la posició amb un fotograma clau. Fem el mateix procés amb la cama esquerra amb una angle de -60º. Dins del fotograma 60, girem -120º per la cama dreta i 120º per la cama esquerra i ho guardem en un fotograma clau. Cinema 4D ja ens farà les animacions automàticament. Ara només falta exportar el personatge en *.fbx* a la carpeta del joc perquè Unity el detecti.



4.4 PROGRAMACIÓ DELS SCRIPTS

JavaScript no és un llenguatge de programació pròpiament dit. És un llenguatge script orientat a documents, com poden ser els llenguatges de macros que tenen molts processadors de text. Mai es podrà fer un programa completament amb JavaScript.

Els scripts són un llenguatge de programació que controlen aplicacions. Els scripts són executats directament des del seu codi font que, generalment, són fitxers de text que contenen llenguatges de marcatge específics.

La sintaxi d'un llenguatge de programació es defineix com el conjunt de regles que s'han de seguir en escriure el codi font dels programes per a considerar-se com correctes per a aquest llenguatge de programació.

Els operadors són un dels elements fonamentals en qualsevol llenguatge de programació, ja que són els que ens permeten treballar amb variables i dades. La taula situada a la dreta exposa els diferents operadors dividits en categories.

Operador	Significat
==	igual que
!=	diferent que
<	més petit que
<=	més petit o igual que
>	més gran que
>=	més gran o igual que
&&	i (operador i operador)
||	o (operador o operador)
!	negació

La sintaxi del JavaScript és molt similar a la d'altres llenguatges de programació com ara Java i C. Les normes bàsiques que defineixen la sintaxi de JavaScript són les següents:

- No es tenen en compte els espais en blanc i les noves línies: l'intèrpret de JavaScript ignora qualsevol espai en blanc sobrant, de manera que el codi es pot ordenar de forma adequada per entendre-ho millor.
- Es distingeixen les majúscules i minúscules: si en JavaScript s'intercanvien majúscules i minúscules, l'script no funciona.
- No es defineix el tipus de les variables: en crear una variable, no cal indicar el tipus de dada que emmagatzemarà. D'aquesta manera, una mateixa variable pot emmagatzemar diferents tipus de dades durant l'execució del script.
- No és necessari acabar cada sentència amb el caràcter de punt i coma: en la majoria de llenguatges de programació, és obligatori acabar cada sentència amb el caràcter (;). Encara que JavaScript no obliga a fer-ho, és convenient seguir la tradició d'acabar cada sentència amb el caràcter del punt i coma (;).
- Es poden incloure comentaris: els comentaris s'utilitzen per afegir informació en el codi font del programa. Encara que el contingut dels comentaris no es visualitza per pantalla, si que s'envia al navegador de l'usuari juntament amb la resta de l'script, per

la qual cosa cal extremar les precaucions sobre la informació inclosa en els comentaris. JavaScript defineix dos tipus de comentaris: els d'una sola línia i els que ocupen diverses línies.

A continuació explicaré scripts senzills aplicats al joc. Els scripts sencers es poden trobar a l'Annex A.

Nom	Nº
Menu_cat	0
Nivell 1	1
Nivell 2	2
Nivell 3	3
Gameover	4
Instruccions	5
Menu_eng	6
Instructions	7
Fi	8

El següent explica el canvi de nivell. Quan el jugador arriba a un objecte, anomenat `Caixa` i hi xoca, automàticament passa al següent nivell. Per programar-ho primer de tot hem de cridar la funció `OnTriggerEnter`, classe definida per Unity, el qual detecta si un personatge entra dintre un objecte. Tot seguit afegim una variable anomenada xoc que serà una col·lisió (`Collider`) i li incorporem un condicional al seu interior.

El que programa interpreta és: si el personatge xoca contra l'objecte `Caixa`, passem al nivell 2, sinó, segueix jugant.

```
1. function OnTriggerEnter (xoc : Collider)
2. {
3.       if (xoc.gameObject.name == "Caixa")
4.       {
5.             Application.LoadLevel(2);
6.       }
7. }
```

Sempre podem incorporar comentaris d'una línia o que ocupin diverses línies. El següent exemple hi apareixen tots dos.

```
1. //Això és un comentari d'una sola línia
2.
3. print("Això és un missatge");
4.
5. /* Això és un comentari que ocupa diverses línies, és molt útil
6. si necessitem incloure força informació en els comentaris */
```

Els comentaris són útils per deixar anotacions i mai seran llegits ni executats per l'intèrpret, per tant, no implica la seva presència.

El proper script explica el comportament de la porta del nivell 2. Després d'anomenar la funció `OnControllerColliderHit`, creem una variable anomenada `hit`. Quan el personatge topi contra l'objecte anomenat `Clau`, s'eliminarà l'objecte (amb la funció `Destroy`) que està etiquetat com a `destruccio`, en aquest cap la porta. Així el jugador podrà sortir del laberint i passar al següent nivell.

```
1. function OnControllerColliderHit
2.      (hit : ControllerColliderHit) {
3.      if (hit.gameObject.name == "Clau")
4.      {
5.           Destroy(GameObject.FindWithTag("destruccio"));
6.      }
7. }
```

Programarem una porció del menú del propi joc. En aquest cas ens bastarem de la funció `OnGUI` ja que permet crear una gran varietat d'interfícies gràfiques d'usuari amb funcionalitat completa molt ràpida i fàcilment. En comptes de crear un objecte d'interfície gràfica d'usuari, posicionant-lo manualment i escriure un script que s'encarregui de la seva funcionalitat, Unity ens ofereix tot això en una petita quantitat de codi.

```
1. function OnGUI () {
2.      GUI.Box (Rect (320,70,350,500), "MIKU");
3.
4.      //Botó del Nivell 1
5.      if (GUI.Button (Rect (400,150,200,80), "Nivell 1")) {
6.           Application.LoadLevel (1);
7.      }
8. }
```

Des de la funció `GUI`, creem un `Box` (un fons) i `Button` (un botó). Els quatre nombres separats per comes indiquen les coordenades de posicionament i la seva mida.

