Mestrado em Engenharia Informática
Dissertação/Estágio
Relatório Final

MOBILE GAME DEVELOPMENT

Departamento de Engenharia Informática
Faculdade de Ciências e Tecnologias
Universidade de Coimbra

Pólo II, Pinhal de Marrocos
3030-290 Coimbra

Wingzstudio, Lda
Bairro Sousa Pinto, 35 - 1º
3000-393 Coimbra

mail@wingzstudio.com

Mestrando:
Ricardo Filipe Gaspar Lopes
rflopes@student.dei.uc.pt

Orientador da empresa:
Nuno Esculcas
nuno.esculcas@wingzstudio.com

Orientador do departamento:
Licínio Roque
lir@dei.uc.pt

3 de Julho de 2013

Resumo

O desenvolvimento de jogos para dispositivos móveis tem revelado uma maior relevância no mercado de aplicações para este tipo de dispositivos. É nesta categoria que se encontram algumas das aplicações mais bem sucedidas e que maior reconhecimento têm no universo das aplicações para dispositivos móveis.

A WingzStudio Lda, pretende tornar-se numa marca de referência na criação deste tipo de produtos, necessitando de ter a ferramenta que serve de base ao desenvolvimento destes mesmos produtos o mais actualizada e completa possível. Com a introdução de novas funcionalidades e a optimização de funcionalidades existentes pretende-se levar essa ferramenta para um nível muito próximo, senão igual ao actual estado da arte neste tipo de produtos de suporte.

O objectivo para este estágio passa numa primeira fase pela melhoria da ferramenta que a empresa desenvolveu, como base para o desenvolvimento de jogos e numa segunda fase pela criação de jogos que possam não só ser verificações práticas das melhorias efectuadas na primeira fase, mas também como produtos que possam contribuir para o reconhecimento da marca no mercado.

Palavras-chave

“Apple”, “desenvolvimento de jogos”, “dispositivos móveis”, “motor de jogo”, “WingzStudio”.

Índice de figuras

Índice de tabelas

Índice

Glossário

Termo	Descrição
Aspect ratio	Relação entre a tamanho vertical e horizontal de um dispositivo
Cocoa	Ferramenta disponibilizada pela Apple para criação de interfaces gráficas (botões, caixas de texto, etc)
In-App Purchases	Mecanismo de compra de produtos no interior das aplicações (subscrições, bens virtuais, etc)
Objective-C	Linguagem de programação disponibilizada pela Apple para desenvolvimento de aplicações para dispositivos de iOS
OpenGL	API multi-plataforma que permite a renderização de gráficos 2D e 3D
Partícula	Textura utilizada para representar objectos de pequenas dimensões
Ponto âncora	Ponto que define a origem do sistema de coordenadas relativo de cada objecto
Sistema de partículas	Sistema que permite a representação de objectos cujas superfícies não estão bem definida (fumo, água, etc)
Spritesheet	Combinação de várias imagens num único ficheiro
Tile Map	Representação de um cenário 2D em que está armazenada a informação sobre as células que compõem o cenário, assim como a posição da textura a que cada célula corresponde
Viewport	Área da janela onde é mapeado o quadrado do plano de projecção

1. Introdução

Este documento reflecte o trabalho planeado e realizado no âmbito da disciplina Estágio / Dissertação do Mestrado em Engenharia Informática referente ao ano lectivo 2012 - 2013.

Este projecto está a ser realizado na empresa Wingzstudio Lda, tendo como orientadores o Engenheiro Nuno Esculcas, por parte da empresa, e do Professor Licínio Roque, por parte do Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologias da Universidade de Coimbra.

A Wingzstudio é uma jovem empresa, sediada em Coimbra que se dedica à criação de jogos para dispositivos móveis. De momento o seu foco está direccionado para os dispositivos comercializado pela marca Apple e que correm o sistema operativo disponibilizado pela mesma marca (iOS).

O mercado dos jogos para dispositivos móveis tem-se mostrado um mercado com grande saída e que derivado da sua grande amplitude tem uma capacidade bastante elevada para acolher novos jogos. Apesar de ser um mercado com grande oferta, é também ele um mercado bastante exigente o que leva a que todos os envolvidos na produção de jogos tenham que utilizar sempre o estado da arte no que a ferramentas e tecnologias diz respeito, sob pena de ficarem para trás num mercado em constante evolução.

A entrada na empresa originou a necessidade por uma ferramenta que permitisse a rápida implementação dos jogos, sem ser necessário efectuar novas configurações para cada projecto. Uma pesquisa de mercado revelou um grande leque de ferramentas que permitiam este tipo de desenvolvimento, contudo e de acordo com a filosofia da empresa de realizar produções integralmente desenvolvidas em Portugal e/ou por Portugueses, decidiu-se enveredar pela criação de uma ferramenta própria e que permitisse aos elementos envolvidos nas tarefas de programação saber cada detalhe da ferramenta.

O processo de criação desta ferramenta revelou-se bastante complexo. Ciente desta complexidade, os elementos da empresa optaram por implementar uma ferramenta que lhes permitisse a criação do seu primeiro projecto, por forma a entrar o mais rapidamente no mercado. Deste facto resultou uma ferramenta que, apesar de permitir a criação de jogos com grande nível de qualidade, apresentava algumas limitações em termos de desempenho e em termos de funcionalidades disponíveis. Por este facto a empresa sentiu a necessidade de melhorar a ferramenta de desenvolvimento que dispunha por forma a que esta se tornasse mais eficaz.

Foi devido a essa necessidade que surgiu este projecto de estágio. Este projecto foi dividido em dois períodos distintos, sendo cada um deles coincidente com cada um dos semestres da disciplina. O primeiro período tem como objectivo a melhoria da ferramenta produzida na empresa. Durante o segundo período deste estágio, pretende-se proceder à criação de dois jogos que servirão não só para aumentar o portefólio da empresa mas também para dar uma maior visibilidade à empresa na loja *online*. A ideia subjacente à criação destes dois objectivos distintos passa pela utilização dos jogos criados no segundo período do estágio como método de validação das funcionalidades adicionadas e/ou alteradas na ferramenta.

A forma delineada para se atingir o primeiro objectivo passa pela análise de soluções e ferramentas existentes no mercado que possam ser utilizadas para substituir a ferramenta desenvolvida na empresa. Esta análise não tem como objectivo a procura de um substituto para a ferramenta própria da empresa, mas sim o levantamento de funcionalidades que se possam vir a revelar mais valias para esta.

Os resultados que se esperam obter no final deste estágio estão também divididos em dois grupos, estritamente ligados aos dois objectivos maiores do projecto. O primeiro conjunto de resultados que se pretende obter é uma ferramenta melhorada que se pretende a um nível próximo, senão no mesmo nível, do estado da arte deste ramo. O segundo conjunto de resultados prende-se com a produção de dois jogos que levem a empresa a ganhar visibilidade no vasto mercado de jogos para dispositivos móveis.

Para apresentar todo o trabalho realizado assim como os resultados produzidos pelo mesmo, elaborou-se este documento que se inicia com as motivações para este trabalho, os objectivos para o mesmo assim como a abordagem a utilizar de modos a que sejam cumpridos. No segundo capítulo é apresentada uma análise a algumas ferramentas disponíveis no mercado e que permitem a criação rápida de jogos para dispositivos móveis. É também apresentada uma análise comparativa entre as ferramentas analisadas e a ferramenta produzida na empresa. O terceiro capítulo incidirá sobre as metodologias utilizadas durante o desenvolvimento assim como as razões que levaram à escolha dessa metodologias. O capítulo número quatro apresenta um retrato do trabalho realizado até ao momento assim como os resultados que esse trabalho produziu. Após os capítulos relativos ao trabalho desenvolvido no primeiro semestre seguem-se os capítulos descritivos do trabalho realizado ao longo da segunda metade deste projecto, tendo especial destaque o capítulo número cinco onde é descrito todo o processo de desenvolvimento do jogo construído para este estágio. O capítulo número seis, apresenta os resultados de uma avaliação de usabilidade do produto bem como a análise desses mesmo resultados.

2. Estado da arte

A análise do estado da arte para este projecto recaiu sobre as tecnologias existentes que servem de base para o desenvolvimento de jogos e sobre as práticas actualmente utilizadas para proceder ao desenvolvimento. De seguida são apresentadas algumas análises a tecnologias que permitem a rápida criação de jogos para dispositivos iOS assim como uma descrição das fases mais utilizadas no desenvolvimento de jogos.

2.1. Tecnologias

Para este projecto, a única tecnologia que se irá utilizar será a ferramenta que servirá de base para a criação dos jogos e que tem vindo a ser desenvolvida pela empresa. Tal como referido no capítulo anterior, apesar de existirem bastantes ferramentas disponíveis no mercado, a empresa optou por criar a sua própria ferramenta. Com o passar do tempo tem sentido a necessidade de a melhorar por forma a esta ser capaz de oferecer mais e melhores soluções na implementação dos produtos que irão ser desenvolvidos no futuro. Para saber que melhorias deveriam ser realizadas, foi efectuada uma análise a alguns produtos disponíveis no mercado, assim como uma comparação destes produtos com a ferramenta desenvolvida na empresa, para que pudesse ser elaborada uma lista de melhorias a implementar na mesma. A análise a estes produtos bem como a comparação de ferramentas encontra-se apresentada de seguida.

Sparrow

O Sparrow é uma ferramenta para desenvolvimento de jogos para iOS escrita em Objective-C e que permite ao utilizador a criação de *interfaces* gráficas com recurso à *framework* Cocoa, o que torna a criação destes elementos bastante simples. Ao contrário da construção das *interfaces* gráficas que podem ser construídas através dos recursos nativos do dispositivos, a mecânica de jogo para quem utiliza esta ferramenta tem que ser implementada com recurso às funcionalidades por ela disponibilizadas.

A implementação de um jogo com recurso a esta ferramenta, tem por princípio a utilização de um classe abstracta que serve de base para todos os objectos capazes de se desenharem no ecrã. Esta classe disponibiliza algumas propriedades básicas para todas as suas subclasses e que são propriedades comuns à grande maioria dos objectos como é o caso da posição, das dimensões ou da rotação. Com recurso a esta classe, o Sparrow apresenta já um conjunto de classes implementadas que permitem ao utilizador a criação de elementos gráficos. Entre estes elementos destacam-se os botões e os campos de texto, sendo estes elementos implementados com recurso à *framework* Cocoa.

Além das classes que permitem a interacção do jogador com o jogo existe também implementada uma classe representativa de uma imagem, que permite ao utilizador representar visualmente os elementos do jogo. A presença deste conjunto de classes pré-implementadas revela-se uma mais valia uma vez que são oferecidas ao utilizador bases para iniciar o desenvolvimento do seu produto.

Uma funcionalidade obrigatória em todas as ferramentas que servem de base ao desenvolvimento de jogos é a possibilidade de representar os objectos visualmente, uma vez que uma boa representação visual é muitas vezes um ponto de partida para uma fácil aceitação do jogo pelo mercado. Para representar graficamente os elementos do jogo são utilizadas duas classes distintas: uma classe representativa da textura e outra representativa da imagem. Na primeira classe é armazenada toda a informação binária do ficheiro onde se encontra localizada a imagem enquanto que a segunda classe tem a capacidade de transformar esses dados em algo visualmente perceptível. A utilização de imagens por parte desta ferramenta pode ser efectuada de duas formas distintas, através de um ficheiro com uma imagem única, ou a utilização de um ficheiro com várias imagens, denominada por *spritesheet*. A utilização da segunda opção torna-se mais vantajosa que a primeira, uma vez que em OpenGL (que é a base do motor gráfico do Sparrow) é apenas mantida uma textura activa em cada momento, ou seja, se o utilizador armazenar uma imagem por ficheiro, sempre que pretender apresentar uma imagem diferente da que está actualmente em memória será necessário enviar toda a informação da nova imagem para

memória, o que pode provocar uma quebra no desempenho devido ao fluxo de informação entre a memória do dispositivo e a memória do dispositivo gráfico. No caso da utilização de *spritesheets* este problema pode não ser tão frequente, uma vez que ao ter em memória várias imagens, que estarão relacionadas entre si, é menos provável que ocorra uma troca de informação na memória do dispositivo gráfico.

A possibilidade de apresentação de texto no ecrã do dispositivo torna-se imperioso a partir do momento em que o jogo requer comunicação (mesmo que unidirecional) com o jogador. Para dar resposta a este requisito, o Sparrow apresenta soluções que podem recorrer a dois tipos de conteúdos. O primeiro tipo de conteúdos suportado por esta ferramenta são os tipos de letra disponíveis no sistema que são apresentados recorrendo aos elementos disponibilizados pelo Cocoa. O segundo tipo de conteúdos permite ao utilizador a apresentação de textos com recursos a tipos de letra personalizados, que são armazenados em ficheiros de imagem. Esta segunda solução permite ao utilizador da ferramenta dar ao jogo um aspecto mais coerente em termos gráficos.

A utilização de imagens acresce por si só algum valor ao jogo, no entanto se as imagens apresentadas forem estáticas, o produto pode vir a tornar-se monótono após algum tempo de jogo. Para isso, o Sparrow tem implementada um sistema que permite a um objecto a apresentação de várias imagens ao longo do tempo, podendo estas imagens estar directamente relacionadas com as acções efectuadas pelo jogador ou com o resultado das suas acções. Este sistema permite que um objecto armazene várias sequências de imagens, devidamente identificadas, e que servirão para dar mais ritmo ao jogo. Sempre que é criada uma desta sequência é possível ao utilizador definir a frequência com que se processa a troca de imagens. A utilização desta função faz com que todas as imagens da sequência fiquem visíveis durante o mesmo período de tempo. No entanto podem existir situações em que seja necessário uma imagem da sequência seja apresentada mais tempo que as restantes, existe a possibilidade de definir o tempo de apresentação para cada imagem da sequência.

Um outro recurso que torna o jogo bastante mais interessante é a reprodução de audio. Com a utilização do Sparrow o programador tem a capacidade de aceder ao sistema som do dispositivo por forma a reproduzir o audio que pretende. Acrescentadas à possibilidade de reprodução de audio estão implementadas funções que permitem a manipulação do audio tal como a sua paragem. a alteração do volume, pause e resumo do mesmo.

Dependendo do tipo de jogo que se pretende implementar pode surgir a necessidade de atribuir funções aos objectos que permitam alterar automaticamente as suas propriedades (posição, rotação, opacidade, entre outras) ao longo do tempo. Para este efeito foi disponibilizado aquilo a que os criadores desta ferramenta chamam de sistema de animações. Este sistema permite atribuir a um objecto uma fun-

ção que altere uma ou várias das suas características durante um intervalo de tempo. É possível definir através deste sistema que um objecto se mova de uma determinada posição para outra num intervalo de tempo específico ao mesmo tempo que o seu tamanho aumente até ao dobro no final do movimento, sendo este aumento gradual durante o movimento. Para evitar que estas alterações se tornem monótonas existe a possibilidade de definir qual a função de transição que será aplicada à animação com recurso a uma biblioteca de funções. É ainda possível definir se o início da animação terá ou não algum atraso relativamente ao momento em que é ordenado o seu começo. O Sparrow utiliza as classes das diversas animações apenas como armazenamento da informação relativa a cada uma das animações, sendo que a aplicação dessas alterações só é efectuado quando a animação é adicionada ao gestor de animações, sendo este gestor responsável pela destruição das animações quando estas chegam ao fim.

Analisando as operações que se podem efectuar sobre os objectos do jogo, podemos verificar que estas operações podem produzir efeitos diferentes se aplicadas sobre diferentes pontos do objecto, como é o caso da rotação, que produz resultados distintos se aplicada sobre um dos cantos do objecto ou sobre o seu centro. Para resolver este problema é disponibilizado um sistema que permite definir os pontos âncora dos objectos. O utilizador pode, sempre que achar conveniente, definir o ponto âncora do objecto em qualquer ponto do seu sistema de coordenadas. Caso não seja definido o ponto âncora, este irá por defeito ficar localizado no canto superior esquerdo do objecto. Na imagem seguinte é possível verificar as diferenças entre a utilização do ponto âncora no canto superior esquerdo ou no centro do objecto, o que nos dá uma noção dos problemas que podem vir a surgir com uma má utilização desta funcionalidade.

Figura 1 - Comparação da rotação de um objecto com diferentes localização do ponto âncora

Quando foi introduzido o iPad no mercado, foi também acrescentada uma dificuldade para os criadores de jogos. Este problema está relacionado com a diferença de dimensões de ecrã entre os dois tipos de dispositivos, iPhone e iPad. Enquanto que o iPhone apresenta um ecrã de 480x320 *pixels* (um *aspect ratio* de 1.5:1), o iPad apresenta um ecrã de 1024x768 *pixels* (1.33:1 no que diz respeito ao *aspect ratio*). Olhando para estes números é possível verificar que o ecrã do iPad é bastante maior que o ecrã do iPhone, sendo apenas proporcional num dos eixos, ou seja, é possível transformar um dos eixos (vertical ou horizontal) do iPhone para ter o mesmo tamanho do eixo respectivo no iPad, contudo para o outro eixo não é possível arranjar uma proporcionalidade. A introdução do iPhone de quinta geração no mercado veio agravar ainda mais essa dificuldade, uma vez que o ecrã deste novo dispositivo tem uma resolução de 1136x640 *pixels* (um *aspect ratio* de 16:9). Para minimizar este problema, o Sparrow duplica o tamanho do *viewport*. Esta operação leva a que cada *pixel* do iPhone passe a corresponde a quatro *pixels* no iPad. No entanto ao serem efectuados alguns cálculos verificamos que existe uma zona de 64x88 *pixels* que não ficarão associados ao *viewport*. Um primeiro problema de efectuar esta escala prende.se com a qualidade das imagens que são apresentadas, uma vez que ao ser escalado o *viewport* também a representação visual dos objectos irá ser alterada. Para resolver este problema estão implementados dois *viewports* ficando o sistema gráfico responsável pela escolha de qual seleccionar para cada dispositivo.

Cocos 2D

Cocos 2D é uma das ferramentas mais conhecidas e utilizadas para o desenvolvimento de jogos para dispositivos móveis. À semelhança do que acontece com o Sparrow, a versão para dispositivos iOS está também implementada sobre a linguagem Objective-C, o que obriga a que o utilizador tenha que ter algum conhecimento básico desta linguagem por forma a iniciar rapidamente o desenvolvimento de um produto.

Tal como acontece na ferramenta analisada anteriormente e como se torna natural neste tipo de tecnologia são disponibilizados ao utilizador mecanismos para representar graficamente os objectos. Esta ferramenta utiliza também duas classes distintas para a representação dos elementos do jogo, uma para a informação binária do ficheiro e outra com a capacidade para representar essa informação no ecrã. Para que seja possível criar uma instância da segunda classe o utilizador tem à sua disposição três métodos distintos. O primeiro método de criar este tipo de objectos utiliza o recurso a um ficheiro onde está armazenada a imagem que representa o objecto. No caso da segunda alternativa, a informação para a representação gráfica do objecto pode estar armazenada numa *spritesheet*, tirando partido das vantagens já referidas deste tipo de armazenamento. O último método disponível para a criação da representação gráfica dos objectos tira partido da informação binária da textura, já armazenada na memória do dispositivo, permitindo assim um acesso mais rápido à informação. Este terceiro método tira partido da leitura da informação dos ficheiros efectuada através do primeiro ou segundo métodos descritos.

Uma funcionalidade que se encontra presente nesta ferramenta e que merece destaque óbvio é o facto de ser possível criar todo um cenário para um jogo 2D através da leitura de um ficheiro de *Tile Maps* que contém as informações necessárias para este tipo de cenário, como são o caso das texturas a ser utilizadas assim como da localização dos objectos no cenário.

Por forma a tornar o jogo mais dinâmico é oferecida ao utilizador a utilização de um sistema de partículas. Este sistema permite a criação de objectos sem uma forma predefinida e que seriam difíceis de representar através de imagens normais, como é o caso de fumo, fogo ou água. A representação deste tipo de objectos é efectuada com recurso a imagens simples que podem ser representadas em grande número e que podem ver alteradas as suas propriedades visuais (cor, tamanho, posição, entre outros) para que formem um objecto mais realista. Para a criação deste sistema são fornecidas duas opções: *Point Particles* ou *Quad Particles*. Estes dois grupos de partículas apresentam algumas diferenças entre si, das quais se salientam as seguintes: as *Quad Particles* permitem efectuar uma rotação de cada partícula sobre o seu próprio eixo, enquanto que as *Point Particles* apenas permitem a rotação do

sistema como um todo, as *Point Particles* apresentam também limitação no tamanho máximo que uma imagem pode ter para fazer parte de um sistema (64x64 *pixels*) apresentando a vantagem de atribuir uma escala a todas as partículas através da aplicação de uma escala ao sistema no global, enquanto que no caso das *Quad Particles* esta alteração tem que ser efectuada para cada partícula.

Tal como o Sparrow, também o Cocos 2D permite a apresentação de imagens sequenciais por forma a dar maior dinâmica ao jogo. Este processo é realizado com recurso a três classes distintas. Uma que serve como armazenamento de todas as texturas que compõem a animação, outra para relacionar o nome de cada imagem com a sua posição na textura e uma outra que permite a representação gráfica da textura no ecrã. Também esta ferramenta oferece a possibilidade de o jogo apresentar texto no ecrã sempre que for necessário, podendo o utilizador escolher entre tipos de letra presentes no sistema ou tipos de letra personalizados e armazenados sobre a forma de imagens.

No sistema audio implementado nesta ferramenta, está presente a distinção entre dois tipos de audio: músicas e efeitos sonoros. Enquanto que o primeiro tipo é geralmente mais prolongado e serve para criar um determinado ambiente no jogo, o segundo permite dar ao jogo uma maior noção de acção, uma vez que oferece a possibilidade de reprodução de um audio curto em consequência de acções. Para o primeiro tipo de audio é permitido um conjunto de manipulações que permitem um grande um controlo sobre o audio que se pretende apresentar de entre as quais se destacam a capacidade de para ou retomar a reprodução assim como alterar o volume com que o audio é reproduzido. Além destas funcionalidades é também possível definir se uma música deve ou não ser tocada em ciclo, retirando ao utilizador a preocupação de recomeçar a música sempre que esta chega ao fim. No que diz respeito aos efeitos sonoros é possível iniciá-los e terminá-los assim como produzir algumas alterações nos seus sinais (*pitch*, *pan* e *gain*). É também possível definir o volume com que o efeito é reproduzido.

