Universitat de Girona
Escola Politècnica Superior

Treball final de grau

Estudi: Grau en Disseny i Desenvolupament de Videojocs

Títol: *Metabolikka*, creació d'un *Serious Game* orientat a la Bioquímica

Document: Memòria

Alumne: Lluís Torres Procas

Tutor: Gustavo Patow i Sílvia Barrabés
Departament: IMAE / Biologia
Àrea: LSI / Bioquímica i Biologia Molecular

Convocatòria (mes/any): Setembre/2022

Agraïments

A Gustavo Patow, tutor del treball, qui en el seu moment va confiar en mi per convidar-me a participar en aquest projecte encarregat pel departament de Biologia. També per sempre estar present quan l'he necessitat. I per últim, agrair-li la paciència infinita tinguda en mi, sempre conservant la fe en que acabaria el projecte, que ha acabat per donar-me la confiança necessària per tirar endavant aquest projecte i saber valorar la feina feta.

A la Dra. Sílvia Barrabés, tutora del projecte, i al Dr. Marc Ribó, qui dins del marc de projecte d'innovació docent, van tenir la idea d'aquest projecte, van decidir muntar un equip per treballar-hi i van confiar en mi per dur-lo a terme, sempre oberts a respondre tots els dubtes que tenia en la matèria i aportants molt bones idees per al disseny del joc.

ÍNDEX

1. Introducció

L'ensenyament i educació de les persones és una de les parts més importants (si no la que més) d'una societat. D'aquest ensenyament dependrà el futur desenvolupament d'aquesta societat i la correcta convivència entre els seus membres. Una societat que no inverteixi en educació i/o no faciliti el seu accés a tothom està destinada al fracàs. A causa de la seva importància però, molta gent es reticent a fer canvis en el sistema, aplicant la coneguda frase, però incorrecta en molts casos, de "no cal canviar el que ja funciona", limitant així l'avenç i la millora. Un d'aquests canvis del que no se n'aprofita tot el seu potencial és la inclusió dels videojocs com a eina educativa.

Si preguntem al públic en general la funció d'un videojoc, pràcticament tothom es limitarà al entreteniment que aquest proporcionen. No obstant, ja s'ha comprovat que els videojocs poden tenir moltes més funcionalitats, i una d'elles és la educativa, amb els anomenats *Serious Games*. Dins dels diferents àmbits d'estudi hi ha molts conceptes, sobretot aquells molt complexos, on l'estudiant no pot fer res més que intentar memoritzar-lo per després oblidar-lo al acabar l'avaluació. Però això no funciona per a molts estudiants, els quals han de fer molts esforços per poder concentrar-se el necessari, a part que per la utilitat si al poc temps allò memoritzat s'acaba oblidant. En aquest cas, els *Serious Games* agafen aquests conceptes i els expliquen utilitzant mecàniques de videojocs, fent que el mètode d'aprenentatge sigui més entretingut i divertit, augment la motivació de l'estudiant i facilitant així l'assimilació del concepte en qüestió. A més, l'aplicació d'aquests tipus de videojocs a l'ensenyament a dia d'avui és molt fàcil, ja que pràcticament tothom té a l'abast algun dispositiu multimèdia (*smartphone*, *tablets*, *pc*, ...).

Amb aquesta voluntat de facilitar l'ensenyament de conceptes complexos per part del departament de Biologia de la Universitat de Girona, en aquest projecte crearem un *Serious Games* capaç d'explicar d'una forma amena i divertida alguns conceptes relacionats amb la respiració cel·lular, en concret la Glicòlisi, el cicle de Krebs i la cadena transportadora d'electrons.

1.1. Motivacions

Aquest projecte es va iniciar quan se'm va oferir ser qui s'encarregui de crear i desenvolupar un videojoc pel departament de Biologia que pugui explicar la respiració cel·lular a estudiants universitaris. Això em va brindar l'oportunitat de posar en pràctica les habilitats adquirides durant el grau en tots els àmbits (disseny, programació, narrativa i art), ja que era un projecte en solitari, així com haver de solucionar pel meu compte problemes que puguin anar sorgint. Aquest projecte em va permetre treballar tant en els meus punts forts, com en aquells on potser no hi tinc tanta pràctica (com en el meu cas pot ser l'apartat artístic).

D'altra banda, també és un disseny de videojoc que s'ha d'ajustar als requeriments que demana "el client", en aquest cas el departament de Biologia (d'ara en endavant l'ús de la paraula client farà referència a aquest departament), afegint així un punt de dificultat. A part, el fet de tenir un client darrere el projecte dóna el punt de pressió de saber que hi ha unes expectatives amb el videojoc resultat.

1.2. Propòsit

Aquest projecte vol crear un primer prototip que permeti tenir una idea clara de com han de ser tots els apartats del joc. Al tractar-se d'un *Serious Game,* també volem que aquest prototip permeti valorar si els conceptes a explicar són clars i entenedors, ja que aquesta és la principal finalitat d'aquest joc.

1.3. Objectius

El propòsit del projecte és crear un prototip de Serious Game el suficientment desenvolupat per poder jugar-hi i valorar tots els seus apartats. Per arribar aquí tenim els següents objectius:

- Entendre els conceptes a explicar: d'inici hem d'entendre com funciona la respiració cel·lular, en concret els processos de Glicòlisi, cicle de Krebs i cadena transportadora de electrons, per tal de poder crear mecàniques que permetin l'assimilació d'aquests conceptes.

- Dissenyar el joc: crear el disseny conceptual del videojoc, incloent tots els apartats, però remarcant la importància de la jugabilitat. Les mecàniques a utilitzar han de ser divertides i estimulants, però sempre buscant que els conceptes a explicar quedin clars. A part, s'ha d'adaptar el disseny del videojoc als requeriments donats pel departament de Biologia.

- Implementació del joc: busquem crear un prototip que ja permeti tenir clara una visió general del joc. Aquest prototip ha de permetre valorar si els qui el juguin poden arribar a entendre els conceptes a explicar mentre que a la vegada el troben divertit i estimulant.

- Redacció TFG: redactar l'evolució i creació del projecte, passant pels diferents apartats i mostrant els coneixements aplicats durant tot el projecte. Aquesta tasca es realitza al llarg de tot el projecte.

1.4. Quadre d'autovaloració

Al tractar-se d'un videojoc complet, tots els apartats tenen el pes, tot i que aquests no es distribueixen per igual. En la següent Taula (1) podem veure una representació aproximada en percentatges del pes de cada apartat en relació a les hores treballades.

Estètica	25 %
Narrativa	5 %
Mecàniques	50 %
Tecnologia	20 %

Taula 1: Quadre d'autoavaluació

En aquest projecte les Mecàniques tenen un pes molt important. Ja des de el nivell de disseny requereixen d'un esforç afegit al buscar aquest equilibri entre diversió i educació a la vegada. Les mecàniques han de ser entretingudes, han de buscar que el jugador vulgui seguir jugant nivells, però a la vegada han de ser una representació del que volem explicar. Si s'allunyen molt del concepte, aquest pot no quedar clar. Si són massa semblants, el joc pot acabar sent avorrit. Es per això que aquest apartat requereix de més atenció i seguiment durant tot el projecte.

Tot i això, al ser una sola persona qui s'encarrega de tot el projecte, tots els apartats tenen un pes, i el segon en importància és la Estètica. Per aquest projecte, la estètica també és molt important ja que com a requeriment del departament de Biologia es vol que aquesta sigui una representació lo més propera a la realitat, no tant a nivell de qualitat gràfica sinó a nivell conceptual.

La tecnologia te certa rellevància en el punt que s'ha d'entendre com funcionen les eines a utilitzar per saber adaptar-les a les nostres necessitats. Hem de saber valorar quines eines tenim al nostre abast, i com aquestes condicionen el futur desenvolupament del projecte, per tal de triar les que ens aportin més facilitats. Això implica decidir per a quina plataforma volem fer el videojoc, quins motors de videojocs treballen sobre aquesta plataforma, etc.

Per últim, la Narrativa a implementar en canvi és mínima, i només busca cohesionar els diferents nivells. Potser en futures evolucions del prototip es podria buscar una narrativa que fos més rellevant.

2. Estudi de viabilitat

Com en tot projecte d'aquestes característiques es necessita un estudi previ de viabilitat. Hem de considerar diferents aspectes, tant tècnics com econòmics, per saber si podrem enllestir el projecte amb els recursos que tenim. És necessari valorar també si aquest projecte ens donarà rendiment un cop finalitzat, i per això hem d'entendre el mercat al que ens volem dirigir. Sense aquest passos previs, podem estar orientats al fracàs i no adonar-nos-en fins que sigui massa tard.

2.1. Recursos tècnics

El desenvolupament d'aquest projecte no requereix una gran quantitat de recursos tècnics, i de fet s'ha realitzat tot utilitzant un sol ordinador i un conjunt de programes gratuïts. La llista dels recursos utilitzats és la següent:

Eines físiques:

- Ordinador portàtil amb Windows 10 de 64 bits utilitzat com a eina principal per al desenvolupament del joc. Característiques: processador Intel® Core™ i7-8750H , targeta gràfica NVIDIA GeForce GTX 1060, 16 Gb de memòria RAM.

- Telèfon mòbil on s'han realitzat totes les probes durant el transcurs del projecte. Característiques: OnePlus 6t amb un Qualcomm® Snapdragon™ 845, GPU Adreno 630, i 8 Gb de memòria RAM.

Programari:

- Unity: és un motor de jocs multi plataforma creat per Unity Technologies. Està disponible per a Windows, Mac i Linux, i té suport de compilació per a més de 20 plataformes, entre les quals es troben PC, les consoles principals, dispositius mòbils i WebGL. Es basa en Mono, un entorn de desenvolupament integrat lliure i gratuït, dissenyat primordialment per al llenguatge C#. Compta amb eines i components per a crear jocs 3D i 2D. És totalment gratuït si no es supera un cert llindar d'ingressos, que si se supera, el desenvolupador ha d'adquirir una llicència pro (de pagament).

- InkScape: software gratuït de dibuix vectorial utilitzat per crear els elements visuals. És l'opció gratuïta del programa de pagament Adobe Illustrator.

- Visual Studio 2019: Com a editor de codi principal per a escriure el codi del joc hem utilitzat l'entorn Visual Studio 2019, un editor de codi molt potent que ens permet debugar els projectes de Unity en temps real (editor predeterminat en Unity).

Tot el software i recursos utilitzats són de llicència gratuïta, per tant no ens ha suposat cap inversió en l'àmbit tecnològic.

2.2. Recursos Humans

Tot projecte d'aquesta envergadura necessitaria d'un equip multidisciplinari al darrera. Tot i que en el nostre cas l'ha realitzat una sola persona, els recursos necessaris per aquest projecte, com a mínim, serien els següents:

- Dissenyador principal i director del joc: és l'encarregat del disseny i redacció del disseny conceptual del videojoc. És el responsable de decidir i organitzar els diferents nivells, així com de les diferents mecàniques a implementar. També ha de saber comunicar aquests requeriments tant a programadors com artistes per que puguin fer les seves tasques amb precisió i cohesió dins de l'equip.

- Programador principal: responsable d'implementar les mecàniques del joc així com de totes les eines necessàries per al desenvolupament. Ha de decidir quins mètodes i funcionalitats són els més adients per a les necessitats del joc, i s'ha d'assegurar de la correcta compilació i rendiment de tot el conjunt.

- Artista principal: encarregat de la creació dels *assets* visuals del joc, tant dels nivells i entorn com de la interfície d'usuari. Ha de mantenir un disseny similar per a tot el projecte i assegurar-se de que el resultat és atractiu per als jugadors.

Aquests tres apartats serien els imprescindibles per a la creació del joc. A partit d'aquí i depenent de la quantitat de feina a realitzar, podríem afegir un segon programador o artista, una persona encarregada de l'apartat sonor, etc.

2.3. Cost econòmic

Tot i que per realitzar aquest projecte s'han utilitzat recursos propis, aquests s'han de tenir en compte a l'hora de valorar el cost econòmic. Tot i això, aquests recursos no són res de l'altre món, ja que els requeriments necessaris tant per utilitzar el programari empleat com per testejar el rendiment del joc no són excessivament

elevats. Com a tal, i suposant un equip de tres persones, una aproximació dels costos podria ser la següent (Taula 2):

RECURS	UNITATS	COST/UNITAT	TOTAL
Ordinador gamma mitja	3	700 €	2.100 €
Smartphone	3	150 €	450 €
TOTAL	**6**	**-**	**2.550 €**

Taula 2: Taula de possibles costos tècnics

Pel que fa als costos del programari, tots els empleats en aquest projecte són gratuïts o de codi lliure, pel que no suposen cap cost indiferentment de la quantitat de membres que formin equip.

Per últim hem de tenir en compte els recursos humans. En el nostre cas simulem tenir un equip multidisciplinari de 3 persones. En equips tant petits els membres solen fer tasques diverses i no només dins del seu àmbit de treball, pel que és difícil calcular les hores que cadascú invertirà en un projecte similar. Tot i això hem fet una aproximació suposant que el projecte té 1 any de duració on tots 3 membres han treballat jornada completa durant tots els dies laborables (Taula 3).

TREBALLADOR	COST/HORA	TOTAL (1920 hores)
Director i dissenyador del joc	16 €/hora	30.720 €
Programador i testes	13 €/hora	24.960 €
Artista	13 €/hora	24.960 €
TOTAL	**42 €/hora**	**80.640 €**

Taula 3: Taula de possibles costos humans

Com s'ha dit, aquests costos són només possibles supòsits, ja que aquest projecte l'ha realitzat una sola persona.

2.4. Estudi de mercat

Previ a la realització del projecte, hem fet un estudi de mercat per tal de veure si la nostra idea ja s'ha realitzat abans. Hem de valorar si ja existeixen jocs similars, i fins a quin punts aquests són competència nostra. A part,

l'estudi ens serveix per veure quin resultat han obtingut els nostres competidors, i valorar quins han estat els seus punts forts i febles per tal d'agafar idees sobre com hem de desenvolupar el joc.

Tot i això, el nostre joc és un *Serious Game* amb una finalitat molt concreta, pel que és difícil que algun joc ja hagi cobert aquest apartat. Si és així i cap joc cobreix les necessitats d'aquest projecte, l'estudi de mercat ens serveix més que res per valorar idees i alternatives jugables, ja que no hi haurà competidors com a tal.

2.4.1. Cerques realitzades

El nostre projecte vol crear un joc 2D compatible amb plataformes Android que expliqui processos bioquímics, i és en aquesta idea on hem basat les nostres cerques. Les cerques contenien paraules clau com "*Biochemistry", "Serious game", "Mobile game"* o *"2D"*. Aquestes cerques les hem començat des d'una cerca molt concreta, buscant directament jocs que realitzessin el que nosaltres volem pel nostre projecte, a molt general, veient altres jocs 2D per mòbils de temàtiques diverses.

Per realitzar les cerques s'ha utilitzat els cercadors de *Google*, *Google Scholar, seriousgames-portal.org* i *sciencegamecenter.org*, fent combinacions de les paraules clau. De totes les cerques realitzades hem seleccionat alguns jocs que ens poden interessar, ja que cobreixen algun dels punts que necessitem.

2.4.2. Resultats obtinguts

2.4.2.1. *Immune Defense*

Joc seriós 2D d'estratègia que tracta temes relacionats amb la immunologia molecular. Els jugadors utilitzen glòbuls blancs per lluitar contra patògens, utilitzant proteïnes de superfície i receptors. Recomanat per majors de 10 anys. Disponible per Windows.

Figura 1: imatge in-game de Immune Defense

2.4.2.2. *Immune Attack*

Joc seriós 3D d'acció en tercera persona que et permet controlar una nau mentre s'exploren les diferents parts del cos humà a nivell molecular i es solucionen problemes activant proteïnes i lluitant contra patògens. Orientat a estudiants de secundaria i universitat. Disponible per Windows.

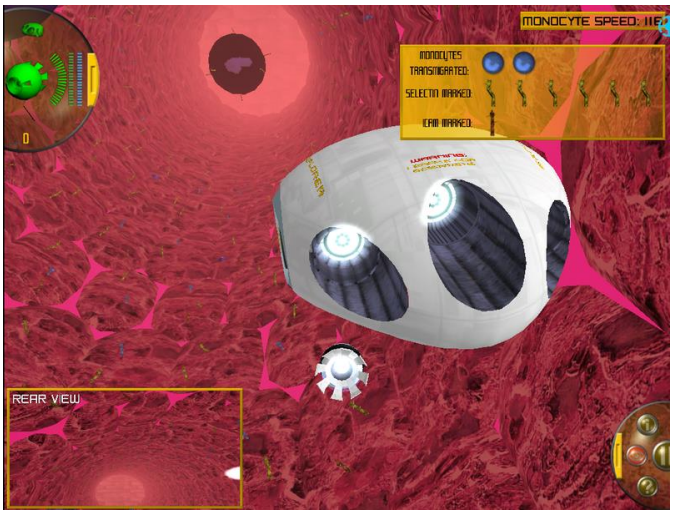


Figura 2: imatge in-game de Immune Attack

2.4.2.3. *Fold it*

Joc seriós de puzles en 3D que mitjançant representacions de les proteïnes permet moure-les per buscar quina és la forma més eficient de plegar-les. Aquests plecs són importants ja que algunes malalties importants es deuen al plec defectuós d'alguns aminoàcids. Disponible per Windows.

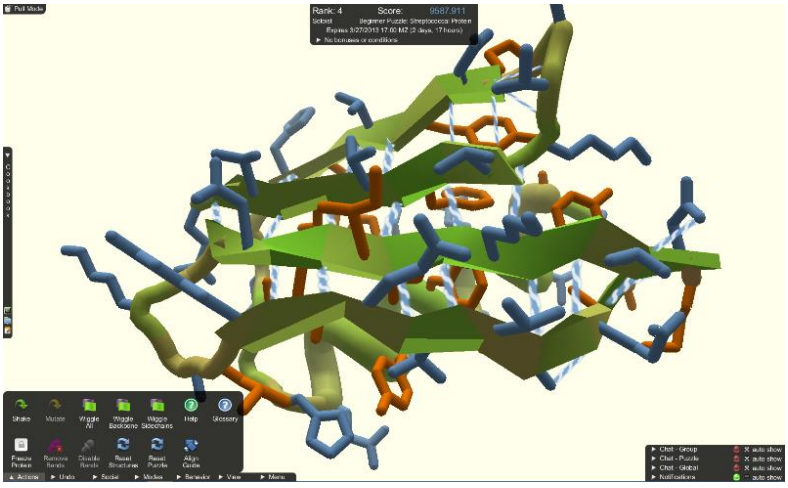


Figura 3: imatge in-game de Fold it

2.4.2.4. *Re-Mission / Re-Mission 2*

Joc seriós 3D d'acció en tercera persona que permet lluitar contra cèl·lules cancerígenes. Orientat a pacients joves amb càncer perquè puguin entendre el desenvolupament d'aquestes malalties. La segona part del videojoc va aplicar estudis que demostraven com certes accions del primer joc comportaven una millora en la motivació i l'estat psíquic dels pacients per tal de reforçar-les. Disponible per telèfons Android.

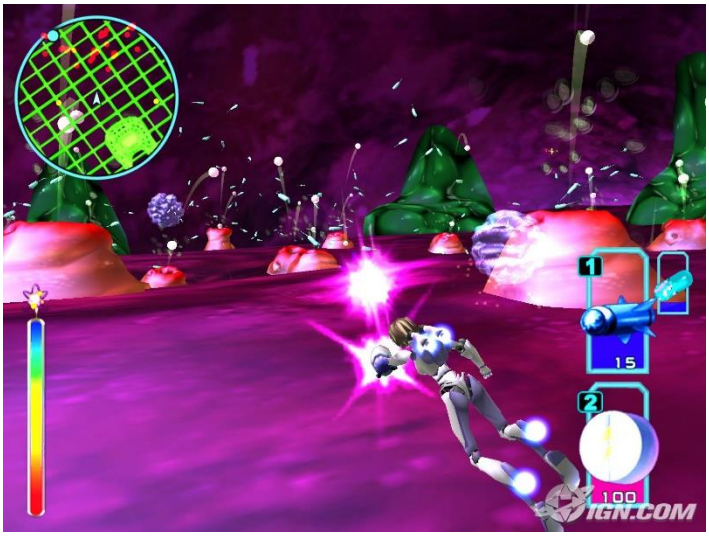


Figura 4: imatge in-game de Re-Mission

2.4.2.5. *CRISPR-VR*

Simulador virtual que permet utilitzar tecnologia VR per tal de visualitzar com és l'interior d'una cèl·lula i els seus teixits, per veure com funciona. Basant-se en estudis i simulacions reals, aquest joc busca crear representacions el més acurades a la realitat possible. Disponible per Windows i sistemes de VR.

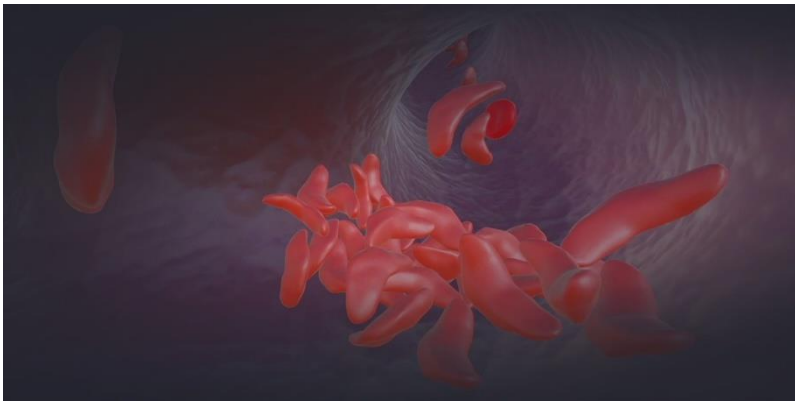


Figura 5: imatge in-game de CRISPR

2.4.2.6. *Build-A-Cell*

Simulador en 2D que et permet construir una cèl·lula i tots els elements que la componen. Es bassa en escollir els orgànuls que volem i veure les característiques de la cèl·lula un cop esta creada. Disponible per Windows.

