Desenvolupament d’un videojoc de plataformes – Benny The last Hope

Aleix Díez Cugat
Grau d’Enginyeria Informàtica

Jordi Duch Gavaldà
Helio Tejedor Navarro

16/06/2017

FITXA DEL TRABALL FINAL

Títol del treball:	*Desenvolupament d'un videojoc de plataformes – Benny The last hope*
Nom de l'autor:	*Aleix Díez Cugat*
Nom del consultor/a:	*Jordi Duch Gavaldà / Helio Tejedor Navarro*
Nom del PRA:	*Aleix Díez Cugat*
Data d'entrega (mm/aaaa):	06/2017
Titulació:	*Grau d'Enginyeria Informàtica*
Àrea del Treball Final:	*05.640 TFG - Videojocs educatius aula 1*
Idioma del treball:	*Català*
Paraules clau	*Videojoc, Plataformes, Desenvolupament*

Resum del Treball

El projecte final del Grau ha consistit en el disseny i desenvolupament d'un videojoc de plataformes per a ordinador, a l'estil Super Mario, obtenint un producte final original, funcional i totalment acabat.

Per un interès personal, tot el desenvolupament s'ha fet evitant la utilització de suports tècnics específics automatitzats, per tal de poder analitzar la mecànica interna de la programació i anar superant totes i cada una de les fases del procés del disseny i de l'executable.

Posteriorment, per obtenir un valor afegit, s'ha dissenyat la possibilitat d'executar el joc també en els dispositius Android.

El treball pretén aplicar totes les fases en la producció d'un videojoc, des de l'anàlisi inicial realitzat que justifica l'opció de videojoc escollida, la concepció, la producció i fins el manteniment i/o actualització del programari.

El videojoc inclou dos escenaris (ampliables). La mecànica correspon a la d'un joc de plataformes convencional, on el jugador haurà d'obtenir la major quantitat de recursos possible (aigua), tenint en compte a l'hora el combustible que li resta (oxigen) i mentre ha d'anar esquivant i/o superant els diferents obstacles que sorgeixen en el nivell, amb l'objectiu d'arribar viu al final de cada

nivell.

El llenguatge utilitzat per la seva codificació ha estat Java emprant Android Studio, com a IDE, amb el framework LibGdx i l'extensió Box2D, com a engine de físiques.

Com a eines de suport s'han fet servir: Gimp 2, com a editor d'imatges i textures, Aseprite, com a editor d'animacions d'Sprites, i Tiled, com editor de tile maps.

El producte desenvolupat ha estat *Benny, the last hope,* un videojoc de plataformes en 2D, clàssic, del tipus Super Mario, d'estil retro, amb tècnica Pixel art, ambientat en un entorn futurista.

A l'hora de presentar aquest Treball, es pot considerar que s'han assolit plenament els objectius plantejats, tant el de la creació d'un producte funcional acabat, com el de conèixer la mecànica interna del desenvolupament imprescindible en el disseny i producció d'un videojoc.

Abstract

The final project consist in to the design and development of a platform game for computer with Super Mario style obtaining a final original product, functional, and finished.

For a personal interest, for all the development we try to evade the use of automatic supports because we wanted to understand and overcome all the phases in the game development.

Afterwards, for obtain an added value we developed the possibility of execution the game in android devices too.

This work pretends to apply all the phases of a videogame development taking in consideration the main analysis which justify the option choose about genre the game, the ideas, the production and maintenance or updates.

The game includes two scenarios. The mechanics correspond to a conventional platform game where player needs to obtain the large quantity of resources (water), watching the remaining oxygen and evade or overcoming the diferents obstacles in the level with the objective of arrive alive.

The coding language used is Java using Android Studio as IDE with LibGdx framework and Box2D extension as physics engine.

As suport tools we are used: Gimp 2 as image and texture editor, Aseprite as animation sprite editor and Tiled as tile map editor.

The developed product is Benny The last hope, it is a classic 2D platform game similar to Super Mario with a retro style using Pixel art technic and futuristic setting.

At the time of presenting the work, we can consider that we accomplished all the objectives since the creation of a functional product and to learn the internal work of development for a design and production of a game.

Índex

1 Introducció

Aquest treball que es presenta, recull el disseny i la execució del Projecte Final del Grau d'Enginyeria Informàtica que s'emmarca específicament en l'Àrea de Videojocs.

La implementació del Projecte implica un repte com enginyer informàtic i programador que pretén endinsar-se "professionalment" en el sector del desenvolupament de videojocs en el que a l'hora és un nouvingut, que ha volgut apropar-se a aquest entorn i constatar, de primera mà tot el que requereix quan a coneixements i habilitats, a esforç i dedicació, el desenvolupament d'un videojoc, en aquest cas concret, del gènere plataformes 2D però que te com referent directe un dels jocs mítics, potser fins i tot, el més conegut de la història, el de Super Mario, amb el que han gaudit milions de jugadors de tot el mon, entre els que també m'hi trobo, en les darreres dècades, però també en l'actualitat.

Així doncs, executar el projecte descrit en aquest treball, hauria de permetre principalment dues coses: per una banda obtenir una perspectiva interna, molt més amplia i detallada del desenvolupament d'un videojoc, de la quantitat i diversitat de coneixements i habilitats que són necessaris si només hi ha un desenvolupador unipersonal i, també que depenent de l'escala del producte , ja és precís disposar d'un equip multidisciplinari i que, a més, estigui ben dirigit i molt coordinat.

La motivació personal i un dels objectius centrals del projecte són coincidents en desenvolupar un videojoc clàssic, de plataformes amb 2D, i amb un protagonista i una ambientació diferent dels jocs del *Super Mario* .

En el joc, el jugador ha de controlar un únic personatge en el seu trànsit per l'escenari i on va trobant tota una sèrie d'elements que dificulten o faciliten l'avanç del jugador.

2 Objectius del Treball

L'objectiu general del treball pretén aconseguir una immersió real en l'entorn dels videojocs i també una aproximació màxima al rol de desenvolupador
D'aquest objectiu general, deriven dos objectius específics:

2.1 Desenvolupar un joc de plataforma tipus Super Mario.

- Conèixer els antecedents i i analitzar l'evolució de les característiques dels videojocs del gènere de plataformes al llarg dels 30 anys i escaig de la seva història (1980 – 2017).
- Analitzar les característiques i evolució dels videojocs de plataforma tipus Super Mario des de la seva aparició fins avui.
- Emprar totes les fases de desenvolupament d'un videojoc per a ordinador: concepció, disseny, planificació, preproducció, producció, proves i manteniment.
- Aconseguir un joc acabat funcional i amb possibilitats d'ampliació.
- Aconseguir executar el joc també en dispositius Android.

2.2 Conèixer la mecànica interna del desenvolupament d'un videojoc

- Evitar la utilització de suports tècnics específics automatitzats, per tal de poder analitzar la mecànica interna de la programació i anar superant totes i cada una de les fases del procés del disseny i de l'executable.
- Determinar el volum i pluralitat de recursos tècnics i tecnològics que requereix el disseny i desenvolupament d'un videojoc en funció dels seus requeriments.
- Utilitzar en el disseny del videojoc, conceptes de programació obtinguts en experiències prèvies de treball amb altres tipus de programaris (CMS, Pàgines web, Eines de Gestió, Programes fets a mida, ...).

3 Metodologia

3.1 Enfocament

Per assolir els objectius d'aquest projecte disposava del bagatge de coneixements i habilitats aconseguits durant els estudis del Grau, l'experiència en programació obtinguda en diferents vessants i adquirida en els mesos anys d'experiència com a programador, encara que no tenia cap experiència específicament en el disseny i programació de videojocs.

Atenent al temps disponible (aproximadament 3 mesos) per al desenvolupament d'un videojoc plenament funcional, va semblar més coherent i assolible dissenyar un tipus de joc força conegut per a mi com a jugador, perquè permetia disposar, a priori, d'un coneixement de la mecànica i establir una exigència mínima quant als requisits de jugabilitat .

Per tant l'opció va estar prendre com a referència un dels videojocs de plataformes en 2D més coneguts en les darreres dècades i també actualment com és la saga o franquícia del *Super Mario*, tot intentant, donar-li algun tret distintiu en la mecànica a l'hora de jugar i que, per tant, pugues suposar un repte addicional al jugador.

Després d'un període de recerca i de reflexió, malauradament, vaig constatar que aquesta innovació, ara, no era viable. La incorporació de noves funcionalitats i sobretot de mecàniques innovadores i per tant no utilitzades anteriorment en videojocs de plataformes, hagués suposat la necessitat d'invertir gran quantitat de temps únicament adreçat a la recerca i anàlisi de desenes o potser fins i tot de centenars de jocs, per tal de definir un element no emprat a hores d'ara.

Donada la limitació temporal pel desenvolupament d'aquesta opció, va haver de descartar-se, intentant però, incorporar algun element al joc dissenyat que el fes diferent al joc de referència escollit.

Aquest videojoc, com tots els que conformen el gènere de plataformes, consta d'unes mecàniques senzilles i que, de forma més bàsica, es poden exemplificar amb accions simples com: caminar o córrer i saltar o trepar.

L'acció es desenvolupa en un escenari en 2D i mitjançant el desplaçament lateral.

3.2 Recursos

Els recursos utilitzats han estat de dos tipus: externs i interns.

Han estat recursos externs els que fan referència a la tecnologia emprada quant a l'ús de les eines de programació .

Els recursos de tipus intern s'han centrat en la part tècnica, concretament en el desenvolupament del joc, donat que ha estat un projecte dissenyat i desenvolupat de forma unipersonal, amb el suport i l'acompanyament dels professors tutors del Treball.

L'únic recurs físic emprat ha estat l'equipament informàtic personal.

3.3 Eines de programació

Com a eines de programació, per realitzar el procés de desenvolupament s'ha optat pel *framework* LibGdx i Box2D com a motor de físiques 2D, parts fonamentals pel desenvolupament del videojoc, en conjunció amb Android Studio, com a entorn de programació o IDE, i utilitzant en el llenguatge de programació Java.

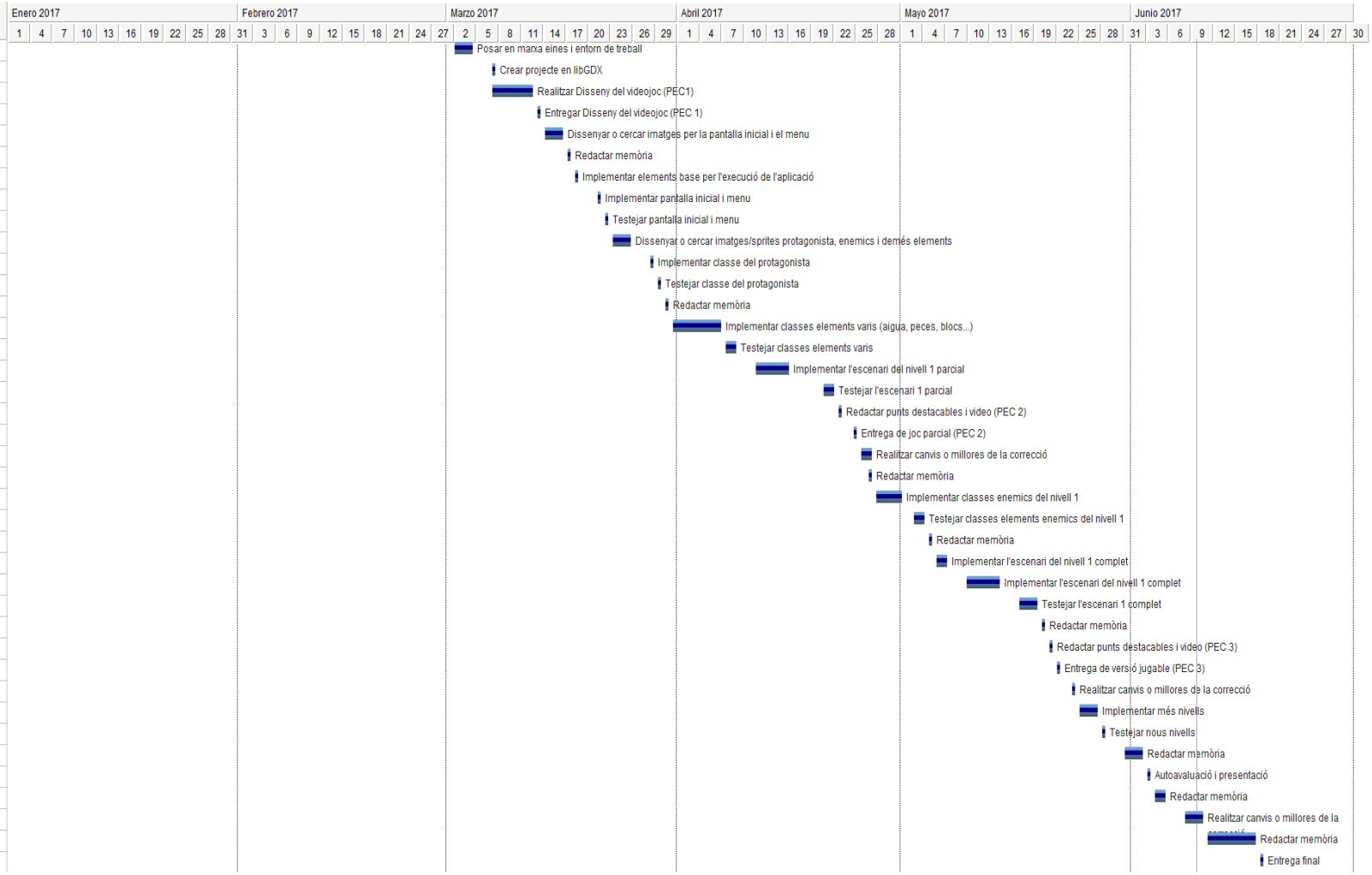
Addicionalment s'han emprat altres programaris com Tiled, per a la creació dels nivells, Gimp, per a la creació o modificació dels elements visuals i Aseprite, per a la creació dels *sprites* animats.

El motiu principal de l'elecció d'aquest *framework* ha estat per a poder realitzar el desenvolupament d'un videojoc utilitzant Java com a llenguatge base, donat que aquest és un els llenguatges amb els que tinc més experiència i per tant amb el que treballo amb més seguretat, i també atenent al fet que, a més, facilita la possibilitat d'exportació a dispositius mòbils.

Així doncs, després d'una cerca i valoració d'eines o *frameworks* amb aquestes premisses, vaig decidir-me pel LibGdx, a més dels motius detallats, també per la gran difusió i acceptació que té.

4 Planificació del Treball

	Nombre	Duración	Inicio	Fin
1	Posar en marxa eines i entorn de treball	3d	01/03/2017	03/03/2017
2	Crear projecte en libGDX	1d	06/03/2017	06/03/2017
3	Realitzar Disseny del videojoc (PEC1)	6d	06/03/2017	11/03/2017
4	Entregar Disseny del videojoc (PEC 1)	1d	12/03/2017	12/03/2017
5	Dissenyar o cercar imatges per la pantalla inicial i el menu	3d	13/03/2017	15/03/2017
6	Redactar memòria	1d	16/03/2017	16/03/2017
7	Implementar elements base per l'execució de l'aplicació	1d	17/03/2017	17/03/2017
8	Implementar pantalla inicial i menu	1d	20/03/2017	20/03/2017
9	Testejar pantalla inicial i menu	1d	21/03/2017	21/03/2017
10	Dissenyar o cercar imatges/sprites protagonista, enemics i demés	3d	22/03/2017	24/03/2017
11	Implementar classe del protagonista	1d	27/03/2017	27/03/2017
12	Testejar classe del protagonista	1d	28/03/2017	28/03/2017
13	Redactar memòria	1d	29/03/2017	29/03/2017
14	Implementar classes elements varis (aigua, peces, blocs...)	7d	30/03/2017	05/04/2017
15	Testejar classes elements varis	2d	06/04/2017	07/04/2017
16	Implementar l'escenari del nivell 1 parcial	5d	10/04/2017	14/04/2017
17	Testejar l'escenari 1 parcial	2d	19/04/2017	20/04/2017
18	Redactar punts destacables i video (PEC 2)	1d	21/04/2017	21/04/2017
19	Entrega de joc parcial (PEC 2)	1d	23/04/2017	23/04/2017
20	Realitzar canvis o millores de la correcció	2d	24/04/2017	25/04/2017
21	Redactar memòria	1d	25/04/2017	25/04/2017
22	Implementar classes enemics del nivell 1	4d	26/04/2017	29/04/2017
23	Testejar classes elements enemics del nivell 1	2d	01/05/2017	02/05/2017
24	Redactar memòria	1d	03/05/2017	03/05/2017
25	Implementar l'escenari del nivell 1 complet	2d	04/05/2017	05/05/2017
26	Implementar l'escenari del nivell 1 complet	5d	08/05/2017	12/05/2017
27	Testejar l'escenari 1 complet	3d	15/05/2017	17/05/2017
28	Redactar memòria	1d	18/05/2017	18/05/2017
29	Redactar punts destacables i video (PEC 3)	1d	19/05/2017	19/05/2017
30	Entrega de versió jugable (PEC 3)	1d	20/05/2017	20/05/2017
31	Realitzar canvis o millores de la correcció	1d	22/05/2017	22/05/2017
32	Implementar més nivells	3d	23/05/2017	25/05/2017
33	Testejar nous nivells	1d	26/05/2017	26/05/2017
34	Redactar memòria	3d	29/05/2017	31/05/2017
35	Autoavaluació i presentació	1d	01/06/2017	01/06/2017
36	Redactar memòria	2d	02/06/2017	03/06/2017
37	Realitzar canvis o millores de la correcció	3d	06/06/2017	08/06/2017
38	Redactar memòria	7d	09/06/2017	15/06/2017
39	Entrega final	1d	16/06/2017	16/06/2017



5 Desenvolupament d'un plataformes 2D

5.1 Origen i evolució del gènere

L'inici del gènere de plataformes es pot situar al 1980 en les màquines *arcade* amb els pioners *Space Panic* i *Donkey Kong* al 1981. L'element diferenciador amb la resta de jocs de l'època va estar la utilització de la gravetat i els salts entre plataformes com el famosíssim *Pac-Man* (no havia gravetat i la perspectiva era zenital).

Però amb tot, va ser amb l'aparició de *Pitfall!* publicat per *Activision* per a la videoconsola *Atari 2600 quan es* va acabar de definir com a gènere.

A causa de la limitació tècnica en les videoconsoles i també a la crisis del videojoc de 1983, els jocs de plataformes que anaven sortint no acabaven de penetrar completament dins de la comunitat de jugadors.

Aquesta realitat va canviar totalment amb l'aparició de *Super Mario Bros* i la videoconsola *NES* de *Nintendo* l'any 1985. Aquest videojoc va suposar una revolució no solament dins el gènere de plataformes sinó també dins de la pròpia industria dels videojocs.

El *Super Mario Bros* va introduir una sèrie d'elements que servirien de base com eren : la recol·lecció d'un nombre d'elements per obtenir-ne un altre o la divisió en subnivells dels propis nivells.

Durant la segona meitat dels anys vuitanta, el regnat de la *NES* i el gènere de les plataformes en 2D va viure els seus millors moments. Les possibilitats d'una videoconsola personal van permetre l'aparició de jocs molt més llargs, donant origen als *jocs de plataformes d'exploració* com *Metroid* (1986) o *Castlevania III* (1989). Aquest subgènere, normalment, tenia una trama i una ambientació més desenvolupada.

