

Candy QOMP

Yaiza Cano
Narcís Terrado

December 2020



Contents

1	Introduction	3
2	Descripció	3
2.1	Objectes i Funcionalitats	3
2.1.1	Candy Cane	3
2.1.2	Spikes	4
2.1.3	Pumpkin	4
2.1.4	Buttons & Barriers	5
2.1.5	Door & Key	5
2.1.6	Chupa Chups	6
2.1.7	Sugus	6
2.1.8	Tilemaps	7
2.2	Instruccions	7
3	Metodologia	7
4	Conclusions	9

1 Introduction

S'ha pres com a referència el joc *QOMP* que és una evolució del *POMG* feta pel mateix desenvolupador, [Stuffed Wombat](#), juntament amb [Britt Brady](#), [Miroko](#) i [Clovelt](#).

A dia d'avui, encara no s'ha fet el llançament del joc a Steam, el qual està previst per aquest 2020. Així doncs, es coneix més aviat poc d'aquest nou videojoc però promet ser, si més no, divertit. Adjuntem tant la [pàgina web](#) com l'entrada a [Steam](#) on podreu trobar més informació i veure el [tràiler oficial](#).

2 Descripció

El joc està inspirat en el joc *POMG*, que, ahora està inspirat en el joc original *PONG*, el qual, si recordem tots, és un videojoc creat per *Atari* i comercialitzat en màquines recreatives per primera vegada el 29 de novembre de 1972.

El joc és una adaptació del tennis de taula a la pantalla i no va més enllà, té un funcionament molt bàsic: cada oponent controla una de les raquetes, i ha d'evitar que l'altre jugador envii la pilota fora del seu camp. El moviment de les raquetes és en vertical, i la pilota es mou en dues dimensions. Si la pilota xoca contra alguna de les dues raquetes, canvia la seva direcció horitzontal, mentre que si xoca contra alguna de les parets, canvia la seva direcció vertical.

Així doncs, el *QOMP* proposa la idea de tot un món que envolta aquestes dues raquetes i la pilota, la qual personifica i declara a la recerca de la llibertat. Nosaltres serem els encarregats d'aconseguir que la pilota aconsegueixi aquesta llibertat derrotant bosses, resolent trencaclosques i superant reptes de plataformes amb la única interacció d'un botó (barra d'espai).

2.1 Objectes i Funcionalitats

La nostra versió, *Candy QOMP*, és una versió més senzilla la qual compta amb algunes de les implementacions del *QOMP* que s'han fet públiques fins ara.

Els objectes s'han modelat a partir d'estructures 3D bàsiques oferides per [Unity](#) i personalitzades a través de materials, colliders, sons, animacions i comportaments implementats per nosaltres. A excepció dels tilemaps, els quals són estructures 2D.

2.1.1 Candy Cane

Aquest objecte fa les funcions de raqueta al joc original. Es mou verticalment seguint a la pilota amb l'objectiu d'impedir el pas. Quan la pilota hi col·lisiona, canvia de direcció horitzontalment. La seva velocitat depèn de l'espai disponible i quan la pilota s'apropa, aquesta augmenta i quan s'allunya torna al seu valor inicial.



Figure 1: Candy Cane

2.1.2 Spikes

Aquest objecte fa les funcions d'element perillós amb punxes. És estàtic, està enganxat a les parets del nivell i mata la pilota quan aquesta col·lisiona.

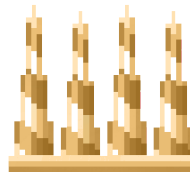


Figure 2: Spikes

2.1.3 Pumpkin

Aquest objecte és una variació de l'anterior. Es mou horitzontalment a un pas-sadís dificultant que la pilota el travessi. Si la pilota hi col·lisiona, mor i torna a la posició inicial del nivell. La velocitat amb la que es mou depèn de l'espai disponible.



Figure 3: Pumpkin

2.1.4 Buttons & Barriers

Aquests objectes fan les funcions de tancar possibles recorreguts als nivells. Les barreres estan implementades a través de dos *tilemaps* diferents. Una sempre està activada (impedint el pas) i l'altre desactivada (es pot travessar). Quan la pilota col·lisiona amb un botó, fa *toggle* de l'activació de les barreres. Aquesta implementació no és exactament com la del joc *QOMP* però ens ha servit d'inspiració.

