Universitat de Barcelona

Treball de Fi de Grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

SAVE ALL THE ROBOTS

Pau Sanchez Valdivieso

Director: Dr. Santi Seguí Mesquida
Realitzat a: Enginyeria Informàtica

Barcelona, 1 de gener de 2018

ÍNDEX

CAPÍTOL 1.

INTRODUCCIÓ I MOTIVACIÓ

1.1. MOTIVACIÓ

Durant l'assignatura de Projecte Integrat de Software vaig tenir el plaer de desenvolupar un joc per la plataforma Android que es va dir Save All The Robots. El joc consistia a salvar un determinat nombre de robots dels atacs dels enemics.

Save All The Robots és un joc basat en algorismes sobre grafs i matrius i s'utilitza un taulell de 8x8 cel·les com a terreny de joc. És un joc basat en posicions, moviments i amenaces que està inspirat en els escacs.

L'assignatura es va fer en grups de 4 persones i tot i que vaig ser un pilar fonamental en el desenvolupament del joc, hi va haver aspectes que calia decidir entre els 4 membres, i en conseqüència, aspectes que hagués desenvolupat d'una manera diferent.

Durant el desenvolupament de l'aplicació per Android vaig començar a tenir molt clar que aquest seria un bon tema per dur a terme el meu TFG, incloent moltes més idees que no van poder tenir cabuda, degut al poc temps de desenvolupament que permet una assignatura petita com *Projecte Integrat de Software*, i el que és més important, portant a terme el desenvolupament sobre les plataformes en què tinc més interès, les aplicacions web.

Així doncs la meva passió per l'algorísmica, concretament sobre grafs i matrius, i l'estima per un joc que jo mateix vaig tenir el plaer de construir una vegada, m'han portat a desenvolupar el Treball de Fi de Grau sobre *Save All The Robots* en la seva versió online per navegador, sota la tutela del professor de l'assignatura que va veure néixer la primera versió del joc.

Cal dir que, malauradament, la primera versió del joc mai va tenir l'oportunitat de sortir de la fase beta en què va quedar a l'acabar l'assignatura. Tot i així els 4 membres del grup vam rebre Matricula d'Honor pel joc.

En els capítols finals s'explicaran les diferències entre les dues versions del joc.

1.2. BREU INTRODUCCIÓ

S'aprofita aquest apartat per explicar, i tenir clar des de l'inici quin és el context del Treball de Fi de Grau i quines tecnologies s'utilitzen pel seu desenvolupament.

Aquesta nova versió, a diferència de l'antiga, està desenvolupada mitjançant la convivència de 5 llenguatges bàsics en aplicacions web: PHP, Html, CSS, JavaScript i MySQL. Així doncs no es reaprofita cap tros de codi ni cap algorisme de la primera versió del joc, a més, com veurem s'inclouen protagonistes nous amb noves lògiques que impedeixen el reaprofitament de codi.

S'ha triat desenvolupar el joc per plataforma web pel motiu que pot ser fàcilment accedit des de qualsevol dispositiu independentment del sistema operatiu utilitzat, obrint a qualsevol usuari la possibilitat de jugar i adquirir l'afició de salvar robots. No obstant, com en la majoria d'aplicacions, la versió per PC és molt més bona i jugable que la versió per mòbil. Una altra de les raons de pes és que els llenguatges de desenvolupament triats són aquells que tinc més interès a aprendre'n tots els secrets.

Un altre dels avantatges de les aplicacions per navegador és que no es necessiten descàrregues ni instal·lacions addicionals per a poder utilitzar l'aplicació. Ni ha prou en tenir un dispositiu i una bona connexió a internet. Com veurem més endavant en aquest document, aquest tipus d'aplicacions també té alguns petits inconvenients, encara que no són tant grans com les múltiples avantatges que proporcionen.

CAPÍTOL 2.

OBJECTIUS

2.1. Què és Save All The Robots?

A continuació s'explica el funcionament del joc. D'entrada s'introdueix el joc des d'una perspectiva global i mica en mica s'aprofundeix en cada concepte i el seu detall.

El joc es troba online, cosa que fa més fàcil comprovar i testejar tot el que s'explica en aquest document seguint el següent enllaç: https://www.savealltherobots.com

Es comença explicant el funcionament del joc i a partir del coneixement d'aquest, s'obren diverses branques que expliquen les diferents modalitats del joc. Els detalls de l'aplicació, organització i desenvolupament s'explicaran en capítols posteriors.

Tot el temari està descrit en l'ordre que s'ha cregut que facilitaria més al lector la comprensió del funcionament del joc. Durant el transcurs de les descripcions s'inclouen imatges i captures de pantalla bàsiques per posar al lector en situació fent-li més fàcil i senzilla la comprensió del funcionament del joc.

2.2. Resum breu i global del joc

Save All The Robots és un joc trencaclosques inspirat en els escacs. Aquesta semblança rau en el fet de ser un joc d'estratègia posicional basat en posicions, moviments i amenaces.

S'utilitza un tauler de 8x8 caselles on cal col·locar estratègicament un determinat nombre de robots, situats inicialment a l'inventari, en un temps determinat.

Damunt el tauler s'hi troben també els enemics, que de bon principi, estan amenaçant certes posicions. Caldrà evitar aquestes posicions o bé protegir-se per a fer-ne ús!

EL joc està basat en nivells de dificultat incremental, de manera que un nivell es considera superat només si s'han col·locat sobre el tauler tots els robots de l'inventari, i una vegada esgotat el temps, s'ha aconseguit salvar-los a tots. En qualsevol altre cas el nivell es considera 'no superat'.

Els robots, els enemics i el temps estan imposats per la dificultat del nivell i seran diferents per a cadascun d'ells.

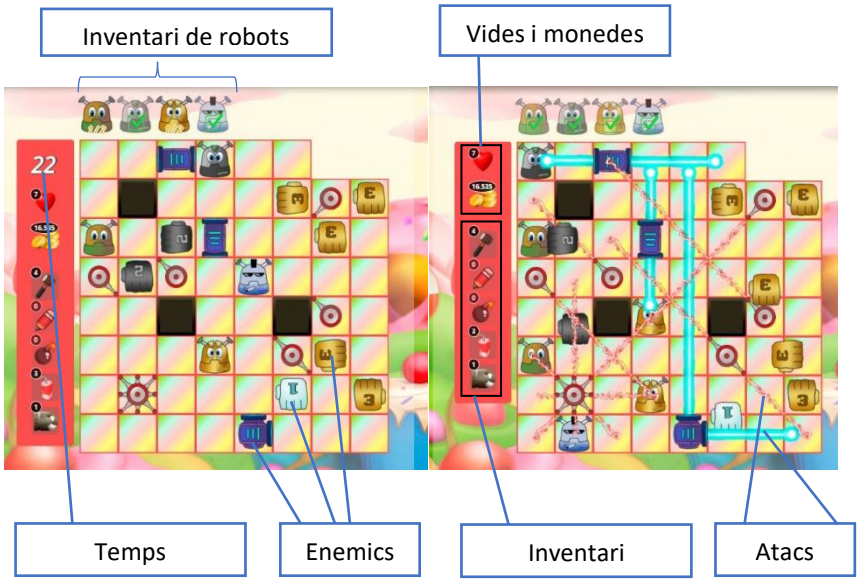


Figura 1. Taulell del joc

A la figura 1 es mostra el tauler del joc així com també l'inventari dels robots, els enemics i el temps. L'inventari de robots és la fila situada a la part superior que conté els diferents tipus de robots que es poden arrossegar per desar-los al taulell de joc. Els enemics estan sempre col·locats damunt del taulell abans que l'usuari comença a jugar. El temps està situat a l'esquerra juntament amb les vides, les monedes i les armes, que s'explicaran més endavant.

La imatge de l'esquerra de la figura 1 correspon a un nivell del joc en el moment en què el jugador està jugant tot i col·locant els robots damunt del taulell a les cel·les que creu que són segures, és a dir, que no estan amenaçades pels enemics.

La imatge de la dreta, en canvi, mostra l'atac dels enemics en el moment en que el temps s'ha esgotat, en el mateix nivell del joc.

2.3. Modalitats de joc

Save All The Robots ofereix tres modalitats de joc a l'usuari, que són la modalitat 'Original', la modalitat 'Arcade' i la modalitat 'Nivells Personalitzats', partint de la modalitat 'Original' com a base i pilar fonamental del joc.

2.3.1. Modalitat Original

És la modalitat principal del joc, aquella que és més semblant a qualsevol joc 'trenca-closques' on la motivació principal no és altra que superar els nivells dels diferents mons del joc.

A cada món es desbloquegen 2 nous enemics i 1 nou robot. Per a poder jugar en aquesta modalitat es necessiten vides i és imprescindible tenir-ne almenys una per a poder accedir a jugar. D'altra banda, cada vegada que no se supera un nivell es perd una vida. (Més endavant, en aquest document es detalla a fons el sistema de vides).

La modalitat original del joc està ambientada en diferents mons que són els següents: beach, ocean, jungle, village, sugar, city, Love i space. Cada món té un color de taulell acord amb el tema que tracta i està personalitzat de manera diferent.

A continuació es mostren els 8 mons o zones del joc. Cadascuna de les zones està composada per 9 nivells. També es mostren els nivells del primer món i un nivell a mode d'exemple.

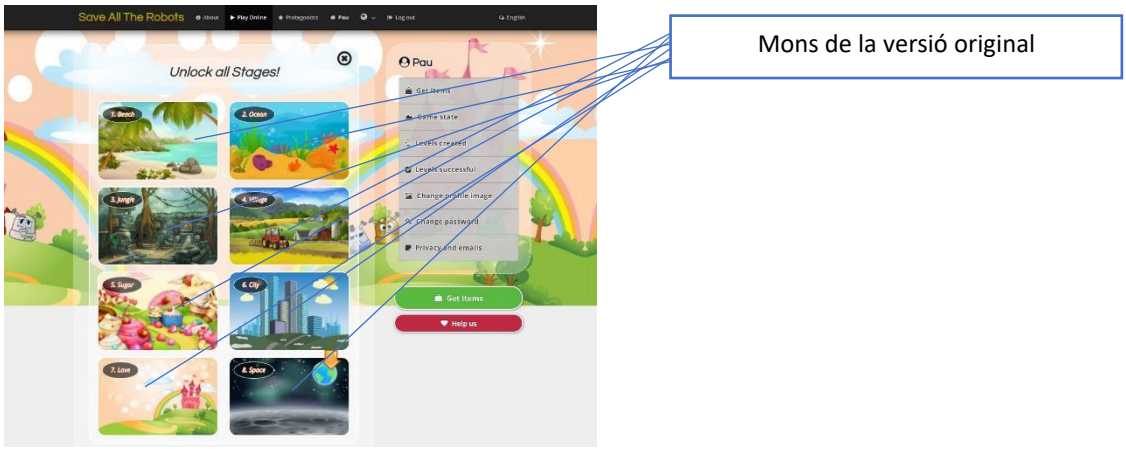


Figura 2. Mons o zones del joc

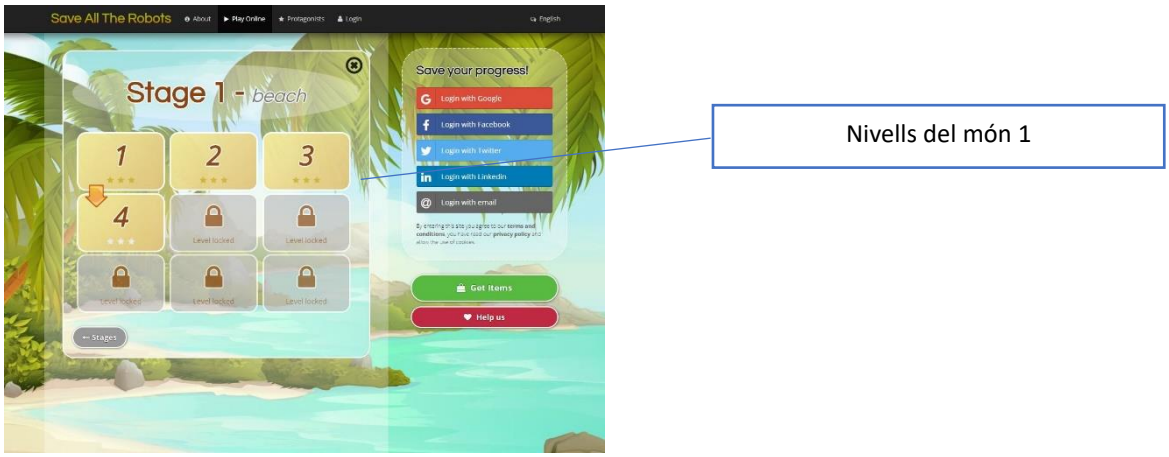


Figura 3. Nivells del primer món

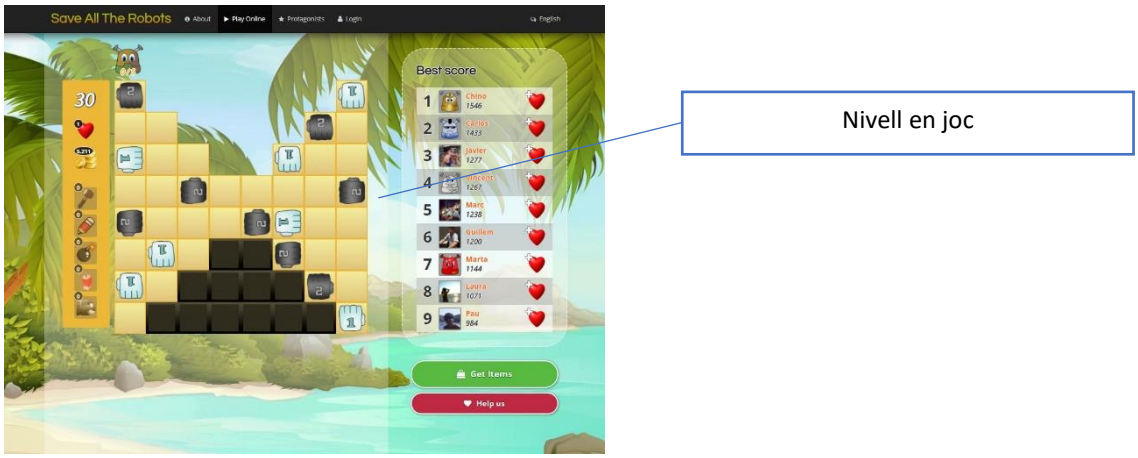


Figura 4. Exemple del quart nivell del primer món

Tant els mons con els nivells es van desbloquejant a mesura que l'usuari avança en el joc superant els diferents nivells proposats. Així doncs la modalitat original del joc acaba quan s'han superat els 9 nivells dels 8 mons, és a dir, els 72 nivells de la modalitat.

Un jugador podrà abandonar la partida en el moment que vulgui i quan torni a jugar el sistema recordarà el món, el nivell, les vides i tota la informació de la partida del jugador. (Això est detallarà a fons més endavant).

2.3.2. Modalitat Arcade

La modalitat Arcade està ambientada d'una manera diferent. D'entrada és un ambient més fosc i les cel·les tenen un aspecte més 'retro'. S'intenta emular la visió de les màquines recreatives dels anys 90. El món on es du a terme cada nivell és sempre aleatori, de manera que a cada nivell superat es canvia sempre l'ambientació del joc.

A la modalitat Arcade els nivells de joc són infinits i es generen de forma instantània i de manera algorísmica, són aleatoris i de dificultat incremental.

La motivació principal és assolir el màxim nivell i aconseguir el rècord absolut. A cada nivell es desbloqueja un nou enemic o un nou robot. Això fa que el joc en sí sigui més ràpid que en la modalitat original ja que l'usuari pot descobrir tot els actors del joc superant menys nivells (recordar que a la modalitat original es desbloquejaven 3 personatges per zona ja siguin enemics o robots, és a dir, 3 personatges cada 9 nivells).

Un altre punt important és que no es necessiten vides per jugar. Cada 5 nivells superats de forma consecutiva s'aconsegueix una vida que es podrà fer servir a la modalitat original del joc, anteriorment esmentada.

Un altre aspecte rellevant és que en aquesta modalitat, quan un jugador perd un nivell haurà de tornar a començar al nivell 1. El rècord s'obté per nivells consecutius guanyats!

La modalitat arcade es pot interpretar de dues maneres:

1. Com a modalitat de suport de la modalitat original, és a dir, es pot entendre com una manera d'aconseguir vides quan aquestes s'esgoten a la modalitat original. D'aquesta manera un jugador mai es queda 'tirat' de vides havent de deixar de jugar, sinó que pot anar combinant les dues modalitats sense cap mena de límit.

2. Com la modalitat preferida, és a dir, hi haurà jugadors que se sentiran més motivats per la modalitat arcade volent aconseguir el rècord absolut, que no pas per la modalitat original del joc.

A continuació (figura 5) es mostren exemples visuals de la modalitat Arcade:

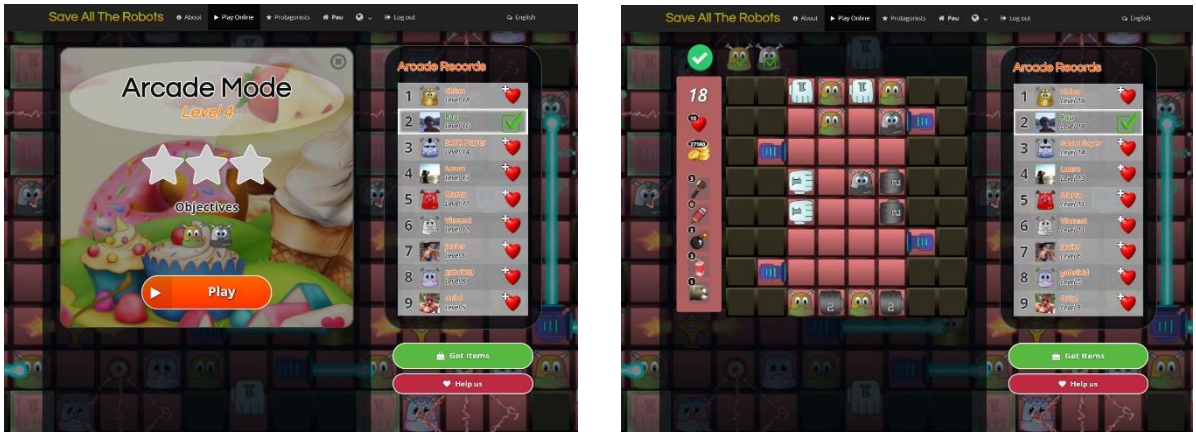


Figura 5. Exemples visuals de la modalitat Arcade

2.3.3. Modalitat de Nivells Personalitzats

Aquesta modalitat es divideix en dues sub-modalitats clarament diferenciades però molt relacionades entre elles:

1. **Creació de nivells personalitzats**: Es permet a l'usuari crear els seus propis nivells mitjançant un editor online que funciona per 'drag and drop'. L'usuari arrossega robots i enemics definint el seu propi nivell.

   Una vegada l'usuari acaba d'editar el seu nivell, el fa públic tenint l'opció de convidar altres usuaris de Save All The Robots a jugar-hi i guanyar la recompensa que el nivell ofereix segons l'estimació de la seva dificultat. D'altra banda el creador del nivell guanya una vida cada per cada 5 jugadors diferents que hi juguin.

Figura 6. Editor de nivells personalitzats

2. **Superar nivells d'altres usuaris**: L'altra cara de la moneda és superar nivells que han estat creats per altres usuaris i aconseguir la recompensa que ofereix cadascun d'ells, a més de millorar les estadístiques de joc i afegir un nivell més a l'historial de nivells superats. La recompensa que ofereix un nivell personalitzat no la tria el seu creador, sinó que es calcula mitjançant un algorisme tenint en compte la dificultat del nivell i la quantitat de jugadors que hi ha jugat. Així doncs les recompenses que oferiran els nivells sovint seran diferents.

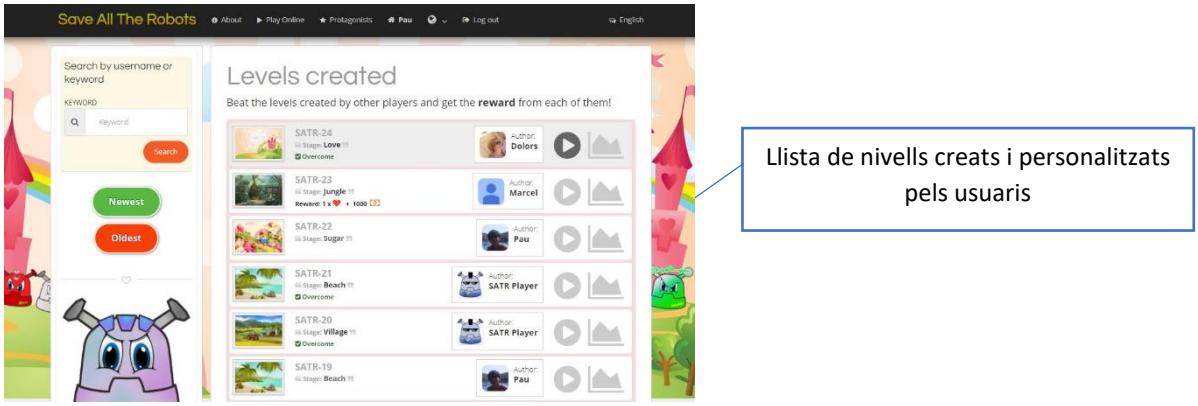


Figura 7. Nivells creats per usuaris

2.4. Resum de les modalitats de joc

Modalitat	Motivació
Original	Superar tots els nivells de tots els mons i arribar al final del joc.
Arcade	Aconseguir el rècord absolut (màxim nivell consecutiu assolit) i aconseguir vides que podran ser utilitzades a la modalitat original
Crear nivells personalitzats	Posar a prova l'habilitat estratègica i la creativitat editant un nivell personalitzat i reptar altres usuaris a intentar superar el nivell.
Jugar nivells personalitzats	Superar nivells creats per altres usuaris aconseguint la recompensa basada en un nombre determinat de vides i monedes segons la dificultat del nivell.

Taula 1. Resum de les modalitats de joc

2.5. Objectiu de vides

Com succeeix a una gran quantitat de jocs, Save All The Robots utilitza un sistema de vides que el jugador ha de conèixer per tal d'optimitzar al màxim les seves partides.

L´única modalitat del joc que consumeix vides és el mode 'Original'. És així degut a que aquesta modalitat és el pilar fonamental i la motivació principal del joc.

Un jugador comença a jugar tenint 5 vides i el màxim de vides que pot acumular són 10. Quan un jugador perd un nivell en la modalitat 'Original' consumeix una vida. Aquesta és l´única manera en què un jugador pot perdre una vida, en canvi existeixen moltes maneres de recuperar vides de forma gratuïta.

Si un jugador abandona una partida a mitges, el nivell al qual estava es considera 'no superat' i a l'usuari se li descompta automàticament una vida. (De no ser així seria molt fàcil fer trampes i refrescar el navegador quan saps del cert que ja no podràs superar el nivell).

Com guanyar vides:

Cada dia, a les 6:00 del matí i de manera automàtica, es regeneren les vides de tots aquells jugadors que en tenien menys de cinc fent que tornin a tenir 5 vides. Si un jugador tenia en possessió cinc vides o més no se li fa cap actualització.

Quan un jugador es queda sense vides pot jugar a la modalitat 'Arcade', que té nivells infinits de dificultat incremental i per cada 5 nivells consecutius superats guanya una vida.

D'altra banda un jugador pot crear un nivell personalitzat i convidar altres usuaris a superar-lo de manera que per cada cinc usuaris diferents que provin el seu nivell creat, el creador guanyarà una vida.

Un usuari que es queda sense vides també té l'opció de jugar als nivells creats per altres usuaris, i en cas de superar-lo, obtenir la recompensa del nivell. (Les recompenses dels nivells creats pels usuaris estan formades per vides i monedes de joc).

