Alumne: Ferran Paredes Gayà
Tutor: Robert Pagès Bigarós

Programació d'un videojoc

Presentació
Quan juguem a un videojoc, per molt simple que pugui arribar a ser, no ens solem plantejar el treball que hi pot haver al darrere, per això en el meu treball de recerca he investigat com funciona la programació de videojocs en 2D per tal conèixer quines estructures i mètodes s'utilitzen i quines habilitats es necessiten. Evidentment, hi ha molts tipus de videojocs i cadascun d'ells pot estar estructurat d'una manera completament diferent. En la seva estructura hi influeix el tipus de motor de programació que s'ha utilitzat i les capacitats que té el programador. En la meva part pràctica he programat un videojoc fent servir el Godot, un motor de programació, i en la part teòrica he explicat els elements utilitzats en la part pràctica. Els motors de programació contenen moltes eines; per tant, explicar-ho tot comportaria molt de temps.

Metodologia
Inicialment no disposava de gaires coneixements sobre programació, i sobretot en el disseny de videojocs; per tant, per començar a desenvolupar el videojoc vaig mirar diversos videotutorials explicatius al YouTube on programaven jocs i et mostraven com funciona el Godot, el motor de programació que he usat. Vaig començar mirant-ne

de simples i progressivament vaig anar augmentant la complexitat fins al punt que notava que ja podria desenvolupar-ne un de prou complex utilitzant alguns mètodes utilitzats en els tutorials. El Godot també inclou una documentació on s'expliquen les rutines i les eines proporcionades per tal de facilitar les coses i permetre que gent inexperta pugui arribar a aprendre a utilitzar el motor. Amb tots aquests recursos vaig aprendre a emprar el llenguatge de programació propi del Godot, el GDScript i les eines que aquest proporciona. A més, el meu pare hi entén d'informàtica i també m'ha ajudat a solucionar problemes que he anat trobant i a proposar idees per incloure en el joc.

Cos del treball

Com he mencionat abans, en la part teòrica només he explicat les eines que he utilitzat en la part pràctic; per tant, cal remarcar que hi ha moltes altres eines i mètodes disponibles per desenvolupar un videojoc.

Per programar un videojoc es necessita un motor de programació, que és una eina que facilita el desenvolupament de videojocs. Es podria programar un videojoc començant des de zero simplement usant un llenguatge de programació, però això comportaria molta feina. Els motors de programació contenen eines i rutines ja programades amb les quals pots treballar i complementar amb els teus propis scripts (talls de codi). Utilitzant únicament les rutines i eines proporcionades pel motor no es podria programar un videojoc, sinó que es necessiten scripts propis per al seu funcionament. Els motors de videojocs inclouen un motor gràfic per renderitzar gràfics 2D i 3D i un motor físic que simula les lleis de la física de forma realista mitjançant un conjunt de funcions i variables que actuen en els diferents objectes del videojoc en el moment de la seva execució, com la gravetat, la massa, la fricció, la força i les col·lisions.

Cal saber que els videojocs s'executen en un bucle, és a dir que a tantes vegades per segon el motor actualitza el món virtual, detecta les entrades (inputs), calcula tot el que s'ha mogut o ha canviat i ho dibuixa a la pantalla. Normalment això sol passar 60 frames per segon (fps) tot i que pot variar. D'aquesta manera, s'obté la il·lusió de moviment en un videojoc.