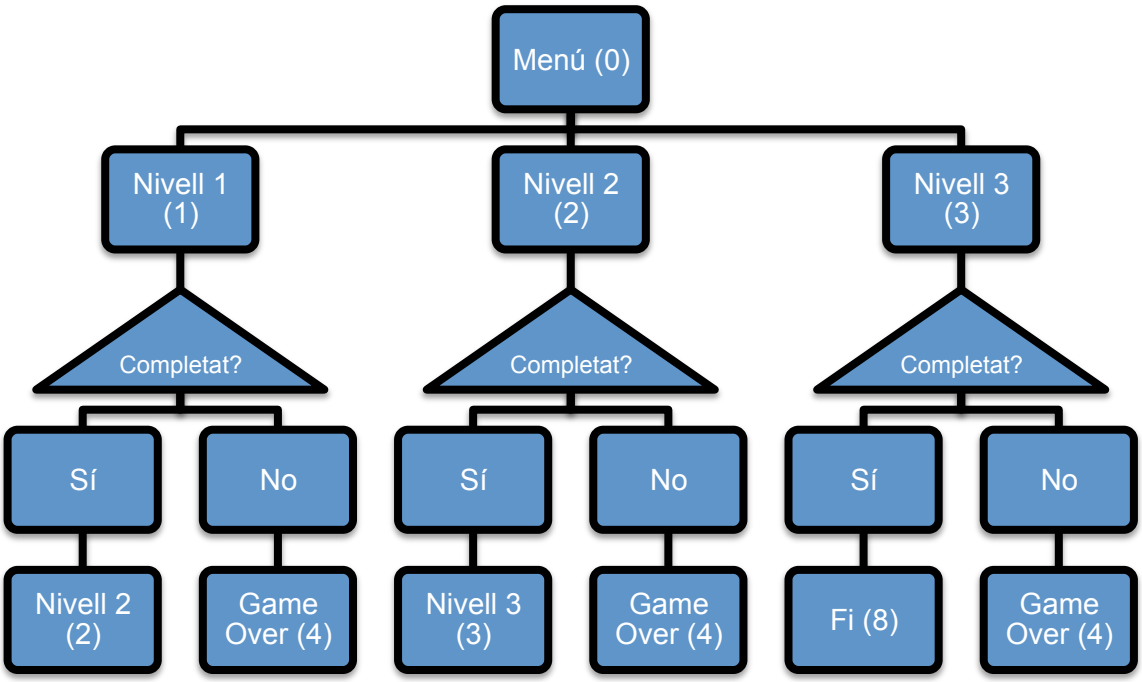
`(400,150,200,80) = (coordenada X, coordenada Y, mida X, mida Y)`

En el moment que premem el botó anomenat `Nivell 1`, passarem al nivell 1 del joc. Es pot veure l'script sencer a l'Annex A, el qual inclou so, els botons de tots els nivells, les instruccions i els idiomes.

L'últim script ens mostra els moviments emprats pel personatge. Iniciem la funció `Start` que pararà qualsevol animació que hi hagi en el moment d'iniciar el joc. Després utilitzem la funció `Update`, que permet controlar cada fotograma que passa en el joc. Quan l'script detecti que el *joystick* esquerre es mogui, el personatge automàticament emprarà l'animació `walk`, sinó estarà quiet (`idle`).

```
1.  function Start ()
2.  {
3.     animation.Stop();
4.     animation.CrossFade ("idle");
5.  }
6.
7.  function Update ()
8.  {
9.     if (Input.GetAxis("Vertical") > 0.2)
10.        animation.CrossFade ("walk");
11.    else
12.       animation.CrossFade ("idle");
13. }
```

Acabarem aquest apartat amb un esquema bàsic del joc:



5. EXPORTACIÓ

Un cop fet tots els procediments descrits, només faltarà exportar el joc al propi mòbil. Per aquesta tasca podem canviar les configuracions des del menú *Player Settings* d'Unity. Podem triar diferents opcions com ara la selecció de la icona del joc, la versió del videojoc o la versió de sistema operatiu que es vol emprar (fig. 39).

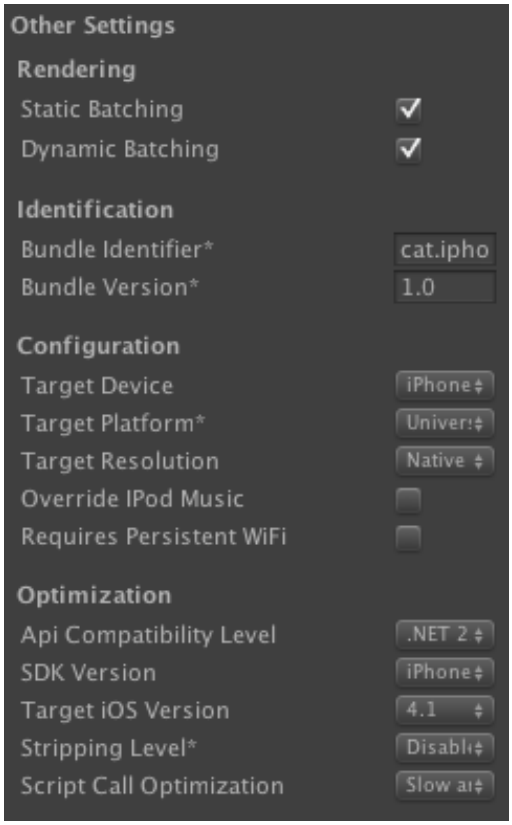


Fig. 39. Configuració de la plataforma

Establertes les configuracions, anem a *Build Settings* i ordenem amb nombres totes les escenes dels jocs perquè el programa les pugui identificar i relacionar amb el codi programat. Podem triar dues opcions: *Build*, que només construirà i guardarà el joc, o *Build & Run*, aquest construirà, guardarà i executarà directament el joc al nostre mòbil.

Anem a la destinació que hem posat prèviament i veurem un arxiu compatible amb XCode. L'obrim i ens apareixerà una finestra amb una gran llista d'arxius (fig. 40). A partir d'aquí podem decidir què fem amb el joc. El podem executar en el mateix ordinador gràcies a un simulador per iPhone o el podem executar al nostre iPhone, tot i que necessitem tenir un registre com a desenvolupador d'iOS.