A més, de manera no gratuïta, un usuari té l'opció de comprar vides a la botiga del joc mitjançant targeta de crèdit o PayPal.

2.6. Protagonistes del joc

Tot seguit es detallen els protagonistes del joc, que no són pas altres que els robots, els enemics i les armes o ítems del joc.

2.6.1. Robots

Robot Simple: Té resistència 0 als atacs, és a dir, si rep l'atac d'un enemic mor a l'acte fent perdre la partida al jugador.

Robot Metàl·lic: Té resistència 1 als atacs, és a dir que pot rebre un cop i seguir viu. A més es pot fer servir d'escut per protegir els robots simples dels atacs dels enemics.

Robot Daurat: Té resistència 2 als atacs i com el robot metàl·lic, també es pot utilitzar d'escut per protegir altres robots.

Robot Diamant: Té resistència 3 als atacs. És el robot més resistent, cosa que permet que com els dos anteriors, pugui ser utilitzat com escut per protegir altres robots.

Robot Pinxo: Si es col·loca un robot d'aquest tipus al costat d'un robot resistent (metàl·lic, daurat o diamant), aquest últim perd la seva resistència als atacs convertint-se en un robot simple.

Robot Desactivador: Té l'habilitat de desactivar els enemics que té al costat privant el seu atac final. Té resistència 0 als atacs i pot morir a causa dels atacs dels enemics que no desactiva.

Robot Explosiu: Aquest robot explota quan acaba el temps destrossant els enemics del seu costat, però l'explosió també mata als robots que té a la vora! Cal col·locar-lo allunyat dels altres robots!

Robot Girl: Aquest robot pot decidir canviar de cel·la quan acaba el temps. Té un 50% de probabilitat de canviar de cel·la. Si decideix canviar només podrà anar a una de les 8 cel·les que té per veïnes.

2.6.2. Enemics

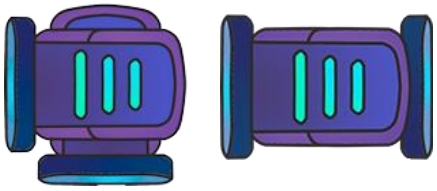
Puny simple: És l'enemic més simple de joc. Ataca amb abast d'una casella en la direcció a la que apunta. El seu atac només pot ser horitzontal o vertical, mai ataca en diagonal.

Puny Doble: Té exactament el mateix tipus d'atac que el puny simple però el seu abast és de 2 caselles en la direcció a la que apunta.

Puny triple: Té el mateix atac que els dos punys anteriors però el seu abast és de 3 caselles en la direcció en la que apunta.

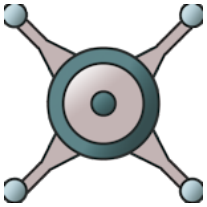
Làser simple: Ataca en direcció perpendicular i té abast infinit, és a dir, ataca fins que troba un objecte, ja sigui un altre enemic, un robot o un bloc. Pot ser bloquejat pels robots resistents.

Làser quàdruple: Té atac perpendicular en les quatre direccions i abast infinit. Es pot entendre el seu funcionament com el de la torre al joc d'escacs.

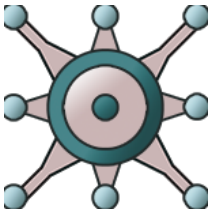


Làsers Bidireccionals: Ataquen com els altres dos làsers però ho fan únicament en dues direccions.

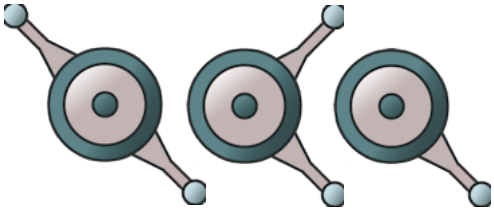
Llença-flames: Té abast 1 i ataca en les 4 direccions perpendiculars a la vegada. No pot ser bloquejat pels robots resistents ja que el seu abast és 1, però aquests poden resistir el seu atac.



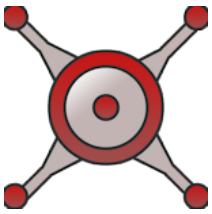
Elèctric diagonal: Ataca en les 4 direccions diagonals i té abast 1. No té sentit bloquejar-lo amb les robots resistents ja que té abast 1 però aquests poden resistir el seu atac.



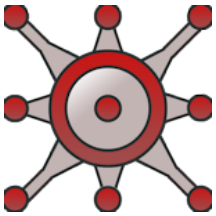
Elèctric múltiple: Ataca en totes direccions i té abast 1. El seu atac és idèntic a l'elèctric anterior però fent-ho en totes direccions, tant perpendiculars com diagonals. Es pot entendre el seu atac com el del rei al joc d'escacs.



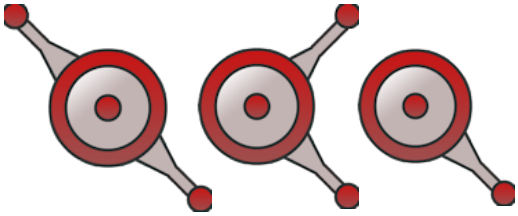
Resta de la família d'elèctrics: El seu atac és diagonal en la direcció a la que apunten i abast 1.



Elèctric diagonal infinit: Ataca en les 4 direccions diagonals amb abast infinit, és a dir fins que troba un altre objecte, ja sigui un altre enemic, un bloc o un robot. Pot ser bloquejat pels robots resistents. El seu comportament es pot entendre com el d'un alfil al joc d'escacs.

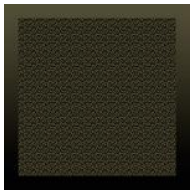


Elèctric múltiple infinit: Ataca en totes direccions i té abast infinit. Pot ser bloquejat pels robots resistents. Emula l'actitud de la reina al joc d'escacs.



Resta de la família d'elèctrics múltiples: Ataquen exactament igual que els dos anteriors però en la direcció a la que apunten.

2.6.3. Blocs



El bloc no és ni un enemic ni un robot. El bloc representa un espai inaccessible tant pels enemics com pels robots, és a dir, un espai ple i inutilitzable. Un bloc sempre fa d'escut als atacs dels enemics amb els que fa intersecció. Per tant un robot amaga darrere un bloc, és un robot que està protegit en aquella direcció.

2.6.4. Blancs

Els blancs són espais al taulell de joc que són invisibles, és a dir que pertanyen a la forma del taulell però no poden ser ocupats per cap protagonista. La seva funció és exactament igual que la d'un bloc canviant només la seva aparença. En aquest document s'ha triat la 'B' com a símbol per a poder representar-lo però en realitat és un espai invisible.

2.6.5. Armes o ítems

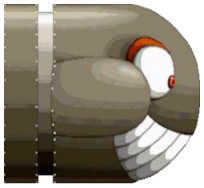
Les armes són ajudes que s'adquiriran a la botiga del joc mitjançant monedes de joc. En aquest punt encara no s'ha parlat ni de les armes, ni de la botiga ni de les monedes. No obstant, les armes formen part dels actors del joc i es convenient conèixer-les en aquest punt del document juntament amb la resta dels protagonistes.

Martell: és capaç de trencar un bloc habilitant una nova cel·la per col·locar, si escau, un robot.

Llapis: pot dibuixar una nova cel·la on hi havia un blanc tot i habilitant un nou espai per col·locar un robot.

Dinamita: aquest ítem explota a la cel·la on és col·locat per l'usuari. Aquesta explosió, a banda d'afectar a la cel·la, afecta també a les 8 cel·les veïnes destrossant els enemics que troba. Els robots no moren a aquesta explosió.

Bomba: aquesta arma explota a la cel·la on és col·locada per l'usuari i té una ona expansiva infinita, és a dir, fins que troba un bloc o fins que acaba el taulell. Destrossa tots els enemics que troba en les 4 orientacions perpendiculars. Els robots sobreviuen a l'explosió.



Míssil: El míssil destrossa tot el que troba a la fila on és col·locat per l'usuari. Trenca tots els blocs que troba i a més mata als enemics. Es considera l'arma més potent de totes sent la més cara a la botiga d'armes.

2.7. Objectius del joc

Ara que el lector ja coneix les regles bàsiques del joc, el sistema de vides i els protagonistes involucrats és més fàcil parlar de quins són els objectius del joc.

2.7.1. Objectiu global

Si bé és cert que cada modalitat té una motivació i uns objectius globals diferents, totes tenen un objectiu comú: **col·locar els robots damunt del taulell i aconseguir que cap robot sigui destruït per un enemic quan acabi el temps.**

Al taulell de joc existeixen posicions que estan sota l'amenaça de l'atac dels enemics, però, aquestes cel·les no es diferencien pas d'aquelles que no estan amenaçades i que són segures. Ha de ser doncs l'habilitat del jugador i la seva capacitat estratègica la que determini de manera visual on ha de col·locar els robots de manera que quan acabi el temps i els enemics ataquin, aquests no siguin destruïts.

Per aconseguir això, el jugador ha de conèixer tant els robots com els enemics i és per aquest motiu que cada vegada que un enemic o robot és desbloquejat i per tant apte per aparèixer durant un nivell proper, és presentat mitjançant una breu explicació i un vídeo d'exemple en el mateix instant, durant la partida, entre un nivell i el següent.

Si el jugador ha acabat de col·locar els robots abans que s'esgoti el temps, no cal que s'esperi fins que aquest arribi a zero, sinó que pot prémer un botó de finalització i immediatament començaran els atacs dels enemics. A més com abans es premi aquest botó més punts s'obtindran en aquest nivell.

Aquest és el **botó de finalització** que permet finalitzar el nivell sense esperar a que el temps arribi a zero, en cas que el jugador hagi col·locat tots els robots.

Figura 8. Botó de finalització

Així doncs queda clar que l'objectiu d'un nivell, sigui de la modalitat que sigui, consisteix a salvar tots els robots continguts en aquest.

A banda d'aquest objectiu global existeixen una sèrie d'objectius parcials continguts a cada nivell. S'expliquen a continuació:

2.7.2. Objectiu monedes de joc

Cada vegada que se supera un nivell s'obtenen monedes de joc tal i com succeeix a la majoria de jocs trenca-closques. Aquests diners de joc serviran per a poder comprar armes (abans esmentades) a la botiga de l'usuari i utilitzar-les durant els nivells de joc per a fer-los més fàcils, si cal.

Els diners estan lligats a les estrelles que l'usuari aconsegueix a cada nivell (les estrelles s'expliquen a continuació) i aquestes estan lligades al temps que l'usuari ha necessitat per superar el nivell. Com més petit sigui el temps, més diners de joc s'obtindran per nivell.

En cas de perdre un nivell s'aconsegueix una suma molt petita de monedes de joc, calculada segons el nombre de robots que han mort. A més robots morts menys diners obtinguts. Mai perdent un nivell s'obtindran més diners que guanyant un nivell.

2.7.3. Objectiu estrelles

Quan se supera un nivell s'aconsegueixen entre 0 i 3 estrelles. Aquestes s'aconsegueixen tenint en compte el temps que ha trigat el jugador a superar el nivell, com menys trigui més estrelles guanyarà. Les estrelles només tenen influència directa sobre els diners que es guanyarà per haver superat aquest nivell, però no afecten a res més.

Les estrelles poden ser una bona motivació per, una vegada acabat el joc, tornar a jugar tots els nivells intentant obtenir les 3 estrelles a tots ells.

Més endavant s'explicarà com es calculen les estrelles i la puntuació mitjançant un algorisme matemàtic.

2.7.4. Botiga d'armes

Un altre objectiu clar per a qualsevol jugador és provar totes les armes (avantatges) del joc. Les armes serveixen per ajudar al jugador a superar un nivell en el que, per exemple, es queda encallat molt de temps sense poder-lo superar. Un exemple d'arma en un altre joc de matrius similar podria ser la 'bomba multicolor' de Candy Crush.

Quan el jugador supera uns quants nivells ràpidament obté una bona quantitat de monedes de joc que podrà fer servir a la botiga per a comprar armes. Cada arma té un preu associat a la seva utilitat, així doncs les armes més poderoses tenen un preu més car a la botiga.

La botiga també ofereix la compra de vides per si un jugador vol fer el bescanvi directament amb monedes de joc sense, per exemple, haver de passar per la versió Arcade per recuperar vides o jugar a nivells personalitzats.

2.7.5. Diagrama de comportament esperat

Si bé és difícil predir quin serà el comportament dels jugadors, el següent diagrama mostra el comportament esperat de la majoria dels usuaris. Aquest comportament, com

no podia ser d'altra manera, depèn de 2 factors: La modalitat 'Original' del joc, que n'és la principal, i les vides que té el jugador.

Inicialment i fins que l'usuari coneix el joc a fons, el que s'espera d'ell és que jugui a la modalitat 'Original' basada en mons i nivells, i que quan esgoti les vides miri de recuperar-les provant altres modalitats. Com és normal, no tothom té els mateixos gustos, cosa que fa possible que part dels usuaris trenquin el següent esquema, que mostra el comportament inicial esperat:

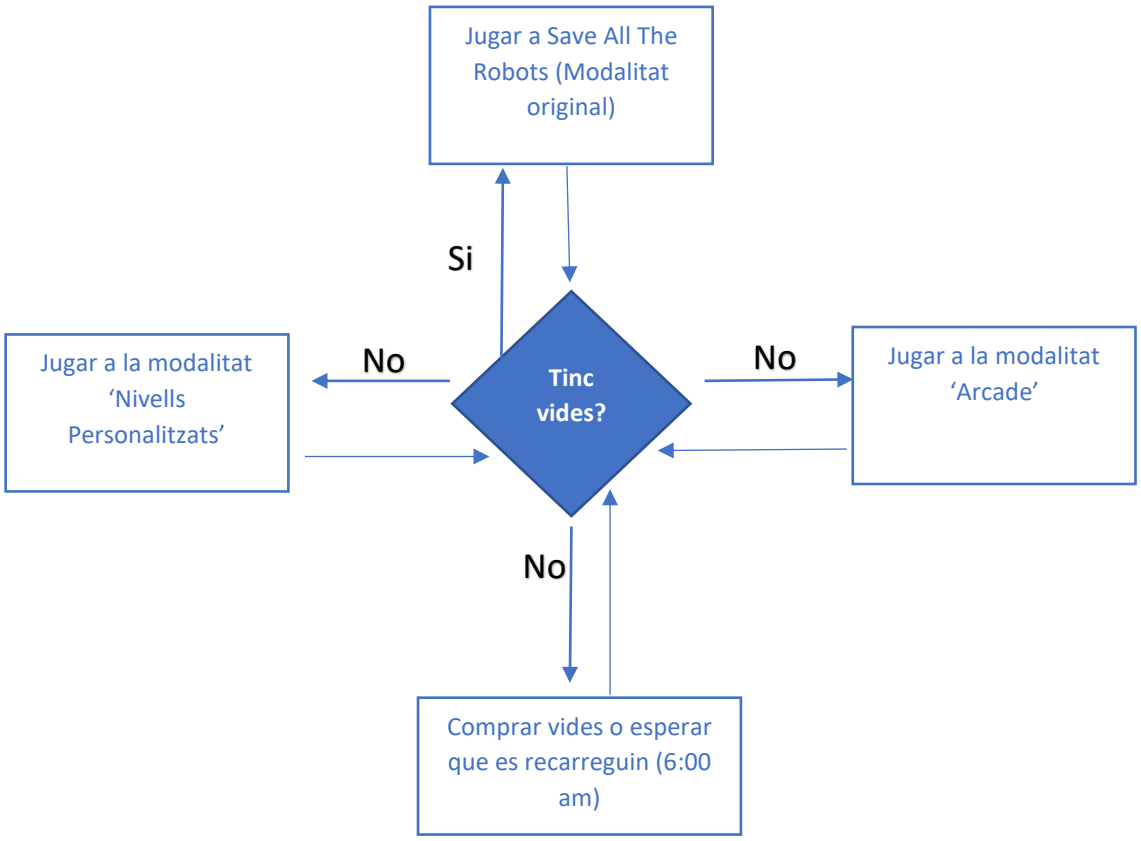


Figura 9. Diagrama de flux que mostra el comportament esperat d'un jugador

Un aspecte a tenir en compte és que la única modalitat de joc que té un final és la modalitat 'Original' i arribar a aquest final és la principal raó de ser del joc.

Per què el diagrama anterior pugui funcionar, un cop el jugador esgota les vides se'l redirigeix a una pantalla que l'insta a recuperar-les jugant altres modalitats de joc. Es mostra a continuació:

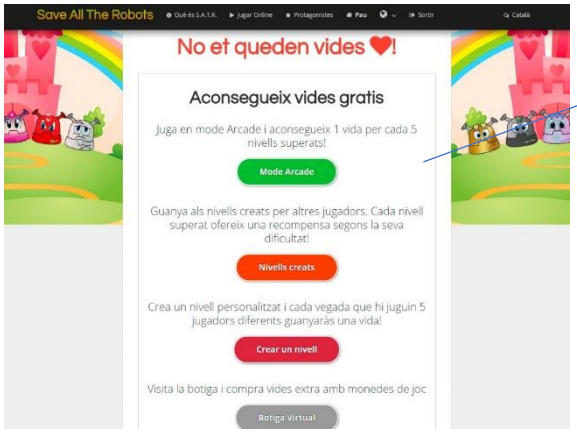


Figura 10. Pantalla per quan s'acaben les vides del jugador

2.7.6. Resum final dels objectius

Arribats a aquest punt el lector és coneixedor dels objectius globals del joc, aquells que pertanyen a un nivell sigui quina sigui la modalitat de joc, com poden ser les monedes, les estrelles i salvar robots. D'altra banda, també es coneixen els diferents objectius que el joc proposa a cadascuna de les seves modalitats.

Ara que el lector té una visió general del funcionament del joc i que de ben segur haurà jugat a totes les modalitats per corroborar el que s'escriu en aquest informe, es pot passar doncs, al detall de la implementació i a tot el desenvolupament, així com a conèixer des de la pantalla d'inici del joc fins a l'últim racó de la pàgina d'usuari i els perfils públics, però abans es fa un incís en la planificació i els terminis acomplerts durant el desenvolupament de 'Save All The Robots'.

CAPÍTOL 3.

PLANIFICACIÓ

Aquest treball de fi de grau es comença durant el mes d'abril de 2017 quan l'autor aconsegueix l'acord oficial amb el doctor Santi Seguí, que serà el director definitiu del treball.

La idea de l'autor, després de mantenir la conversa inicial amb el director del treball, era ràpidament determinar la viabilitat del projecte en els llenguatges de programació triats.

Així doncs després d'una setmana de proves s'arriba a la conclusió que el joc es viable. "*...Primer es construeix el taulell i les corresponents cel·les, després es prova de col·locar-hi els robots i es mira que els dissenys tinguin una qualitat equivalent a la de qualsevol joc de la competència, qualsevol joc d'avui dia que funcioni sobre matrius...*"

D'aquesta manera s'arriba al mes de maig amb la idea de joc encarada i la seguretat que molt aviat es faria realitat degut a la seva alta viabilitat.

3.1. Setmanaris de treball

Setmana 1

Determinació de la viabilitat del projecte. Es triga una setmana a verificar que l'aplicació pot ser portada a terme en els llenguatges triats i a més es demostra que té sentit tirar-ho endavant. Les principals proves han estat:

- Construcció de la matriu lògica i gràfica
- Moviment i col·locació dels robots mitjançant 'click' i 'drag and drop'
- Creació de 2 nivells i fent les crides necessàries al servidor mitjançant AJAX per obtenir els nivells en temps real i poder utilitzar-los. (Això s'explicarà en detall més endavant).
- Els 3 passos anteriors repetits però des d'un dispositiu mòbil.

Setmanes 2 i 3

Dissenys dels nous robots, dels enemics i establiment de les 8 zones de joc. L'autor triga dues setmanes a fer els nous dissenys de robots, d'enemics, d'efectes dels enemics i de les zones de joc, a més del disseny base de tota la interfície.

Setmanes 4 i 5

Maqueta funcional del joc. En aquestes dues setmanes d'sprint l'autor va aconseguir una maqueta funcional del joc. La maqueta va consistir en un joc acabat i jugable amb totes les zones , nivells i transicions acabades fins arribar a la pantalla de final del joc.

L'objectiu de la maqueta era superar els 72 nivells amb el simple fet de col·locar els robots sobre les cel·les, encara sense enemics, amenaces ni atacs de manera que cada nivell tenia una dificultat idèntica al nivell següent i al nivell anterior.

Aquesta maqueta va servir com a base de desenvolupament. Un joc que funcionava i sobre el qual s'anirien afegint funcionalitats que mica en mica incrementarien de forma notable la complexitat i la dificultat fins arribar al resultat final.

Setmanes 6 i 7

Xarxa d'usuari. S'implementen els inicis de sessió via email, l'userhome (pàgina d'usuari) on aterrarà un jugador després d'iniciar sessió i que permetrà interactuar amb totes les dades de joc de l'usuari i amb les seves dades personals.

Després d'això es va passar a fer les aplicacions de les 4 xarxes socials bàsiques: facebook, twitter, linkedin i google. Aquestes aplicacions permeten connectar Save All The Robots amb les APIs de les xarxes socials i poder programar els inicis de sessió amb un clic a través d'aquestes.

Una vegada es van tenir tots els inicis de sessió funcionals es va passar a programar l'userhome de manera que permeti a l'usuari accions bàsiques com canviar-se la foto de perfil, canviar la seva clau d'accés al sistema, donar-se de baixa del sistema, etc.

Setmanes 8, 9, 10 i 11

Part algorísmica. Desenvolupament de la part algorísmica del joc, és a dir, les funcionalitats dels diferents robots, els atacs dels enemics, els efectes dels atacs dels enemics, etc.

Després d'aquest llarg *sprint* d'un mes de treball es va obtenir el joc completament acabat i funcional, és a dir, la versió 'Original' del joc que en aquell moment era el joc

algorísmicament acabat ja que l'autor encara no havia decidit dur a terme la versió 'Arcade' ni la versió 'Nivells Personalitzats'.

Setmanes 11, 12 i 13

SQL, Transformacions de la matriu, models i editor de nivells. Una vegada el joc 'Original' estava algorísmicament acabat calia pensar quina seria la millor manera per guardar els nivells del joc: fitxers, bases de dades, serialitzacions, etc.

Com no podia ser d'una altra manera l'autor va decidir utilitzar la persistència de dades mitjançant bases de dades SQL on cada nivell seria representat per una fila a la taula de nivells, cada enemic una fila a la taula d'enemics i cada robot una fila a la taula de robots, de manera que mitjançant reunions naturals (joins) es podria construir un nivell en mil·lèsimes de segon cada vegada que la base de dades rebés una sol·licitud per *http* procedent del domini de Save All The Robots. (Tot el funcionament s'explica a fons en capítols posteriors).

En aquest punt l'autor va detectar un problema molt important: un jugador podia fer una captura de pantalla, i si perdia un nivell, aprendre-se'l i aleshores tornar a jugar i passar-se'l sense problemes. Per tant s'havia de dotar els nivells de certa aleatorietat i a més construir diversos models per nivell. Per donar aleatorietat es var decidir fer transformacions a la matriu de manera que un nivell pogués tenir 16 representacions diferents aconseguint que fos simètricament idèntic però visualment diferent a cada transformació, i que per tant, dificultés molt aprendre's un nivell. (Les transformacions de la matriu s'expliquen a fons al capítol de desenvolupament)

A més es va decidir crear 2 models per nivell de manera que un mateix nivell podria tenir 32 representacions diferents. Impossible de memoritzar ja que cada model i cada transformació es tria aleatòriament quan es carrega un nivell, fent que la probabilitat de repetir nivell sigui 1/32.