Durant la primera meitat dels anys 90, el gènere es va estandarditzar per complet, tant a les consoles de 16 bits (*SNES* i *Sega Mega Drive*) com en les consoles portàtils (*Game Boy* i *Sega Game Gear*). Dins dels 16 bits van

sortir grans títols com *Sonic the Hedgehog* per part de *Mega Drive* i *Super Mario World* per part de la *SNES.*

La darrera gran revolució, dins les plataformes de 16 bits, va donar-se amb el llançament de *Donkey Kong Country* de *Rare* que va utilitzar un sistema avançat de pre-renderitzat de gràfics que utilitzava tot el potencial de la *SNES.*

Amb l'aparició del 3D i la nova generació de consoles, es va experimentar la conversió del gènere. Els primers títols, com *Clockwork Knight* per *Sega Saturn,* presentaven uns *gràfics tridimensionals* però amb un *esquema de joc lineal* similar als plataformes de tota la vida. A aquests jocs se'ls considera 2.5D.

Amb l'aparició de *Super Mario 64* es va aconseguir la fórmula correcta per adaptar el gènere a les tres dimensions.

Super Mario 64 va introduir escenaris completament tridimensionals on la llibertat i la quantitat de moviments per l'escenari aconseguien traslladar la jugabilitat, característica essencial dels plataformes, a les 3D.

L'aparició de les noves consoles de 32 bits, amb alta capacitat d'emmagatzemament dins un *CD-ROM*, va permetre realitzar grans jocs de plataformes en 2D que van revitalitzar el gènere com: *Rayman*, *Castlevania: Symphony of the Night* o *Abe's Oddysee* que va introduir una innovadora *intel·ligència artificial* als enemics.

També van aparèixer jocs molt bons en 2.5D que aprofitaven les capacitats tècniques de la nova generació de consoles mantenint un estil de joc més clàssic com: *Crash Bandicoot* o *Klonoa* per *PlayStation* o *Nights into Dreams* per *Sega Saturn.*

En la dècada dels 2000 s'han continuat les formules establertes a les generacions dels 32 i 64 bits quan als jocs 3D i 2.5D. En aquesta època destaca el que, possiblement, ha estat el millor joc de *Sonic* de tots els temps, el *Sonic Adventure 2* per a la consola *Dreamcast* on posteriorment es llançaria per *GameCube.*

L'aparició de *Game Boy Advance* durant la primera meitat de la dècada també va aportar un nou públic jove als jocs en 2D, sortint al mercat nombrosos títols

que continuaven l'estil de mercat de l'època daurada dels 16 bits, amb un toc retro, així com reedicions de jocs clàssics de *NES* i *SNES*.

Cal fer menció, en aquest ressorgiment dels jocs de plataformes clàssics, del *Cave Story* (2004) un joc *freeware* per a *PC* que combina elements de plataformes clàssics, d'exploració i *arcade,* que malgrat que té una estètica de 16 bits, té un acabat tècnic actual i que per tant podria ser considerada com una obra mestra del gènere.

Per a les videoconsoles *Playstation 2* es van crear moltes noves franquícies com *Jak and Daxter* desenvolupada per *Naughty Dog*. Aquest joc va revolucionar la industria per ser el primer joc en 3D sense incloure temps de càrrega, utilitzant el llenguatge *GOAL*, el qual ofereix un món 3D més ampli i fluït.

Amb les noves interfícies de les consoles de *Nintendo* és van experimentar noves formules per a la jugabilitat, fent us de la pantalla tàctil de la *Nintendo DS* o del comandament de la *Wii*. Alguns dels jocs que aprofitaven aquestes noves formules han estat: *Super Mario 64 DS, Super Mario Galaxy, Super Monkey Ball* i *Rayman*.

A principis de la dècada actual, es va produir un augment de plataformes en 2D de tal envergadura, que ha aconseguit convertir les plataformes 3D en un subgènere minoritari.

Moltes franquícies que anys abans havien fet el salt al 3D, van tornar al 2D perquè redueixen els costos de desenvolupament, tenen la possibilitat d'oferir un apartat artístic més elaborat i per l'eliminació dels problemes de càmera que molts jocs de plataformes en 3D presentaven i que no s'aconseguien solucionar.

Alguns exemples de jocs que van tornar al 2D son: *The Impossible Game*, *Rayman Origins*, *Donkey Kong Country Returns* i *Sonic Generations*.

Per una altra banda, sobretot en les desenvolupadores independents, també es van crear plataformes 2D completament nous com: *Super Meat Boy* i *Fez.*

Resumint, és evident que avui al 2017, els videojocs del gènere de plataformes en 2D tenen plena vigència.

5.2 Benny The last hope

S'ha creat aquest videojoc de plataformes que incorpora les mecàniques jugables bàsiques típiques de la franquícia *Super Mario*. A diferència d'aquests, l'ambientació és futurista i el protagonista és Benny, un astronauta .

Aquest videojoc, com tots els que conformen el gènere de plataformes, consta d'unes mecàniques senzilles i que, de forma més bàsica, es poden exemplificar amb accions simples com són caminar o córrer i saltar o trepar.

L'acció es desenvolupa en un escenari en 2D i mitjançant el desplaçament lateral. Es basa en l'avanç d'un personatge, en Benny, pels diferents nivells, que són dos en aquest cas, mentre esquiva i/o supera obstacles, com desnivells i forats o els derrota, en aquest cas enemics, mentre busca, troba i recull elements de premi com són vides, aigua i oxigen.

L'objectiu del joc consisteix en que el personatge partint d'un punt A (Inici) arribi a un punt B (Final) en cada un dels nivells, que són 2 en aquest joc, i obtenir la màxima puntuació (en aquest cas, aigua).

L'avanç de nivell implica fer el desplaçament amb un grau de dificultat més alt (menys oxigen).

En cada nivell el joc finalitza quan Benny arriba a la seva nau espacial.

El videojoc incorpora elements o recursos addicionals com és la possibilitat de allargar el temps disponible per superar un nivell agafant bombones d'oxigen. La necessitat d'agafar aquests objectes estarà en funció del nivell on es juga o de l'expertesa i/o de l' habilitat del jugador.

A més, s'ha afegit la possibilitat d'executar el joc en dispositius Android amb controls adaptats, utilitzant botons en lloc del teclat que s'utilitza en la versió d'ordinador. Aquesta opció pot activar-se o desactivar-se des del Menú (Opcions).

5.3 Controls

Els controls que permeten jugar i també controlar altres elements, són els següents:

- Control de moviment:
    - Desplaçament lateral:
        - **A**: moviment del personatge cap a l'esquerra.
        - **D**: moviment del personatge cap a la dreta.
    - Salt:
        - **W** o ***Space***: salt del personatge.
- Control del so:
    - **M**: silencia tots els sons provinents del joc. Pot utilitzar-se des de múltiples pantalles.
- Altres controls:
    - **Esc**: envia al jugador al menú. Només funciona a la pantalla principal.
    - **K**: habilita el mode *debug* on es visualitzarà el *render* per visualitzar els *bodies* dels diferents elements. Sol funciona a la pantalla principal. El seu ús és per motius de *test* i *debug*.
    - **Fletxa cap amunt**: supera el nivell de forma automàtica. Sol funciona a la pantalla principal. El seu ús és per motius de *test* i *debug*.

- Controls extra (només per a la versió per Android):
    - Botó de desplaçament cap a l'esquerra
    - Botó de desplaçament cap a la dreta
    - Botó de salt
    - Botó per sortir al menú

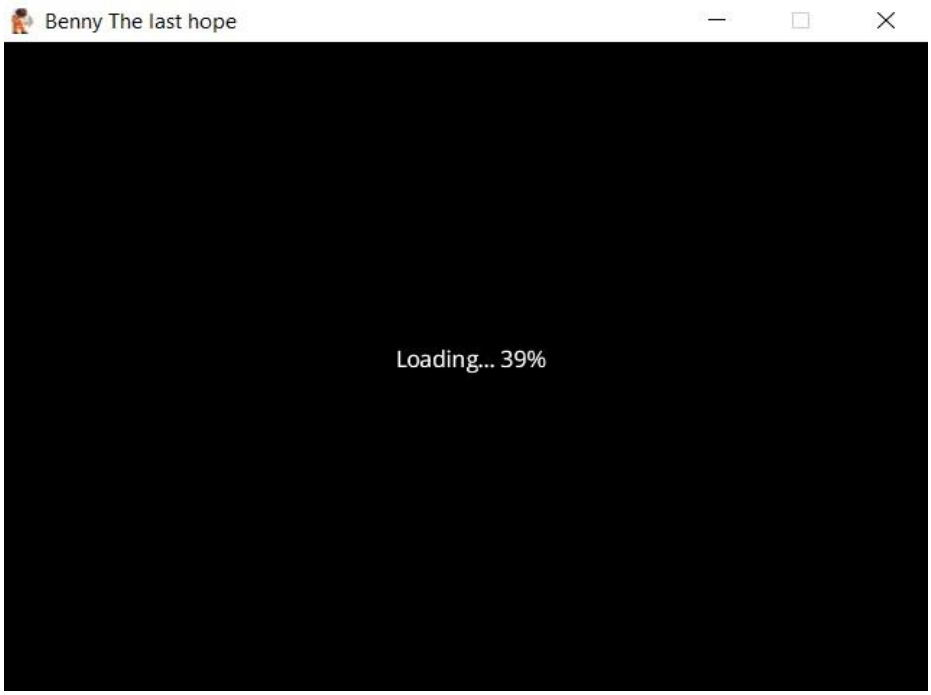
5.4 Estructura del joc

El joc està estructurat per diverses pantalles:

- Pantalla de càrrega
- Pantalla de menú
- Pantalla d'opcions
- Pantalla amb la història
- Pantalla de selecció de nivells
- Pantalla principal
- Pantalla de nivell superat
- Pantalla de mort del personatge
- Pantalla de crèdits

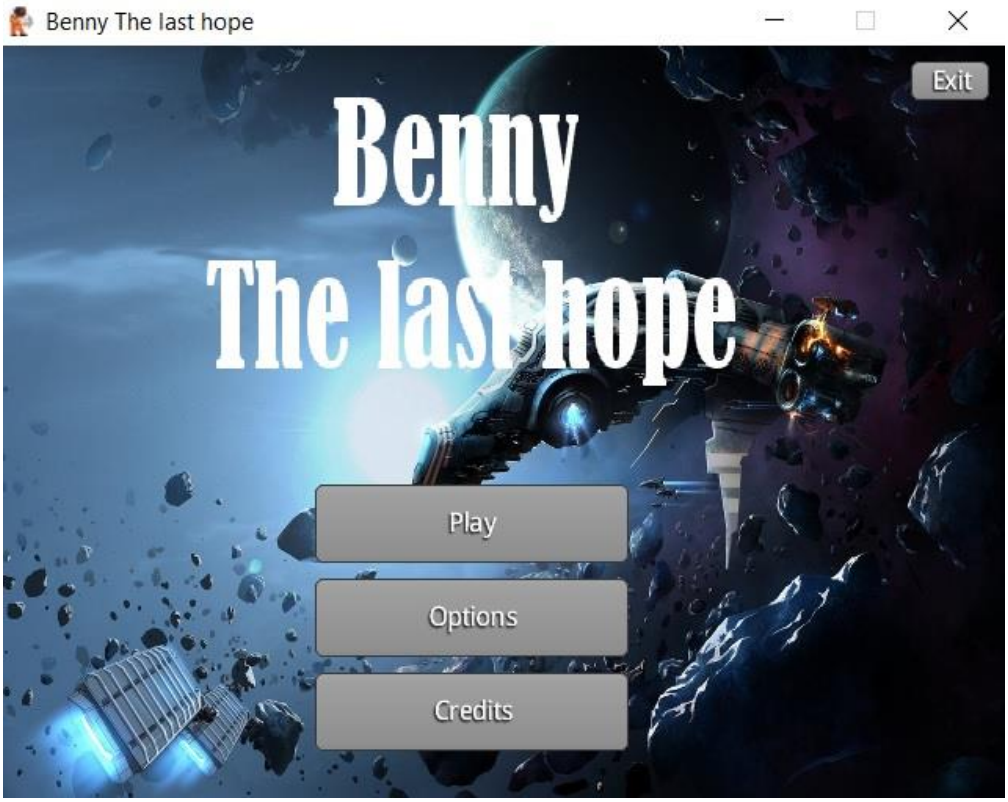
5.4.1 Pantalla de càrrega

En aquesta pantalla es mostra, en forma de percentatge, l'estat de càrrega dels elements necessaris pel joc. El motiu principal d'aquesta pantalla es poder tenir carregats la majoria de recursos un cop el jugador està en el menú per tal d'evitar temps d'espera dins del joc.



5.4.2 Pantalla de menú

En aquesta pantalla es mostra el menú principal del joc amb una sèrie de botons.

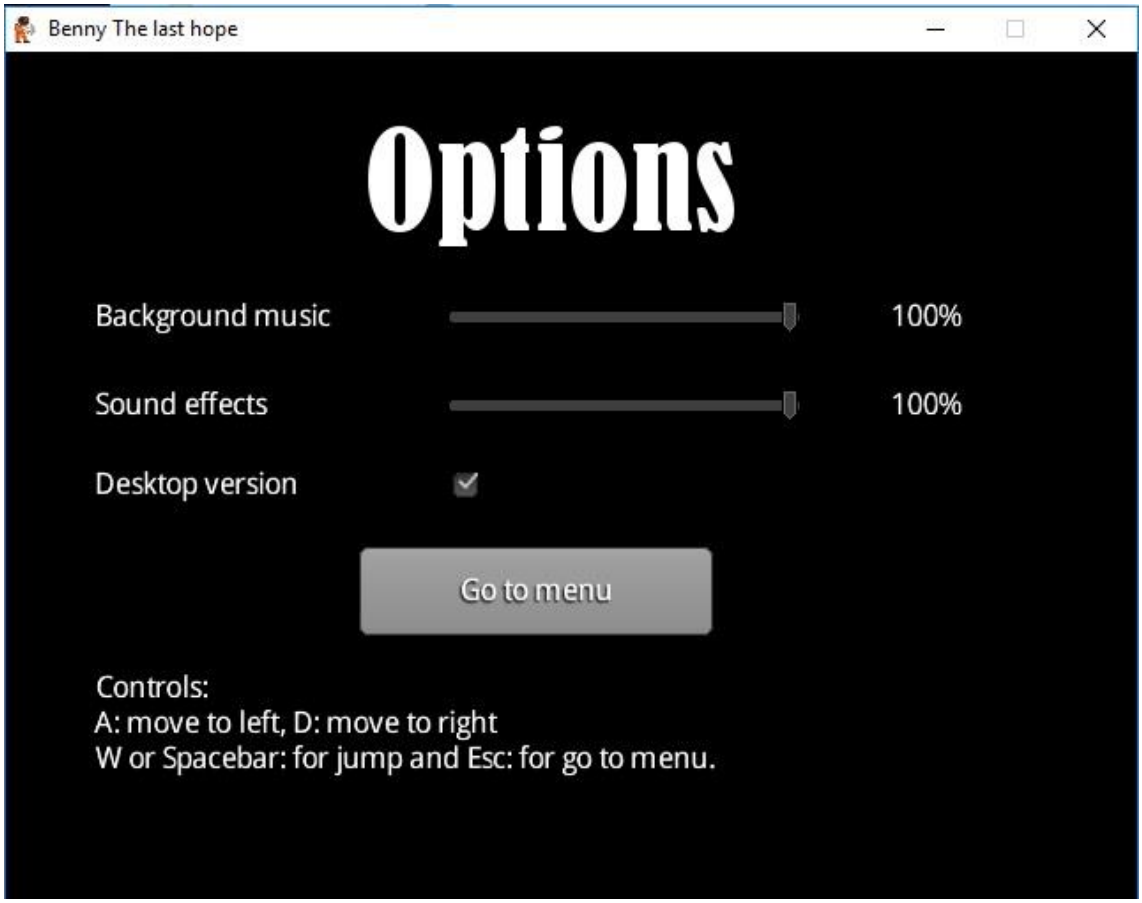


Els botons que apareixen són els següents:

- **Play**: botó que ens portarà pantalla d'història del joc.
- **Options**: botó que ens portarà a la pantalla d'opcions del joc.
- **Credits**: botó que ens portarà a la pantalla de crèdits del joc.
- **Exit**: botó que ens farà sortir del joc.

5.4.1 Pantalla d'opcions

En aquesta pantalla es mostren les diferents opcions del joc i una breu descripció dels controls.

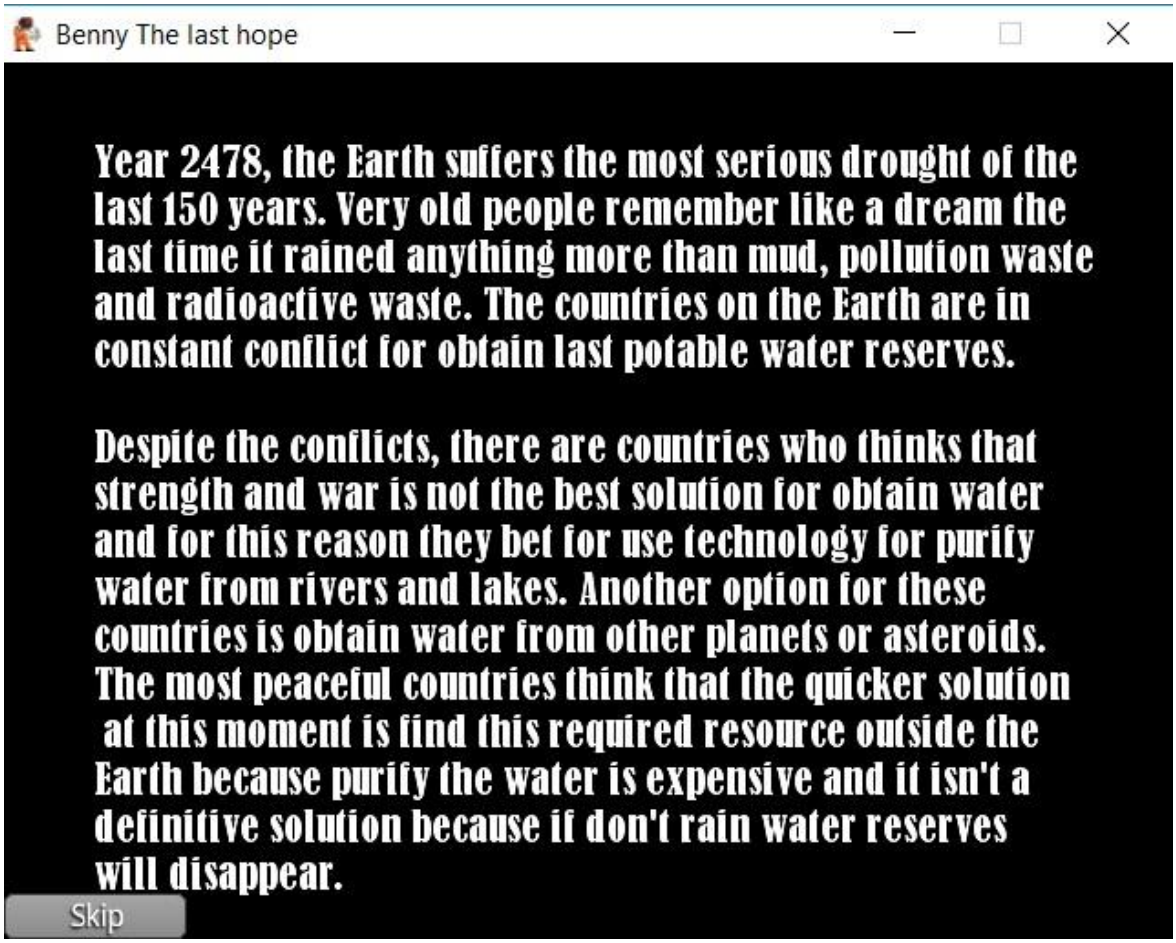


Els botons que apareixen són els següents:

- **Go to menu**: botó que ens portarà al menú.