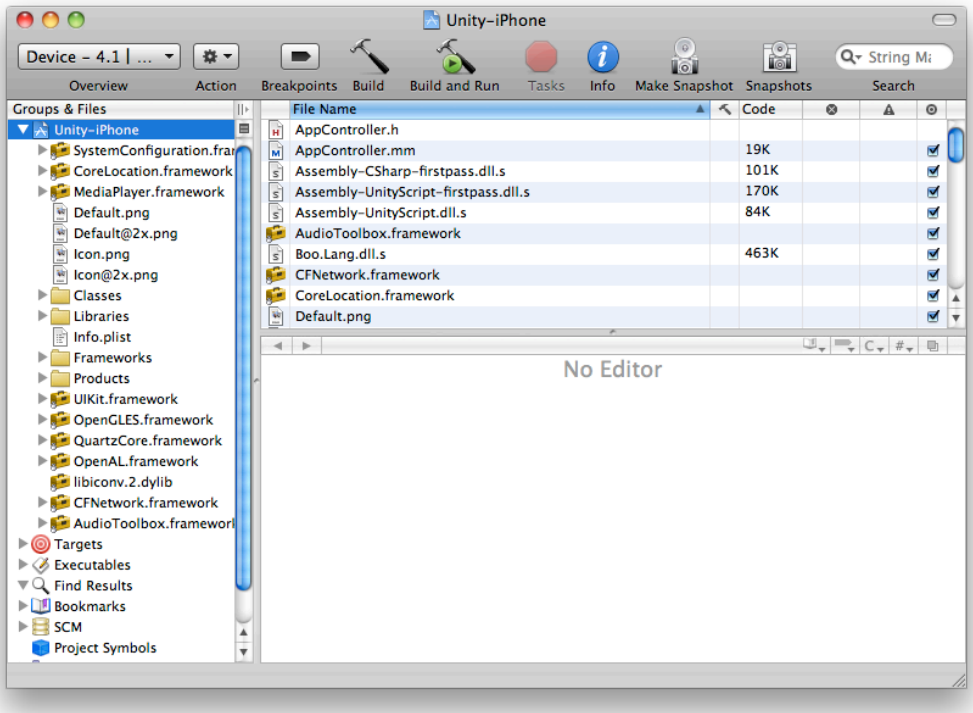


Fig. 40. Captura de XCode

Miku es pot trobar a l'App Store per a la versió 4.1 o superior d'iOS. Per aconseguir-lo hem d'entrar a l'iTunes Store o des de l'aplicació App Store d'un dispositiu iOS. Després anem a la secció Buscar i escrivim Miku. Finalment premem Buscar i seleccionem el seu nom per poder-lo baixar.

6. CONCLUSIONS

Primer de tot, crec que he anat completant els objectius marcats de bon principi. Sense tenir cap idea concreta de com arribaria a ser el joc, vaig planejar fer més del doble de nivells que he fet. Després, amb els mínims coneixements que vaig aconseguir i en veure que cada nivell portava molta feina, vaig decidir tornar a plantejar-me el joc.

M'he trobat amb la gran dificultat d'expressar els passos a seguir perquè sigui comprensible per a tothom ja que s'utilitza un llenguatge molt tècnic i enrevessat per qui mai ha sentit a parlar d'aquests temes.

No tenia cap coneixement previ de la creació d'objectes en 3D i menys d'un personatge. No tenia cap experiència amb cap programa treballat amb aquest joc i, a base d'errors, m'hi he anat familiaritzant. No ha estat una tasca fàcil adaptar-se a les funcionalitats de cada programa, sobretot *ZBrush*, el qual té una mecànica ben diferent als altres programes que he provat mai.

Tenia una idea equivocada: pensava que era fàcil i ràpid crear un personatge en 3D pel joc. Crear un personatge és una de les feines més dures que hi ha. S'ha de mirar fins al més mínim detall des de la seva producció fins a les seves animacions. A l'hora d'aplicar-ho, me'n vaig adonar que va ser la feina més feixuga que vaig realitzar en tot el joc. Fins i tot, podia haver fet un treball a part només dedicat al personatge. En canvi, l'apartat de programació ha estat una de les parts més divertides i fàcils d'aportar al joc gràcies al gran nombre de funcionalitats que ofereix *Unity*.

He pogut comprovar com, realment, en un joc es necessita un equip ben especialitzat. Com a mínim es requereix algú que inventi l'argument, un altre que faci els gràfics i algú que s'ocupi de la programació i la lògica del joc.

Tot i les grans dificultats que m'he pogut trobar, he acabat fent un videojoc bàsic i adaptat a la plataforma que em proposava. Com a possibles millores que se li podrien afegir, i segurament s'hi afegiran a partir d'actualitzacions, són enemics amb intel·ligència artificial i un nou nivell més per allargar la durada del joc.

7. AGRAÏMENTS

Primer de tot vull agrair al tutor, que sempre m'ha donat suport i sempre s'ha mostrat a la meva disposició pel que fes falta. També agraeixo a tots els meus amics que han provat el joc i m'han donat consells per millorar-lo. Especialment en David Palomino, qui ha provat el meu joc basat en Android i al meu cosí, qui m'ha ajudat a construir i millorar aquest treball.

Per acabar, agraeixo a totes les persones que s'han mostrat interessades pel meu treball i m'han donat suport ja que han permès que arribes a fer un treball amb aquest alt grau de dificultat.

8. BIBLIOGRAFIA

Informació d'Unity 3D:

@ http://en.wikipedia.org/wiki/Unity_%28game_engine%29

@ http://unity3d.com/unity/engine

Història de les videoconsoles:

@ http://en.wikipedia.org/wiki/Video_game_console

@ http://en.wikipedia.org/wiki/Handheld_game_console

@ http://www.thegameconsole.com

@ http://www.pixfans.com/historia-de-las-primeras-videoconsolas

@ http://indicelatino.com/juegos/historia/consolas

@ http://vimeo.com/1732767

Tutorials de disseny 3D i Unity 3D:

@ http://vimeo.com/3281126

@ http://forum.unity3d.com

@ http://www.youtube.com/watch?v=wN9tbyJmH0U

@ http://www.youtube.com/tornadotwins

Programació:

@ http://unity3d.com/support/documentation/ScriptReference/MonoBehaviour.html

CAE, S.A., *Programación cliente*. Gràfiques BDM, C.B. – València.