A manipulação de propriedades dos objectos é também uma das funcionalidades presentes nesta ferramenta. Neste caso existem dois tipos de funções que podem ser aplicadas sobre as propriedades de um objecto: acções instantâneas e acções que são aplicadas gradualmente ao longo de um certo período de tempo. Como se depreende pelo seu nome, o primeiro tipo de acções efectua a alteração das propriedades do objecto imediatamente, enquanto que o segundo tipo de acções vai alterando as propriedades gradualmente. Esta ferramenta disponibiliza ao utilizador a possibilidade de reverter acções aplicadas ao objecto, que pode ser útil no caso de se pretender que um objecto vagueie de um lado para o outro do ecrã. Assim basta atribuir uma função a esse objecto e aplicar o seu inverso quando este terminar. É também apresentada nesta ferramenta a possibilidade de criar cadeias de funções,

sendo possível escolher entre três variedades de cadeias: sequências, paralelização e repetição. Na primeira variante, é definido um conjunto de funções que são executadas pela ordem pré estabelecida enquanto que na segunda variante todas as acções são aplicadas em simultâneo. As cadeias em repetição são um casos específico de cadeias sequenciais, onde no final da última acção é retomada a primeira. Neste tipo de cadeias é possível definir se o ciclo é executado indefinidamente ou se termina após um determinado número de repetições.

Esta ferramenta apresenta também a possibilidade do utilizador definir pontos de âncora para os objectos, tendo as vantagens desta solução sido referidas na análise da ferramenta anterior. A utilização de um motor de física, que simula o comportamento real dos objectos, é também uma possibilidade durante o uso desta ferramenta, sendo esta simulação feita com recurso ao motor de físicas Box2D.

Corona SDK

Contrariamente à implementação utilizada pelas duas ferramentas anteriores, esta ferramenta não utiliza a linguagem nativa dos dispositivos iOS, utilizando uma linguagem de *scripting* denominada LUA.

A representação gráfica dos objectos é também, tal como era expectável, uma das características apresentadas por esta ferramenta tendo já implementado um conjunto de classes que permitem a representação de um variado conjunto de objectos tais como rectângulos, círculos, imagens ou texto, que tal como nas anteriores ferramentas possuem propriedades destacam-se posição e rotação do objecto. Nesta ferramenta cada objecto tem ainda a possibilidade de ser transformado num botão. Cada um destes objectos descende de uma classe genérica que pode ser tratada como uma tabela normal de LUA, significando isto que em caso de necessidade é possível definir novos métodos e novas propriedades para os objectos, tendo apenas em consideração que as novas funcionalidades não podem entrar em conflito com as funcionalidades presentes no objecto.

Tal como nas ferramentas anteriores também esta fornece ao utilizador a possibilidade de armazenar as suas imagens em *spritesheets* tirando partido das vantagens já enumeradas para este tipo de solução. A apresentação de texto no ecrã é outra funcionalidade presente nesta ferramenta, mas ao contrário do que se encontrou nas anteriores, neste caso apenas existem mecanismos para apresentar tipos de letra nativos do sistema, sendo necessário que o utilizador crie o mecanismo para a apresentação de tipos de letra personalizados se optar por essa solução. Também presente no Corona SDK está a possibilidade de um objecto ter uma sequência de imagens identificado por um nome, tendo esta a capacidade para armazenar também o tempo de exposição de cada imagem e a informação sobre se a animação deve ou não ser reproduzida em ciclo. No que diz respeito ao audio, esta ferramenta fornece ao utilizado a possibilidade de reproduzir sons de duas naturezas distintas: músicas e efeitos sonoros. Tal como referido na análise à ferramenta anterior, as músicas são utilizadas para criar um determinado ambiente no jogo, enquanto que os efeitos sonoros são utilizados para dar maior realismo a certos acontecimentos que sucedem no decorrer do jogo. Uma funcionalidade que apenas se encontra presente nesta ferramenta é a capacidade para efectuar a reprodução de vídeos, realizada através da utilização do reprodutor de vídeo nativo do sistema operativo, ficando o utilizador impedido de definir quaisquer propriedades para essa reprodução.

À semelhança do que foi encontrado nas duas ferramentas anteriores, também nesta está disponível a possibilidade de se atribuírem funções para alterar as propriedades dos objectos. É assim possível definir que um conjunto de propriedades

do objecto seja alterado ao longo de um intervalo de tempo, permitindo ainda ao utilizador a definição de que propriedades devem ser alteradas e em que medida isso irá acontecer. Existe também a opção de definir a duração de cada função, assim como qual o atraso que deve ser aplicado a cada uma.

A interacção com o utilizador foi também um factor tido em conta no desenvolvimento desta aplicação estando disponíveis ao utilizado três tipos de elementos para este tipo de funções: botões, caixas de alerta e campos de texto. Os botões oferecem ao utilizador a possibilidade de receber informação sobre as intenções do jogador. As caixas de alerta são utilizadas para transmitir informação ao jogador durante o decurso do jogo, enquanto que os campos de texto permitem ao jogador a introdução de informação escrita.

Uma funcionalidade que esta ferramenta oferece e que não foi encontrada nas anteriores é um sistema para tratar *in-app purchases*. Esta solução permite que o utilizador se abstraia de todos os passos necessários à realização das operações relacionadas com este sistema e onde se incluem o carregamento dos produtos disponíveis para compra, a compra de produtos e a reposição de produtos já comprados.

Wingzstudio Engine

A última ferramenta que se analisou para o estado da arte foi a ferramenta desenvolvida no seio da empresa com o objectivo de esta análise servir de base para a comparação com as restantes ferramentas analisadas.

A ferramenta desenvolvida pela empresa, e contrariamente ao que foi possível concluir na análise das restantes ferramentas, foi desenvolvida com o objectivo de ser facilmente migrável para outros sistemas operativos que não os desenvolvidos pela Apple. Desta forma, e uma vez que os elementos que compõem a equipa de desenvolvimento da empresa não estavam muito familiarizados com linguagens de *scripting*, a escolha para o desenvolvimento desta ferramenta caiu sobre a linguagem C++, deixando apenas a camada de acesso ao sistema dependente da linguagem nativa de cada sistema.

Nesta ferramenta todos os objectos são descendentes de uma classe que guarda propriedades básicas e que são comuns à maioria dos objectos (posição, rotação, velocidade, entre outras). Uma vez que poderia ser necessário efectuar comparações entre os diversos objectos foi implementado um sistema de identificação que permite atribuir a cada objecto um identificador único. É também possível colocar uma etiqueta em cada objecto por forma a ser mais fácil realizar a identificação de cada um. Para além de definir as propriedades de cada objecto, esta classe define ainda dois métodos comuns a todos os objectos do jogo, um método de representação gráfica e outro de actualização do objecto.

No que diz respeito ao tratamento de imagens , existem duas classes que são importantes de salientar. Uma classe representativa da informação da textura no contexto do OpenGL, e outra classe que representa o aspecto gráfico dessa textura. Nesta segunda classe estão armazenadas algumas informações importantes, tais como a textura de onde deve ser retirada a informação bem como a posição de onde deve ser retirada a informação. Esta classe tem também, um contador de dependentes que permite que seja destruída sempre que não existir nenhum objecto dependente da representação gráfica, sendo esta funcionalidade bastante útil na gestão de memória. À semelhança das anteriores ferramentas também esta ferramenta suporta o armazenamento de várias imagens num único ficheiro por forma a tirar partido das vantagens deste tipo de armazenamento. Para este efeito existe uma classe que trata da leitura e do armazenamento das informações desse tipo de ficheiro. A utilização de imagens dinâmicas é também suportada e para esse efeito existe uma classe que guarda a informação das imagens que fazem parte da sequência, do identificador da sequência e do tempo que cada imagem deve estar exposta no ecrã. Além de ser permitido, ao utilizador, adicionar de uma só vez um conjunto de imagens à animação e definir o tempo de exposição para todas é também

possível adicionar imagens individualmente e definir o tempo de apresentação para cada uma. Tirando partido de todas estas funcionalidades relacionadas com imagens, foi implementado um sistema de apresentação de texto com base em ficheiros de imagens. Contrariamente às ferramentas anteriores, esta ferramenta não permite a utilização de tipos de letra presentes no sistema, sendo apenas possível utilizar tipos de letra com recurso a ficheiros de imagem. Todas as funcionalidades implementadas para o manuseamento de imagens, permitiu a implementação de um sistema de partículas, que utiliza um classe descendente da classe Objecto para representar cada partícula, tirando esta classe partido das propriedades presentes na classe Objecto.

Similarmente ao que acontece com todas as ferramentas anteriores, também esta ferramenta permite ao utilizador a reprodução de audio fazendo igualmente a distinção entre dois tipos de audio: músicas e efeitos sonoros. Para cada um destes tipos de audio é possível efectuar algumas operações básicas tais como o início, a pausa e o recomeço do audio ou o ajuste do volume com que este é reproduzido.

Tal como nas ferramentas analisadas anteriormente também a ferramenta desenvolvida pela empresa permite a utilização de funções para alterar propriedades dos elementos do jogo. Esta ferramenta apresenta um conjunto de classes que permitem a alteração de algumas propriedades dos objectos tais como a posição, a rotação ou a opacidade, sendo que caso se sinta a necessidade de efectuar alterações a outras propriedades o sistema está preparado para isso.

No que concerne à interacção com o utilizador esta ferramenta apresenta um conjunto de soluções onde se encontram os típicos botões, campos de apresentação de texto, botões de estado (em que com cada movimento de pressão o estado do botão é alterado) e caixas de apresentação de informação ao jogador, oferecendo estas últimas a possibilidade do jogador fornecer uma resposta à informação apresentada.

Caso o jogo a ser desenvolvido pela empresa necessite de simular físicas do mundo real, esta ferramenta está para ser integrada com um motor de física que irá efectuar essa simulação e produzir os resultados pretendidos. O motor de física utilizado pela ferramenta é o Box2D.

Comparação de ferramentas

		Sparrow	Cocos 2D	Corona SDK	Wingzstudio Engine
Linguagem de implementação		Objective-C	Objective-C	LUA	C++
Interface gráfica		Cocoa	Cocoa	Próprio	Cocoa + Próprio
Apresentação de texto		sistema + personalizado	sistema + personalizado	sistema	personalizado
Ficheiros de imagens únicas		✓	✓	✓	✓
Ficheiros de imagens múltiplas		✓	✓	✓	✓
Apresentação de sequências de imagens		✓	✓	✓	✓
Multimédia	Audio	✓	✓	✓	✓
	Vídeo	✓	✓	✓	✓
Efeitos sobre objectos	Efeito linear	✓	✓	✓	✓
	Funções easing	✓	✓	✓	✓
Sistema de partículas			✓		✓ a)
Pontos âncora		✓	✓		
Suporte para multi-dispositivos		✓			✓ b)
Tile Maps			✓		
Suporte para motor de física			✓	✓	✓
Sistema de in-app purchases				✓	

a) bastante simples. recurso aos objectos genéricos
b) não suporta iPhone 5 nem iPad 3

Tabela 1 - Comparação das ferramentas de desenvolvimento de jogos

Analisando a tabela anterior podemos verificar que apesar de se encontrar num estado de desenvolvimento bastante aceitável, a ferramenta desenvolvida na empresa ainda pode vir a sofrer algumas melhorias e é também possível verificar que algumas das funcionalidades não implementadas podem vir a ser bastante úteis no desenvolvidos dos projectos futuros da empresa.

Box 2D

Como foi anteriormente analisado, existem algumas ferramentas que dão suporte à utilização de um motor de físicas sempre que se pretendem simular físicas do mundo real. Em todos os casos analisados o motor escolhido é o Box 2D. De seguida, é apresentada uma breve descrição deste motor e das funcionalidades que ele oferece.

O Box 2D é um biblioteca de simulação de corpos rígidos, idealizado para o desenvolvimentos de jogos a duas dimensões. Este motor é utilizado para que os programadores façam mover os objectos do seu jogo de uma forma realista com o objectivo de tornar o mundo de jogo mais interactivo. O facto deste motor ter sido escrito em C++ torna-o bastante portável, uma vez que a linguagem mais utilizada no desenvolvimento de aplicações e jogos para iOS permite a utilização de código escrito em C++.

O motor de física está dividido em três módulos distintos: um módulo de matéria comum outro para detecção de colisões e um outro para dinâmica de objectos. O módulo de matéria comum contém todas as definições do motor assim como todos os mecanismos de gestão de memória inerentes à sua utilização. Está ainda presente neste módulo toda a matemática relacionada com vectores utilizada por este motor. No segundo módulo estão contidas as formas geométricas e as funções que operam sobre elas. Estas formas geométricas descrevem a geometria da colisão e podem ser usadas à parte do sistema de simulação de físicas. No motor estão definidos quatro tipos de geometria para os objectos: círculos, polígonos, arestas e cadeias. No caso dos círculos as únicas propriedades associadas a este tipo de figuras geométricas são a posição do seu centro e o raio da circunferência associada e são a forma mais simples de representar um objecto. Os polígonos são formas geométricas convexas, ou seja, formas em que todos os seus ângulos são obtusos (com uma amplitude superior a 90º). No caso de um polígono ter sido definido de forma não convexa, esta forma é dividida em subformas de maneira a que todas elas sejam convexas. Duas outras regras que definem um polígono para este motor são o facto de este ter que ter três ou mais vértices e não poder ser oco. O terceiro tipo de formas que é possível definir são as arestas e são definidas por segmentos de recta. Estas são disponibilizadas para ser possível a criação de ambientes livre para o jogo, o que significa que se estiver a ser desenvolvido um jogo de plataformas, estas devem ser definidas como um conjunto de arestas e não como polígonos ou círculos. O quarto e último tipo de formas disponível neste motor são as cadeias que são grupos de segmentos de recta (arestas) interligadas entre si.

O módulo das dinâmicas de objectos é a parte mais complexa do motor e assenta sobre os dois módulos anteriores. Neste módulo os componentes mais impor-

tantes são as *fixtures* que contêm um conjunto de informação utilizada para descrever os corpos físicos no qual se inclui uma forma descritiva do objecto e as suas densidades, restituição e fricção. Este tipo de elementos pode pertencer a dois grupos distintos: corpos rígidos ou sensores.

Os corpos rígidos são a representação de todos os corpos físicos que podem ser designados por palpáveis, ou sejam, comportam-se como corpos reais com massa e volume determinados. Os sensores são corpos que têm um corpo virtual e que não geram qualquer tipo de reacção de colisão ao choque com corpos rígidos ou outros sensores. Uma das utilizações que se pode pensar para este tipo de corpos é a criação de pontos de controlo numa corrida de carros. Enquanto os carros eram representados por corpos rígidos para reagirem às colisões entre si, os pontos de controlo seriam representados por sensores para não geraram qualquer reacção física à colisão com os carros.

Os corpos definidos para cada objecto podem ser definidos por três tipos de corpos: corpo estático, corpo dinâmico e corpo cinemático. Os corpos estáticos são corpos que não se movem sobre qualquer circunstância e comportam-se como se tivessem massa infinita. É de realçar que este tipo de corpo não gera qualquer colisão quando em contacto entre assim nem quando entra em contacto com os corpos cinemáticos. Os corpos dinâmicos, como o próprio nome indica, são corpos cuja dinâmica é completamente simulada. Este tipo de corpos pode ser movido manualmente pelo utilizador ou pode mover-se devido a acções de forças aplicadas sobre eles. Este tipo de corpos geram colisões sempre que estiver em contacto com qualquer tipo de corpo deste sistema. Por fim, os corpos cinemáticos movem-se, na simulação, relativamente à sua velocidade, podendo esta ser alterada de duas formas, através da sua definição ou através de acções que a alterem como é o caso das colisões entre objectos. No que diz respeito à geração de colisões, este tipo de corpos apenas o faz quando em contacto com corpos dinâmicos.

2.2. Práticas

Após a análise às ferramentas que servem de base para o desenvolvimento de jogos para dispositivo iOS foi possível verificar que todos seguem a mesma ideia para a apresentação dos ecrãs de jogo, tendo essa ideia a base em dois conceitos essenciais: cena e camada. A cena é a representação de cada cenário de jogo, ou seja, é o contentor de todos os objectos que estão presentes em cada ecrã. As camadas são os subcontentores que armazenam e organizam os vários objectos. Nesta abordagem a cena aparece como o local onde são apresentados todos os objectos que no entanto precisam de ser organizados e agrupados de forma a que seja mais fácil dispô-los no ecrã, situação que é possível através da utilização de camadas. Desta forma é possível definir uma certa hierarquia na relação destes dois aglomeradores de objectos, enquanto os objectos são agrupados e organizados em camadas estas são por sua vezes agrupadas numa cena. Sempre que se pretende efectuar a actualização do jogo, basta actualizar a cena que tratará de propagar essa chamadas a todas as cenas, sendo estas responsáveis pela actualização dos objectos.

No que diz respeito ao processo de desenvolvimento de jogos, a tendência para por um conjunto de etapas que são descritas de seguida. A primeira etapa que se apresenta a quem pretende desenvolver um jogo é a criação de um conceito para o mesmo. Esta é talvez a etapa mais importante do processo, uma vez que é nesta fase que são definidas as formas como o utilizador interage com o jogo e a forma como se desenrola todas a acção durante o mesmo. Seguidamente à definição do conceito, é feita uma análise ao mercado por forma a verificar que jogos existentes podem ser englobados no conceito que se acabou de criar. Este ponto é importante pois permite analisar os pontos fortes e os pontos fracos dos produtos que já se encontram disponíveis para o público. Esta análise serve também para identificar as diferenças entre os produtos disponíveis e o produto que se pretende lançar por forma a saber se o conceito definido anteriormente é distinto o suficiente. A terceira etapa do desenvolvimento passa pela análise do público que se pretende atingir, por forma a perceber quais as características que o jogo deve ter para satisfazer as suas exigências e quais as formas preferenciais de interacção com o jogo por parte destes utilizadores. A etapa que se segue está relacionada com a implementação de um protótipo do jogo, ou seja, uma versão inacabada e quer servirá para efectuar alguns testes tanto a nível de jogabilidade como a nível de usabilidade. Esta versão do produto servirá também para distribuir por um grupo de utilizadores com as características identificadas na terceira fase por forma a receber indicações dos problemas que o jogo ainda tem e dos pontos fortes que os utilizadores admiram. Depois do período de testes é necessário realizar uma análise às respostas recebidas para

identificar as alterações que devem ser introduzidas por forma a que a versão final do jogo vá de encontro às especificações previamente definidas. Após terem sido analisadas essas repostas e identificadas as alterações a produzir iniciar-se-á a fase final da implementação onde são introduzidas as funcionalidades em falta bem como melhorar algumas falhas que tenham entretanto sido entretanto identificadas. A última etapa passe pela elaboração de um conjunto de testes, realizados internamente, por forma a verificar que todos os mecanismos implementados estão a funcionar correctamente e dentro do plano inicialmente delineado.

3. Metodologia

Esta secção está dividida em duas subsecções: abordagem de desenvolvimento e planeamento de actividades. A primeira subsecção descreve a abordagem de desenvolvimento escolhida e as razões que levaram a essa escolha, enquanto que a segunda subsecção apresenta o planeamento idealizado para o projecto assim como a versão real de como decorreu o mesmo.

3.1. Abordagem de desenvolvimento

Para o desenvolvimento deste projecto foi escolhida uma metodologia de desenvolvimento ágil, uma vez que o facto de o desenvolvimento de jogos ser altamente mutável permite tirar grande partido das principais características deste tipo de metodologias. Entre essas características tem especial destaque a capacidade de divisão das tarefas em pequenos incrementos, o que leva a que se possa realizar um planeamento a curto prazo. O facto de as metodologias ágeis darem azo à utilização de pequenas equipas de desenvolvimento leva a que a comunicação entre os membros da equipa seja feita cara-a-cara levando a que seja mais fácil aos elementos da equipa perceber e responder a dúvidas. O facto de este este projecto levar à necessidade de utilização de uma pessoa com capacidades gráficas faz com que esta vantagem também se torne importante no processo de desenvolvimento. Outra das características destas abordagens e que se verifica uma vantagem para este tipo de projectos é o facto de a medição do progresso do produto ser o seu funcionamento, ou seja, do final de cada tarefa deve resultar um produto funcional, mais completo e cada vez mais próximo da sua versão final.

Para este projecto a escolha caiu sobre a metodologia SCRUM, que apesar de ser uma metodologia desenvolvida para pequenas equipas de desenvolvimento permite a utilização de artefactos constituintes da sua estrutura por programadores que trabalhem sozinho, como são o caso do *Product Backlog* e do *Sprint Backlog*.

3.2. Planeamento de actividades

Para se poder efectuar uma gestão mais correcta de todas as tarefas incluídas neste projecto foi elaborado um diagrama de Gantt com o planeamento a ser dividido em duas fases (uma por cada semestre do projecto). Esta tarefa pretende organizar todos os passos do desenvolvimento assim como definir prazos para a sua conclusão.

Seguidamente são apresentados os planos iniciais, os planos elaborados após a reunião com o orientados e o diagrama real do desenvolvimento do projecto. É também efectuada uma análise comparativa dos vários diagramas assim como alguns comentários e explicações às diferentes existentes entre eles.

1ª versão

O plano inicialmente traçado para o primeiro semestre estava dividido em cinco fases. A primeira fase prendia-se com o levantamento do estado da arte, ou seja, a análise de motores de jogo disponíveis no mercado assim como a análise de algumas técnicas de desenvolvimento de jogo, com especial ênfase no desenvolvimento para dispositivos móveis. Pretendia-se igualmente analisar de forma cuidada o motor de jogo desenvolvido pela empresa por forma a posteriormente ser possível apontar as fragilidades da ferramenta e alguns melhoramentos que podiam ser realizados. A segunda fase foi orientada para efectuar uma análise comparativa dos vários motores analisados por forma a verificar quais as deficiências que o motor criado pela empresa evidenciava e efectuar o levantamento de algumas funcionalidades que se poderiam vir a revelar mais valias para esta ferramenta. A terceira fase tinha como objectivo tanto a implementação das novas funcionalidades que foram levantadas na fase anterior como efectuar melhoramentos a funcionalidades existentes. Para a quarta fase estava prevista a criação de um primeiro jogo que pudesse vir a tirar partido das novas funcionalidades implementadas. A quinta e última fase pensada para o primeiro semestre era a elaboração do relatório intermédio.

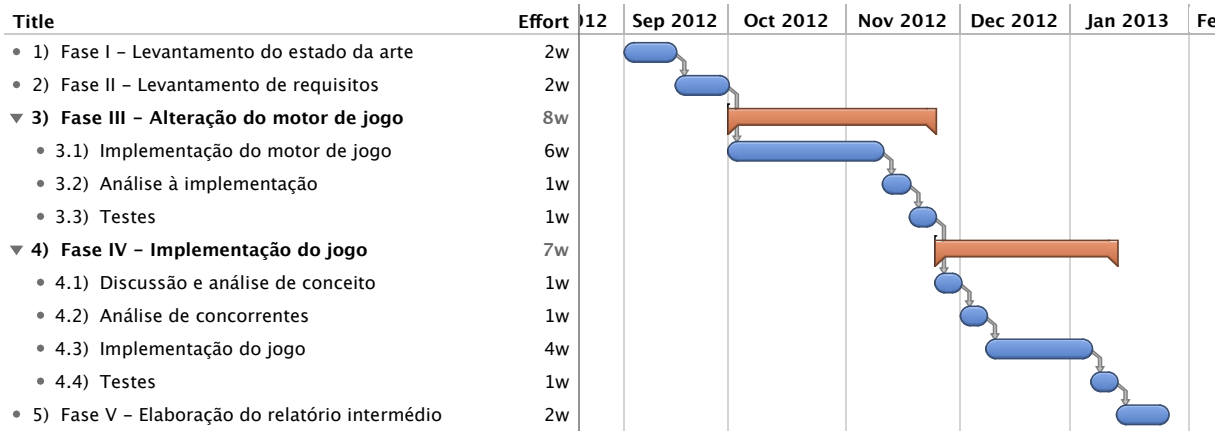


Figura 2 - Primeira versão do planeamento para o primeiro semestre do estágio

Para o segundo semestre foi planeado o desenvolvimento de três novo jogos para incluir no portefólio da empresa. O espaço reservado para este desenvolvimento foi de vinte semanas e estava dividido em três blocos (um para cada jogo). Cada um desse blocos era composto por quatro tarefas: discussão do conceito do jogo, análise de concorrentes disponíveis no mercado, implementação e testes (desempenho, usabilidade e jogabilidade). Para este semestre foi também reservado um período de três semanas para a elaboração do relatório final.