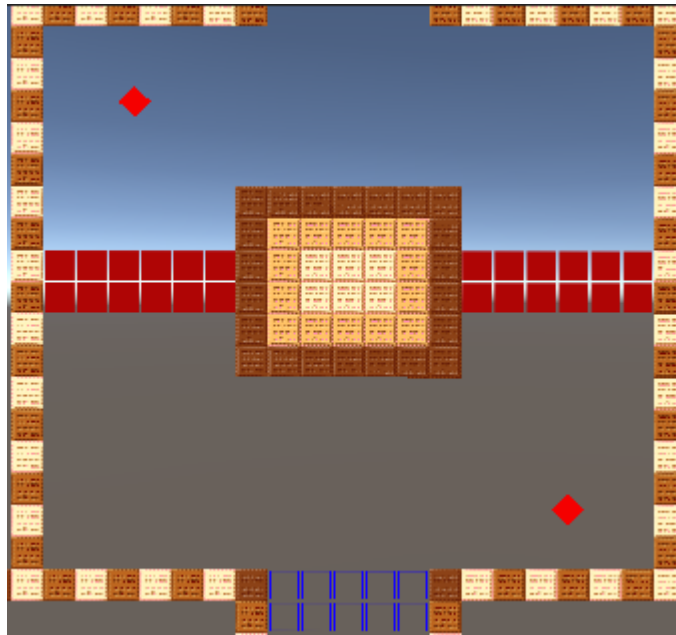


Figure 4: Buttons & Barriers

2.1.5 Door & Key

Aquests objectes fan les funcions de bloquejar possibles camins necessaris per aconseguir la llibertat. Si la pilota col·lisiona amb la porta abans de que ho faci amb la clau, canviarà de direcció horitzontal, si ho fa després, simplement passarà a través. Si la pilota col·lisiona amb la clau, aquesta desapareixerà. Cada clau pertany a la seva pròpia porta.



Key



Door

Figure 5: Door & Key

2.1.6 Chupa Chups

Aquest objecte fa les funcions de reward del nivell. Quan la pilota hi col·lisiona passa al següent nivell. Si el reward és el de l'últim nivell, hem aconseguit la llibertat i s'acaba el joc.



Figure 6: Chupa Chups

2.1.7 Sugus

Aquest objecte fa les funcions de pilota. Es mou tant verticalment com horitzontalment, el moviment horitzontal només pot ser modificat pel rebot donat de la col·lisió de la pilota amb elements del joc; el moviment vertical pot ser modificat pel jugador prement la barra espaiadora així com per elements del joc al col·lisionar.



Figure 7: Sugus

2.1.8 Tilemaps

S'han implementat 3 tipus de tilemaps diferents: 1 que fa les funcions d'acotar el nivell, és a dir, són les parets que delimiten les pantalles del lloc; i 2 més per implementar la funcionalitat de barreres i botons. Veure figura 4.

2.2 Instruccions

Les instruccions del joc són força senzilles:

- Prem la barra d'espai per canviar la direcció vertical del sugus.
- Intenta evitar les *spikes* i les *pumpkins*.
- Esquiva les *candy cane*.
- Resol els puzzles donats per les barreres vermelles i blaves i els diamants màgics.
- Troba les *candy keys* per obrir les *chocolate doors*
- Prem la tecla escape en tot moment per sortir al menú.
- Diverteix-te!

3 Metodologia

No es pot dir que hàgim seguit la millor metodologia durant el desenvolupament del joc, ja que no hem usat moltes de les eines disponibles que ens haguessin servit per organitzar-nos millor. Tot i això, no ens ha donat la sensació que hagi fet falta, ja que hi havia uns objectius molt clars en tot moment, amb una reunió setmanal per *Discord* i parlant ocasionalment via text per *Telegram* n'hi ha hagut prou.

A *Telegram* hi apuntàvem cada setmana els objectius a complir, i anàvem ratllant els elements en completar una tasca.

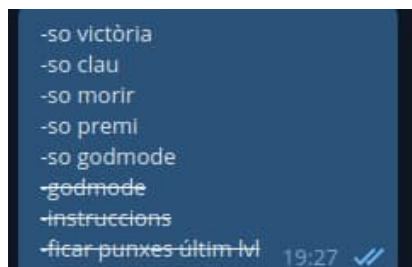


Figure 8: Llistat de tasques a *Telegram*

Al diagrama de Gantt de la figura 9, es pot veure els temes que parlàvem cada setmana, marcats amb blau i verd, a més del temps aproximat que vam tardar a implementar cada element.