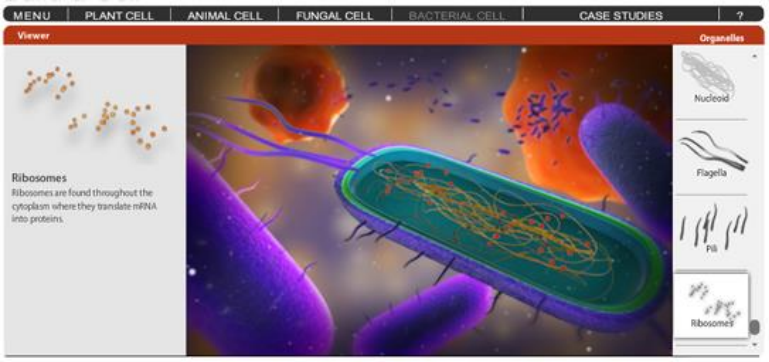


Figura 6: imatge in-game de Build-A.Cell

2.4.3. Matriu de competitivitat

La matriu de competència que hem creat per comparar els diferents joc mencionats anteriorment està basada en les característiques que volen que tingui el projecte. Moltes d'aquestes opcions les hem reduït a verdader o fals, sent sempre verdader la opció que nosaltres busquem.

Joc	**Serious Game**	**Plataformes 2D**	**Relacionat: Bioquímica**	**Relacionat: respiració cel·lular**	**Públic objectiu**	**Plataforma**
Immune Defense	SÍ	SÍ	EN PART	NO	General	Windows
Immune Attack	SÍ	NO	EN PART	NO	General	Windows
Fold it	SÍ	NO	SÍ	NO	Estudiants	Windows
Re-Mission /2	SÍ	NO	EN PART	NO	Pacients	Android
CRISPR	SÍ	NO	SÍ	NO	Estudiants	Windows/VR
Build-A-Cell	SÍ	SÍ	SÍ	NO	Estudiants	Windows

Taula 4: Matriu de competència

2.4.4. Conclusions estudi de mercat

Amb aquesta cerca s'ha vist que no som els primers en crear un *Serious Game* relacionat amb la Bioquímica. Aquesta és una branca de la ciència molt complexa, amb conceptes molt difícils d'assimilar si no es té una base gràfica al darrera amb una bona explicació. Es per això que la quantitat de jocs educatius que hem trobat en aquest àmbit és importat. Tot i això, cap d'aquests jocs treballen sobre la respiració cel·lular, el nostre concepte a explicar, pel que el nostre projecte no perd la seva finalitat.

L'estudi mostra una gran preferència pel PC com a plataforma en front d'altres. Això pot donar-se a que alguns d'aquests projectes tenen molts anys i no era tant habitual treballar amb *smartphones* i per Android. En qualsevol cal, i pel tipus de joc que volem crear, creiem que plataformes mòbils segueix sent el que millor s'ajusta als objectius, ja que facilita a l'usuari tant la baixada (una simple cerca a *Google Play*) com el trobar un moment per jugar-hi (utilitzant transport públic, en una cua d'espera, etc.).

2.5. Públic objectiu i perfil de jugador

El joc té un *target* de jugador molt específic. En concret aquest projecte es crea per ajudar a estudiants universitaris de carreres que incloguin la matèria de Bioquímica a entendre com funcionen els processos de la respiració cel·lular i les seves etapes. Això inclou Bioquímica, departament que ens ha encarregat el projecte, però també Medicina, Veterinària, Biotecnologia, etc. Al tenir un públic objectiu tant específic, podem assumir les següents característiques:

- L'edat és superior als 18 anys.
- És estudiant universitari o està en sectors relacionats amb la bioquímica.
- Té uns coneixements previs dels conceptes més bàsics de la biologia cel·lular i les molècules.
- Té un interès per aprendre com funciona els processos de Glicòlisi, Cicle de Krebs i respiració cel·lular.

Dit això, un *Serious Game* ha de ser tant educatiu com entretingut i ha de tenir mecàniques divertides. Això pot fer que el nostre públic pugui ampliar-se si ala jugabilitat esta ben assolida, tot i que el públic aliè a la Bioquímica sempre tindrà un nivell de satisfacció inferior.

Pel que fa a la tipologia de jugadors i seguint la classificació segons *Richard Burtle*, el joc es centra clarament en els *"Achievers"*. Aquests son aquells jugadors que tenen com objectiu resoldre els reptes i dificultats proposats pel joc. El joc planteja diferents reptes al jugador a mida que va avançant pels diferents nivells, amb més o menys dificultat, i és d'aquests reptes d'on el jugador també n'extraurà els coneixements que el joc vol transmetre. Es

veritat que podríem incloure elements que donessin motivació a altres tipus de jugadors, com podria ser un sistema de puntuacions per tal que els jugadors poguessin competir entre ells, però és un aspecte que s'hauria de considerar en futures iteracions del prototip, no d'inici.

2.6. Conclusió de la viabilitat del projecte

Un cop estudiat el mercat i vist els diferents requeriments del projecte, podem concloure que aquest és viable, tot i que potser amb certes limitacions.

L'estudi de mercat deixa clar que tot hi haver bastants jocs seriosos relacionats amb la bioquímica, cap d'ell inclou els apartats que nosaltres volem aplicar. Pel mateix motiu podem concloure que utilitzar jocs seriosos per temes acadèmics relacionats amb aquesta matèria és una bona idea, donat que ja s'ha realitzat amb èxit en altres apartats.

Sent que el projecte el realitza una sola persona, els recursos tècnics no són un problema, ja que les eines físiques les tenim disponibles, i la resta de recursos com el programari són gratuïts. La principal limitació la trobem en els recursos humans, ja que el tractar-se d'una sola persona, el desenvolupament del prototip es pot veure limitat.

3. Planificació

En aquest apartat es mostra com s'organitzarà tota la feina a fer al llarg del projecte. Per tal de poder assolir tots els objectius que volem, hem d'estructurar i repartir el temps del qual disposem. Per poder fer-ho bé hem de considerar molts factors, des de quan de temps volem dedicar a cada apartat, fins a possibles inconvenients que poden anar sorgint.

És important també tenir clar fins on volem o podem arribar. Crear un videojoc des de zero és molta feina, i com s'ha dit, normalment la du a terme un equip. Sent que aquest projecte el realitza una sola persona, s'ha de ser realista a l'hora de decidir quins punts podem assolir en un primer prototip i quins és millor deixar-los per més endavant. Tot això sabent a més que aquest "primer prototip" haurà passat per diferents iteracions on s'hauran hagut de refer o modificar diferents aspectes. A l'hora de realitzar un videojoc sempre és important anar realitzant proves durant el projecte per avaluar-lo i anar veient com el podem millorar.

Per últim, cal remarcar que la major part d'aquest projecte es va realitzar molt abans de decidir que s'utilitzaria com a projecte de fi de grau, i en el seu moment no es va dur a terme un registre del temps utilitzat per assolir cada objectiu ni les dates concretes de les reunions realitzades, tant amb el tutor del projecte com amb l'equip del departament de la Facultat de Biologia que l'havia encarregat. La planificació que veurem en els següents apartats és una aproximació semblant a la duta a terme en el seu moment, però aplicant la distribució de les diferents etapes al curs acadèmic 21/22.

3.1. Pla de treball

Com en molts projectes de videojocs, per a la realització del projecte seguirem un procés iteratiu per a tots els apartats del desenvolupament. Tant per la estètica, mecànica o narrativa, anirem subdividint el projecte per crear prototips bàsics on poder avaluar el funcionament i tornar, si fos necessari, a una fase de disseny on modificar el que no acabi de funcionar o directament refer-lo del tot. A part, i al treballar en solitari, treballarem els diferents apartats de la creació d'un videojoc (narrativa, estètica, tecnologia i mecàniques) en paral·lel per tal d'utilitzar el temps de manera més eficient i no trobar-nos en que un apartat ens limiti la continuació dels altres.

D'altra banda, i a diferència de molts dels projectes de creació de videojocs, el nostre no inicia amb una tabla rasa on fer una pluja d'idees sobre què volem fer. El nostre projecte comença amb la sol·licitud del departament de Biologia de crear un *Serious Games* que expliqui la respiració cel·lular. Això té avantatges i inconvenients, però on afecta també és en els següents apartat:

- Per una part pot afectar a l'abast del projecte. En el nostre cas no tenim un termini d'entrega determinat, però sí tenim el requeriment que estigui acabat dins del curs acadèmic. Sent que el projecte és també un Treball de Fi de Grau, no suposa una limitació extra de la que ja tindríem si el projecte fos totalment autònom.

- També afecta al cronograma. La planificació ha d'incloure reunions periòdiques amb l'equip del departament de Biologia per tal de mostrar i discutir tots els apartats del joc. Tot el desenvolupament de les mecàniques, de l'estètica o de la narrativa ha d'estar en sintonia a les demandes realitzades durant la creació del projecte. De fet, l'ideal d'aquestes reunions és realitzar-les dins de cada avaluació del procés iteratiu, ja que algunes característiques que poden ser adients per nosaltres, poden no ser del gust del client, i al final és el client qui en certa mesura decideix.

3.1.1. Abast del projecte

Per definir l'abast que tindrà el projecte hem definit una sèrie d'objectius que volem assolir en el prototip final. Per definir aquests objectius s'ha tingut en comte el temps de treball disponible en relació al que costa desenvolupar un videojoc sencer. S'ha de ser realista i veure que la finalització del projecte com a producte final és impossible, així que es busca crear un prototip amb el màxim d'elements representatius del joc possibles. Aquests objectius també inclouen, com no pot ser d'una altra manera, tots i cadascun dels requisits plantejats pel departament de Biologia.

Per tal d'explicar quins són aquests objectius, els hem dividit en els quatre apartats essencials del desenvolupament d'un videojoc en ordre d'importància dins del projecte:

- **Mecàniques:** el projecte busca explicar 3 processos en concret del metabolisme de les cèl·lules, i com a tal volem crear 3 nivells diferenciats, un per cadascun dels processos. Volem implementar un mínim de 2 dels 3 nivells amb les corresponents mecàniques ja acabades i on es pugui avaluar si arribem a transmetre els coneixements a explicar. Aquestes mecàniques buscaran en la mesura del possible no allunyar-se molt de la realitat, ja que així ens ho demana el client.

- **Estètica:** volem crear tots els *assets* necessaris per la implementació dels 2 nivells escollits, així com per una portada, un índex i una IU inicial. Un cop més ho farem sota els requeriments i supervisió del client, buscant el seu vistiplau del resultat.

- **Tecnologia:** amb el projecte aprofundirem el necessari tant en el motor de videojocs com en les demés eines per tal de poder crear un prototip de joc 2D per mòbil amb sistemes Android.

- **Narrativa:** en el nostre cas aquest apartat és el de menor pes. Busquem donar un mínim de narrativa per tal de buscar una cohesió entre els nivells.

3.1.2. Diagrama de Gantt

Aquest tipus de diagrama és una representació gràfica del que serà el desenvolupament del projecte, passant per totes les etapes i distribuint-les en el temps necessari per a cadascuna d'elles.

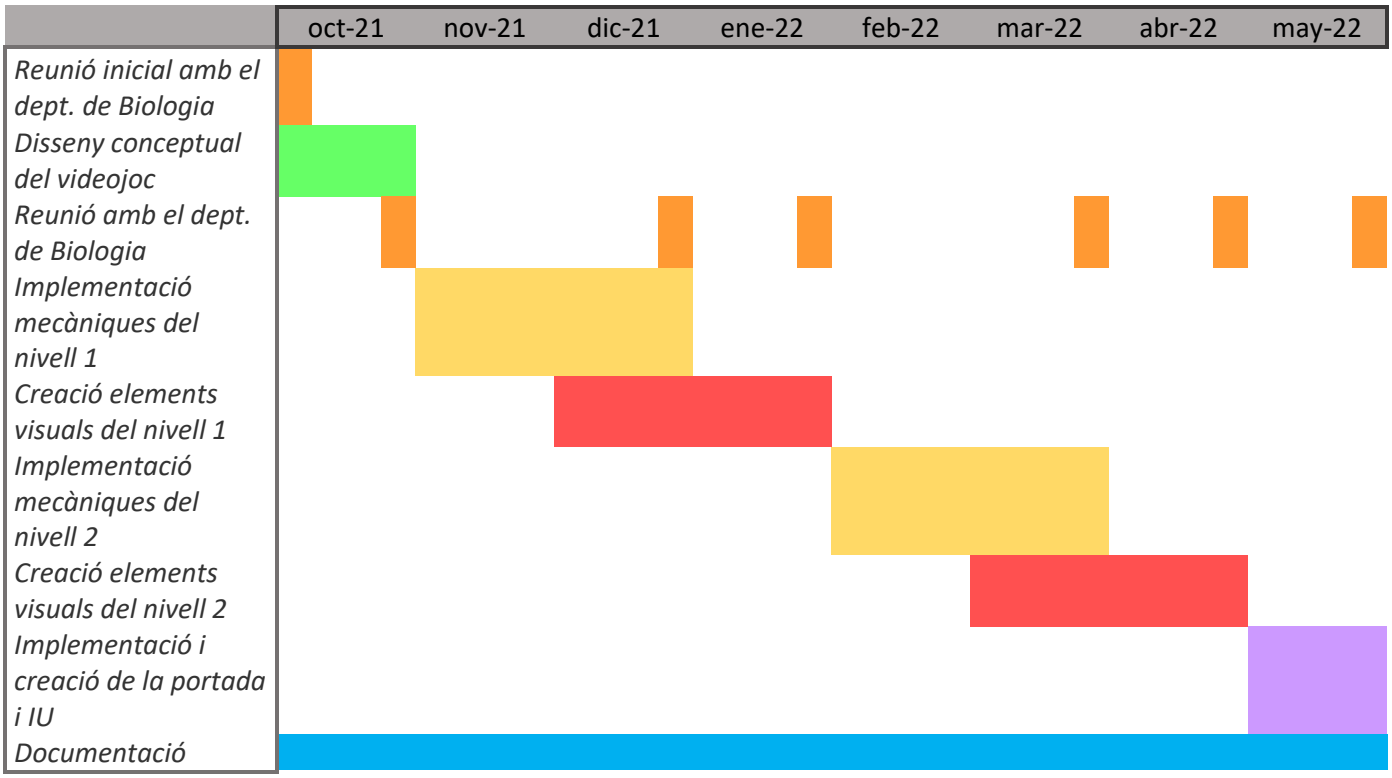


Figura 7: Diagrama de Gantt del projecte

4. Marc de treball i conceptes previs

A l'hora de crear un *Serious Game* és molt important haver entès a la perfecció tots els conceptes a explicar, per tal de idear les mecàniques més adients. En aquest punt definirem breument tots els processos i conceptes que aniran apareixent durant el desenvolupament dels nivells. Sent que el joc seriós treballa la Respiració cel·lular, començarem definint de que es tracta, quines són les parts implicades i quins processos engloba.

4.1. Respiració cel·lular:

La respiració cel·lular és la degradació total, per mitjà de l'oxidació, d'algunes substàncies orgàniques fins a matèria inorgànica per alliberar energia. El combustible pot ser la glucosa, un àcid gras o d'altres molècules orgàniques com aminoàcids o cossos cetònics. En el cas dels éssers humans, i dels animals en general, el carburant s'obté mitjançant la digestió i arriba a les cèl·lules a través del sistema circulatori. Pel que fa a l'oxigen, s'extreu de l'aire mitjançant l'acció dels pulmons o de les brànquies i també arriba a les cèl·lules gràcies al transport per la via sanguínia, viatjant fixat en l'hemoglobina que hi ha en els eritròcits (glòbuls vermells).

Aquests són alguns dels conceptes previs a entendre abans de seguir aprofundint:

- On es realitzen aquests processos:
    - **Cèl·lula eucariota**: S'anomena cèl·lula eucariota a totes les cèl·lules amb un nucli cel·lular delimitat dins d'una doble capa lipídica, l'embolcall nuclear, la qual és porosa i conté el seu material hereditari, fonamentalment la seva informació genètica (Figura 8).

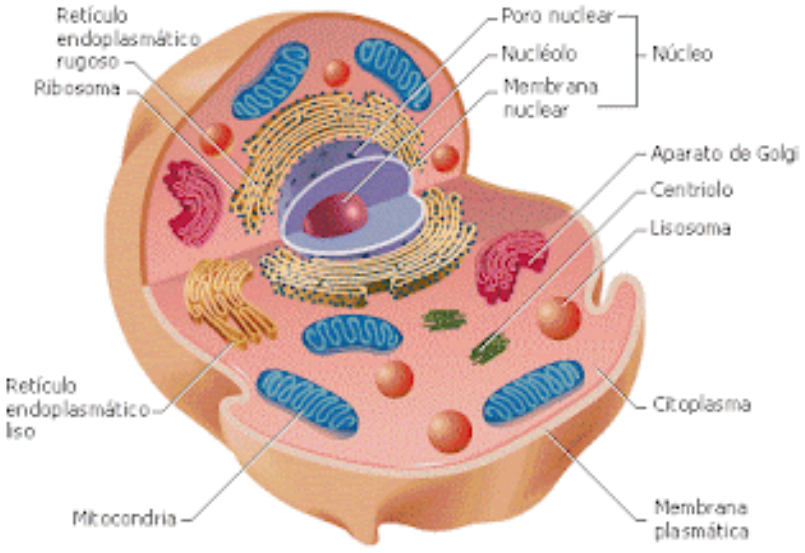


Figura 8: Cèl·lula eucariota i els seus orgànuls

- **Citoplasma**: és la part de la cèl·lula eucariota que es troba entre el nucli cel·lular i la membrana. La seva funció és contenir els orgànuls cel·lulars i contribuir al moviment. El citosol és el lloc on es produeixen molts dels processos metabòlics de la cèl·lula.

- **Orgànul**: Un orgànul és cadascuna de les parts formades per material orgànic microscòpic que formen part d'una cèl·lula i s'encarreguen de dur a terme les diferents funcions vitals cel·lulars. Cada orgànul té una funció específica. Hi ha molt tipus d'orgànuls, particularment en les cèl·lules eucariotes.

- **Mitocondri**: Un mitocondri és un orgànul envoltat per una doble membrana que es troba en la majoria de les cèl·lules eucariotes. A vegades es descriuen els mitocondris com a «plantes d'energia de les cèl·lules» perquè generen la major part dels subministraments de trifosfat d'adenosina (ATP) que necessita la cèl·lula com a font d'energia química. A més a més de subministrar energia, els mitocondris estan implicats en diferents processos, com ara la comunicació, la diferenciació i l'apoptosi, així com el cicle cel·lular i el creixement.

- Com es realitzen aquests processos:

  - **Procés metabòlic**: és el conjunt de reaccions químiques que tenen lloc en un organisme per a mantenir-lo viu. Aquests processos permeten als organismes de créixer i reproduir-se, de mantenir les estructures i respondre al medi. El metabolisme se sol subdividir en dues categories: el catabolisme (procés de transformació de molècules orgàniques complexes en molècules senzilles, emmagatzemant l'energia despresa en el procés) i l'anabolisme (procés invers al catabolisme, i per tant, utilitza energia).

  - **Molècula orgànica:** és un compost químic que conté una cadena d'àtoms de carboni, enllaçats entre ells mitjançant enllaços covalents, i enllaçats a àtoms d'hidrogen.

  - **Enzim**: Els enzims són substàncies orgàniques, gairebé sempre de natura proteica, que acceleren reaccions químiques. Els enzims interaccionen amb molècules de partida (substrats) i catalitzen la seva transformació en altres de diferents (productes). Intervenen amb un paper important en gairebé tots els processos cel·lulars. Com que tenen un alt grau de selectivitat respecte al substrat i cadascun d'ells catalitza unes reaccions molt concretes, el conjunt d'enzims que es produeixen en una cèl·lula determina les rutes metabòliques que s'hi poden dur a terme.

- **Procés d'oxidació:** és el procés electroquímic pel qual un ió o àtom perd un o diversos electrons. Atès que l'oxidació representa una pèrdua d'electrons i que aquests han d'ésser guanyats per un altre element (oxidant), tota oxidació va acompanyada d'una reducció (oxidoreducció), d'aquí que també se les conegui per reaccions REDOX (figura 9).

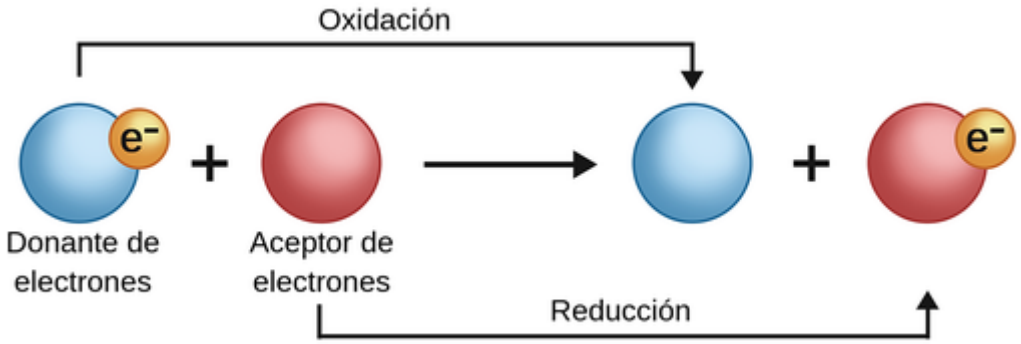


Figura 9: Ex quema conceptual d'una reacció REDOX

- **Nicotinamida adenina dinucleòtid (NAD):** és un coenzim d'oxidoreducció present en totes les cèl·lules vives. En el metabolisme, el NAD^+ està implicat en les reaccions redox, en les quals transporta electrons d'una reacció a una altra. Així doncs, a les cèl·lules el coenzim és present en dues formes diferents (NAD^+ i NADH). El NAD^+ és un agent oxidant: accepta electrons d'altres molècules, quedant així reduït. Aquesta reacció forma el NADH, que pot ser utilitzat com a agent reductor per a cedir electrons.

- **Trifosfat d'adenosina (ATP):** és un nucleòtid multifuncional que té un paper important en la biologia cel·lular com a coenzim. L'ATP (figura 10) transporta energia química a l'interior de les cèl·lules per al metabolisme. És una font d'energia produïda durant la fotosíntesi i la respiració cel·lular i és consumit per molts enzims en una multitud de processos cel·lulars, incloent-hi les reaccions de biosíntesi, motilitat i divisió cel·lular. L'ATP pot ser produït en reaccions redox utilitzant sucres simples i complexos (carbohidrats) o lípids com a font d'energia.

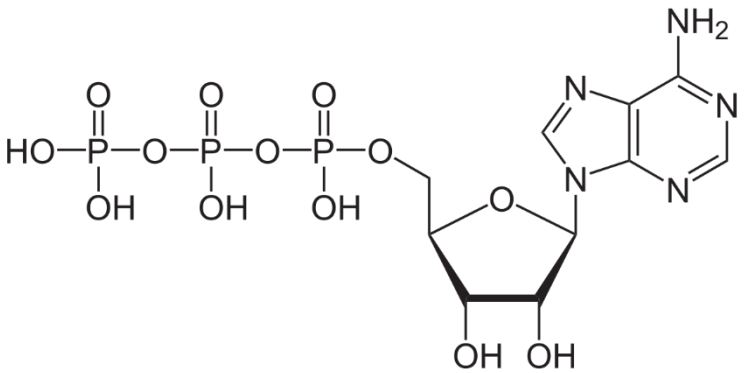


Figura 10: Estructura del trifosfat d'adenosina (ATP)

Tornant a la respiració cel·lular, en les cèl·lules eucariotes la respiració es realitza en els mitocondris i passa en tres etapes: oxidació de l'àcid pirúvic, Cicle del Krebs i la cadena transportadora d'electrons o fosforilació oxidativa de l'ADP a ATP. Tot i això, i previ a aquestes etapes, és necessari que fora del mitocondri, en el citoplasma, s'hagi produït la Glicòlisi.