ANNEX A:

CODI FONT DEL VIDEOJOC

Script del menú:

```
1.  var customSkin : GUISkin;
2.  var angles : Texture2D;
3.  var catala : Texture2D;
4.  var click : AudioClip;
5.
6.  function OnGUI () {
7.       var volume : float;
8.
9.       GUI.skin = customSkin;
10.
11.      //Fons
12.      GUI.Box (Rect (320,70,350,500), "MIKU");
13.
14.      //Nivell 1
15.      if (GUI.Button (Rect (400,150,200,80), "Nivell 1")) {
16.           audio.PlayOneShot (click);
17.           Application.LoadLevel (1);
18.      }
19.
20.      //Nivell 2
21.      if (GUI.Button (Rect (400,250,200,80), "Nivell 2")) {
22.           audio.PlayOneShot (click);
23.           Application.LoadLevel (2);
24.      }
25.
26.      //Nivell 3
27.      if (GUI.Button (Rect (400,350,200,80), "Nivell 3")) {
28.           audio.PlayOneShot (click);
29.           Application.LoadLevel (3);
30.      }
31.
32.      //Idioma
33.      GUI.Box (Rect (690,70,250,175), "Idioma");
34.      if (GUI.Button (Rect (710,130,100,100),catala)) {
35.           Application.LoadLevel (0);
36.           }
37.
38.      if (GUI.Button (Rect (820,130,100,100),angles)) {
39.           Application.LoadLevel (6);
40.           }
41.
42.      //So
43.      if (GUI.Button (Rect (690,280,250,100), "So")) {
44.           if (audio.volume == 1.0) {
45.                audio.volume = 0.0;
46.                audio.PlayOneShot (click);
```

```
47.           }
48.           else {
49.                audio.PlayOneShot (click);
50.                audio.volume = 1.0;
51.           }
52.      }
53.
54.      //Instruccions
55.      if (GUI.Button (Rect (690,420,250,100), "Instruccions")) {
56.           audio.PlayOneShot (click);
57.           Application.LoadLevel (7);
58.      }
59. }
```

Script per obrir la porta:

```
1.  function OnControllerColliderHit
2.       (hit : ControllerColliderHit) {
3.       if (hit.gameObject.name == "Clau")
4.       {
5.            Destroy(GameObject.FindWithTag("destruccio"));
6.       }
7.  }
```

Script per transformar el moviment de la plataforma en el moviment del personatge:

```
1.  var plataforma : Transform;
2.  var jugador : Transform;
3.
4.  function OnTriggerEnter ()
5.  {
6.       var lloc = plataforma;
7.       var anar = jugador;
8.       anar.transform.parent = lloc.transform;
9.  }
10.
11. function OnTriggerExit ()
12. {
13.      var anar = jugador;
14.      var lloc = plataforma;
15.      anar.transform.parent = null;
16. }
```

Script del comptador de plàtans:

```
static var NUMERO_PLATANS = 0;

function OnControllerColliderHit (cop : ControllerColliderHit)
{
      if (cop.gameObject.tag == "platan")
      {
            Destroy(cop.gameObject);
            NUMERO_PLATANS += 1;
            GameObject.Find("numeros_platans").guiText.text =
""+NUMERO_PLATANS;
      }
}
```

Script del funcionament de les vides:

```
var vida1 : Texture2D; //no queda cap vida
var vida2 : Texture2D; //queden 1 vides
var vida3 : Texture2D; //queden 2 vides
var vida4 : Texture2D; //queden 3 vides

static var VIDES = 4;

function Update()
{
      switch(VIDES)
      {
            case 4:
                  guiTexture.texture = vida4;
               break;
            case 3:
                  guiTexture.texture = vida3;
               break;
           case 2:
            guiTexture.texture = vida2;
            break;
           case 1:
            guiTexture.texture = vida1;
            break;
           case 0: //GAME OVER
            print("GAME OVER");
            Application.LoadLevel(4);
            break;
      }
}
```

Script del funcionament de la caiguda:

```
1.  private var PosicioInicial : Vector3;
2.
3.  function Awake()
4.  {
5.       PosicioInicial = transform.position;
6.  }
7.
8.  function OnTriggerEnter(caiguda : Collider)
9.  {
10.      if(caiguda.gameObject.name == "caiguda")
11.      {
12.           Vides.VIDES -= 1;
13.           PosicioInicial.y += 4;
14.           transform.position = PosicioInicial;
15.      }
16. }
```

Script que s'ocupa del text del comptador de plàtans.

```
7.  function Awake () {
8.       guiText.text = "" + 0;
9.  }
```