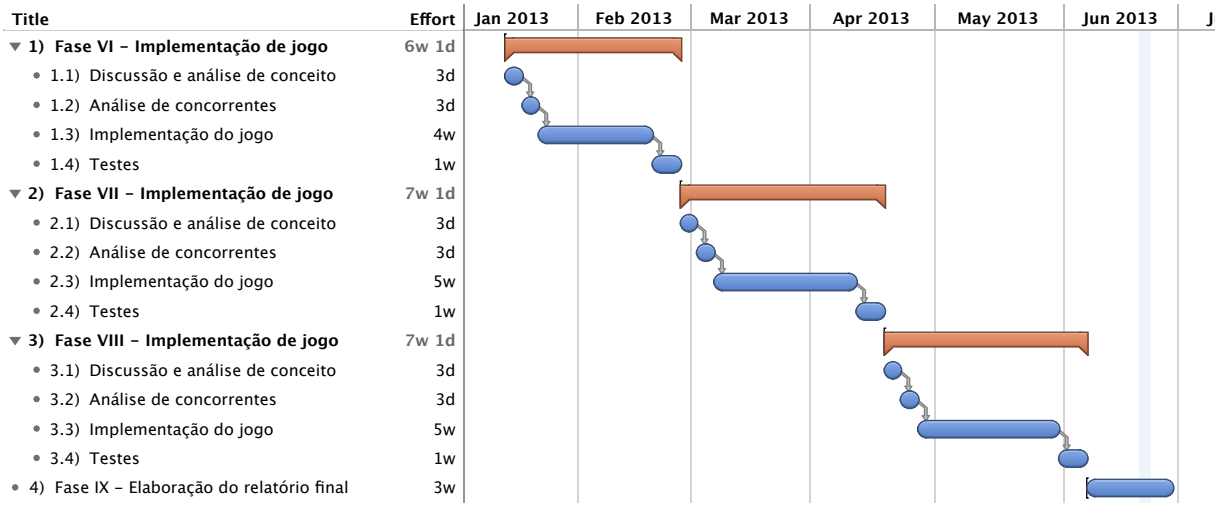


Figura 3 - Primeira versão do planeamento para o segundo semestre do estágio

2ª versão

Tal como foi referido anteriormente, o plano de trabalhos sofreu alterações significativas após a reunião com o orientador do Departamento, uma vez que, durante a mesma se chegou à conclusão que o plano de trabalhos inicialmente elaborado não era o mais indicado para este tipo de projecto. Após a análise das duas versões dos diagramas é possível notar grandes diferenças entre os dois planeamentos.

O planeamento inicialmente elaborado para o primeiro semestre sofreu um conjunto de alterações que passam a ser descritas de seguida. Enquanto que no primeiro planeamento era reservado um espaço de oito semanas para as alterações ao motor de jogo e sete semanas para a implementação de um primeiro jogo, na segunda versão são reservadas catorze semanas para efectuar as alterações ao motor não estando planeado o desenvolvimento de nenhum jogo para este período. A ideia por trás destas alterações foi o facto de ser necessário mais tempo para realizar os melhoramentos ao motor de jogo por forma a que estes fossem devidamente ponderados e implementados. Um outro ponto que levou a este novo planeamento foi o facto de na reunião com o orientados do Departamento se ter decidido que seria mais benéfico para o desenvolvimento do projecto que as alterações fossem efectuadas antes de se iniciar a implementação de novos jogos por forma a que não tivessem que ser efectuadas alterações de maior envergadura durante o processo de desenvolvimento dos mesmos.

No que toca ao planeamento para o segundo semestre, também foram realizadas algumas alterações. Durante a reunião foi possível perceber que o espaço inicialmente reservado para a criação de um jogo não era suficiente pois havia funções nesse desenvolvimento que não estavam a ser tidas em conta. Sendo assim, o planeamento para o segundo semestre prevê a criação de apenas dois jogos contrariamente aos três inicialmente planeados, tendo sido reservado um espaço de doze semanas para o desenvolvimento de cada um deles.

Para o desenvolvimento dos produtos propostos para este estágio escolheu-se uma abordagem que passou pela divisão do tempo de produção em cinco fases distintas: discussão do conceito, análise de mercado, análise de público alvo, implementação e teste. A primeira fase tem por objectivo chegar a um conceito para o jogo, o que significa que neste ponto é necessário definir de uma forma detalhada todo o modelo de jogo, estando incluído neste modelo a história que guia o desenvolvimento do jogo, os objectivos que se pretendem apresentar ao jogador, a forma como esses objectivos podem e devem ser atingidos bem como a forma como o utilizador poderá interagir com o jogo. Como resultado desta fase deve surgir uma ideia clara do que se pretende implementar, sendo esta ideia passível de ser alterada durante a fase de implementação uma vez que nesta fase podem surgir situações

não previstas durante a discussão do conceito e que podem levar a uma melhor qualidade do produto. Na segunda e terceira fases o objectivo é efectuar uma análise de mercado para o conceito que foi criado. Nestas fases há duas grandes perguntas que são centrais a toda a análise: o que há no mercado que seja semelhante ao nosso produto? e Quem são as pessoas que queremos a utilizar o nosso produto? Com as respostas para a primeira pergunta é possível encontrar os pontos fortes e fracos dos produtos já existentes no mercado assim como perceber o que valoriza o nosso produto relativamente aos concorrentes. A segunda pergunta dá indicações relativamente às características que o produto deve possuir para satisfazer o público-alvo. A quarta fase é talvez a fase mais importante de todo o processo de desenvolvimento, uma vez que é nesta fase que se constrói o produto idealizado nas fases anteriores. Como já foi referido, esta fase pode também levantar algumas questões que não foram tidas em conta na fase de criação de conceito, podendo essas questões levar a ajustes tanto no próprio conceito como também na forma como o jogador interage com a aplicação. A fase de implementação serve também para dar destaque a situações que não foram pensadas durante a idealização do produto assim como fazer sobressair erros de *design* que não foram visíveis durante a fase de conceptualização do produto. A fase final do ciclo de desenvolvimento do produto ºe a fase onde todas as funcionalidades são testadas e onde são feitas as verificações do produto de forma a garantir que o jogo cumpre os objectivos inicialmente estabelecidos. Durante esta fase, o produto é apresentado a um conjunto de utilizadores que irão testá-lo e dar conhecimento de problemas que detectaram no jogo, ou sugestões que pensem servir para melhorar o jogo.

Nas imagens seguintes são apresentadas as duas versões do planeamento para o primeiro semestre.

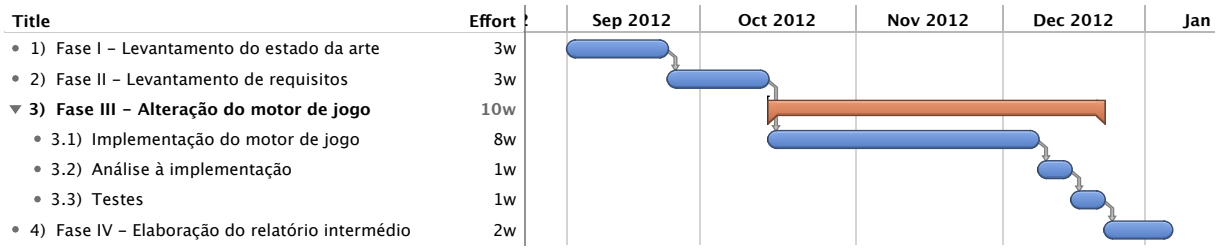


Figura 4 - Segunda versão do planeamento para o primeiro semestre do estágio

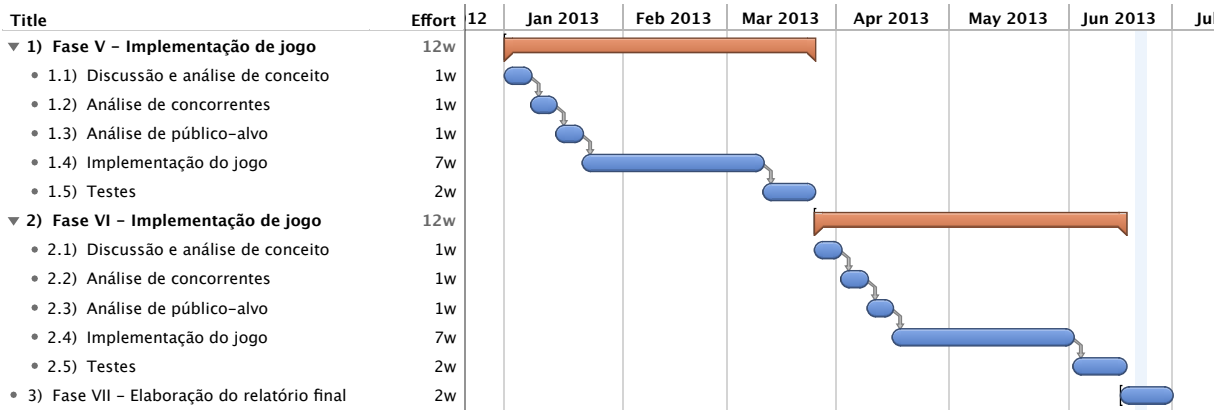


Figura 5 - Segunda versão do planeamento para o segundo semestre do estágio

Plano realizado

O plano idealizado para o primeiro semestre foi cumprido integralmente não havendo grandes desvios a assinalar. Relativamente ao plano do segundo semestre existiram alguns desvios, sendo o mais relevante o facto de não ter sido possível efectuar a implementação do segundo jogo, uma vez que o primeiro jogo que se implementou acabou por demorar mais tempo do que o inicialmente previsto. O facto de este jogo estar a ser desenvolvido em parceria com uma entidade externa levou a que este projecto tivesse que ser desenvolvido com grande cuidado uma vez que o produto final estaria sujeito a uma avaliação externa. O facto de este produto ter objectivos contratualmente definidos entre a empresa e a entidade externa levou a que este desenvolvimento não tivesse grande margem de erro, preferindo-se um maior tempo de implementação em vez de se arriscar um desenvolvimento rápido por forma a cumprir o planeamento idealizado para o segundo semestre. A facto de o desenvolvimento ter sido mais prolongado que o inicialmente previsto, levou a que as tarefas realizadas tenham sido efectuadas de uma forma mais ponderada bem como a análise efectuada foi também mais criteriosa e profunda.

Na imagem seguinte é possível verificar o plano realizado durante este segundo semestre.

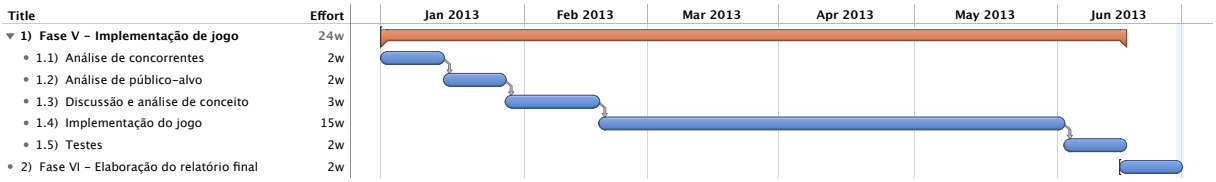


Figura 6 - Plano realizado no segundo semestre

4. Melhoramento do motor de jogo

Tal como foi definido no plano de trabalhos para este projecto, a primeira fase do projecto esteve relacionada com o melhoramento da ferramenta que serve de base ao desenvolvimento dos jogos da empresa. Como é possível avaliar pela tabela comparativa das ferramentas analisadas, a ferramenta que foi desenvolvida pela empresa encontra-se já num grau de maturidade relativamente aceitável, podendo no entanto sofrer alguns melhoramentos para aumentar o leque de funcionalidades oferecidas.

4.1. Levantamento de alterações

Antes de se iniciar o melhoramento da ferramenta foi efectuado o levantamento de um conjunto de melhorias e novas funcionalidades que podiam ser aplicadas ao motor de jogo por forma a torná-lo mais completo e versátil.

Após uma análise aprofundada ao motor de jogo e ao seu desempenho e tendo por base a análise comparativa efectuadas a todas as ferramentas analisadas foram identificadas algumas funcionalidades que podiam ser alvo de melhoramentos:

- alteração do sistema de partículas para passar a usar *Vertex Buffer Object*;
- alteração do motor para suportar as diversas resoluções de ecrã encontradas nos dispositivos iOS;
- melhoramento da leitura de ficheiros de imagem.

Com a comparação entre as diversas ferramentas analisadas foi também possível levantar um conjunto de funcionalidades que deveriam ser adicionadas ao motor de jogo da empresa por forma a que este aumente a sua qualidade. As funcionalidades levantadas foram as seguintes:

- implementação de um sistema que permita *in-app purchases*;
- implementação de um sistema de pontos âncora;
- implementação de um sistema básico de detecção de colisões.

De seguida são descritas em pormenor tanto as alterações efectuadas como as novas funcionalidades implementadas para melhorar o motoro de jogo.

Para a implementação de todas estas alterações foi elaborado um plano de trabalho para este produto tendo sido o trabalho necessário sido dividido em várias tarefas de menor dimensão através. O controlo desta tarefas foi realizado com recurso às

técnicas de controlo de desenvolvimento disponibilizadas pela metodologia SCRUM sendo utilizadas para este caso o *Product Backlog* e o *Sprint Backlog*. Para cada alteração que se idealizou para o motor de jogo foi criado um *sprint* contendo este tarefas mais detalhadas sobre cada mudança que foi efectuada no motor de jogo. Na tabela seguinte está apresentado o planeamento para estas alterações sendo possível verificar que apesar de em alguns casos a estimativa de tempo para algumas tarefas ter tido um desviou mínimo, houve um caso onde existiu um grande desvio, no caso dos testes finais às alterações, uma vez que a existência de testes preliminares no final de cada *sprint* permitiu que a fase final de testes não fosse tão exaustiva como o que estava planeado.

Tarefa	Estimado (h)	Real (h)	dia 1	dia 2	dia 3	dia 4	dia 5	dia 6	dia 7
Análise de limitações	**2**	**1.5**	1.5	0.0	0.0	0.0	0.0	0.0	0.0
Análise e avaliação de alternativas	**5**	**6.0**	6.0	0.0	0.0	0.0	0.0	0.0	0.0
Alteração do sistema de partículas	**8**	**7.5**	0.0	7.5	0.0	0.0	0.0	0.0	0.0
Testes preliminares	**1**	**1.0**	0.0	1.0	0.0	0.0	0.0	0.0	0.0
Sprint 1	**16**	**16.0**	**7.5**	**8.5**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Análise de limitações	**2**	**1.0**	1.0	0.0	0.0	0.0	0.0	0.0	0.0
Avaliação de soluções	**1**	**0.5**	0.5	0.0	0.0	0.0	0.0	0.0	0.0
Alteração do mecanismo de leitura de ficheiros de imagem	**4**	**2.0**	2.0	0.0	0.0	0.0	0.0	0.0	0.0
Testes preliminares	**1**	**1.0**	1.0	0.0	0.0	0.0	0.0	0.0	0.0
Sprint 2	**8**	**4.5**	**4.5**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Avaliação de mecanismos	**5**	**3.5**	3.5	0.0	0.0	0.0	0.0	0.0	0.0
Implementação de suporte para iPhone 3.5 polegadas	**4**	**4.5**	4.5	0.0	0.0	0.0	0.0	0.0	0.0
Implementação de suporte para iPhone 4 polegadas	**3**	**2.5**	0.0	2.5	0.0	0.0	0.0	0.0	0.0
Implementação de suporte para iPad	**4**	**3.5**	0.0	3.5	0.0	0.0	0.0	0.0	0.0

Tarefa	Estimado (h)	Real (h)	dia 1	dia 2	dia 3	dia 4	dia 5	dia 6	dia 7
Junção das varias implementações	**3**	**2.5**	0.0	2.5	0.0	0.0	0.0	0.0	0.0
Testes preliminares	**1**	**1.0**	0.0	0.0	1.0	0.0	0.0	0.0	0.0
Sprint 3	**20**	**17.5**	**8.0**	**8.5**	**1.0**	**0.0**	**0.0**	**0.0**	**0.0**
Leitura de documentação	**2**	**2.0**	2.0	0.0	0.0	0.0	0.0	0.0	0.0
Implementação de mecanismo In-App Purchase genérico	**5**	**4.5**	4.5	0.0	0.0	0.0	0.0	0.0	0.0
Testes preliminares	**1**	**1.0**	1.0	0.0	0.0	0.0	0.0	0.0	0.0
Sprint 4	**8**	**7.5**	**7.5**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Definição de mecanismos para localização dos pontos âncora	**1**	**1.0**	1.0	0.0	0.0	0.0	0.0	0.0	0.0
Alteração dos objectos para suporte de pontos âncora	**1**	**1.5**	1.5	0.0	0.0	0.0	0.0	0.0	0.0
Alteração das funções de manipulação de objectos	**10**	**11.5**	11.5	0.0	0.0	0.0	0.0	0.0	0.0
Testes preliminares	**3**	**2.0**	2.0	0.0	0.0	0.0	0.0	0.0	0.0
Sprint 5	**15**	**16.0**	**16.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Criação de objectos com caixas de colisão	**10**	**9.0**	9.0	0.0	0.0	0.0	0.0	0.0	0.0
Criação de gestor de colisões	**24**	**24.5**	0.0	8.0	7.5	9.0	0.0	0.0	0.0
Criação de mecanismos de alerta de colisões	**2**	**2.0**	0.0	0.0	0.0	0.0	2.0	0.0	0.0
Testes preliminares	**8**	**6.0**	0.0	0.0	0.0	0.0	6.0	0.0	0.0
Sprint 6	**44**	**41.5**	**9.0**	**8.0**	**7.5**	**9.0**	**8.0**	**0.0**	**0.0**
Análise à implementação	**16**	**15.0**	9.0	6.0	0.0	0.0	0.0	0.0	0.0
Testes	**24**	**15.5**	0.0	0.0	8.0	7.5	0.0	0.0	0.0
Sprint 7	**40**	**30.5**	**9.0**	**6.0**	**8.0**	**7.5**	**0.0**	**0.0**	**0.0**

Tabela 2 - Product Backlog para as alterações ao motor de jogo

4.2. Alteração de funcionalidades

Neste ponto do relatório estão descritas as funcionalidades que foram melhoradas ou implementadas de novo na ferramenta, a forma como estas foram desenvolvidas e a forma como foi validado o seu correcto funcionamento.

Sistema de partículas

Um sistema de partículas é um sistema que permite a representação de objectos que cujas superfícies não estão bem definidas como e o caso de fumos, fogos ou água.

Tal como foi referido no capítulo do estado da arte o sistema de partículas que foi inicialmente implementado era bastante rudimentar, uma vez que recorria a objectos normais para representar graficamente as partículas. Esta implementação apresentava algumas limitações em termos de desempenho, uma vez que se fosse necessário utilizar um sistema com um elevado número de partículas este sistema tornava-se bastante lento reduzindo drasticamente o desempenho do jogo.

Após uma análise à forma como o sistema tinha sido implementado decidiu-se avançar para a remodelação do sistema por forma a utilizar uma funcionalidade disponível no OpenGL que permite o carregamento de dados directamente para a memória do dispositivo de vídeo. Esta funcionalidade traz grandes vantagens, uma vez que todos os dados ficam alojados na memória do dispositivo de vídeo sendo acedidos mais rapidamente do que quando se encontram na memória do dispositivo móvel. Outra vantagem que este sistema apresenta relativamente ao anterior é o facto de não serem criadas todas as partículas de uma só vez, sendo criadas gradualmente durante o tempo de vida do emissor. O facto de apenas ser possível permitido associar uma textura a cada sistema de partículas torna-se uma limitação, apesar de não ser crítica, para este tipo de solução. No entanto esta limitação é facilmente ultrapassável com a criação de vários sistemas de partículas em situações em que seja necessário a utilização de mais que uma textura para representar o objecto.

Esta nova implementação, à semelhança do que sucedia com o sistema anterior, permite alterar algumas propriedades das partículas entre as quais se destacam a posição, a cor e o tamanho. Para efectuar a validação deste novo sistema foram realizados alguns testes em que se executam os dois sistemas de partículas com dados semelhantes, nomeadamente no que diz respeito ao número de partículas do sistema e à textura utilizada. Para esta validação foram tidos em conta os tempos de actualização dos sistemas assim como o tempo necessário para a placa de vídeo reproduzir os sistemas no ecrã.