4.1.1. Glicòlisi

La glicòlisi, glucòlisi o via *d'Embden-Meyerhof*, és una via metabòlica per la qual una molècula de glucosa (Glc) és oxidada fins a dues molècules d'àcid pirúvic o piruvat (Pyr). La glicòlisi és l'inici tant de la respiració aeròbica com de l'anaeròbica i en aquest sentit és l'arquetip d'un procés metabòlic universal que es pot trobar (amb molt poques variacions) a la majoria de tipus cel·lulars de gairebé tots els organismes. És la forma més ràpida d'aconseguir energia per a la cèl·lula i, en el metabolisme dels hidrats de carboni, generalment és la primera via a la que es recorre, utilitzant com a font principal la glucosa proporcionada per l'alimentació, la glucosa obtinguda de novo (gliconeogènesi) i l'alliberada en la degradació de les reserves de glucogen.

És la via inicial del catabolisme (degradació) d'hidrats de carboni (sucres) i té tres funcions principals:

- La generació de molècules que poden actuar com a font d'energia cel·lular o poder reductor (ATP i NADH).
- La producció de piruvat, necessari per iniciar el cicle de l'àcid cítric com a part de la respiració aeròbica.
- La producció de compostos intermediaris de sis i tres carbonis que poden ésser utilitzats en altres vies metabòliques.

En gairebé totes les cèl·lules es desenvolupa de la mateixa manera: una molècula de glucosa es converteix en dues molècules de piruvat, formant-se ATP i NADH. En eucariotes i procariotes, la glicòlisi té lloc al citosol de la cèl·lula. La glicòlisi consisteix en 10 reaccions enzimàtiques que converteixen una molècula de glucosa en dues de piruvat, el qual és capaç de seguir altres vies metabòliques com és l'inici de la Respiració cel·lular. Aquestes 10 reaccions es divideixen en dues fases: la primera, de consum d'energia, i la segona, d'obtenció d'energia.

La primera fase consisteix a transformar una molècula de glucosa en dues molècules de gliceraldehid-3-fosfat (una molècula de baix nivell energètic) mitjançant l'ús de dues molècules d'ATP. Aquest pas permet

duplicar els resultats de la segona fase d'obtenció d'energia, ja que les molècules inicials són de 6 àtoms de carboni i les finals únicament de 3.

En la segona fase el gliceraldehid 3P pateix una oxidació (que provoca la reducció del coenzim NAD+) i es transforma en un compost d'alt nivell energètic, el qual generarà una molècula d'ATP (en realitat dues, ja que es generen dues molècules de gliceraldehid per cada glucosa). Aquesta obtenció d'energia s'aconsegueix mitjançant la unió d'una reacció fortament exergònica després d'una lleument endergònica. Aquesta unió té lloc una vegada més en aquesta fase, generant dues molècules de piruvat i dues més d'ATP. D'aquesta manera en la segona fase s'obtenen 4 molècules d'ATP.

Així doncs la reacció global de la glicòlisi és:

glucosa + 2 NAD^+ + 2 ADP + 2 P_i → 2 piruvat + 2 NADH + 2 ATP + 2H^+ + H_2O

4.1.1.1. Etapes de la glicòlisi

Fase I o de despesa de energia:

- Primer Pas - Hexocinasa: primera reacció de la glicòlisi consisteix en la fosforilació de la glucosa per convertir-la en glucosa 6-fosfat. És una reacció irreversible en la qual es consumeix ATP.

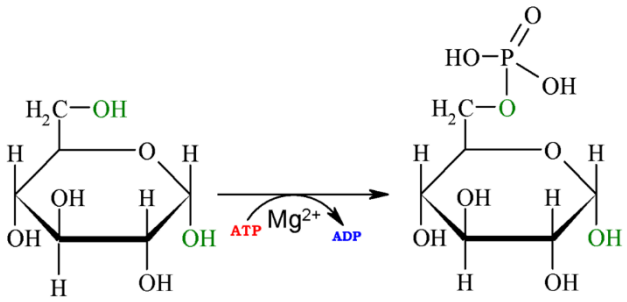


Figura 11: Primer pas de la glicòlisi

- Segon pas - Glucosa-6-fosfat isomerasa: en aquesta reacció, una isomerització aldosa-cetosa catalitzada per l'enzim glucosa-6-fosfat isomerasa, isomeritza la glucosa 6-fosfat a fructosa 6-fosfat.

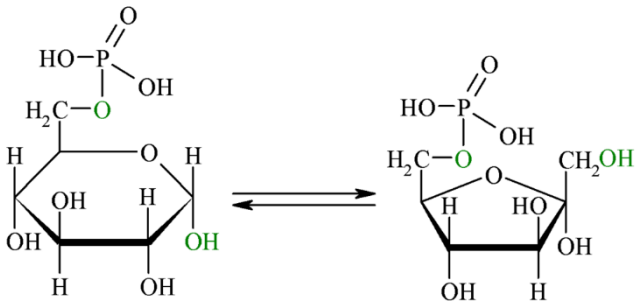


Figura 12: Segon pas de la glicòlis

- Tercer pas. Fosfofructocinasa: En aquesta reacció es fosforila el carboni 1 de la fructosa 6-fosfat generant fructosa 1,6-bisfosfat. És una reacció irreversible, catalitzada per l'enzim fosfofructocinasa-1 (PFK-1) i en la qual es consumeix ATP.

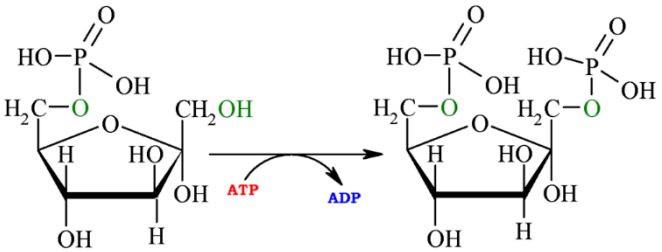


Figura 13: Tercer pas de la glicòlisi

- Quart pas. Fructosa-1,6-bisfosfat aldolasa: mitjançant una condensació aldòlica reversible, aquest enzim trenca la fructosa 1,6-bifosfat en dues molècules de tres carbonis (trioses): dihidroxiacetona fosfat i gliceraldehid 3-fosfat.

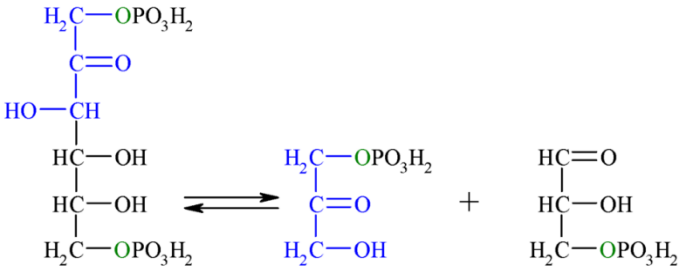


Figura 14: Quart pas de la glicòlisi

- Cinquè pas. Triosafosfat isomerasa: La dihidroxiacetona fosfat generada en el pas anterior és isomeritzada a gliceraldehid 3-fosfat (G3P) mitjançant l'enzim triosafosfat isomerasa, de manera que s'obtenen dues molècules de G3P per cada molècula de glucosa.

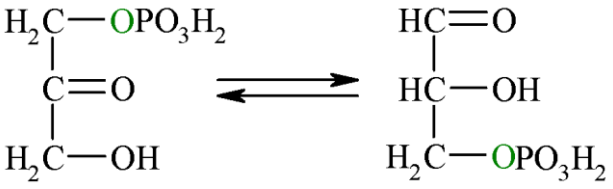


Figura 15: Cinquè pas de la glicòlisi

Fase II o de benefici energètic:

- Sisè pas. Gliceraldehid-3-fosfat deshidrogenasa: catalitzada per l'enzim gliceraldehid-3-fosfat deshidrogenasa (GAP deshidrogenasa) en cinc passos, consisteix a oxidar el gliceraldehid 3-fosfat utilitzant NAD^+ per afegir un grup fosfat a la molècula, augmentant d'aquesta manera l'energia del compost.

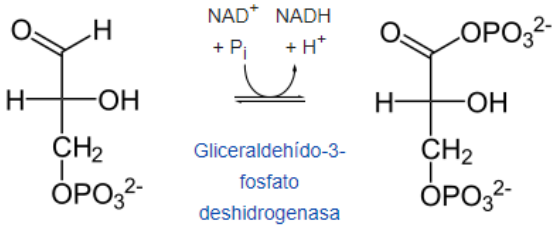


Figura 16: Sisè pas de la glicòlisi

- Setè pas. Fosfoglicerat cinasa: En aquest pas l'enzim fosfoglicerat cinasa transfereix el grup fosfat de l'1,3-bifosfoglicerat a un ADP, generant així la primera molècula d'ATP de la via. En aquesta etapa es recuperen 2 ATP, ja que la glucosa es transforma en dues molècules de gliceraldehid.

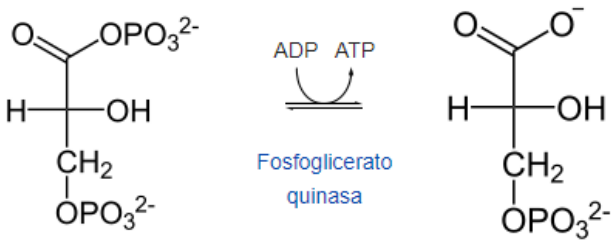


Figura 17: Setè pas de la glicòlisi

- Vuitè Pas. Fosfoglicerat mutasa: El 3-fosfoglicerat provinent de la reacció anterior s'isomeritza donant lloc a 2-fosfoglicerat mitjançant l'enzim fosfoglicerat mutasa.

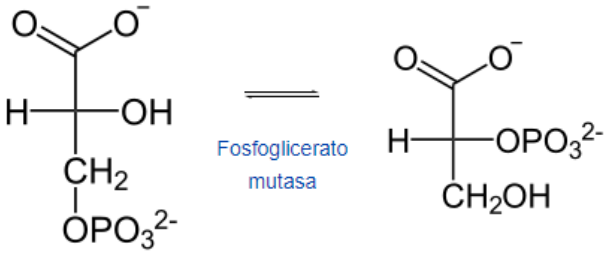


Figura 18: Vuitè pas de la glicòlisi

- Novè pas. Enolasa: L'enzim enolasa propicia la formació d'un doble enllaç en el 2-fosfoglicerat, eliminant una molècula d'aigua formada per l'hidrogen del C_2 i l'OH del C_3.

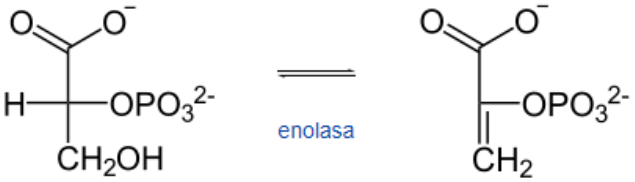


Figura 19: Novè pas de la glicòlisi

- Desè pas. Piruvat cinasa: L'enzim piruvat cinasa (depenent de Mg2+ i K+) transfereix el grup fosfat del fosfoenolpiruvat a una molècula d'ADP, generant ATP i la forma enòlica del piruvat, que tautomeritza a piruvat.

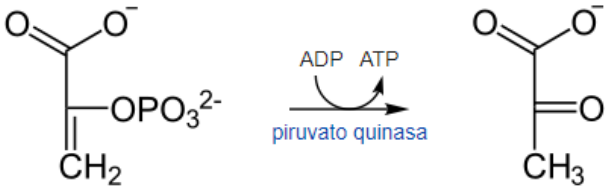


Figura 20: Desè pas de la glicòlisi

4.1.2. Descarboxilació del piruvat:

El piruvat obtingut en el procés de la glicòlisi és metabolitzat a través d'una de tres rutes catabòliques. En el cas que ens interessa, en organismes i teixits aeròbics, es produirà una descarboxilació a través de la piruvat deshidrogenasa per formar el grup acetil de l'acetilcoenzim A. En les cèl·lules eucariotes, la descarboxilació del piruvat es dona exclusivament en els mitocondris.

4.1.3. Cicle de Krebs:

El cicle de Krebs (també anomenat cicle de l'àcid cítric o cicle dels àcids tricarboxílics) és una ruta metabòlica, és a dir, una successió de reaccions químiques que formen part de la respiració cel·lular en totes les cèl·lules en les quals l'acceptor final sigui una molècula inorgànica, el sofre o l'oxigen.

L'acetil-CoA format en la descarboxilació del piruvat s'incorpora al cicle de Krebs i es degrada completament a CO_2. Amb això es generen "equivalents de reducció" en forma de NADH I FADH2 (o també anomenats transportadors d'electrons), que s'utilitzen en les reaccions posteriors de la fosforilació oxidativa per produir grans quantitats d'ATP. També la degradació de greixos i aminoàcids va a parar al cicle de Krebs principalment a través d'acetil-CoA.

D'altra banda, el cicle de Krebs genera també components per a la biosíntesi. Les funcions integradores del cicle de Krebs fan que aquest sigui un element central del metabolisme, que compren importants passos de degradació i anabòlics, i està completament regulat en coordinació amb altres rutes. El cicle té lloc a la matriu mitocondrial.

A diferència de la seqüència lineal de reaccions de la glicòlisi, el cicle de Krebs és una seqüència tancada de nou reaccions en la qual el producte inicial i el producte final són els mateixos, l'oxalacetat (veure figura 21). En cada volta del cicle s'agafa un grup d'acetil-CoA i s'oxida a dos CO_2, igualment s'originen tres NADH, un $FADH_2$ i un GTP.

4.1.3.1. Etapes del cicle de Krebs

- Primer pas: La reacció inicial del cicle és la unió de l'oxolacetat amb l'acetil-CoA formant citrat duta a terme per l'enzim citrat sintasa.
- Segon i tercer pas: l'enzim aconitasa isomeritza el citrat a isocitrat.

- Quart pas: l'isocitrat deshidrogenasa, la primera de quatre oxidoreductases del cicle de Krebs, descarboxila l'isocitrat oxidativament a α-oxoglutarat, així s'allibera CO_2 i es forma NADH.
- Cinquè pas: es produeix una altra descarboxilació oxidativa també molt exergònica, que està catalitzada pel complex multienzimàtic α-oxoglutarat deshidrogenasa. Aquí el complex converteix l'α-oxoglutarat mitjançant una descarboxilació oxidativa a succinil-CoA. En aquest procés s'allibera un altre CO_2 i es forma NADH.
- Sisè pas: el succinil-CoA-sintetasa catalitza el trencament del succinil-CoA formant GTP a partir de GDP i Pi i també regenera el CoA-SH.
- Setè pas: el succinat deshidrogenasa oxida el succinat a fumarat formant $FADH_2$.
- Vuitè pas: aquesta és la hidratació del fumarat a L-malat. L'enzim fumarasa catalitza aquesta reacció d'addició estereoespecífica.
- Novè pas: el malat deshidrogenasa converteix el L-maltat a oxolacetat.

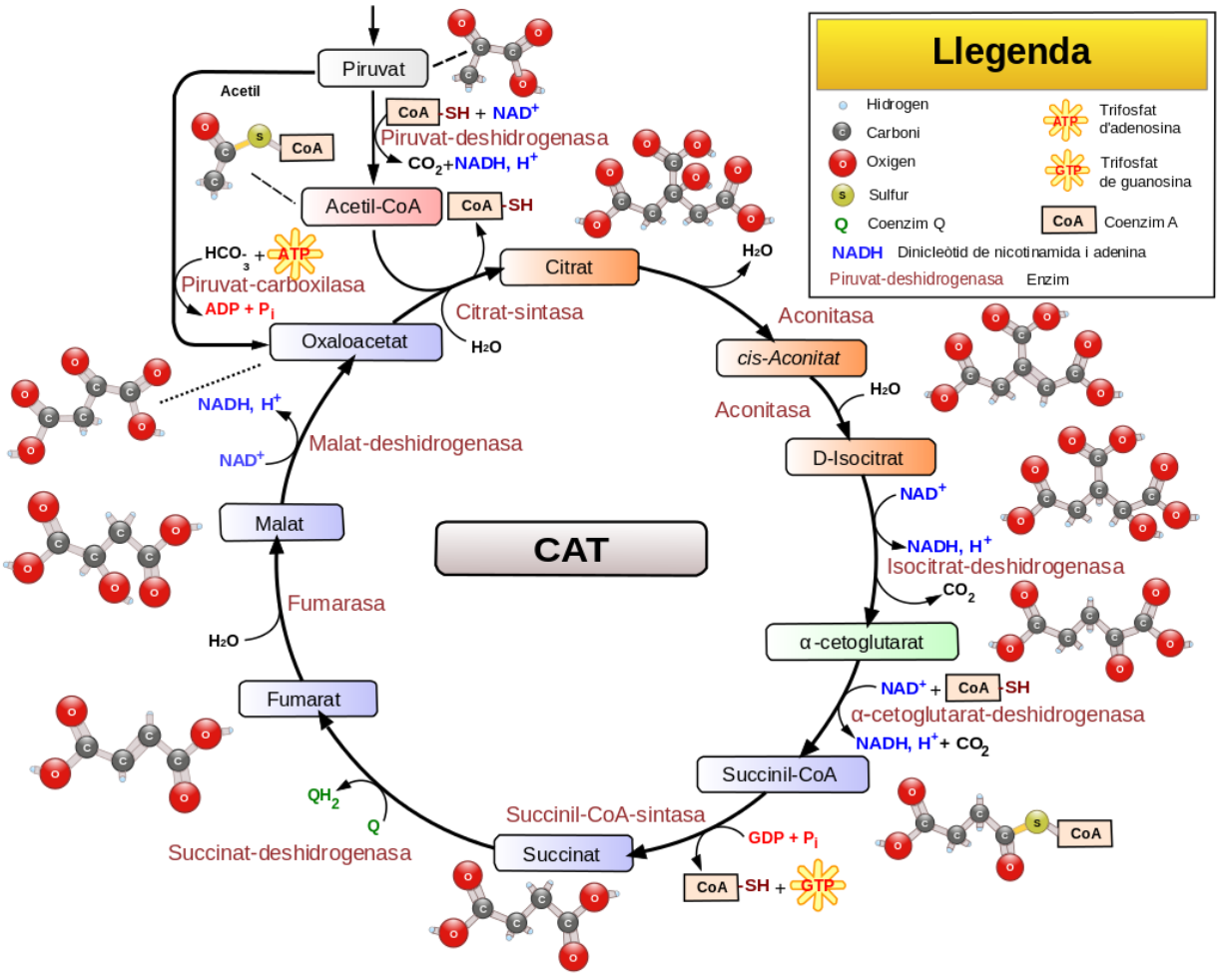


Figura 21: Esquema cicle de Krebs

Així es tanca el cicle. El cicle ha arribat de nou al seu inici. La reacció neta és:

$Acetil\text{-}CoA + 3NAD^+ + FAD + GDP + P_i + 2H_2O \dashrightarrow 2CO_2 + CoA\text{-}SH + 3NADH + 3H^+ + FADH_2 + GTP$

4.1.4. Fosforilació oxidativa

La fosforilació oxidativa és una via metabòlica que utilitza energia alliberada per l'oxidació de nutrients per produir adenosina trifosfat (ATP). Encara que les diverses formes de vida a la terra utilitzen una gran varietat de nutrients, quasi totes duen a terme la fosforilació oxidativa per produir ATP, la molècula que subministra energia al metabolisme. Aquesta via probablement és tan generalitzada perquè és una manera molt eficient d'alliberar energia, en comparació amb els processos alternatius de fermentació, com la glucòlisi anaeròbica.

Durant la fosforilació oxidativa, els electrons són transferits des dels donadors d'electrons fins als acceptadors d'electrons com l'oxigen, en reaccions redox. Aquestes reaccions redox alliberen energia que es fa servir per formar ATP. En cèl·lules eucariotes, aquestes reaccions redox són realitzades per una sèrie de complexos proteics del mitocondri i hi estan implicats cinc complexos proteics.

L'energia alliberada pel flux d'electrons a través d'aquesta cadena transportadora d'electrons es fa servir per transportar protons a través de la membrana interna del mitocondri, en un procés anomenat quimiosmosi. Això genera energia potencial en forma d'un gradient de pH i un potencial elèctric a través d'aquesta membrana. Aquest emmagatzematge d'energia és aprofitat permetent als protons que tornin a través de la membrana a favor del gradient, a través d'un enzim anomenat ATP sintasa. Aquest enzim utilitza aquesta energia per generar ATP a partir d'adenosina difosfat (ADP), en una reacció de fosforilació. Aquesta reacció és regulada pel flux d'electrons, que provoca la rotació d'una part de l'enzim; la ATP sintasa és un motor mecànic rotatori.

Encara que la fosforilació oxidativa és una part vital del metabolisme, produeix espècies reactives de l'oxigen com el superòxid o el peròxid d'hidrogen. I en conseqüència, es propaguen radicals lliures que danyen la cèl·lula i contribueixen a malalties i, possiblement, a l'envelliment. Els enzims que realitzen aquesta via metabòlica també són la diana de moltes drogues i verins que inhibeixen la seva activitat.

4.1.4.1. Quimiosmosi

La quimiosmosi és un dels dos modes de biosíntesi de trifosfat d'adenosina (ATP), que es produeix a nivell de membrana. Els electrons, que gràcies a la glicòlisi i el cicle de Krebs han estat carregats als transportadors d'electrons NADH i $FADH_2$, són cedits a la cadena de transport d'electrons (constituïda per quatre complexos proteics situats a la membrana interna del mitocondri). El pas dels electrons comporta

l'alliberament d'energia, que és emmagatzemada en enllaços de 36 molècules de difosfat d'adenosina (ADP), per mitjà de l'enllaç del grup fosfat i la síntesi de molècules de trifosfat d'adenosina (ATP).

Mentre que els electrons baixen per la cadena de transport, els ions H^+ presents a la matriu són transportats activament a l'espai intermembranal. Així es genera una diferència de concentració dels H^+ als dos vessants de la membrana interna mitocondrial (veure figura 22). A causa d'aquest gradient de concentració, els ions H^+ tendeixen a tornar a entrar per difusió. Com que la membrana és impermeable, per a travessar-la els cal una proteïna de transport: l'ATP sintetasa, un complex enzimàtic que catalitza la síntesi de l'ATP a partir d'ADP i fosfat. D'aquesta manera, la reentrada dels ions forneix a la reacció de síntesi de l'ATP l'energia necessària.