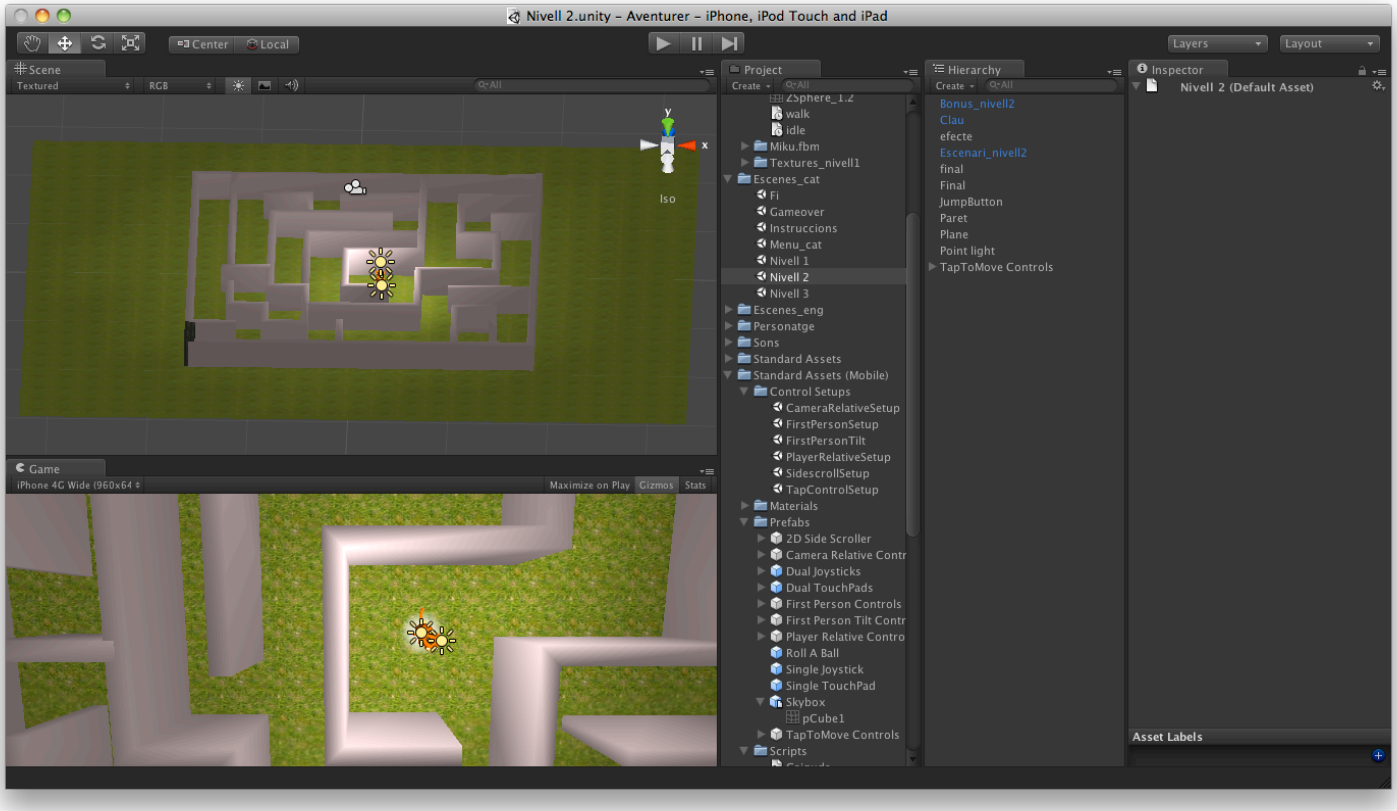
Script del final de cada nivell

```
1.  function OnTriggerEnter(xoc : Collider)
2.  {
3.       if(xoc.gameObject.name == "final")
4.       {
5.            Application.LoadLevel(2); //depèn de cada nivell
6.       }
7.  }
```

ANNEX B:

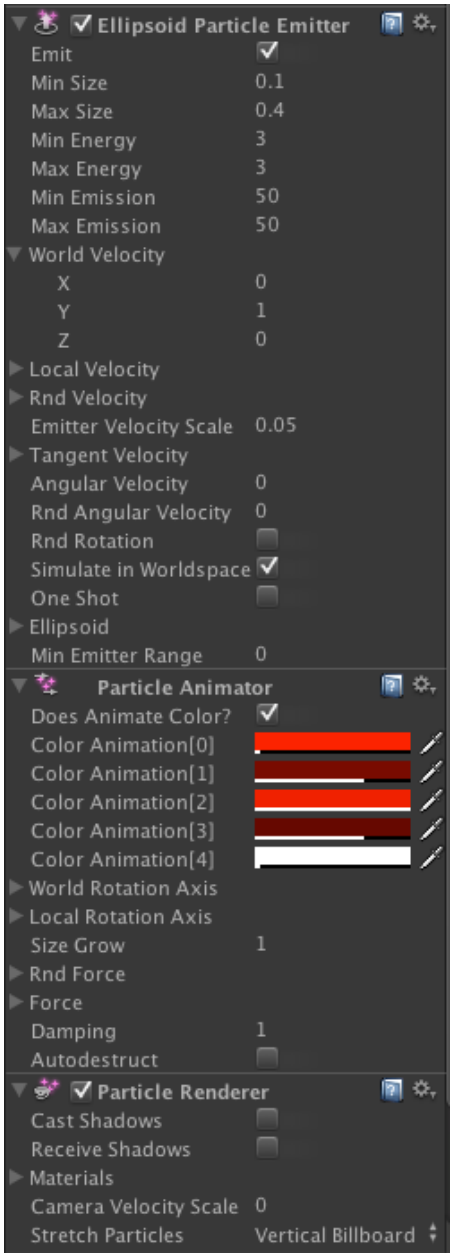
TOTA LA INFORMACIÓ DELS NIVELLS

En total, el joc està compost per tres nivells. El primer ha estat explicat pas a pas en el treball. Els altres dos nivells s'han creat de la mateixa manera que el nivell 1 però amb diferents modes de jocs. En aquest apartat deixaré algunes imatges mostrant els nivells fets a Unity així com les seves configuracions.