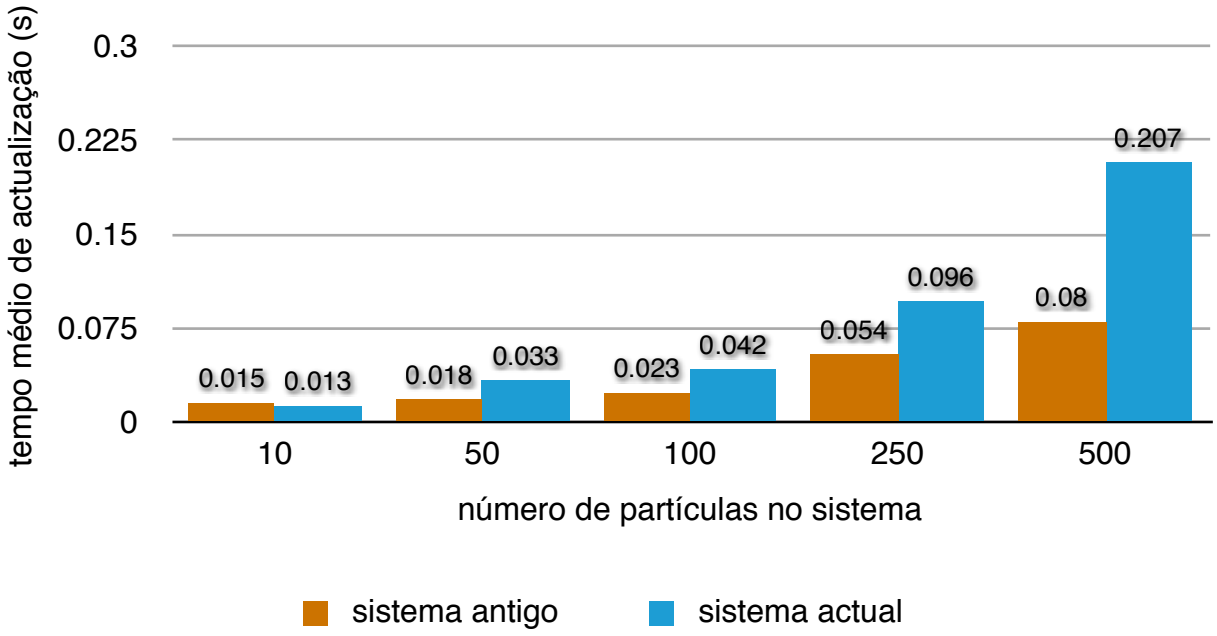


Figura 7 - Gráfico do tempo médio de actualização dos sistemas de partículas

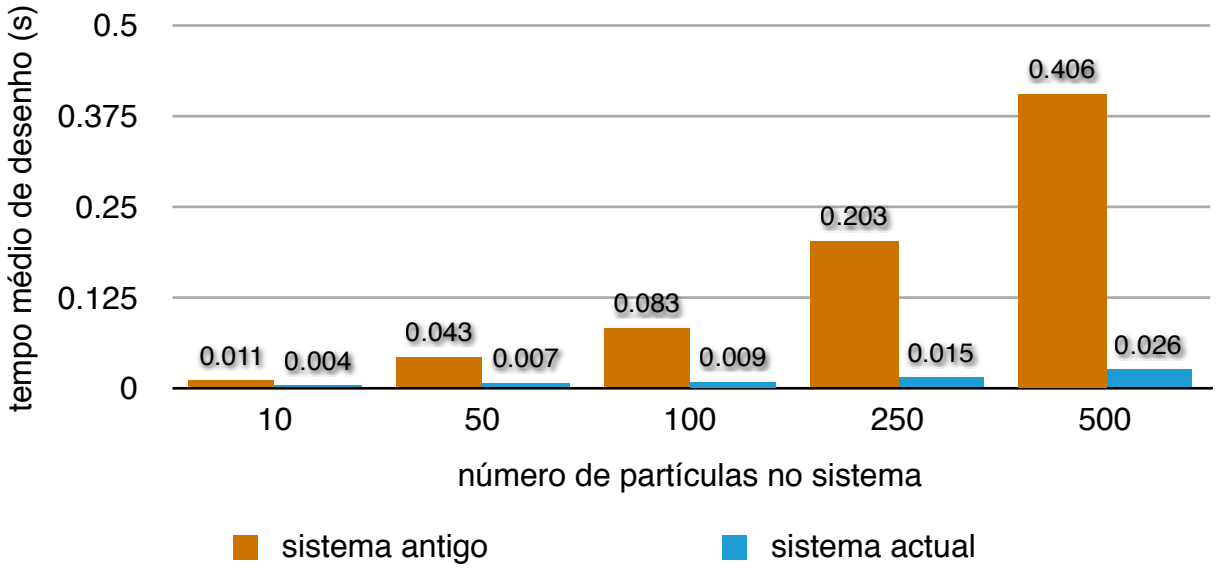


Figura 8 - Gráfico do tempo médio de desenho dos sistemas de partículas

Ao analisarmos os dois gráficos anteriores, podemos verificar que relativamente à actualização dos sistemas de partículas a opção com recurso aos *Vertex Buffer Object* se torna um pouco mais lenta que a solução implementada anteriormente, devendo-se este facto ao fluxo de informação entre a memória do dispositivo móvel e a memória do dispositivo de vídeo, que faz com que os valores de cada partículas sejam actualizados no dispositivo de vídeo. No entanto essa quebra de desempenho é largamente compensada quando se efectuam as operações de desenho dos sistemas.

Melhoramento da leitura de imagens

O sistema de ficheiros de imagem estava implementado com recurso a uma biblioteca externa que tratava do carregamento dos dados do ficheiro para memória. Esta biblioteca funcionava perfeitamente nesta função, mas deixava bastante a desejar no que diz respeito ao seu desempenho. Com esta preocupação em mente, decidiu fazer-se alguma pesquisa para verificar se existia alguma forma de realizar este processo com melhor desempenho que aquele que era obtido actualmente. Após algum tempo de pesquisa, foi possível encontrar uma solução que tinha passado completamente despercebida aquando da implementação do motor de jogo e que passava pela utilização das funções disponibilizadas pelo OpenGL para leitura deste tipo de ficheiros. Apesar da alteração ter sido mínima, o impacto que teve no desempenho foi bastante significativo como é possível verificar no gráfico seguinte.

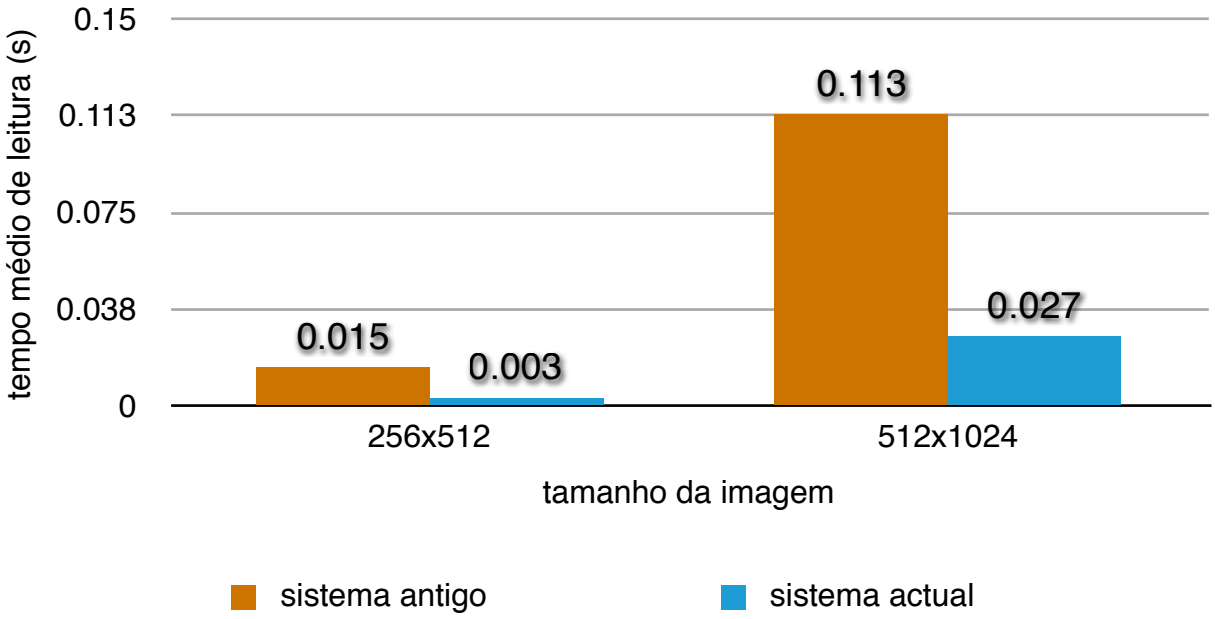


Figura 9 - Gráfico comparativo da leitura de ficheiros de imagem

Suporte multi-dispositivo

Uma funcionalidade necessária para se produzirem aplicações universais é a utilização de código capaz de lidar com as várias resoluções encontradas nos diversos dispositivos iOS. Na versão anterior do motor de jogo, este suporte era feito de forma pouco limpa uma vez que existia uma função para definir a área de desenho para cada tipo de dispositivo. A solução que se pretendia implementar era a utilização de uma função única que fosse capaz de lidar com os diversos dispositivos.

A principal dificuldade que se encontrou para criar uma solução genérica para este problema está relacionada com o facto dos dispositivos iOS disponíveis no mercado terem vários tipos de ecrã, com *aspect ratio* distintos entre si.

Figura 10- Comparação dos tamanhos de ecrã dos diferentes dispositivos

Além de existir uma evidente diferença no tamanho do ecrã, o facto que levantou mais problemas foi a não existência de um relacionamento directo entre os vários dispositivos, uma vez que cada modelo apresenta um *aspect ratio* distinto. No caso do iPad, o *aspect ratio* é de 1.33:1, a nova versão do iPhone (4 polegadas) apresenta um ecrã 16:9 enquanto que a versão anterior (3.5 polegadas) apresenta um *aspect ratio* de 16:10 (1.5:1).

Para resolver o problema apresentado por esta situação implementou-se uma solução que passa pela utilização de um ecrã virtual com o *aspect ratio* que permita uma melhor adaptação nos restantes dispositivos, sendo neste caso o utilizado pelo iPhone de 3.5 polegadas. A utilização deste ecrã virtual fazia com que a área de desenho fosse desenhada de forma igual em todos os dispositivos. No entanto a colocação desta área de desenho nos restantes dispositivos (iPad e iPhone de 5 polegadas) levava a que o ecrã não fosse completamente preenchido.

Para preencher este espaço vazio procurou-se o menor valor que seria necessário acrescentar ao ecrã virtual para que este abrangesse todo o ecrã. No caso do dispositivo utilizado ser um iPad, aumentou-se proporcionalmente a área de desenho até uma das dimensões (vertical ou horizontal) atingir o limite do ecrã, o que aconteceu quando o factor de escala era 1.07, sendo o tamanho do ecrã virtual nesta altura de 1024x684 *pixels*. Uma vez que a resolução do iPad é 1024x768 *pixels* faltava ainda preencher uma faixa de 84 *pixels*. Essa faixa foi dividida em duas de igual tamanho (42 *pixels*) sendo cada uma adicionada a um dos limites do ecrã virtual por forma a efectuar todo o preenchimento do ecrã. Para o iPhone de 5 polegadas a solução utilizada foi a mesma, não havendo contudo a necessidade de escalar o ecrã virtual, dado que a menor dimensão de ambos os dispositivos é igual (640 *pixels*). Este facto levou a que apenas fosse necessário acrescentar 2 faixas de 88 *pixels* (1136-960 = 176 *pixels*) nos limites superior e inferior da área de desenho.

As alterações efectuadas no motor de jogo localizaram-se no módulos de gráficos, sendo que também foi necessário refazer o submódulo correspondente ao sistema de partículas. Na imagem seguinte é possível verificar a localização exacta das alterações implementadas.

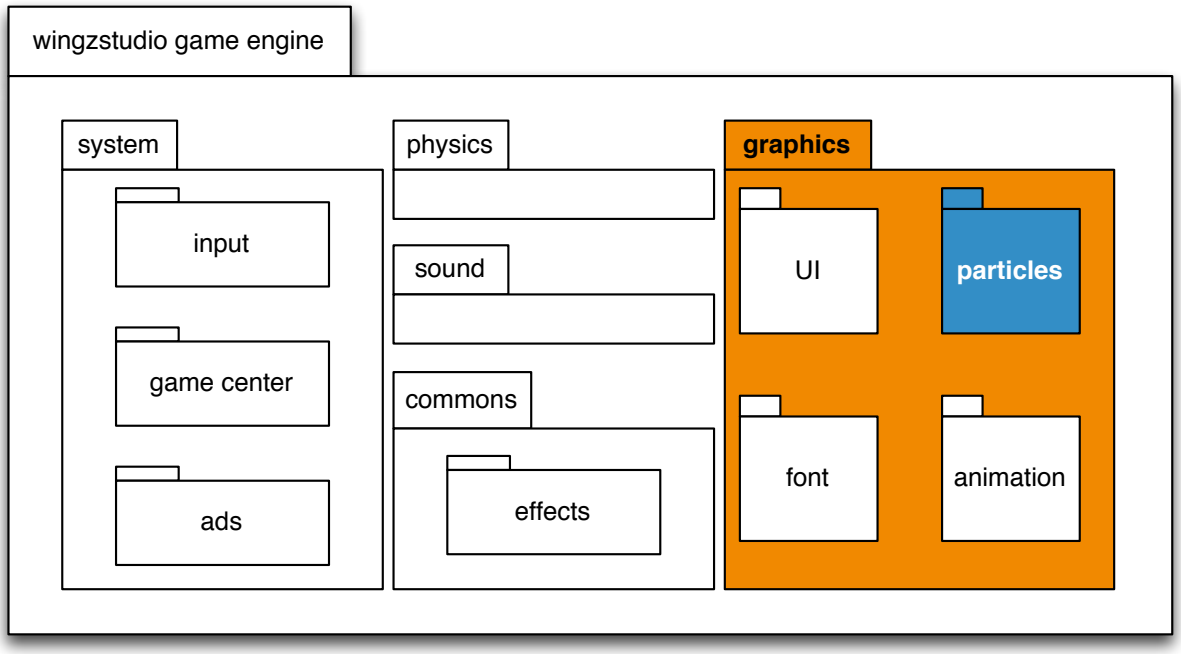


Figura 11 - Localização das alterações efectuadas no motor de jogo

4.3. Implementação de novas funcionalidades

Neste ponto do relatório estão indicadas e descritas as novas funcionalidades que foram implementadas no sistema com o objectivo de o tornar mais completo.

In-app purchases

O desenvolvimento de jogos para iOS tem que seguir as tendências de mercado e essas tendências podem mudar sem qualquer aviso prévio. A primeira tendência que foi detectada no mercado foi a publicação de duas aplicações distintas para o mesmo produto. Uma versão servia de demonstração e era distribuída sem custos e a outra versão correspondia à versão completa do produto tendo esta que ser adquirida por um valor definido pelos produtores. A tendência que se seguiu foi a criação de apenas uma aplicação, que tinha todas as funcionalidades disponíveis e que estava associado a um custo monetário para quem a pretende ter ao dispor. Seguidamente os produtores de aplicações iniciaram a tendência de publicar aplicações sem custos, mas com restrições no uso. Para que essas restrições fossem retiradas, era necessário efectuar a compra de produtos dentro da aplicação (as denominadas *in-app purchases*).

Neste momento a tendência é a de distribuir aplicações completas tendo o utilizador a possibilidade de comprar elementos que agregam valor à aplicação. No caso específico dos jogos esta tendência está associada à compra de dinheiro virtual da própria aplicação para que o utilizador possa utilizar o mercado interno para adquirir produtos para o seu jogo. Um exemplo deste caso é a compra de moedas virtuais que podem depois ser utilizadas para comprar super-poderes para atribuir às personagens do jogo.

Vendo a empresa a necessidade de adicionar esta capacidade ao motor, passou-se à implementação de um novo módulo para a ferramenta e fosse capaz de realizar as operações necessárias para a utilização deste mercado.

Este módulo fornece ao programador a possibilidade de receber a lista dos produtos disponíveis para a aplicação, efectuar a compra dos produtos, verificar que produtos já foram comprados e efectuar a reposição de produtos que já tenham sido comprados e que sejam perpétuos, ou seja, só precisam de ser adquiridos uma vez. Este caso é importante porque o utilizador da aplicação pode eliminá-la do dispositivo, no entanto tem que ser capaz de recuperar as suas compras se instalar a aplicação noutro dispositivo.

A implementação de um sistema de *in-app purchases* é semelhante para todos os criadores de aplicações diferindo apenas nos identificadores dos produtos disponíveis para compra. O protocolo necessário para efectuar uma compra dentro da

aplicação é bastante simples e encontra-se especificado pela Apple. Este protocolo consiste numa troca de mensagens entre a aplicação e os servidores da Apple disponibilizados para este efeito, estando este módulo dependente do funcionamento desses servidores. A imagem seguinte ilustra a troca de mensagens necessária para efectuar uma compra através do serviço de *in-app purchases*.

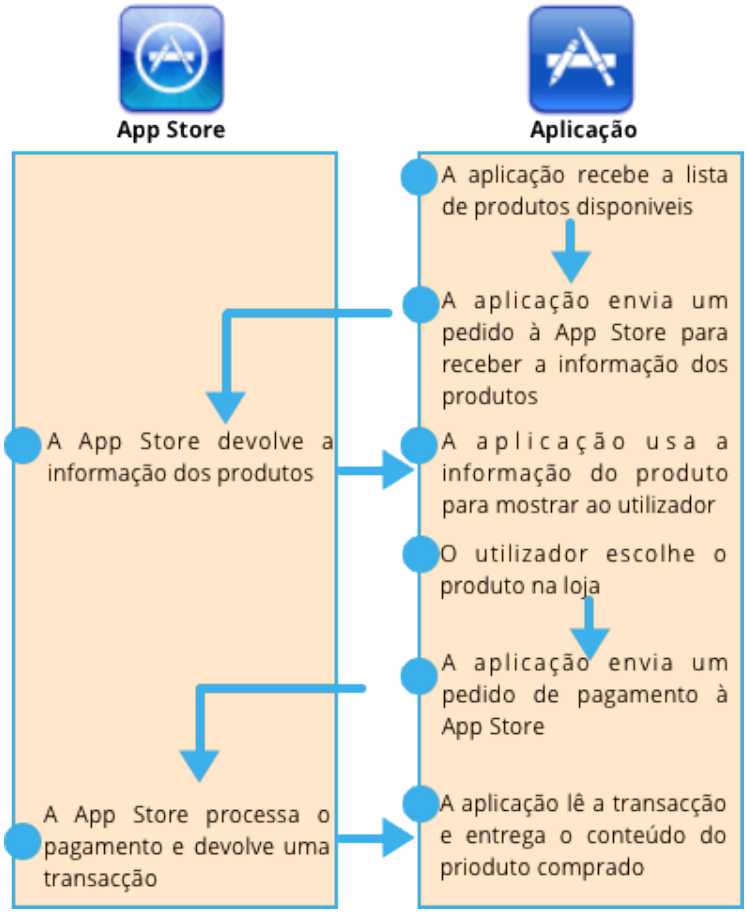


Figura 12- Fluxo de mensagens para compra de um produto através de in-app purchases

Pontos âncora

Tal como foi verificado na análise do estado da arte, a utilização de pontos âncora traz consigo um conjunto de vantagens no que toca ao desenvolvimento de jogos, pelo que foi decidido implementar este sistema na ferramenta que tem vindo a ser desenvolvida pela empresa.

A implementação desta funcionalidade foi realizada com a adição de um propriedade ao objecto mais básico, e que permite definir a localização do ponto âncora do mesmo. Contrariamente ao que foi definido por algumas das ferramentas analisadas, esta implementação apenas permite colocar o ponto âncora em cinco localizações distintas do objecto: centro, canto superior esquerdo, canto superior direito e ambos os cantos inferiores. Esta implementação levou também a que tivessem que ser actualizadas todas as funções de manipulação de objectos por forma a poderem lidar com esta nova característica.

Figura 13 - Localizações possíveis para os pontos âncora

Detecção de colisões

Apesar de o sistema estar preparado para a utilização de um motor de física que contém um módulo de detecção de colisões, a implementação desta funcionalidade pode parecer um pouco incoerente uma vez que estaria a duplicar uma funcionalidade já existente. No entanto não é esse o caso, uma vez que o objectivo que se pretende atingir com esta implementação é a redução do processamento necessário para detectar colisões simples entre objectos, pois o módulo disponibilizado pelo motor de físicas apresenta bastante complexidade. Por este motivo optou-se pela criação de um módulo mais simples que lidasse apenas com as colisões.

Para estas funcionalidades ficarem disponíveis forma implementados dois conjuntos de classes distinto. O primeiro representa as formas que serão utilizadas para dar um corpo de colisão aos objectos, enquanto que o segundo ficará responsável por todo o processamento dessas formas e consequente sinalização de colisões sempre que houver a detecção dessas situações.

A primeira ideia que surgiu para as formas representativas dos objectos passava pela utilização de duas formas geométricas no sistema (círculos e rectângulos). No entanto e após alguma experimentação desta solução verificou-se que esta solução podia levantar alguns problemas dado que eram detectadas colisões mesmo quando os objectos não estavam em contacto.

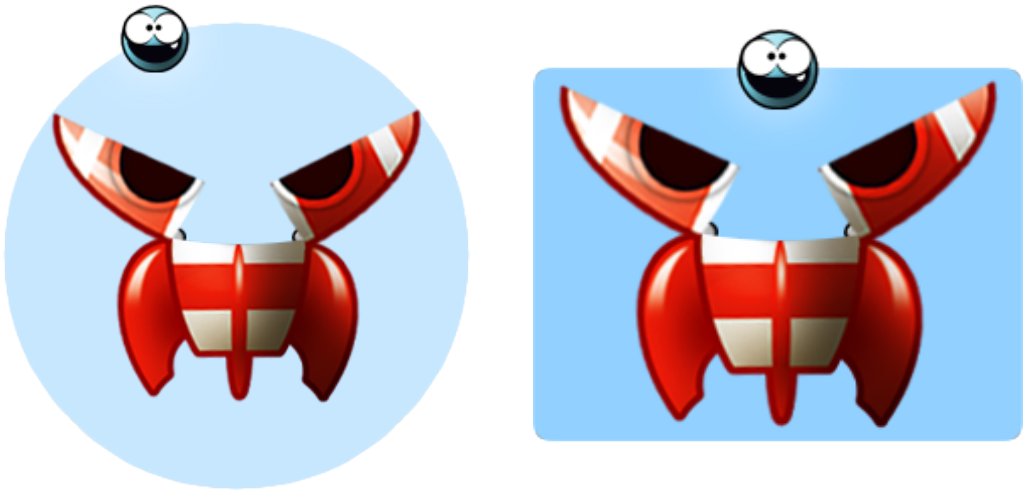


Figura 14 - Situação de detecção errada de colisões

Nesta imagem é possível verificar duas situações em que são erradamente detectadas colisões, uma vez que os dois objectos não estão em contacto entre si. Chegou-se então à conclusão que a melhor ideia seria manter as duas formas definidas anteriormente mas apenas para definir a forma principal do corpo de colisão, sendo utilizado um polígono (definido por um conjunto de pontos) como forma se-

cundária o que levaria a que a colisão fosse detectada com mais precisão. A primeira forma é utilizada apenas para verificar se existe a possibilidade de dois corpos colidirem entre si. No caso de existir uma possibilidade de colisão será utilizada a forma secundária, definida por um polígono, para verificar se a colisão é real. No caso de um objecto não ter definido a forma mais específica, será assumido que a sua caixa de colisão é a forma mais genérica.