Una vegada solucionat l'anterior, sorgia un nou problema. Crear els nivells manualment a la base de dades a partir de dibuixos fets a mà era una tasca del tot inviable. Calia idear un sistema mitjançant el qual l'administrador del joc pogués fer i desfer nivells més ràpidament. D'aquesta manera neix l'editor de nivells.

L'editor de nivells disposa d'un inventari de robots i un inventari d'enemics que mitjançant 'drag and drop' l'autor pot arrossegar damunt d'un taulell de joc, veure com queda realment un nivell, després triar un temps pel nivell i finalment fer un insert a la base de dades a través d'un simple click des de la pàgina web. (L'editor es detalla més endavant d'una manera més complerta).

Setmanes 14 i 15

Nivells personalitzats. Una vegada la modalitat 'Original' estava completament funcional (encara sense puntuació, vides ni estrelles) l'autor va pensar que podria utilitzar l'editor de nivells per crear una modalitat nova on cada jugador pogués crear els seus propis nivells i convidar altres jugadors a superar-los.

Així doncs, aprofitant l'editor de nivells i els algorismes de joc de la modalitat 'Original' neix la versió 'nivells personalitzats' que triga unes dues setmanes a estar completament funcional.

Setmana 16

Versió Arcade del joc. S'aprofiten els algorismes de la versió 'Original' del joc per a construir una versió 'Arcade'. El funcionament i els objectius d'aquesta versió s'han explicat anteriorment, només cal afegir que per fer que en aquesta versió els nivells siguin infinits, es parteix de 30 nivells amb 4 models per nivell, és a dir, 4x16 = 64 versions de nivell diferents per cada nivell i amb probabilitat 1/64 de repetir un nivell.

Per fer-ho infinit, quan s'arriba al nivell 30 (tasca dificilíssima per qualsevol jugador), aleshores es tria aleatòriament un nivell entre el 5 i el 30, és a dir que a partir del 30 es tornen a repetir aleatòriament. Matemàticament això implica 25x4x16 = 160 models diferents per nivell a partir del nivell 30, és a dir, una probabilitat de 1/160 de repetir nivell.

Tot això dota el programa d'una aleatorietat més que notable on l'usuari té la total impressió que els nivells són sempre aleatoris. Realment, però, són pseudo-aleatoris.

Setmanes 17 i 18

Vides, puntuacions, estrelles i altres petits detalls. Es desenvolupa el sistema de vides (explicat anteriorment), el sistema de monedes de joc i el sistema de puntuacions, el qual es farà referència més a fons a l'apartat de desenvolupament.

En aquest punt el que cal dir és que s'ha triat el mateix criteri de puntuacions i estrelles per totes les modalitats de joc, tret dels nivells personalitats que ofereixen una recompensa diferents segons la seva dificultat, però d'això ja se n'ha parlat i explicat anteriorment.

Setmanes 19 i 20

Adaptació Mòbil. Es va desenvolupar paral·lelament la versió PC i la versió Mòbil. Tot i així es van necessitar dues setmanes extres per fer que la versió mòbil quedés perfecta i absolutament jugable.

Setmanes 21 i 22

Testing. Finalment, durant les dues últimes setmanes de desenvolupament es va demanar als amics més propers que es registressin a la pàgina web i que juguessin al joc per tal de testar que funcionés correctament.

Setmanes 23, 24 i 25

Correcció d'errors i altres peticions del professor. Les últimes setmanes de TFG el que s'ha fet ha estat corregir errors. D'altra banda s'han satisfet les peticions del tutor del TFG que són les següents:

- **Elaboració d'una gràfica que mostra la dificultat de cada nivell del joc.** Aquest gràfic s'ha realitzat sobre la versió 'Original', la versió 'Arcade' i en cada nivell personalitzat creat per algun jugador, de manera que cada nivell que existeix a Save All The Robots sigui quina sigui la seva modalitat, disposa d'una gràfica que mostra les seves estadístiques i la seva dificultat.
- **Elaboració d'una gràfica que mostra el percentatge de jugadors que han vist un determinat nivell.** Aquesta gràfica s'ha dut a terme sobre les versions 'Original' i 'Arcade' del joc. Permet saber en quin nivell abandonen el joc els usuaris, de

manera que si es detecta una caiguda molt forta, es podrà actuar ràpidament sobre el nivell que propicia que els jugadors abandonin i arreglar-ne els problemes pertinents.

A la plana següent es mostra el diagrama de Gantt de la planificació

3.2. Diagrama de Gantt

Activitat	Duració	Setmanes																								
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Determinació de la viabilitat del projecte	1 Setmana																									
Disseny dels nous robots i enemics, i construcció dels 8 mons del joc	2 Setmanes																									
Maqueta funcional del joc	2 Setmanes																									
Xarxa d'usuari	2 Setmanes																									
Part algorísmica	4 Setmanes																									
Transformacions de la matriu i editor de nivells	3 Setmanes																									
Nivells personalitzats	2 Setmanes																									
Versió Arcade del joc	1 Setmanes																									
Vides, puntuacions i estrelles	2 Setmanes																									
Adaptació de la versió 'mobile'	2 Setmanes																									
Testing	2 Setmanes																									
Correcció d'errors i analítiques del joc	3 Setmanes																									

3.3. Metodologia de planificació i desenvolupament

Com es pot observar s'han utilitzat metodologies àgils per desenvolupar el projecte en forma de diferents sprints, seguint els següents principis:

- **Bona adaptació de nous requisits**: Permet fer canvis sobre la marxa ja que les funcionalitats estan programades per capes.
- **Objectius periòdics**: La divisió del treball en fases productives és la base de la metodologia. Després d'un sprint s'obtenen uns resultats tangibles.
- **És fàcil mesurar el progrés**: L'evolució dels processos no és un element subjectiu. Es pot mesurar el progrés de forma molt concreta.
- **Simplicitat**: Les tasques de cada sprint són tant senzilles com sigui possible. Si alguna no ha pogut ser executada, s'ha dividit en iteracions fins a reduir la seva complexitat.
- **Adaptació a circumstàncies canviants**: Un projecte sol acabar amb una forma diferent a la que es preveu d'inici. Un bon exemple d'adaptació és la construcció i adaptació de les noves modalitats de joc construïdes mentre es feia camí.

CAPÍTOL 4.

DESENVOLUPAMENT

Una vegada s'ha conegut el funcionament del joc gràcies a l'apartat 'Objectius' i les tasques més importants que s'han realitzat gràcies a l'apartat 'Planificació' es pot passar al plat fort d'aquest document, que no és pas altre que el desenvolupament del joc.

Per facilitar la comprensió al lector, aquesta secció es desgrana en 3 grans apartats:

1. **Entorn web del joc i xarxa social**

   Explica detalladament l'entorn de desenvolupament del joc parlant de diversos aspectes i tractant les xarxes com a tema principal. Es parla a fons de la xarxa del sistema i de la xarxa social d'usuari.

2. **Motor algorísmic del joc**

   Tracta tots els aspectes algorísmics de Save All The Robots i la lògica del motor del joc.

3. **Analítiques del joc**

   Entorn del joc per a poder analitzar-ne el funcionament i poder prendre decisions per a futures actualitzacions.

Cadascun del apartats, a més, es desgrana en petits subapartats que expliquen els detalls i les tècniques més importants utilitzades durant la fase de desenvolupament de Save All The Robots.

4.1. ENTORN WEB DEL JOC I XARXA D'USUARI

4.1.1. Aspectes bàsics de desenvolupament

Save All The Robots és un joc online programat d'una manera molt peculiar, doncs no deixa de ser una aplicació web. Així doncs tota la lògica, efectes, transicions, etc.., estan programades com a qualsevol aplicació web, en altres paraules, no es tracta d'una aplicació multi plataforma, com per exemple, Candy Crush Saga. És una aplicació

executable en qualsevol navegador des de qualsevol dispositiu i sense importar el sistema operatiu utilitzat.

Aquest sistema té l'avantatge que no es necessiten descarregues addicionals i instal·lacions pesades per a poder jugar, sinó que l'únic que necessita una persona és connexió a internet, cosa que avui dia es pot aconseguir des de pràcticament qualsevol lloc.

D'altra banda també té inconvenients: un tall a la xarxa o deixar de tenir connexió a internet suposa no poder continuar jugant.

Aquí cada persona pot tenir una opinió diferent sobre què és millor i què és pitjor. L'autor del TFG, però, es decanta per l'aplicació web per tirar endavant el treball.

4.1.2. Funcionament bàsic del sistema

Save All The Robots és una aplicació distribuïda de tipus client-servidor on el codi es divideix en dos blocs distingits:

- **Front-End:** és la part del codi que s'executa a la computadora del client, és a dir, al navegador. Aquest codi està format bàsicament per la barreja d'Html, CSS i JavaScript.

  Html és el llenguatge de marques que sap interpretar un navegador. Cap aplicació web podria existir sense html. Per la seva banda CSS és el llenguatge que 'decora' l'aplicació i serveix per organitzar-ne la presència visual.

  JavaScript és el llenguatge clau del FrontEnd, doncs s'encarrega de la part lògica de l'aplicació a la màquina del client i s'executa al navegador.
- **Back-End:** És la part de codi que s'executa a la computadora del servidor, és a dir, al núvol. Aquest codi està format exclusivament per llenguatge PHP (en el nostre cas), i com a servidor web s'utilitza *Apache*.

  El back-end també gestiona les peticions a base de dades. El servidor de base de dades rep les consultes procedents de la màquina servidora i retorna les respostes que el servidor enviarà al client.

Tot seguit podem veure-ho més clar en un diagrama:

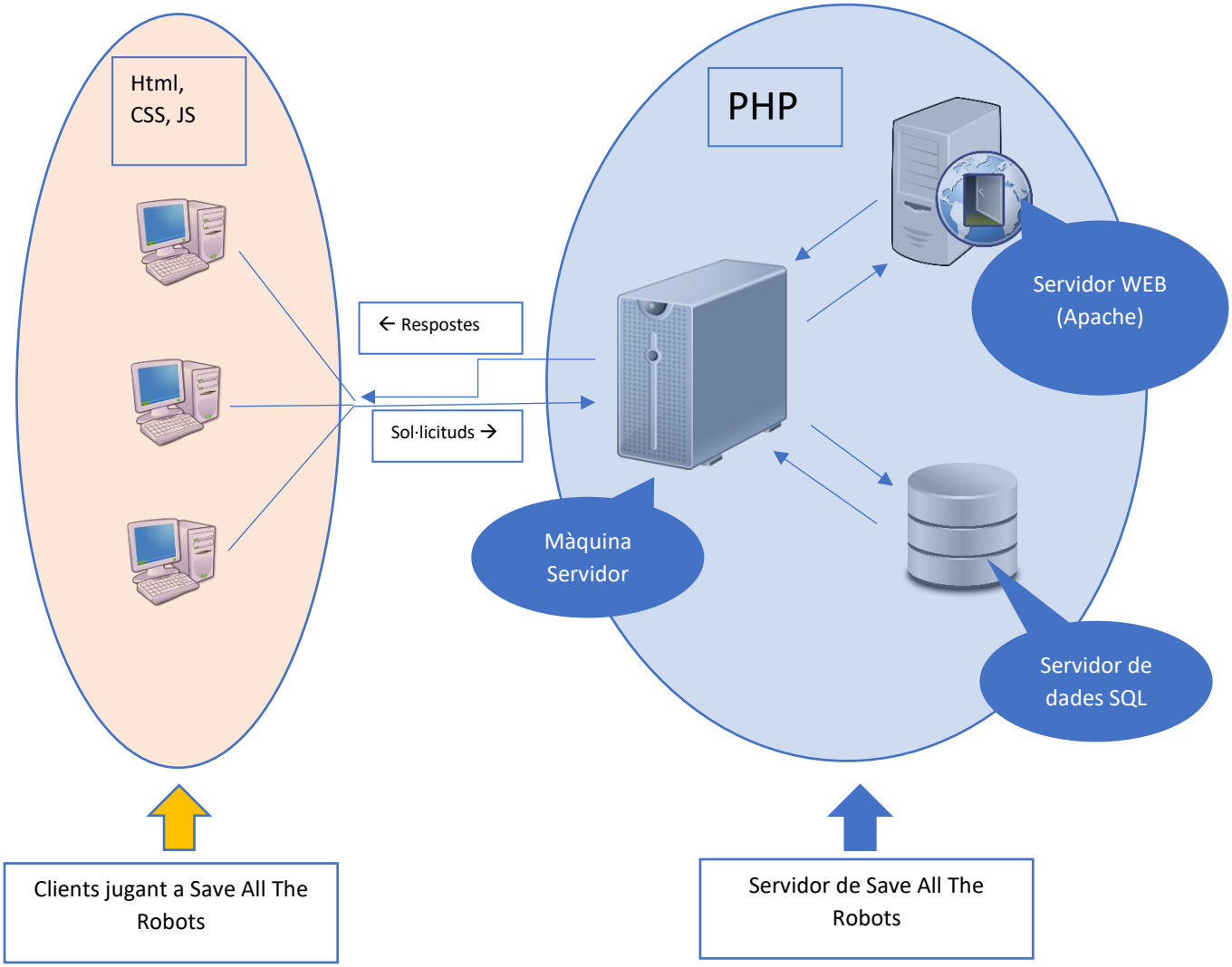


Figura 11. Save All The Robots com a aplicació client-servidor

La figura 11 mostra l'esquema de peticions i respostes de Save All The Robots, ara bé, més endavant quan s'entri en detall s'explicarà que aquestes poden ser síncrones i bloquejants, que necessiten un refresc de la pàgina, o bé asíncrones i per tant, no bloquejants, que no necessiten refrescar tota la pàgina. Save All The Robots les utilitza totes dues segons la necessitat.

La lògica de Save All The Robots està repartida entre el front-end i el back-end segons la necessitat de cada moment del joc, estant totes dues parts en comunicació constant.

4.1.3. Back End

Aquesta secció explica com està organitzat el Back End de Save All The Robots, és a dir la part de PHP i SQL amb tota la lògica que això implica.

PHP és un llenguatge de programació orientat a objectes amb una sintaxi molt semblant a Java o C que s'executa al servidor.

Save All The Robots està programat en PHP natiu sense l'ús de cap FrameWork que ajudi a l'organització del codi, concretament s'utilitza la versió 5.2. de PHP. L'autor ha preferit crear des de zero el seu propi disseny d'aplicació construint els fonaments a un nivell d'abstracció més baix del que s'acostuma a actuar quan es treballa amb Frameworks, ja que aquests ofereixen molta feina feta ja pre-programada, però obliguen a utilitzar un determinat disseny. L'autor ha preferit implantar el seu propi disseny de software des de zero.

L'autor, però, si que utilitza llibreries de funcions al back-end per gestionar certs aspectes de l'aplicació. Són els següents:

- **MDB2 (PEAR)** → Pear és un repositori d'extensions i aplicacions de PHP. MDB2 és un combinació de capes d'abstracció per manipular les bases de dades amb més comoditat mitjançant l'ús d'una llibreria de funcions.
- **PHPMailer** → PHPMailer és una llibreria de funcions per enviar missatges de correu electrònic basats en el component actiu ASPMail. Permet de forma senzilla tasques complexes com per exemple: enviar missatges de correu amb fitxers adjunts.
- **MobileDetect** → Llibreria de funcions que detecta el tipus de dispositiu que està fent la sol·licitud al servidor via http, en altres paraules, identifica el tipus de dispositiu a través del qual s'ha accedit a Save All The Robots.

A part d'aquests 3 llibreries no se n'utilitza cap altra pel desenvolupament del back-end de Save All The Robots.

4.1.3.1. Back End: Disseny i funcionament

Quan un usuari accedeix al sistema genera una petició a través del protocol http al servidor, el primer que fa el servidor és executar el fitxer que l'usuari està sol·licitant a través de l'aplicació client. Cadascun d'aquests fitxers està representat per una classe de PHP i cadascuna d'aquestes classes és una extensió d'una classe abstracta mare anomenada Settings.

Com era de suposar, el primer fragment de codi que executarà sempre el servidor quan rebi una petició serà el constructor de la classe Settings, que executa els aspectes bàsics comuns a totes les peticions, són els següents:

- Connectar-se amb la base de dades i testar la connexió
- Establir el llenguatge preferit de l'usuari (Anglès, Català o Castellà)
- Establir la metadata segons quina sigui la pàgina sol·licitada
- Detectar el tipus de dispositiu que fa la sol·licitud i, si escau, redireccionar
- Instanciar objectes imprescindibles, com els controladors i els manipuladors
- Comprovar si hi ha sessió d'usuari
- Altres tasques...

Una primera aproximació per entendre el disseny s'explica a continuació mitjançant la figura 12 en un petit diagrama simplificat de classes:

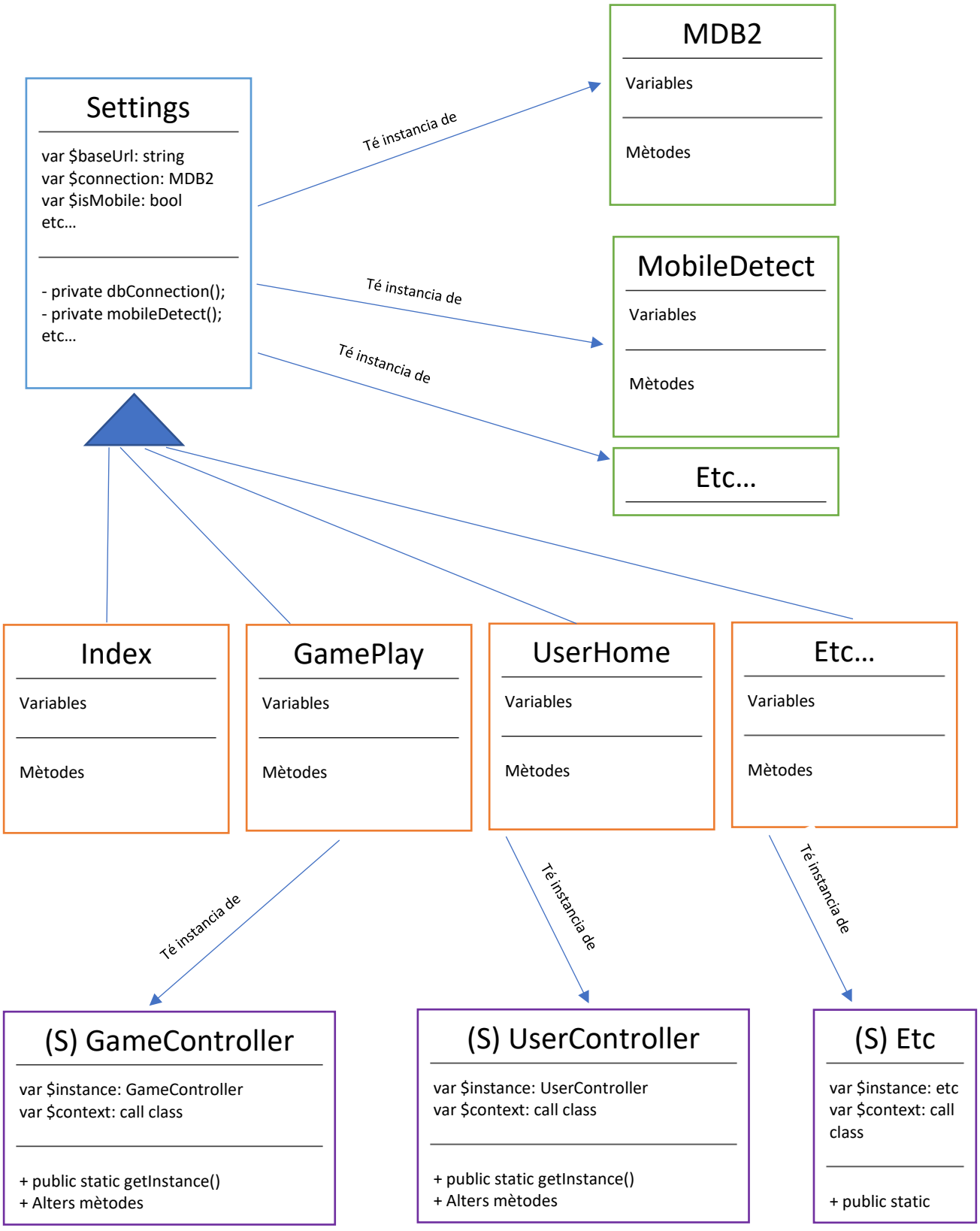


Figura 12. Diagrama de classes simplificat

El diagrama anterior és una representació de classes simplificada (la realitat és més complexa) però que serveix per entendre el funcionament bàsic del back-end a qualsevol execució. Les classes amb el requadre groc representen les classes que són executades directament pel client. Agafem com a exemple la classe GamePlay:

- L'usuari fa la petició a la URL, s'executa el constructor d'aquesta classe que el primer que fa és cridar al constructor de la classe mare Settings. Per tant el primer que s'està executant és la classe Settings tal i com s'havia esmentat en anteriorment.
- La classe Settings crea les instàncies necessàries per l'execució del programa (el diagrama en mostra dues, però són moltes més), i configura aspectes bàsics com el llenguatge, entre d'altres.
- Un cop executat Settings, el constructor de la classe GamePlay continua la seva execució creant les instàncies necessàries per a l'execució d'aquesta classe en concret. Tal i com es mostra, crea doncs una instància del controlador del joc. (En crea moltes altres però es pot aplicar el mateix exemple, per això se'n mostra només una).
- Finalment, quan tot això s'ha executat en el back-end (màquina del servidor al núvol), aquesta classe desplega l'html del fitxer i el mostra per pantalla.

Aquest exemple tant senzill és aplicable per qualsevol petició de pàgina. Sempre és el mateix i es repeteix constantment a mesura que l'usuari navega pel website. Ara bé, aquest és l'esquema síncron. De l'asíncron se'n parla més endavant.

Abans de seguir avançant cal entendre un aspecte fonamental: aquí no estan relacionades totes les classes com en una aplicació d'escriptori normal, per exemple construïda en Java Swing o Android. Aquí la classe *Index* no sap res sobre l'existència de la Classe *GamePlay* ni li importa, de fet no es comunicaran mai. Les classes executables es poden entendre com aplicacions diferents dins un conjunt d'aplicacions i cadascuna d'aquestes aplicacions té un diagrama de classes i un funcionament diferent. Cadascuna d'elles s'anomena pàgina web i tot el conjunt rep el nom website ('sitio' web). Aquesta és una de les raons per la qual s'ha triat un diagrama de classes senzill però entenedor

enlloc d'un diagrama de classes gran i complert que ocuparia més de 10 pàgines, quedaria seccionat per aquestes, i finalment el lector no entendria res.

4.1.3.2. Back-end – Base de dades

Save All The Robots utilitza una base de dades de tipus relacional i fa servir el model entitat-relació on les entitats representen objectes del mon real que tenen atributs característics i que estan relacionats entre ells.

A la pàgina següent es mostra un esquema reduït de la base de dades de Save All The Robots. S'han inclòs aquells aspectes que s'ha considerat que són bàsics per a la presentació del projecte, però com ha succeït amb el diagrama de classes, la realitat és molt més complexa i ocuparia massa espai innecessari el fet de realitzar el diagrama complert de la base de dades.

> El diagrama simplificat de la base de dades es pot veure a la pàgina següent.