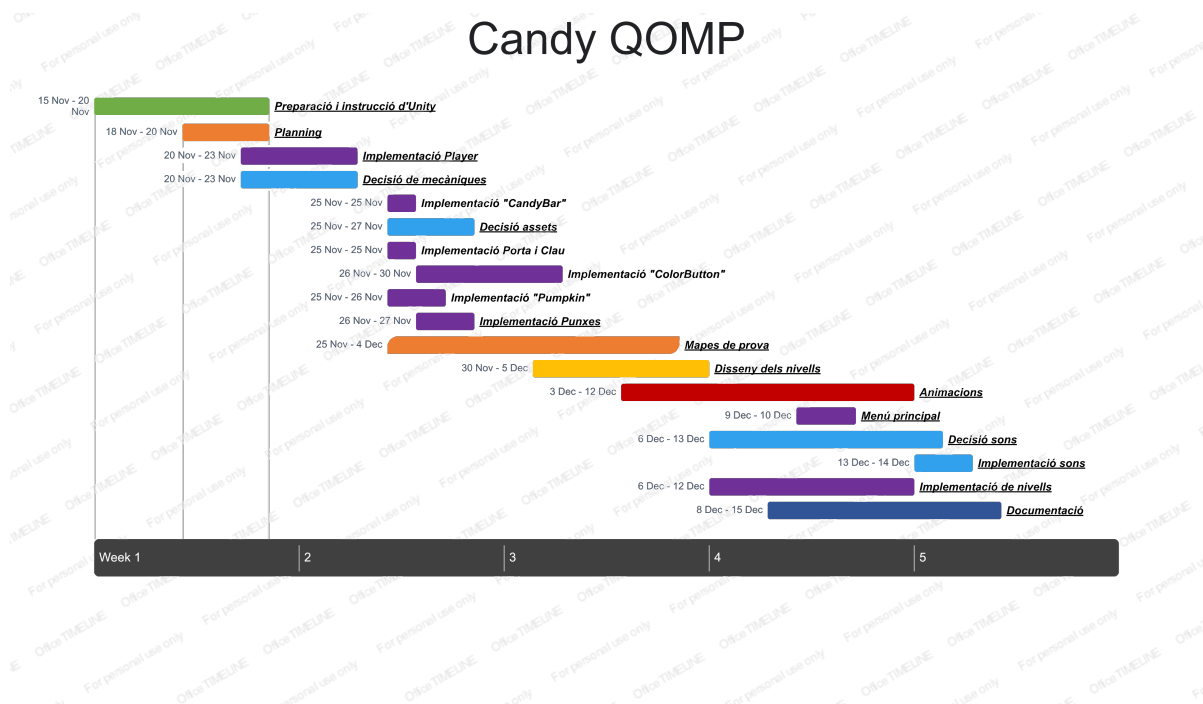
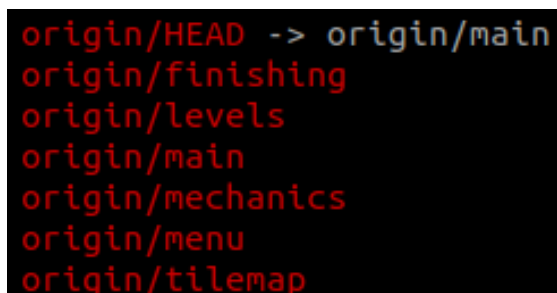


Figure 9: Diagrama de Gantt

A les *branches* de *git* també es pot observar la metodologia seguida, ja que usavem una *branch* per a cada element o objectiu que consideràvem important durant el desenvolupament del joc. En podem trobar principalment 5.

1. Mechanics: Aquesta va ser la *branch* on vam començar a implementar totes les mecàniques que es van decidir. Usavem un mapa de prova molt simple per anar fent les proves.
2. Tilemap: En aquesta *branch* vam implementar i provar l'ús del *tilemap* per a crear els nivells a partir de tiles.
3. Menú: Aquí vam crear el menú principal del joc, en el que es mostren instruccions del joc, els crèdits amb els nostres noms, l'opció de jugar i la d'escollir nivell.
4. Levels: En aquesta *branch* vam crear les *scenes* per a tots els nivells del joc on vam implementar el que ja havíem dissenyat prèviament.

5. Finishing: En aquesta *branch* es van fer els últims canvis per a deixar el joc més polit, es van afegir els sons als elements del joc i se'n van resoldre petits problemes que es van anar trobant al provar.



```
origin/HEAD -> origin/main
origin/finishing
origin/levels
origin/main
origin/mechanics
origin/menu
origin/tilemap
```

Figure 10: Llistat de *branches* de *git*

4 Conclusions

Com a conclusions, el que ens ha quedat més clar és que tot i que *Unity* és una bona eina per realitzar jocs fàcilment i que ofereix moltes funcionalitats, fa moltes coses pel seu compte i que per tant, és més difícil detectar i solucionar un error si no es té un coneixement molt extens del motor. Tot i això, donat que té una comunitat molt activa, és relativament senzill trobar les solucions a la majoria de problemes que trobes. També cal dir que hi ha funcionalitats, com la de les *scenes* que faciliten molt la creació de nivells i el manteniment d'aquests i la dels *prefabs* i *materials* que agilitzen molt la creació d'objectes i permeten personalitzar-los d'una manera ràpida i senzilla aportant varietat als diferents nivells. Dit això, hem de dir que vam gaudir més fent el joc amb *OpenGL*, ja que teníem més control sobre el joc i, en cas que es donés més temps per fer el joc 3D, sense dubte l'hauríem intentat fer amb *OpenGL*.

References

- [1] Preguntes freqüents sobre Unity. <https://answers.unity.com/questions/index.html>.
- [2] Snippets de codis d'exemple. <https://stackoverflow.com/>.