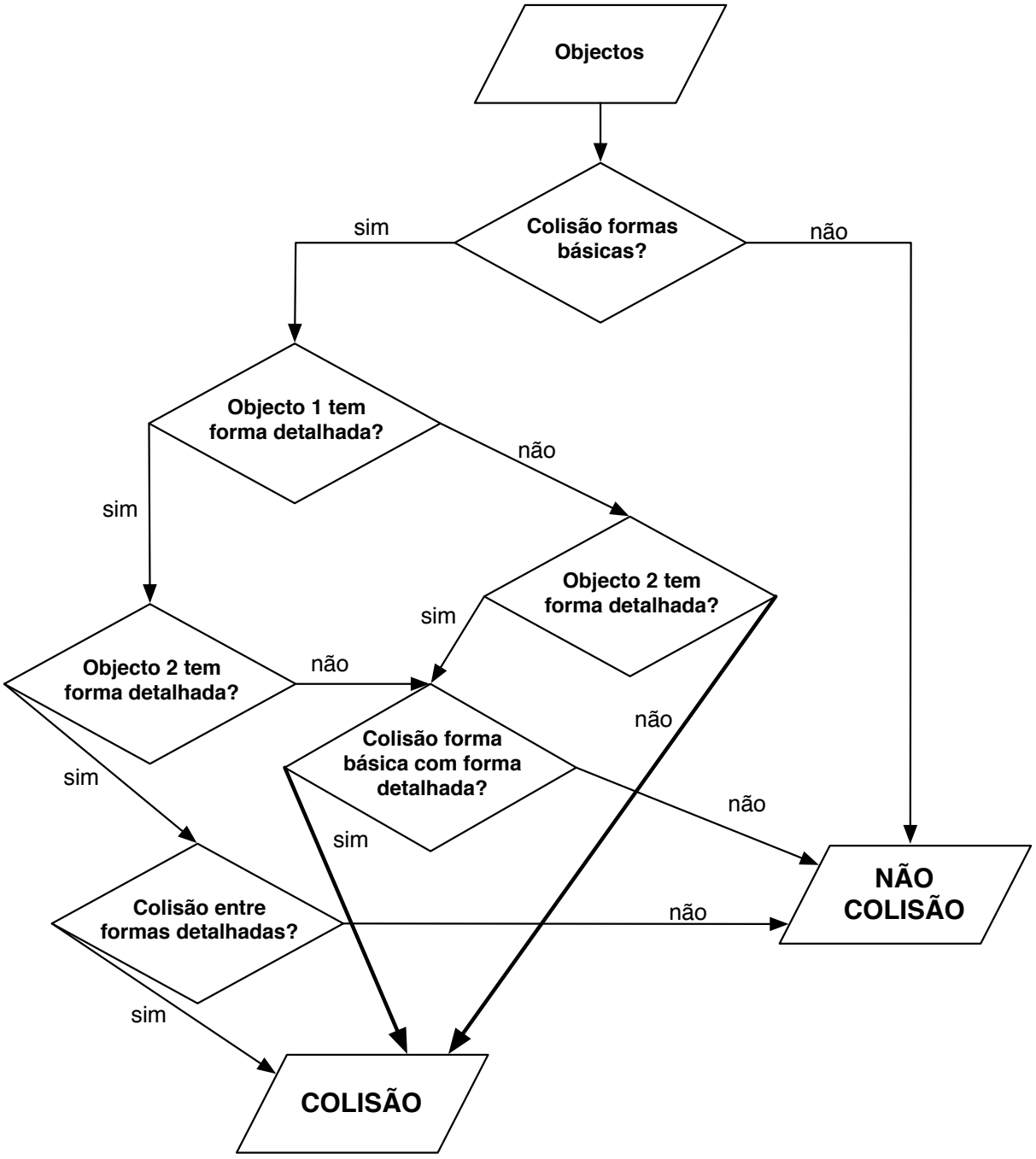


Figura 15 - Algoritmo utilizado na detecção de colisões

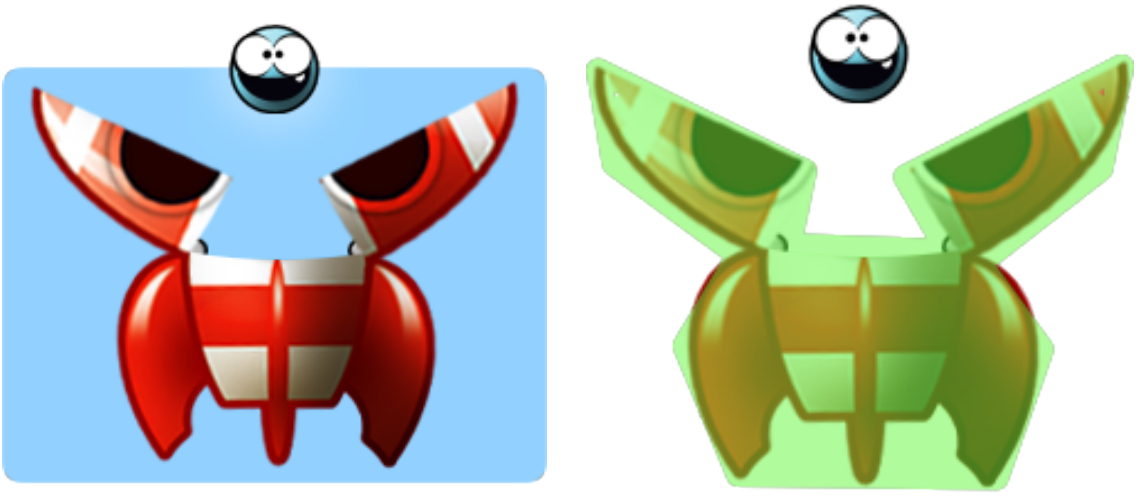


Figura 16 - Utilização de duas formas distintas para detecção de colisões

Na figura anterior é possível verificar o processo que leva à detecção de uma colisão. Através da utilização da forma genérica do objecto foi despertada uma possível colisão, no entanto o recurso à forma mais específica veio revelar que não existiu nenhum choque entre os objectos. Através da comparação das duas imagens podemos concluir que esta segunda versão é muito mais fiável que a versão anterior.

A introdução destas novas funcionalidades levou a uma alteração na estrutura do motor de jogo tendo sido adicionados dois novos módulos e alterado um. Na imagem seguinte é possível verificar o estado do motor de jogo após terem sido efectuadas estas alterações, estando assinalados a verde os módulos que foram criados e a amarelo os módulos que foram alterados.

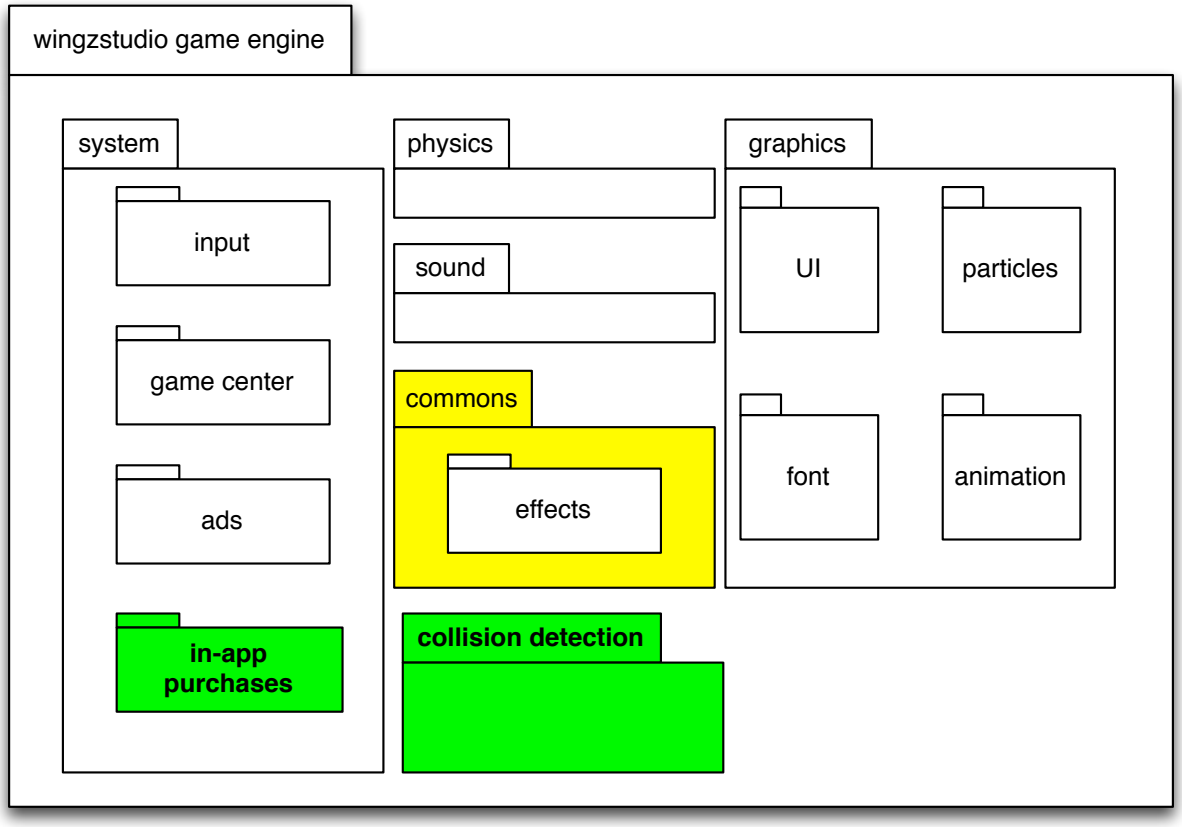


Figura 17 - Estrutura do motor de jogo depois da implementação das novas funcionalidades

5. Falcao VS Aliens

Criado não só para aumentar o portefólio da empresa mas também como resultado de uma parceria estabelecida entre a wingzstudio e uma empresa gestora de direitos de imagem de atletas mundiais, o jogo Falcao VS Aliens foi um dos produtos que esteve a ser desenvolvido durante a segunda metade do estágio.

Visto ser um projecto de parceria, o conceito para este jogo estava já limitado às soluções apresentadas pela empresa à entidade externa sendo que foram apresentados dois conceitos para o jogo, um recorrendo ao estilo arcade, mais concretamente ao conceito Arkanoid, e outro que levaria à criação de um jogo que se iria incluir na categoria de puzzle. Depois da apresentação dos conceitos à entidade externa, a decisão recaiu sobre o conceito arcade, uma vez que se sabe que este tipo de jogos gera uma grande aceitação por parte do público.

Existindo neste jogo uma personagem principal decidiu-se elaborar uma história que servisse de pano de fundo a todo o desenvolvimento de jogo. A elaboração desta história teve por base os dois tipos de personagens que irão constituir este jogo: a personagem principal, Falcao e as personagens secundárias, um grupo de extra-terrestres. Após interceptarem uma transmissão de um jogo de futebol, os extra-terrestres decidiram visitar o planeta Terra para aprender mais sobre este desporto. No entanto, o facto de não conhecerem as regras fez com que estes personagens de feitio difícil não fossem bem recebidos pelos humanos, o que levou a que se apoderassem de estádios espalhados por todo o planeta ao mesmo tempo que afugentavam os espectadores. Vendo o seu desporto favorito ser destruído Falcao, o herói do jogo, decide começar uma longa aventura com o objectivo de recuperar os estádios controlados pelos extra-terrestres. Para isso terá que vencer os extra-terrestres num jogo sem regras e resistir a todas as estratégias utilizadas por estas criaturas de mau feitio.

Após ser definida a história que serviria de base ao desenvolvimento do jogo, o processo de implementação foi divido em quatro fases distintas. A primeira fase a ser realizada foi a análise de concorrentes disponíveis no mercado com o objectivo de retirar ideias que pudessem ser úteis para trazer um acréscimo de qualidade ao jogo. Na segunda fase passou-se à avaliação do público alvo para este produto, tendo esta fase o objectivo de avaliar as características que o jogo deveria apresentar para captar a atenção dos jogadores que se pretendem trazer para o jogo. A terceira fase foi a fase de concepção do jogo, ou seja, a fase onde se definiu quais os objectivos do jogo, qual o seu desenvolvimento e qual o caminho que o jogador teria que percorrer no jogo. Como forma de simular todo este processo, e para evitar imple-

mentação desnecessária, recorreu-se à técnica de prototipagem em papel. Uma vez que este projecto é bastante centrado na interacção com o utilizador, esta técnica é uma grande mais valia, uma vez que permite a simulação de todos os processos de interacção sem ter que ser necessário implementar os mecanismos necessários para simular esse processos. Apesar de ser um método estático de simulação, esta técnica permite verificar se os modelos de navegação estão a funcionar de forma conveniente, permitindo também verificar se a localização dos elementos interactivos é proveitoso para o bom desenvolvimento do jogo. Além do benefício de evitar um processo de implementação que possa ter que vir a ser alterado, esta técnica permite também ir efectuando modificações à disposição dos elementos no ecrã de uma forma simples, permitindo uma poupança de tempo no caso de estes testes serem efectuados com recurso à implementação das soluções. A quarta fase foi a implementação do jogo tendo por base o protótipo elaborado na fase anterior. Nesta fase foram implementadas todas as funcionalidades desejadas para o jogo, entre as quais se destacam a mecânica de jogo e a navegação do jogador entre dos vários ecrãs idealizados no processo de concepção.

5.1. Análise de mercado

Antes de se iniciar a implementação do jogo foi feita uma análise a jogos semelhantes que se encontram disponíveis no mercado por forma a efectuar um levantamento de algumas características que façam com que estes jogos sejam procurados pelo público. Além das mais valias que estes jogos possam apresentar foram também procurados pontos fracos que pudessem ser explorados. De seguida é apresentada uma análise a dois dos jogos disponíveis no mercado.

Breakout: Boost

Breakout: Boost é um jogo desenvolvido pela Atari para os dispositivos iOS. Tal como os restantes jogos deste estilo, o objectivo deste jogo passa pela destruição de blocos com uma bola. Para evitar que a bola se perca, o jogador controla uma barra localizada no fundo ecrã, utilizando-a para redireccionar a bola.

O facto de ter sido desenvolvido pela Atari, um dos maiores nome da indústria dos jogos últimos 40 anos é sem dúvida um dos maiores pontos fortes deste jogo. No entanto não é o único, sendo também de destacar o facto de o jogador ter a possibilidade de ajustar a velocidade com que a bola se desloca e consequentemente aumentar o valor de cada bloco destruído. Esta funcionalidade permite ao jogador ajustar a dificuldade durante do jogo em vez de ter uma dificuldade pré-definida, permitindo assim um maior dinamismo ao jogo. Outro ponto forte deste jogo é o facto de dar bastantes possibilidades ao jogador de perder a bola de jogo, uma vez que permite que o jogador utilize um total de dez bolas antes de perder.

A utilização de uma visão horizontal e simultaneamente uma barra lateral para mostrar informação faz com que a área de jogo seja bastante reduzida, o que leva a que esta opção se torne num ponto fraco. A forma como é ajustada a velocidade da bola é também uma das falhas deste jogo, uma vez que ao ser oferecida a possibilidade de o jogador ajustar a velocidade com apenas um toque no ecrã pode levar a que isso aconteça inadvertidamente e que o jogador saia prejudicado com isso.

Smash

Smash é um jogo apresentado por *Magma Mobile Games* e que é mais uma versão de um jogo baseado no Arkanoid. À semelhança dos restantes, também neste caso o objectivo é destruir blocos com uma bola enquanto o jogador tenta evitar que a bola se perca através da utilização de uma barra para redireccionar a bola.

Durante a análise a este jogo, foi possível encontrar alguns pontos fortes, de entre os quais se destacam três, a possibilidade de escolha entre dois modos de jogo, desafio ou *arcade*, a forma como se desenrola o modo *arcade* e a possibilidade de o jogador começar o modo de desafio em qualquer dos níveis já jogados. A possibilidade de o jogador poder escolher entre dois modos de jogo torna-se uma mais valia pelo facto de oferecer ao jogador duas experiências de jogo completamente distintas. O facto de o modo *arcade* permitir ao jogador jogar sem ter um final de nível definido, só perde caso a pilha de blocos atinja um determinado número de linhas ou através da perda de todas as bolas disponíveis, faz com que o jogador tenha uma experiência de jogo mais fluída e com menos interrupções podendo esta situação tornar-se num ponto fraco caso a duração da sessão de jogo se torne demasiado prolongada. A escolha de iniciar o jogo em qualquer dos níveis já jogados é um ponto a favor deste jogo pois evita que o jogador tenha a necessidade de jogar um grande número de níveis para chegar ao nível que pretende. Além de o modo *arcade* poder ter uma longa duração, também os sons usados neste jogo são uma fraqueza, uma vez que a utilização de sons demasiado agudos faz com que o jogador opte por desligá-los, não usufruindo de um ambiente propício para a jogabilidade. Também a representação gráfica da barra controlada pelo utilizador foge um pouco ao usual neste tipo de jogos, podendo ser um ponto contra a utilização deste jogo.

5.2. Análise do público-alvo

Outra etapa antes de se avançar para a implementação do jogo foi a análise do público que se pretendia atingir com este jogo. Esta análise é bastante importante pois permite definir que características devem ser incluídas no jogo por forma a que este gere procura no mercado.

O objectivo deste jogo é ser um jogo para ser utilizado por pessoas de todas as idades, estando o jogo incluído na categoria de jogos casuais, ou seja, jogos que são utilizados em momentos onde o jogador está num momento de pause, como são exemplo as viagens de transportes públicos, as esperas para as consultas ou os intervalos no trabalho.

Apesar de o jogo ser destinado a todo o tipo de pessoas, irão existir dois grupos que será representativos da maioria dos jogadores: crianças e pessoas com idade entre os 20 e os 30 anos. De acordo com estas características foram elaborados dois perfis que têm como objectivo perceber quais os objectivos destes utilizadores durante a utilização do jogo.

Samuel
5 anos

Frequenta uma escola privada e todos os dias o pai ou a mãe vão buscá-lo e levá-lo. Só gosta das viagens de carro quando os pais o deixam jogar no iPad. Gosta de jogos com personagens divertidas e que sejam visualmente apelativos.

José
Contabilista - 27 anos

Vive nos subúrbios, mas na trabalha numa empresa no centro da cidade. Todos os dias apanha o metro para ir até ao escritório passando as viagens a jogar no iPhone. Gosta particularmente de jogos desafiantes e onde se possa superar a ele próprio.

Figura 18 - Perfis de jogadores para o jogo Falcao VS Aliens

Com a análise destes perfis foi possível chegar à conclusão que o jogo deve ser visualmente apelativo ao mesmo tempo que apresenta algum desafio ao utilizador para fazer que com que os jogadores se sintam satisfeitos com o jogo. É também possível perceber que enquanto um grupo de utilizadores apenas está interessado na diversão que o jogo pode causar, outro grupo está interessado no desafio que o jogo lhe apresente.

5.3. Design

Para dar a possibilidade ao jogador de viver esta aventura foi necessário redesenhar um pouco o conceito do Arkanoid, que consiste na destruição de blocos com uma bola ao mesmo tempo que o jogador controla uma barra para evitar que a bola saia do ecrã. Este redesenho foi pensado com o objectivo de criar um jogo que não fosse apenas mais uma versão deste conceito. Uma das primeiras ideias que surgiu, e uma vez que o tema do jogo é futebol, foi a colocação de uma baliza na zona oposta àquela onde se encontra a personagem central do jogo, o que levaria a que o jogador tivesse que marcar um golo para passar ao próximo nível.

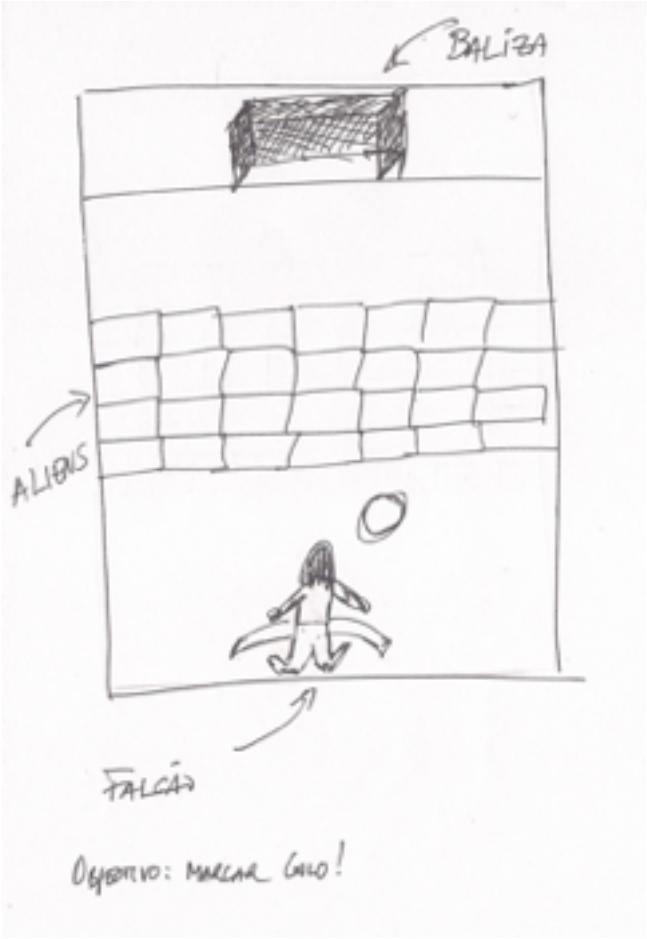


Figura 19 - Rascunho do ecrã de jogo com a localização da baliza

Apesar de esta solução parecer ideal, pouco tempo foi necessário para descobrir um problema que retiraria todo o prazer de jogo ao jogador. O facto de a baliza, objectivo que se pretende atingir, estar completamente desprotegida, tornava fácil ao jogador atingi-la e completar o nível sem grande dificuldade. Com este problema em mente procuraram-se soluções que pudessem resolver, ou pelo menos atenuar, este problema, sendo no entanto necessário que estas soluções se enquadrassem

no conceito definido para o jogo. A primeira solução pensada e também a que mais sentido fazia era a colocação de um extraterrestre que tivesse como função a defesa da baliza. Uma outra solução seria a colocação de uma barreira (parede? muro?) que evitasse que o jogador marcasse um golo sem concluir uma determinada tarefa que levasse a que a barreira fosse destruída.

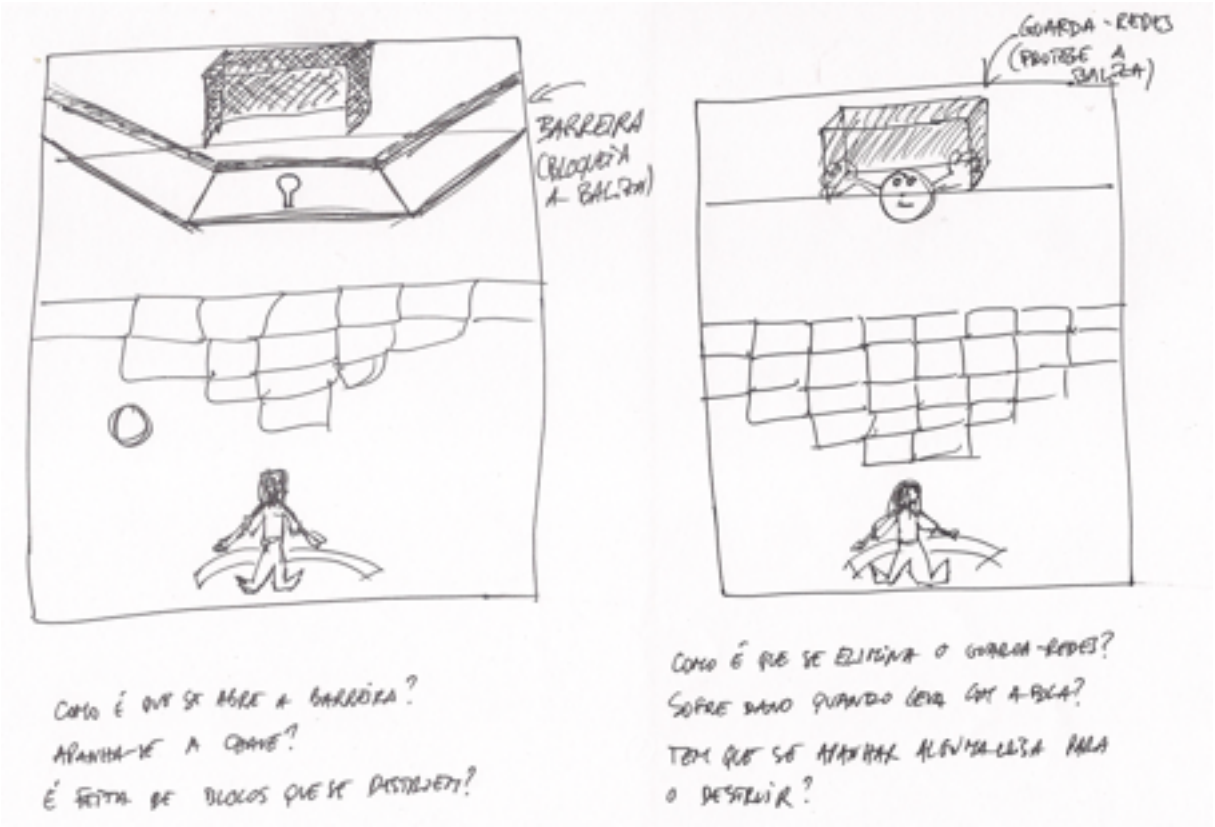


Figura 20 - Rascunho da solução de protecção da baliza

Uma vez que ambas as soluções pareciam aceitáveis, optou-se por deixar a decisão de qual utilizar para depois de verificar os resultados que cada uma iria produzir após alguns testes à jogabilidade com cada um destes obstáculos.