El Godot Engine ha estat el motor de programació que he utilitzat per programar el meu primer videojoc. Aquest és un dels motors més recomanables per gent que vol començar a desenvolupar videojocs, ja que és totalment gratuït, sense cap clàusula addicional, i proporciona animacions integrades, físiques en 2D i 3D, ocupa molt poc espai (menys de 40 MB) i té el seu propi llenguatge de programació, el GDScript, tot i que també es poden fer servir altres llenguatges com el C# o C++. A més, té una gran comunitat disposada a compartir els seus projectes i ajudar a solucionar problemes en els projectes d'altres persones. El funcionament del Godot es basa en una sèrie de blocs de construcció essencials i altres eines: els nodes, les escenes, les instàncies, els senyals, les capes i les màscares. Els nodes són blocs de construcció fonamentals per crear un joc. Realment són el que en programació s'anomenen objectes. Hi ha molts tipus de nodes, cadascun amb una varietat de funcions especialitzades. Els nodes tenen un nom, propietats editables, mètodes propis, i es poden estendre mitjançant scripts propis. Es pot afegir en un altre node com un fill (node fill), de tal manera que s'adopta una forma d'arbre. Així doncs, sempre hi ha un node principal que serà una escena, que està composta per un grup de nodes organitzat jeràrquicament. Les escenes es poden desar al disc amb l'extensió .tscn, es poden carregar (reutilitzar) i se'n poden fer instàncies. Bàsicament, l'editor del Godot és un editor d'escenes. Ofereix moltes eines per editar escenes en 2D i 3D i interfícies d'usuari.

Les instàncies són un mecanisme que permet crear còpies d'una plantilla, que sol ser una escena, de manera programàtica i executar-les tantes vegades com es vulgui dins d'una escena en un determinat moment. S'utilitza, sobretot, en projectes complexos per organitzar-los, ja que et permet editar múltiples instàncies a la vegada.
Els senyals permeten a un node enviar un missatge quan unes determinades circumstàncies succeeixen, i qualsevol altre node podrà rebre aquest missatge i respondre. Un bon exemple d'això seria quan es clica un botó o un cos entra en una àrea. En aquests casos, s'envia un senyal al node que volem que respongui executant una funció.
Finalment, tots els objectes del videojoc es troben organitzats en un sistema de capes de col·lisió. Aquest sistema permet construir interaccions complexes entre una varietat d'objectes. El Godot permet definir fins a vint capes de col·lisió diferents. Cada objecte pot assignar-se a qualsevol d'aquestes capes i al mateix temps se li pot indicar amb quines capes interactua (màscara).
El Godot disposa d'un llenguatge de programació propi, el GDScript. Com tots els altres llenguatges, conté variables, funcions, constants, instruccions i operadors, però com que és un llenguatge utilitzat únicament per a la programació de videojocs, proporciona variables i funcions ja programades per a cada node per tal de facilitar el desenvolupament del videojoc.
En la meva part pràctica he programat un videojoc que he anomenat «Ninja Trivial». És un videojoc de plataformes en el qual hi ha enemics i obstacles. Per passar de nivell, s'han de col·leccionar tres claus que es troben en tres tresors escampats pel nivell. Un cop s'obren els tresors, apareixerà una pregunta tipus test. Aquestes preguntes poden ser de geografia o de ciència. Si s'encerta la pregunta, s'obté una clau. Si no s'encerta, es torna a començar des de l'inici. El videojoc està compost per dos nivells i el jugador disposa de tres vides i deu punyals, tot i que un cop es passa de nivell, es tornen a reiniciar. Al llarg dels nivells hi ha monedes i diamants que et donen puntuació, igual que si es mata algun enemic. El jugador és un ninja que pot atacar amb una espasa o amb uns punyals limitats. Els enemics són zombis i genis que poden matar el jugador amb els seus atacs. Aquest últim deixa anar boles que poden matar el jugador. Per desenvolupar el joc he hagut de dissenyar els nivells amb un node anomenat TileMap, incorporar decoracions i col·locar les monedes, els tresors, els enemics i els obstacles en posicions determinades, programar les físiques dels objectes, programar el temporitzador de puntuació, de vides, de claus i de punyals, proporcionar animacions i efectes de so, programar els atacs dels enemics i del jugador, generar preguntes de manera aleatòria i procurar que no es repeteixin... Tot això són detalls que no es tenen presents quan es juga al joc.