5.4.2 Pantalla amb la història

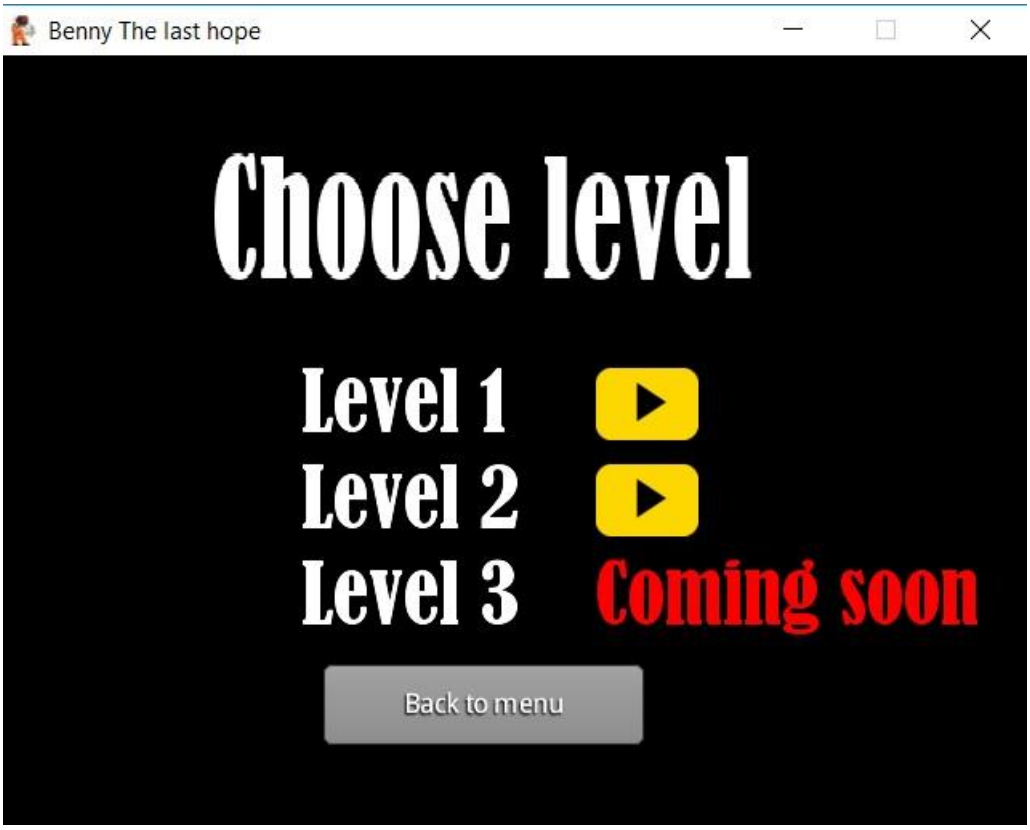
En aquesta pantalla es mostra la història o trama del joc, és a dir, posa al jugador en situació sobre els motius que impulsen al personatge per dur a terme la seva aventura.



En aquesta pantalla, donada la llargada de la història, hi ha un botó, a la part inferior esquerra, que permet passar a la següent pantalla de selecció de nivells sense haver d'esperar a que es visualitzi tot el text.

5.4.3 Selector de nivells

En aquesta pantalla es mostra les opcions que te el jugador per poder accedir als diferents nivells del joc, opcions que s'activent pressionant els botons grocs.



En l'estat actual del desenvolupament, el joc únicament consta de dos nivells, el primer a mode d'iniciació, perquè el jugador disposa amplament de temps per superar el nivell donada la gran quantitat d'oxigen que té al seu abast.

En el segon nivell en canvi, l'oxigen disponible es molt més escàs i per tant es fa indispensable recollir la major quantitat possible de bombones.

Per últim, donada la limitació temporal pel desenvolupament del joc, no s'ha pogut incloure un tercer nivell.

A més, a la part de sota i en el centre, hi ha un botó que permet anar al menú principal.

5.4.4 Pantalla principal

En aquesta pantalla es mostra l'escenari on es desenvoluparà el joc.

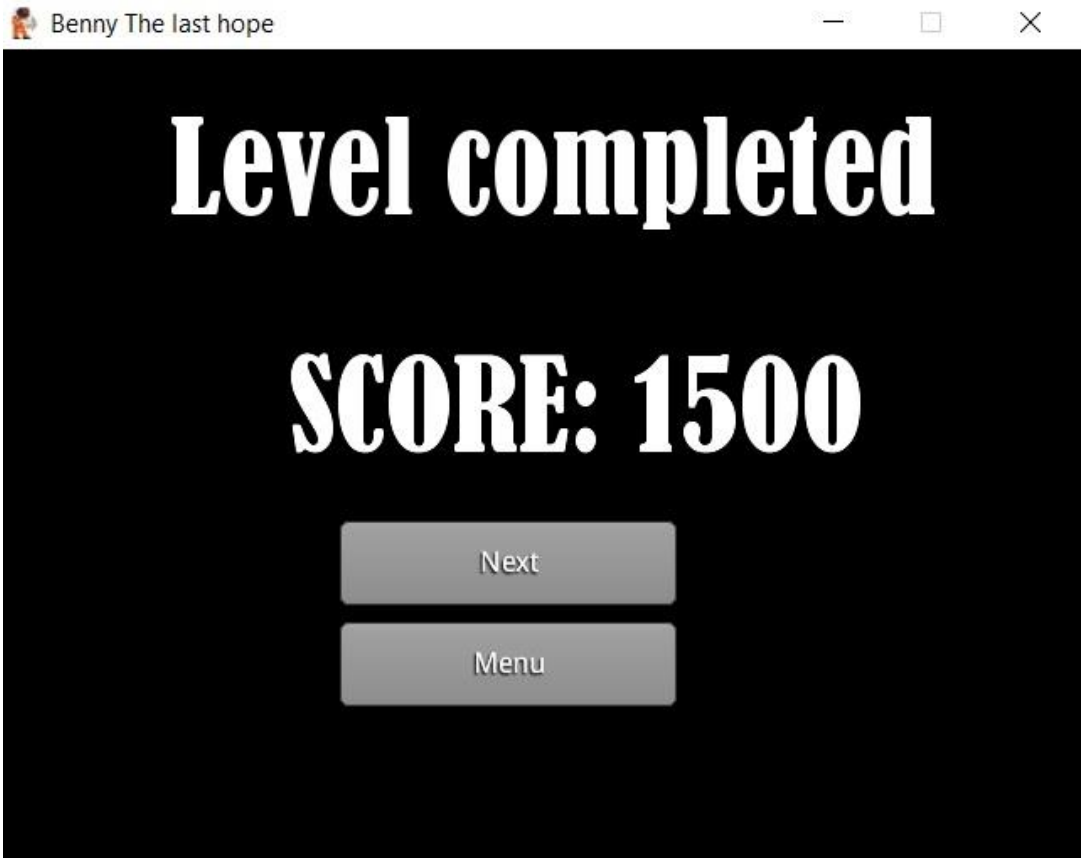
La interpretació dels diferents objectes que conté aquesta pantalla, són els següents:

1. **Vides o *Lifes***: els astronautes situats a la part superior esquerra de la pantalla representen les vides disponibles que té el jugador.
2. **Oxigen o *Oxygen*** (comptador): representa el temps d'oxigen disponible per completar la missió i està situat en la part superior central.
3. **Nivell o *Level***: representa el nivell on el jugador esta actualment i està ubicat en la part central superior, sota del nivell d'oxigen.

4. **Aigua o *Water*** (comptador): representa la quantitat d'aigua obtinguda pel jugador i està situat en la part superior dreta de la pantalla.
5. **Caixa o *Box***: representa una caixa a la que el jugador colpeja per sota amb la finalitat d'obtenir aigua o vides. Un cop colpejada, la caixa canvia de textura per indicar al jugador que ja és buida.
6. **Caixa buida o *Empty***: representa una caixa que el jugador ja ha colpejat per sota
7. **Plataforma o *Platform***: representa una superfície mòbil que el jugador pot utilitzar per desplaçar-se pel mapa.
8. **Oxigen o *Oxygen*** (objecte): representa l'objecte de la bombona d'oxigen que el jugador pot agafar.
9. **Aigua o *Water*** (objecte): representa l'objecte de l'aigua que el jugador pot agafar.
10. **Bloc destructible o *Block***: representa un tipus de bloc destructible pel jugador, si el colpeja per sota.
11. **Benny** (protagonista): representa el personatge controlat pel jugador.
12. **Enemic o *Enemy***: mena de robot que representa un enemic. Si toca al nostre personatge, el matarà i el jugador perdrà una vida. Pot ser eliminat si se li salta a sobre.
13. **Bloc no destructible o *Ground***: representa un tipus de bloc no destructible pel jugador, pot servir d'obstacle o utilitzar-se com a sòl.
14. **Punxes o *Spikes***: representen un obstacle pel jugador donat que si aquest les toca, mataran al personatge i li faran perdre una vida al jugador.
15. **Terra o *Ground***: representa un tipus de bloc no destructible pel jugador, pot servir d'obstacle o o utilitzar-se com a sòl.

5.4.5 Pantalla de nivell superat

En aquesta pantalla es mostra que el jugador ha aconseguit completar el nivell amb una puntuació determinada.



A més de la puntuació obtinguda en el nivell que s'ha superat, a sota també apareixen dos botons:

- **Next:** botó per avançar al següent nivell
- ***Menu***: botó per tornar al menú

5.4.6 Pantalla de mort del personatge

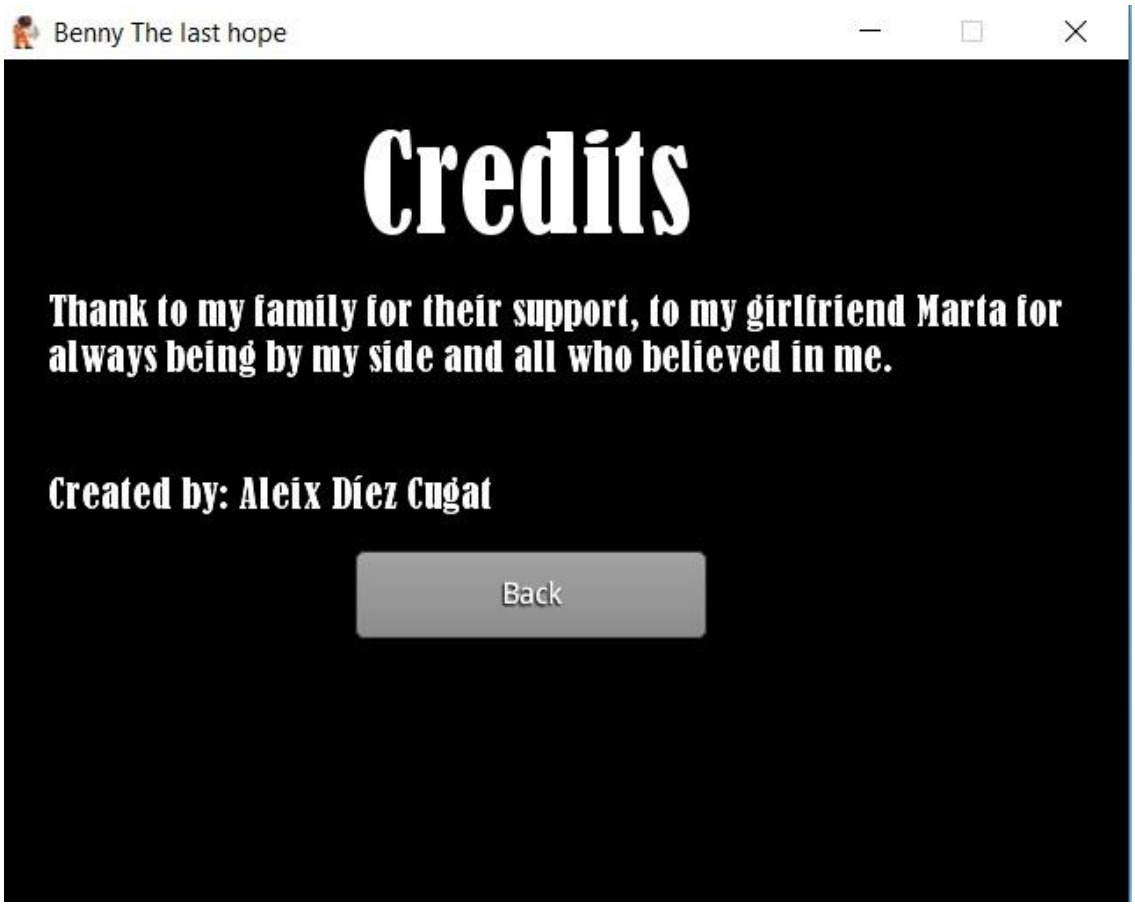
En aquesta pantalla es mostra que ha finalitzat el joc perquè el jugador ha perdut totes les vides.

A més de la puntuació obtinguda en el nivell, també es visualitzen dos botons:

- ***Retry***: botó per tornar a començar el nivell
- ***Menu***: botó per tornar al menú

5.4.7 Pantalla de crèdits

En aquesta pantalla es mostren els crèdits del joc



També és visualitzen els següents botons:

- ***Back***: botó per tornar al menú

5.5 Diari de desenvolupament

Les eines de programari que s'han emprat pel desenvolupament del videojoc han estat:

- LibGdx com a *framework* de desenvolupament.
- Box2D com a motor de físiques 2D.
- Android Studio com a entorn de programació.
- Gimp 2 com a editor de textures i altres elements visuals.
- Aseprite v1.1.1 com editor per a la creació d'*sprites* i animacions.
- Tiled com a editor de nivells en 2D.

El codi font del videojoc s'ha programat en llenguatge Java.

En primer lloc, com en qualsevol tipus de projecte, es va realitzar el disseny del joc que es voldria desenvolupar. Aquest pas resulta especialment important per tal de tenir clar els elements que composaran el producte final i evita desenvolupar les parts principals sobre la marxa.

Així doncs, com si d'una pel·lícula es tractes, aquest document permet crear un guió mitjançant el qual queden definides les parts fonamentals que descriuran el desenvolupament del joc i per tant permetran, d'entrada, no oblidar cap part fonamental. Per suposat, aquest document no representa una directriu inamovible sinó que, a partir d'aquest i a mesura que avança el desenvolupament, es poden ampliar els diferents elements descrits per tal d'enriquir els diferents conceptes que hi apareixen.

A continuació, un cop clares les idees principals, es va procedir a realitzar una cerca dels diferents elements necessaris per a les pantalles inicial o de menú. Aquesta cerca tenia com a objectiu trobar elements gràfics, de lliure utilització, que permetessin obtenir el resultat visual pensat i així accelerar el procés de desenvolupament al no tenir que invertir tant temps com seria necessari en cas de tenir que dissenyar i realitzar aquests elements.

Després de visitar desenes de pàgines on hi havia contingut de diverses tipologies es va decidir que pel cas de la pantalla inicial o menú primer es provaria el resultat visual que podia aportar els propis elements del *framework.*

Tot i aquesta cerca infructuosa, si que es va aprofitar per recollir les referencies de les pàgines on més contingut podria haver pels models dels diferents elements que apareixerien en la pantalla de joc.

També es va procedir a desenvolupar la base del que seria la pantalla principal. Si bé aquesta part quan es va pensar en la fase de disseny semblava bastant senzilla, no ho va ser, donada la naturalesa del *framework* ja que, com sol passar quan s'utilitza una eina nova, cal aprendre la manera de treballar-hi i, de vegades, també cal *canviar el xip* a l'hora de fer les coses.

Va ser en aquest punt on va ser necessari l'aprenentatge de nous conceptes dins de la programació, però en aquest cas, de videojocs. Així doncs ja no solament es va tractar d'utilitzar una classe enlloc d'una altra com bé pot passar quan utilitzem alguna llibreria, que sobreescriu les funcionalitats d'una classe pròpia de Java, sinó que va caldre analitzar i assolir la manera de com el *framework* organitzava i dirigia tots els elements del desenvolupament. Emprar les paraules com "organitzar" o "dirigir" no és casual ja que, si bé pot semblar que són més pròpies de la creació d'una pel·lícula o d'una obra de teatre, quan es tracta del desenvolupament d'un videojoc, també tenen un sentit i a continuació s'explicarà el perquè.

Quan parlem d'un videojoc en 2D amb físiques, conceptes com: *Game, Scene/Screen, Stage*, *Camera*, *World* o *Actor* són indispensables ja que la forma de composar el joc tal com s'ha dit amb anterioritat, a l'hora de desenvolupar un videojoc una forma molt bona visualitzar-lo és com si es tractés d'una pel·lícula que el programador dirigeix.

Així doncs podem establir una sèrie de relacions lògiques dins d'aquests conceptes prenent la idea del joc com si fos una pel·lícula i que facilitaran la comprensió de com esta realitzat el joc seguint aquesta lògica, i aquests com són:

- Un joc conté:
    - Un conjunt de pantalles
    - Un món (si conté físiques)
- Una pantalla conté:
    - Un escenari
- Un món conté:
    - Unes físiques
- Un escenari conté:
    - Una càmera
    - Uns actors
- Una càmera conté:
    - Un espai de visualització
- Un actor conté:
    - Unes accions que pot realitzar

Un cop enteses i interioritzades aquestes relacions i el seu funcionament pràctic, va ser possible la creació dels elements base. Amb això es vol dir que es van crear les classes i relacions necessàries per disposar de tantes pantalles com es consideressin adients.

Primer es va crear una classe abstracta anomenada *BaseScreen,* la funció de la qual era de servir base per a les altres futures pantalles. En aquesta classe hi havia definits, però no implementats, els diferents mètodes que calien per a gestionar una *Screen*. La raó principal per a crear aquesta classe va ser la de fer que les noves pantalles disposessin d'un codi més net i per tant únicament caldria implementar els mètodes imprescindibles.

Seguidament, es va crear una altra classe, que seria la classe principal, que estendria de *Game* ja que contindria totes les instancies de les diferents pantalles i els bits emprats per a la detecció de col·lisions explicades més endavant. A més, també es carregarien en l'*AssetManager* la majoria de textures, sons, musiques i imatges emprades dins del joc i que per millorar l'experiència del jugador es precarregarien abans de visualitzar la pantalla de menú. La pantalla que es visibilitzaria fins que tots els recursos estiguin

carregats seria la de *LoadingScreen* que informa a l'usuari de l'estat de càrrega de recursos (en percentatge).

A continuació es va procedir amb la implementació de la pantalla del menú que consistiria en un fons negre, que més endavant va estar substituïda per una imatge, tres botons, un títol en forma d'imatge i una música de fons. Tenint en compte la forma de treballar dins d'aquest *framework* es van afegir els elements que es volien visualitzar al *Stage* i es van crear els esdeveniments relacionats amb els botons. La sorpresa, però, va ser que no es van donar els resultats esperats.

Després d'una recerca extensa per la documentació es va trobar la solució: una instrucció que s'encarrega d'establir el focus a l'hora de capturar esdeveniments de teclat i ratolí. Un cop solventat aquest problema es va poder crear i implementar les classes per a les pantalla de opcions i crèdits sense major dificultat, perquè aquestes havien de ser visualitzades en el moment que l'usuari premés el botó corresponent en el menú.