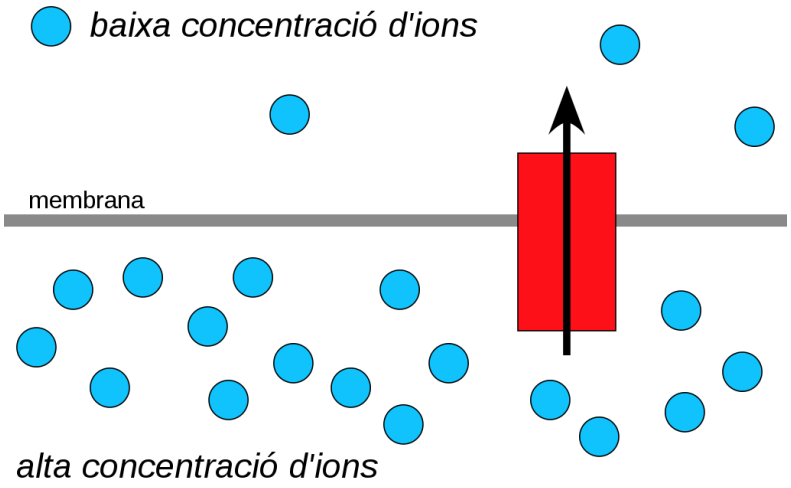


Figura 22: Esquema conceptual del gradient d'electrons

Així doncs es completa la respiració cel·lular, principal font d'energia de les cèl·lules. Tenint en compte els diferents processos explicats, el balanç energètic del conjunt global és el següent:

- Glicòlisi: 2 ATP, 2 NADH i 2 piruvat.
- Descarboxilació del piruvat: 2 molècules de piruvat donen 2 NADH.
- Cicle de Krebs: 2 molècules d'acetil-CoA donen 2 ATP, 2 $FADH_2$, i 6 NADH.

En total són 4 ATP, 10 NADH i 2 $FADH_2$. Assumint que per cada NADH podem produir 3 ATP i del $FADH_2$ en podem produir 2:

- Fosforilació oxidativa: 34 ATP a partir dels 10 NADH i dels 4 $FADH_2$.

5. Disseny del videojoc

En aquesta secció s'explica a nivell conceptual com realitzarem tots els apartats del nostre joc, així com perquè hem decidit fer-ho de la manera escollida. És aquest disseny el que ens servirà de guia durant tot el desenvolupament del joc, i el que marcarà les passes a seguir per assolir els objectius que volem. Durant aquest apartat marcarem com serà la estètica del joc, les mecàniques que tindrà, o l'ús del programari que farem servir per crear-lo.

5.1. Gènere

En el nostre cas, els *Serious games* són un gènere com a tal. En aquest gènere s'engloben tots aquells videojocs que més enllà del divertiment dels jugadors, busquen que hagin assimilat certs coneixements. Això no vol dir que aquests jocs no puguin englobar-se a la vegada en altres gèneres, ja que tots ells extreuen molts elements dels companys. El més important a l'hora d'escollir quin gènere s'adequa més a les nostres necessitats és tenir clar quin tipus de mecàniques utilitzaren, ja que aquestes seran en última instancia les que transmetran aquells coneixements que volem (Figura 23). Així, les plataformes que utilitzaran els nostres usuaris també és important.

Així doncs, tenint en compte els apartats mencionats, nosaltres hem escollit el gènere de puzles i plataformes 2D. Els jocs d'aquest gènere van perfectes en plataformes mòbils com poden ser telèfons i tabletes, permeten una gran flexibilitat a l'hora de dissenyar els seus nivells i adaptar-los a les nostres necessitats, i s'ajusta a les mecàniques que hem escollit per els diferents nivells, mecàniques que explicarem en els següents apartats.

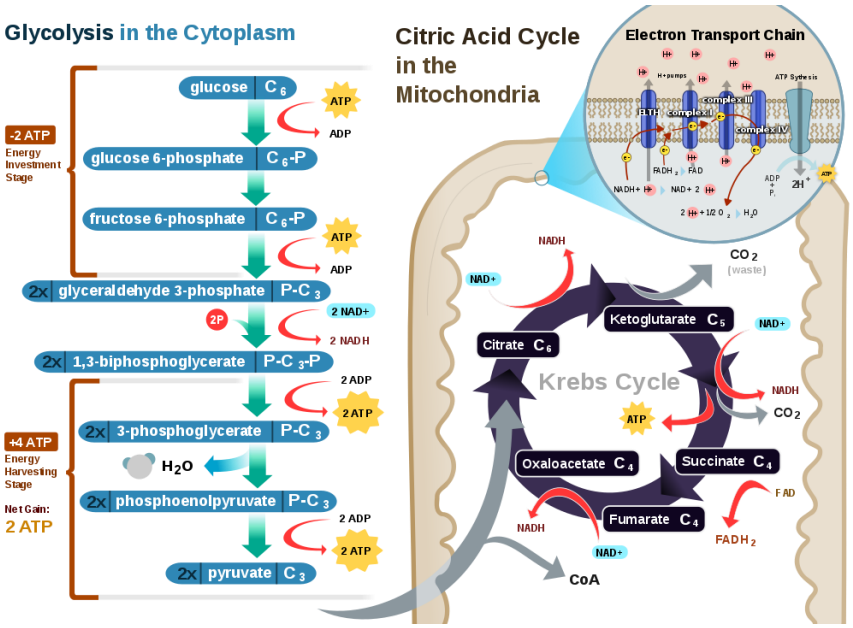


Figura 23: Esquema de la respiració cel·lular

5.2. Level Design

Abans d'endinsar-nos a explicar les mecàniques explicarem com hem fet la distribució de nivells, ja que ens ajudarà a entendre que trobarem en cadascun i el perquè de les mecàniques. Tot i això, un objectiu que se'ns requereix des del departament de Biologia és buscar una interpretació el més realista possible de com es realitzen aquests processos, i aquest condicionant marcarà tant els nivells com les mecàniques utilitzades en cada nivell.

En àmbit general, el joc està dividit en 3 nivells, un per cada "macro concepte" a explicar. Aquests conceptes, com hem vist en apartats anteriors, són la glicòlisi, el cicle de Krebs i la fosforilació oxidativa. El fet de adjudicar cada concepte a un nivell diferent ens permet adaptar al màxim les mecàniques i els diferents elements a la transmissió dels mateixos.

5.2.1. Nivell 1: Glicòlisi

Aplicant el requeriment de buscar el màxim de realisme, s'ha optat per fer un escenari sense límits. Com hem dit, al moure'ns a nivell molecular, l'espai dins del citoplasma de la cèl·lula és immens (Figura 24), i per tant hem ideat un sistema per tal d'enganyar al jugador creant uns límits invisibles, on al creuar un d'ells, automàticament se'ns teletransporta al costat contrari. Els orgànuls de la cèl·lula són merament estètics i no tenen cap interacció amb el jugador.

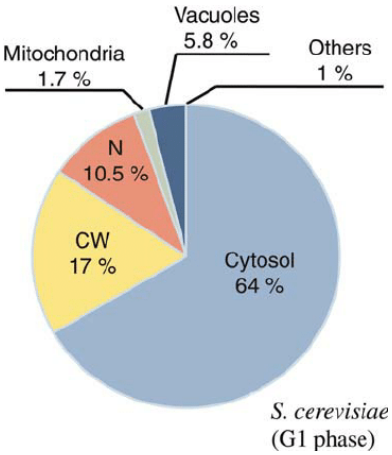


Figura 24: Proporcions del contingut d'una cèl·lula

El jugador tindrà la seva molècula al centre de l'escenari, en una zona segura. Fins que no es realitzi algun moviment i aquesta surti d'allà, no es generaran la resta de molècules. Un cop surti i comencin a generar-se la resta de molècules, es pot interaccionar amb elles. Aquesta interacció pot variar segons el nivell de dificultat, com es veurà més endavant.

Per últim, el nivell inclou un enzim per cada etapa i trobar-lo és l'objectiu del jugador. Aquest enzim és genera a un lloc aleatori, sempre lo suficientment lluny del centre per tal que el jugador no el pugui localitzar d'inici. Quan la molècula principal interacciona amb l'enzim, canviem de fase o acabem el nivell, segons correspongui. A cada etapa es reinicia l'escenari, amb la molècula principal al centre.

La Glicòlisi consta d'un total de 10 fases. Entre fases varia la molècula principal, el enzim, i característiques com poden ser la seva reversibilitat.

5.2.2. Nivell 2: Cicle de Krebs

En aquest nivell trobem una repetició dels elements anteriors, ja que el funcionament i les mecàniques són els mateixos. Les diferencies entre aquest i el primer són estètiques, i que les fases d'un i l'altre poden variar (nombre de fases reversibles, etc.).

El cicle de Krebs consta d'un total de 9 fases. Entre fases varia la molècula principal, el enzim, i característiques com poden ser la seva reversibilitat.

5.2.3. Nivell 3: Fosforilació oxidativa

Al ser la fosforilació oxidativa un dels conceptes més complexes, en aquest nivell hem deixat més de costat el realisme i hem optat per un disseny que ajudi, junt amb les mecàniques, a la l'explicació. Aquí ens trobem en un nivell de plataformes, limitat a esquerra i a dreta per una barrera física, i on el nivell avança en sentit únic cap a baix a través de les plataformes, algunes d'aquestes mòbils per augmentar la dificultat.

El jugador es trobarà aquí amb un grapat d'electrons en forma de caniques, que haurà d'anar desplaçant entre plataformes fins arribar al final, procurant que ningun es perdi pel camí.

5.2.4. Economia interna

Tot i que en altres jocs, sobretot *Serious Games*, aquest és un tema més secundari, en el nostre cas juga un paper clau i molt important. Els conceptes que estem explicant amb el nostre joc tenen una finalitat molt importat, donar energia a la cèl·lula, i com a tal, no només hem d'explicar com passa això, sinó quanta energia aconseguim al final de tot el procés (Figura 25). És per això que el nostre joc té un comptador d'electrons i un altre d'ATPs. A més comentar que aquests comptadors mantenen els valors d'un nivell a l'altre.

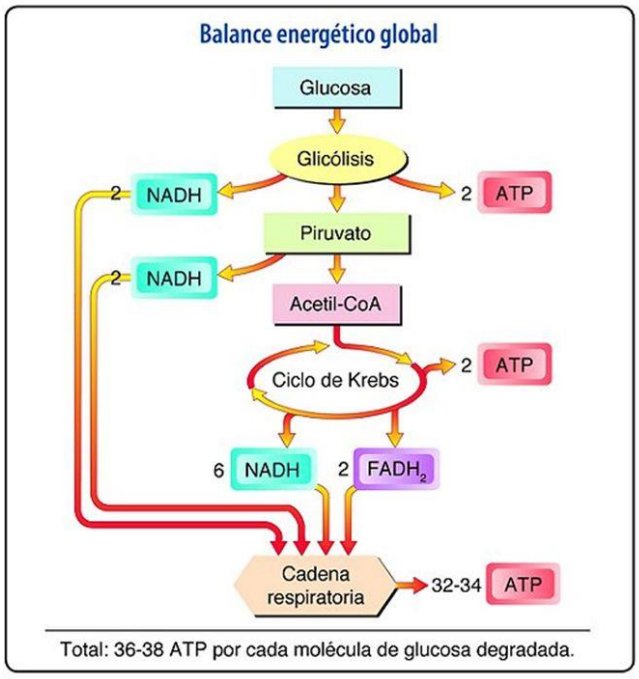


Figura 25:Balanç energètic al final dels diferents processos

El comptador d'electrons engloba dues molècules, els NAD^+ i els FAD^+. Podríem fer un comptador per tots dos per separat, però complicaríem el concepte i en resum tots dos compleixen la mateixa funció. Totes dues molècules s'utilitzen per crear ATP, el producte final que dona energia a la cèl·lula. Com a tal, aquest comptador va augmentant a mida que avancem entre els dos primers nivells, que és quan aquests es generen, i va disminuint en el últim nivell, on aquests són convertits en ATP.

El comptador d'ATPs començarà a 0 en el primer nivell, i a mida que avancem anirà en augment a mida que es generin, acabant el tercer nivell amb el nombre màxim que haguem aconseguits d'aquest.

Tot i que en la realitat aquest electrons mai es perden i sempre és genera els mateixos ATPs, a nivells de dificultat superiors el jugador pot ser que en perdi algun pel camí, fent que hagi de repetir el nivell si vol aconseguir el màxim de puntuació.

5.3. Gameplay

El *Gameplay* és la font de l'entreteniment en tots els videojocs. Deixant de banda que els Serious Games tenen el factor de l'aprenentatge, el *Gameplay* és la pedra angular de tots els videojocs i el primer que es té en compte a l'hora de dissenyar-los. Aquest està definit pels reptes plantejats als jugadors i per les mecàniques a implementar per superar aquests reptes, i l'equilibri entre les dues dona el component interactiu divertit de tots els videojocs.

Cal puntualitzar que l'objectiu d'aprenentatge, el fet de voler que els jugadors aprenguin els processos bioquímics, no és un repte en si. Això s'ha d'aconseguir de manera gairebé passiva durant la interacció dels jugadors amb els diferents nivells, ha de ser un aprenentatge de forma natural i fluida.

5.3.1. Reptes i la seva jerarquia

El repte global del videojoc és guanyar la partia, completar tot el contingut i arribar al final. Però aquest repte es va dividint en subconjunts de diferents reptes, i que fan més interessant el joc en sí. Així doncs, trobem reptes de dos tipus, de nivell atòmic o no divisibles, o els no atòmics o divisibles. Aquesta divisió ens permet crear una jerarquia de reptes.

A l'hora de fer els reptes, el dissenyador s'ha de de posar en la pell del jugador. El prototip treballat requereix un balanç en el disseny molt acurat, ja que depenent de com els equilibrem, de la dificultat que l'hi donem el joc, dependrà en gran part de la motivació i interès que tingui el jugador a completar el joc. En el nostre cas, si el jugador no te interès i no assoleix el final del joc, no haurà assolit els coneixements que volem transmetre, amb tota seguretat.

En el joc els reptes varien segons el nivell, ja que la forma de completar-los és diferent. Tot i això el objectiu global és el mateix, passar per les diferents etapes de cada procés bioquímic. Amb tot, la nostra jerarquia de reptes és la següent (Figura 26):

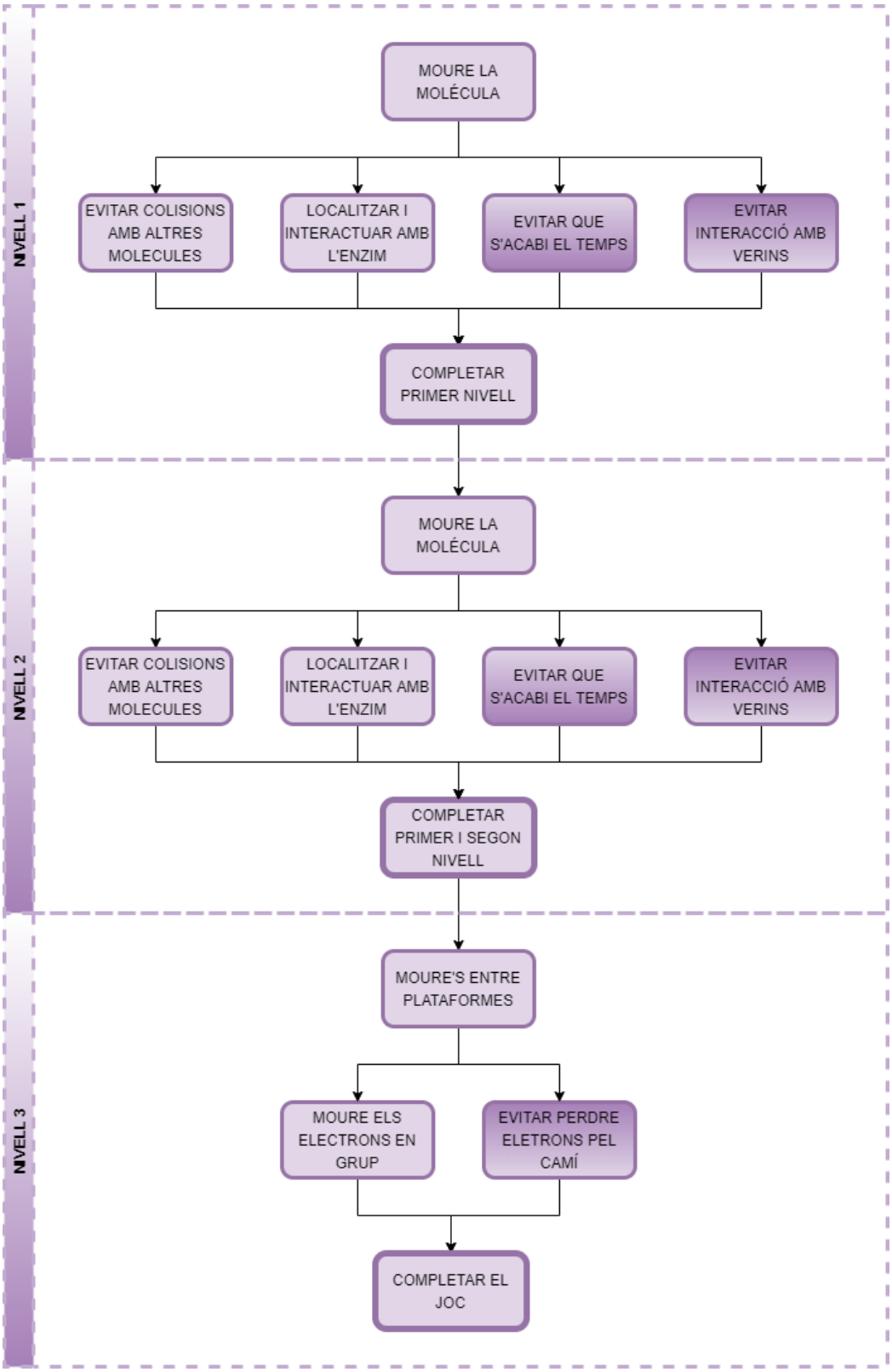


Figura 26: Jerarquia de reptes

Llegenda:

- Reptes permanents
- Reptes puntuals
- Reptes a nivell difícil

5.3.2. Mecàniques

A l'hora de fer el joc, des del departament de biologia han estat molt concisos de quins requeriments volen que tingui el projecte. Junt amb el realisme que ja hem citat abans, i de fet molt lligat a aquest mateix requeriment, es vol evitar mecàniques que desentonessin molt del funcionament de la cèl·lula. En general es vol que el joc doni la sensació que l'interior de la cèl·lula no es un lloc buit on no hi passa res, sinó que és un lloc gegant, farcit d'altres molècules i elements, que creen en conjunt un ambient caòtic per on és difícil moure's.

Com a tal i per donar aquesta sensació de caos se'ns dona les següents bases:

- Buscar al màxim la sensació d'escenari sense límits (ja tractada al disseny de nivells)
- Que el jugador no sigui un personatge com a tal, sinó una mà invisible que ajuda a les diferents molècules a passar per les diferents reaccions
- Les molècules no són sers vius que poden saltar, disparar, etc., pel que les mecàniques han de ser molt limitades per ajudar al realisme

Per tal d'assolir tot això hem optat per limitar el joc a la mecànica de moviment, donant-li a aquesta una certa complexitat per fer-la més entretinguda; i a la interacció que tenen les diferents molècules (els nostre personatges en aquest cas) amb diferents elements de l'escenari. Combinant aquests dos elements assolim plantejar al jugador els reptes abans explicats.

5.3.2.1. Moviment

La mecànica de moviment és la principal en el joc, i per tant no l'hem limitat als típics controls WASD amb les direccions endavant, endarrere, dreta i esquerra. Hem creat unes mecàniques que permeten al jugador influir en el comportament dels personatges (molècules), però d'una forma no precisa, de manera similar als autos de xoc, per aconseguir transmetre que el que passa a l'interior de les cèl·lules és mes caòtic i que costa moure's (Figura 27).

Per poder explicar com hem dissenyat aquesta mecànica farem una diferenciació entre els 2 primers nivell, que utilitzen la mateixa, i el tercer nivell, que n'utilitza una de totalment diferent. Això es deu a que el concepte que el nostre *Serious Game* vol explicar en el tercer nivell és molt complex, i hem necessitat crear un escenari bastant diferent, amb una mecànica de moviment que no te res a veure amb

la primera. Tot i això, totes dues tenen en comú que no busquen aquesta precisió a l'hora de moure les molècules.

Figura 27: Autos de xoc, inspiració a l'hora de crear el moviment

- Moviment nivells 1 i 2

Per situar-nos una mica, en aquest nivell les molècules es troben al centre d'un escenari sense límits aparents, i s'ha de trobar amb un enzim que es trobar en algun lloc aleatori.

El moviment en aquests dos nivells l'hem dissenyat utilitzant la pantalla tàctil dels mòbils o tabletes (que són les plataformes a les que va destinat el joc). Utilitzant qualsevol dit, tot i que per practicitat es recomana l'índex, el jugador ha de tocar qualsevol punt de la pantalla, i sense aixecar-lo en cap moment, moure'l fins al punt que ell desitgi. Aquest gest com a tal no mou la molècula directament, d'aquí que no faci falta tocar-la. En canvi, aquest moviment ens marca dos punts, un inicial on s'ha establert el contacte amb la pantalla, i un final on s'ha aixecat el dit. D'aquests dos punts n'obtenim un vector que ens marcarà la direcció en que volem moure la molècula, i la distancia entre els dos punts ens marcarà la intensitat o força a aplicar.

Així doncs, si cliquem la pantalla però no desplacem el dit, la força serà 0 i la molècula no es veurà influenciada. Tanmateix, si movem el dit al llarg de tota la pantalla, la força a aplicar serà la màxima (la tenim limitada amb valors interns), i la molècula modificarà la seva direcció com s'escaigui.

Cal mencionar que en tot moment estem parlant de forces i no de valors purs de velocitat. Això vol dir que sí tenim la molècula movent-se en una direcció a la màxima velocitat, i apliquem una força en la direcció contraria, potser que no sigui suficient per canviar la direcció i que l'únic que fem fos frenar-la. En aquest exemple, el que hauríem de fer seria repetir la mateixa acció per tal d'anar reduint la inèrcia i acabar modificant la direcció.

Amb tot això aconseguim que sigui difícil controlar amb precisió la molècula, donant al jugador un punt de dificultat. A part, al tractar-se de forces, les col·lisions amb altres molècules ens dificultaran encara més aquest moviment, tal i com veurem més endavant.