Após ter sido definido o objectivo para cada nível passou-se à análise da forma como o utilizador iria controlar a personagem e quais as acções que poderia aplicar sobre ela. A acção mais básica que o jogador tem sobre a personagem é o controlo do seu movimento, ou seja, o jogador deve ser capaz de controlar livre e eficazmente a personagem por forma a conseguir atingir o objectivo de cada nível. Sendo que o jogo está a ser desenvolvido para dispositivos iOS e estes apenas possuem uma forma de interacção entre o utilizador e o dispositivo que é o ecrã táctil, o que levou a que qualquer das formas de controlo que se pensassem tinham que ter em conta não só as vantagens desta tecnologia bem como as suas limitações.

Controlo da personagem

Após análises das várias formas que poderiam ser utilizadas para controlar a personagem, conclui-se que para este tipo de jogo apenas três métodos eram viáveis: arrastamento do dedo sobre o ecrã, utilização de botões ou utilização do acelerómetro. A análise destas três soluções levou à conclusão que a utilização do acelerómetro poderia não ser indicada para este caso, uma vez que é necessário um controlo preciso da personagem, o que não é possível com este mecanismo, dado que para obter este comportamento seria necessário que o jogador efectuasse movimentos suaves sob pena de o movimento da personagem não ser o ideal. Além desta limitação, o facto de o jogador ter que estar sempre a mover o dispositivo poderia ser outro factor negativo para esta solução, uma vez que o jogador poderia em algumas situações não ter uma visualização completa do ecrã de jogo, ou em situações extremas não conseguir de todo ver o que se estava a passar no ecrã. Os testes permitiram também verificar algumas limitações da solução que utiliza botões para controlar o jogador. Nesta fase foi possível verificar que a utilização de botões levaria a que a área de jogo ficasse demasiado reduzida, uma vez que para que o controlo da personagem fosse o correcto, os botões teriam que ser suficientemente grandes para que o utilizador pudesse controlá-los de forma eficiente. Apesar de ser possível utilizar esta solução sem a apresentação visual dos botões, sendo o utilizador informado previamente que cada metade do ecrã correspondia ao movimento para cada um dos lados, com os testes verificou-se que tal como a solução que utiliza botões visuais esta solução iria levar a uma área de jogo bastante reduzida, uma vez que o jogador teria a necessidade de ter dois dedos prontos para interagir em cada instante, não podendo ser colocado qualquer tipo de informação na zona que acabaria por ser tapada pelos dedos do utilizador.

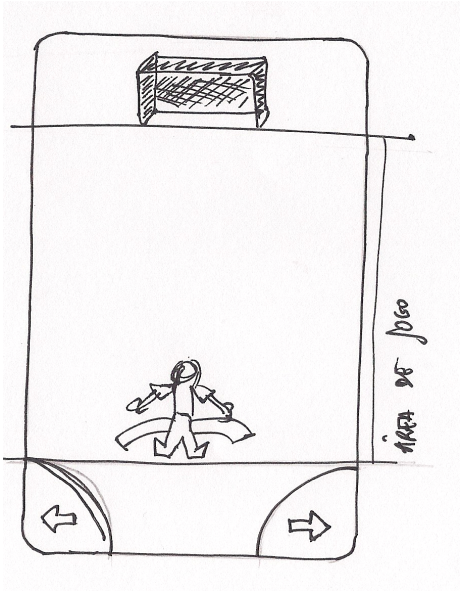


Figura 21 - Simulação da utilização de botões para o controlo da personagem

Localização da informação

Estando definido a forma como se desenvolve o jogo e como o jogador irá controlar o jogador restava apenas definir a localização da informação necessária ao jogador, como é o caso da pontuação actual, da melhor pontuação e do número de bolas restantes. Ao contrário do que é normal neste tipo de jogo a solução de colocar a informação de jogo numa barra lateral não é conveniente, uma vez que a reduzida dimensão do ecrã levaria a que o espaço disponível para a área de jogo ficasse bastante reduzida. Uma das soluções pensadas passou pela colocação desta informação no topo do ecrã, mas após a simulação desta solução iria ocupar uma grande quantidade de espaço originando uma reduzida área de jogo. A solução final acabou por ser a divisão da informação em duas barras, uma localizada no topo outra no fundo do ecrã. No topo ficaria a informação mais relevante para o jogador tais como as pontuações, actual e melhor, e o resultado do jogo, indicador do progresso realizado pelo jogador enquanto que no fundo ficaria informação que o jogador só precisa de ver quando não está a jogar, como é o caso do número de bolas restantes.

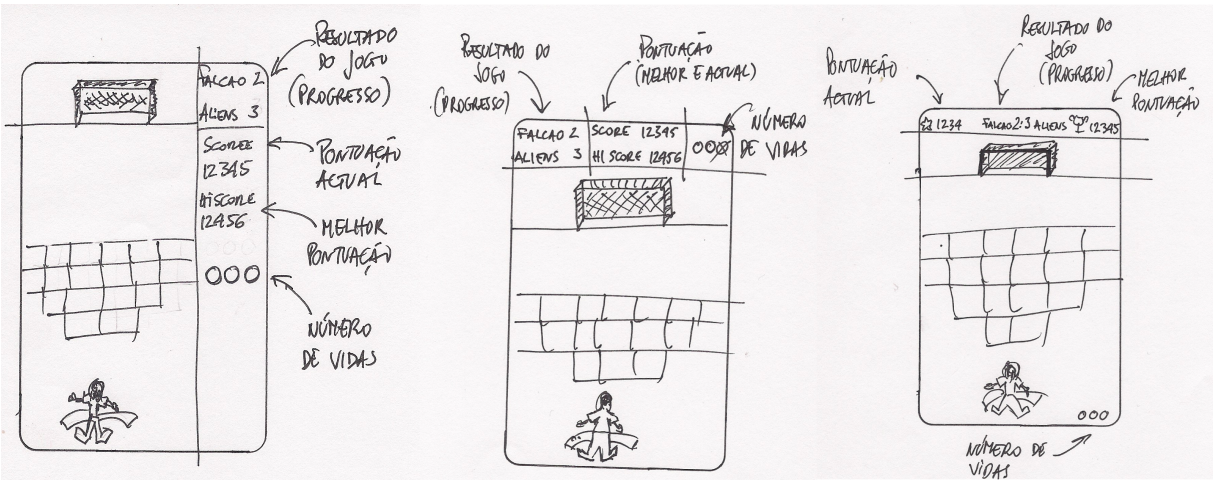


Figura 22 - Simulação da colocação das informações no ecrã de jogo

Definição de ecrãs

Para qualquer jogo que seja idealizado é necessário que exista uma boa organização das tarefas que o jogador possa desenvolver dentro da aplicação. Com esse objectivo as tarefas são divididas em cada ecrã, sendo que cada um destes organiza as tarefas que se encontram relacionadas entre si. Este jogo não é excepção e de seguida são apresentados os ecrãs que foram pensados para este jogo, assim como a função que cada um irá realizar.

Antes mesmo de se iniciar o desenho dos ecrãs que iriam compor o jogo e os testes à sua usabilidade foram definidas algumas regras no que diz respeito à navegação entre os ecrãs de jogo assim como à sua concepção. Para este jogo, optou-se por criar uma navegação simples entre ecrãs, ou seja, num ecrã não existirão muitas opções de mudança para outro ecrã. Esta opção foi tomada como forma de proteger o jogador, para que este após algumas mudanças entre ecrãs não se sentisse perdido ao ponto de sentir alguma frustração devido ao facto de não conseguir chegar onde pretende. Pela mesma razão foi também definido um número máximo de opções que cada ecrã poderia oferecer ao jogador, tendo esse valor sido definido em cinco opções.

O primeiro ecrã a ser desenhado, foi o ecrã principal, uma vez que o jogador necessita de um ponto onde iniciar a sua participação no jogo. Para este ecrã estão guardadas as opções de navegação para os restantes ecrãs do jogo. Deste modo, a partir deste ecrã o jogador tem a capacidade de navegar para os ecrãs mais importantes da aplicação como são o ecrã de jogo e o ecrã das opções, onde podem ser efectuadas algumas configurações ao jogo. Além destas duas opções existe também a possibilidade de o jogador se dirigir ao ecrã onde pode verificar que prémios já conquistou no decorrer do jogo. A quarta opção disponível neste ecrã servirá como entrada para a listagem de pontuações, onde o jogador pode comparar os seus resultados com os resultados dos restantes jogadores. A apresentação destes resultados será um sobreposição no ecrã principal, não sendo criado nenhum ecrã específico para esse fim.

Outro ecrã que foi criado para este jogo, e como já foi referido anteriormente é o ecrã de jogo, sendo este ecrã o local onde se prevê que o utilizador irá passar a maior parte do tempo enquanto estiver a interagir com a aplicação. Neste ecrã irá existir apenas uma opção para o utilizador, sendo esta a pausa do jogo. Sempre que o utilizador pausar o jogo, será apresentada uma camada de sobreposição que irá oferecer a opção de retomar o jogo ou sair para o menu principal, sendo que esta segunda hipótese fará com que todo o progresso realizado até ao momento seja descartado. Como forma de salvaguardar o estado do jogo, sempre que o utilizador escolher a opção que o faz retornar ao menu principal, ser-lhe-á exibida uma men-

sagem onde este é questionado se tem a certeza que pretende perder o progresso e regressar ao menu principal. Esta solução previne que o utilizador perca todo o jogo realizado até ao momento por um toque não intencional no botão.

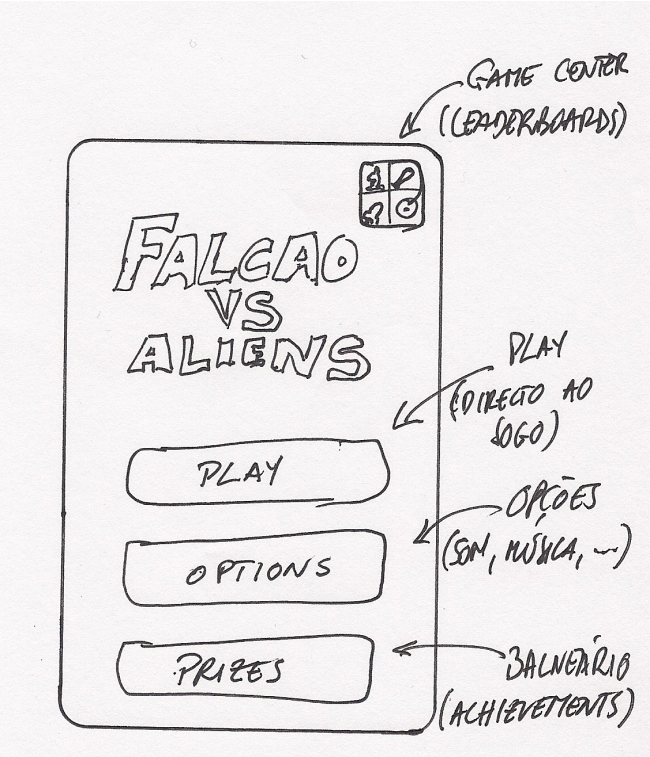


Figura 23 - Simulação do ecrã principal do jogo

Para o ecrã de opções, estão guardadas as configurações de alguns factores do jogo. Neste caso, e visto tratar-se de um jogo para dispositivos móveis, o jogador pode configurações que se limitam ao estado da música e dos efeitos sonoros. Para estes dois componentes do jogo, é possível ligá-los e desligá-los conforme a opção do jogador. Uma vez que este jogo funciona num conceito onde são guardados os melhores resultados do jogador, assim como todo o progresso que este já fez no jogo, é também possível ao jogador repor o estado original do jogo, voltando a bloquear os estádios que já tinham sido desbloqueados assim como levar os extre-terrestres a reconquistar os estádios que já tinham perdido para a personagem principal. Neste ecrã encontra-se ainda uma ligação para a lista de pessoas que estiveram por trás do desenvolvimento deste jogo.

Um ecrã que poderá gerar algum interesse no jogador é o ecrã onde são apresentados os prémios que já conquistou ao longo do jogo. Neste ecrã estarão presentes todos os prémios que são possíveis de conquistar com a conclusão de pequenas tarefas que não interferem na forma como o jogo se desenrola, no entanto os prémios que ainda não foram conquistados encontram-se representados por uma máscara que apenas dá ideia da sua forma, não dando qualquer indicação do seu aspecto visual. Neste ecrã poderão também existir ligações que levam o utilizador para as páginas *web* de empresas que possam vir a publicitar a sua marca neste produto. Tal como nos restantes ecrãs, existe também a possibilidade de o jogador retroceder para o ecrã anterior, neste caso o ecrã principal, por forma a ser possível navegar pelas restantes partes do jogo.

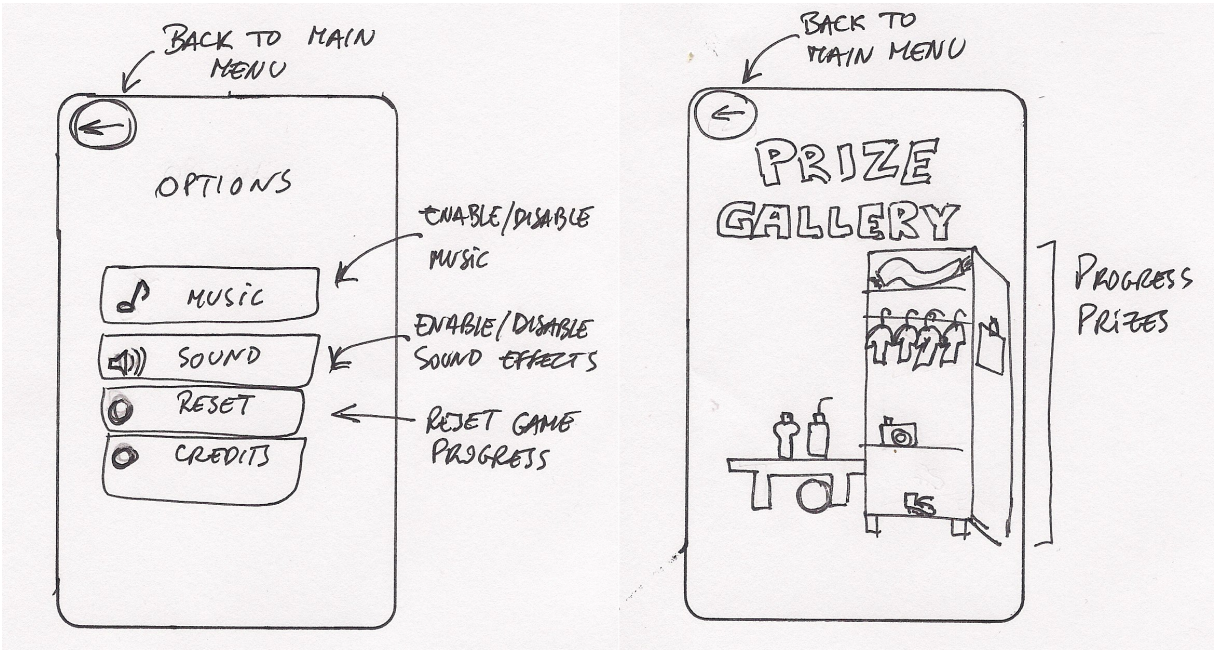


Figura 24 - Simulação do ecrã de opções e do ecrã de prémios.

Progresso no jogo

Dado que este jogo é baseado no progresso e constituído por níveis, e como foi também referido no ponto anterior foi elaborado um ecrã onde o utilizador possa ver o seu progresso no jogo, podendo também avaliar quanto é que lhe falta para chegar ao final do jogo. Neste ecrã, e além das características já referidas anteriormente, o utilizador tem a possibilidade de navegar para dois ecrãs: o ecrã de jogo e o ecrã principal. Sempre que o jogador pretende navegar para o ecrã de jogo, basta-lhe tocar em qualquer ponto do ecrã, com excepção da localização do botão de saída para o menu principal, e será encaminhado para o ecrã de jogo, sendo apresentado um novo nível de cada vez que isto venha a acontecer. No caso de o jogador pressionar o botão de saída para o menu principal, ser-lhe-á apresentada uma camada que servirá para este atestar da sua vontade de regressar ao menu principal, mesmo sabendo que o seu progresso será descartado. Se o jogador se encontrar neste ecrã após ter perdido todas as vidas que tinha durante o jogo, não lhe será possível regressar ao ecrã de jogo, assim como também não lhe será apresentada nenhuma informação quando activar o botão de regresso ao ecrã de jogo.

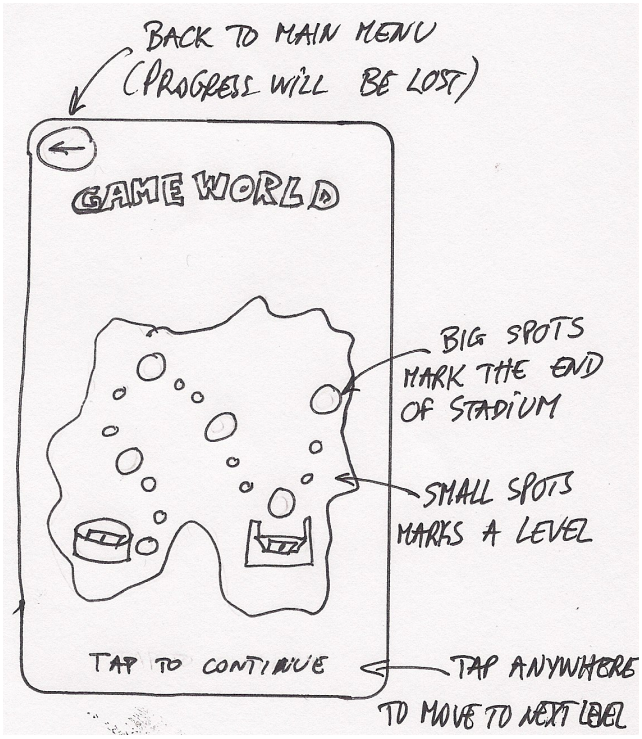


Figura 25 - Simulação do ecrã de progresso no jogo

Navegação entre ecrãs

Além destes ecrãs e uma vez que em cada estádio, o jogador se encontra perante um resultado diferente, sempre que são completados os três níveis de cada estádio será apresentado um ecrã onde o utilizador terá a possibilidade de verificar qual o resultado obtido neste estádio. É também possível ao utilizador visualizar a pontuação que fez até ao momento assim como a sua pontuação máxima em todo o jogo. Em termos de navegação, é apenas permitido ao utilizador a navegação para o ecrã de progresso.

Na imagem seguinte é possível verificar a navegação permitida ao utilizador, sendo visíveis as regras que levam a essa navegação. Sendo que destas regras a mais importante é a confirmação de que o utilizador pretende navegar para um ecrã mesmo que isso implique a perda do progresso que realizou até esse momento.

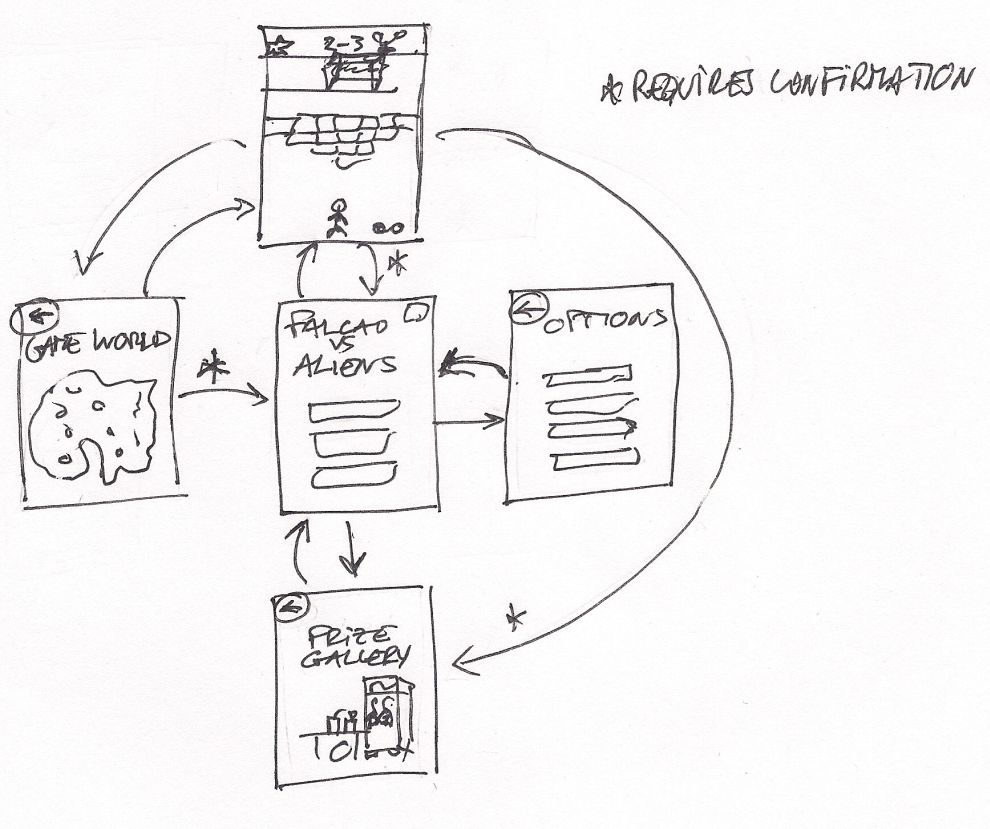


Figura 26 - Esboço do fluxo entre ecrãs do jogo

5.4. Desenvolvimento

A fase de desenvolvimento começou pela divisão do trabalho realizado num conjunto de tarefas de menor dimensão que permitisses uma melhor organização do projecto. Desta forma era possível ter um registo mais fiel do trabalho que já tinha sido desenvolvido e do que faltava ainda realizar. Apesar desta divisão, as tarefas ainda apresentavam alguma complexidade e pouco detalhe relativamente ao esperado pelo que foram divididas em subtarefas que definiam objectivos mais específicos. De seguida é apresentada a organização de tarefas para este projecto:

- **Implementação da mecânica de jogo**
    - movimento da personagem principal (Falcao)
    - comportamento dos inimigos (Aliens)
        - aliens estáticos
        - aliens dinâmicos (movem-se)
        - aliens ofensivos (disparam)
        - aliens guarda-redes
        - criação de bónus
        - criação de estrelas
    - tratamento de colisões
        - colisão da bola com os aliens
        - colisão da bola com o guarda-redes
        - colisão da bola com a baliza
        - colisão da bola com a barreira
        - colisão da bola com outro objectos
        - colisão dos bónus com o Falcao
        - colisão dos tiros com o Falcao
    - leitura de níveis
    - transição entre níveis
    - funcionamento da barreira
- **implementação do ecrã de progresso**
- **implementação do menu principal**
- **implementação do menu de opções**
- **implementação da sala de prémios**
- **implementação da navegação**

Como forma de controlar o tempo utilizado na realização das tarefas que se definiram como essenciais para o projecto, foi elaborado um *Product Backlog* que se encontra especificado na tabela abaixo. Pela avaliação da tabela é possível verificar que o tempo estimado para a realização de cada tarefa não apresenta grandes diferenças relativamente ao tempo que foi necessário para concluir essas mesmas tarefas.