Conclusions

Gràcies a aquest treball he pogut veure que programar un videojoc comporta temps,

molta creativitat i una gran capacitat per solucionar problemes, a part de coneixements sobre física, lògica i anglès. El treball m'ha servit per introduir-me en la programació, que és el que m'agradaria estudiar en el futur, i he pogut treballar conceptes lògics i físics. Abans de realitzar aquest treball només havia treballat els conceptes bàsics del llenguatge HTML, però mai havia programat un videojoc utilitzant codi. Simplement n'havia programat de molt simples utilitzant algunes pàgines web que proporcionaven funcions predefinides. Aquest treball també m'ha servit per desenvolupar la capacitat per resoldre problemes, ja que al llarg del desenvolupament del videojoc m'he trobat amb molts bugs difícils d'identificar que m'han fet perdre bastant de temps i he hagut de buscar en diferents llocs per tal de trobar una solució. També m'ha servit per desenvolupar la meva creativitat a l'hora de dissenyar els nivells i els conceptes del videojoc.

Bibliografia i bibliografia web

– Wikimedia Foundation. (14 jul. 2020). Motor de videojuego. Wikipedia. <https://tuit.cat/L0OO0> [Consulta: 23 ag. 2021] – BBVAAPI_Market. (1 gen. 1970). Los mejores motores gráficos de videojuegos (I): soluciones de código abierto. <https://tuit.cat/w3ML8> [Consulta: 25 ag. 2021] L. (2020, July 3). Godot Engine vs Unity, Unreal, o Game Maker. La Cueva del Lobo. <https://tuit.cat/jafjZ> [Consulta: 31 ag. 2021] – The 4 essential building blocks. (n.d.). GDQuest. <https://tuit.cat/kGboV> [Consulta: 12 jul, 2021] – Scenes and nodes. (n.d.). Godot Engine Documentation. <https://tuit.cat/1nfbh> [Consulta: 2 set. 2021] – Instancing. (n.d.). Godot Engine Documentation. <https://tuit.cat/5vb3y> [Consulta: 2 set. 2021] – Signals. (n.d.). Godot Engine Documentation. <https://tuit.cat/72bvf> [Consulta: 4 set. 2021] Collision Layer vs Mask. (10 maig 2020). Godot Engine - Q&A. <https://tuit.cat/i17zK> [Consulta: 8 set. 2021] – KinematicBody2D. (n.d.). Godot Engine Documentation. <https://tuit.cat/7ugsW> [Consulta: 6 set. 2021] – RigidBody2D. (n.d.). Godot Engine Documentation. <https://tuit.cat/wUrVG> [Consulta: 7 set. 2021] – Sprite. (n.d.). Godot Engine Documentation. <https://tuit.cat/xmyE2> [Consulta: 6 set. 2021]
CollisionShape2D. (n.d.). Godot Engine Documentation. <https://tuit.cat/aN5WW> [Consulta: 6 set. 2021] – Area2D. (n.d.). Godot Engine Documentation. <https://tuit.cat/gBfiz> [Consulta: 6 set. 2021] VisibilityEnabler2D. (n.d.). Godot Engine Documentation. <https://tuit.cat/48c3F> [Consulta: 7 set. 2021] – Camera2D. (n.d.). Godot Engine Documentation. <https://tuit.cat/0QAVh> [Consulta: 7 set. 2021] – Using TileMaps. (n.d.). Godot Engine documentation. <https://tuit.cat/0gvw1> [Consulta: 7 set. 2021] – Timer. (n.d.). Godot Engine Documentation. <https://tuit.cat/zr20b> [Consulta: 7 set. 2021] AnimationPlayer. (n.d.). Godot Engine Documentation. <https://tuit.cat/ltdin> [Consulta: 7 set. 2021] – GDScript basics. (n.d.). Godot Engine Documentation. <https://tuit.cat/jvYCj> [Consulta: 10 set. 2021] – Make Your First 2D Game with Go-

dot: Player and Enemy (beginner tutorial part 1). (2 oct. 2019). YouTube. <https://tuit.cat/rGJbp> [Consulta: 8 jul. 2021] – Make Your First 2D Game with Godot: Coins, Portals, and Levels (beginner tutorial part 2). (21 nov. 2019). YouTube. <https://tuit.cat/2OEar> [Consulta: 8 jul. 2021] Make Your First 2D Game with Godot: Menus, Pause, and Score (beginner tutorial part 3). (5 des. 2019). YouTube. <https://tuit.cat/Cw2tT> [Consulta: 8 jul. 2021]