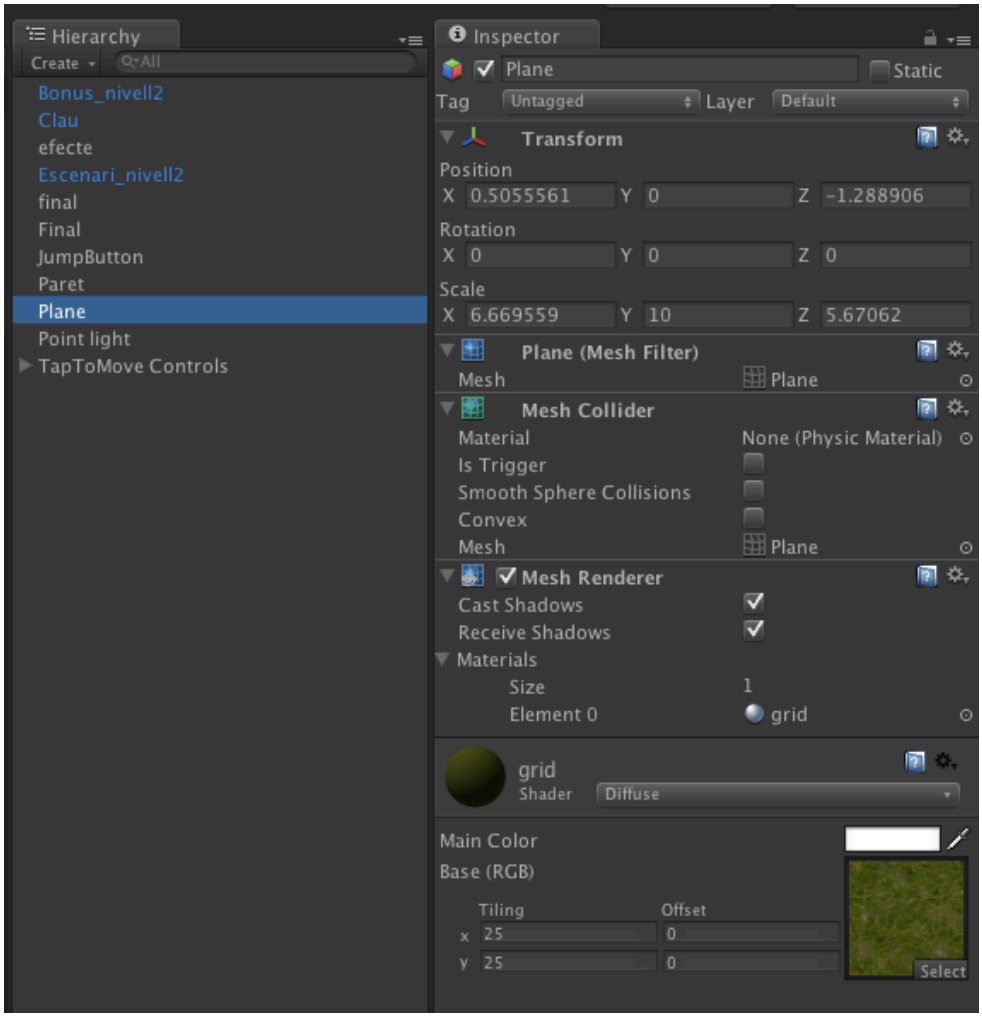


I. Visualització completa del nivell 2

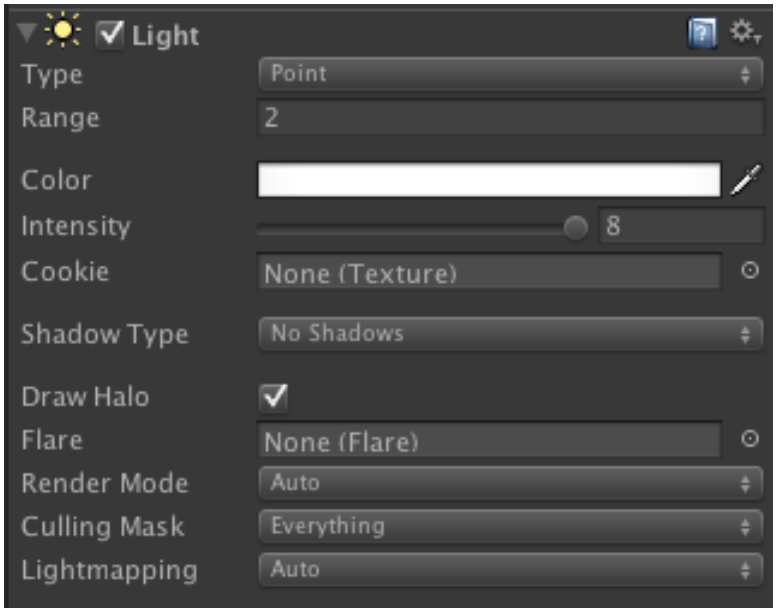
II. Imatge del segon nivell (laberint)



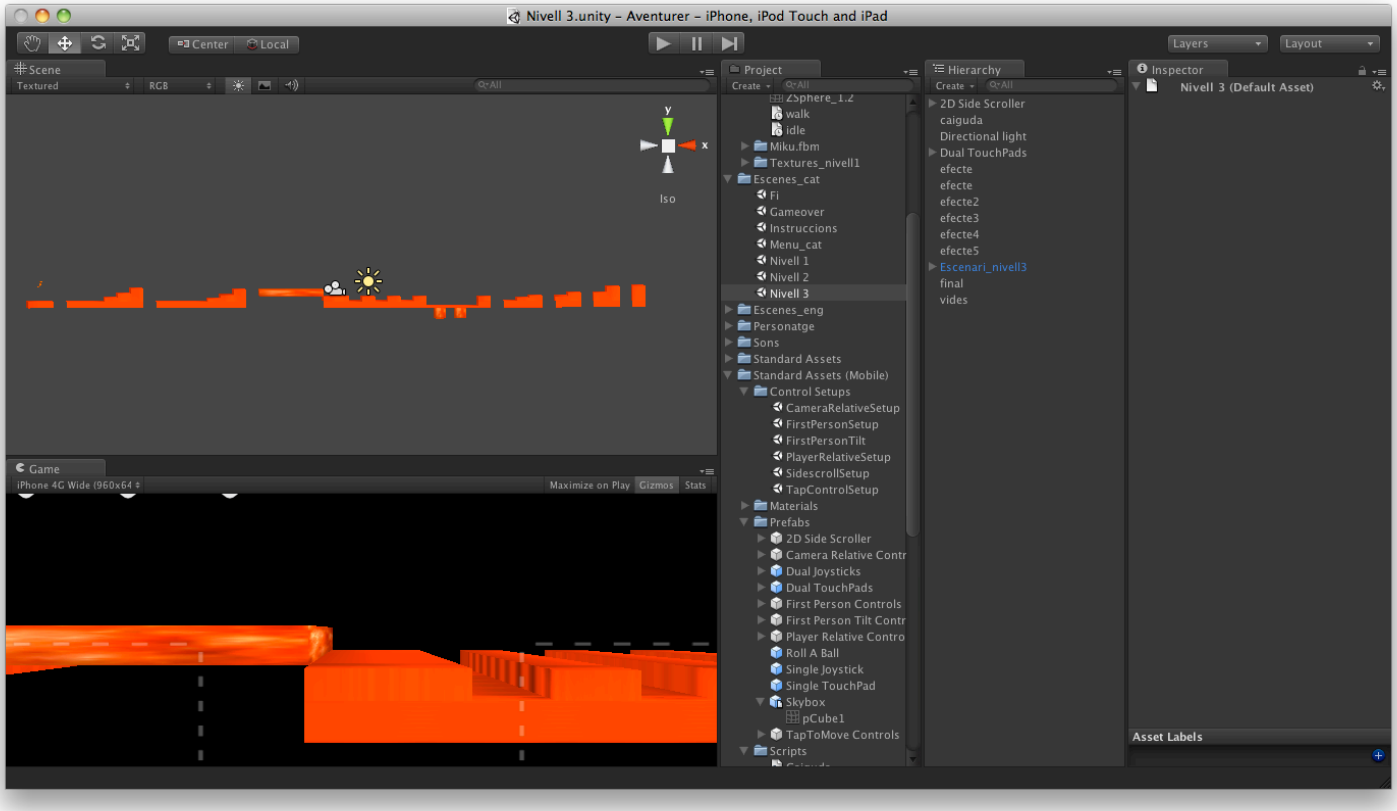
III. Configuració de l'efecte especial del laberint