Tarefa	Estimado (h)	Real (h)	dia 1	dia 2	dia 3	dia 4	dia 5	dia 6	dia 7
Movimento da personagem com botões	**10**	**9.0**	8.0	1.0	0.0	0.0	0.0	0.0	0.0
Movimento da personagem com acelerómetro	**10**	**10.0**	0.0	4.0	6.0	0.0	0.0	0.0	0.0
Movimento da personagem com arrastamento do dedo	**10**	**9.0**	0.0	0.0	0.0	6.0	3.0	0.0	0.0
Testes preliminares	**10**	**9.5**	1.0	2.0	1.5	1.0	4.0	0.0	0.0
Sprint 1	**40**	**37.5**	**9.0**	**7.0**	**7.5**	**7.0**	**7.0**	**0.0**	**0.0**
Inimigo estático	**8**	**7.0**	7.0	0.0	0.0	0.0	0.0	0.0	0.0
Inimigo dinâmico	**12**	**11.0**	0.0	8.0	3.0	0.0	0.0	0.0	0.0
Inimigo ofensivo	**12**	**9.5**	0.0	0.0	4.5	5.0	0.0	0.0	0.0
Inimigo guarda-redes	**8**	**6.0**	0.0	0.0	0.0	2.0	4.0	0.0	0.0
Sprint 2	**40**	**33.5**	**7.0**	**8.0**	**7.5**	**7.0**	**4.0**	**0.0**	**0.0**
Criação de estrelas	**10**	**8.0**	7.0	1.0	0.0	0.0	0.0	0.0	0.0
Criação de bónus	**20**	**19.5**	0.0	8.0	8.5	3.0	0.0	0.0	0.0
Sprint 3	**30**	**27.5**	**7.0**	**9.0**	**8.5**	**3.0**	**0.0**	**0.0**	**0.0**
Colisão da bola com os aliens	**20**	**21.0**	8.0	7.0	6.0	0.0	0.0	0.0	0.0
Colisão da bola com o guarda-redes	**12**	**12.0**	0.0	0.0	3.0	9.0	0.0	0.0	0.0
Colisão da bola com a barreira	**8**	**7.5**	0.0	0.0	0.0	0.0	7.5	0.0	0.0
Sprint 4	**40**	**40.5**	**8.0**	**7.0**	**9.0**	**9.0**	**7.5**	**0.0**	**0.0**

Tarefa	Estimado (h)	Real (h)	dia 1	dia 2	dia 3	dia 4	dia 5	dia 6	dia 7
Colisão da bola com a baliza	**16**	**14.5**	7.5	7.0	0.0	0.0	0.0	0.0	0.0
Colisão da bola com outros objectos	**16**	**15.5**	0.0	2.0	8.5	5.0	0.0	0.0	0.0
Colisão da bola com o fundo do ecrã	**8**	**7.5**	0.0	0.0	0.0	4.0	3.5	0.0	0.0
Sprint 5	**40**	**37.5**	**7.5**	**9.0**	**8.5**	**9.0**	**3.5**	**0.0**	**0.0**
Colisão dos bónus com Falcao	**16**	**14.0**	8.0	6.0	0.0	0.0	0.0	0.0	0.0
Colisão dos tiros com Falcao	**16**	**15.0**	0.0	2.0	9.0	4.0	0.0	0.0	0.0
Comportamento da barreira	**24**	**22.0**	0.0	0.0	0.0	7.0	9.0	6.0	0.0
Sprint 6	**56**	**51.0**	**8.0**	**8.0**	**9.0**	**11.0**	**9.0**	**6.0**	**0.0**
Leitura de níveis	**16**	**15.0**	7.0	8.0	0.0	0.0	0.0	0.0	0.0
Transição entre níveis	**24**	**23.5**	0.0	0.0	8.0	7.5	8.0	0.0	0.0
Sprint 7	**40**	**38.5**	**7.0**	**8.0**	**8.0**	**7.5**	**8.0**	**0.0**	**0.0**
Ecrã progresso	**16**	**15.0**	7.0	8.0	0.0	0.0	0.0	0.0	0.0
Ecrã principal	**16**	**14.5**	0.0	0.0	8.0	6.5	0.0	0.0	0.0
Ecrã de opções	**8**	**9.0**	0.0	0.0	0.0	0.0	9.0	0.0	0.0
Sprint 8	**40**	**38.5**	**7.0**	**8.0**	**8.0**	**6.5**	**9.0**	**0.0**	**0.0**
Sala de prémios	**24**	**24.5**	8.0	7.5	9.0	0.0	0.0	0.0	0.0
Navegação	**16**	**15.0**	0.0	0.0	0.0	7.0	8.0	0.0	0.0
Sprint 9	**40**	**39.5**	**8.0**	**7.5**	**9.0**	**7.0**	**8.0**	**0.0**	**0.0**

Tabela 3 - Product Backlog para o jogo Falcao VS Aliens

Movimento da personagem principal

Depois de definida a lista de tarefas deu-se início à implementação do jogo, iniciando-se este desenvolvimento pela implementação da dinâmica de jogo, pois desta forma foi possível começar a verificar onde poderiam existir falhas e detectá-las ainda numa fase inicial do desenvolvimento. A implementação dos menus foi deixada para segundo plano uma vez que estes ecrãs ao não serem tão dinâmicos nem tão complexos como o ecrã de jogo permitiam uma maior rapidez de implementação.

A primeira funcionalidade a ser implementada foi o controlo da personagem principal, sendo para isso experimentadas as três formas de controlo discutidas no processo de concepção do jogo. A solução que estava relacionada com a utilização de botões foi a primeira a ser testada, tendo sido efectuados testes às duas versões referidas na fase de *design*, e que são a utilização da botões visíveis e utilização de zonas de controlo no ecrã, ficando cada metade do ecrã responsável pelo movimento para o respectivo lado. Com os testes realizados a esta solução foi possível verificar que os problemas previstos na fase de *design* estavam certos, uma vez que a utilização de botões visíveis ocuparia bastante espaço do ecrã, por forma a que o jogador tivesse uma zona suficientemente grande para poder controlar a personagem, o que levava a que área de jogo ficasse bastante menor que o pretendido para este jogo. A solução de utilizar duas zonas no ecrã também não se revelou boa para este caso. No primeiro teste a esta solução colocou-se a personagem na zona onde se pretendia que esta estivesse durante o jogo e foi possível verificar que a necessidade de o jogador ter dois dedos prontos para interagir levava a que ao estarem colocados sobre o ecrã, os dedos iriam esconder a personagem principal durante grande parte do jogo, ficando o jogador com grandes dificuldades no controlo da personagem. O segundo teste levou à subida da personagem no ecrã, contudo foi também possível verificar que esta solução, tal como a que utilizaria botões visíveis iria roubar bastante área de jogo, não sendo este facto aceitável para este jogo.

O segundo método de controlo implementado e testado foi o controlo da personagem através de movimentos do dispositivos e perceptíveis através da utilização do acelerómetro. Esta solução veio confirmar os problemas identificados na fase de concepção do jogo, ou seja, é bastante difícil calibrar o movimento do jogador através dos dados recebidos do acelerómetro, uma vez que para o controlo da personagem ser perfeito os movimentos exercidos pelo jogador tinham que ser suaves, o que levantava desde logo o problema do jogador não ser capaz de controlar a personagem de forma eficaz fazendo com que perdesse vidas sem cometer erros que pudessem provocar esses resultados. Outro ponto que levou à exclusão desta solução como método de controlo da personagem foi o facto de a estar a movimentar o dispositivo, o jogador não iria ter a percepção visual que deveria ter para ter uma

jogabilidade aceitável, podendo este facto levar a que mais uma vez o jogador fosse penalizado por erros que não eram da sua responsabilidade.

O terceiro método, e o que acabou por ser o escolhido para controlo da personagem, foi a utilização do arrastamento do dedo sobre o ecrã. desde início foi possível verificar algumas das vantagens que este método oferece, entre as quais um controlo preciso da personagem, uma vez que a personagem se move consoante o movimento do dedo do jogador, sendo este um facto bastante relevante no que diz respeito à jogabilidade pretendida para este jogo. O facto de a personagem seguir o movimento do dedo do jogador fez com que não existisse uma probabilidade tão grande do jogador perder o contacto visual com a personagem, uma vez que saberia sempre da sua localização ao olhar para a posição onde se encontra o seu dedo, no entanto e para evitar que o dedo ficasse demasiado sobreposto à personagem principal, foi dada alguma margem no fundo do ecrã para o jogador a poder controlar.

Comportamento dos inimigos

Após ter sido implementado o herói passou-se à implementação dos inimigos, por forma a ser possível, no final desta fase, a realização de teste de colisões tanto entre a bola e a personagem principal como entre a bola e os inimigos. A realização destes teste permitia ter uma noção da jogabilidade e de alguns detalhes que pudessem ser ajustados. Tal como evidenciado no planeamento, a implementação dos inimigos foi realizada em três fase, sendo cada fase destinada a um tipo específico de inimigo. No caso dos inimigos estáticos, a implementação não consumiu bastante tempo, uma vez que se tratam apenas de objectos estáticos que não sofrem movimentos durante o decorrer do jogo. A implementação dos inimigos dinâmicos, ou seja, os que apresentam movimento no decurso do jogo foi mais trabalhosa devido a ser necessário encontrar uma solução que permitisse o movimento deste tipo de objectos entre vários pontos. Para dar resposta a esta situação, sempre que um destes objectos era criado era passada uma lista de posições que marcavam os pontos onde o inimigo tinha que passar durante o seu movimento, devendo esta lista ter no mínimo duas posições para que fosse possível efectuar qualquer movimento. Para realizar este movimento, cada inimigo dinâmico aplica a si próprio uma velocidade no sentido do vector director da primeira posição para a segunda posição, sendo em cada instante verificado se a posição alvo já tinha sido atingida. Caso isso tivesse acontecido, era actualizada a posição alvo para a posição seguinte na lista e aplicada uma velocidade nesse sentido. Sempre que o objecto chegava a uma das posições localizadas nos limites das listas iniciava o movimento inverso, isto é, se a lista de posições fosse composta por quatro elementos, o objecto inicialmente iria efectuar o movimento pela seguinte ordem 1 - 2 - 3 - 4, uma vez no final da lista iniciava o

movimento inverso, indicado pela ordem 4 - 3 - 2 - 1, invertendo o movimento sempre que chegava a um dos limites da lista. No caso de se pretender que o objecto efectuasse um movimento circular pelas posições, isto é, ao chegar à ultima posição da lista voltar para a primeira posição, bastava a passagem de um atributo para que tal fosse tido em conta.

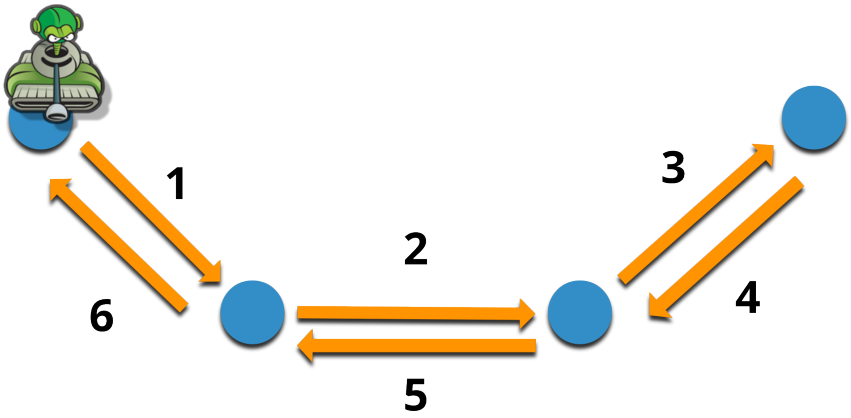


Figura 27 - Exemplo de movimento padrão de um inimigo dinâmico

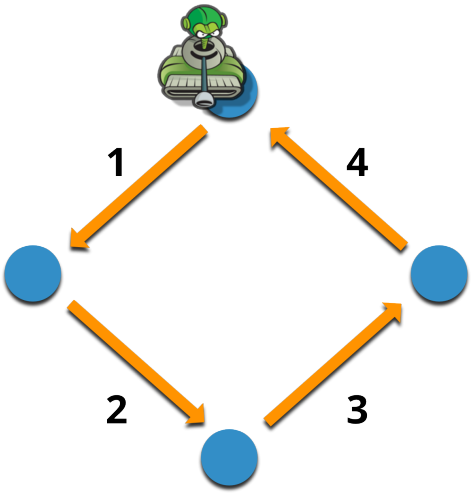


Figura 28 - Exemplo de movimento circular de um inimigo dinâmico

O terceiro tipo de inimigos que se podem encontrar no jogo são inimigos com poder de fogo, ou seja, têm a capacidade de disparar projécteis na direcção da personagem principal. Este tipo de inimigos foram criados recorrendo a uma extensão da classe representativa dos inimigos dinâmicos que inclui um temporizador para o disparo. Ao ser criada uma instância deste tipo de inimigos, o temporizador é inicializado e de cada vez que o tempo definido chega ao fim é disparado um projéctil. A classe utilizada para servir de temporizador estava já incluída no motor de jogo e para que o seu manuseamento fosse o correcto, bastou implementar um método que serve para o temporizador alertar o seu responsável que o tempo chegou ao

fim. O quarto e último tipo de inimigo implementado foi o inimigo que serve como guarda-redes, sendo este um inimigo estático com algumas características diferentes como é o caso do corpo físico que é utilizado para gerar as colisões. De seguida é apresentado o diagrama de classes que representa a hierarquia entre as classes que representam os vários tipos de inimigos.

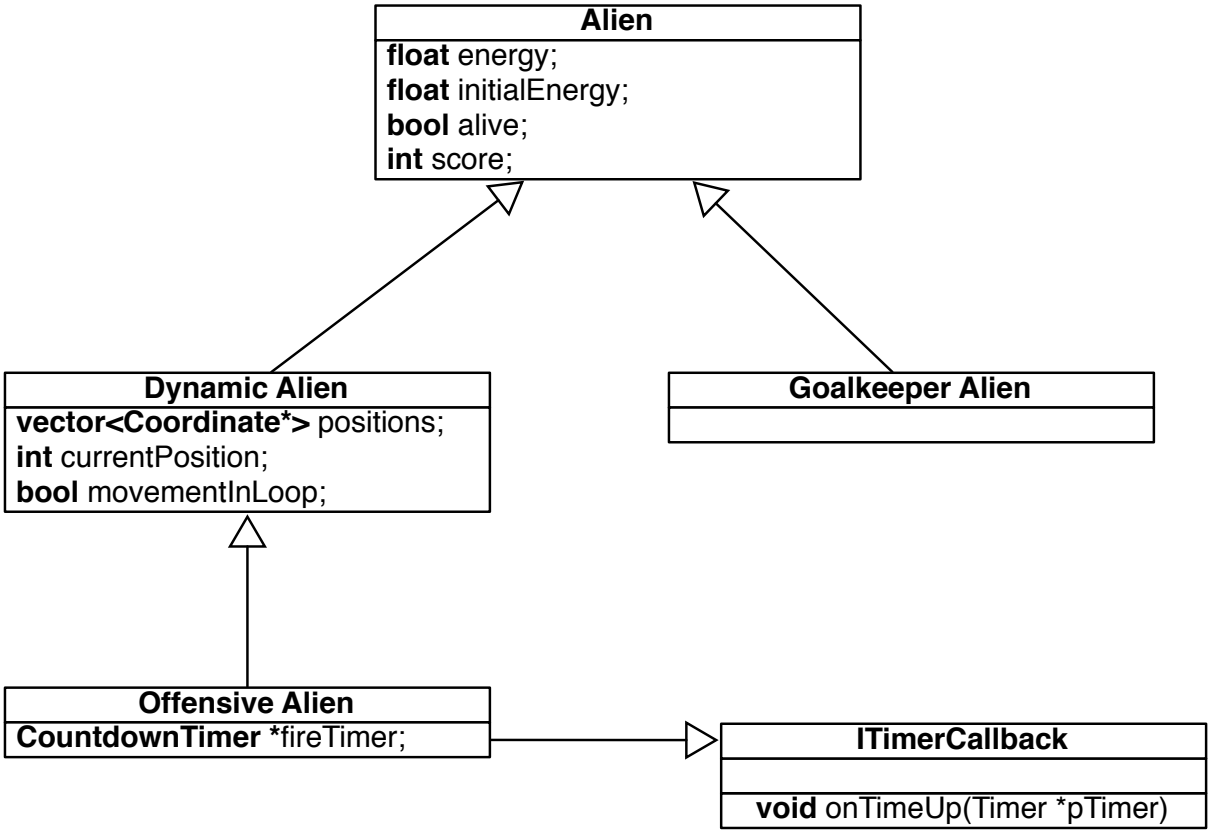


Figura 29 - Estrutura de classes utilizada para a representação dos inimigos

Tratamento de colisões

O processo do tratamento de colisões por si só não ofereceu grande complexidade de implementação, uma vez que foi utilizado um motor de físicas (Box 2D) para este projecto, sendo este o responsável por todo o processo de simulação das físicas e da detecção de colisões. Sempre que era detectada uma colisão era despertado um evento que era transmitido até à cena responsável pelo ecrã de jogo, sendo aí efectuado o tratamento. O processo mais moroso este processo foi a realização de operações sempre que determinadas colisões ocorriam.

Sempre que era despertada uma colisão entre a bola e um dos inimigos era causado dano a esse mesmo inimigo, podendo a quantidade de dano levar a que o inimigo fosse destruído. Uma vez que o tratamento do dano causado e do facto de este ser ou não suficiente para destruir o inimigo é efectuado na classes própria de cada inimigo. Desta forma foi necessário implementar um mecanismo de *callback* que enviasse informação à cena de jogo por forma a que esta tivesse conhecimento da destruição do inimigo e consequentemente efectuar os procedimentos identificados para essa situação, como é o caso do aumento da pontuação, ou a remoção do inimigo da lista de inimigos activos. Esta solução era também aplicada no caso da colisão detectada ser entre uma bola e o guarda-redes.

Se a colisão detectada fosse entre a bola e o interior da baliza, foi utilizado um corpo rígido para simular a física da baliza e um sensor para detectar quando era marcado um golo, eram iniciado o procedimento de finalização do nível, ou seja, eram descartadas todas as colisões detectadas a partir desse momento, assim como apresentada uma mensagem indicativa de que o utilizador tinha terminado o nível. A bola gerava ainda colisões quando em contacto com outros objectos como é o caso das estrelas que foram utilizadas para a abertura da barreira que servia de protecção da baliza. Sempre que a bola colidia com uma dessas estrelas, esta era eliminada e era dada indicação à barreira que tinha sido apanhada mais uma estrela, sendo esta responsável por avisar a cena de jogo se não precisava de mais estrelas para ser aberta, ou seja, se a barreira tinha sido desactivada e a protecção da baliza deveria agora ficar a cargo do guarda-redes. Outro tipo de objectos que geravam colisões com a bola eram garrafas de bebidas energéticas que estavam espalhadas pelo cenário de jogo e que no caso de serem apanhadas ofereciam um certo período de protecção à personagem principal que ficava imune ao efeito dos projecteis disparados pelos inimigos. Era também detectada a colisão da bola com um sensor posicionado imediatamente abaixo da personagem principal e que servia para verificar se a bola tinha sido perdida. Sempre que era detectada uma situação deste tipo, era efectuada uma verificação do número de bolas em jogo e no caso de haver mais que uma bola em jogo, era apenas destruída a bola que colidira com o sensor. No caso

de ser a última bola em jogo, era efectuado o procedimento de substituição da bola caso o jogador ainda tivesse bolas disponíveis ou o procedimento de fim de jogo caso fosse a última bola que o jogador poderia utilizar.

Outro objecto muito utilizado na detecção de colisões era o objecto que representava a personagem principal, sendo que este podia ser atingido por dois tipos de objectos: bónus ou projécteis. No caso de colisão com os bónus, que são utilizados para dar alguns benefícios ao utilizador, era verificado o tipo de bónus que tinha gerado a colisão e efectuado o tratamento correspondente. Para este jogo foram definidos os seguintes bónus que podem ser aplicados tanto à personagem principal como à bola: bola extra, controlos invertidos, bolas gigantes, bolas minúsculas, armas, alta velocidade e bolas duplas. A colisão do Falcao com o bónus de bola extra faz com que seja adicionada uma bola ao número de bolas disponíveis desde que este número não exceda o máximo de bolas permitidas. Quando o bónus capturado é o relativo a bolas gigantes, todas as bolas presentes no cenário de jogo são aumentadas para o dobro do tamanho original, enquanto que o bónus bolas minúsculas reduz o tamanho destas para metade do tamanho original. Se o bónus apanhado for relativo à inversão de controlos, a forma como o utilizador controla a personagem passa para o inverso, ou seja, sempre que o utilizador mover o dedo para a esquerda a personagem move-se para a direita e vice-versa. O bónus alta velocidade faz com que a velocidade de todas as bolas aumente tornando o jogo mais dinâmico ao mesmo tempo que aumenta a dificuldade do utilizador em redireccionar as bolas. Por fim, o bónus bolas duplas faz com que todas as bolas presentes no ecrã sejam duplicadas não podendo no entanto o número total de bolas ultrapassar as seis por forma a que o desempenho do jogo não saia prejudicado. Outro tipo de colisão com a personagem que é possível detectar é a colisão com os projécteis disparados pelos inimigos ofensivos. Neste caso, sempre que uma colisão entre estes dois objectos é detectada, a personagem principal fica inactiva durante um determinado período de tempo, o que significa que o utilizador perde o controlo da personagem não conseguindo movê-la da posição onde esta foi atingida até que o tempo de inactividade ou até a bola colidir com uma garrafa de bebida energética, que activa imediatamente a personagem devolvendo o controlo da mesma ao utilizador.