- Moviment nivell 3

Per situar-nos un altre cop, en aquest nivell ens trobem en la membrana dels mitocondris, movent-nos per un gradient per tal de poder sortir al citoplasma un altre cop.

La millor forma que hem trobat de transmetre aquest gradient ha estat aplicant en aquest nivell moviment vertical, en concret només cap a baix. Per fer-ho hem extret la idea "del joc de les caniques", on mitjançant l'oscil·loscopi dels telèfons i tabletes, i aplicant forces de gravetat, aconseguim que al girar el telèfon aquestes es moguin.

En el nostre cas les caniques són els electrons que han d'anar sortint intercanviant-se per ATPs, i el jugador els ha d'anar movent a través de plataformes, algunes d'elles mòbils, intentant que sempre estiguin tots agrupats per tal de no perdre ningun pel camí. En general vindria a ser similar al joc mostrar en la Figura 28.

Figura 28:Joc on s'han de moure les caniques pel laberint

5.3.2.2. Interacció amb altres objectes

L'altre mecànica que tenim en el joc són les diferents interaccions que tenen les molècules entre elles o amb altres objectes. Totes elles tenen alguna importància de cara al jugador, ja sigui el objectiu a

complir, o interaccions a evitar. Un cop més i com ja ha passat amb la mecànica de moviment, hem de distingir entre els dos primers nivells i el tercer.

- Interaccions nivells 1 i 2

Per situar-nos una mica, en aquest nivell les molècules es troben al centre d'un escenari sense límits aparents, i s'ha de trobar amb un enzim que es trobar en algun lloc aleatori. En aquests dos nivells trobem 3 tipus de interacció diferents: interacció amb enzims, interacció amb altres molècules i interacció amb verins.

a) Interacció amb enzims:

En aquesta interacció es basa l'objectiu del jugador en cada etapa dels dos primers nivells. Com s'ha vist anteriorment, tant la glicòlisi com el cicle de Krebs estan dividits en diferents fases, i en cada fase està implicat un enzim diferent. Com a tal, cada cop que el jugador aconsegueix que la molècula interaccioni amb el enzim, aquest avança de fase, i així fase per fase fins a completar el nivell.

Tot i això, algunes d'aquestes interaccions entre molècula i enzim en la vida real són reversibles. Per reflectir això, alguns dels enzims, al interaccionar amb ells, activen un cronòmetre amb un compte enrere. Si aquest temps acaba sense que el jugador així avançat de fase, aquest retrocedeix a la fase anterior.

b) Interacció amb molècules innòcues:

Com s'ha dit abans, l'interior de les cèl·lules és enorme i esta replet de diferents molècules. Tot i que en la realitat moltes d'aquestes no poden interaccionar entre elles, nosaltres les hem utilitzat per crear un altre factor de dificultat al joc.

Un cop el jugador surt de la zona segura es comencen a general moltes molècules de forma, velocitat i direcció aleatòria. Això fa que la nostra molècula moltes vegades xoqui amb elles, donant així a canvis de direcció i velocitat de totes dues. Per contrarestar això, el jugador es veu obligat a corregir la trajectòria de la molècula principal molt sovint, ja sigui per evitar col·lisions o per corregir-les un cop aquestes ja hagin passat (Figura 29).

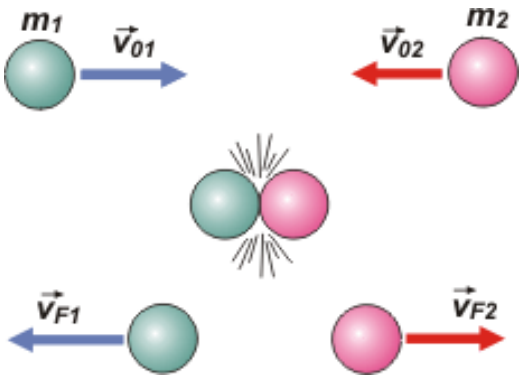


Figura 29:Col·lisió entre partícules

c) Interacció amb verins:

Aquests verins es troben només en dificultat elevada. Els trobem actuant com la resta de molècules, movent-se aleatòriament i amb diferents velocitats, i estèticament es poden distingir de les altres, però la principal diferencia és en la col·lisió. Aquests verins "segresten" la nostra molècula, fent que el jugador retrocedeixi de fase automàticament i perdi ATPs del comptador al fer-ho. Així doncs, el jugador ha de fer tot el possible per detectar verins aviat i esquivar-los mentre busca l'enzim.

- Interacció nivell 3

Per situar-nos un altre cop, en aquest nivell ens trobem en la membrana dels mitocondris, movent-nos per un gradient per tal de poder sortir al citoplasma un altre cop.

En aquest nivell no tenim objectes que interaccionin d'alguna manera amb els electrons. Aquests els anirem movent entre plataformes al llarg del nivell fins arribar al final d'aquests mateix, evitant perdre electrons en el trajecte.

5.3.3. Nivells de dificultat

El nostre joc tindrà dos nivells de dificultat, un de normal i un d'avançat. Això és així per estimular als jugadors a rejugar-lo, ja que el nivell normal pot ser poc exigent. Durant les múltiples reunions amb l'equip de Biologia es va remarcar la importància que tenia per ells que el jugador completés el joc evitant complicacions en uns conceptes que de per si ja són difícils d'assimilar. Així doncs hem decidit fer un nivell bastant assequible, sense condicions de mort com a tal, i un altre de més exigent que només és desbloquejable al completar el primer.

Un exemple d'un concepte que no podem tocar en el nivell normal, però si es veurà afectat en el nivell difícil, és el comptador d'electrons i ATPs. En la vida real, quan una cèl·lula ha acabat els tres processos que tenim entre mans, sempre haurà generat el mateix numero d'electrons i ATPs, cadascuna de les vegades. I aquests valors són molt importants i formen part del coneixement que els alumnes han d'assolir. Així doncs, en el nivell de dificultat normal, aquests valors varien a mida que avancem durant el joc, tal i com ho farien en la realitat, acabant sempre amb els valors que generaria una cèl·lula normal. En canvi, en nivell difícil, hem fet que aquests valors canviïn si el jugador interacciona amb verins, perd electrons al nivell 3, etc.

5.3.3.1. Diferències entre nivells de dificultat

- Nivells 1 i 2:
  - Augmenta considerablement la quantitat de molècules que poden xocar amb la principal
  - Inclou verins, molècules que apareixen aleatòriament i en nombre molt baix, que d'interaccionar amb la molècula principal el fan retrocedir de fase i perdre ATPs.
  - Es redueix el temps que te el jugador per evitar que les reaccions reversibles el facin retrocedir
- Nivells 3:
  - Perdre electrons pel camí afecta considerablement al nombre d'ATPs que generarà al final
  - Perdre tots els electrons implicar repetir nivell
  - Canvi en els patrons de moviments de les plataformes

5.3.4. Tutorial

Les mecàniques són senzilles i fàcils d'explicar. Es per això que a mode de tutorial hem optat per fer unes animacions breus a l'inici de cada nivell on es veu visualment com funcionen les diferents mecàniques i com aconseguir els objectius.

5.4. *Flowchart*

El *flowchart* és un tipus de diagrama que representa un procés o un flux de treball. En el següent diagrama (Diagrama 2) es pot veure el *flowchart* del joc o aplicació. En ell es representa els diferents moviments que pot fer l'usuari entre les diferents pantalles del joc, ensenyant quines són i com s'accedeixen a elles (Figura 30).

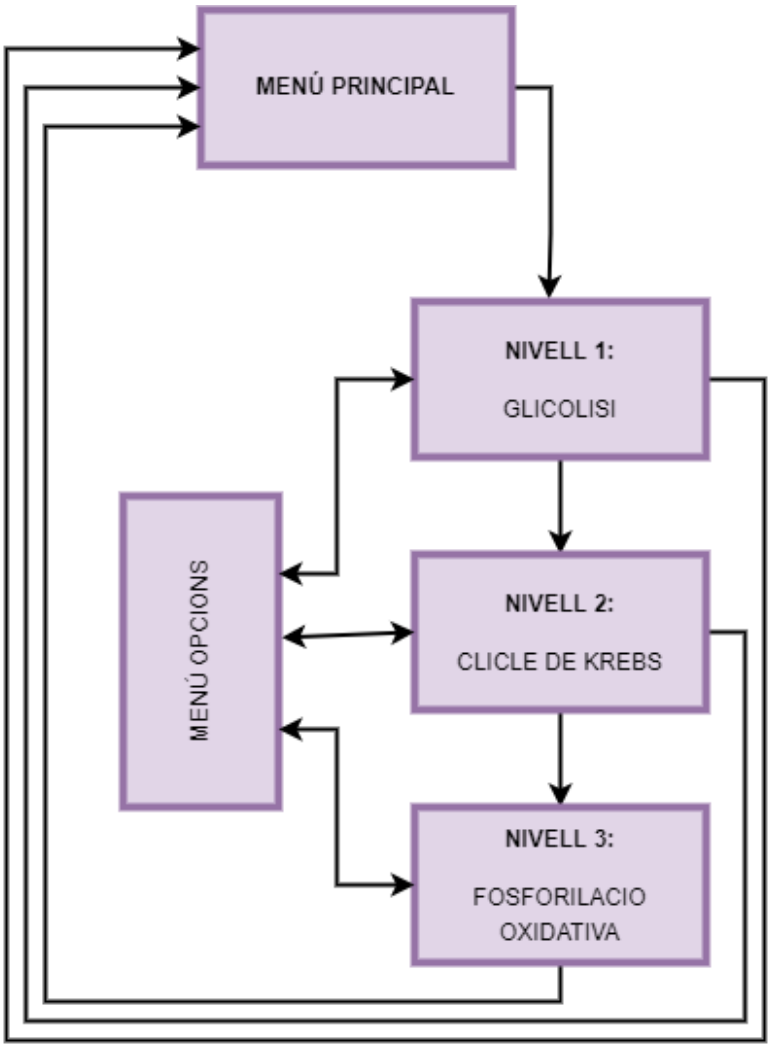


Figura 30: Flowchart

En aquest diagrama podem veure com, a partir del menú principal, s'inicia el joc, i que cada nivell porta al següent, fins completar-lo i acabar el joc. A la vegada, des de cada nivell podem accedir tant al menú principal com al menú d'opcions, i des del menú d'opcions podem tornar a reprendre la partida.

5.5. Narrativa

El nostre joc no te una narrativa especialment profunda ni ens fa falta per poder explicar els conceptes que tenim mentre es juga. La nostra narrativa són els propis processos bioquímics i com es van desenvolupant mentre es juga. Així doncs el jugador podrà veure mentre va jugant como una glucosa del nivell 1 es va transformant fins acabar reduïda en un grup d'electrons que acabaran convertits en ATPs. És una narrativa breu, concisa i molt rodona, no cal més pel projecte.

5.5.1. Personatges

Tot i que el nostre joc no te personatges com a tal, cal recalcar que les molècules principals que anirà movent el jugador, així com els enzims amb els que ha d'interaccionar, han de ser molt clars i identificables d'alguna manera, ja que aquests formen part dels coneixements que el jugador haurà d'assolir (Figura 31).

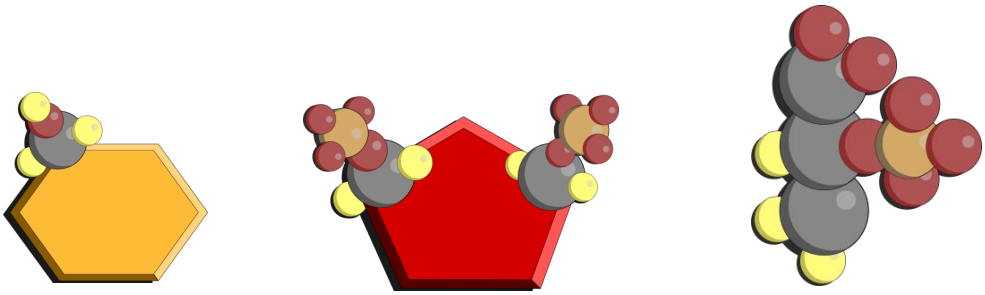


Figura 31: Exemples dels personatges/molècules que mourem en el nostre joc

5.5.2. Món del joc

En aquest apartat veurem com es situa el món del joc amb els seus escenaris. Per fer-ho però separarem els tres nivells que tenim per tal de poder donar més detall.

5.5.2.1. Nivell 1: Glicòlisi

- Dimensió ambiental: ens trobem en el citoplasma d'una cèl·lula. Nosaltres ens mourem a nivell molecular, pel que tot i estar dins d'una cèl·lula, l'espai és immens. En aquest espai els orgànuls de la cèl·lula són gegants, i tot el citoplasma està replet d'altres molècules.

- Dimensió física: aplicant el requeriment de buscar el màxim de realisme, s'ha optat per fer un escenari sense límits. Com hem dit, al moure'ns a nivell molecular, l'espai dins del citoplasma de la cèl·lula és immens, i per tant hem ideat un sistema per tal "d'enganyar" al jugador creant uns límits invisibles, on al creuar un d'ells automàticament se'ns teletransporta al costat contrari. Els orgànuls de la cèl·lula són merament estètics i no tenen cap interacció amb el jugador. Sí hi ha interacció amb la resta de molècules, però aquesta pot variar. Per últim, el nivell inclou un enzim per cada etapa i trobar-lo és l'objectiu del jugador.

- Dimensió temporal: tot i no haver una temporalitat global, sí compta com a temporalitat els comptes enrere d'aquelles etapes on les reaccions són reversibles.

5.5.2.2. Nivell 2: Cicle de Krebs

- Dimensió ambiental: sortim del citoplasma i entrem a un orgànul en concret, el mitocondri. Tot i tècnicament estar en un lloc molt més reduït, a nivell molecular som tant petits que seguim tenint un escenari immens.

- Dimensió física: un cop més s'ha optat per fer un escenari sense límits per les mateixes raons que el nivell anterior, i per tant comptem amb el sistema de barreres invisibles. Aquí no trobem orgànuls, sinó deformacions de la paret del mitocondri (ja que aquests són rugosos), però un cop més són estètiques. Un cop més hi ha interacció amb la resta de molècules i aquesta pot variar, i el nivell inclou un enzim per cada etapa altre cop sent aquest l'objectiu del jugador.

- Dimensió temporal: igual que en el cas anterior, tot i no haver una temporalitat global, sí compta com a temporalitat els comptes enrere d'aquelles etapes on les reaccions són reversibles.

5.5.2.3. Nivell 3: Fosforilació oxidativa

- Dimensió ambiental: aquí ens trobem a la membrana del mitocondri, sortint cap al citoplasma. Al ser la fosforilació oxidativa un dels conceptes més complexes, en aquest nivell hem deixat més de costat el realisme i hem optat per un disseny que ajudi, junt amb les mecàniques, a la seva explicació.

- Dimensió física: ens trobem en un nivell de plataformes, limitat a esquerra i a dreta per una barrera física, i on el nivell avança en sentit únic cap a baix a través de les plataformes, amb algunes d'aquestes mòbils.

- Dimensió temporal: aquest nivell si que no compta amb temporalitat de cap tipus.

5.6. Interfície gràfica

Aquest apartat engloba tots aquells elements que ajudaran al jugador a moure's entre els diferents nivells i escenes del joc, o durant els propis nivells a tenir una referencia d'aquelles dades que necessiti la partida. És molt important que tot marqui una cohesió entre sí i que a nivell artístic res desentoni. A continuació es descriuen els diferents elements que inclou el nostre joc.

5.6.1. Pantalla inicial

Pantalla inicial que apareix tant bon punt iniciem la aplicació. Serveix a mode de presentació del joc amb el títol i el logo, i només cal prémer el botó inici per moure'ns al menú principal (Figura 32).

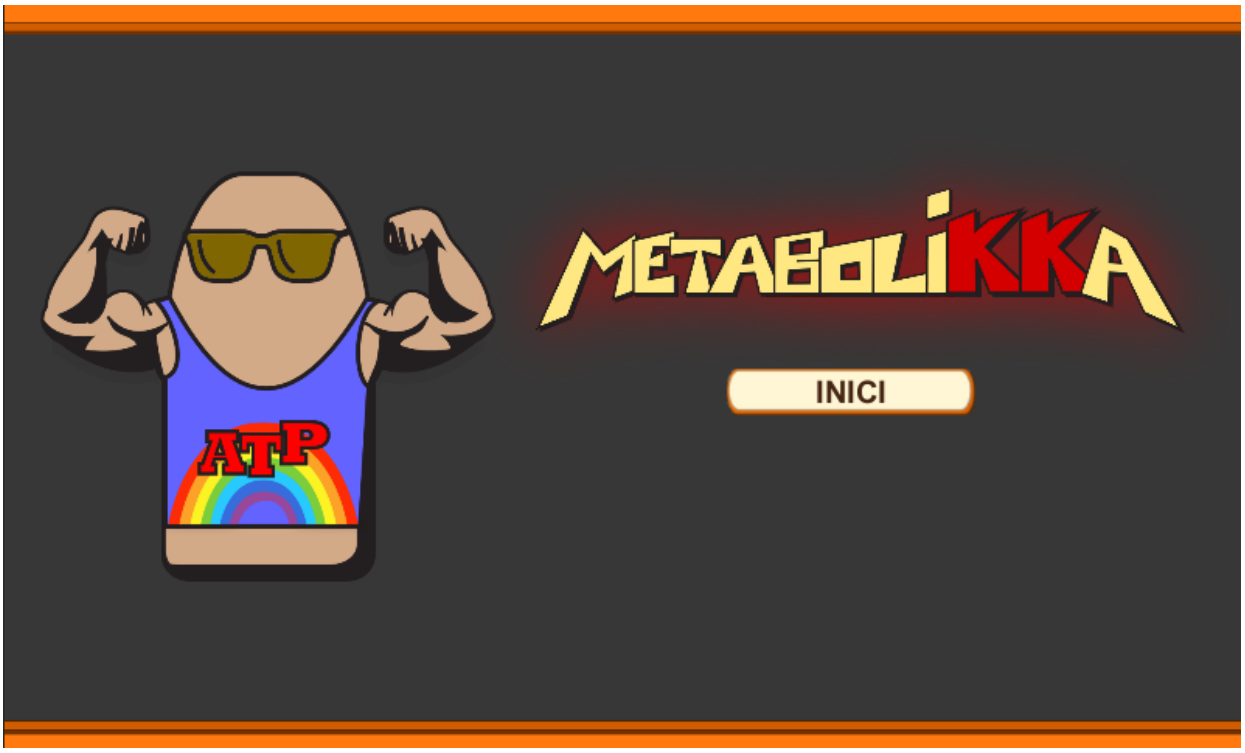


Figura 32: Pantalla inicial

5.6.2. Menú principal

Les partides del joc comencen sempre des del primer nivell, ja que busquem que el jugador hagi completat tots els nivells per assegurar-nos que ha vist tots els conceptes que explica el joc. Això fa que tinguem un únic botó per iniciar partida, i que no es pugui accedir als nivells de forma individual.

A part d'això, trobem un botó que ens deriva al menú d'opcions, i un altre que tanca la aplicació (Figura 33).

Figura 33: Menú principal

5.6.3. Menú opcions

Actualment el menú d'opcions en el projecte actua més com un botó de pausa. Al accedir-hi des de dins dels diversos nivells, la partida queda congelada i el jugador pot optar per anar al menú principal (perdent així la partida iniciada) o seguir endavant amb la partida (Figura 34).

Figura 34: Menú opcions obert primer des del menú principal, i després des de dins de partida

- El botó de Retorn ens torna a la partida o al menú principal segons on ens trobem.
- El botó idiomes permet la selecció d'aquest (actualment inactiu)

- El botó de configuració permet obrir un altre menú on canvia la resolució o la configuració del so (actualment inactiu)

En un futur aquest menú hauria d'incloure opcions per modificar el so, la resolució, l'idioma o el selector de dificultat, entre d'altres que es puguin necessitar.

5.6.4. Nivell

En aquest apartat enumerem i expliquem breument els diferents elements de la interfície gràfica que es troben durant la partida. Tot i que anteriorment hem hagut de fer distincions entre els dos primers nivells i el tercer, la interfície gràfica és comuna a tots 3 nivells. Això és a causa que la interfície va donant un resum de com va el procés bioquímic en general. Això també ajuda a la cohesió entre nivells.

- Comptador ATPs:

Aquest primer comptador ens mantindrà informats de quants ATPs hem aconseguit a mida que avança el la partida. Com a tal recordar que aquest comptador en dificultat normal canviarà segons ho fa el procés en la realitat d'una cèl·lula. Això vol dir que només baixarà o pujarà si després d'una fase hi ha hagut canvis, independentment de com hagi jugat el jugador. En difícil en canvi, el jugador podrà perdre'ls si interactua amb verins, retrocedeix fases, etc (Figura 35).

Figura 35: Comptador ATPs

- Comptador electrons:

Aquest comptador engloba els NADHs i FADHs que es van generant o gastant durant la partida. A l'igual que amb el comptador d'ATPs, aquest també modifica el comportament segons la dificultat (Figura 36).

Figura 36: Comptador electrons

- Cronòmetre:

El cronòmetre només està actiu en aquelles fases dels nivells 1 i 2 on les reaccions són reversibles. En la resta apareix el requadre en blanc. El temps l'hem decidit arbitràriament per donar-li jugabilitat als nivells. En la realitat aquesta reversibilitat no te un temps concret, i a vegades fins i tot pot no dependre del temps (Figura 37).

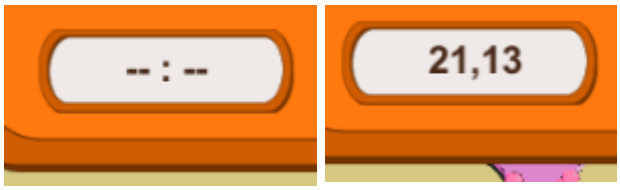


Figura 37: Cronòmetre

- Indicador de proximitat:

Per facilitar la tasca al jugador de trobar l'enzim, ja que aquest pot aparèixer en qualsevol zona del mapa, s'ha creat aquest indicador que simula el joc de nens on, a través de les indicacions "calent" o "fred", es marca la proximitat amb algun objecte. En el nostre cas, el termòmetre puja visualment de temperatura quan el jugador està a prop de l'enzim (Figura 38).