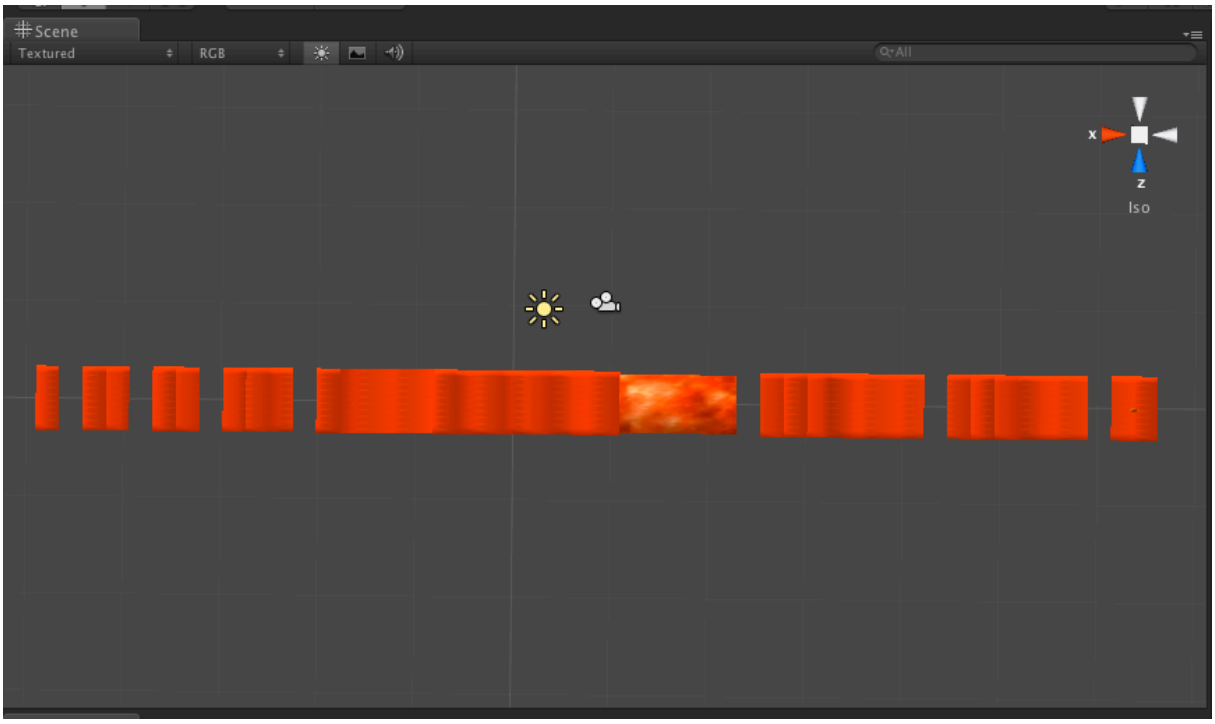
IV. Propietats del terreny límit del nivell 2



V. Configuració de la llum que segueix el personatge per il·luminar el laberint



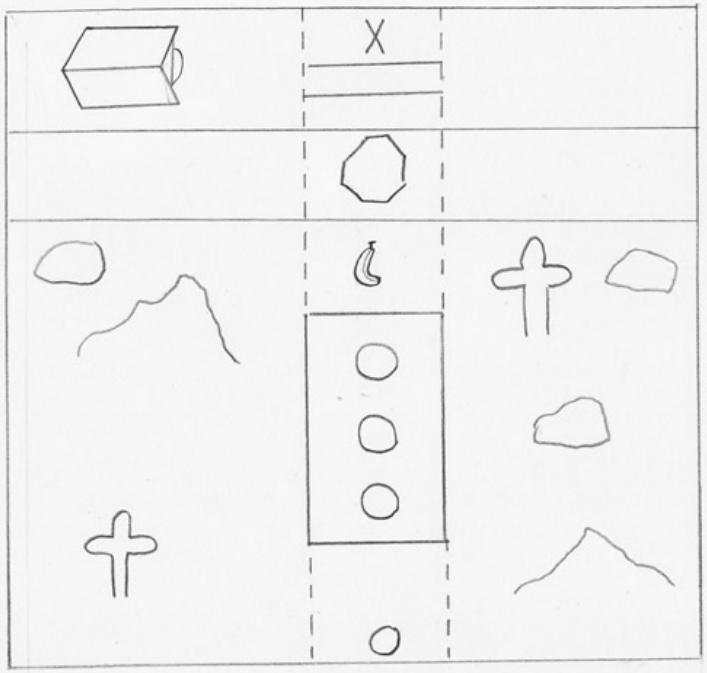
VI. Captura del tercer nivell



VII. Vista en planta de l'escenari del tercer nivell

ANNEX C:
ESBOSSOS DELS ESCENARIS

Vista en planta del nivell



Nom del nivell: Wild West Nº: 1

Objectes destacats:

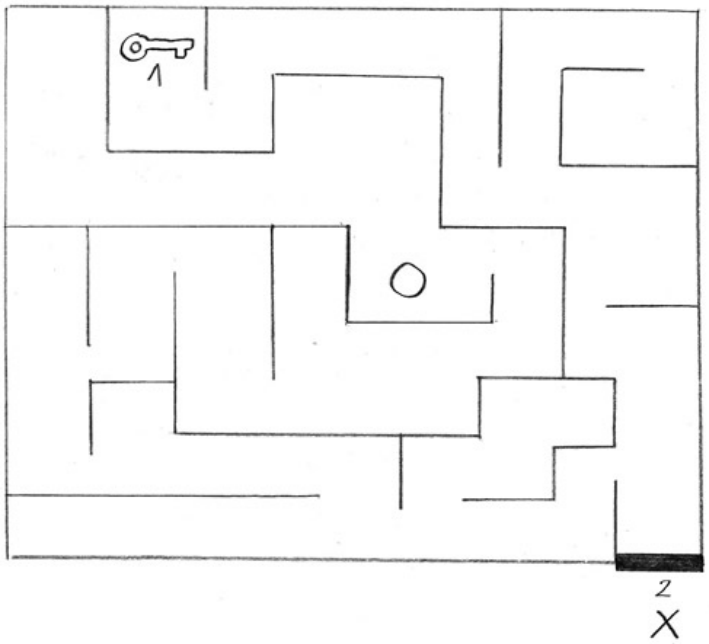
1. Cactus
2. Plataforma mòbil
3. Platan
4. Turons
5. Pedres
6. Pou

Descripció: primer nivell del joc que implicarà un mode totalment en 3D amb dos joysticks. El joystick de l'esquerra controlarà la càmera, mentre que el de la dreta el moviment. Si es toca dos cops el joystick dret durs vegades, el personatge saltarà. S'haurà de saltar sobre tres cilindres, agafar un platan i passar per una plataforma que es mourà per arribar al final del nivell.

Llegenda:

O	Inici
X	Final
→	Moviment
⊘	Prohibit el pas
- - -	Límit

Vista en planta del nivell



Nom del nivell: el Laberint Nº: 2

Objectes destacats:

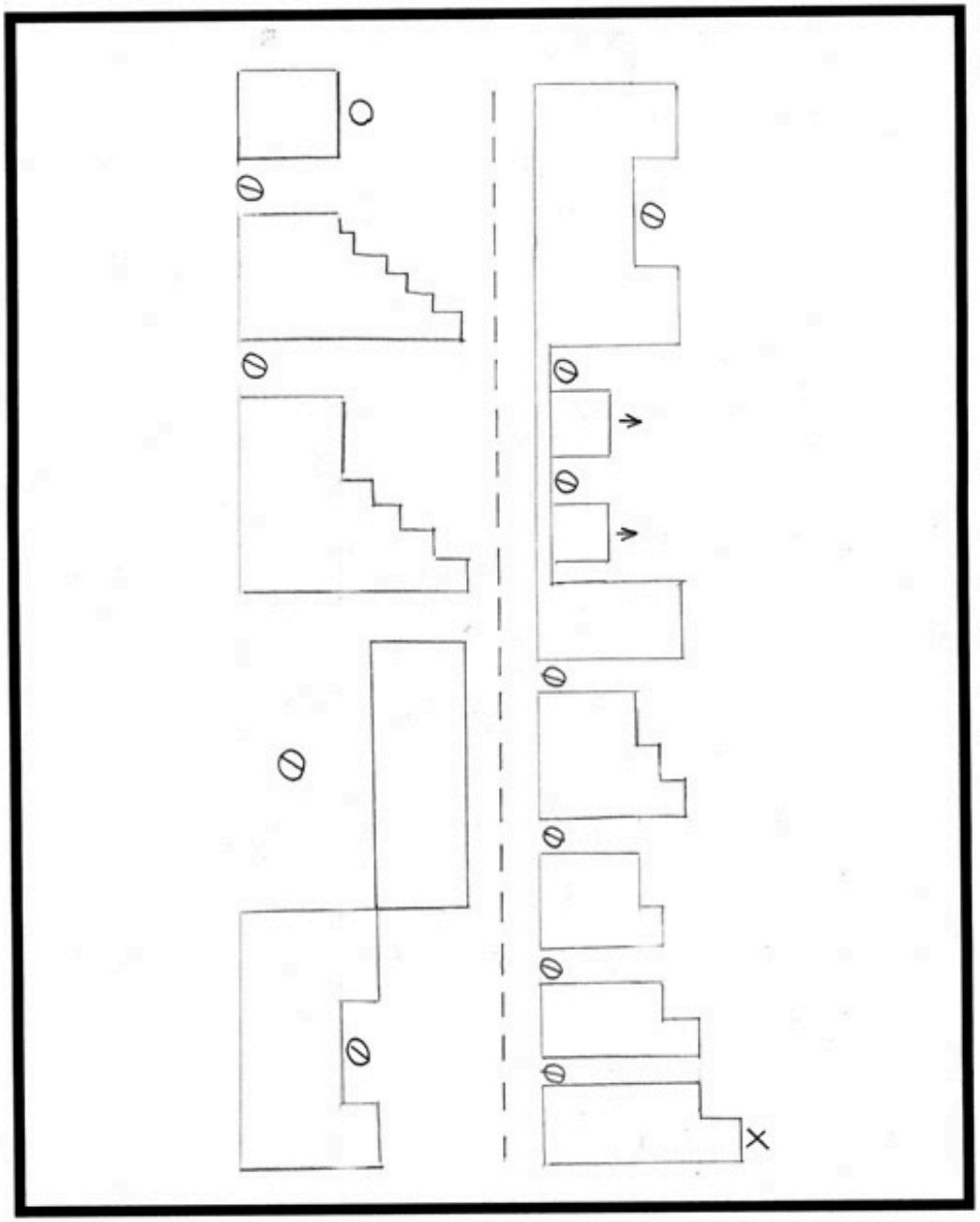
1. Clau
2. Porta
3.
4.
5.
6.

Descripció: es tracta d'un laberint que, resseguint el camí amb el dit, s'haurà d'aconseguir una clau que obrirà la porta per passar al següent nivell.

Llegenda:

O	Inici
X	Final
→	Moviment
⊘	Prohibit el pas
- - -	Límit

Alçat del nivell



Nom del nivell: Scrolling 2D level Nº: 3

Objectes destacats:

1. Blocs mòbils
2.
3.
4.
5.
6.

Descripció: nivell amb jugabilitat en 2D però amb gràfics 3D, que per controlar-lo hi haurà dos botons a la part inferior de la pantalla. El de l'esquerre controlarà el moviment (dreta o esquerre) i el botó de la dreta, els salts.
Hi haurà plataformes mòbils que aniran pujant i baixant amb la intenció de dificultar el pas.

Llegenda:

O	Inici
X	Final
→	Moviment
⊘	Prohibit el pas
- - -	Límit