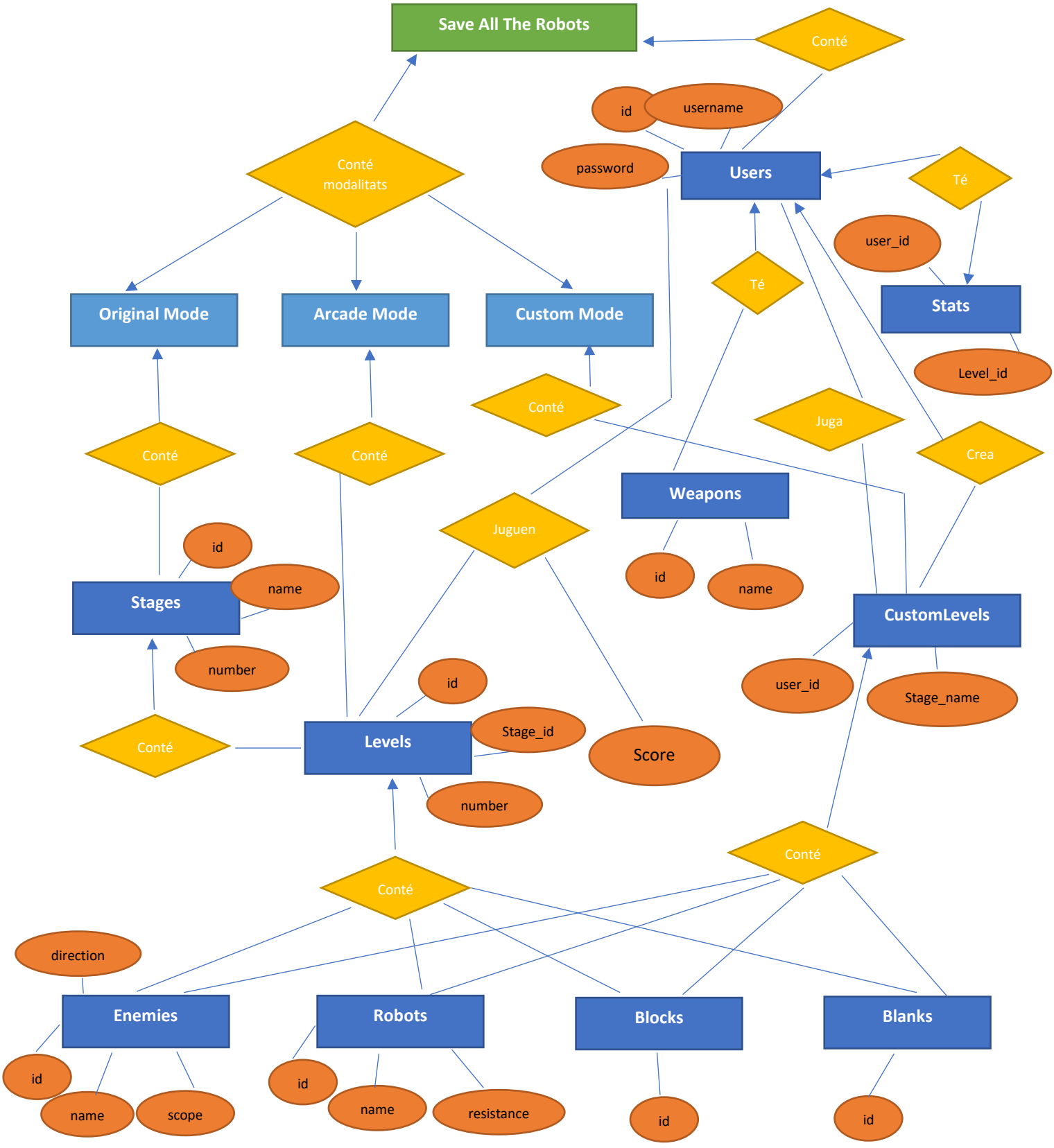


Figura 13. Diagrama de bases de dades simplificat

4.1.4. Front End

El front-end, que correspon a la banda del client de Save All The Robots és la part de l'aplicació que s'executa a la màquina del client.

El cos del front-end de Save All The Robots està implementat en Html, CSS i JavaScript fent ús de la llibreria jQuery, però sense la utilització de frameworks. Tot està, doncs, implementat en JavaScript nadiu i jQuery.

Aquesta part de l'aplicació, que també utilitza el paradigma de programació orientada a objectes, utilitza objectes propis del llenguatge JavaScript però no pas nous tipus de dades abstractes com era el cas del back-end.

A part dels tipus de dades natives com els enters, flotants, i strings, el front-end està basat en objectes de tipus 'Array' i 'Diccionaris' del JavaScript fent que les accions estiguin basades en funcions enlloc de mètodes.

La raó per la qual s'ha triat elaborar el front-end d'aquesta manera és la simplicitat. Aquí els objectes es buiden i s'omplen de manera reiterada, asíncrona i transparent a l'usuari.

En aquest punt val la pena explicar la tècnica utilitzada per entendre millor la utilització dels objectes, saber com, quan i perquè es buiden i s'omplen i quin tipus de dades guarden.

4.1.4.1. Front End: Objectes asíncrons via AJAX/JSON

Definició de conceptes:

AJAX →Async JavaScript And Xml: és una tècnica de desenvolupament per aplicacions interactives que s'executen a la banda del client, és a dir, al navegador dels usuaris, mentre es manté una comunicació asíncrona amb el servidor en segon pla. D'aquesta manera és possible realitzar canvis a les pàgines sense necessitat de recarregar-les com a les sol·licituds i respostes normals abans esmentades. Millora la usabilitat i la velocitat de les aplicacions.

JSON → Java Script Object Notation: és un format lleuger d'intercanvi de dades fàcil de llegir i escriure per humans i molt fàcil de ser interpretat i generat per les màquines. Bàsicament està format per un array de diccionaris representat com un string alhora de ser enviat per la xarxa.

Utilització general JSON/AJAX:

1. El front-end necessita dades del servidor en temps real, per exemple, els enemics d'un determinat nivell amb els seus atributs.
2. Aleshores el front-end envia una petició asíncrona via AJAX invocant un mètode remot que forma part del back-end
3. El back-end rep la sol·licitud, crea un objecte amb els enemics, el codifica com a JSON, cosa que converteix l'objecte en un string, i l'envia per la xarxa via AJAX (http és un protocol basat en text)
4. El front-end rep l'string, descodifica el format JSON i obté el mateix objecte que va crear el back-end però ara en llenguatge JavaScript apunt per ser utilitzat.

Els 4 passos anteriors són completament transparents a l'usuari, és a dir que aquest no s'adona de res. *Mentre un usuari juga a SaveAllTheRobots, s'estan enviant sol·licituds i rebent respostes durant gairebé tot el temps. És fàcil adonar-se'n ja que quan un usuari està jugant, no es recarrega mai la pàgina quan es canvia de nivell, quan es canvia de món, etc. Tot es fa "per sota" construint i destruint fragments html i objectes de JavaScript en temps real i de forma paral·lela.*

4.1.4.2. Front End: Creació d'un nivell qualsevol del joc

Imaginem que un usuari que està jugant, per exemple, a la modalitat 'Original' del joc fa click sobre el nivell 4 del primer món per jugar-lo:

Quan el jugador fa clic sobre el nivell es desencadena un esdeveniment que col·loca a memòria el nou nivell clicat i destrueix el nivell anterior. A més regenera el nou cos visible del nivell (html) destruint l'anterior. Tot això succeeix mentre l'usuari espera (milisegons) a que el nivell es carregui.

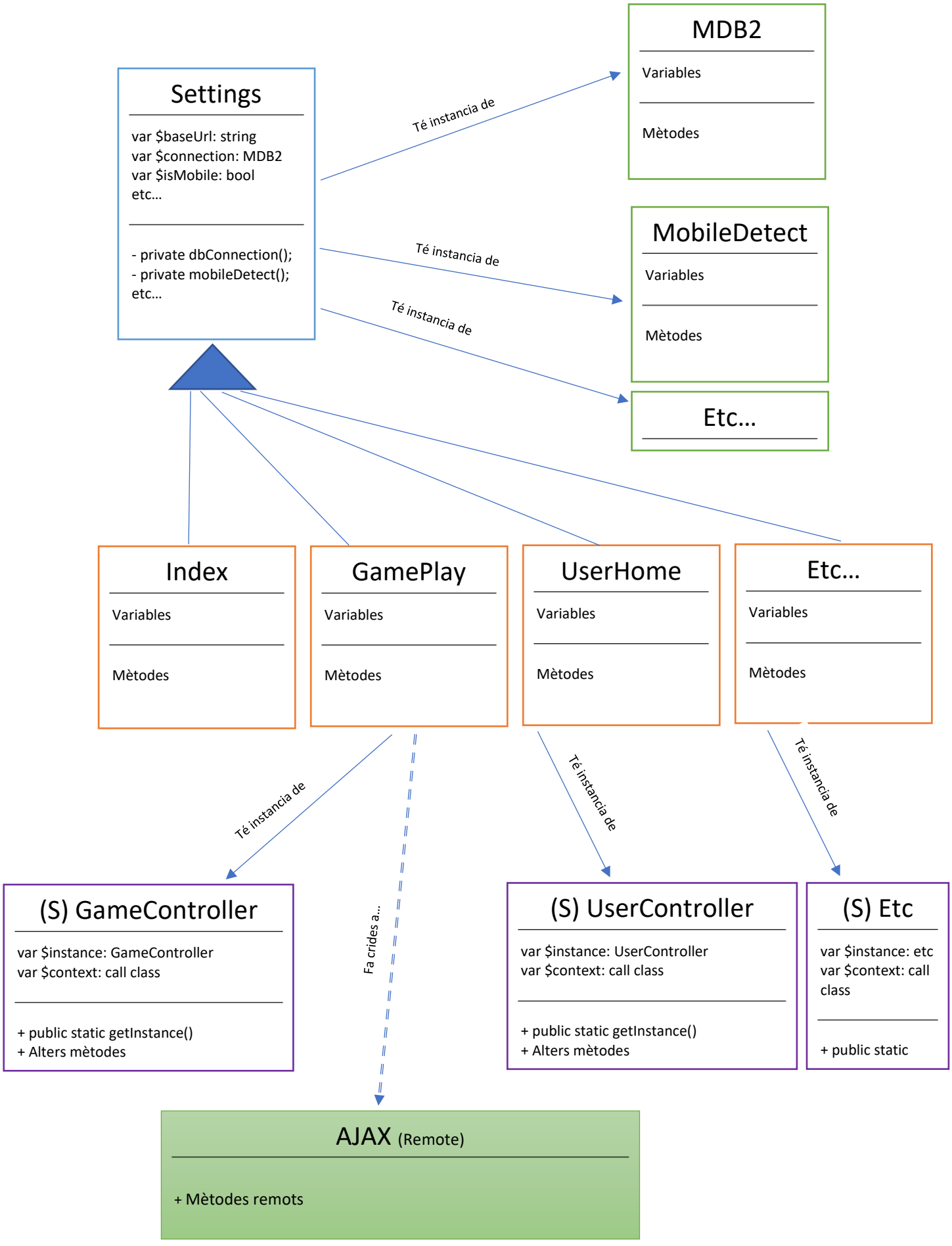
Durant el petit període de temps que l'usuari contemplava la imatge del robot donant voltes sobre el seu eix llegint la paraula 'Carregant...' (imatge segona) s'han transferit dades, creat i destruït objectes, etc... S'explica a continuació:

1. L'usuari fa clic sobre el nivell i activa l'esdeveniment de canvi i creació de nou nivell.
2. El Front-end sol·licita al servidor via Ajax un model aleatòri del nivell (recordar que cada nivell té diversos models diferents) activant un mètode remot.
3. El back-end rep la sol·licitud, selecciona els models d'aquell nivell de la base de dades i retorna un ID de nivell aleatori com a nombre enter.
4. El front-end rep l'ID del nivell i l'utilitza tornar a sol·licitar al servidor via Ajax totes les dades relacionades amb aquell ID de nivell (robots, enemics, blocs i blancs):
    - Se sol·liciten i es retornen els robots de la mateixa manera Ajax/Json
    - Se sol·liciten i es retornen els enemics de la mateixa manera Ajax/Json
    - Se sol·liciten i es retornen els blocs de la mateixa manera Ajax/Json

    - Se sol·liciten i es retornen els blancs de la mateixa manera Ajax/Json
    - Se sol·licita i es retorna el temps del nivell d'igual manera
5. Ara el front-end té tots els objectes de JavaScript que ha sol·licitat i els utilitza per regenerar l'html del nivell amb els nous enemics, el nou número de robots, el nou inventari amb els robots corresponents per aquell nivell, el nou temps de nivell, etc.
6. Una vegada regenerat l'html i col·locats els enemics es calculen de manera algorísmica les cel·les amenaçades pels enemics.
7. Una vegada s'ha fet tot això el nivell està apunt per a ser jugat.

Nota: Aquet recorregut és exactament el mateix quan es carrega un nivell de les versions 'Arcade' i 'Nivells Personalitzats'

Tornem a l'exemple del diagrama de classes per a veure quin canvi s'ha produït (Pàgina següent):


MDB2
Variables
Mètodes
Settings
var $baseUrl: string
var $connection: MDB2
var $isMobile: bool
etc...
- private dbConnection();
- private mobileDetect();
etc...
Té instancia de
Té instancia de
Té instancia de
MobileDetect
Variables
Mètodes
Etc...
Index
Variables
Mètodes
GamePlay
Variables
Mètodes
UserHome
Variables
Mètodes
Etc...
Variables
Mètodes
Té instancia de
Fa crides a...
Té instancia de
Té instancia de
(S) GameController
var $instance: GameController
var $context: call class
+ public static getInstance()
+ Alters mètodes
(S) UserController
var $instance: UserController
var $context: call class
+ public static getInstance()
+ Alters mètodes
(S) Etc
var $instance: etc
var $context: call class
+ public static
AJAX (Remote)
+ Mètodes remots

Cal que el lector tingui en compte que les crides a AJAX no es fan des de la classe GamePlay ja que aquesta és una classe de PHP del back-end. Les crides a AJAX tenen lloc des del fitxer on es troba la classe GamePlay, però a la banda del front-end i per JavaScript. Per aquest motiu s'ha triat una línia discontínua per a la seva representació.

En aplicacions web un fitxer amb extensió '.php' pot intercalar fins a 5 tipus de llenguatges diferents (MTML, PHP, JavaScript, CSS i MySQL).

Si el lector se n'adona, aquest apartat de nivells s'està explicant dins la secció del Front-End però no és exclusivament així ja que s'està executant codi tant de la banda del client com de la banda del servidor. Quan hi ha operacions asíncrones resulta molt complicat classificar les accions, ara bé, s'ha triat explicar-ho al front-end ja que és des d'on es fan les crides i des d'on té sentit utilitzar els objectes de retorn de les funcions.

Arribats a aquest punt, el lector ja coneix de quina manera es creen els diferents nivells del joc de forma asíncrona, aleshores, és un bon moment per fer-lo coneixedor de com es resolen aquests nivells per poder donar pas als següents.

4.1.4.3. Front-end: Resolució d'un nivell qualsevol del joc

Quan el temps d'un nivell acaba el primer que cal determinar és si el jugador ha guanyat o ha perdut mitjançant un algorisme, i quan això està decidit, torna a entrar en acció la parella AJAX/JSON per actualitzar les estadístiques del jugador, avançar al següent nivell, calcular la puntuació, i moltes altres coses que cal fer del costat del servidor per desar-les a la base de dades. Tot aquest intercanvi de dades s'explica a continuació:

- **Si s'ha guanyat:**
    1. El client fa una crida per Ajax a un mètode remot del servidor per actualitzar-ne les estadístiques: incrementar el nombre de victòries, el nombre de robots salvats i el nombre de robots perduts. Aquest mètode no té cap retorn, únicament actualitza la base de dades, de manera que el client no necessita cap objecte de retorn.
    2. El client calcula les estrelles i la puntuació obtinguda pel jugador i cridant a un mètode remot fa el mateix que abans, actualitza la base de dades.

3. El client fa una crida asíncrona al servidor amb el nombre del nivell i de món actuals. El servidor incrementa el nivell i retorna el nou nombre de nivell i de món (si hi ha canvi de món) perquè tot quedi preparat per quan l'usuari es decideixi a fer clic al següent nivell.

- **Si s'ha perdut:**
  1. Es fa exactament el mateix excepte el punt 3, doncs no s'ha d'incrementar el nivell perquè el jugador ha perdut.

Cada vegada que es carrega un nou nivell apareixen les puntuacions d'aquell nivell en concret a la dreta de la pantalla. Bé, això s'obté de manera idèntica mitjançant crides per AJAX i ja no se n'expliquen els detalls. L'únic que cal saber és que el client sol·licita les puntuacions al servidor, aquest sol·licita les dades a la base de dades i les retorna com un objecte de JSON que el client rep, converteix a un objecte JavaScript i el fa servir per regenerar l'html de les puntuacions. Sempre és el mateix.

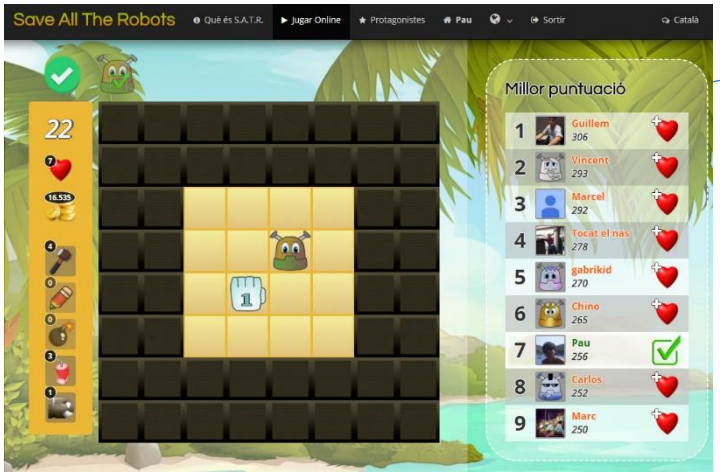


Figura 16. Puntuacions d'un nivell

Nota: A la versió 'Arcade', que enlloc de veure's les puntuacions es veuen els rècords de nivell dels usuaris. S'aconsegueix de la mateixa manera.

Una petita reflexió: -Quantes vegades caldria recarregar la pàgina si no s'utilitzés Ajax a cada partida? Tantes que seria molt incòmode jugar. Ajax és un dels pilars bàsics perquè l'elaboració de Save All The Robots com a aplicació web i no com a aplicació d'escriptori, hagi estat viable.

4.1.4.4. Optimització

Arribats a aquest punt el lector es podria preguntar el perquè de tanta comunicació asíncrona i pensar que seria millor descarregar totes les dades del servidor en una sola sol·licitud (mapes, nivells, puntuacions, mons, etc), guardar aquestes dades en objectes a memòria, aleshores jugar sense necessitat de transferència de dades i un cop acabat utilitzar una altra sol·licitud per desar els resultats. D'aquesta manera no caldria tenir cap petició asíncrona. Això no és viable per les següents raons:

- Com més petita sigui una pàgina més ràpid reacciona als canvis que hi fa l'usuari, per exemple, en una pàgina molt molt gran el simple fet de poder arrossegar els robots amb el ratolí no seria viable, es patirien delays inesperats.
- Caldria tenir en memòria objectes que no s'utilitzaran, per exemple, un usuari que es troba al nivell 2 del món 1 no té perquè tenir en memòria tots els models del nivell 9 del 8è món amb tots els seus enemics, robots i blocs. Entre 'Original', 'Arcade' i 'Nivells Personalitzats' el joc té més de 1000 models de nivells, i creixent cada dia...
- Caldria tenir processat l'html de tots els nivells de totes les versions del joc. Simplement, l'aplicació trigaria tant a carregar-se que els usuaris marxarien abans de començar.

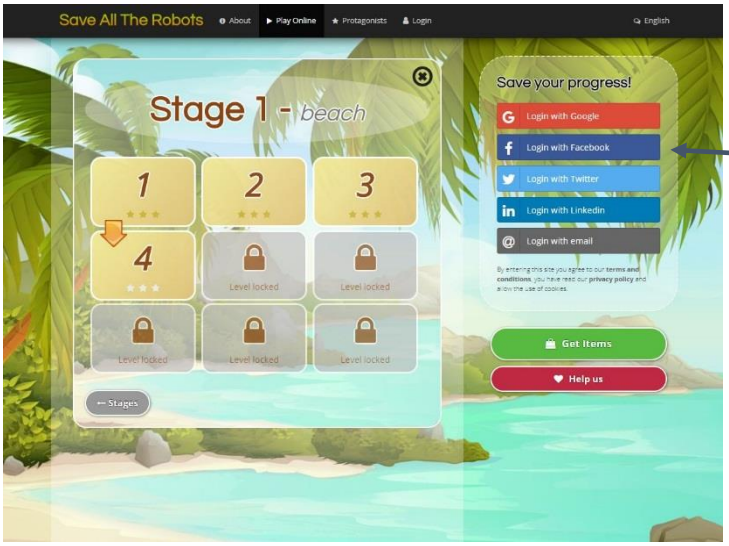
4.1.5. Desenvolupament de la xarxa d'usuari

Una vegada s'ha parlat de l'estructura del joc i les seves estratègies de desenvolupament, toca parlar del desenvolupament i funcionament de la xarxa d'usuari que és un aspecte molt important pel bon funcionament del joc.

Per afavorir la bona comprensió del lector es farà el recorregut per la xarxa des del punt en què l'usuari es registra al sistema i s'explicaran totes les funcionalitats que es trobarà un cop sigui usuari registrat.

1. L'usuari aterra a la web savealltherobots.com

Suposem que un usuari arriba al website, comença una partida i després de guanyar uns quants nivells s'adona que el joc li agrada. Aleshores està decidit a ser usuari del sistema amb l'objectiu de poder continuar amb la seva partida qualsevol altre dia.



El sistema insta constantment el jugador a guardar el seu progrés registrant-se al sistema, i ho fa de manera continuada, cada vegada que acaba un nivell, si detecta que l'usuari no té sessió.

A la imatge se suposa que un jugador ha jugat sense sessió fins al 4t nivell, que el joc li ha agradat i que es registra en aquell moment.

Figura 17. El sistema insta l'usuari a registrar-se constantment

2. L'usuari es registra

Quan l'usuari es registra el sistema li demana si vol continuar amb la seva partida o vol començar una partida nova, i ho pot fer ja que quan no hi ha sessió, la partida es guarda en una cookie durant 24 hores. (Això està fet perquè si un usuari no té intenció de registrar-se pugui jugar igualment).

Una vegada l'usuari ha triat opció aterra al seu 'userhome' que és la seva pàgina de gestió. Té el següent aspecte:

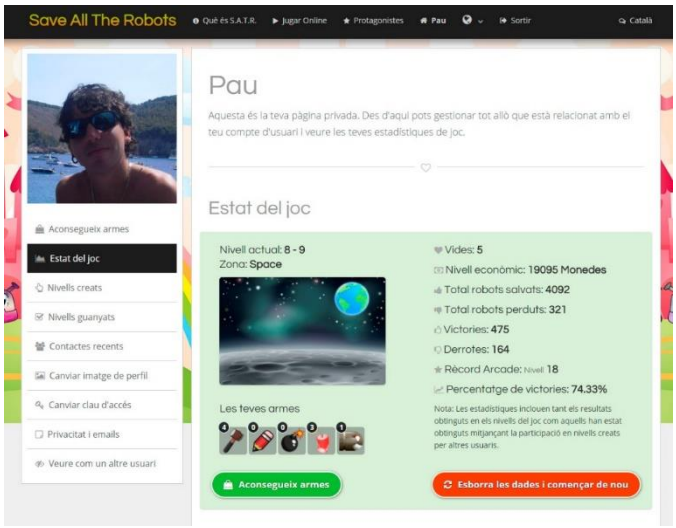


Figura 18. Pàgina d'usuari

La pàgina d'usuari està formada per 8 sub-pàgines privades d'usuari com es pot veure al menú de la dreta (sota la fotografia de perfil). A l'aterrar, el primer que veu l'usuari és el seu 'Estat del joc'. Analitzem-lo!

L'estat del joc mostra el nivell i el món al qual es troba el jugador, les vides que té en aquell moment, la quantitat de monedes que posseeix, el total històric del robots salvats, el total històric de robots perduts, l'històric de victòries, l'històric de derrotes, el percentatge de victòries i el nivell rècord que ha assolit a la versió Arcade del joc.
A més disposa d'un botó de color carbassa que serveix per re-sestejar totes les dades i tornar a començar una partida nova.