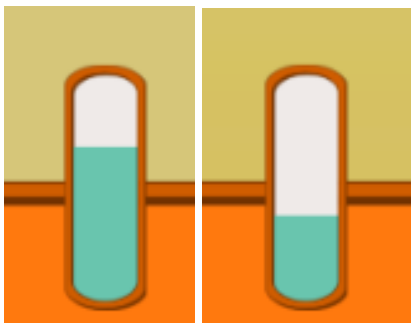


Figura 38: Indicador de proximitat

- Finestres d'ensenyament:

Aquestes finestres com la que veiem aquí ens ajuden a transmetre els coneixements. Aquestes apareixen durant els nivell 1 i 2 just després de cada fase, i en ella podem veure la molècula que estàvem jugant, l'enzim amb el que ha reaccionat, i la molècula en la que s'ha convertit. Les finestres ens ajuden perquè en elles podem incloure els noms de tots els elements, que en algun cas son molt llargs i no seria pràctic incloure'ls mentre juguem. A l'aparèixer aquestes pantalles el joc entra en pausa, per tal que el jugador les pugui examinar, i per continuar només cal clicar a la pantalla.

Al final del nivell 3 n'apareix una (la única d'aquest nivell), que ens mostra un resum de tots els processos vistos durant el joc, i en acaba mostrant una pantalla final amb el global de ATPs, NADHs i FADH gastats i produïts durant el joc.

5.7. Art del joc

Tots els elements artístics inclosos en aquest joc han estat creats per nosaltres mateixos, utilitzant el programa gratuït Inkscape. Aquest programa permet crear molts tipus de formes i dibuixos utilitzant línies vectorials, el que ens permet crear figures i elements tot i no tenir un especial talent pel dibuix.

Com estil gràfic hem optat per colors vius, que facilitessin els contrastos i que permetessin al jugador diferenciar uns elements dels altres. També s'ha intentat mantenir cert punt de realisme amb aquells elements que ho permetien, tot i que al trobar-nos a nivell molecular, molts elements tenen formes molt estranyes o fins i tot no definides, poden variar.

5.7.1. Escenaris

Tot i ser un projecte en 2D, amb els escenaris hem volgut donar certa sensació de profunditat. Per fer-ho hem jugat amb les capes i les sensacions de perspectives. Així doncs, en els nivells trobem fins a tres capes. El joc en sí succeeix a la més propera a la càmera, i ens les altres capes, els elements presents en l'escenari es van reduint de mida i difuminant-se una mica.

En els escenaris trobem un fons, sense cap mena de límits visibles, i diferents objectes que representen els orgànuls col·locats aleatòriament per donar més varietat a les partides.

5.7.2. Personatges / Molècules

En el joc serien les diferents molècules les que jugarien el paper de personatges. Tot i això nosaltres tenim les principals, aquelles que el jugador mourà i farà interaccionar amb els enzim, i les secundaries, que no juguen un paper tant important.

- Principals:

Per decidir com representar aquestes molècules s'han creat diferents esbossos, i junt amb el departament de Biologia s'ha triat la més escaient.

Per fer-les s'ha agafat la representació estructural en 2D i en 3D de cada molècula i s'ha fet una simplicació. Bàsicament s'ha creat un cos central amb una forma general de la molècula, i se l'hi ha afegit elements que representessin els diferents enllaços fora dels Carbonis. De fet són aquestes molècules les que era molt important fer representacions característiques per cada una d'elles, i com a tal cada molècula de cada fase té la seva representació (Figura 40).

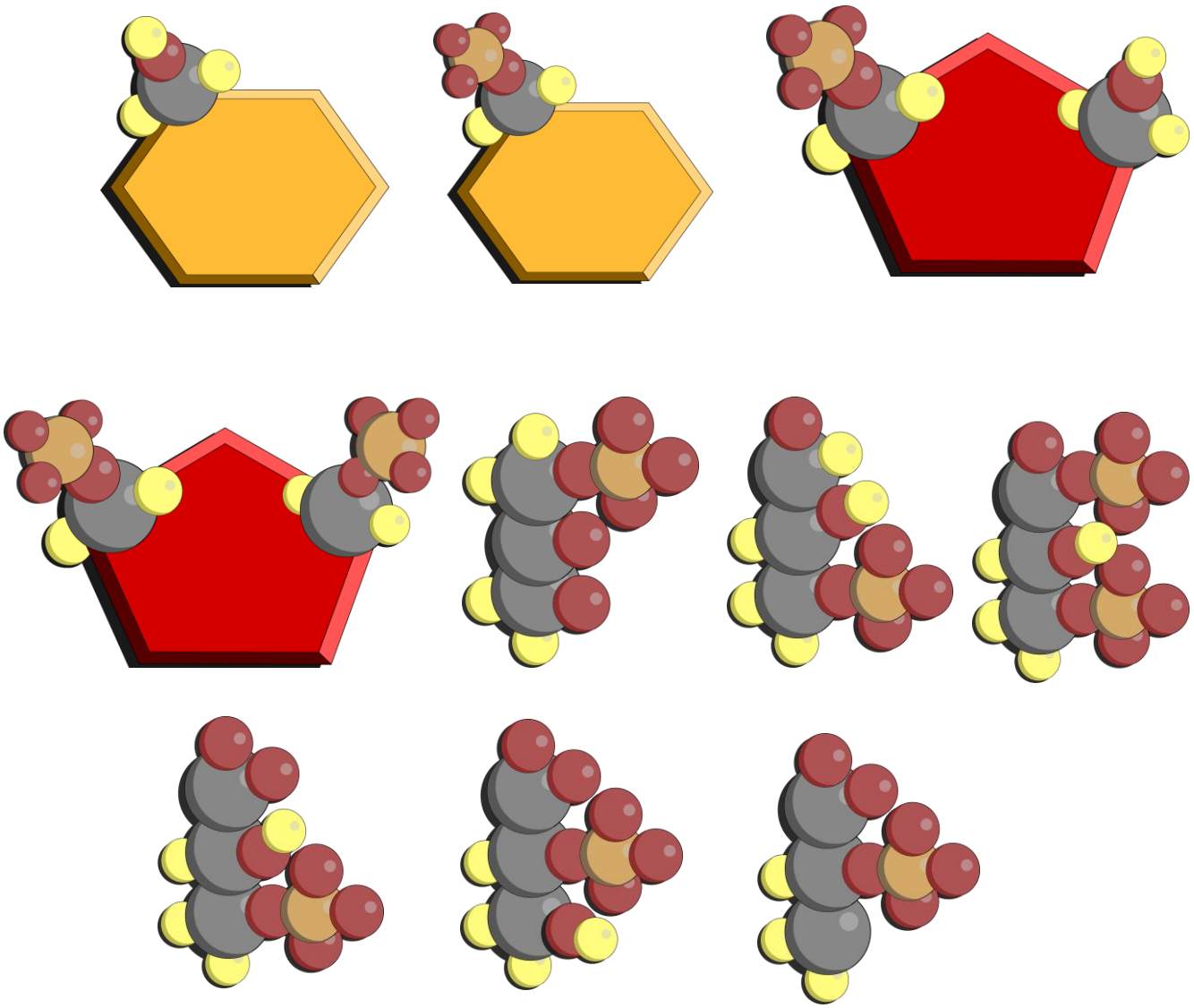


Figura 40: Seqüencia de les molècules de la glicòlisi segons van apareixent per fase

- Secundaries:

Aquestes són la resta de molècules que circulen per l'escenari dificultant la partida al jugador. Com en la realitat aquestes poden ser molt variades, s'ha fet un parell de representacions sense

buscar cap mena de similitud i buscant que es poguessin diferenciar clarament de les molècules principals (Figura 41).

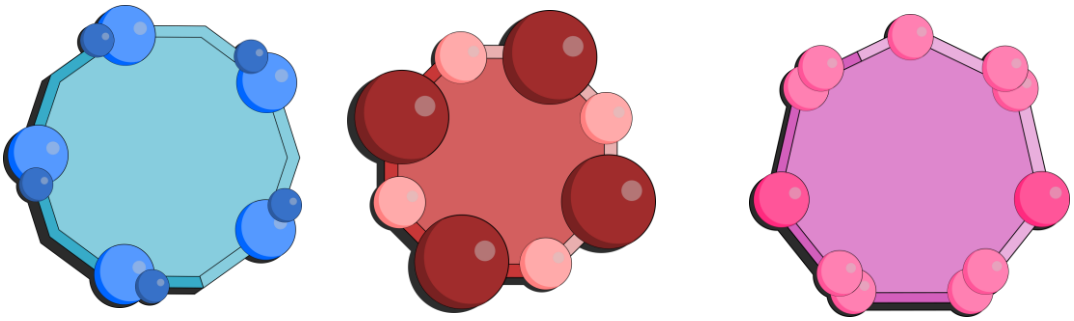


Figura 41: Disseny de les molècules secundaries

5.7.3. Altres Sprites

Aquí trobem els enzims. Igual que amb les molècules secundaries, s'ha creat una representació sense buscar cap mena de similitud i buscant que es poguessin diferenciar clarament de la resta d'elements. Sí s'ha mantingut com a element realista és la proporció de mida entre una molècula i un enzim, on aquests últims són molt més grans (Figura 42).

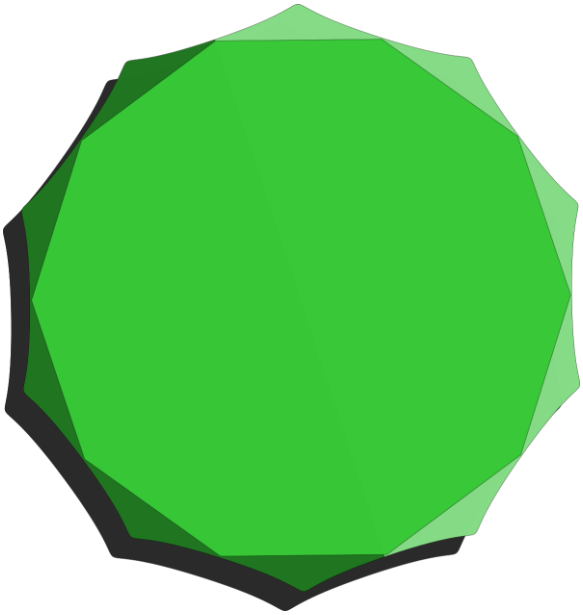


Figura 42: Disseny d'un enzim

5.7.4. Esbossos

Aquests són alguns dels esbossos fets per les molècules principals (Figura 43):

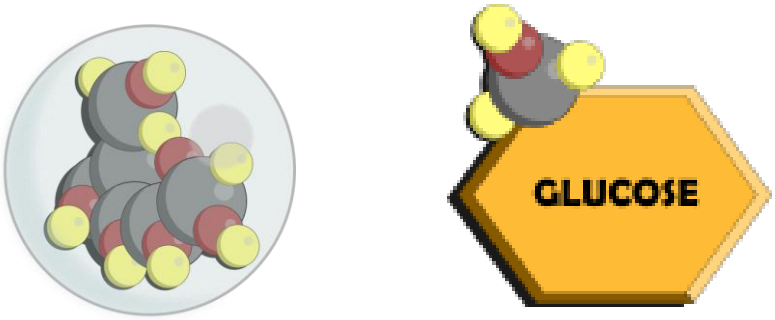


Figura 43: Esbossos descartats de la molècula principal

5.8. Efectes de so

El videojoc, com a mínim el prototip, no consta d'efectes sonors, ja que s'ha prioritzat altra feina abans. En tot cas, en no tenir les eines ni personal que tingui la capacitat per generar-lo, es disposaria a utilitzar sons i música de llicència oberta.

5.9. Canvis realitzats de disseny

Durant tot el desenvolupament del projecte s'han anat fent reunions d'equip conjuntes amb el departament de Biologia, que era qui havia encarregat el projecte. Com a tal, en aquestes reunions s'han anat fent molts canvis tant des del punt de vista de disseny com el del funcionament. De fet tot, el projecte s'ha anat creant per trams per tal que es poguessin anar fent correccions quan fos necessari. Tots els canvis realitzats durant aquestes reunions ja estan inclosos en els apartats anteriors, però a continuació enumerarem alguns dels importants i els justificarem.

a) Fins a trobar el disseny de les molècules principals del joc es va passar per varies iteracions.
b) Inicialment l'escenari tenia moltes menys molècules recorrent-lo, però es va voler incrementar l'ambient caòtic d'una cèl·lula normal
c) L'augment de dificultat no estava inclòs en el disseny inicial, així com la inclusió de verins
d) Els temporitzadors de les reaccions reversibles es van afegir posteriorment per tal de reflectir aquest fet i que el jugador pogués aprendre quines de elles ho eren. A més aquest va ser important perquè incloïa un repte més a la jugabilitat del joc

e) Durant les proves es va veure que era necessari incloure algun mètode per tal que el jugador pogués localitzar l'enzim més fàcilment. Es va optar per l'explicat anteriorment a causa que és fàcil d'implementar i no donava una indicació massa clara que acabes facilitant massa el joc

5.10. Pla de màrqueting

Com a últim apartat del disseny, mencionar algunes consideracions respecte al pla de màrqueting. Aquest projecte, al ser un encàrrec del departament de Biologia, el màrqueting és un aspecte poc important, i en tot cas, un cop finalitzat el projecte serà aquest departament el que s'haurà de plantejar si el vol comercialitzar o no. De moment ens han dit que serà una aplicació per ús propi de la Universitat de Girona.

El que si cobra més importància és la distribució, ja que hem de buscar una forma que els estudiants hi tinguin fàcil accés i sigui fàcil d'instal·lar. Tenint en compte que la nostra aplicació està orientada a dispositius mòbils, hi ha tres plataformes que ens serien de gran utilitat:

I. *Google Store*: aquesta és la principal plataforma de distribució i venda d'aplicacions de qualsevol dispositiu que utilitzi *Android* com a sistema operatiu. És de fàcil accés, esta inclosa en la majoria de dispositius per defecte, permet una fàcil instal·lació de les apps i permet al desenvolupador pujar-la per poder distribuir-la de forma gratuïta o marcant un preu, i anar-les actualitzant periòdicament.

II. *Apple Store*: equivalent a l'aplicació anterior però orientada a tots els dispositius que utilitzen sistema iOS. Normalment les aplicacions es fan compatibles pels dos sistemes i es pugen a ambdós botigues.

III. Itch.io: aquesta web és un mercat obert per creadors digitals independents, amb un especial focus en els videojocs. La plataforma permet a tothom vendre contingut creat, però també té l'opció que sigui completament gratuït. Tot i esta orientat a jocs per ordinador, també permetria descarregar-se un fitxer *apk* (instal·lador d'aplicacions pel mòbil) i fer la instal·lació manual al dispositiu que fos. Perd practicitat i no s'actualitzaria automàticament al modificar el fitxer, però és una opció interesant a considerar.

Totes 3 plataformes inclouen els mitjans per fer-ne publicitat, però això ho hauríem de gestionar segons els criteris que vulguin aplicar al departament de Biologia. Si la intenció d'ús és només entre alumnes, la publicitat seria totalment innecessària.

6. Implementació i proves

En aquest apartat veurem com s'han implementat tots els elements presents en el prototip final. Per la quantitat d'hores disponibles i tractant-se de una sola persona, hi ha molts elements que no s'han pogut implementar o no s'hagin finalitzat, però tots aquells elements incomplets no els inclourem aquí. Farem un repàs dels diversos Scripts que hem creat, però abans comentarem una mica la forma d'organitzar el projecte o altres pràctiques a l'hora de programar.

6.1. Programari i generalitats

Com s'ha comentat en apartats anteriors, el joc s'ha fet utilitzant *Unity*. Aquest és un motor de jocs multi plataforma creat per Unity Technologies. Està disponible per a Windows, Mac i Linux, i té suport de compilació per a més de 20 plataformes, entre les quals es troben PC, les consoles principals, dispositius mòbils i *WebGL*. Es basa en Mono, un entorn de desenvolupament integrat lliure i gratuït, dissenyat primordialment per al llenguatge C#. Compta amb eines i components per a crear jocs 3D i 2D.

En el nostre cas, l'hem utilitzat configurat per jocs 2D i utilitzant les SDK *d'Android* (les llibreries necessàries per compilar qualsevol projecte per *Android*) per fer les proves. Per realitzar aquestes proves, Unity ha creat una aplicació, *Unity Remote* 5, que un cop instal·lada en el mòbil o tableta, connectant el dispositiu amb l'ordinador, el propi motor el pot detectar i compilar les execucions directament allà. Les execucions s'han de configurar per tal que els gràfics no siguin els òptims (Figura 44) o sinó baixa molt el rendiment, però segueix sent molt pràctic per fer proves ràpides sense haver de crear una Apk (instal·lador d'una app) i instal·lar el projecte manualment cada vegada.

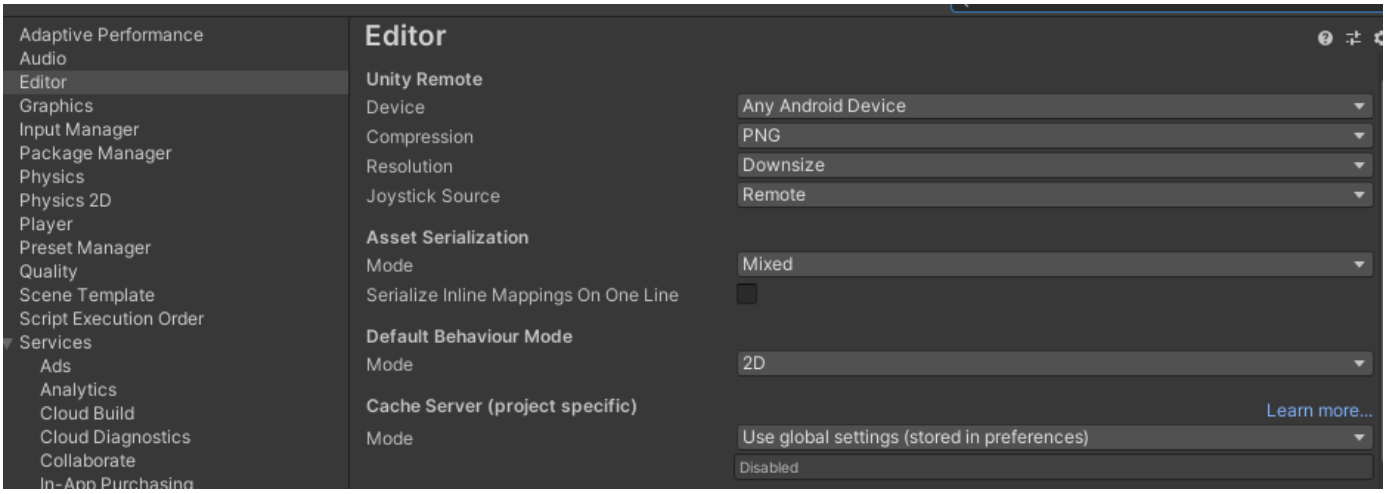
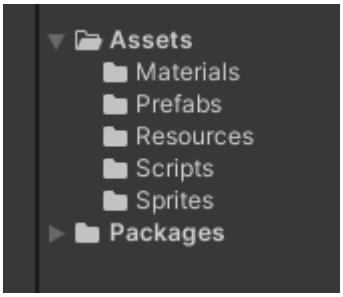


Figura 44: Opcions del editor de Unity on configurar el Unity Remote

El dispositiu utilitzat per fer les proves ha estat un OnePlus 6t amb un Qualcomm® Snapdragon™ 845, GPU Adreno 630, i 8 Gb de memòria RAM.

6.1.1. Estructura general del projecte

El contingut del projecte s'ha dividit en unes carpetes bàsiques que ens permetin tenir els assets segons cada tipus. Aquesta organització ja la implementa Unity en els projectes, i tot i que es pot modificar sense problema. En el nostre cas ja anava en sintonia amb els nostres interessos. Cada subcarpeta però te altres subcategories per tal de tenir-ho millor estructura.



6.2. *Scripts*

Tot i que aquí veurem quines són les funcions més rellevants de cada Script i quina finalitat tenen, la totalitat del codi està en el Annex 2 .

- Camera.cs

Fitxer molt petit però amb una funció molt efectiva. Quan inicialment vam aplicar la càmera sobre les molècules que movíem, vam veure que el moviment era molt rígid, i fins i tot, incòmode. Així doncs hem creat aquest script que conté una funció que fa reaccionar la càmera més lentament, donant una sensació de fluïdesa que queda molt bé dins del mon del joc.

- Paràmetres:
  - ***public float*** lerp; -> aquest valor ens permet configurar amb quan de retràs avança la càmera, sent 1 el valor que la fa instantània.
- Funcions**:**
  - ***void*** FixedUpdate(); -> aquesta funció crea una interpolació entre la posició de la molècula i la posició actual de la càmera a partir del lerp que l'hi diem, i avança la càmera utilitzant aquesta vector creat. El contingut d'aquesta funció es crida a cada frame.

```
void FixedUpdate()
{

    Vector2 pos = Vector2.Lerp((Vector2)transform.position,
    (Vector2)player.transform.position, lerp);
    transform.position = new Vector3(pos.x, pos.y, transform.position.z);

}
```

- Controls.cs

Aquest fitxer inclou el codi dels controls que fem servir per moure la molècula principal tant al nivell 1 com al nivell 2. En aquest script es detecta quan el jugador toca la pantalla per tal de realitzar algun moviment, agafa el vector creat pel jugador i la intensitat, i la trasllada a la molècula en forma de força. A part, també crea un efecte de fregament per tal que lo molècula vagi frenant si no detecta inputs.

- Paràmetres:
  - ***public float*** speed; -> multiplicador aplicat a la magnitud del vector introduït pel jugador.
  - ***public float*** brake; -> valor que marca la intensitat en la que la molècula redueix velocitat.
  - ***public float*** maxSpeed; -> valor que marca la velocitat màxima que tindrà la molècula, independentment de la quantitat de inputs.
  - ***private*** Rigidbody2D rb2D; -> agafa el Rigidbody que ja té la molècula per poder modificar les forces aplicades a aquest.
- Funcions:
  - ***void*** Update(); -> quan aquesta funció detecta input a la pantalla, el trasllada a la molècula en forma de força. Si no, li aplica fregament per frenar-la.

```
void Update()
{
    if (Input.touchCount > 0)
    {
        Touch touch = Input.GetTouch(0);
        if (touch.phase == TouchPhase.Began)
        {
            iniPos = touch.position;
        }

        if (touch.phase == TouchPhase.Ended)
        {
            // Get movement of the finger since last frame
            endPos = touch.position;
            // Move object across XY plane
           Vector2 direccio = endPos - iniPos;

            rb2D.AddForce(direccio * (direccio.magnitude * speed));
        }
    }

    Vector2 vel = rb2D.velocity;

    if (vel.magnitude > 0)
    {
        rb2D.AddForce(-vel/brake);

        if (vel.x > maxSpeed) rb2D.velocity = new Vector2(maxSpeed,
        rb2D.velocity.y);
```

```
            else if (vel.x < -maxSpeed) rb2D.velocity = new Vector2(-maxSpeed,
            rb2D.velocity.y);
            else if (vel.x < 0.1 && vel.x > -0.1) rb2D.velocity = new Vector2(0,
            rb2D.velocity.y);

            if (vel.y > maxSpeed) rb2D.velocity = new Vector2(rb2D.velocity.x,
            maxSpeed);
            else if (vel.y < -maxSpeed) rb2D.velocity = new Vector2(rb2D.velocity.x, -
            maxSpeed);
            else if (vel.y < 0.1 && vel.y > -0.1) rb2D.velocity = new
            Vector2(rb2D.velocity.x, 0);
        }

    }
```

- EnzymeCollision.cs

Fitxer que determina el que passa quan la molècula principal colisiona amb l'enzim, que bàsicament és que canvia de fase.