Tot seguit, es va procedir a buscar elements gràfics per a la que seria la pantalla principal del joc, això incloïa des del personatge principal, els enemics, els elements dels nivells de joc, etc. Es van poder recopilar força recursos que si bé, la majoria no van ser utilitzats de forma directa en el joc, si van servir d'inspiració en la millora de la idea del que es volia aconseguir. Per aquesta raó, alguns elements com blocs, aigua, oxigen, es van dibuixar emprant l'editor d'imatges. Cal deixar clar que donada la dificultat i el temps necessari per a la realització de les animacions, aquestes no es van incloure fins més endavant.

Un cop es van recopilar els recursos de la pantalla principal, resultants de la cerca o el disseny, es va procedir a la implementació d'una pantalla principal amb els elements indispensables per poder provar posteriorment la classe *PlayerEntity* a mesura que s'implementava.

En la classe *PlayerEntity,* que estén d'*Actor,* s'implementen els mètodes imprescindibles, propis d'un *Actor,* com són: *act* i *draw* encarregats d'executar-se en la fase de actualització i dibuixat de l'*Stage* respectivament.

Per una banda, es va definir dins del constructor d'aquesta classe els *bodies* que composaran el personatge i també el tipus: *Dynamic*, les *fixtures*, el bit amb el que ens referirem a aquest tipus d'objecte pel tractament de col·lisions, la mascara de bits que ha de servir per saber amb quins elements ha de col·lidir i el tamany.

Per una altra banda, dins del mètode *draw* es va definir la posició per dibuixar el cos del objecte i la seva textura, i dins del mètode *act* els comportaments de l'objecte tals com: la velocitat i la direcció al prémer un botó. A mesura que va avançar el desenvolupament es van anar afegint condicions pel desplaçament del jugador, per exemple: el jugador és viu.

Un cop la classe ja va estar implementada, es procedí a provar-la en la pantalla inicial per tal de verificar el comportament, però de forma similar a quan es va crear el menú per primer cop, els resultats no van estar els esperats.

Davant d'aquest problema cal tornar a cercar solucions a partir de l'anàlisi de la documentació del *framework*. A partir d'aquí, es visualitza que el problema podria estar relacionat amb el tractament de les col·lisions, així doncs cal trobar la manera de gestionar-les dins de la pantalla principal creant una nova classe que implementa la classe *ContactListener*.

Aquesta nova classe permet capturar els efectes que esdevenen quan dos objectes col·lecten, i no sols això, sinó que permet capturar quan comença i acaba la col·lisió i també obtenir informació addicional mentre s'està duent a terme aquesta: la velocitat al moment de la col·lisió i la de després. Finalment, un cop va quedar implementada aquesta nova classe, es torna a provar *EntityPlayer* i aquest cop, funciona sense cap problema.

A continuació es va procedir a crear i implementar les classes de gairebé tots els objectes inclosos en el joc : *EnemyEntity, PlatformEntity*, *WaterEntity*, *OxygenEntity*, *BlockEntity*, *GroundEntity*, i *SpikeEntity*.

La classe *EnemyEntity* és força similar a *PlayerEntity* donat que la principal diferencia rau en que els enemics tenen un desplaçament constant mentre

col·lideixin amb el terra o amb un bloc si és per la part de sobre d'aquest mentre que canvien de direcció si col·lideixen lateralment.

Les classes *WaterEntity*, *OxygenEntity*, *SpikeEntity, BlockEntity* i *GroundEntity* presenten una diferencia força important i és que no es mouen a causa de que tenen *bodies* de tipus *Static*.

Per últim, la classe *PlatformEntity* també és diferent perquè el *body* és del tipus *Kinematic* i a més, el desplaçament, tant la direcció com la distancia, esta fixat inicialment, al crear el objecte.

Per raons tècniques es decideix incloure les plataformes plenament funcionals i les punxes més endavant.

Tota vegada les classes dels objectes que havien de composar el joc van estar implementades, es va procedir a la construcció parcial del nivell. Per això es va decidir emprar l'eina *Tiled* que permet la construcció de nivells en 2D de forma força intuïtiva, encara que per a que el mapa generat en format TXM pugui ser utilitzat, cal realitzar certes adequacions dins la classe de la pantalla principal, com són la definició d'un conjunt d'objectes addicionals de les classes:

- *OrthogonalTiledMapRenderer*
- *TmxMapLoader*
- *TiledMap*

A més, cal recollir els objectes creats a l'editor dins de recipients, com per exemple llistes o *arrays*, per tal de poder tractar-los dins del codi.

Quan el mapa ja va estar creat i els seus objectes tractats correctament dins el codi, es va procedir a realitzar proves per tal de verificar el comportament amb tots els elements. Durant aquesta etapa de proves s'observa alguns problemes menors que són corregits sense dificultats, ajustant valors dins les classes *PlayerEntity, EnemyEntity* i *PlatformEntity.*

Seguidament, es començà amb la creació de l'anomenat *HUD*, que és l'espai on apareix informació: les vides, l'oxigen restant i l'agua recollida. La construcció d'aquest es va realitzar utilitzant un objecte de la classe *Timer* per tenir en compte el temps que va transcorrent, diversos objectes de la classe

Label pels valors a mostrar, un objecte de la classe *SpriteBatch* per a pintar aquests valors en pantalla i diverses propietats del tipus *int* de la classe *GameScreen* per tal de poder gestionar els valors quan s'agafen.

Un cop implementats aquests elements es proven per tal de verificar el correcte funcionament.

Després de rebre el *feedback* dels professors tutors, es decideix afegir una sèrie de canvis i millores.

El primer d'aquests canvis pretén millorar la forma com es carreguen els nivells de joc. Es decideix, doncs, crear una nova classe *Lvl* que encapsularà tota la lògica de crear el nivell i així millorarà la llegibilitat del codi de forma notòria i d'aquesta forma la creació d'un nivell dins la classe *GameScreen* es reduirà a la creació d'un objecte de la classe *Lvl*.

Seguidament es resolen els problemes tècnics detectats amb les classe *PlatformEntity* i *SpikeEntity* de forma que ja poden ser inclosos en els nivells.

A continuació, es realitzen canvis visuals en la tipografia de les pantalles: menú, opcions, crèdits, nivell completat i personatge mort. També s'augmenta el tamany de la finestra quan es troba en la pantalla de joc i es canvia la tipologia de lletra i tamany *HUD* per millorar-ne la llegibilitat.

Un cop realitzats aquests canvis es va procedir a crear nous *Sprites* emprant l'editor d'imatges per poder crear els blocs que es trenquen i les caixes, plena i buida, que el personatge colpeja per obtenir aigua o objectes. Quan es va disposar dels elements visuals, es va procedir a la implementació d'aquests tipus de blocs especials.

Donat que l'aspecte visual ,tant dels blocs destructibles com de les caixes era definit a l'editor de nivells, calia fer la implementació d'una forma especial. Aquesta consistiria en crear un mètode que s'executés quan es detectes la col·lisió del cap del jugador amb el bloc, aquest mètode contindria el codi necessari per obtenir la cel.la del mapa creat amb l'editor que ocupa la posició del bloc colpejat.

Per una banda i quan es tracta del bloc destructible, un cop obtinguda la posició de la cel.la s'assigna a *null* i es canvia el bit de col·lisió del bloc colpejat a destruït. D'aquesta manera s'aconsegueix que quan ha estat colpejat, el bloc no es torni a dibuixar i que, per tant, cap element pugui xocar-hi.

Per l'altra banda, quan es tracta d'una caixa, el que es calia fer era canviar el tipus de cel.la a visualitzar en el moment que es colpejada, però aquest cop sense assignar el bit de destruït.

D'aquesta manera es van poder aconseguir els efectes desitjats de que es destrueixi un bloc i que la caixa queda oberta, sumant aigua al jugador. A més, s'inclouen sons quan es produïen aquests efectes.

A continuació es van realitzar les animacions del personatge i dels enemics emprant l'editor d'animacions. La part del personatge controlat pel jugador va constar de quatre animacions en total, en estat *idle* i *walking* mirant cap a la dreta i cap a l'esquerra. L'animació *idle* consta de dos fotogrames i la *walking* de si fotogrames. La part dels enemics, en canvi, consta solament de l'estat *walking* i esta conformat per quatre fotogrames.

Una vegada creades les animacions es van incloure en el joc substituint les antigues.

Arribats a aquest punt, es va decidir finalitzar el disseny i creació del primer nivell de joc per poder posar en practica tots els elements i realitzar millores més endavant.

Després de rebre el *feedback* dels professors es decideix afegir una sèrie de canvis i millores.

La primera d'aquestes millores és la creació de la pantalla de selecció del nivell. Aquesta estarà constituïda amb textos i botons i la seva funció serà la de deixar elegir el nivell al jugador donant així un aspecte més polit.

A continuació, es decideix crear una nova classe *LifeEntity* que permetrà crear objectes que el jugador podrà recollir per obtenir vides addicionals. La implementació d'aquesta classe és molt similar a la de *EnemyEntity* amb la variant que el comportament quan el jugador impacti amb l'objecte, serà sumar una vida i no perdre-la i tornar a començar el nivell. Es determina que l'objecte

es crearà en el moment que el jugador colpegi unes caixes en concret. Per tant, que una caixa proporcioni aigua o una vida al jugador dependrà d'un atribut establert a l'hora de crear els objectes a l'editor, d'aquesta manera dins el codi sol caldrà llegir el tipus de caixa que es i aquesta oferirà un element o un altre.

Donat que es disposava d'un selector de nivells, se'n va dissenyar un de nou però posant especial èmfasis en l'oxigen disponible del personatge, per aquesta raó aquest nou nivell requereix recollir la majoria, sinó totes, les bombones d'oxigen. La raó principal de la creació d'aquest nou nivell es perquè en el primer nivell, de tipus introductori, aquest element queda en un segon terme atenent a la quantitat d'oxigen inicial.

També es decideix incloure una pantalla en el moment que el jugador premi *Play* en el menú, on es mostrarà la història del joc per introduir al jugador en la situació del protagonista. Aquesta pantalla la conforma un text que es desplaça i un botó per passar a la pantalla de selecció de nivell.

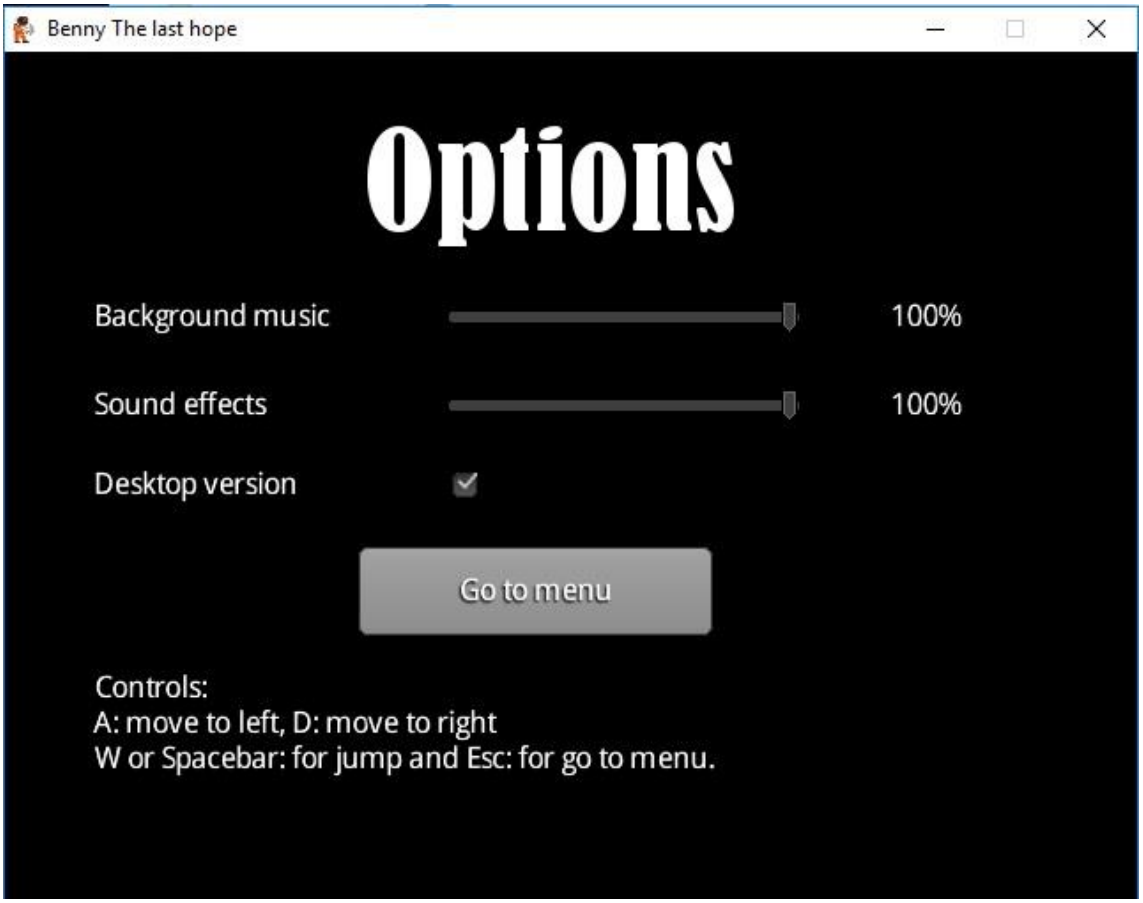
Per últim, s'inclouen adaptacions perquè el joc pugui ser utilitzat en dispositius mòbils. Aquestes adaptacions s'explicaran en el següent apartat.

5.6 Adaptació a dispositius mòbils

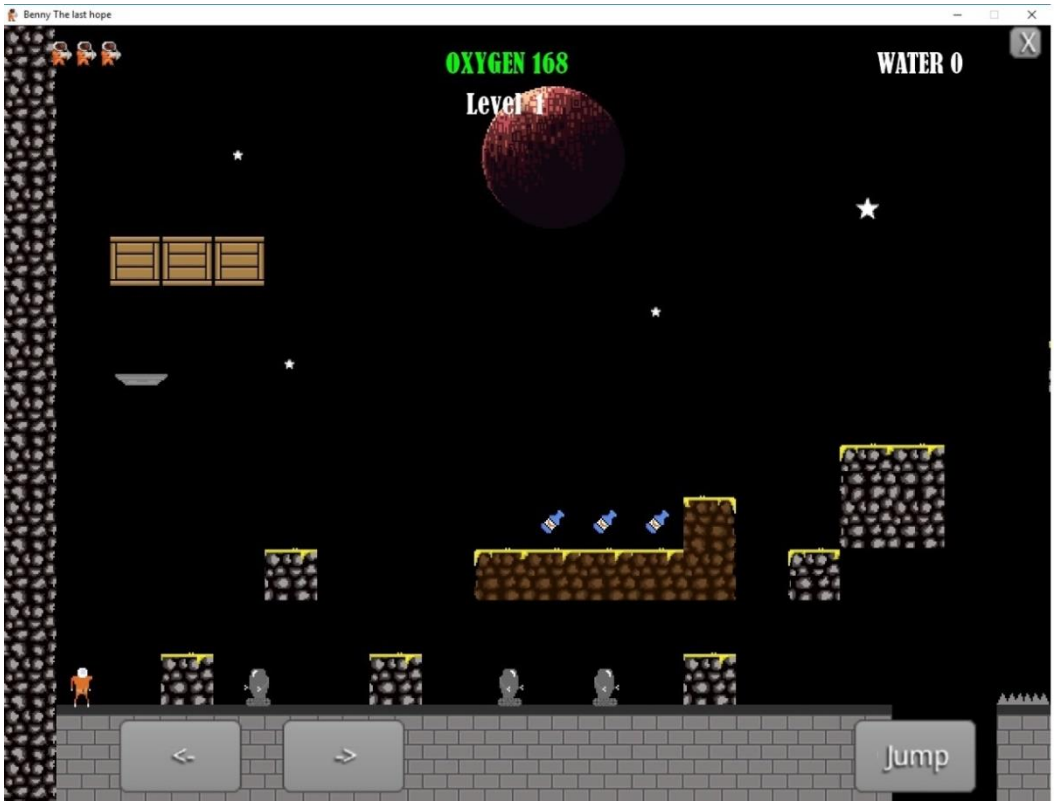
Per tal de que el joc pugui ser jugat en dispositius mòbils Android es decideix incloure una sèrie de botons a mode de controls i per tant fer-lo plenament funcional en aquesta plataforma.

A més, s'afegeix un requadre en les opcions que permet al jugador canviar del mode escriptori (requadre marcat) al mode per dispositius mòbils (requadre desmarcat). A continuació es mostraran imatges del resultat:

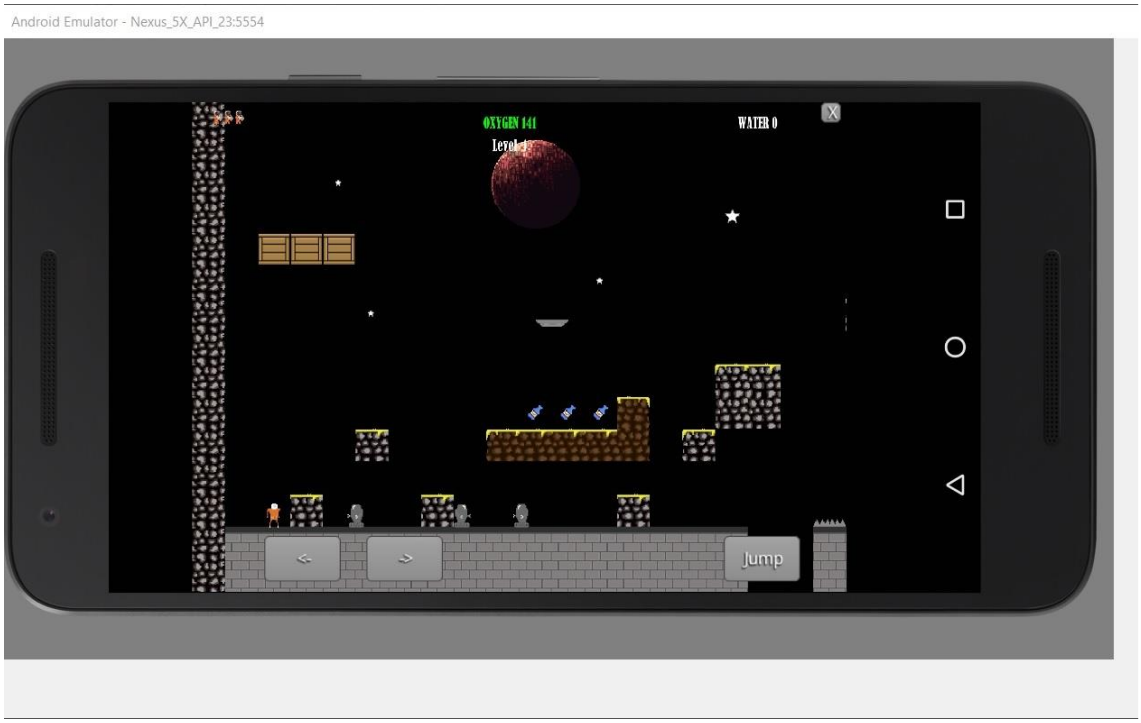
Pantalla d'opcions



Pantalla del joc (execució ordinador)



Pantalla del joc (execució en emulador)