3. L'usuari explora les possibilitats

Suposem que l'usuari, que ja ha vist el seu userhome, continua la seva exploració utilitzant el menú de la dreta i tria l'opció 'Nivells Creats':

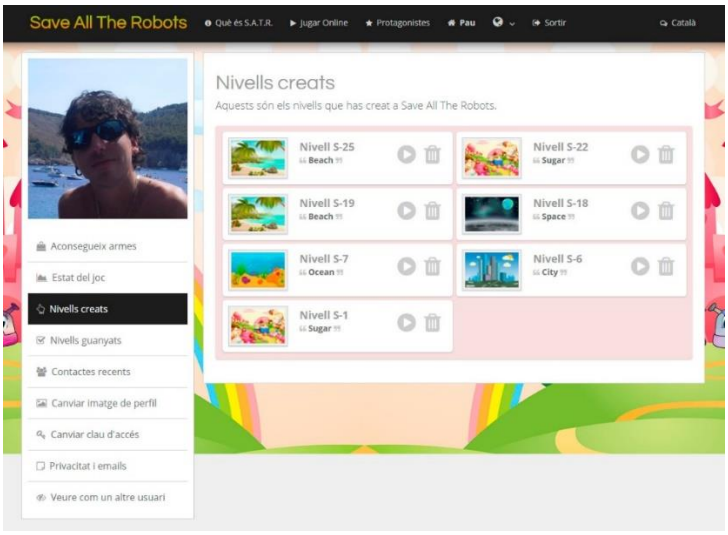


Figura 19. Nivells Creats

Aquesta pàgina mostra els nivells que el jugador ha creat utilitzant la modalitat 'Nivells Personalitzats'. El jugador té l'opció d'esborrar un nivell que ha creat si així ho veu convenient.

Suposem ara que l'usuari continua navegant i clica sobre l'opció 'Nivells Guanyats':

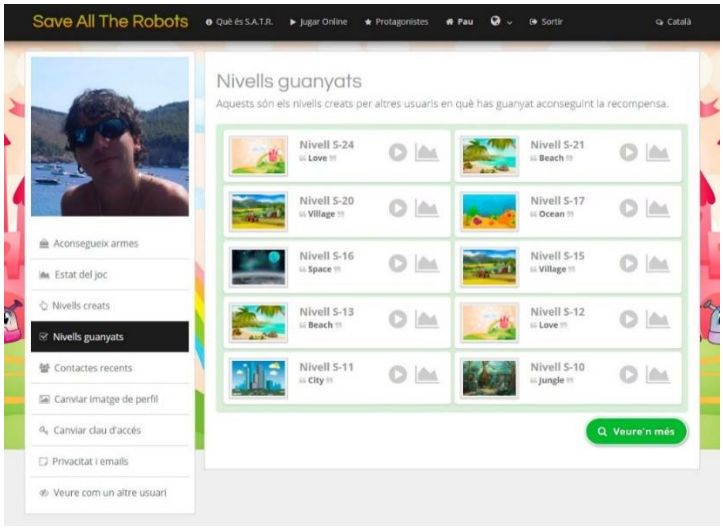


Figura 20. Nivells d'altres usuaris que el jugador ha guanyat

Apareix una llista dels nivells creats per altres usuaris en els quals el jugador ha participat, ha guanyat, i ha aconseguit la recompensa que el nivell oferia.

A tot això, el jugador continua navegant i en aquesta ocasió fa 'click' sobre la pestanya 'Contactes Recents':

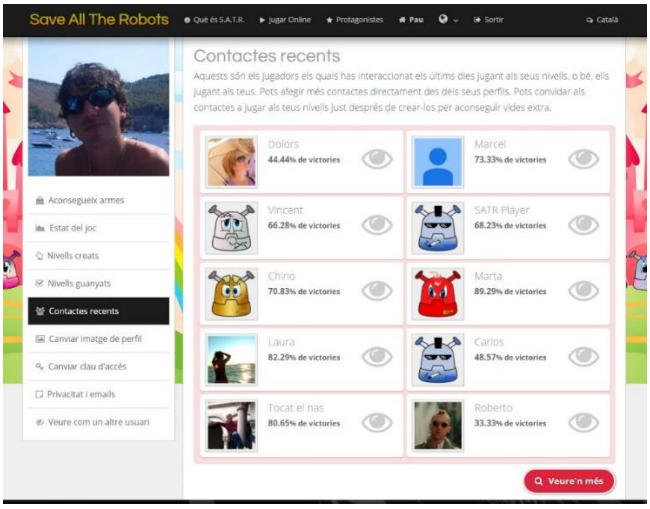


Figura 21. Contactes recents

Els contactes recents a Save All The Robots són les persones amb les quals s'ha interactuat de forma indirecta, és a dir, aquelles persones que han jugat als nivells que tu has creat o bé aquells usuaris els quals tu has jugat als seus nivells. Aquests contactes són les persones les quals un jugador pot reptar directament a superar un nivell que acaba de crear, és a dir, les persones a les quals pot enviar una invitació per jugar.

Més tard, el jugador segueix explorant i arriba a la secció de canviar la foto de perfil.

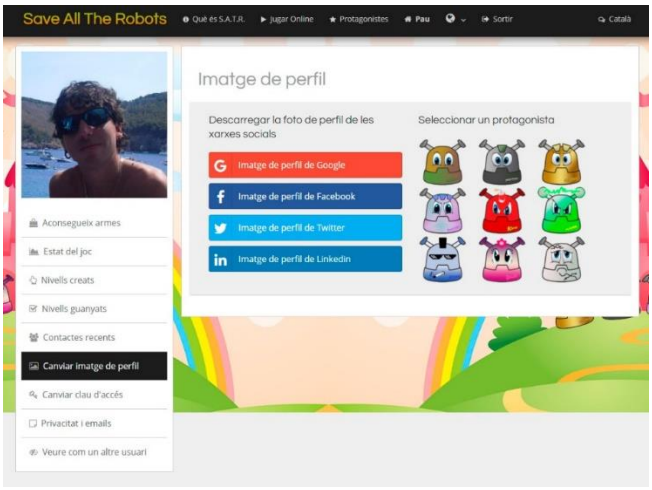


Figura 22. Imatge de perfil

S'ofereixen dues maneres de canviar la foto de perfil:

- Utilitzar la imatge de perfil de les xarxes socials: l'usuari pot triar entre la seva foto de perfil a facebook, twitter, google o linkedin.
- Utilitzar un Avatar protagonista del joc, un robot.

Quan l'usuari es registra, si s'ha registrat via email se li afegeix un avatar aleatori com a foto de perfil, en canvi si ha utilitzat les xarxes socials per a registrar-se, se li col·loca directament la seva foto de perfil de la xarxa social amb la qual fa el registre.

Passem a la secció de canvi de clau d'accés:

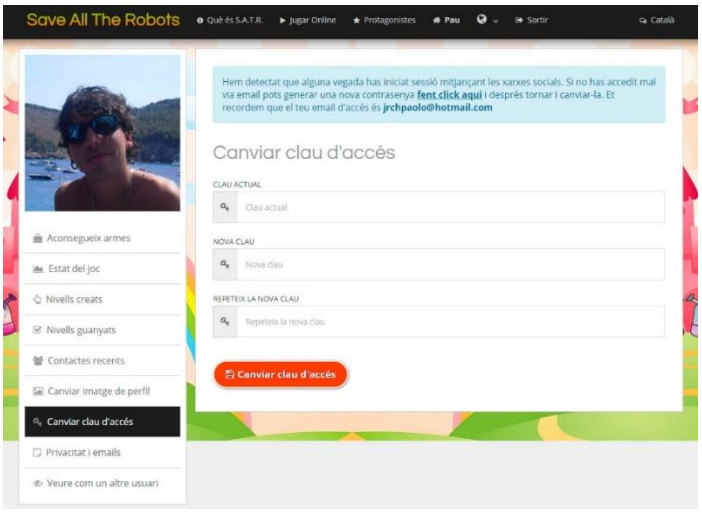


Figura 23. Canviar clau d'accés

Si l'usuari ha accedit via email ha necessitat registrar-se amb una clau d'accés que pot canviar quan vulgui. En canvi si ha accedit via xarxa social podrà sol·licitar una clau aleatòria, rebre-la al seu email i després canviar-la per la seva clau preferida. S'adverteix d'això al requadre blau superior de la figura 23.

L'ultima pàgina privada d'usuari és la pàgina de privacitat i emails des d'on l'usuari pot activar o desactivar els emails que vol rebre (Save All The Robots no pretén fer SPAM sense consentiment) i també pot donar-se de baixa del sistema:

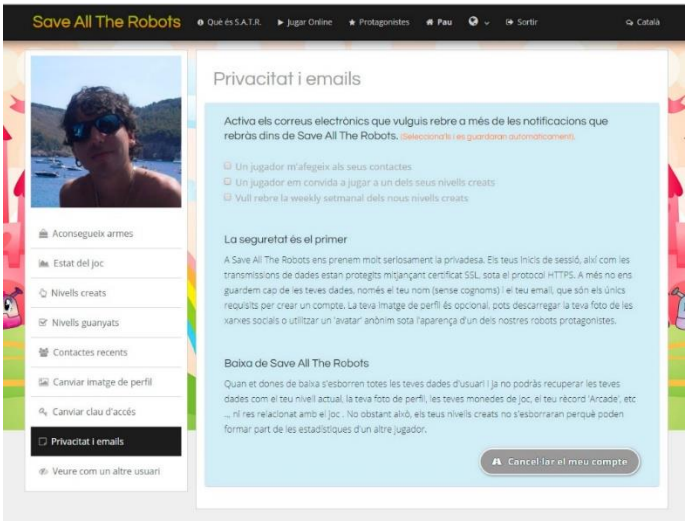


Figura 24. Privacitat

4.1.6. Botiga d'armes

Cada jugador pot accedir a comprar armes a la botiga del joc en el moment que vulgui. Si no té prou diners se l'instarà a jugar partides per guanyar-ne i després tornar a visitar la botiga:

Figura 25. Botiga d'armes

Quan el jugador adquireix una arma, es descompten les monedes del seu saldo i s'afegeix l'arma al seu inventari.

S'aprecia com també hi ha la possibilitat d'adquirir vides i monedes de joc per comprar armes a través de la plataforma de pagaments virtuals PayPal o també amb targeta de crèdit. Està tot implementat en PHP fent ús dels seus IPN corresponents.

4.1.7. Perfils públics

A Save All The Robots cada jugador té un perfil públic que mostra el les seves estadístiques bàsiques i un resum de nivells d'altres usuaris que ha superat i també dels nivells que ha creat. Una altra motivació al joc pot ben ser aconseguir tenir un perfil públic bo i complert. És el següent (figura 26):

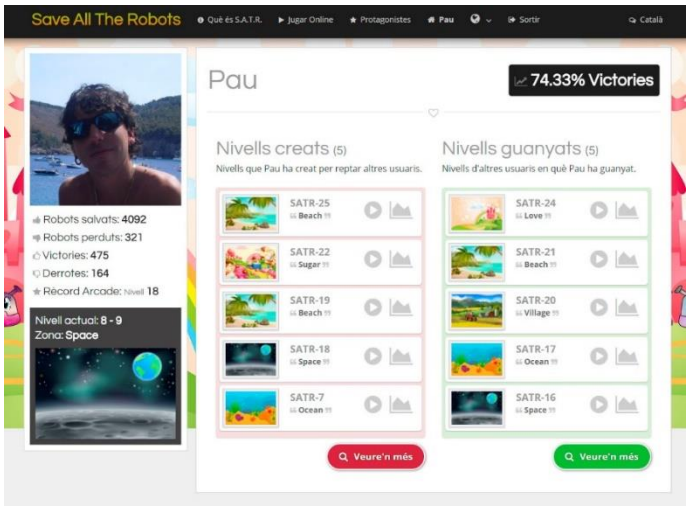


Figura 26. Perfil públic

4.2. MOTOR ALGORÍSMIC DEL JOC

Una de les parts de més pes de l'aplicació són els algorismes decisió, que decideixen coses com si s'ha superat o no un nivell, si un nivell és superable i per tant vàlid o no, com han d'atacar els enemics, etc. A continuació es mostren els algorismes més importants que s'han desenvolupat.

4.2.1. Algorisme de construcció dels nivells

Com s'ha esmentat anteriorment, les descàrregues via AJAX quan l'usuari ha seleccionat el nivell, han deixat una sèrie d'objectes a la banda del client, que corresponen als enemics, els robots, els blocs, els blancs i el temps del nivell. (Estem en el punt el qual un usuari ha accedit a un nivell per jugar).

Després d'això l'usuari fa clic al botó 'play'. Aquest botó és l'encarregat de construir la matriu del joc amb els objectes esmentats al paràgraf anterior.

Vegem doncs com es crea la matriu:

- La lògica del front-end, el primer que fa quan l'usuari clica sobre el botó 'play' és fer una crida a la funció de JavaScript que afegeix els objectes (enemics, blocs i blancs) a la matriu, i els robots a l'inventari.
- El primer que fa aquesta funció és buscar una transformació aleatòria de la matriu per evitar que un mateix nivell surti repetit. Les transformacions de la matriu estan basades en rotacions i transposicions i asseguren un taulell de joc diferent a ulls del jugador però simètricament idèntic.
- Una vegada s'ha transformat la matriu s'afegeixen els objectes de la següent manera:
    1. S'afegeixen els blancs
    2. S'afegeixen els blocs
    3. S'afegeixen els enemics (s'afegeixen els últims perquè ara caldrà obtenir les cel·les amenaçades.

4. Es col·loquen els robots a l'inventari apunt per ser arrossegats i/o seleccionats per part de l'usuari.

- S'obtenen les cel·les amenaçades

4.2.2. Transformacions de la matriu

Tal i com s'ha explicat en el punt immediatament anterior, el primer que es fa és buscar una transformació aleatòria de la matriu de joc per tal de donar dinamisme i realisme a la partida i que un mateix model de nivell pugui ser presentat de 16 maneres diferents.

La primera transformació que rep la matriu és una rotació. Es fa rotar el taulell sencer 90º, 180º, 270º o 0º.

La segona transformació és una mica més complexa. Cal fer les transposicions necessàries per deixar la matriu com si fos un mirall horitzontal, vertical o vertical + horitzontal. Vegem doncs uns exemples de transformacions de la matriu a continuació:

Figura 27. Transformacions de la matriu

Com es pot apreciar els 4 nivells anteriors representen el mateix, cosa que dificulta al jugador el fet d'acabar aprenent-se un nivell. L'exemple mostra només 4 maneres, però n'hi ha 16!

4.2.3. Algorisme per obtenir les cel·les amenaçades

Ara que es tenen tots els actors sobre el taulell i la matriu ha rebut una de les 16 transposicions possibles aleatòriament és moment d'obtenir les cel·les amenaçades i la seva quantitat d'amenaça. L'algorisme és el següent:

+ Per cada cel·la de la matriu:
    + Si la cel·la conté un enemic:
        + Per cada orientació d'atac de l'enemic:
            + Per cada cel·la en la direcció d'orientació d'atac de l'enemic
                + Incrementar el valor de l'amenaça en una unitat a cada cel·la fins a topar amb un altre actor (blanc, bloc o enemic) o esgotar l'abast d'atac de l'enemic.

Vegem un exemple gràfic d'això:

Figura 28. Taulell abans i després d'obtenir les amenaces

A la figura anterior es mostra l'estat del taulell (a mode de debug) de l'obtenció de les amenaces. Una cel·la més envermellida representa que té una amenaça més alta. Per exemple l'avant-penúltima fila de la primera columna té el nivell d'amenaça més elevat ja que un elèctric diagonal, un làser i un cop de puny estan amenaçant la cel·la al mateix temps. Amb menys intensitat trobem, per exemple, la cel·la de la fila immediatament inferior que està amenaçada només per un puny. Les cel·les que no tenen gens de vermell representen cel·les lliures d'amenaça.

Naturalment les amenaces no es veuen vermelles durant les partides reals dels jugadors. Això s'ha fet per dues coses: a mode de debug per facilitar la feina al desenvolupador i a mode d'exemple perquè el lector del document vegi amb més claredat el resultat de l'algorisme d'obtenció d'amenaces. A més com que en aquest punt el lector ja coneix tots els enemics, els seus atacs i els seus abasts d'atac li resultarà molt fàcil entendre-ho, i més encara amb la imatge anterior.

Nota: Si el lector encara no jugat al joc, se l'insta a fer-ho, si més no per verificar tot el que diu el document que està llegint i entendre amb més facilitat tot el que aquí s'explica i s'explicarà. https://www.savealltherobots.com

En aquest punt pot començar el compte enrere i començar a jugar col·locant els robots sobre les cel·les de la matriu.

4.2.4. Algorisme de col·locació de robots

Ara que el temps està funcionant vegem què succeeix quan el jugador fa l'acció de treure un robot de l'inventari i el col·loca sobre una cel·la del taulell.

Una vegada ja sabem la intensitat d'amenaça que té una cel·la segons la posició i l'orientació dels enemics, el que cal tenir en compte és el tipus de robot que s'està col·locant sobre la cel·la, ja que igual que succeeix amb els enemics, no tots els robots tenen les mateixes propietats, analitzem-ho!

El primer que cal fer és separar els robots en dos grans grups:

- Robots resistents: **Simple Robot**, **Metal Robot**, **Golden Robot** i **Diamond Robot**
- Robots especials: **Radio Robot**, **Boom Robot**, **Thug Robot** i **Girl Robot**

El grup dels robots resistents conté els robots que tenen característiques idèntiques i que l'única cosa que canvia és el seu grau de resistència als atacs. El segon grup conté robots que tenen un comportament extremadament diferents els uns dels altres i cal gestionar cada cas per separat. Per tant quan un robot és col·locat sobre una cel·la, el primer que cal mirar és de quin tipus de robot es tracta, i si el robot s'ha col·locat directament des de l'inventari o si s'ha fet un canvi de posició dins la mateixa matriu.

4.2.5. Algorismes de gestió dels robots resistents

A Save All The Robots hi ha dues possibles maneres de col·locar els robots: col·locar-los directament des de l'inventari o moure'ls de cel·la a cel·la. Aquestes dues maneres cal tractar-les algorísmicament de diferent manera ja que és un fet que afecta les cel·les amenaçades d'una forma diferent.

4.2.5.1. Col·locació d'un robot procedent de l'inventari

Quan es col·loca un robot des de l'inventari, el primer esdeveniment que té lloc és 'Obtenir cel·les lliures d'amenaça'. Aquest esdeveniment comprova quines amenaces de quins enemics està blocant el robot que s'ha col·locat, i per tant a quines cel·les ha de restar intensitat d'amenaça. Aquest algorisme s'executa cada vegada que l'usuari col·loca un robot, i tot i que la realitat és molt més complexa, es pot resumir de la següent manera.

```
+ Per cada enemic de la matriu:
        + Si l'enemic està amenaçant el robot:
                + Obtenir l'orientació en que el robot interseca amb l'amenaça
                + Si algun robot havia intersecat abans l'amenaça:
                        + No fer res i acabar
                + Si ningú havia intersecat abans l'amenaça:
                        + Si hi ha un robot en una posició posterior:
                                + Eliminar amenaça a les cel·les fins al robot posterior
                        + Si no hi ha ningú en una posició posterior:
                                + Eliminar amenaça fins topar amb bloc o acabar les cel·les
```

Ara que el lector coneix l'algorisme bàsic es dóna pas a un exemple de caràcter visual:

Estat del taulell de joc just abans que el jugador desplaci els robots de l'inventari a la matriu.

El jugador col·loca un robot a la fila 3 – columna 5. Com que el robot està blocant un raig làser en direcció 'Est' allibera una unitat d'amenaça a totes les cel·les de la fila afectades per aquest fins acabar el taulell.

Cel·les que alliberades de l'amenaça del raig làser.

El jugador col·loca un robot a la fila 6 columna 5 blocant l'amenaça d'un enemic elèctric en direcció Nord-Est.

Cel·les que alliberades de l'amenaça de l'enemic elèctric.

El jugador col·loca un robot a la fila 2 – columna 8 blocant l'atac d'un puny doble en direcció Sud.

Cel·les que alliberades de l'amenaça de l'enemic elèctric.

Figura 21. Debug de les amenaces

4.2.5.2. Canvi de cel·la d'un robot dins la matriu

Quan es mou de cel·la un robot cal tenir en compte dos aspectes: el primer és el mateix que abans, obtenir cel·les lliures d'amenaces tenint en compte les interseccions que provoca el robot amb les amenaces dels enemics a la cel·la a la qual va a parar. El segon és que cal restablir les amenaces tenint en compte la cel·la que el robot deixa lliure, és a dir, restablir les amenaces naturals de la cel·la de sortida i obtenir cel·les lliures d'amenaça a partir de la cel·la d'arribada.

A continuació es mostra l'algorisme que restableix les amenaces a les cel·les a partir de la cel·la de sortida d'un robot. (Una vegada més val a dir que la realitat és molt més complexa i que aquí es mostra l'algorisme bàsic).

```
+ Per cada enemic de la matriu:
        + Si l'enemic està amenaçant la cel·la de sortida:
                + Obtenir l'orientació en que la cel·la interseca amb l'amenaça
                + Si algun robot havia intersecat abans l'amenaça:
                        + No fer res i acabar
                + Si ningú havia intersecat abans l'amenaça:
                        + Si hi ha un robot en una posició posterior:
                                + Restablir l'amenaça a les cel·les fins al robot posterior
                        + Si no hi ha ningú en una posició posterior:
                                + Restablir amenaça fins topar amb bloc o acabar les cel·les
```

És fàcil apreciar que aquest segon algorisme és proporcionalment invers a l'anterior. És a dir que restablir amenaces és inversament proporcional a alliberar-les. Cal seguir els mateixos passos però en un ordre invers.

Per a mostrar un exemple visual s'ha optat per seguir l'exemple anterior en el punt al qual havia quedat.

Partim de la situació anterior on el jugador havia col·locat 3 robots a la matriu de joc.

El jugador mou el robot que bloquejava el làser col·locant-lo a l'altra banda.

Això provoca dues crides: una a l'algorisme que allibera l'amenaça de les cel·les sobre la cel·la d'arribada. La segona crida restableix les amenaces segons la cel·la de sortida del robot.

Aquí s'han restablert les amenaces!

Aquí, en canvi, s'ha alliberat l'amenaça de les cel·les.

Figura 30. Debug del restabliment de les amenaces

4.2.6. Algorismes de gestió dels robots especials

En el cas dels robots especials, tot i haver de considerar casos afegits, també s'utilitzen els algorismes d'alliberació i restabliment d'amenaces, amb l'únic afegit que han d'aplicar més algorismes per contemplar aquells casos que els fan, precisament, robots especials.

4.2.6.1. Algorisme de gestió del Robot Radio (Robot Desactivador)

El Robot Radio, com el lector ja sap, té l'habilitat de desactivar els enemics que té a la vora privant així el seu atac final.

Una possible manera d'haver fet això hagués estat la següent:

- Primer aplicar els algorismes d'alliberament i restabliment d'amenaces.
- Després cercar les cel·les veïnes al robot, recórrer-les i alliberar les amenaces dels enemics que es troben en aquetes cel·les en totes les seves direccions.
- Restablir les amenaces dels enemics afectats per la desactivació una vegada el robot es mou de cel·la i fer-ho en totes les direccions d'atac.
- Tenir en compte casos extrems, per exemple, un enemic pot ser que estigui parcialment blocat per un altre robot, caldria tenir-ho en compte alhora d'alliberar les amenaces quan es desactiva tal enemic...etc...