  - ***void*** Update(); -> quan aquesta funció detecta input a la pantalla, el trasllada a la molècula en forma de força. Si no, li aplica fregament per frenar-la.

```
private void OnCollisionEnter2D(Collision2D collision)
{
  if (collision.gameObject.name == "Player") levelscr.changeLevel(levelscr.level + 1);
}
```

- LevelFunction.cs

Aquest fitxer tant serveix pel nivell 1 com pel nivell 2, i genera i controla cada escenari. Inicialitza la molècula principal dins de la zona segura, genera l'enzim en una posició aleatòria, genera la resta de molècules quan així es necessita, i marca en quina fase de cada nivell estem, canviant els Sprites de la molècula principal, activant si fa falta el cronòmetre, o mostrant la pantalla d'informació després de completar cada fase. També defineix els límits que tindrà el mapa.

- Paràmetres:
  - ***public int*** level; -> indica a quina fase estem dins de cada nivell.
  - ***public int*** numLvlx; -> indica quina numero de sprite toca per la molècula principal.
  - ***public float*** tLvlx; -> indica el temps per completar aquelles fases on les reaccions són reversibles.
  - ***public float*** maxPox_x / maxPos_y; -> indica els límits del mapa. Ho fa amb un distancia màxima en cada eix.

- ***public float*** minDistEnz;-> quan l'enzim es genera aleatòriament, aquesta serà la distància mínima al jugador.
- ***Private bool*** timeTrial; -> "true" quan estem en una fase amb cronòmetre.
- ***Private bool*** safe; -> "true" la molècula principal no ha sortit de l'àrea segura.
- ***Private*** List<GameObject> moleList = new List<GameObject>(); -> crea la llista de molècules que després estaran per l'escenari rebotant i dificultant el moviment al jugador.
- ***Private*** BoxCollider2D view; -> delimita la zona segura on inicia la partida.

- Funcions
  - ***void*** Start(); -> inicialitza l'escenari col·locant al jugador, creant l'enzim i inicialitzant variables.
  - ***void*** Update(); -> realitza la gestió de les molècules secundaries (les destrueix quan surten de l'escenari i en crea de noves) i controla el temps contrarellotge si fa falta.
  - ***void*** OnTriggerExit2D(Collider2D other)(); -> detecta quan el jugador abandona la zona segura i canvia el boleà pertinent.
  - ***Public void*** changeLevel (int lvl) (); -> rep la fase actual i reinicia la molècula principal segons això.
  - ***void*** safeArea(); -> quan canvia de fase i el jugador està en zona segura. Aquesta funció elimina totes les molècules secundaries que puguin quedar
  - ***void*** newEnzyme(); -> destrueix l'enzim anterior i en crea un de nou en una nova posició.

- PositionInMap.cs

Controla en quina posició es troba el jugador (molècula principal) i obtenint els límits del mapa, teletransporta el jugador d'un costat a l'altre quan fa falta juntament amb la càmera. Aquest script és el que ens ajuda a simular que mapa no té límits.

- Paràmetres:
  - ***public float*** maxPox_x / maxPos_y; -> indica els límits del mapa. Ho fa amb un distancia màxima en cada eix.
- Funcions:
  - ***void*** Update(); -> quan el jugador surt d'algun dels límits del mapa, el jugador i la càmera són transportats al costat contrari.

```
void Update () {

    float dist_x = Mathf.Abs(transform.position.x - cam.transform.position.x);
    float dist_y = Mathf.Abs(transform.position.y - cam.transform.position.y);
```

```
    if (transform.position.x > maxPos_x)
    {
        transform.position = new Vector2(-maxPos_x,transform.position.y);
        cam.transform.position = new Vector3(-maxPos_x - dist_x,
        cam.transform.position.y, cam.transform.position.z);
    }
    else if (transform.position.x < -maxPos_x)
    {
        transform.position = new Vector2(maxPos_x, transform.position.y);
        cam.transform.position = new Vector3(maxPos_x + dist_x,
        cam.transform.position.y, cam.transform.position.z);
    }
    if (transform.position.y > maxPos_y)
    {
        transform.position = new Vector2(transform.position.x, -maxPos_y);
        cam.transform.position = new Vector3(cam.transform.position.x, -maxPos_y -
        dist_y,  cam.transform.position.z);
    }
    else if (transform.position.y < -maxPos_y)
    {
        transform.position = new Vector2(transform.position.x, maxPos_y);
        cam.transform.position = new Vector3(cam.transform.position.x, maxPos_y +
        dist_y, cam.transform.position.z);
    }
}
```

- Projectile.cs

Script utilitzat per controlar el comportament de les molècules secundaries que hi ha en els dos primers nivells. Aquest fitxer crea, inicialitza i li dona una velocitat i direcció aleatòries. També les destrueix quan aquestes ja han xocat i han perdut la inèrcia, sempre que el jugador no les tingui en el camp de visió.

- Paràmetres:
  - ***public float*** minSpeed/maxSpeed; -> indiquen quin rang de velocitat poden tenir aquestes molècules.
  - ***public float*** temps; -> indica el temps mínim que passa abans no es destrueixen un cap ja han xocat.
  - ***public bool*** destroy; -> “true” quan les molècules poden ser destruïdes.
  - ***public*** Sprite ene1, ene2, ene3; -> són possibles aparences per aquestes molècules.
  - ***Private float*** speed; -> velocitat actual a la que es mouen.
  - ***Private bool*** colided; -> “true” quan les molècules han xocat amb algun element.
- Funcions
  - ***void*** Start(); -> inicialitza la molècula secundaria, atribuint-li també una velocitat i direcció aleatories.

- ***void*** Update(); -> duu un control de les possibles col·lisions que hagin pogut patir, i les prepara per a la destrucció.

- ProjectilePosInMap.cs

Semblant al fitxer que controla la posició del jugador i la càmera en el mapa. Aquest ho fa per les molècules secundaries, transportant-les d'un costat a l'altre.

- Paràmetres:
  - ***public float*** maxPox_x / maxPos_y; -> indica els límits del mapa. Ho fa amb un distancia màxima en cada eix.
- Funcions:
  - ***void*** Update(); -> quan una partícula secundària surt d'algun dels límits del mapa, aquesta és transportada al costat contrari.

- ButtonHandlerIni.cs/ButtonHandler

Ambdós fitxers contenen aquelles funcions que s'utilitzen en tots els botons que hi ha en el projecte. Són totes elles funcions públiques que s'utilitzen "onClick()". En aquest cas, el primer fitxer correspon als botons de la primera escena (pantalla principal), mentre que el segon cobreix la segona escena (nivell 1).

- ***Públic void*** botoInici(); -> surt de la portada per obrir el menú d'inici.
- ***Públic void*** botoIniciaPartida (); -> canvia d'escena per començar el primer nivell.
- ***Públic void*** botoSortir (); -> tanca la aplicació.
- ***Públic void*** obrirMenuOpc (); -> obre el menú d'opcions.
- ***Públic void*** retornPartida (); -> torna al menú principal.

- ***Públic void*** sortirMenuPral (); -> pausa la partida i demana confirmació per sortir al menú principal.
- ***Públic void*** cancelaSortir (); -> cancel·la la sortida i reanuda la partida.
- ***Públic void*** acceptaSortir (); -> canvia d'escena per tornar a la del menú d'inici.
- ***Públic void*** obrirMenuOpc (); -> obre el menú d'opcions.
- ***Públic void*** retornPartida (); -> tanca el menú d'opcions i reanuda la partida.

6.3. Proves

Durant el desenvolupament del joc hem anat realitzant de forma contínua exhaustives proves per a comprovar el correcte funcionament de les mecàniques que anàvem implementant. A més, s'han anat prenent decisions i canvis de disseny en les diferents reunions d'equip segons la sensació que ens provocava el joc. Aquests canvis van des de l' incorporació d'elements com els cronòmetres, fins a decisions merament estètiques com afegir més elements al escenari per crear la sensació de caos.

Per a realitzar les proves principalment hem utilitzat la aplicació Unity Remote 5, que, tal com s'ha explicat abans, permet executar el projecte en un dispositiu mòbil sense necessitat d'instal·lar-lo prèviament, facilitant molt tot el procés.

7. Resultats

En aquest apartat valorarem els resultats obtinguts partint dels objectius plantejat a l'inici d'aquest projecte. Valorarem tant el resultat obtingut com fins a quin punt compleixen els paràmetres que volíem aconseguir.

Abans de començar però, recordar que el projecte en global és un encàrrec del departament de Biologia de la UdG per tal de crear un joc el més acabat possible que serveixi com a *Serious Game* per formar als estudiants de diverses carreres d'aquesta facultat com fabrica energia la cèl·lula eucariota. A partir d'aquesta entrada és com s'ha començat el projecte, dividint-lo en els diferents apartats, fins assolir els resultats següents en cadascun dels objectius inicials:

- Entendre els conceptes a explicar: en aquest punt comptàvem amb l'avantatge de tenir cert bagatge en bioquímica d'estudis anteriors. Això, junt amb l'ajuda de llibres especialitzats i certes webs, s'ha acabat entenent els conceptes lo suficientment bé com per poder explicar-los nosaltres mateixos. Cal destacar, però, que arribar a aquesta comprensió dels conceptes no ha reduït la seva complexitat a l'hora de transportar-los a mecàniques i que aquestes compleixin a més els paràmetres presentats pel departament de Biologia. Ha estat a causa d'aquesta complexitat també que hem dedicat tot un capítol d'aquest treball a resumir el possible tots aquests conceptes, i tot i això s'ha prolongat més de l'esperat.

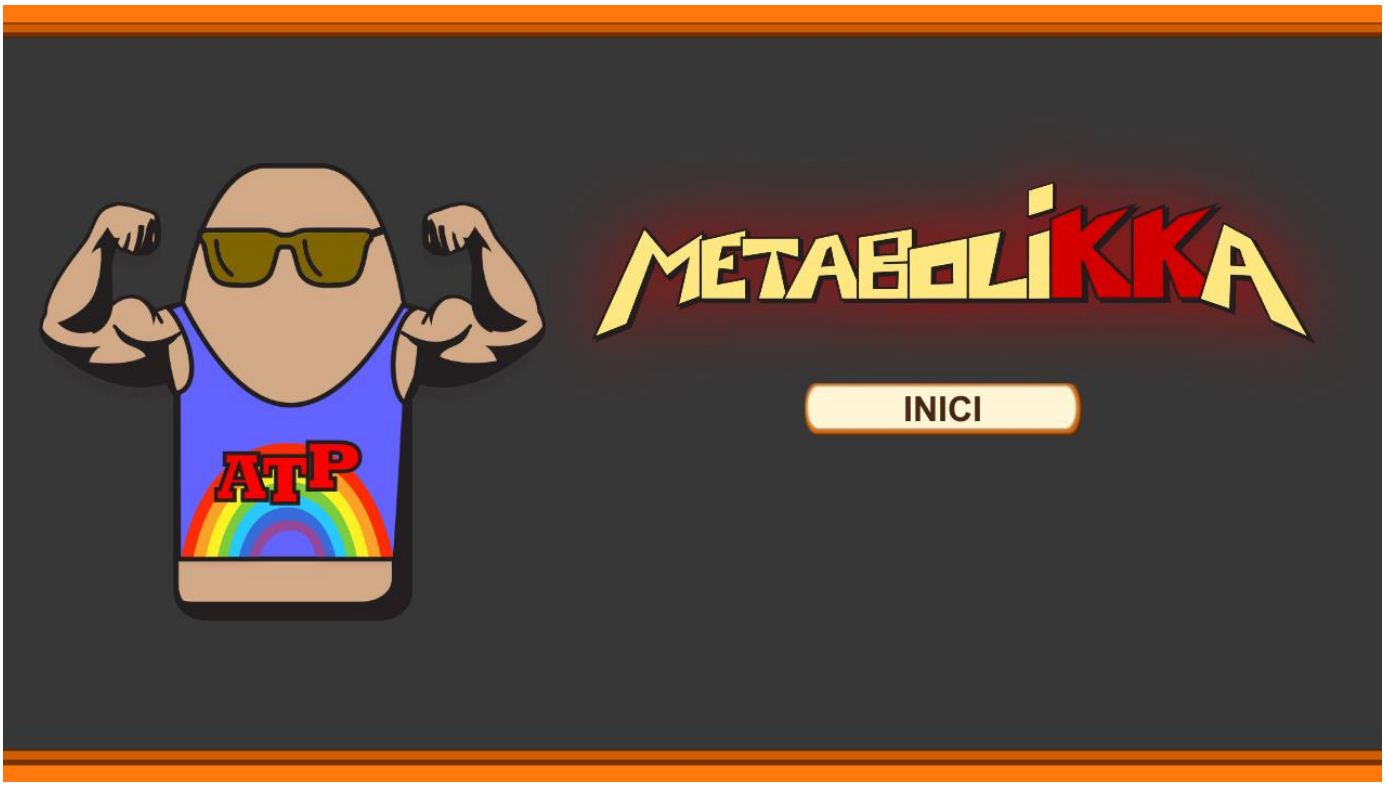
- Dissenyar el joc: s'ha pogut crear un disseny conceptual complet de tot el joc, marcant uns criteris i característiques claus que volem per cada apartat. Certament aquest disseny ha implicat varies iteracions i reunions amb els diferents membres implicats en el projecte, però les mecàniques pensades són diferents i divertides, i s'adapten el suficient als requeriments que teníem des de l'inici, i la forma de transmetre els coneixements creiem que pot funcionar. Amb tot, el disseny final està prou pensat i detallat per poder traduir-se a un joc que s'acostés molt a un producte final.

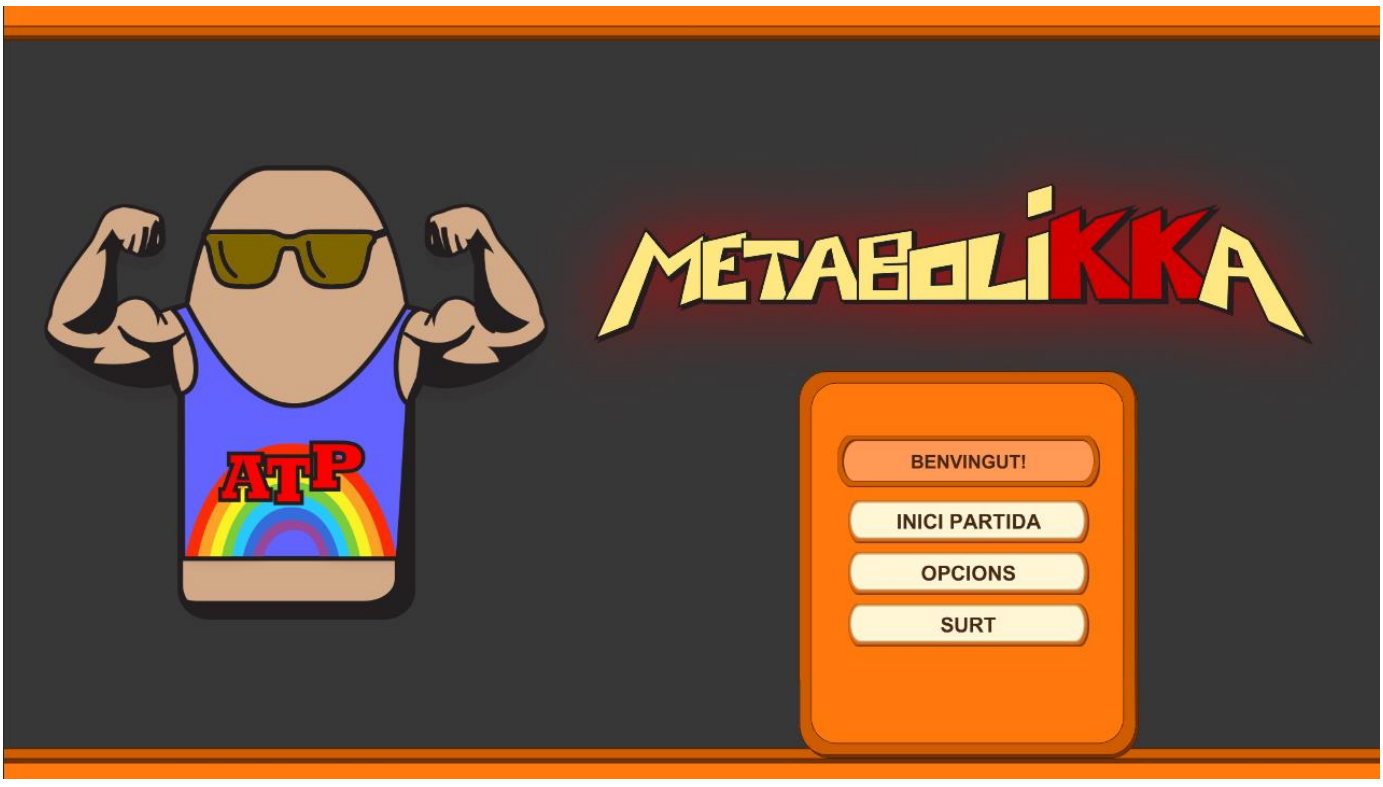
- Implementació del joc: aquest és el punt que ha quedat més endarrerit dins del projecte. Durant el temps que teníem i considerant que tots les elements de la implementació els ha creat una sola persona, en el prototip final hem aconseguit el següent:

    - S'ha dissenyat completament el primer nivell dels 3 que es volien, amb tots els elements necessaris per a que sigui totalment jugable tant visuals com de programació.
    - El segon nivell, al compartir moltes característiques amb el primer, s'ha pogut desenvolupar a nivell de codi, però no hi ha hagut temps per recrear els *assets* visuals.

- Pel que fa al tercer i últim nivell, s'han fet proves de codi i execució amb la mecànica principal, però sense que aquesta hagi quedat del tot enllestida.
- S'han creat els diferents menús i la pantalla principal, encara que amb un disseny provisional.

A continuació s'adjunten captures de pantalla de com podria ser una partida amb el contingut ja creat (Figura 45):






-- : --
0
0


-- : --
0
0

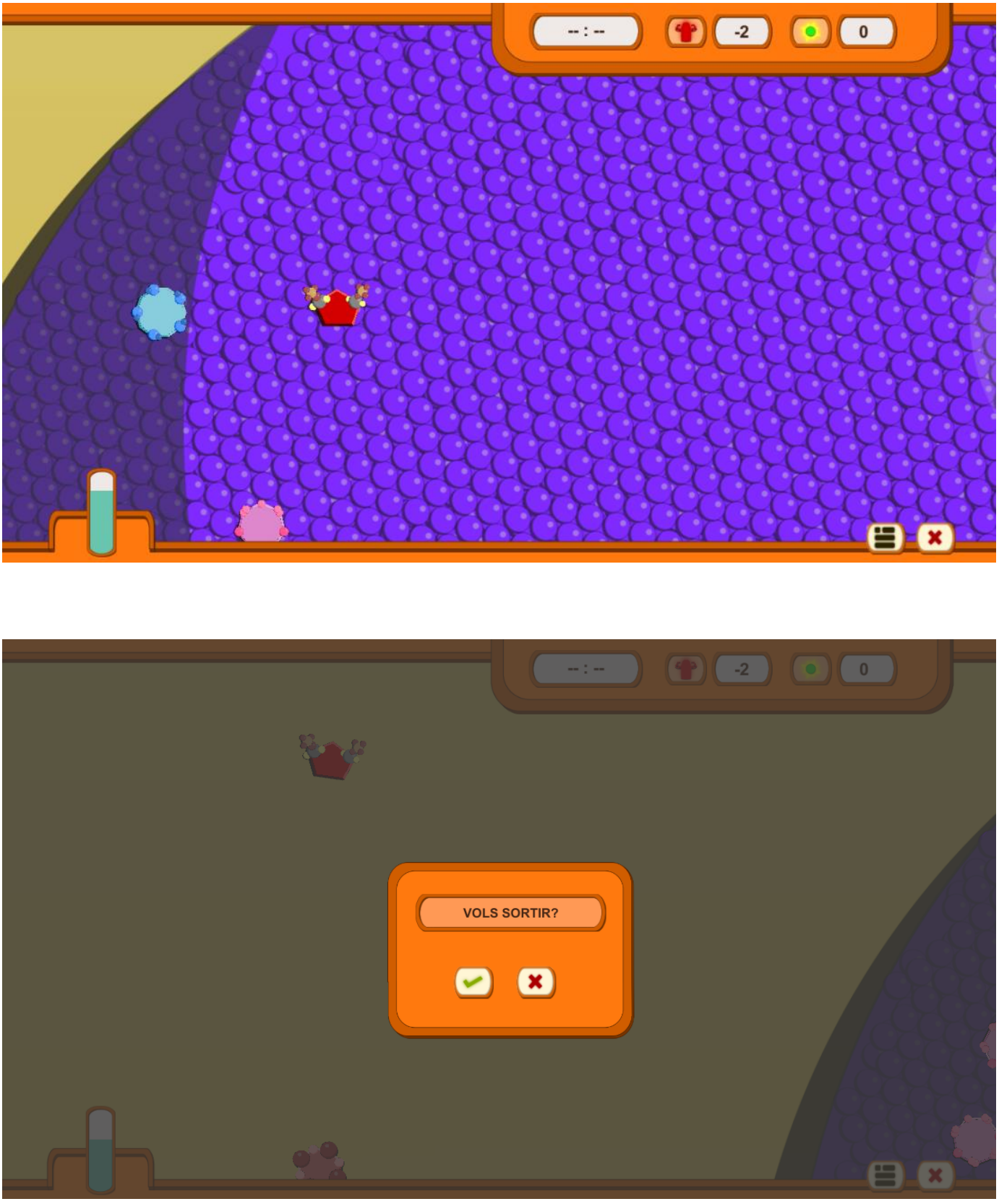


Figura 45: Cronologia en captures de pantalla d'una possible partida.

8. Conclusions

Aquest projecte volia crear un primer prototip que permetés tenir una idea clara de com han de ser tots els apartats del joc. Tanmateix, el producte final ha quedat lluny de ser un prototip complet. El projecte ha permès veure que tot i ser en aparença petit, amb només 3 nivells jugables, és molt fàcil que la feina sigui excessiva per a una sola persona, com era el cas. El executable té un primer nivell totalment jugable i a partir del disseny ja creat es podria seguir amb el projecte, així que creiem que en conjunt s'ha avançat molta feina tenint en compte la quantitat d'hores invertides.