5.7 Eines emprades

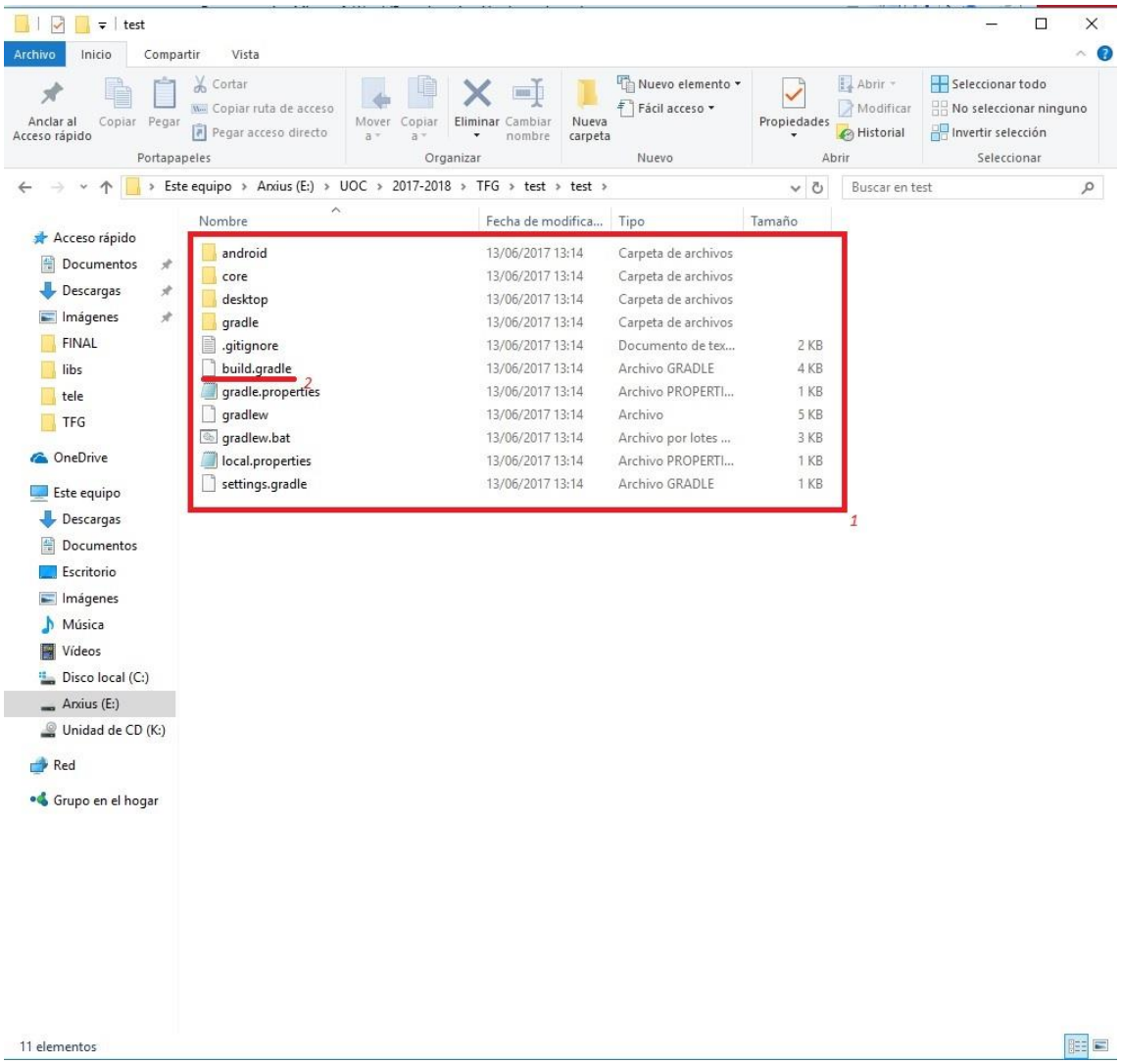
Per poder mostrar les eines emprades durant el desenvolupament d'aquest projecte i per evitar una explicació feixuga, les explicacions s'han dividit en apartats i s'han utilitzat captures de pantalla i notes a peu d'aquestes.

5.7.1 LibGdx

Aquesta captura correspon amb el generador de projectes de *LibGdx,* aquest resulta gairebé imprescindible si es vol desenvolupar un projecte amb aquest *framework* ja que s'encarrega de crear, segons els paràmetres de les caselles: 1, 2, 3 i 4, tots els fitxers i carpetes necessaris.

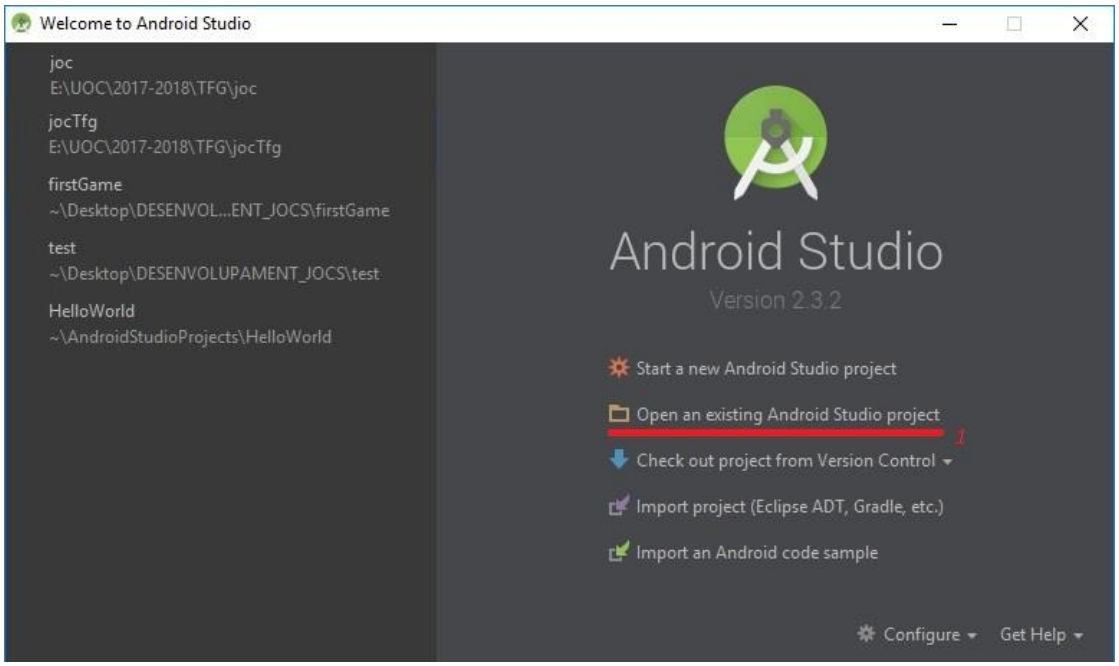
A continuació, s'explicaran cada un dels punts que apareixen en la captura.

1. Informació general: nom del projecte, paquet, classe principal del joc, directori de creació i directori del *SDK*.
2. Plataformes amb les que funcionarà.
3. Extensions integrades que es volen incloure en el projecte.
4. Extensions de tercers que es volent incloure en el projecte.
5. Opcions de configuració addicionals del generador.
6. Botó per generar el projecte.



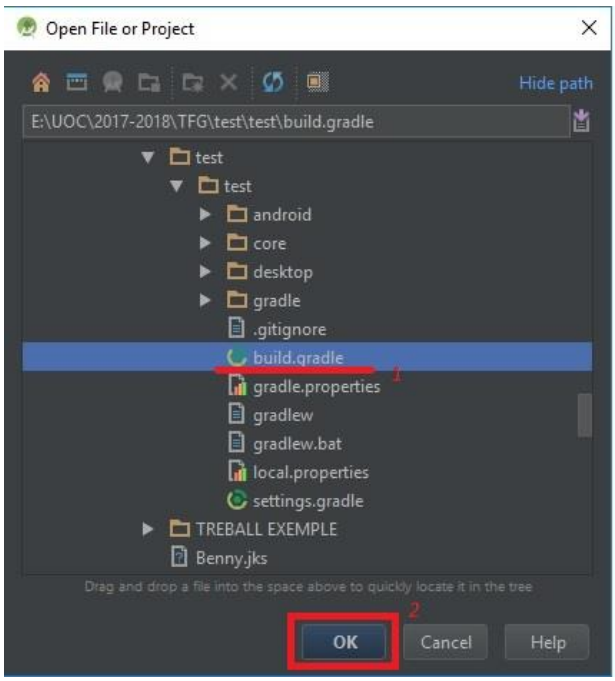
1. Conjunt de carpetes i fitxers del projecte.
2. Fitxer específic per obrir el fitxer amb *Android Studio.*

5.7.2 Android Studio

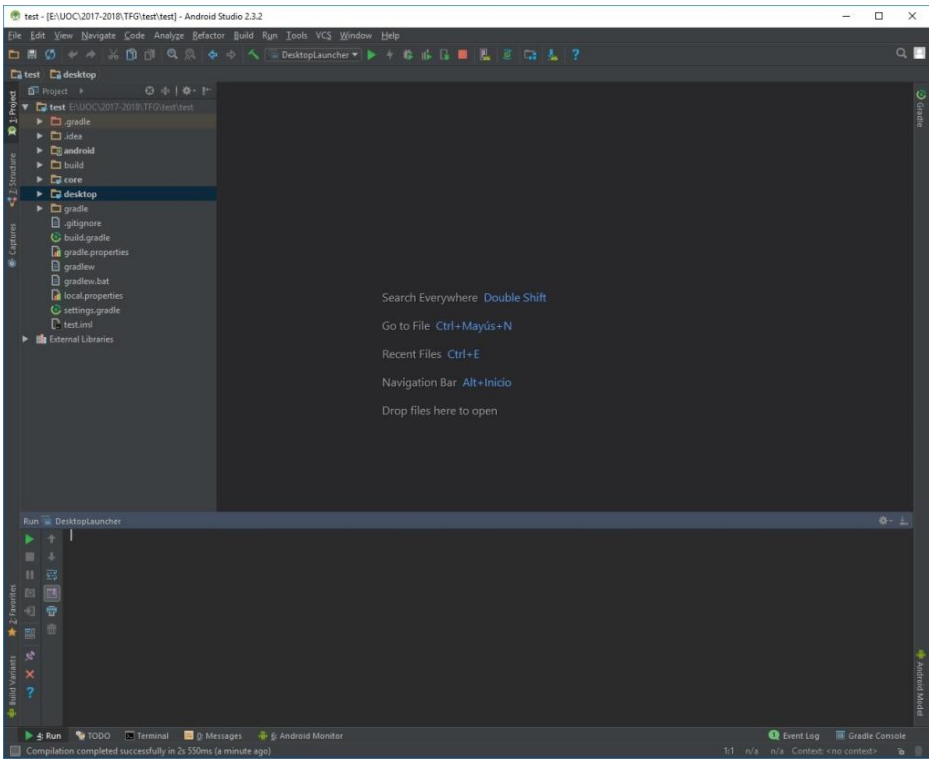


Correspon a la pantalla inicial d'*Android Studio* quan no hi ha cap projecte obert.

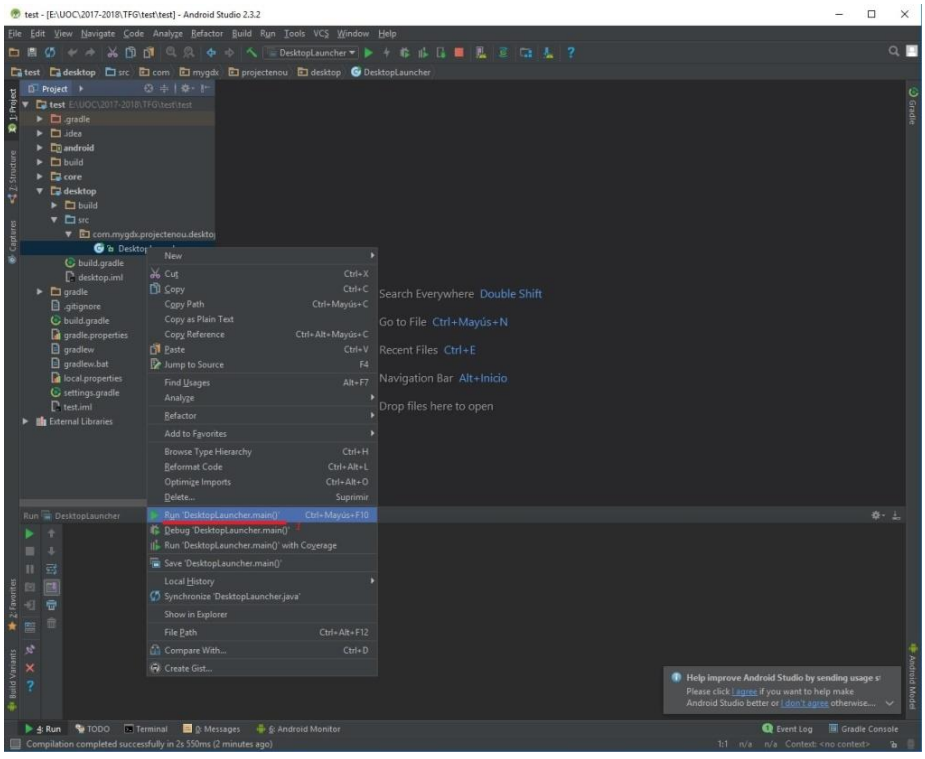
1. Opció per obrir un projecte. Al prémer sobre aquesta s'obre i apareix:



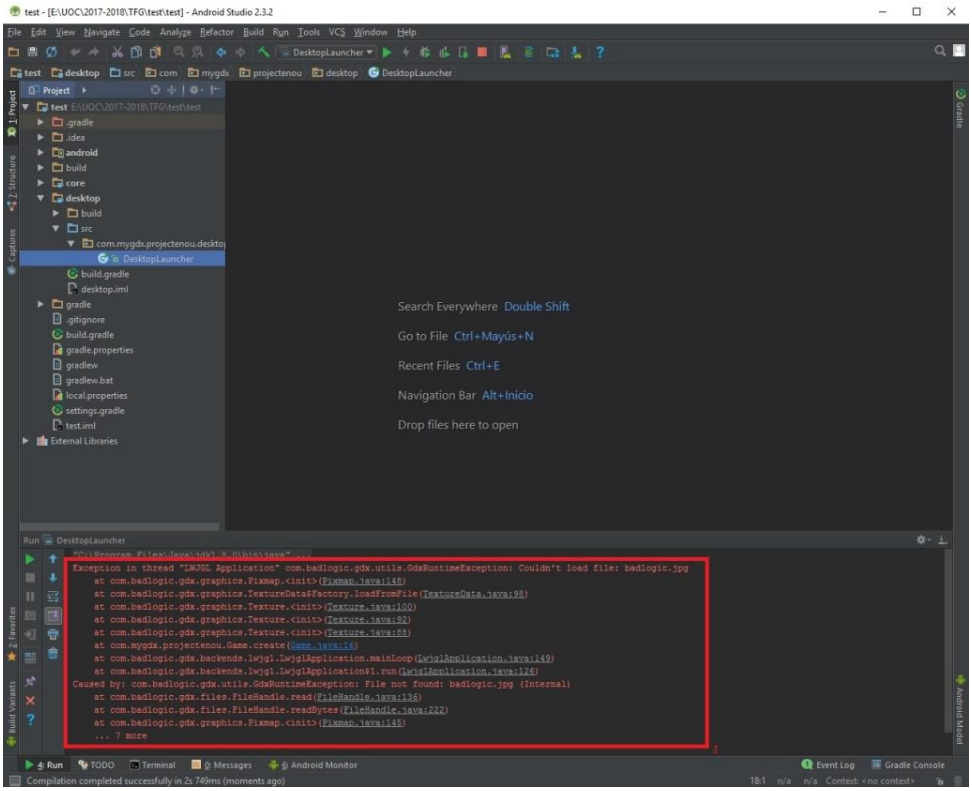
Aquí cal seleccionar el fitxer *build.gradle* del projecte generat amb l'eina anterior i prémer ***OK***.



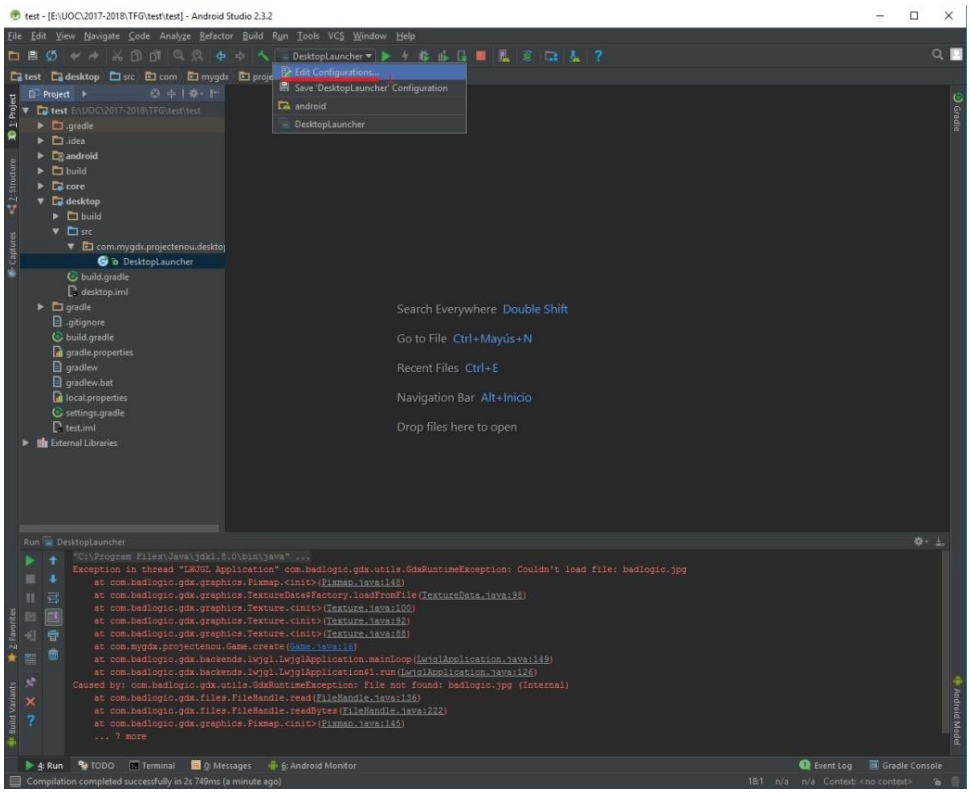
Aquesta es la pantalla que se'ns mostrarà quan s'obri el projecte.



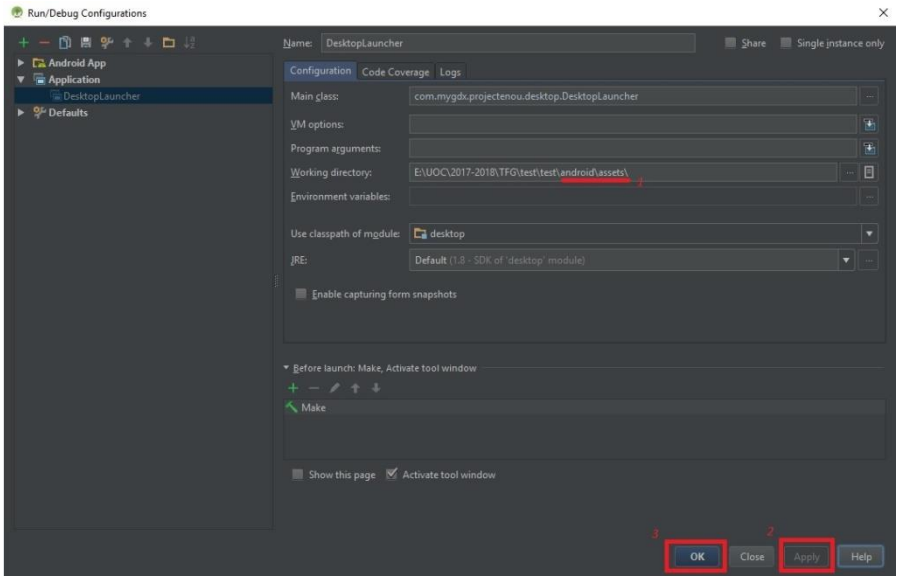
Per tal de comprovar que tot està correcte, executem prement l'opció assenyalada en el punt 1.