Aquesta és una possible implementació algorísmica per gestionar els robots especials, ara bé, no s'ha fet així. Masses casos, masses tasques a realitzar cada vegada que un jugador mou un robot. Cal tenir en compte que ens trobem tot just al primer robot especial i n'hi ha tres més. Cal buscar una solució més simple ja que no podem permetre que un algorisme trigui a resoldre's, el jugador no pot adonar-se de res. Ha de ser ràpid! Com veurem la solució triada no només soluciona això sinó que a més soluciona altres qüestions de seguretat que es veuran als capítols finals, i a més ho fa de forma simple, que no pot donar lloc a cap error de càlculs.

L'algorisme final funciona de la següent manera:

- S'aprofiten els algorismes d'alliberament i restabliment d'amenaces i les posicions de tots els robots es van desant en un vector de posicions.

  Cada vegada que un jugador col·loca un robot desactivador es destrueix la matriu sencera, s'assigna 'null' a l'objecte que conté la matriu i durant mil·lèsimes de segon la matriu de joc deixa d'existir.

  Aquí cal fer un incís: si el lector ho recorda, teníem tots els objectes de la matriu a memòria en diccionaris de JavaScript provinents d'una crida al servidor mitjançant AJAX que tenia lloc al carregar el nivell. Hi segueixen sent! El que es fa aleshores és tornar a formar la matriu col·locant els robots (n'hem guardat les posicions en un vector!), els blocs i els blancs els enemics (això té complexitat lineal amb una n = (files * columnes) = 64, és molt barat computacionalment, més que calcular algorísmicament tots els casos dels robots especials!).

El cas especial es dóna en el moment de col·locar els enemics. Si el lector ho recorda, al carregar un nivell cada vegada que es col·locava un enemic al taulell se n'extreien les amenaces s'assignaven a les cel·les. Ara, en canvi, cada vegada que es col·loca un enemic es recorren les seves 8 cel·les veïnes i es comprova si alguna de les cel·les conté un robot desactivador i si és així no s'extreuen les amenaces de l'enemic i per tant no s'assigna a les cel·les l'amenaça dels enemics desactivats.

Cal notar que l'enemic desactivat es col·loca igualment al taulell ja que pot estar fent d'escut casual de l'atac d'un altre enemic, aleshores el que es fa és aplicar un efecte visual perquè el jugador sàpiga que l'enemic està desactivat

Vegem què succeeix aplicant aquesta manera de fer:

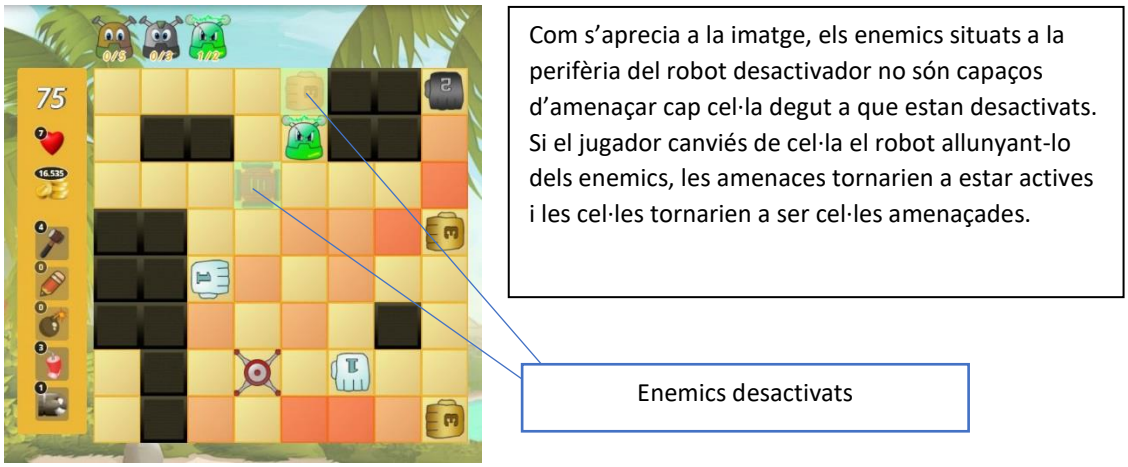


Figura 31. Algorisme del Radio Robot

Però encara ho podem fer millor...

L'anterior manera de fer és gairebé perfecta ja que no necessita més algorismes dels que són estrictament necessaris, i a més, són extremadament ràpids, però encara poden millorar més...

Per explicar-ho cal fer un petit incís: *tot això té lloc a la computadora del client. El desplegament de les amenaces, els algorismes de col·locació dels robots i totes les accions que tenen lloc entre la càrrega del nivell i la finalització del temps no necessiten*

connexió a internet i s'executen directament a la lògica del navegador i per tant cal aprofitar-se d'aquest fet i treure'n el màxim profit.

Aleshores s'aplica la següent manera de fer, que segueix l'esquema de l'anterior però és molt més intel·ligent (aquesta és la manera definitiva):

- Quan el jugador col·loca un robot s'apliquen els algorismes d'alliberament i restabliment d'amenaces però no es té en compte el tipus de robot. La lògica del joc en aquest punt no distingeix el tipus de robot sinó que els tracta tots per igual.
- Ara doncs quan es col·loca un robot desactivador aquest no desactiva els enemics sinó que només s'aplica l'efecte visual per 'enganyar' al jugador i que tingui la sensació que estan desactivats.
- Per tant, la matriu del joc no s'ha destruït ni tornat a crear en cap moment mentre el jugador està jugant.
- Quan finalitza el temps, ara si, es destrueix la matriu i es torna a crear seguint exactament la manera de fer abans explicada i quan es col·loquen els enemics, no s'extreu l'amenaça d'aquells que tenen un robot desactivador a la perifèria, i que per tant, estan desactivats.

El que es guanya amb aquesta manera de fer és no haver de crear i destruir la matriu i tots els seus elements cada vegada que es col·loca un robot especial, **la matriu només es refarà un sol cop,** quan acaba el temps.

4.2.6.2. Algorisme de gestió del Robot Explosiu

El robot explosiu té la particularitat que explota quan acaba el temps i destrossa els enemics de la seva perifèria però també mata els robots que té a la vora, sigui quina sigui la seva resistència.

Aquest robot té una part que es pot tractar de manera idèntica que l'anterior: la seva relació amb els enemics. Aquí enlloc de desactivar-los els destrueix però algorísmicament es pot tractar de la mateixa manera.

La diferència rau en què, al manipular aquest robot, a més de tenir en compte els enemics de la perifèria cal tenir en compte si hi ha altres robots a la vora. Això es pot fer utilitzant el mateix recorregut, només caldrà afegir una condició: 'si hi ha algun robot en alguna cel·la de la perifèria'.

A més té una altra diferència: es mostra un efecte visual de les cel·les que explotaran.

Vegem un exemple de tractament de robot especial:

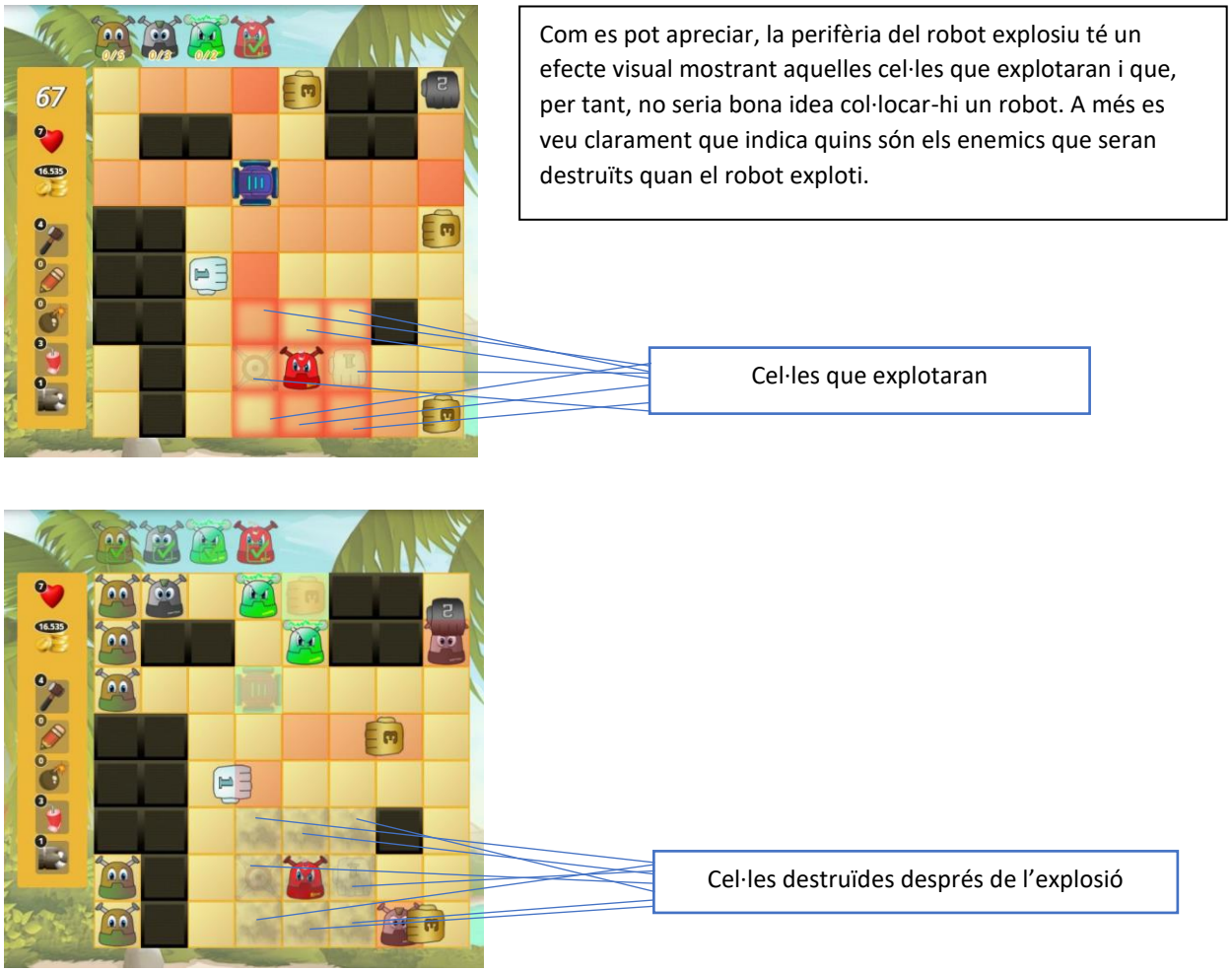


Figura 32. Algorisme del robot explosiu

4.2.6.3. Algorisme de gestió del Robot Girl

La manera de fer establerta per tractar aquest robot, és molt diferent a la realitzada en els dos robots especials anteriors però tots tres comparteixen un aspecte comú: - Es poden tractar només al final -.

Així doncs els únics algorismes que s'aplicaran a aquest robot en tems de joc seran els que alliberen i restableixen amenaces, tal i com es fa als robots més simples.

L'algorisme del Robot Girl té lloc després de refer la matriu, és a dir, quan el temps acaba es tenen en compte primer els algorismes dels robots especials 'desactivador i explosiu' i ja sobre la nova matriu reconstruïda, tenen lloc els algorismes que gestionen el Robot Girl i el Robot Thug (que es veurà més endavant).

Vegem com funciona l'algorisme del Robot Girl:

+ Obtenir les coordenades de la matriu que contenen robots del tipus girl
+ Per cada coordenada de robots girl:
    + Generar un enter aleatori entre 0 i 5
    + Si l'enter anterior és inferior a 2.5:
        + No fer res el robot no es mourà del lloc
    + Si l'enter és superior a 2.5:
        + El robot canviarà de cel·la per anar a una cel·la veïna buida
        + S'obtenen les cel·les veïnes que estan buides amb un recorregut
        + Es tria una cel·la veïna a l'atzar
        + Es canvia de cel·la el robot
        + S'apliquen els algorismes d'alliberament i restabliment d'amenaces

Nota: recordar que el 'robot girl' canviaria de cel·la amb una probabilitat del 50% i si ho feia només podia anar a una cel·la veïna que estigués buida.

Vegem un exemple gràfic del comportament de l'algorisme del robot girl:

Figura 33. Algorisme del Robot Girl

4.2.6.4. Algorisme de gestió del Robot Thug

Aquest robot, com ja sabem, si es col·loca a la perifèria d'un robot resistent (metàl·lic, daurat o diamant), aquest últim perd la seva resistència convertint-se en un robot simple.

Aleshores cal fer dues coses:

- Canviar l'aparença visual dels robots resistents afectats i convertint-los en robots simples
- Canviar la resistència del robot a la variable corresponent i posar el valor de resistència a 0.

L'estratègia és la mateixa de sempre: es canvia l'aparença visual a cada moviment però no cal alterar el valor de la resistència fins l'última jugada. L'algorisme funciona de la següent manera:

+ Per cada robot dels robots del taulell:
    + Si el robot és del tipus 'thug':
        + Per cada cel·la veïna al robot:
            + Si la cel·la conté un robot resistent:
                + Canvia l'aparença del robot a 'robot simple'
                + Si és la jugada final (temps exhaurit):
                    + Posa la resistència del robot a 0

Vegem un exemple visual de com treballa l'algorisme:

Situació del taulell abans de col·locar els Thug Robots

Fixem, però, la mirada en aquests dos robots resistents, que aparentment, resistirien l'atac de l'amenaça a la qual estan sotmesos

Situació del taulell després de col·locar els Thug Robots

Els robots resistents han passat a ser robots simples, perdent així la seva capacitat per resistir qualsevol atac!

Situació del taulell quan ataquen els enemics

Com que els robots han perdut la seva màgia, exploten al no poder resistir els atacs, i per tant, acaben morint.

Figura 34. Algorisme del Robot Thug

4.2.7. Algorisme per decidir si s'ha superat un nivell

Tècniques per elaborar algorismes n'hi ha tantes que cada autor podria fer una proposta diferent i seria bona. Ara bé hi ha dues maneres diferenciades de fer-ho:

1. **Un algorisme de cerca decideix si s'ha superat el nivell**. Aquest hauria de tenir en compte les direccions de les amenaces, l'abast dels enemics, el tipus de robots del nivell i les posicions que ocupen aquests.

- **Avantatges**: un sol algorisme i una sola execució.
- **Inconvenients**: poca escalabilitat, si s'amplia el tipus de robots o enemics aquest algorisme ja no serveix i s'ha de canviar sencer. A més, com més robots i enemics diferents hi hagi, més casos cal tenir en compte i més augmenta la complexitat de l'algorisme i si el temps de resolució és elevat pot produir espais temporals de congelació entre la finalització del temps de nivell i l'atac dels enemics (un delay bloquejant).

2. **Diàleg constant usuari-sistema**: Cada vegada que el jugador col·loca un robot en una cel·la es recalculen totes les amenaces dels enemics que intersequen amb el robot col·locat produint un 'diàleg home-màquina' a cada pas.

- **Avantatges**: menys casos a contemplar, doncs es tenen en compte només els enemics involucrats i un sol tipus de robot, el que s'està col·locant en aquell moment. La matriu va quedant recalculada en temps real i no es depèn de la decisió final d'un sol algorisme. Més velocitat. Disminueix la complexitat, doncs cada càlcul individual té complexitat lineal, només cal recórrer en la direcció de l'amenaça fins que acaba el taulell o topa amb un altre actor. Es pot debugar de manera visual posant i traient color a les cel·les amenaçades a mesura que anem jugant (això ho veurem). La decisió final dependrà només de si s'han col·locat tots els robots o no, i de si hi ha o no cel·les per explotar (cel·les on el nivell d'amenaça supera la resistència del robot que l'ocupa).
- **Inconvenients**: Més d'un algorisme. Tantes execucions com moviments faci el jugador.

L'autor alhora de desenvolupar la lògica del joc ha triat la *segona opció,* de manera que tot el que s'ha explicat anteriorment i tot allò que s'expliqui a partir d'aquí es basa en un diàleg 'usuari-sistema' que té lloc a cada moviment.

4.2.8. Algorismes de resolució del nivell

Com que s'ha mantingut un 'diàleg usuari-sistema' on els algorismes han tingut lloc a cada jugada que ha fet el jugador, no es necessita un algorisme explícit que calculi la solució del nivell.

La solució del nivell té complexitat lineal O(n) i n'hi ha prou en recórrer la matriu un sol cop i comprovar dues coses:

- Que el jugador ha col·locat **tots els robots**
- Que no hi ha cap robot col·locat amb **resistència < quantitat d'amenaça**

El nivell s'ha anat resolent sol a mesura que el jugador ha anat jugant, i s'ha fet alliberant i restablint amenaces a cada moviment del jugador. És just en aquest punt o s'agraeix tota la feina realitzada als passos previs, una vegada acaba el temps no cal fer cap càlcul algorísmic i pot passar directament als algorismes de xarxa (abans explicats) que interactuaran amb el back-end per desar les dades i actualitzar estadístiques. I això si que té cost!

Per tant el que s'ha guanyat és donar fluïdesa al codi i no provocar 'coll d'ampolla' al finalitzar un nivell, doncs de no fer-ho així, un cop hagués acabat el temps, el temps computacional dels algorismes de xarxa s'hagués sumat al temps dels algorismes per trobar la solució al nivell. Ens ho hem estalviat.

4.2.9. Algorismes de gestió dels atacs

Els atacs dels enemics de Save All The Robots tenen lloc quan el temps acaba, independentment que l'usuari guanyi o perdi el nivell.

Els efectes dels atacs estan elaborats amb imatges d'extensió '.gif' en alguns casos com les flames, o directament amb efectes de jQuery en altres casos com el moviment dels punys i els rajos vermells.

L'algorisme funciona de la següent manera:

```
+ Per cada enemic del taulell:
        + Per cada orientació d'atac de l'enemic:
                + Si l'enemic no està desactivat:
                        + Nombre de cel·les a atacar = 0
                        + Per cada cel·la en direcció a l'atac que és inferior a l'abast de l'enemic:
                                + Nombre de cel·les a atacar += 1
                                + Si hi ha un bloc, enemic, o robot:
                                        + Break per deixar de mostrar efectes a les següents
                        + Comprovar quin tipus d'enemic és a través del seu atribut nom
                        + Mostrar l'efecte a les cel·les segons el seu abast, el tipus d'enemic i orientació
```

4.2.10. Algorismes de gestió de les armes

En aquest punt el lector ja ha jugat al joc una infinitat de vegades i ha testat tot el que s'ha explicat fins aquest punt. De tant de jugar, haurà guanyat una quantitat considerable de monedes de joc. Ara és el moment que el lector vagi la botiga del joc i compri una arma de cada per testar el que s'explicarà en aquest apartat.
Si el lector ho prefereix trobarà vídeos demostratius al següent enllaç:
https://www.savealltherobots.com/ca/page/items/

Totes les armes actuen durant el temps de joc, és a dir que els algorismes actuen cada vegada que el jugador col·loca una arma sobre una cel·la de la matriu de joc, a més, els efectes visuals es disparen de manera instantània, en temps real, mentre l'usuari juga.

4.2.10.1. Algorismes del MARTELL

El Martell només pot actuar sobre les cel·les que contenen un bloc i el seu algorisme funciona de la següent manera cada vegada que un jugador l'utilitza:

```
+ Si la cel·la conté un bloc:
        + La cel·la deixa d'estar ocupada
        + Es fa explotar el bloc amb un efecte visual
        + Per cada bloc dels blocs del nivell:
                + Si el bloc té les coordenades de la cel·la:
                        + S'elimina el bloc dels blocs del nivell
+ Si no:
        + No fer res i ja que el martell només pot actuar sobre un bloc
```

Es pot veure un exemple visual del martell seguint el següent enllaç:

https://www.savealltherobots.com/ca/page/item/hammer

4.2.10.2. Algorismes Llapis

El llapis 'fabrica' una cel·la on hi havia un espai blanc. Aquesta cel·la podrà ser utilitzada per col·locar un robot. L'algorisme funciona de la següent manera:

```
+ Si la cel·la conté un blanc:
        + La cel·la deixa d'estar ocupada pel blanc
        + Es construeix una cel·la amb un efecte visual
        + Per cada blanc dels blancs del nivell:
                + Si el blanc té les coordenades de la cel·la:
                        + S'elimina el blanc dels blancs del nivell
+ Si no:
        + No fer res i ja que el llapis només pot actuar sobre un blanc
```

Es pot veure un exemple visual del llapis seguint el següent enllaç:

https://www.savealltherobots.com/ca/page/item/pencil

4.2.10.3. Algorismes Dinamita

La dinamita destrossa els enemics que troba en una àrea de 9 caselles al voltant de la dinamita, és a dir, fa esclatar les seves cel·les veïnes manant-ne els enemics que hi hagi allotjats. L'algorisme funciona de la següent manera:

```
+ Si la cel·la conté el que sigui excepte un blanc:
        + Obtenir les cel·les veïnes de la cel·la d'impacte
        + Per cada cel·la veïna:
                + Si la cel·la veïna no és un blanc:
                        + Aplicar l'efecte visual d'explosió
                        + Si la cel·la conté enemic:
                                + Deshabilitar l'enemic
```

Es pot veure un exemple visual de la dinamita seguint el següent enllaç: https://www.savealltherobots.com/ca/page/item/dynamite

4.2.10.4. Algorismes de la bomba

La bomba destrossa els enemics en creu, és a dir, l'explosió té un abast perpendicular infinit matant tots els enemics que troba fins a trobar un bloc, un blanc, o s'acaba el taulell. L'algorisme funciona de la següent manera:

```
+ Si la cel·la no conté un blanc o un bloc:
        + Per cada orientació (Nord, Sud, Est, Oest):
                + Per cada cel·la en aquella orientació:
                        + Si la cel·la no és un blanc o un bloc:
                                + Fer explotar la cel·la
                                + Si la cel·la conté enemic:
                                        + Deshabilitar l'enemic
                        + Si no:
                                + Break
```

Es pot veure un exemple visual de la bomba seguint el següent enllaç:
https://www.savealltherobots.com/ca/page/item/bomb

4.2.10.5. Algorismes del Míssil:

El míssil recorre una fila sencera de la matriu i trenca tots els objectes que troba durant el seu recorregut, ja siguin blocs o enemics. El seu algorisme funciona de la següent manera:

```
+ Per cada cel·la de la fila:
        + Afegir efecte d'explosió a la cel·la
        + Si la cel·la conté bloc o  blanc:
                + Fer desaparèixer el bloc o el blanc i alliberar l'ocupació la cel·la
        + Si la cel·la conté enemic
                + Fer desaparèixer l'enemic, deshabilitar-lo i des-ocupar la cel·la
```

Es pot veure un exemple visual del Míssil seguint el següent enllaç:
https://www.savealltherobots.com/ca/page/item/missile

4.2.11. Algorisme per calcular les estrelles

Quan se supera un nivell cal calcular el nombre d'estrelles que ha obtingut el jugador per haver guanyat. Les estrelles van directament lligades al temps amb què s'ha superat el nivell, com menys es triga a passar-se el nivell més estrelles es guanyen.

L'algorisme funciona de la següent manera:

+ Obtenir el percentatge de temps per superar el nivell de la següent manera:

percentatge = temps que triga el jugador a guanyar * 100 / temps del nivell

+ Si percentatge > 60:
    + Obté 3 estrelles
+ Si no, Si percentatge > 40:
    + Obté 2 estrelles
+ Si no, Si percentatge > 20:
    + Obté 1 estrella
+ Si no:
    + Obté 0 estrelles

4.2.12. Algorisme per calcular la puntuació

Un cop obtingudes les estrelles, el proper pas és calcular la puntuació que ha obtingut el jugador per un determinat nivell.