El projecte també ha permès conèixer de primera mà el que representa treballar per tercers, en lloc de fer un projecte totalment personal. Tot i que des del departament de Biologia no haguessin pot posar més facilitats, treballar per externs implica fer reunions continues per valorar si la feina feta compleix el que es demana, implica modificacions inesperades que poden obligar a modificar gran part del projecte, o implica tenir requeriments amb els quals un, com a dissenyador, pot no estar del tot d'acord, però que s'ha d'adaptar igualment.

Tot i això, valorant la feina feta, el disseny conceptual del joc ha quedat molt complet i podria perfectament traduir-se en un projecte viable de tenir més temps i recursos. També estic content de com he aconseguit implementar algunes mecàniques. De fet, aquest era el primer projecte que he implementat per Unity orientat o dispositius Android, i això m'ha permès aprendre molt sobre com compilar i executar aquests tipus d'aplicacions, o sobre com utilitzar controlador poc habituals com pot ser la pantalla tàctil o el oscil·loscopi d'un telèfon.

Així doncs, tot i no assolir completament els objectius inicials, estic content i orgullós per la feina aconseguida, tant de disseny, d'implementació, com de la vessant artística. Tot i que ser un equip d'una sola persona pot semblar un inconvenient, en el meu cas m'ha permès afrontar problemes des de tots els àmbits del projecte, poden aprofundir en la utilització de *Unity* en programació o *d'Inkscape* en disseny. A part he pogut aplicar molts dels coneixements apressos durant la carrera, que m'han ajudat a solucionar molts imprevistos, donant fe de la pràctica guanyada en aquests anys.

9. Treball Futur

Tot i l'estat del prototip final, creiem que la feina a fer no es tanta i es limita a dur a terme tot el contingut del disseny conceptual del joc. El disseny està complet i prou refinat per poder dur a terme un prototip molt més acabat que pugui acostar-se molt a un disseny definitiu. Entre les tasques a fer hi ha:

I. Finalitzar el nivell 2 i implementar el nivell 3 en la seva totalitat. Això implica crear mecàniques i polir elements que hagin pogut passar per alt en el disseny conceptual.

II. Ajudar-se d'un artista en la creació dels assets visuals per aconseguir el realisme que es buscava inicialment. Certament, tenir uns *sprites* amb més detalls de segons quins elements podria ajudar favorablement a transmetre els coneixements.

III. Organitzar tests de la aplicació entre alumnes als que pugui anar destinat aquest joc. Aquests tests han de tenir unes preguntes d'avaluació final que ens permetin saber en quin grau el joc ensenya el funcionament dels 3 processos bioquímics implementats.

IV. Adaptar-lo a diferents idiomes ajudaria que el joc el poguessin utilitzar alumnes d'altres universitats.

10. Bibliografia

i. David L. Nelson , Michael M. Cox, (2017) *Principles of Biochemistry*, W. H. Freeman; Seventh Edition.

ii. Khan Academy, *Pasos de la respiración celular*, https://es.khanacademy.org/science/ap-biology/cellular-energetics/cellular-respiration-ap/a/steps-of-cellular-respiration

iii. Unity Technologies, *Unity Manual,* https://docs.unity3d.com/Manual/Unity2D.html

iv. Unity Technologies, *Unity Blog,* https://blogs.unity3d.com/

v. Brackeys, *Youtuber*, https://www.youtube.com/user/Brackeys

vi. Wikipedia Foundation. *Wikipedia, the free encyclopedia,* https://www.wikipedia.org

vii. Imma Boada, Núria Puig*, Disseny Conceptual de Videojocs*, Assignatura de GDDV, Universitat de Girona

11. Annexos

11.1. ANNEX 1: Carpeta del projecte

Com a annex, hem adjuntat un fitxer “txt” que conté un enllaç de Google Drive. Aquest permet descarregar el projecte sencer de Unity i els diagrames de flux de cada uns dels bosses en el fitxer que conté aquesta memòria. Dins de la carpeta del projecta es poden veure tots els assets utilitzats, des dels sprites fins als arxius de codi. Per accedir als sprites, cal anar a “MetabolikkaDemo/Assets/Sprites/”, mentre que pel codi és “MetabolikkaDemo /Assets/Scripts”.

11.2. ANNEX 1: Fitxers de codi

A continuació mostrem el codi integra del nostre projecte, mostrant tots els fitxers per ordre alfabètic.

11.2.1. ButtonHandler.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class ButtonHandler : MonoBehaviour
{
    private GameObject pantPausa;
    private GameObject menuPausa;
    private GameObject menuOpcions;

    // Start is called before the first frame update
    void Start()
    {
        pantPausa = GameObject.Find("PantallaPausa");
        menuPausa = GameObject.Find("MenuSegur");
        menuOpcions = GameObject.Find("MenuOpcions");
        pantPausa.SetActive(false);
        menuPausa.SetActive(false);
        menuOpcions.SetActive(false);
    }

    public void sortirMenuPral()
    {
        Time.timeScale = 0f;
        pantPausa.SetActive(true);
        menuPausa.SetActive(true);
    }

    public void cancelaSortir()
    {
        Time.timeScale = 1f;
        pantPausa.SetActive(false);
```

```
        menuPausa.SetActive(false);
    }

    public void acceptaSortir()
    {
        Time.timeScale = 1f;
        SceneManager.LoadScene(0);
    }

    public void obrirMenuOpc()
    {
        Time.timeScale = 0f;
        pantPausa.SetActive(true);
        menuOpcions.SetActive(true);
    }

    public void retornPartida()
    {
        Time.timeScale = 1f;
        pantPausa.SetActive(false);
        menuOpcions.SetActive(false);
    }
}
```

11.2.2. ButtonHandlerIni.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class ButtonHandlerIni : MonoBehaviour
{
    private GameObject menuIni;
    private GameObject menuOpcions;

    // Start is called before the first frame update
    void Start()
    {
        menuIni = GameObject.Find("MenuOpcionsIni");
        menuOpcions = GameObject.Find("MenuOpcions");
        menuIni.SetActive(false);
        menuOpcions.SetActive(false);
    }

    public void botoInici()
    {
        menuIni.SetActive(true);
    }

    public void botoIniciaPartida()
    {
        menuIni.SetActive(false);
        SceneManager.LoadScene(1);
    }

    public void botoSortir()
    {
        Application.Quit();
    }
```

```
    public void obrirMenuOpc()
    {
        menuIni.SetActive(false);
        menuOpcions.SetActive(true);
    }

    public void retornPartida()
    {
        menuOpcions.SetActive(false);
        menuIni.SetActive(true);
    }
}
```

11.2.3. Camera.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Camera : MonoBehaviour
{
    public float lerp = 0.5f;
    private GameObject player;

    // Use this for initialization
    void Start()
    {
        player = GameObject.Find("Player");
    }

    // LateUpdate is called after Update each frame
    void FixedUpdate()
    {

        Vector2 pos = Vector2.Lerp((Vector2)transform.position,
(Vector2)player.transform.position, lerp);
        transform.position = new Vector3(pos.x, pos.y, transform.position.z);

    }
}
```

11.2.4. Controls.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Controls : MonoBehaviour {

    public float speed = 0.1F;
    public float brake = 5f;
    public float maxSpeed = 3f;
    private Vector2 iniPos;
    private Vector2 endPos;
    private Rigidbody2D rb2D;

    private void Start()
```

```
    {
        rb2D = this.gameObject.GetComponent<Rigidbody2D>();
    }

    void Update()
    {
        if (Input.touchCount > 0)
        {
            Touch touch = Input.GetTouch(0);
            if (touch.phase == TouchPhase.Began)
            {
                iniPos = touch.position;
            }

            if (touch.phase == TouchPhase.Ended)
            {
                // Get movement of the finger since last frame
                endPos = touch.position;
                // Move object across XY plane
               Vector2 direccio = endPos - iniPos;

                rb2D.AddForce(direccio * (direccio.magnitude * speed));
            }
        }

        Vector2 vel = rb2D.velocity;

        if (vel.magnitude > 0)
        {
            rb2D.AddForce(-vel/brake);

            if (vel.x > maxSpeed) rb2D.velocity = new Vector2(maxSpeed, rb2D.velocity.y);
            else if (vel.x < -maxSpeed) rb2D.velocity = new Vector2(-maxSpeed, rb2D.velocity.y);
            else if (vel.x < 0.1 && vel.x > -0.1) rb2D.velocity = new Vector2(0, rb2D.velocity.y);

            if (vel.y > maxSpeed) rb2D.velocity = new Vector2(rb2D.velocity.x, maxSpeed);
            else if (vel.y < -maxSpeed) rb2D.velocity = new Vector2(rb2D.velocity.x, -maxSpeed);
            else if (vel.y < 0.1 && vel.y > -0.1) rb2D.velocity = new Vector2(rb2D.velocity.x, 0);
        }

    }
}
```

11.2.5. EnzymCollision.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class EzymeCollision : MonoBehaviour {

    private GameObject lvl;
    private LevelFunction levelscr;

    // Use this for initialization
    void Start () {
        lvl = GameObject.Find("Level");
        levelscr = lvl.GetComponent<LevelFunction>();
    }

    // Update is called once per frame
```

```
        void Update () {

        }

    private void OnCollisionEnter2D(Collision2D collision)
    {
        if (collision.gameObject.name == "Player") levelscr.changeLevel(levelscr.level + 1);
    }
}
```

11.2.6. LevelFunction.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class LevelFunction : MonoBehaviour {

    public int level = 0;
    public int numLvL1 = 25;
    public int numLvL2 = 30;
    public int numLvL3 = 40;
    public int numMolBack = 40;
    private float tLvl3 = 30;
    private float tLvl6 = 25;
    private float tLvl7 = 20;
    private float tLvl8 = 20;
    private float tLvl9 = 20;
    public float maxPos_x = 40;
    public float maxPos_y = 40;
    public float minDistEnz = 20;
    public GameObject prefab;
    public GameObject prefabEnz;
    public Sprite splvl1, splvl2, splvl3, splvl4, splvl5, splvl6, splvl7, splvl8, splvl9, splvl10;
    public List<GameObject> moleList = new List<GameObject>();
    public List<GameObject> moleListBack = new List<GameObject>();
    public float time = 0;
    public Text textTemps;
    public Text textATP;
    public Text textElect;
    public Image proxBar;
    private int compptATP = 0;
    private int comptElect = 0;
    private float numMol;
    private bool timeTrial = false;
    private bool safe = true;
    private GameObject pl;
    private GameObject enzyme;
    private BoxCollider2D view;

    // Use this for initialization
    void Start () {
        pl = GameObject.Find("Player");
        pl.GetComponent<PositionInMap>().maxPos_x = maxPos_x;
        pl.GetComponent<PositionInMap>().maxPos_y = maxPos_y;
        prefab.GetComponent<ProjectiloPosInMap>().maxPos_x = maxPos_x;
        prefab.GetComponent<ProjectiloPosInMap>().maxPos_y = maxPos_y;
        view = pl.GetComponent<BoxCollider2D>();
        newEnzyme();
        numMol = numLvL1;
```

```
    textTemps.text = "-- : -- ";
    textATP.text = compptATP.ToString();
    textElect.text = comptElect.ToString();
    proxBar.fillAmount = Mathf.Clamp(Vector3.Distance(pl.transform.position,
   enzyme.transform.position) / Vector3.Distance(new Vector3(maxPos_x,maxPos_y,0), new
   Vector3(0,0,0)), 0, 1f);

    for (int i = 0; i < numMolBack; i++)
    {
          Vector3 origin = new Vector3(Random.Range(maxPos_x, -maxPos_x),
          Random.Range(maxPos_y, -maxPos_y), 15);
        GameObject molecule;
        if (!view.OverlapPoint(origin))
        {
            molecule = (GameObject)Instantiate(prefab, origin, Quaternion.identity);
            molecule.GetComponent<Projectile>().moleList = moleListBack;
            float scale = Random.Range(0.04f, 0.06f);
            molecule.transform.localScale = new Vector3(scale, scale, scale);
            molecule.layer = LayerMask.NameToLayer("Background");
            moleListBack.Add(molecule);
        }
    }
}

// Update is called once per frame
void Update () {
    if (!safe)
    {
        if (moleList.Count < numMol)
        {
          Vector3 origin = new Vector3(Random.Range(maxPos_x, -maxPos_x),
          Random.Range(maxPos_y, -maxPos_y), 0);
            GameObject molecule;
            if (!view.OverlapPoint(origin))
            {
                molecule = (GameObject)Instantiate(prefab, origin, Quaternion.identity);
                molecule.GetComponent<Projectile>().moleList = moleList;
                moleList.Add(molecule);
            }
        }

        if (timeTrial)
        {
            time -= Time.deltaTime;
            if (time <= 0) changeLevel(level - 1);
            textTemps.text = time.ToString("#.00");
        }
    }
    if (moleListBack.Count < numMolBack)
    {
          Vector3 origin = new Vector3(Random.Range(maxPos_x, -maxPos_x),
          Random.Range(maxPos_y, -maxPos_y), 15);
        GameObject molecule;
        if (!view.OverlapPoint(origin))
        {
            molecule = (GameObject)Instantiate(prefab, origin, Quaternion.identity);
            molecule.GetComponent<Projectile>().moleList = moleListBack;
            float scale = Random.Range(0.04f, 0.06f);
            molecule.transform.localScale = new Vector3(scale, scale, scale);
            molecule.layer = LayerMask.NameToLayer("Background");
            moleListBack.Add(molecule);
        }
    }
```

```
    proxBar.fillAmount = 1f - Mathf.Clamp(Vector3.Distance(pl.transform.position,
    enzyme.transform.position) / Vector3.Distance(new Vector3(maxPos_x, maxPos_y, 0), new
    Vector3(-maxPos_x, -maxPos_y, 0)), 0, 1f);
}

void OnTriggerExit2D(Collider2D other)
{
    if (other.gameObject.name == "Player") safe = false;
}

public void changeLevel (int lvl)
{
    level = lvl;
    pl.transform.position = Vector3.zero;
    pl.transform.rotation = Quaternion.identity;
    pl.GetComponent<Rigidbody2D>().velocity = Vector3.zero;
    pl.GetComponent<Rigidbody2D>().angularVelocity = 0;
    safeArea();
    newEnzyme();
    safe = true;
    switch (level)
    {
        case 1:
            numMol = numLvL1;
            break;
        case 2:
            numMol = numLvL1;
            pl.GetComponent<SpriteRenderer>().sprite = splvl2;
            timeTrial = false;
            textTemps.text = "-- : -- ";
            compptATP = -1;
            comptElect = 0;
            textATP.text = compptATP.ToString();
            break;
        case 3:
            numMol = numLvL1;
            pl.GetComponent<SpriteRenderer>().sprite = splvl3;
            timeTrial = true;
            time = tLvl3;
            textTemps.text = time.ToString("#.00");
            compptATP = -1;
            comptElect = 0;
            break;
        case 4:
            numMol = numLvL2;
            pl.GetComponent<SpriteRenderer>().sprite = splvl4;
            timeTrial = false;
            textTemps.text = "-- : -- ";
            compptATP = -2;
            comptElect = 0;
            textATP.text = compptATP.ToString();
            break;
        case 5:
            numMol = numLvL2;
            pl.GetComponent<SpriteRenderer>().sprite = splvl5;
            timeTrial = false;
            textTemps.text = "-- : -- ";
            break;
        case 6:
            numMol = numLvL2;
            pl.GetComponent<SpriteRenderer>().sprite = splvl6;
            timeTrial = true;
            time = tLvl6;
            textTemps.text = time.ToString("#.00");
```

```
                comptElect = 0;
                textElect.text = comptElect.ToString();
                break;
            case 7:
                numMol = numLvL3;
                pl.GetComponent<SpriteRenderer>().sprite = splvl7;
                timeTrial = true;
                time = tLvl7;
                textTemps.text = time.ToString("#.00");
                compptATP = -2;
                comptElect = 2;
                textATP.text = compptATP.ToString();
                textElect.text = comptElect.ToString();
                break;
            case 8:
                numMol = numLvL3;
                pl.GetComponent<SpriteRenderer>().sprite = splvl8;
                timeTrial = true;
                time = tLvl8;
                textTemps.text = time.ToString("#.00");
                compptATP = 0;
                comptElect = 2;
                textATP.text = compptATP.ToString();
                textElect.text = comptElect.ToString();
                break;
            case 9:
                numMol = numLvL3;
                pl.GetComponent<SpriteRenderer>().sprite = splvl9;
                timeTrial =  true;
                time = tLvl9;
                textTemps.text = time.ToString("#.00");
                compptATP = 2;
                comptElect = 2;
                textATP.text = compptATP.ToString();
                textElect.text = comptElect.ToString();
                break;
        }
    }

    void safeArea()
    {
        while (moleList.Count > 0)
        {
            for (int i = 0; i < moleList.Count; i++)
            {
                GameObject obj = moleList[i];
                moleList.Remove(obj);
                Destroy(obj);
            }
        }
    }

    void newEnzyme()
    {
        if (enzyme != null) Destroy(enzyme);
        Vector3 origin = new Vector3(Random.Range(maxPos_x, -maxPos_x), Random.Range(maxPos_y, -
        maxPos_y), 0);
        while (Vector3.Distance(origin,pl.transform.position) < minDistEnz ||
        Vector3.Distance(origin, pl.transform.position) > maxPos_x - 5) origin = new
        Vector3(Random.Range(maxPos_x, -maxPos_x), Random.Range(maxPos_y, -maxPos_y), 0);
        enzyme = (GameObject)Instantiate(prefabEnz, origin, Quaternion.identity);
    }
}
```

11.2.7. PositionInMap.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PositionInMap : MonoBehaviour {

    public float maxPos_x = 20f;
    public float maxPos_y = 20f;

    private GameObject cam;

    // Use this for initialization
    void Start ()
    {
        cam = GameObject.Find("Camera");
    }

       // Update is called once per frame
       void Update () {

        float dist_x = Mathf.Abs(transform.position.x - cam.transform.position.x);
        float dist_y = Mathf.Abs(transform.position.y - cam.transform.position.y);

        if (transform.position.x > maxPos_x)
        {
            transform.position = new Vector2(-maxPos_x,transform.position.y);
              cam.transform.position = new Vector3(-maxPos_x - dist_x, cam.transform.position.y,
              cam.transform.position.z);
        }
        else if (transform.position.x < -maxPos_x)
        {
            transform.position = new Vector2(maxPos_x, transform.position.y);
              cam.transform.position = new Vector3(maxPos_x + dist_x, cam.transform.position.y,
              cam.transform.position.z);
        }
        if (transform.position.y > maxPos_y)
        {
            transform.position = new Vector2(transform.position.x, -maxPos_y);
              cam.transform.position = new Vector3(cam.transform.position.x, -maxPos_y - dist_y,
              cam.transform.position.z);
        }
        else if (transform.position.y < -maxPos_y)
        {
            transform.position = new Vector2(transform.position.x, maxPos_y);
              cam.transform.position = new Vector3(cam.transform.position.x, maxPos_y + dist_y,
              cam.transform.position.z);
        }
       }
}
```

11.2.8. Projectile.cs

```
using System.Collections;
using System.Collections.Generic;
```

```
using UnityEngine;

public class Projectile : MonoBehaviour {

    public float minSpeed = 7f;
    public float maxSpeed = 10f;
    public float temps = 30f;
    public float focusDist = 10f;
    public Sprite ene1, ene2, ene3;
    public List<GameObject> moleList = new List<GameObject>();

    private float speed;
    private bool colided = false;
    private GameObject pl;
    private Rigidbody2D rb2D;
    private BoxCollider2D view;
    private CircleCollider2D plCol;

    void Start()
    {
        pl = GameObject.Find("Player");
        view = pl.GetComponent<BoxCollider2D>();
        plCol = pl.GetComponent<CircleCollider2D>();
        rb2D = transform.GetComponent<Rigidbody2D>();
        speed = Random.Range(minSpeed, maxSpeed);

        Vector3 focusPoint = new Vector3 (Random.Range(pl.transform.position.x - focusDist,
        pl.transform.position.x + focusDist), Random.Range(pl.transform.position.y - focusDist,
        pl.transform.position.y + focusDist),0);
        Vector2 heading = focusPoint - transform.position;
        rb2D.velocity = (heading / heading.magnitude) * speed;
        Sprite[] arraySprite = { ene1, ene2, ene3 };
        GetComponent<SpriteRenderer>().sprite = arraySprite[Random.Range(0, 3)];
    }

    // Update is called once per frame
    void Update ()
    {
        if (GetComponent<CircleCollider2D>().IsTouching(plCol)) colided = true;

        temps -= Time.deltaTime;

        if (temps <= 0 && !view.OverlapPoint(transform.position) || colided &&
        !view.OverlapPoint(transform.position) )
        {
            moleList.Remove(gameObject);
            Destroy(gameObject);
        }
    }
}
```

11.2.9. ProjectilePosInMap.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ProjectiloPosInMap : MonoBehaviour {

    public float maxPos_x = 20f;
    public float maxPos_y = 20f;
```

```
    // Use this for initialization
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {

        if (transform.position.x > maxPos_x)
        {
              transform.position = new Vector3(-maxPos_x, transform.position.y,
              transform.position.z);
        }
        else if (transform.position.x < -maxPos_x)
        {
              transform.position = new Vector3(maxPos_x, transform.position.y,
              transform.position.z);
        }

        if (transform.position.y > maxPos_y)
        {
              transform.position = new Vector3(transform.position.x, -maxPos_y,
              transform.position.z);
        }
        else if (transform.position.y < -maxPos_y)
        {
              transform.position = new Vector3(transform.position.x, maxPos_y,
              transform.position.z);
        }
    }
}
```

12. Manual d'usuari

Per tal d'obtenir la carpeta del projecte caldrà que la descomprimim de l'arxiu ZIP amb un programa de compressió de fitxers com WinRAR, WinZip o 7-Zip.

Pel que fa all fitxer Apk adjunt amb el projecte, només funciona en dispositius Android. La instal·lació i execució és molt senzilla:

1. Connectem el dispositiu al ordenador, habilitant el dispositiu per transferir fitxers.
2. Copiem el fitxer Apk dins del dispositiu en alguna carpeta on el puguem localitzar (per exemple en la carpeta "Downloads").
3. Ja des del dispositiu, busquem el fitxer i l'executem. Ens dirà si estem segur de que el fitxer és segur abans d'instal·lar-lo. Acceptem i es duu a terme la instal·lació.
4. L'aplicació apareixerà llesta per la seva execució en el nostre menú del mòbil.