Com es pot observar, ha succeït algun problema, tal com es mostra en el punt 1.

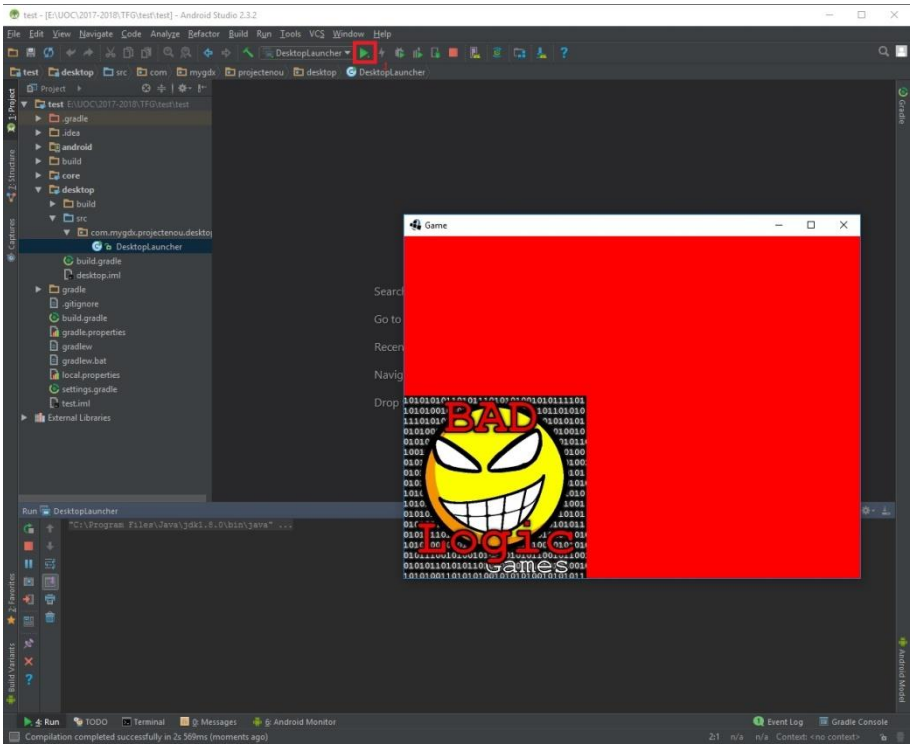


Per tal de solucionar-ho pressionarem l'opció assenyalada en el punt 1.



En aquesta finestra caldrà afegir la part assenyalada en el punt 1 llavors prémer en el botó del punt 2 i per últim prémer el botó del punt 3.

L'explicació del canvi és que cal assenyalar d'on ha d'obtenir els recursos (*assets*) quan s'afegeix que *Android* sigui una de les plataformes les quals volem que funcioni la nostra aplicació.



Ho podem verificar tornant a executar el codi premen la opció assenyalada al punt 1.

5.7.3 Box2D

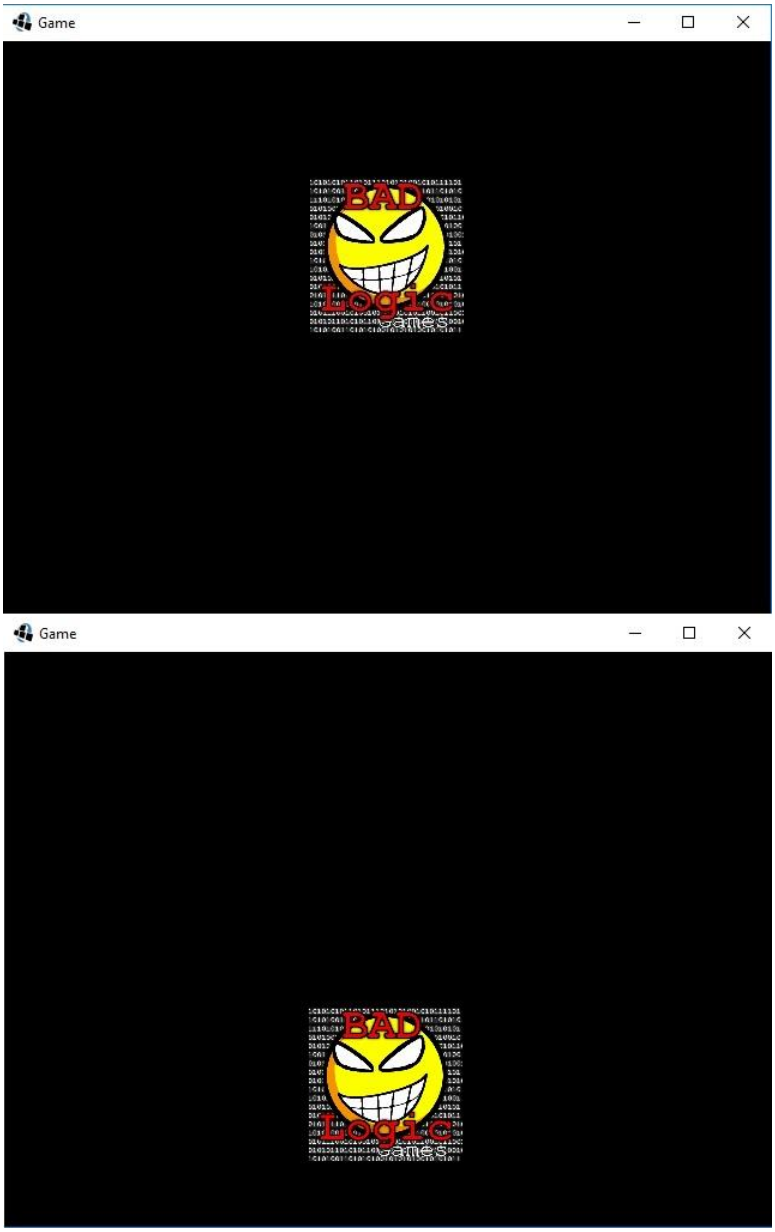
```
package com.mygdx.projectenou;

import ...

public class Game extends ApplicationAdapter {
    SpriteBatch batch;
    Sprite sprite;
    Texture img;
    World world; 1
    Body body; 2

    @Override
    public void create() {

        batch = new SpriteBatch();
        img = new Texture("badlogic.jpg");
        sprite = new Sprite(img);

        sprite.setPosition(Gdx.graphics.getWidth() / 2 - sprite.getWidth() / 4,
                Gdx.graphics.getHeight() /2);

        world = new World(new Vector2(0, -20f), true); 3

        BodyDef bodyDef = new BodyDef(); 4
        bodyDef.type = BodyDef.BodyType.DynamicBody;
        bodyDef.position.set(sprite.getX(), sprite.getY());
        body = world.createBody(bodyDef);


        PolygonShape shape = new PolygonShape();
        shape.setAsBox(sprite.getWidth()/2, sprite.getHeight()/2);

        FixtureDef fixtureDef = new FixtureDef();
        fixtureDef.shape = shape;
        fixtureDef.density = 10f;

        Fixture fixture = body.createFixture(fixtureDef);

        shape.dispose();
    }

    @Override
    public void render() {
        world.step(Gdx.graphics.getDeltaTime(), 6, 2); 5
        sprite.setPosition(body.getPosition().x, body.getPosition().y);

        Gdx.gl.glClearColor(0, 0, 0, 1);
        Gdx.gl.glClear(GL20.GL_COLOR_BUFFER_BIT);
        batch.begin();
        batch.draw(sprite, sprite.getX(), sprite.getY(), sprite.getWidth()/2, sprite.getHeight()/2 );
        batch.end();
    }

    @Override
```

El que es pot veure en aquesta captura és un exemple de codi simplificat per mostrar com fer funcionar aquesta extensió opcional, que s'ha utilitzat en el desenvolupament d'aquest projecte.

Els següents punts mostrats són el codi mínim utilitzat per fer servir *Box2D*:

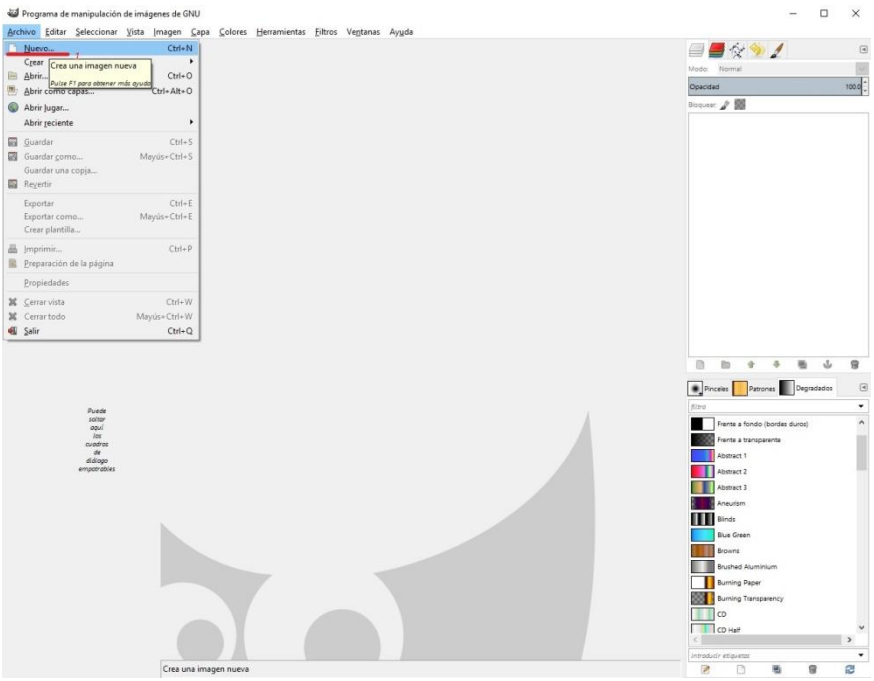
1. *World*: classe bàsica pel funcionament. Conté el motor de físiques.
2. *Body*: classe bàsica per visualitzar un element dins de *Box2D.*
3. Instrucció per a la creació d'un món de 2 dimensions amb gravetat 20 unitats (a la Terra són 9.8).

4. Conjunt d'instruccions per a la definició i creació d'un cos de tipus *Dynamic* amb forma de caixa, centrat amb la finestra, amb densitat de 10 unitats i amb la textura de la imatge per defecte: *badlogic.jpg*.
5. Instrucció que s'encarrega de refrescar el món creat tenint en compte els paràmetres que té.

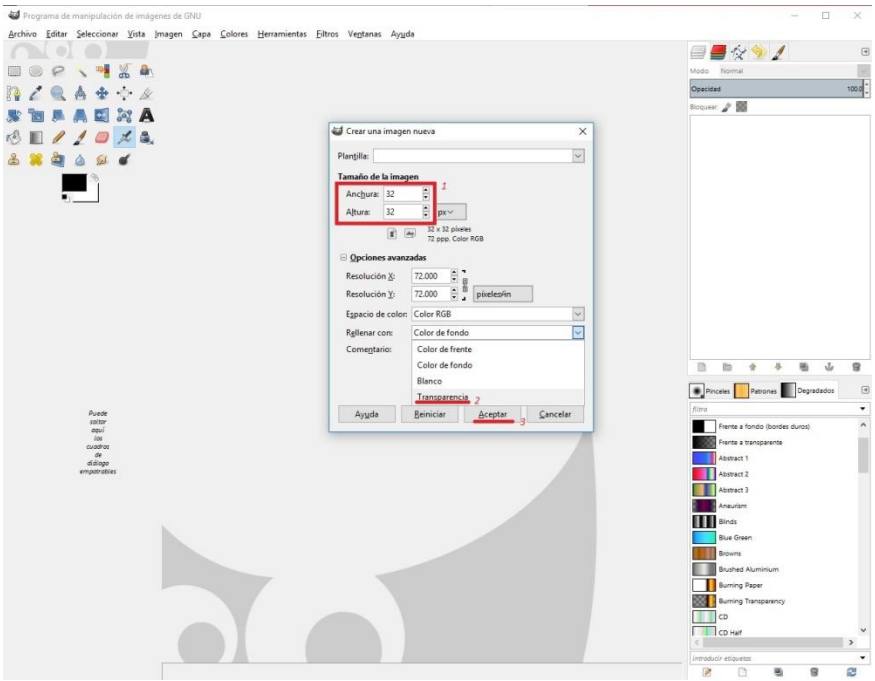


Aquestes captures realitzades en moments diferents de l'execució mostren el desplaçament de caiguda degut a la gravetat que sofreix el cos des de la posició inicial (la primera pantalla) i la posició instants més tard (segona pantalla).

5.7.4 Gimp 2



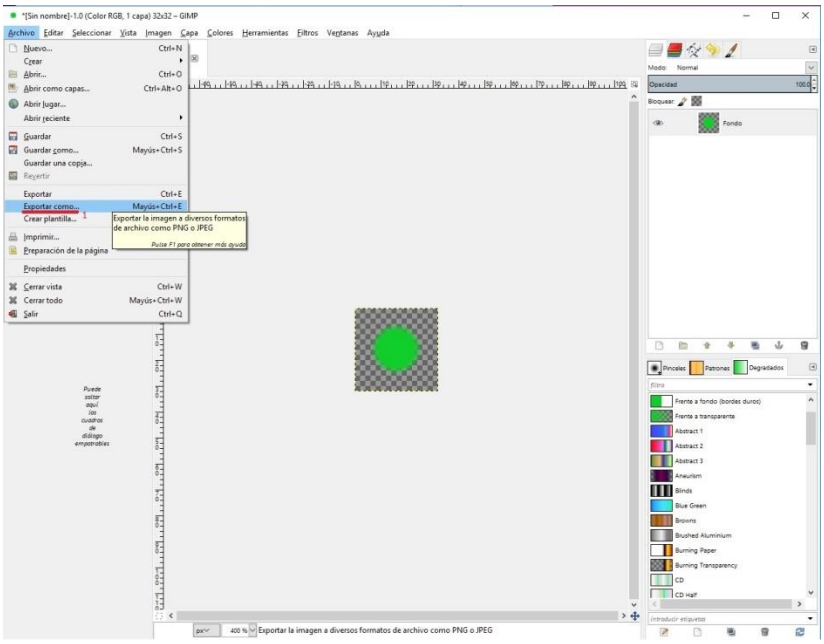
En aquesta captura es mostra com crear un document nou si es prem l'opció assenyalada en el punt 1.



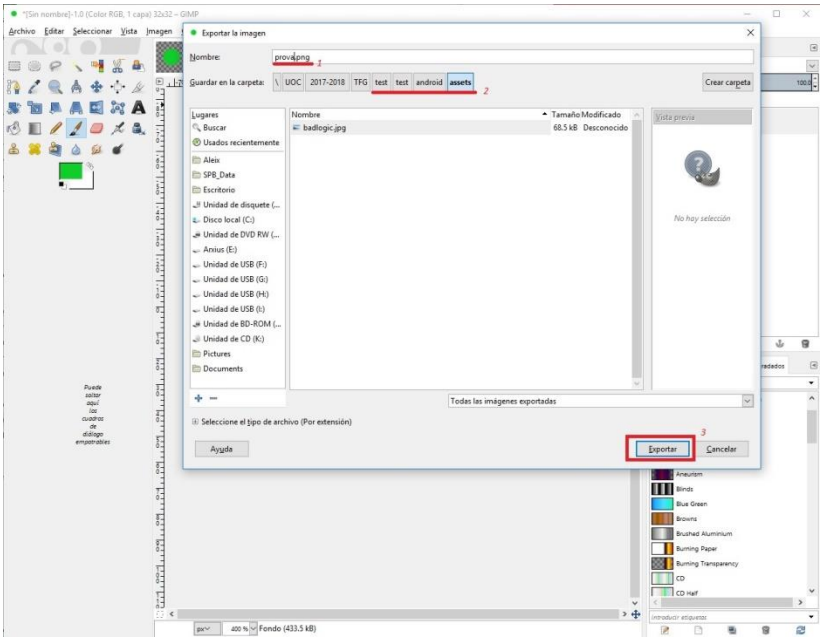
Es pot observar l'aparició d'una nova finestra on es permet triar diverses opcions de les què hi ha assenyalades les principals (punts 1 i 2) a tenir en compte que són:

1. Amplada i alçada de la imatge que volem crear.
2. Tipus de fons de la imatge.

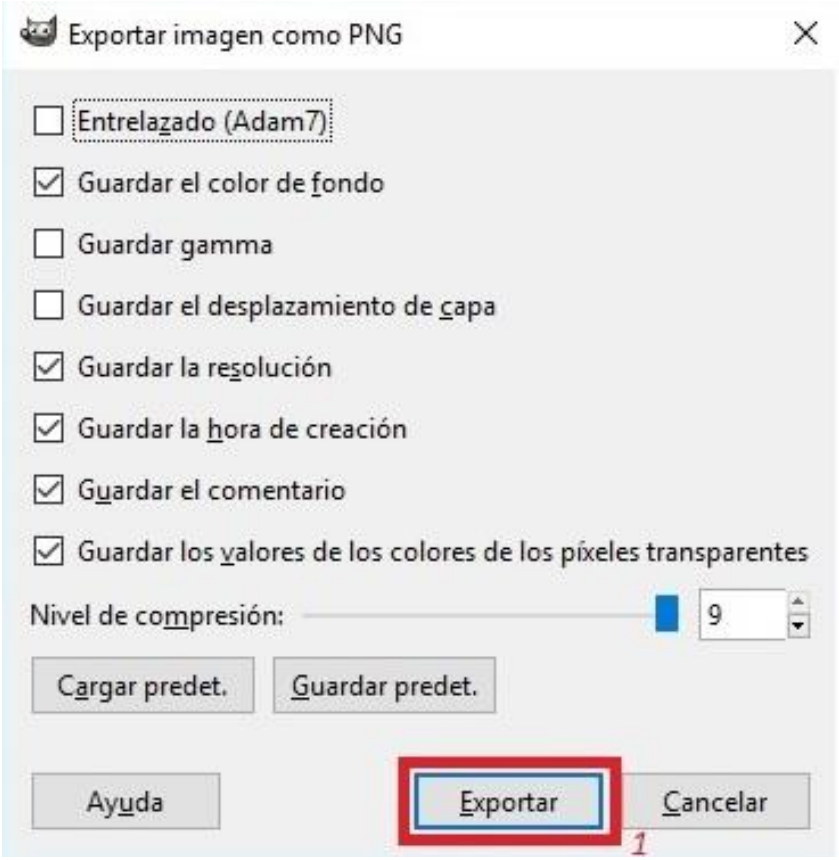
Un cop triats cal prémer el boto del punt 3.



Quan es disposi d'una imatge acabada, es pot exportar per la seva posterior utilització, prement la opció assenyalada en el punt 1.