L'algorisme, que depèn doncs de les estrelles queda de la següent manera:

+ Obtenir les estrelles aconseguides
+ Obtenir la quantitat de punts per robots salvat (depèn del nivell)
+ Calcular → Puntuació per robot * nombre de robots
+ Aplicar un factor que depèn del món en el qual es troba el nivell (si el món és més alt, més puts s'obtenen):
    Puntuació *= (nombre de món / nombre total de mons) + 1
+ Puntuació *= estrelles aconseguides
+ Aplicar un petitíssim factor aleatori per evitar repetir moltes vegades una mateixa puntuació

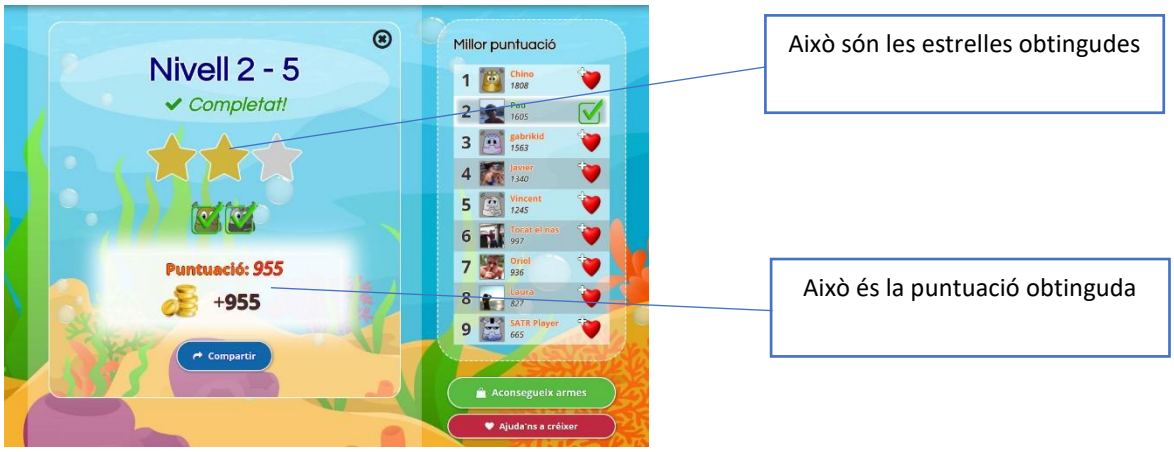


Figura 35. Estrelles i puntuacions

4.2.13. Desenvolupament de les versions 'Arcade' i 'Nivells Personalitzats'

Una vegada s'han vist els algorismes genèrics del joc, cal explicar breument el desenvolupament de les dues modalitats extres que ofereix el joc, recordant que aquestes modalitats també utilitzen els algorismes explicats a l'apartat anterior.

4.2.13.1 Desenvolupament de la versió Arcade

La versió 'Arcade' del joc té la particularitat que els nivells són infinits i que no està distribuïda en zones, sinó que tot el joc transcorre dins un sol món, l'Arcade, que va canviant d'aspecte a cada nivell, adoptant els dissenys dels diferents mons del joc original de manera aleatòria.

Per exemple pot ser que el nivell 1 transcorri a la platja i que el nivell 2 tingui lloc a l'espai, i així successivament i absolutament aleatori.

Però, com es fa que els nivells siguin infinits?

- La primera opció és fer un generador de nivells algorísmic. Tot i que aquesta idea és molt bona i va ser implementada amb èxit a la versió del joc per Android de l'assignatura Projecte Integrat de Software, té un gran problema: - construïa nivells vàlids però que no sempre eren divertits i agradables de forma visual, dues coses absolutament necessàries per 'enganxar' a l'usuari al joc.

- La segona opció és considerar un model pseudo-aleatori amb molts models per nivell. S'ha triat aquesta opció i s'explica a continuació:

4.2.13.1.1. Arcade: Model infinit i aleatori

El que s'ha fet és crear els nivells utilitzant l'editor de nivells i desant-los a la base de dades tal i com es va fer a la versió original del joc, però enlloc de crear 2 models per nivell se n'ha creat 4 i s'ha utilitzat el mateix sistema de transposicions de la matriu. Ara entre models i transposicions tenim:

4 models * 4 rotacions * 4 transposicions = 64 models per nivell

Ara la probabilitat de repetir un nivell és 1/64 aconseguint una aleatorietat gairebé absoluta però creant nivells bonics visualment i sobretot divertits.

Tot això soluciona el problema de l'aleatorietat, ara bé, cal veure com es fa que els nivells siguin infinits:

D'entrada la versió 'Arcade' té 30 nivells, de manera que a partir que l'usuari supera el nivell 30 (encara no ho ha aconseguit ningú) es tria un nivell aleatori entre el 5 i el 30, i així successivament fins que l'usuari mor i es veu obligat a tornar a començar al nivell 1.

La intenció de l'autor és anar incrementant mica en mica els 30 nivells fins arribar, per exemple als 50, tot i que de moment 30 és molt més que suficient ja que ningú ha passat mai del 19.

La implementació dels algorismes dels robots, enemics i armes és exactament igual als que s'han explicat a la versió original del joc.

4.2.13.2. Editor de nivells personalitzats

El desenvolupament de l'editor de nivells està desenvolupat per fases. Tot i que utilitza els algorismes de resolució de les altres modalitats, aquesta és la part més complexa i difícil de construir del TFG. Només a la pantalla principal de creació s'aprecia ja que es

poden arrossegar o clicar tots els ítems del joc i per tant tots han d'estar en correspondència amb tots.

Suposem que un usuari vol crear un nivell, vegem quins passos segueix i què va succeint a cada pas.

1. L'usuari construeix el seu nivell arrossegant els enemics, els blancs, els blocs i els robots cap a la matriu de joc. Hi ha dues coses a tenir en compte quan es construeix un nivell per tal que aquest sigui vàlid:
   - ➔ Que contingui almenys un robot i un enemic
   - ➔ Que no mori cap robot

   És així perquè no té cap sentit un nivell sense robots o un nivell sense enemics, en canvi si que es concebeix la possibilitat de crear un nivell sense blocs o sense blancs, ja que aquests són actors secundaris del joc.

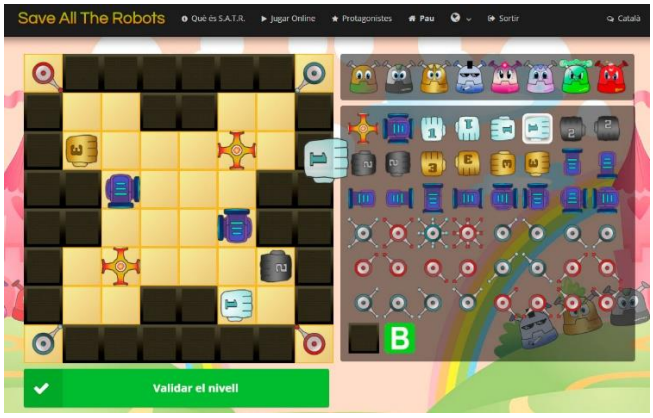


Figura 36. Editor de nivells

2. Quan l'usuari clica sobre el botó 'Validar nivell', a part dels algorismes de resolució que ja coneixem, té lloc una acció rellevant: tant si el nivell és vàlid com si no, es guarda una 'fotografia' de la matriu tal i com està ara per si l'usuari en algun moment decideix tornar enrere i editar-la. (Una fotografia de la matriu és una còpia de l'objecte en el seu estat actual).

3. L'usuari valida el seu nivell, vegem què passa si no col·loca cap robot i intenta validar l'exemple anterior:

Figura 37. Nivell Invàlid

El sistema s'ha queixat ja que el taulell de joc no contenia cap robot i els robots són necessaris a qualsevol nivell perquè són els protagonistes principals del joc. L'usuari se n'adona, col·loca els robots i valida el nivell:

Figura 38. Nivell Vàlid

4. El sistema insta a l'usuari a seleccionar el temps del nivell mitjançant el botó verd 'Seleccionar el temps'. El jugador fa click sobre el botó i selecciona el temps pel nivell.

Figura 39. Seleccionant el temps

5. El jugador ha seleccionat 51 segons, ara se l'insta a superar el seu nivell amb el temps que ell mateix ha triat. D'aquesta manera es comprova si el nivell és assequible per altres jugadors. Si no és capaç de superar-lo se l'instarà a tornar-ho a provar augmentat el temps i així fins obtenir l'equilibri temps-dificultat esperat.

Figura 40. Tems seleccionat i apunt per jugar

6. En aquest punt, just quan el jugador ha seleccionat el temps i abans de testejar el seu propi nivell es fa una altra 'fotografia' de la matriu però ara només amb els enemics, blancs i blocs. D'altra banda es crea l'inventari de robots. Aquestes dues coses seran utilitzades al proper pas, on l'usuari ha de jugar el seu propi nivell i superar-lo per validar-lo definitivament.

Figura 41. Jugant el nivell

7. L'usuari juga al seu nivell per intentar superar-lo amb el temps que ha triat:

Figura 42. Nivell Superat

8. L'usuari supera el seu nivell, això vol dir que ara el nivell és absolutament vàlid tant per protagonistes com per temps, per tant tria l'opció 'Guardar i Continuar'

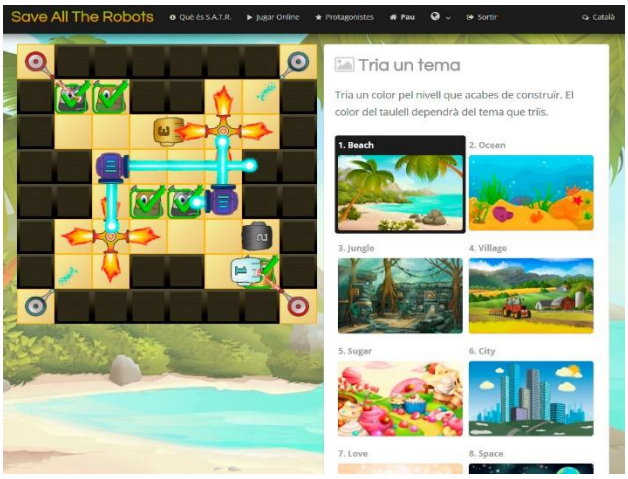


Figura 43. Triant tema

9. Finalment l'usuari tria un tema pel seu nivell, en altres paraules, pot escollir a quin escenari tindrà lloc el seu nivell.

   Després d'això, quan l'usuari ha fet click sobre el seu tema preferit, el nivell es publica i el sistema pregunta a l'usuari si vol compartir el seu nivell a les xarxes

socials i si vol convidar als seus contactes d'usuari a jugar al seu nivell, fomentant d'aquesta manera la participació d'altres usuaris de Save All The Robots i donant a conèixer el joc a través de publicar el seu nivell a les xarxes socials.

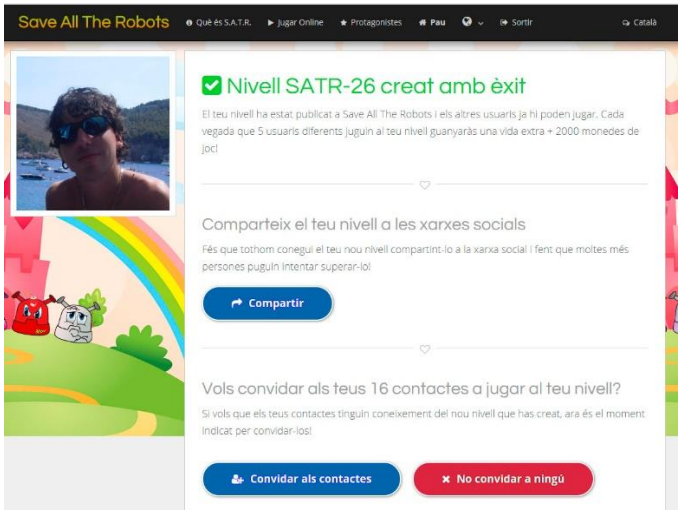


Figura 44. Compartint el nivell

Quan un usuari convida a jugar als seus contactes als seus nivells creats es genera un efecte 'xarxa social' on a més de jugar contra una màquina, s'està jugant contra altres humans. Això dóna la possibilitat de saber que s'està guanyant o perdent contra algú i fomenta l'esperit competitiu del jugador. A més d'això permet al jugador posar en pràctica la seva imaginació i creativitat i construir nivells realment difícils de superar.

D'altra banda la motivació de superar nivells creats per altres usuaris és màxima, ja que els nivells guanyats se situen al perfil de l'usuari augmentant el seu històric de victòries i el seu nombre de reptes superats. A més cada nivell personalitzat ofereix una recompensa al guanyador que es calcula segons la dificultat del nivell.

4.3. Analítiques de joc

Una vegada l'autor ha acabat definitivament de construir el joc i aquest funciona perfectament, cal tenir un pla de seguiment per veure que tot va segons l'esperat. Per exemple, hem de ser capaços de detectar coses com:

- Si d'un nivell a un altre hi ha un increment massa fort de la dificultat i això provoca l'abandonament massiu dels jugadors en aquest nivell.
- Si la dificultat no incrementa de manera correcta i per tant els usuaris s'avorreixen
- Quin percentatge dels jugadors ha vist cada nivell (indica l'abandonament)
- Quina és la dificultat d'un nivell en termes de 1 a 100
- Quantes vegades s'ha guanyat el nivell
- Quantes vegades s'ha perdut
- Percentatge de victòries
- Quantes vegades (en mitja) cal jugar a un nivell per acabar-lo guanyant
- Etc...

4.3.1. Analítiques del mode 'original'

El mode original està composat per 8 mons i 9 nivells per món. Cada món té la seva pròpia gràfica que mesura la corba de la dificultat d'aquell mon i els estadístics individuals de cadascun dels seus nivells. Veges-m'ho:

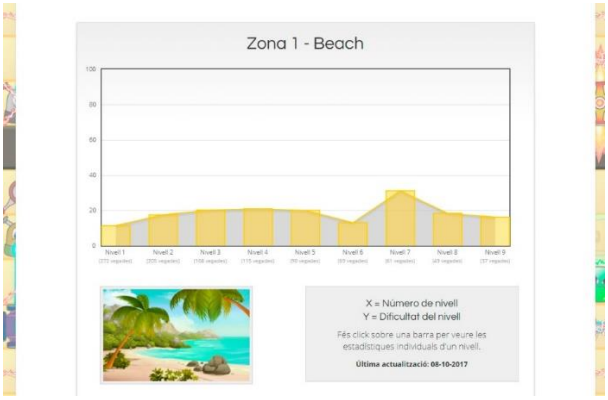


Figura 45. Estadístiques del món 1

A simple vista la gràfica principal ofereix el detall de la corba de la dificultat del món on l'eix de les X és el nombre de nivell i les Y representen la dificultat del nivell.

Si fem clic sobre una barra obtenim les estadístiques individuals del nivell que s'ha clicat. Per exemple fem clic al nivell 1 i obtenim:

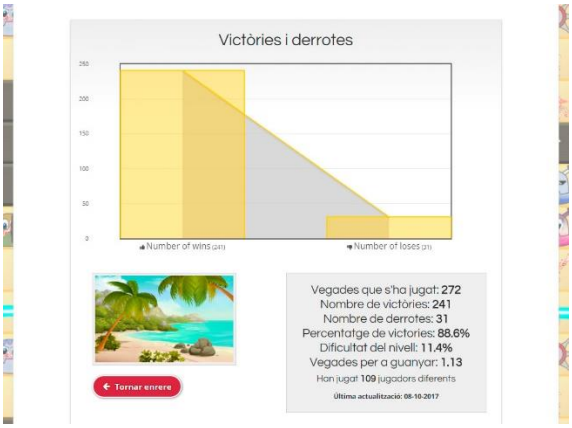


Figura 46. Estadístiques del nivell 1-1

En aquest punt, abans de veure els algorismes matemàtics que calculen les dades, és un bon moment per veure els estadístics del món 8 (l'últim) i veure si la dificultat ha incrementat.

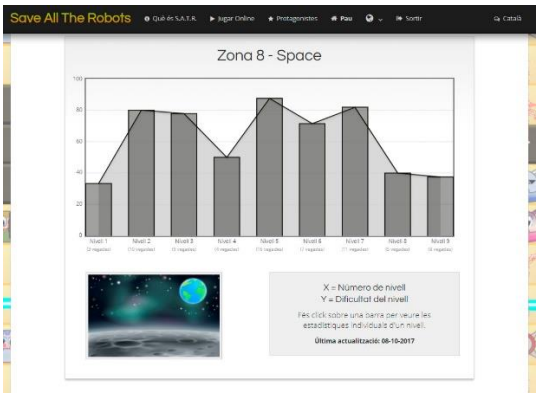


Figura 47. Estadístiques del món 8

Com era d'esperar el resultat és correcte. L'últim món del joc té la dificultat molt més elevada.

En aquest punt i abans de passar als càlculs algorísmics de la dificultat s'insta al lector a explorar per ell mateix els estadístics de joc al següent enllaç:

https://www.savealltherobots.com/ca/stats/original-game

Ara que el lector ha vist en primera persona de forma directa els estadístics resultarà més fàcil que entengui els càlculs que s'expliquen a continuació.

4.3.2. Càlculs matemàtics per extreure els estadístics

- **Vegades que s'ha jugat:** Comptar quantes vegades ha estat jugat el nivell. Per evitar comptar hi ha un registre a la base de dades que ho indica, només cal sol·licitar-lo.
- **Vegades que s'ha perdut:** Són les vegades que els jugadors en general han perdut el nivell.
- **Vegades que s'ha guanyat:** Són les vegades que els jugadors en general han guanyat el nivell.
- **Percentatge de victòries:** Vegades que s'ha guanyat / Vegades que s'ha jugat
- **Dificultat del nivell:** 100 - Percentatge de victòries
- **Vegades per guanyar:** són les vegades que un jugador necessita jugar per a poder superar el nivell:
  (Vegades que s'ha perdut / Vegades que s'ha guanyat) + 1

4.3.3. Analítiques del mode 'arcade'

Els estadístics del mode Arcade funcionen exactament igual que els de la versió original, però en aquest cas com que no hi ha mons, es fan directament els estadístics incrementals sobre els nivells.

Figura 48. Estadístiques Arcade

Les estadístiques Arcade es troben al següent enllaç:

https://www.savealltherobots.com/ca/stats/arcade-game

4.3.4. Gràfiques d'abandonament

Les estadístiques d'abandonament són un aspecte clau de la gestió del joc. Mostren quin percentatge d'usuaris han vist un nivell, en altres paraules, quins són els nivells en què més usuaris deixen de jugar a Save All The Robots.

Com és d'imaginar aquesta gràfica ha de ser de caire descendent, però sense caigudes massa brusques. Una caiguda brusca seria un clar indicatiu que un nivell està fallant per alguna raó:

- Massa difícil: caldria baixar-ne la dificultat
- Error de codificació: un error al codi fa que, per exemple, el joc es pengi en un cert nivell.
- Etc..

Al mode original, la gràfica d'abandonament té el següent aspecte:

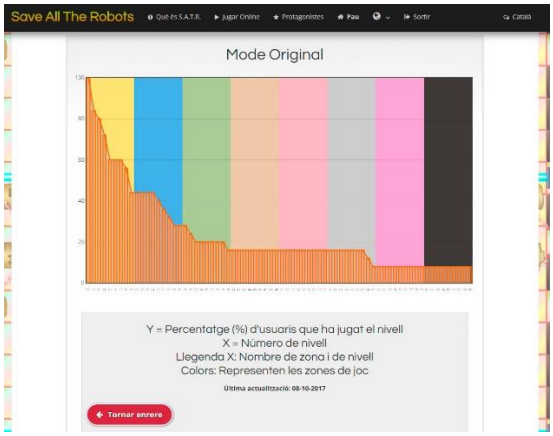


Figura 49. Gràfica d'abandonament 'Mode Original'

Cada barra representa un nivell del joc i cada color de fons representa un món. Es pot apreciar que la forma de la gràfica és tal i com s'esperava, descendent i sense caigudes brusques.

Vegem ara l'aspecte de la gràfica d'abandonament del mode Arcade:

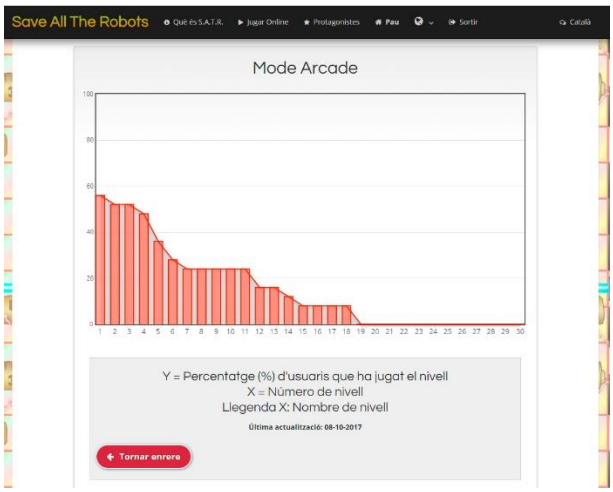


Figura 50. Gràfica d'abandonament 'Mode Arcade'

Les gràfiques d'abandonament del mode original i del mode Arcade es poden trobar al següent enllaç:

https://www.savealltherobots.com/ca/stats/users-levels

CAPÍTOL 5.

PROVES I RESULTATS

5.1. Proves i resultats individuals

Les proves a nivell individual es van anar realitzant a la finalització de cada sprint durant el període de desenvolupament.

No s'ha utilitzat cap software de suport per testejar el codi font, enlloc d'això, l'autor després de cada sprint dedicava unes hores a testar que els resultats fossin els esperats i no es passava al següent sprint fins que tot estava solucionat.

D'aquesta manera es pot dir que tot el que es construïa, reposava sobre uns fonaments robustos i testats a fons per l'autor.

Del temps de desenvolupament del joc, una bona part, almenys un 20% ha anat destinat a testar cada funcionalitat.

Tot i així, com no podia ser d'altra manera, si en un sprint s'ha detectat un error arrossegat involuntàriament d'un sprint anterior, el primer que s'ha fet abans de prosseguir amb el desenvolupament ha estat arreglar tal error.

5.2. Proves col·lectives

Una vegada finalitzat el desenvolupament, s'han fet 3 tipus de proves col·lectives:

1. **Proves a nivell familiar dins la xarxa domèstica**

   L'autor va demanar als seus familiars testar l'aplicació a nivell de jugabilitat, comprensió, errors, etc. De manera que els va demanar jugar a Save All The Robots des de l'inici, sense conèixer res, amb la intenció de tenir primeres opinions del joc.

   El resultat d'això és que l'opinió va ser molt bona d'entrada, però van detectar un problema: Al joc no hi havia manera de treure el volum, de manera que havien de treure tot el volum al PC en cas que no en volguessin.

2. Proves a nivell del grup d'amics

Una vegada superat el test familiar es va proposar als amics més propers fer exactament el mateix per obtenir una visió més ampla abans de publicar el joc definitivament a usuaris reals.

La conclusió va ser la mateixa: joc molt bo i adictiu però aquest grup més gran de persones va trobar altres errors. La versió mòbil no era capaç de suportar una taxa de transferència de dades tant alta si no es disposava d'una connexió potent, per exemple, jugar a Save All The Robots mentre es viatja en tren resultava molt complicat.

En aquest punt l'autor va solucionar aquests dos problemes abans de posar en marxa el joc definitivament.

El problema de l'àudio va ser un tres i no res. Només va caler codificar la possibilitat de poder treure el volum en qualsevol moment del joc.

El tema de la versió mòbil en canvi va ser més complicat. Primer es va reduir la transferència de dades a la versió mòbil 'capant' tot allò que es va poder per jugar amb normalitat però sense extres.

Una vegada fet això, el joc tirava una mica més però si no es disposava de connexió Wifi o 4G com a mínim no funcionava.

La realitat és que totes les aplicacions web necessiten connexió per funcionar i Save All The Robots no és l'excepció. De manera que el problema de la versió mòbil no té més solució ja que no és una aplicació instal·lable directament sobre el sistema operatiu, sinó que és una aplicació que s'executa dins una altra aplicació (navegador) que a la vegada s'executa al sistema operatiu. Hi ha 3 passos i no 2. I necessita internet com totes les webs.

No obstant això, el que s'ha aconseguit amb la versió mòbil és una brutalitat, en el sentit que des del mòbil es pot fer exactament el mateix que des de la versió web, crear nivells, editar-los, jugar Original, jugar Arcade...., tot absolutament.