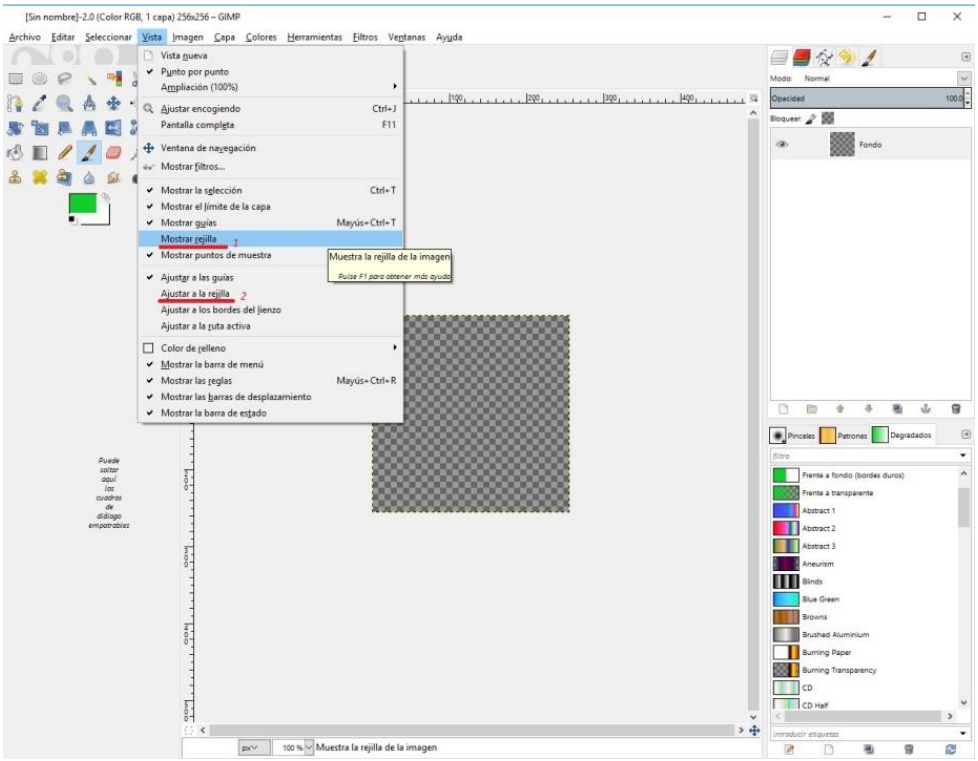
Cal donar-li un nom tal com apareix en el punt 1 i si la imatge es vol utilitzar en el projecte, cal seleccionar la carpeta *assets*, tal i com es mostra en el punt 2 de la captura i posteriorment cal prémer el botó del punt 3. Aleshores apareixerà la següent finestra:



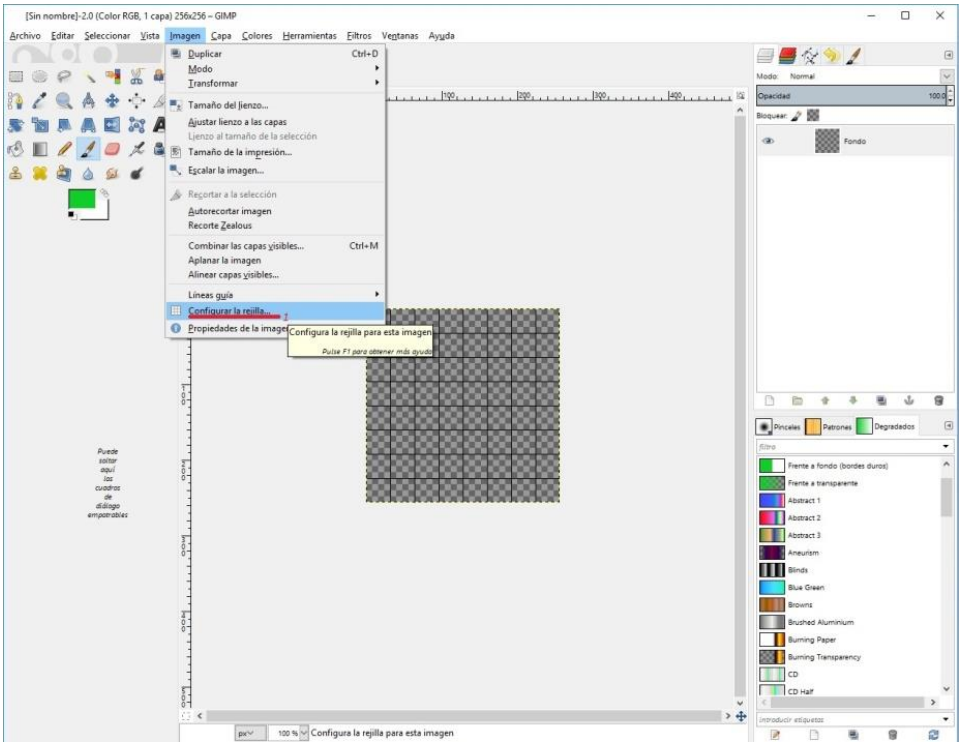
Unicament cal prémer el botó assenyalat a l'opció 1 i la imatge quedarà exportada al directori dels recursos.

Si es vol crear un document amb totes les textures per utilitzar en el joc, també anomenat *tile set,* només cal tornar a crear un nou document, tal i com s'ha explicat. La diferencia estaria en les dimensions donat que, enlloc de ser 32x32 com seria per un *tile*, seria tan gran com es desitgés, tenint en compte però que si el joc esta conformat per cel·les de tamany X, aquest nou fitxer ha de tenir sempre d'alçada i/o d'amplada, un múltiple d'aquest tamany X.

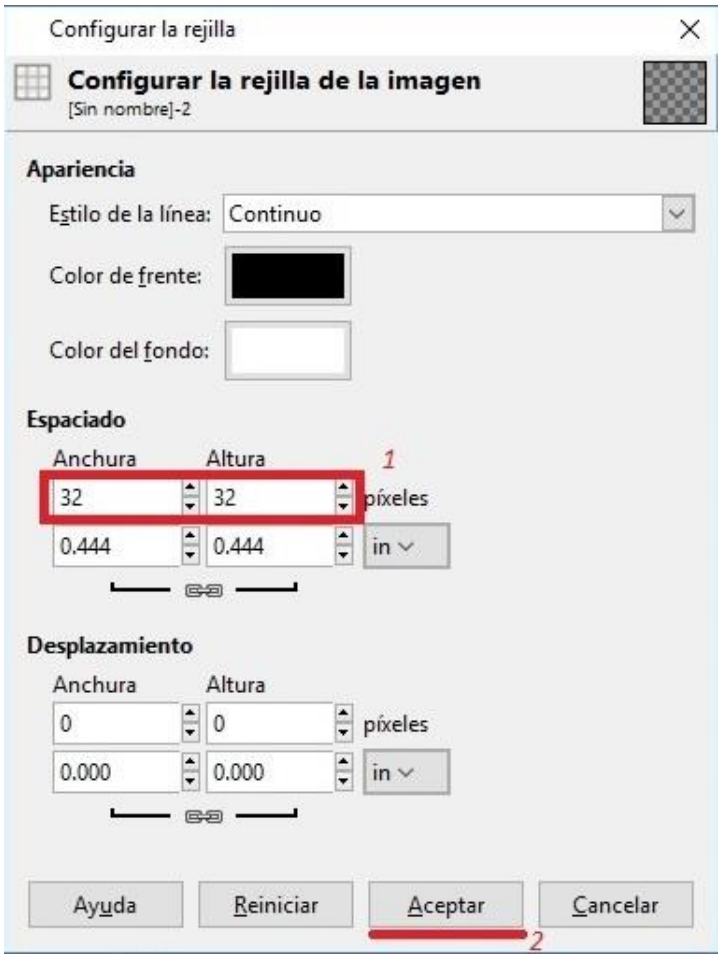
En la següent captura es mostra un *tile set* de 256x256.



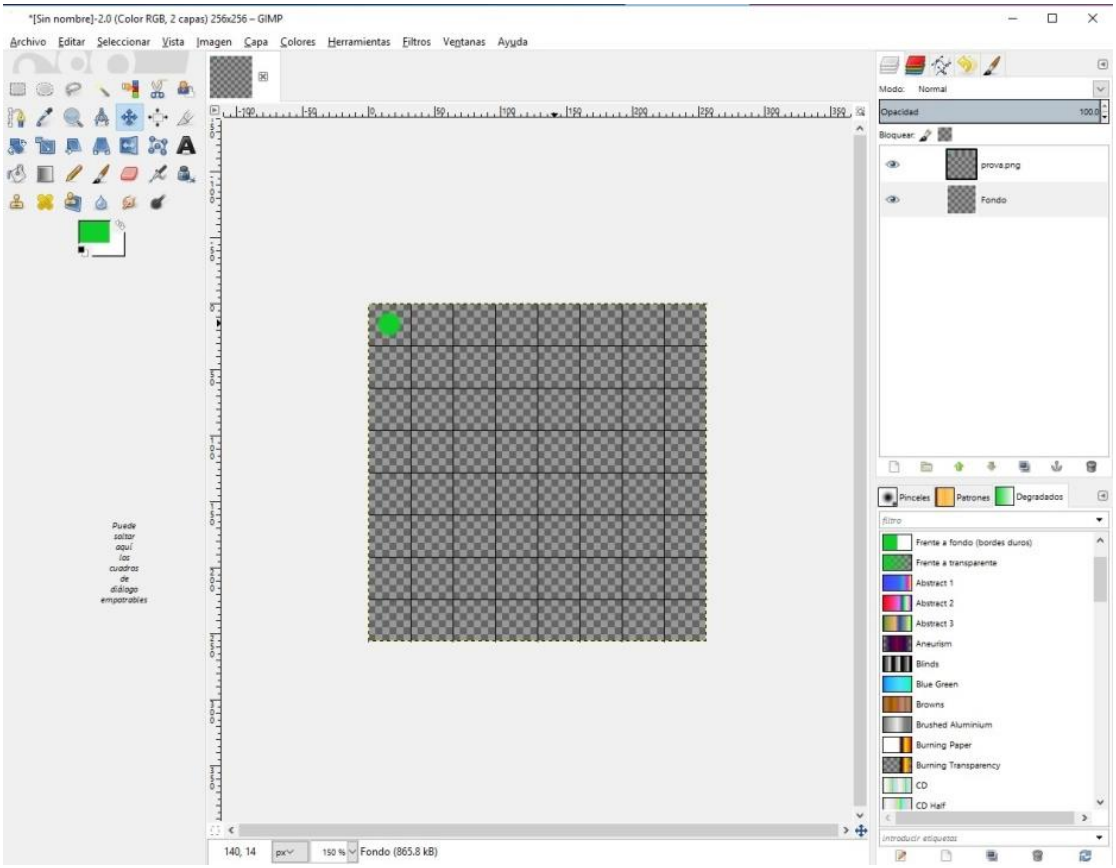
Les opcions assenyalades en els punts 1 i 2 són necessàries perquè després és més fàcil anar omplint el *tile set* de forma correcta respectant les cel·les.



En aquesta captura és mostra com accedir a la configuració de la graella. Al prémer el botó de l'opció 1 apareixerà la següent finestra:



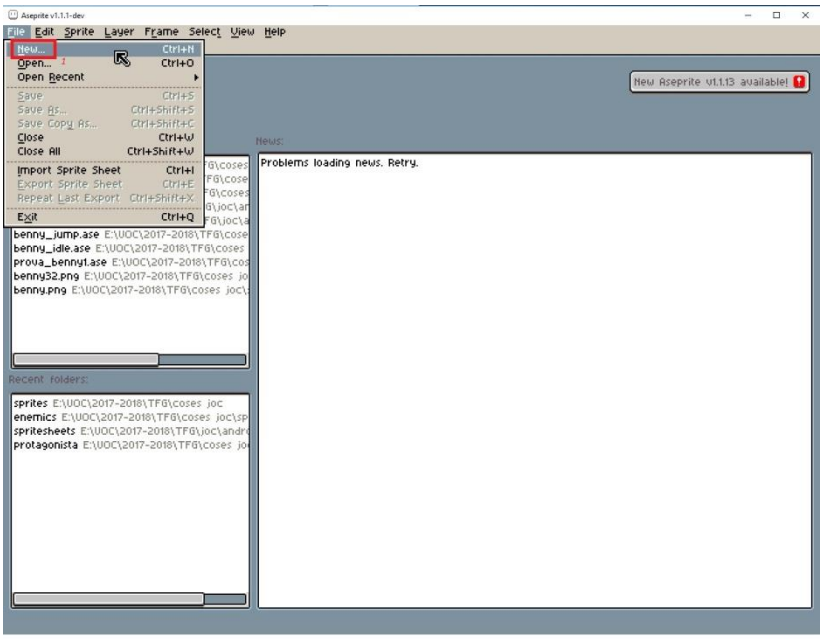
La part important d'aquesta finestra és l'assenyalada en el punt 1, perquè l'amplada i l'alçada seran les que defineixen el tamany de cada cel.la de la graella. Per continuar només cal prémer el botó del punt 2.



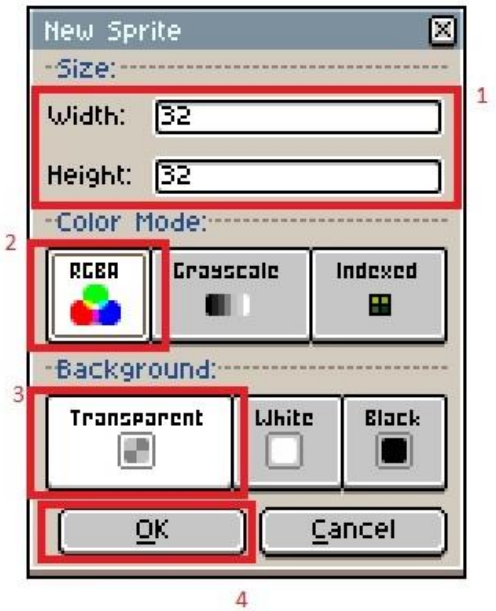
Aquesta darrera captura mostra com queda la textura realitzaea anteriorment dins d'aquest nou *tile set*.

L'avantatge principal en la utilització d'un *tile set* envers un conjunt d'imatges individuals és el rendiment. Això és perquè la targeta gràfica gestiona de manera més optima l'accés a diferents *Regions* d'una imatge que accedir a textures independents i redibuixar-les.

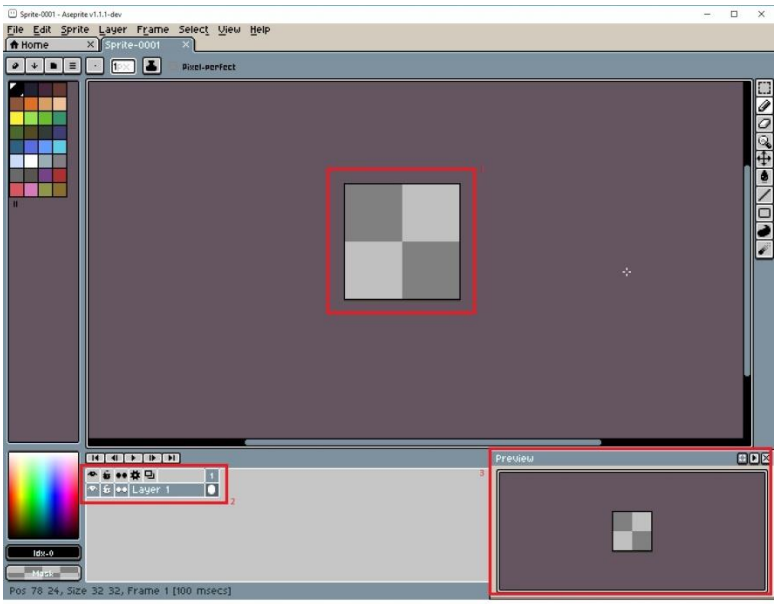
5.7.5 Aseprite



En aquesta captura es mostra com crear un nou *sprite* prement el botó assenyalat pel punt 1. Un cop realitzada aquesta acció s'obre una nova finestra com la que es mostra a continuació:

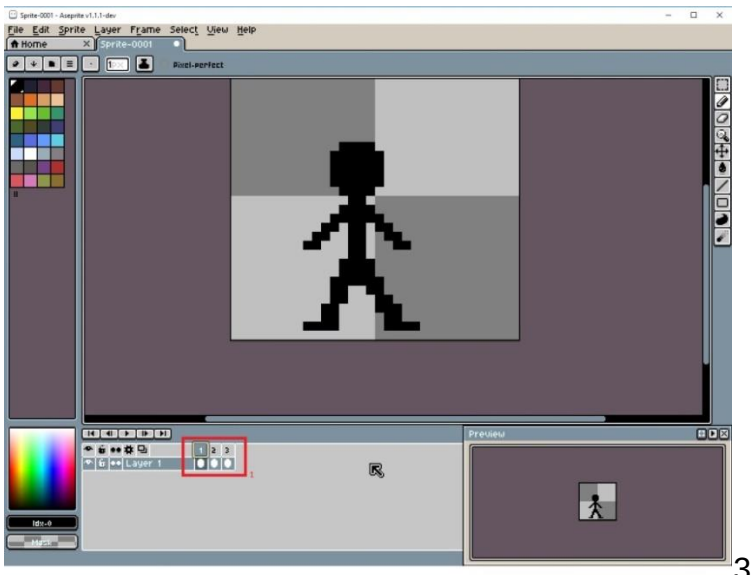


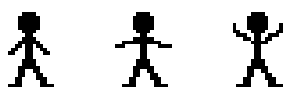
En aquesta caldrà configurar el *Width* (amplada) i el *Height* (alçada) que apareixen en el requadre del punt 1, el tipus de color (*RGBA* en el nostre cas) del punt 2 i color de fons del punt 3. Un cop es disposa de tots els valors, es pot prémer el botó del punt 4 per continuar.



La pantalla que apareixà conté 3 elements importants :

1. Quadre de dibuix amb les dimensions establertes en la finestra anterior.
2. Panell que mostra el nombre de capes (visibles i no visibles), si estan bloquejades, per evitar-ne l'edició, i els fotogrames que composen la capa.
3. Finestra de previsualització que permet veure el resultat de la interpolació dels fotogrames de les capes que estiguin dibuixats.

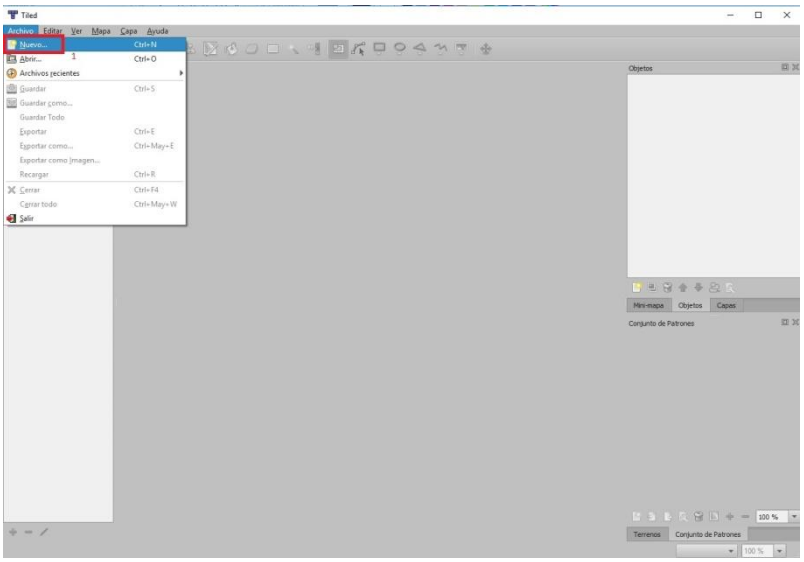




3 fotogrames

Un cop finalitzats els fotogrames, es poden exportar com *tile set*.