S'aprofita per instar al lector a utilitzar l'aplicació des del telèfon mòbil en un entorn on la connexió sigui potent, per exemple a casa connectat al wifi, i que comprovi com totes les funcionalitats poden ser aplicades sense problemes.

3. Proves amb usuaris reals

La prova final es va dur a terme publicant el joc definitivament a la xarxa i fer que usuaris reals coneguessin Save All The Robots i comencessin a jugar.

Actualment Save All The Robots es troba en aquesta tercera fase amb uns resultats molt bons però amb poquets usuaris de moment.

CAPÍTOL 6.

CONCLUSIONS I TREBALL FUTUR

6.1. Conclusions

El fet de poder realitzar el Treball de fi de grau sobre Save All The Robots ha estat un fet meravellós. Cal tenir en compte l'estima de l'autor del TFG per un joc en què ell mateix va participar en el passat, per tant només la temàtica ja és un regal.

D'altra banda han estat uns mesos molt durs. És una aplicació d'un tamany molt gran i que contempla molta algorísmica, tant a nivell de lògica com a nivell de xarxa, a més és una aplicació oberta al públic i una fallada pot suposar la pèrdua de milers d'usuaris. A més del TFG, l'autor té ara aquesta pressió afegida.

L'aplicació és tant extensa que per explicar-la sencera caldrien 3 memòries com aquesta, aleshores el que s'ha fet és reduir els conceptes, explicar-los simplificats i finalment deixar d'explicar funcionalitats que s'han cregut secundàries, per exemple, no s'ha parlat dels algorismes que gestionen les sessions, les cookies, les transferències de dades de les cookies a la base de dades, les codificacions de les plataformes de pagament, etc... i molts altres aspectes que no han pogut tenir cabuda però que hi són presents i que han portat molta feina.

S'ha cregut més convenient explicar la part de xarxes i la part d'algorísmica que són les coses que més hem treballat a la carrera d'Enginyeria Informàtica. Tot i així, aquests aspectes han estat simplificats per a poder ser explicats en aquest document.

6.2. Treball futur

El treball futur consta bàsicament de 3 aspectes clau:

- Seguir amb el manteniment de l'aplicació
- Crear noves funcionalitats quan sorgeixi la demanda
- Donar a conèixer Save All The Robots

La conclusió final després d'acabar el Treball de Fi de Grau és la mateixa que l'autor ha extret després d'acabar qualsevol altre projecte i és la següent:

"Programar una aplicació és la part fàcil, és la part que tothom pot fer, uns necessitaran més temps, uns altres potser menys, però tot enginyer ho acabarà aconseguint. Una vegada s'ha assolit l'objectiu principal de tenir l'aplicació funcionant, aleshores ve l'autèntica guerra, que és aconseguir que els usuaris la utilitzin. I aleshores arriba el cop de realitat, l'etapa de frustració i una lluita realment llarga i complicada..."

ANNEX 1.

SISTEMA DE NOTIFICACIONS ASÍNCRONES

S'ha triat explicar aquest apartat en un annex degut a que el sistema de notificacions asíncron té a veure més amb temes de xarxa que no pas amb el joc en si. D'aquesta manera, al no introduir aquest apartat abans s'ha facilitat al lector la comprensió global del joc, que ja per si sola és prou enrevessada.

Sovint, el lector, si té perfils a les xarxes socials s'haurà adonat que quan un altre usuari interacciona amb el seu perfil, apareix en temps real una notificació que té per objectiu informar de la interacció iniciada per l'altre, per exemple: comentaris en un post, clic al botó 'm'agrada', etc... Això fa més real l'aplicació i més fluida, 'enganxa' a l'usuari encara més a l'aplicació. **Save All The Robots** també ho incorpora i a continuació se'n desgranen tots els detalls.

Tipus de notificacions

1. Un usuari afegeix a un altre usuari als seus contactes.
2. Un usuari aconsegueix 1 vida extra i 2000 monedes quan 5 jugadors diferents juguen a un dels seus nivells creats.
3. Un jugador aconsegueix el rècord absolut a la versió 'Arcade'.
4. Un jugador que tenia el rècord absolut a la versió 'Arcade' el perd essent superat per un altre jugador.
5. Un jugador convida altres jugadors a jugar al seu nivell creat.
6. Agraïments per part del sistema a un usuari que fa una donació al projecte.

A la següent figura es pot veure com canvia la barra de menús del joc apareixent un indicador de 'notificació pendent'

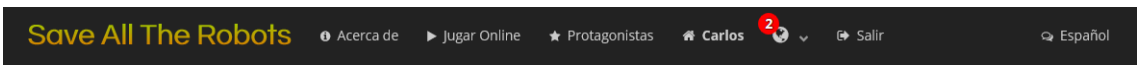


Figura 51. Notificacions pendents

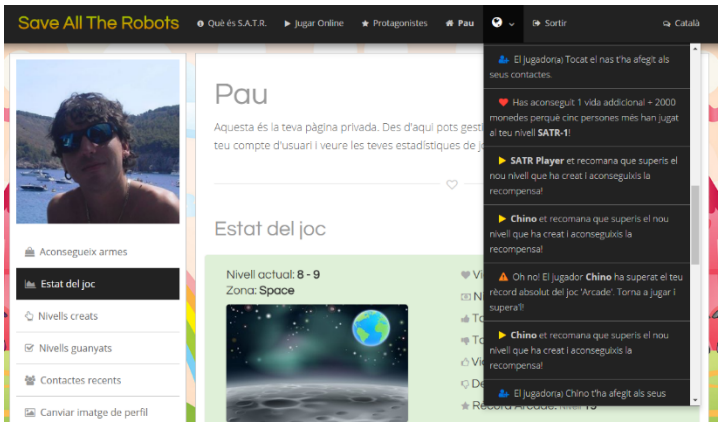


Figura 52. Panell de notificacions complert

En aquesta darrera figura es pot apreciar com el jugador rep les notificacions de cada tipus, diferenciades per un símbol particular i un text descriptiu. Cada notificació conté un enllaç cap a la pantalla del joc relacionada, per exemple, les notificacions de l'Icona groga porten al nivell al qual ens estan convidant a jugar, la notificació de l'Icona blava porta al perfil del jugador que ens ha afegit als seus contactes, etc.

Cal tenir en compte que, a grans trets, *Save All The Robots* a més de ser un joc fa també de xarxa social. No és difícil arribar a aquesta conclusió ja que si es navega amb la sessió d'usuari iniciada, es troben aspectes propis d'una xarxa social, com poden ser: perfils d'usuari, imatge de perfil, contactes d'usuari, interaccions entre usuaris, nivells creats per un usuari que són públics a altres usuaris, i un llarg etcètera.

Implementació de les notificacions

Per construir el sistema de notificacions s'ha hagut de triar entre dues possibles implementacions diferents: 'Long Polling' i 'Passive Piggyback', es resumeixen a continuació:

- Long Polling: El client se subscriu a un canal de connexió al servidor mitjançant un request. El servidor manté el request i espera a que tingui lloc un esdeveniment. Quan l'esdeveniment succeeix la resposta d'un missatge totalment complert es retornada al client. A continuació el client envia

immediatament una nova petició. Per tant, el servidor disposa d'un canal de comunicació amb el client a cada moment.

- Passive Piggyback: Quan el servidor ha d'enviar una actualització d'alguna informació, espera fins la propera vegada que el navegador faci un request i li envia l'actualització juntament amb el response que el navegador estava esperant.

En poques paraules, si s'utilitzés el segon mètode les notificacions no esdevindrien en temps real, per aquest motiu l'autor ha implementat les notificacions fent ús de la tècnica del LongPolling, i per tant, els missatges entre usuaris esdevenen en temps real.

Implementació de la xarxa i algorismes

El front-end és l'encarregat d'iniciar la interacció subscrivint-se al canal de comunicació proposat pel back-end. En primer lloc el *front* comprova si hi ha sessió d'usuari iniciada, i si és així, crea un interval d'execució temporal asíncrona, és a dir, una crida a una funció que s'anirà repetint cada cert temps en segon pla, en el cas que ens ocupa, es crida cada 10 segons.

Les crides, que es fan mitjançant *Ajax* (anteriorment explicat) a la banda del *front*, invoquen un mètode remot al *back*. Aquest mètode remot consulta la taula de notificacions de la base de dades per saber si existeixen notificacions pendents de llegir que el seu destinatari sigui l'ID de l'usuari de la sessió. Si hi ha notificacions aquestes es filtren pe tipus per tal d'afegir-hi el text, el símbol i l'enllaç adient a cada cas. Un cop s'han obtingut totes les notificacions, es col·loquen totes en un array i es retornen al *front* enviant-les per la xarxa en format de text JSON. Quan l'array de notificacions transformat en text arriba al *front,* aquest el transforma en un objecte propi de JavaScript i desgrana les notificacions a la vegada que els dóna format html i css per ser degudament mostrades per pantalla.

Per altra banda un altre usuari ha d'haver creat les notificacions per tal que els algorismes anteriors puguin funcionar. Així doncs quan un usuari, per exemple, afegeix a un altre usuari als seus contactes o bé convida altres usuaris a jugar un dels seus nivells

creats, etc.., s'insereix una fila a la taula de notificacions de la base de dades amb l'usuari origen, l'usuari destí i el tipus de notificació corresponent, que serà llegida la propera vegada que l'usuari destí es connecti o bé immediatament (temps real) si l'usuari destí està connectat.

Nota: L'explicació descriu les notificacions de manera resumida, la realitat, tot i que segueix al peu de la lletra aquests estàndards, és bastant més complexa.

ANNEX 2.

DIFERÈNCIES APP D'ANDROID I APLICACIÓ WEB DEL TFG

Diferència de localitat

L'aplicació per Android portada a terme a durant l'assignatura *Projecte Integrat de Software* és de caire local. Un sol jugador interactua amb l'aplicació de forma local on els fitxers executables es troben instal·lats al seu dispositiu mòbil.

L'aplicació web, en canvi, és de caire distribuït. Molts jugadors alhora interactuen amb l'aplicació i executen els mateixos fitxers, ja que aquests estan instal·lats en un servidor que suporta múltiples connexions simultànies. A més els canvis que un jugador realitza al programa, els reben, en temps real, tots els altres jugadors, com per exemple les puntuacions o les notificacions, ja que a més de ser una aplicació distribuïda, és un programa multi fil que manté constantment les dades actualitzades en temps real.

Diferències funcionals

L'aplicació inicial d'Android disposa, només, d'una modalitat de joc, concretament la versió 'Arcade' de nivells infinits.

L'aplicació web (TFG), en canvi, disposa de 3 modalitats de joc: 'Original', 'Arcade' i 'Nivells Personalitzats', aquesta última amb funcionalitat doble (jugar i construir).

Diferències de protagonistes

L'aplicació inicial per Android disposa de menys robots i menys enemics. Com que el temps d'una assignatura és inferior al temps d'un TFG, es va crear el joc sense complicar-se més de compte, amb els robots i enemics justos per fer l'app jugable.

Així doncs l'aplicació per Android no conté ni el Robot Thug ni el Robot Girl.

Amb els enemics, però, hi ha més diferències. L'aplicació inicial només conté el puny simple, el raig làser, el llença-flames i l'elèctric més senzill de tots, mentre que la versió web ha gairebé triplicat els enemics.

Diferències de complexitat

Del punt immediatament anterior es pot veure fàcilment que l'aplicació web del TFG és d'una complexitat molt més elevada, ja que cada vegada que s'introdueix un robot o enemic nou, incrementen de forma exponencial els casos a tenir en compte alhora de dissenyar i implementar els algorismes del joc. Així i tot, com s'ha vist en capítols anteriors, amb una mica d'enginy s'ha aconseguit resoldre totes les parts del joc amb complexitat lineal, amb una 'n' igual al nombre de cel·les de la matriu de joc.

Diferències algorísmiques

Algorismes de resolució

A l'aplicació d'Android els algorismes de decisió tenen lloc al final, quan acaba un nivell es fa una crida a l'algorisme que decidirà si el nivell s'ha superat o no, i per tant es passa al nivell següent.

A l'aplicació del TFG, com s'ha explicat en capítols anteriors, s'opta per un diàleg usuari-sistema que té lloc a cada jugada i que va construint la solució per ell mateix a cada pas, combinat amb una destrucció i re-construcció de la matriu de joc a la jugada final.

Les dues maneres són igual de vàlides sempre i quan funcionin correctament, tot i que l'autor opta per la segona basant-se en tots els avantatges explicats en capítols anteriors.

Algorismes de generació de nivells

L'altra gran diferència algorísmica rau en el generador de nivells infinits de la versió 'Arcade' del joc.

Mentre que a l'aplicació per Android es va optar per un generador completament algorísmic que construïa nivells aleatoris, l'aplicació del TFG s'ha construït un generador pseudo-aleatori aprenent dels errors del passat per tal de millorar el present. La gràcia del generador aleatori era que construïa nivells sempre diferents, ara bé, tenia dos grans inconvenients que han estat solucionats amb el generador pseudo-aleatori, que són:

1. Temps de generació: sovint quan es jugava en nivells avançats on intervenien tots els actors del joc, el jugador patia llargues esperes fins que l'algorisme construïa el nivell, degut a l'alta complexitat dels càlculs (era un algorisme de complexitat exponencial que buscava totes les combinacions possibles fins que en trobava una de bona). Quan això passava, s'optava per llençar un nivell *per defecte* i aturar l'algorisme. Sovint el jugador veia aparèixer masses vegades aquest 'nivell per defecte'.
2. Qualitat del nivell: sovint el generador construïa nivells vàlids però que no tenien gaire sentit visual, és a dir, no es podia dir coses com ara: -*'Vaig a construir un nivell cabró on als usuaris els costi trobar la solució',* o bé, -*'Vull que aquest nivell sigui completament simètric'..* etc... No es tenia cap control sobre els nivells.
   I és que el 'bon gust' humà és una de les coses que no pot simular un algorisme.

La conclusió final de les diferències algorísmiques és que cada manera de fer aporta les seves 'cosetes' bones i que cada autor té unes preferències. L'autor del TFG ha programant l'aplicació tenint en compte la complexitat i les sensacions del jugador.

Altres diferències

Armes

La versió del TFG té armes com a nous protagonistes amb tota l'algorísmica i disseny que això implica. Concretament són 5 armes amb tota la seva funcionalitat.

Botiga online

La nova aplicació disposa d'una botiga online on es poden adquirir armes i vides fent ús de les monedes de joc o bé directament amb diners reals utilitzant la plataforma de pagament PayPal i havent deixat el codi apunt per afegir un TPV Virtual en cas que la demanda sigui elevada.

Diferències Visuals

Es pot apreciar a través de les diferents captures del joc, que nova versió també ha millorat molt els dissenys del joc i també ha tractat amb més cura els aspectes visuals, fent més atractiva la seva aparença.

Captures de la versió Android (PIS)

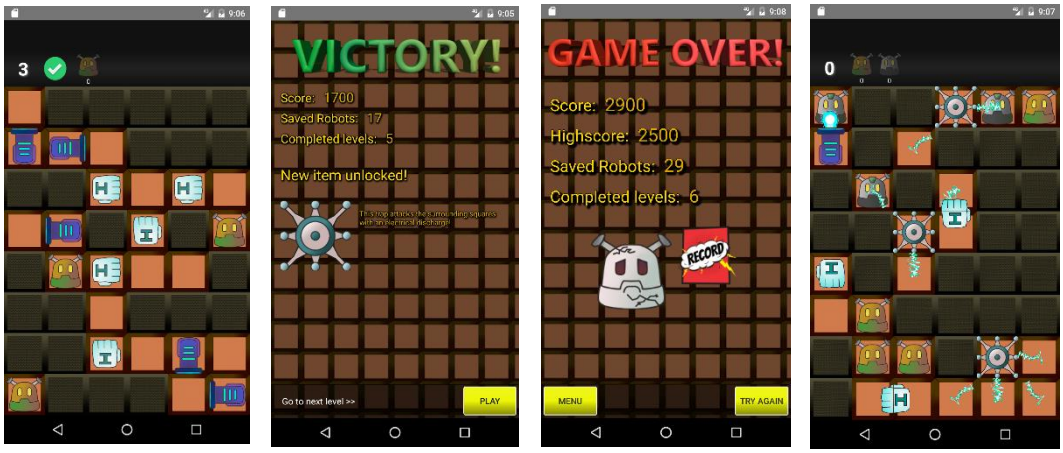


Figura 53. Captures de la versió anterior

Captures de la nova versió

Figura 54. Captures de la versió nova

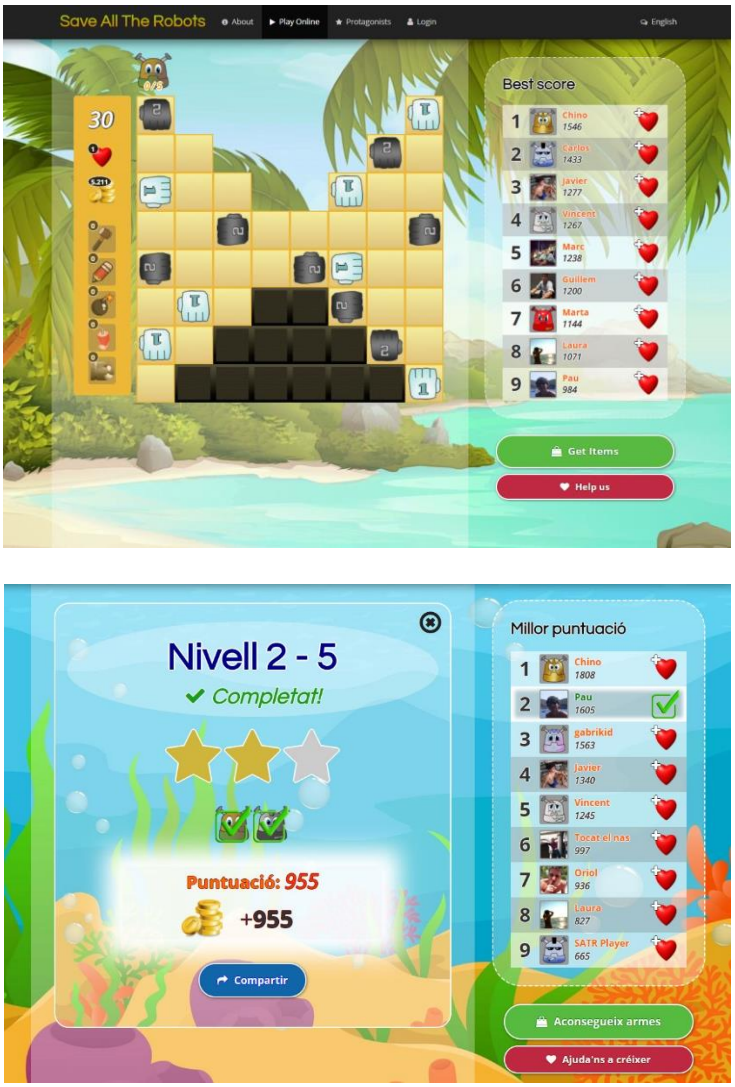


Figura 55. Més captures de la versió nova

Protagonistes de la versió antiga Vs Protagonistes de la versió nova

Es pot apreciar com els dissenys dels protagonistes de l'antiga versió del joc han millorat també en la nova versió:

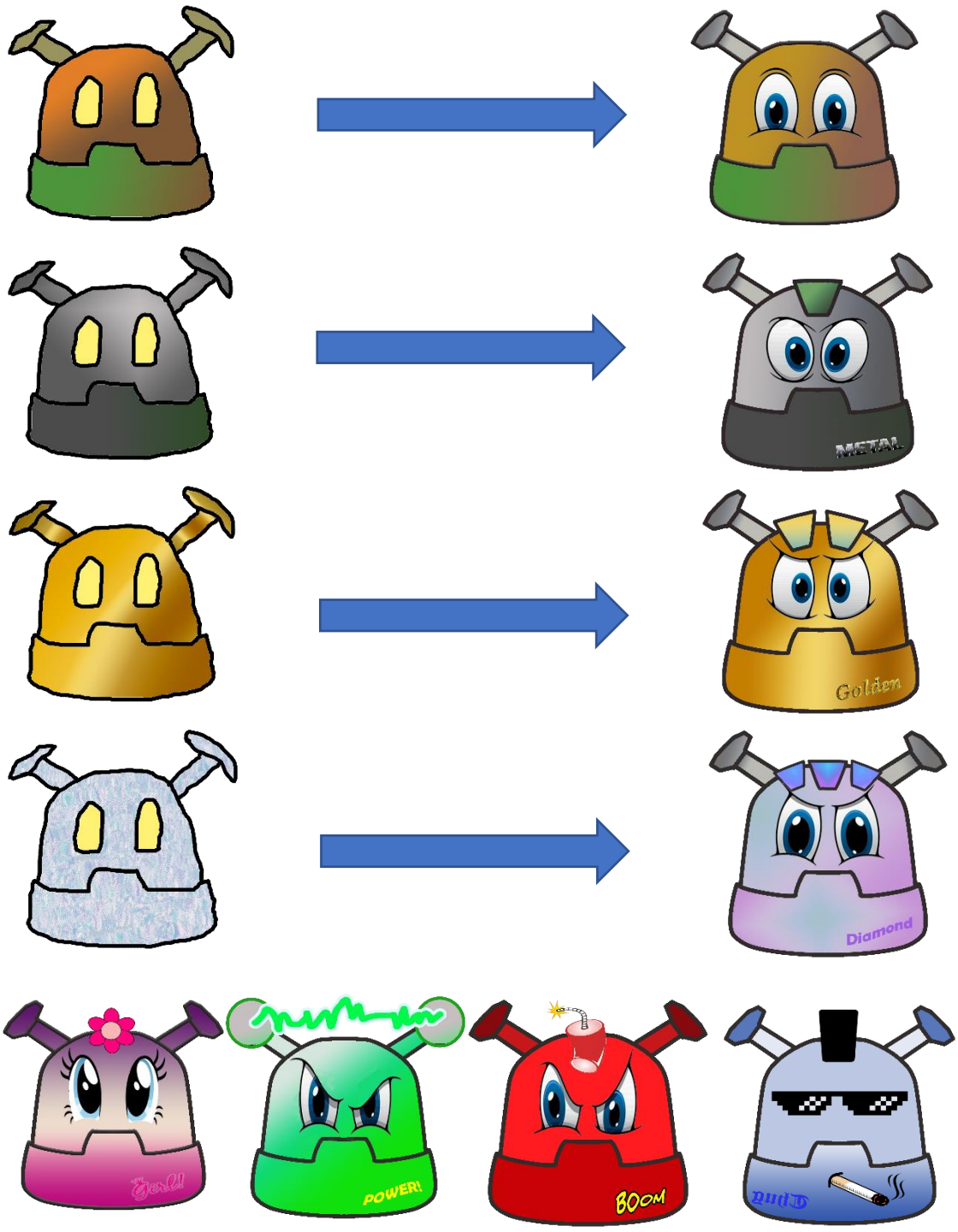


Figura 56. Millora dels robots

Bibliografia

Bibliografia

Durant el desenvolupament de Save All The Robots l'autor ha fet ús de les següents guies i manuals:

Api de PHP: http://www.php.net/

Guia de JavaScript: https://developer.mozilla.org/es/docs/Web/JavaScript/Guide

Api de jQuery: https://api.jquery.com/

Api de jQueryUI: https://api.jqueryui.com/

Manual de jQueryUI: https://jqueryui.com/

Manual de CSS: https://www.w3schools.com/css/

Manual html: https://www.w3schools.com/html/

Manual de PEAR: https://pear.php.net/manual/en/

Api de MySQL: https://dev.mysql.com/doc/refman/5.7/en/

Guia de Adobe Photoshop: https://helpx.adobe.com/es/photoshop/user-guide.html