5.7.6 Tiled

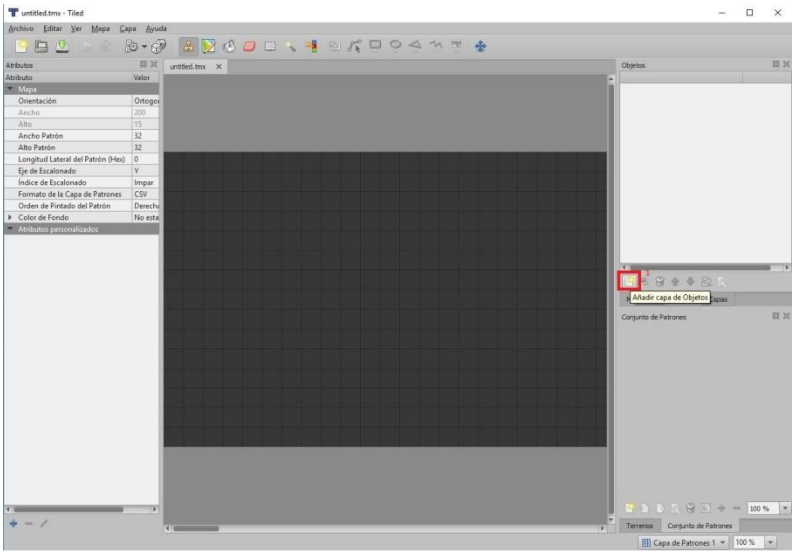


Si es prem el botó, s'obre una nova finestra com la mostrada a continuació:

En aquesta finestra hi ha 2 apartats importants com són:

1. L'ample i l'alçada del mapa: defineix el tamany del mapa.
2. L'ample i l'alçada del patró: defineix el tamany d'una cel.la.

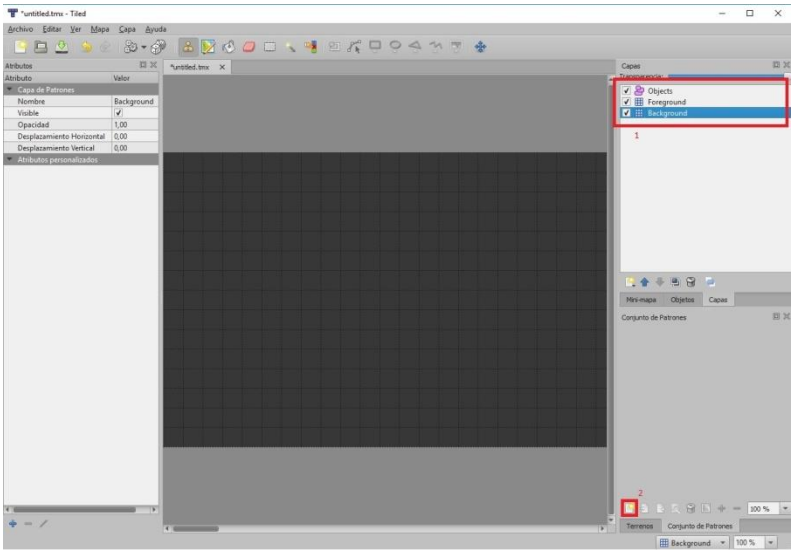
Un cop estiguin configurats aquests paràmetres, cal prémer el botó del punt 3.



En la captura es mostra el botó per a crear noves capes. Aquestes capes poden ser de:

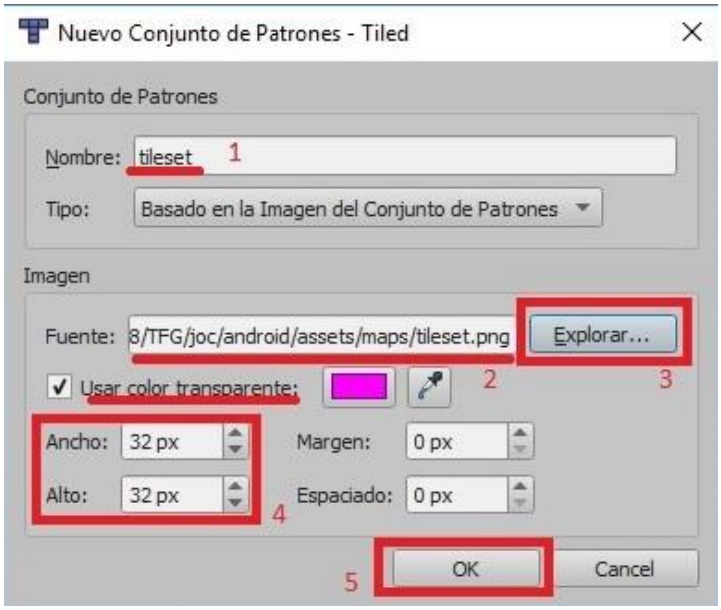
- Patrons
- Objectes
- Imatges

A continuació es mostra el resultat de crear diverses capes.

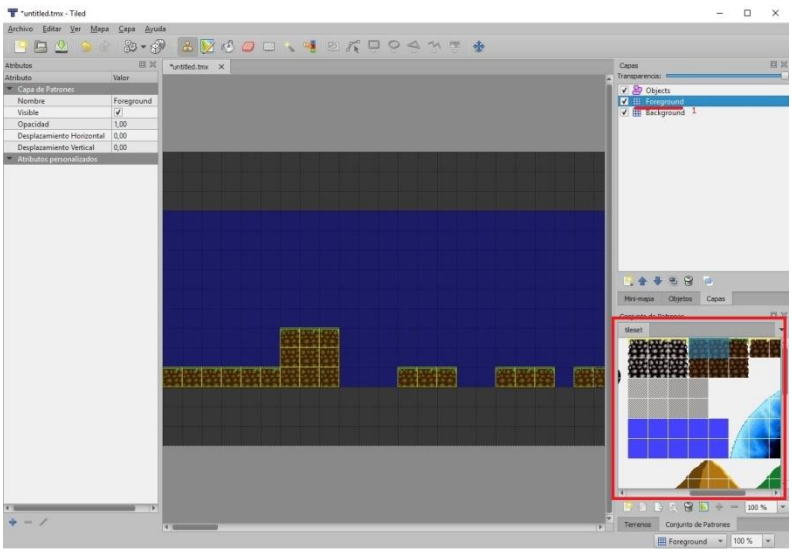


El punt 1 mostra les capes creades amb els noms: *Objects, Foreground* i *Background.* En el punt 2, es troba el botó per crear nous conjunts de patrons.

Al prémer el botó del punt 2 es presenta la finestra següent:



Per tal de seleccionar un *tile set* o conjunt de patrons, cal prémer el botó del punt 3 i després seleccionar-lo. De forma addicional, es pot definir el color de fons que fa servir com *alpha* dins del *tile set*. També cal assegurar-se que el tamany del *tile* és l'adequat. Una cop tot esta configurat, cal prémer el botó del punt 5.



En aquesta captura es mostra en el punt 2 el *tile set* carregat, amb el que es pot començar a construir el nivell.

Com a estàndard es sol dividir en les capes:

- *Objects*
- *Foreground*
- *Background*

La primera constitueix tots els elements dels que ha d'estar composat el nivell i que el seu comportament serà gestionat en el codi. Pot subdividir-se en varies capes per motius de llegibilitat.

La segona constitueix tots els elements gràfics que formaran el nivell. Pot ser de patrons o d'imatges.

La tercera constitueix tots els elements gràfics de fons, és a dir, que formarien part del paisatge. Pot ser de patrons o d'imatges.

6 Conclusions

Ara, un cop finalitzat el projecte, puc afirmar que les conclusions a les que he arribat han estat moltes i diverses i tenen a veure amb els diferents àmbits del desenvolupament del videojoc i també amb els moments del procés de producció.

Malgrat que sembli ser una realitat obvia, l'experiència que ha comportat aquest projecte m'ha confirmat que el desenvolupament d'un videojoc, encara que sigui senzill, és una tasca complexa i que, per tant, requereix garantir de que es disposa d'un nivell mínim en un seguit de competències tècniques i de coneixements i experiència en disciplines diverses, per a poder fer front als reptes que es van plantejant des del mateix moment en que es concep la idea del joc, en el disseny, en el transcurs del desenvolupament i en l'etapa en que es finalitza.

Els problemes que s'han presentat han estat de molts tipus i depenent del moment o fase on han sorgit, podien haver resultat desastrosos pel desenvolupament del projecte.

A més, el fet de que un únic professional ha hagut de fer-se càrrec de tot el desenvolupament, tal i com ha estat en aquest cas, encara ho fa tot més feixuc. Val a dir que al marge de disposar d'un bagatge de coneixements específics i transversals i d'haver adquirit tot un seguit de competències tècniques que aportà el Grau d'Enginyeria Informàtica i malgrat poder-hi afegir una experiència prèvia en programació, hi ha hagut situacions molt complicades que, segons el moment del projecte, m'han fet pensar que no superaria.

He constatat que per desenvolupar un videojoc, a més de disposar de coneixements i habilitats per a la programació, cal tenir-ne també pel disseny, per a l'edició gràfica, per a la il·luminació, pel so, per a l'edició de nivells, habilitats narratives per a elaborar una història i per al modelat si es fan ús d'elements 3D.

M'ha quedat més que palès, que per a desenvolupar un videojoc més complex és imprescindible disposar d'un equip interdisciplinari d'experts en les esmentades habilitats, ben coordinat, amb suficients recursos i amb una acurada planificació.

En definitiva, el desenvolupament del videojoc ha estat un procés exigent, tot i així, es pot afirmar que he pogut assolir tots els objectius plantejats.

Pel que fa a la planificació, he pogut seguir pràcticament totes les fites que s'havien proposat a excepció de la redacció parcial i continuada de la memòria.

Si bé aquesta tasca estava present durant tot el desenvolupament del projecte, en alguns moments ha estat necessari aturar-la i dedicar el temps reservat per aquesta tasca, a l'aprenentatge d'alguna de les eines emprades per a realitzar el joc.

Amb tota seguretat, si hagués de tornar a iniciar el projecte, amb les experiències viscudes fins aquí, el plantejament quant a la distribució del temps i esforç per al treball seria diferent. Posaria molt més èmfasi, atenció i sobretot ampliaria el temps per a la fase de planificació i disseny, donat que he verificat que aquesta fase és clau per tal de disposar d'un producte complet, a nivell teòric, abans de començar a *picar* codi.

Des del moment d'endinsar-me en el context històric on van sorgir els jocs de plataformes i en la seva evolució al llarg dels més de 30 anys de la seva existència, que quasi bé coincideixen amb la meva, i fins a la finalització del meu videojoc, he invertit moltes hores de treball, moltes més de les que estaven previstes per a la realització del projecte i també moltes més de les que jo planificava dedicar-hi, malgrat això no em penedeixo, perquè sóc conscient que ha estat una gran inversió per al meu bagatge professional.

A més de desenvolupar un videojoc de les característiques que havia plantejat inicialment, amb la qual cosa s'ha assolit el primer dels dos objectius plantejats, també he pogut endinsar-me en la lògica interna del procés, fins i tot elaborant "artesanalment" els gràfics, els escenaris, les animacions, el personatge, etc., per tal de mesurar els nivells de complexitat que comporta la seva realització, assolint per tant, l'altre objectiu, que per a mi era prioritari.

7 Glossari

- **Super Mario Bros:** És un videojoc de plataformes dissenyat per Shingeru Miyamoto i desenvolupat per la companyia Nintendo l'any 1985. Degut al seu espectacular èxit, va donar inicia a una nisaga o franquícia que actualment encara esta en funcionament. El joc descriu les aventures dels germans Mario i Luigi que han de rescatar a la princesa Peach que ha estat segrestada pel rei dels Koopas i en Bowser i es troba en el Regne Xampinyó.
- **Pixel art:** És una forma d'art digital, creat a partir de l'ús de programari d'imatges rasteritzades, on les imatges són editades a nivell de píxel.
- **Retro:** En terminologia de videojocs significa antic.
- **Android Studio:** És el entorn de desenvolupament integrat oficial per la plataforma Android. Addicionalment pot emprar-se pel desenvolupament de qualsevol projecte amb llenguatge Java.
- **IDE** (Entorn de desenvolupament integrat): És una aplicació informàtica que proporciona serveis integrals per a facilitar al desenvolupador o programador el desenvolupament de programari.
- **Framework:** És un conjunt estandarditzat de conceptes, pràctiques i criteris a l'hora d'enfocar o resoldre un tipus de problemàtica.
- **LibGdx:** És un *framework* pel desenvolupament de videojocs multiplataforma, suportant en l'actualitat Windows, Linux, Mac, OS, Android, IOS i HTML5.
- **Box2D:** És una biblioteca lliure que implementa un *engine* de físiques en dos dimensions.
- **Engine** (Motor): És un terme que fa referència a una sèrie de rutines de programació que permeten el disseny, la creació i la representació d'un videojoc. La funcionalitat bàsica es proveir al videojoc d'un motor de renderitzat per gràfics 2D i/o 3D, un motor de físiques, control de col·lisions, sons, ...
- **Gimp 2:** És una aplicació pel tractament d'imatges.

- **Aseprite:** És un editor d'*Sprites* animats i una eina molt bona per realitzar *Pixel art*.
- **Sprite:** És un element gràfic que es pot desplaçar sobre la pantalla. L'ús dels sprites és una tècnica fonamental en els videojocs 2D i principalment en la realització d'efectes especials en videojocs 3D.
- **Tiled:** És un editor potent de mapes 2D. Suporta mapes ortogonals i isomètrics. El format d'arxiu és TMX.
- **Tile map** (Mapa de rajoles): Mapa creat a partir de tiles.
- **2D:** És la representació gràfica que només utilitza dos de les tres dimensions de l'espai. Sistema de representació utilitzat des de els primer jocs.
- **Desplaçament lateral:** És un tipus de joc de plataformes on el jugador únicament pot desplaçar-se d'esquerra a dreta i viceversa.
- **Arcade:** És el terme genèric per referir-se a les maquines recreatives de videojocs que solien estar en llocs públics.
- **Pla zenital:** És el pla en que el punt de vista d'una càmera es troba perpendicular respecte al sòl.
- **NES** (Nintendo Entertainment System)**:** És una videoconsola de sobretaula de 8 bits desenvolupada per Nintendo l'any 1983.
- **SNES** (Super Nintendo Entertainment System)**:** És una videoconsola de sobretaula de 16 bits desenvolupada per Nintendo l'any 1990.
- **Sega Mega Drive:** És una videoconsola de sobretaula de 16 bits desenvolupada per Sega Enterprises l'any 1988.
- **Game Boy:** És una videoconsola portàtil de 8 bits desenvolupada per Nintendo l'any 1989.
- **Sega Game Gear:** És una videoconsola portàtil de 8 bits Sega Enterprises l'any 1990.
- **2.5D:** També anomenada perspectiva 3/4. És basa en la projecció en 2D i tècniques que permeten fer que una sèrie d'elements o escenes semblin 3D quan en realitat no ho són.
- **3D:** És la representació gràfica de les tres dimensions de l'espai. Sistema de representació més modern que va sorgir a la dècada dels 90.

- **Dreamcast:** És una videoconsola de sobretaula de 32 bits desenvolupada per Sega Enterprises l'any 1998.
- **Nintendo 64**: És una videoconsola de sobretaula de 64 bits desenvolupada per Nintendo l'any 1996.
- **Nintendo GameCube:** És una videoconsola de sobretaula de 64 bits desenvolupada per Nintendo l'any 2001.
- **Game Boy Advance:** És una videoconsola portàtil de 32 bits desenvolupada per Nintendo l'any 2001.
- **Freeware** (Programari gratuït): Es tracta de la denominació que té el programari que es d'utilització lliure i que per tant es distribueix sense cost però que manté el *copyright* i per tant no es pot modificar o vendre.
- **Playstation:** És una videoconsola de sobretaula de 32 bits desenvolupada per Sony Computer Entertainment l'any 1995.
- **Playstation 2:** És una videoconsola de sobretaula de 128 bits per Sony Computer Entertainment l'any 2000.
- **GOAL** (Game Oriented Assembly Lisp)**:** És un llenguatge de programació de videojocs desenvolupat per Andy Gavin i l'equip de desenvolupament de Naughty Dog per a la nisaga del videojoc Jak and Daxter.
- **Nintendo DS:** És una videoconsola portàtil de 32 bits desenvolupada per Nintendo l'any 2005.
- **Wii:** És una videoconsola de sobretaula de 64 bits desenvolupada per Nintendo l'any 2006.
- **Debug:** El debug mode o mode de depuració és una eina que utilitzen els desenvolupadors i provadors de jocs per accedir o executar certes rutines i accions difícilment reproduïbles en condicions normals.
- **Body:** Objecte i part fonamental en la física d'una escena dins de Box2D. Aquests objectes tenen massa, velocitat, inèrcia rotacional, velocitat angular, localització i angle.

    - **Body Static:** Són tipus de cossos que no es veuen afectats per la gravetat i que no es mouen. Poden produir una col·lisió però no se'n veuran afectats.
    - **Body Dynamic:** Són tipus de cossos que es veuen afectats per la gravetat, les col·lisions i que es mouen. Poden produir una col·lisió i se'n veuran afectats.
    - **Body Kinematic:** Són tipus de cossos que no es veuen afectats per la gravetat, les col·lisions i que es mouen. Poden produir una col·lisió però no se'n veuran afectats.
- **TMX** (Translation Memory eXchange)**:** És un estàndard d'XML que serveix per l'intercanvi de memòries de traducció.
- **HUD** (Head-up display)**:** Element gràfic en la pantalla encarregat de mostrar al jugador, però sense entorpir-lo, informació important del videojoc. Un exemple seria mostrar les vides restants, puntuació, ...
- **SDK** (Kit de desenvolupament de programari)**:** És un conjunt d'eines de desenvolupament de programari que li permet al programador o desenvolupador de programari crear aplicacions per un sistema concret.
- **Asset:** Cadascun dels elements que composen el joc (animacions, models, IA, sons, ...).
- **Tile:** Representa una rajola dins un mapa de rajoles. Normalment conté un sprite.
- **Tile set:** És un full que conté tots els *sprites* (tiles) emprats en un joc de 2D.
- **Alpha:** És el canal que actua com a mascara de transparència en una imatge.

8 Bibliografia

- The Indie Game Developer Handbook, Richard Hill-Whittall, Focal Press https://www.amazon.es/Indie-Game-Developer-Handbook/dp/1138828424
- La Gran Història de los Videojuegos, Steven L.Kent, NOVA https://www.amazon.es/Gran-Historia-Los-Videojuegos-NOVA/dp/8466655026/ref=pd_sim_14_3?_encoding=UTF8&psc=1&refRID=SCYNTVQHYGRWF378PFPZ
- https://twitter.com/the_fab_makario
- http://opengameart.org/
- https://www.spriters-resource.com
- https://audiojungle.net/
- https://www.premiumbeat.com/
- http://soundbible.com/
- https://es.wikipedia.org
- https://ca.wikipedia.org
- https://en.wikipedia.org/
- http://www.gamerdic.es
- https://elmundodeubuntu.blogspot.com.es/2015/11/aseprite-editor-de-sprites.html
- http://citrogamers.blogspot.com.es/2015/05/tipos-de-juegos-de-plataformas.html
- https://github.com/libgdx/libgdx/wiki
- http://box2d.org/manual.pdf
- http://www.iforce2d.net/b2dtut/
- https://stackoverflow.com