Leitura de níveis

Uma vez que este jogo é baseado num conceito que marca o progresso em níveis, foi necessário implementar uma forma de carregar os dados de cada nível. Para esse efeito foi definido um ficheiro que contém a informação de todos os níveis, sendo essa informação constituída pelas seguintes informações: número e nome do nível, o resultado do jogo nesse nível (que só é utilizado no primeiro nível de cada estádio), o número de garrafas de bebidas energéticas presentes nesse nível e a sua localização, o número de inimigos presentes no nível e as suas características, que incluem o tipo de inimigo que é, se o inimigo esconde alguma estrela ou não e as posições onde os inimigos se encontram, sendo que no caso de ser um inimigo estático existe apenas uma posição. A informação dos níveis é armazenada numa classe que depois é acedida sempre que se pretende carregar a informação de um dos níveis.

Transição entre níveis

Tal como foi referido na secção onde está descrito o processo de *design* deste jogo, após o jogador completar um nível é direccionada para um ecrã que mostra o seu progresso no jogo. O regresso ao ecrã de jogo, por parte do jogador, faz com que seja criado um novo nível o que leva à leitura da informação desse nível assim como à verificação da necessidade de alterar o relvado utilizado, uma vez que cada estádio tem o seu próprio relvado por forma a dar uma maior noção de progresso ao jogo. O processo de carregamento das informações do novo nível ocorre num ecrã que não foi planeado inicialmente e que tem como único propósito informar o jogador que o jogo se encontra numa fase de carregamento para que este perceba o que se passa e não pense que aquele momento de espera é devido a um mau funcionamento da aplicação.

Implementação do ecrã de progresso

Após ter sido implementada toda a mecânica de jogo e de ter sido testado o seu funcionamento, deu-se início à implementação dos ecrãs complementares, sendo o primeiro a ser desenvolvido o ecrã relativo ao progresso do jogador no jogo. Este ecrã é composto por um caminho marcado com pontos de dois tamanhos distintos. Os pontos de maior dimensão representam o primeiro nível de cada estádio enquanto que os pontos mais pequenos representam os restantes níveis. Além destes pontos indicativos dos níveis existe também um grupo de estádios que representam os estádios de que os extra-terrestres se apoderaram e que a personagem principal tem que reconquistar. Nestes estádio existem dois indicadores do seu estado,

um cadeado que ao estar visível significa que o jogador não pode ainda jogar os níveis desse estádio porque não completou o anterior e uma nave espacial que no caso de estar visível significa que o jogador ainda não conseguiu vencer o jogo disputado nesse estádio, estando este ainda na posse dos extra-terrestres. Para que seja possível armazenar a informação do estado em que se encontra o jogo, nomeadamente no que diz respeito ao estado dos estádios, foi criada uma base de dados para esse efeito sendo que de cada vez que é criado o ecrã de progresso é efectuada a leitura da informação dos estádios. Sempre que se termina um nível e é efectuada a transição do ecrã de jogo para este ecrã é verificado se a conclusão do nível leva a alguma alteração no estado dos estádios e no caso de isso acontecer essa alteração é propagada para a base de dados por forma a que da próxima vez que iniciar o jogo a informação relativa aos estádios esteja correcta. Neste ecrã existe ainda uma representação da personagem principal que se vai movendo de um ponto para outro por forma a representar o trajecto que o jogador tem efectuado ao longo do jogo. Na imagem seguinte é possível visualizar a representação gráfica de um estádio bloqueado assim como dois pontos que fazem parte do caminho.

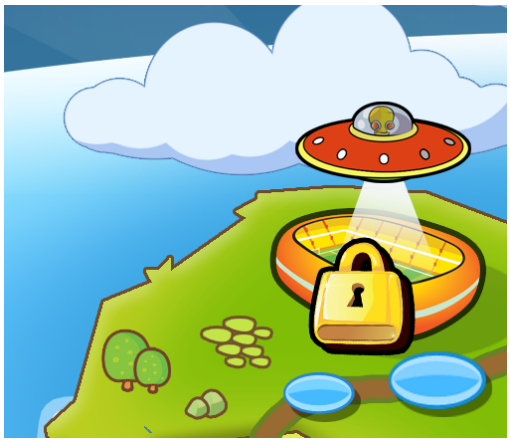


Figura 30 - Imagem representativa de um estádio no ecrã de progresso

Implementação do menu principal

A implementação do menu principal foi a tarefa que seguiu no processo de desenvolvimento. A ideia inicial para o apresentação deste menu passava por uma solução estática, ou seja, sempre que este menu era mostrado ao jogador, todos os seus componentes já se encontravam localizados nas suas posições, no entanto após uma conversa com o departamento gráfico da empresa optou-se por uma solução que entregasse mais dinamismo a este ecrã. Essa solução passou pela queda dos botões definidos para este ecrã (jogar, opções e sala de prémios) desde o topo do ecrã até às suas posições finais sendo que o utilizador apenas poderia interagir com eles depois de terminado o movimento.

Implementação da sala de prémios

A sala de prémios foi o ecrã que mais tempo levou a implementar a seguir ao ecrã de jogo. Esta razão prende-se com o facto de todos os objectos terem que ser posicionados em posições exactas por forma a dar um aspecto realista ao ecrã. Além do posicionamento, outro processo que levou a que a implementação deste ecrã fosse morosa foi o facto de os prémios só puderem ser apresentados quando o objectivo relativo a esse prémio estava cumprido. Desta forma, e para armazenar a informação relativa a cada prémio, foi criada uma tabela na base de dados que contém informação relativa ao identificador do prémio, ao objectivo que ele representa e ao valor que deve ser atingido para se cumprir o objectivo. Além desta tabela foi também criada uma classe que pudesse representar o prémio na cena, sendo que esta classe continha a informação para cada prémio, carregada da base de dados.

Implementação do menu de opções

O menu de opções é aquele que permite ao utilizador efectuar algumas configurações ao jogo. Além das configurações comuns a todos os jogos e que foram referidas na descrição do processo de design deste jogo como são o casos da configuração da música e do som, este jogo permite também ao jogador reiniciar o jogo, fazendo-o voltar ao seu estado original, eliminando todo o progresso que o jogador tenha feito até ao momento. Uma vez que este jogo tem como personagem principal uma figura reconhecida a nível mundial levou a que fosse também adicionada uma opção para alterar a linguagem do jogo. As alterações efectuadas ao audio permitem ao jogador ligar ou desligar tanto a música como os efeitos sonoros, com recurso a botões de estado, sendo estas definições guardadas numa zona de memória disponibilizada para cada aplicação. A retoma do estado inicial do jogo passa pela eliminação do progresso relativo aos estádios que o jogador já conquistou e/ou desbloqueou, sendo este facto registado através da alteração dos valores armazenados na base de dados. A alteração da linguagem do jogo foi um processo que levou a algumas alterações na estrutura das cenas, uma vez que foi necessário a implementação de uma função para alterar os textos apresentados ao jogador. Para efectuar a alteração da linguagem, e uma vez que existem mais que duas linguagens definidas para este jogo, foi criado um tipo de elemento de interacção que não existia no motor de jogo, e que passou pela extensão do botão de estado. De cada vez que o estado muda é efectuada uma chamada ao gestor de cenas para que esta mudança seja propagada para todas as cenas que desta forma irão alterar os textos apresentados ao jogador.

5.5. Organização do código

Para o desenvolvimento deste produto foi implementado um grande conjunto de classes que serviu maioritariamente para representar os objectos que fazem parte do ecrã do jogo, uma vez que os restantes elementos (interface gráfica, cenas, camadas, etc.) já se encontram implementados no motor de jogo. Na imagem seguinte á apresentado o diagrama de classes resultante deste projecto.

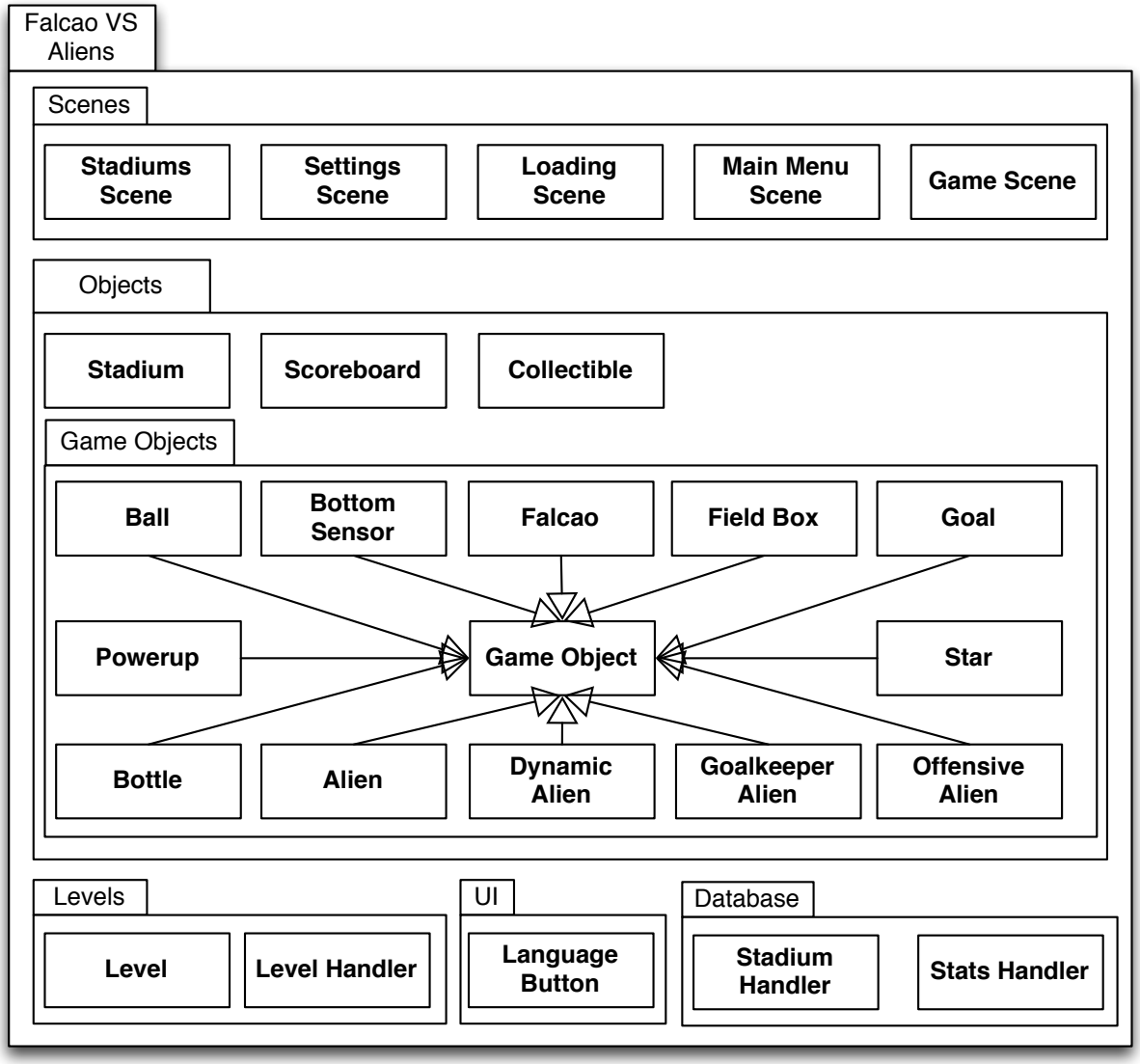


Figura 31 - Organização das classes utilizadas no jogo

Na organização do código é possível verificar que existem dois módulos principais: *scenes e objects*. O primeiro módulo é o local onde se encontram todas as classes que representam os ecrãs de jogo, sendo cada uma delas uma subclasse da classe *Scene* presente no motor de jogo e construída com o objectivo de armazenar todos os objectos que compõem cada ecrã. O segundo módulo aglomera todas as classes de objectos que são utilizados no ecrã do jogo, tendo especial destaque o

sub-módulo *GameObjects* que é onde se encontram localizados os objectos que representam os objectos de jogo, sendo todos estes descendentes da classe *GameObject* criada para oferecer a possibilidade de utilização de corpos físicos através do Box2D. Neste conjunto de classes é possível visualizar as classes representativas de todos os tipos de inimigos (*Alien*, *DynamicAlien*, *GoalkeeperAlien* e *OffensiveAlien*). Além de conterem os corpos físicos necessários à simulação do jogo estas classes são também subclasses de *AnimatedSprite*, uma classe localizada no motor de jogo e que permite a apresentação de objectos animados. Além dos objectos físicos existem também objectos utilizados para representar visualmente diversos componentes do jogo. A classe *Stadium* é a responsável por representar cada estádio do jogo no ecrã de progresso, sendo que tem também a responsabilidade de mostrar o seu estado, ou seja, apresentar o cadeado caso esteja bloqueado assim como a nave extraterrestre caso o jogador ainda não tenha conquistado o estádio. A classe *Scoreboard* é utilizada no ecrã de jogo para informar o utilizador da sua pontuação actual, da pontuação máxima que já atingiu bem como do resultado do jogo. Por fim, a classe *Collectible* é utilizada para representar os prémios que o utilizador já conquistou tendo a capacidade de se representar consoante o estado do objectivo, completo ou incompleto.

Além destes dois módulos existem ainda mais três módulos que foram utilizados na implementação deste jogo: *Levels*, *UI* e *Database*. O primeiro módulo é responsável pelo manuseamento dos níveis do jogo, sendo a classe *Level* responsável por armazenar a informação de cada nível como é o caso da localização dos inimigos bem como o seu tipo. A classe *LevelHandler* é a responsável pela leitura e armazenamento dos diversos níveis, sendo utilizada pela *GameScene* sempre que é necessário obter a informação relativa a um nível.

No módulo *UI* está localizada apenas uma classe, uma vez que apenas foi necessário criar um elemento de interacção com o utilizador para este jogo, sendo este elemento o botão utilizado no menu de opções para o utilizador ter a possibilidade de alterar a linguagem em que o jogo é apresentado. Esta classe é uma extensão da classe *ToggleButton* (representativa de um botão de estado) disponibilizada pelo motor de jogo.

No último módulo estão localizadas as classes que armazenam em memória informação armazenada na base de dados, como é o caso do estado em que se encontram os diversos estádios (classe *StadiumHandler*) assim como a classe utilizada para manusear as estatísticas do jogo, entre as quais se destacam os objectivos que o jogador tem para cumprir.

6. Avaliação

A avaliação realizada para este jogo, até ao momento da escrita deste relatório foi a realização de um inquérito de usabilidade que visa verificar se as funcionalidades desenhadas e implementadas estão a cumprir os objectivos para que foram elaboradas. Para esta tarefa distribuído o jogo por um conjunto de pessoas que iriam efectuar testes ao jogo para detecção de falhas. No final desta avaliação o grupo de teste respondeu a um questionário que serviu para se efectuar uma análise mais concreta dos pontos que poderiam ser melhorados. Este questionário oferece cinco opções de resposta para cada questão sendo os valores representativos das seguintes categorias: 1 - discordo completamente, 2 - discordo, 3 - não concordo nem discordo, 4 - concordo e 5 - concordo completamente. De seguida são apresentadas as respostas a perguntas que se encontravam nesse questionário, estando apresentadas apenas algumas questões que se julgam de maior importância para analisar as falhas do produto.

Questão	1	2	3	4	5
A disposição do ecrã é eficiente e visualmente apelativa?	0	0	0	0	5
A navegação é lógica, consistente e minimalista?	0	0	0	1	4
Os controlos são consistente e seguem convenções?	0	0	0	1	4
Os controlos são convenientes e flexíveis?	0	0	2	2	1
As sessões de jogo podem ser iniciadas rapidamente?	0	0	0	0	5
As interrupções são tratadas convenientemente?	0	0	0	0	5
A primeira experiência de jogo é encorajadora?	0	0	0	4	1
O jogo tem momentos de estagnação?	0	1	1	1	2

Tabela 4 - Avaliação do jogo pelos utilizadores de teste

A avaliação do jogo pelos utilizadores de teste foi bastante importante porque permitiu encontrar alguns problemas que não foram detectados durante a implementação do jogo assim como permitiu também verificar que alguns dos objectivos propostos para este jogo estão a ser cumpridos.

Um dos objectivos proposto para este jogo e que se encontra atingido é o facto de a aparência gráfica do jogo ser bastante apelativa,o que é confirmado pelas respostas dos avaliadores de teste à questão relativa a este facto. Outro objectivo que

se definiu para este jogo está relacionado como a navegação é efectuada, tendo sido referido anteriormente que o principal desejo para esta funcionalidade era a simplicidade para que o utilizador não se sentisse frustrado por não conseguir chegar onde pretendia. Também a rapidez com que o jogador pode iniciar o jogo foi um factor de satisfação entre os avaliadores o que leva a concluir que o facto de não existirem muitas etapas desde que o jogador inicia a aplicação até ao momento em que se encontra a jogar é um ponto favorável a este jogo. Além deste ponto, e devido ao facto de este jogo ser destinado a dispositivos móveis o facto de a forma como as interrupções são tratadas ter satisfeito os utilizadores de teste mostra que a implementação desta funcionalidade foi bem sucedida.

No entanto existem alguns pontos que não satisfizeram completamente os avaliadores. Por um lado, as respostas dos avaliadores relativamente à conveniência e flexibilidade dos controlos mostra que o controlo da personagem ainda não está perfeito. Depois de analisados os comentários deixados a esta pergunta, é possível notar que a maior razão para esta insatisfação dos utilizadores se prende com o facto de não ser permitido ao utilizador escolher que controlos pretende utilizar. Outro ponto fraco que os avaliadores encontraram neste jogo foi o facto de o jogo ter alguns momentos de estagnação, o que acontece quando no decorrer do jogo a bola se encontra num movimento quase horizontal, o que deixa o jogador incapacitado de efectuar qualquer operação para resolver este problema, sendo a única solução a colisão da bola com algum dos objectos que fará com que a direcção da bola se altere. Outro ponto que levou a que os avaliadores indicassem que o jogo tem momentos de estagnação foi o facto de quando o jogador apenas necessita de colocar a bola dentro da baliza existir alguma dificuldade para essa tarefa, o que pode causar alguma frustração no jogador.

Pela análise destes resultados, é possível verificar que este jogo tem pontos fortes mas também pontos fracos. A avaliação mostrou que ainda há alguns pontos do jogo que podem ser melhorados, como é o caso do controlo do jogador, onde uma boa ideia seria oferecer ao utilizador mais opções de controlo da personagem, ficando ele responsável por escolher o que mais lhe interessa. Outro ponto que pode ser melhorado na revisão do protótipo é uma forma de evitar que a bola entre em movimento quase horizontal. No caso de não ser possível evitar este caso, um a outra solução pode passar pela definição de uma velocidade mínima da bola segundo o eixo vertical. Relativamente ao facto de o jogador sentir alguma dificuldade em colocar a bola na baliza quando apenas isso o impede de concluir o nível é possível que a colocação de ajudas para direccionar a bola em direcção à baliza sejam uma boa solução para resolver este problema, sendo que estas ajudas apenas seriam colocadas quando já não existirem obstáculos que o jogador tem que eliminar.

7. Trabalho futuro

Antes de se poder lançar esta aplicação para o mercado existem ainda algumas situações para melhorar. Entre estas situações encontram-se a afinação dos tutoriais que irão fornecer ao jogador algumas dicas da forma como deve jogar, assim como encontrar uma solução para o facto de a bola poder entrar em movimento horizontal e impossibilitar o jogador de ter qualquer tipo de acção no jogo.

Também terá que ser tida em conta a avaliação efectuada pelos avaliadores a este jogo, sendo que esta avaliação sugeriu já algumas alterações que podem vir a revelar-se benéficas para o sucesso do produto. Entre as mais importantes encontra-se a possibilidade de oferecer ao jogador a escolha de várias formas para controlar a personagem principal assim como pela resolução das situações que levam a que a bola possa entrar num movimento quase horizontal.

Após o lançamento do produto irão existir também tarefas que terão de continuar a ser realizadas por forma a manter os jogadores interessados, entre as quais se salientam a criação de novos níveis e a criação de novos inimigos por forma a trazer maior dinamismo e variedade ao jogo. Além destes pontos poderá também ser importante pensar em novos métodos de controlar a personagem por forma a oferecer ao jogador uma maior variedade de opções. Outra funcionalidade que pode vir a sofrer melhorias é a quantidade de línguas disponíveis no jogo, podendo esta opção trazer um valor acrescido ao jogo pelo facto de os jogadores de diversas zonas do planeta sentirem que estão a ser tidos em conta e aumentarem o seu interesse pela aplicação.

8. Conclusões

Efectuando uma análise ao trabalho realizado durante este estágio, é possível concluir que este foi bastante produtivo, estando esta produtividade não só visível nas alterações efectuadas ao motor de jogo e que contribuíram para um aumento da sua qualidade mas também a nível do jogo produzido durante o segundo semestre. As alterações ao motor de jogo oferecem agora à equipa de desenvolvimento da empresa um conjunto mais alargado de soluções bem como um a melhoria qualitativa em algumas das funcionalidades que já eram disponibilizadas, levando estas alterações a tornarem o motor de jogo bem mais completo.

O projecto realizado durante o segundo semestre deste estágio, foi extremamente satisfatório, uma vez que foi um projecto bastante desafiantes em termos de desenvolvimento, levando à utilização de conhecimentos adquiridos durante os vários anos de estudo. A aplicação destes conhecimentos levou a que se verificasse como eles são importante no processo de desenvolvimento industrial de aplicações. Quanto ao produto em concreto, existe uma grande expectativa relativamente à sua aceitação por parte do mercado, uma vez que é um projecto que tem grande qualidade e que tem potencial para poder vir a ser um dos produtos mais marcantes por parte da empresa. A realização deste produto levou uma grande satisfação pois o produto final vai de encontro aos objectivos para ele traçados, superando mesmo alguns desses objectivos.

A nível pessoal este projecto de estágio foi bastante gratificante, uma vez que permitiu a integração numa equipa de desenvolvimento que tem como objectivo a produção de produtos com o objectivo de os lançar para o mercado. Esta situação levou a que tivessem que ser postas em prática muitas das técnicas de desenvolvimento que foram sido aprendidas durante todo o percurso realizado não só durante o Mestrado mas também durante a Licenciatura. O facto de ao aplicar estas técnicas de desenvolvimento se conseguir obter uma melhor gestão do processo de desenvolvimento levou à verificação da importância que estas técnicas ocupam na produção de aplicações. Dado que o produto que foi desenvolvido durante o segundo semestre tem como base uma parceria com uma entidade externa foi também possível agregar algum conhecimento do mundo negocial ligado a este tipo de parcerias sendo.

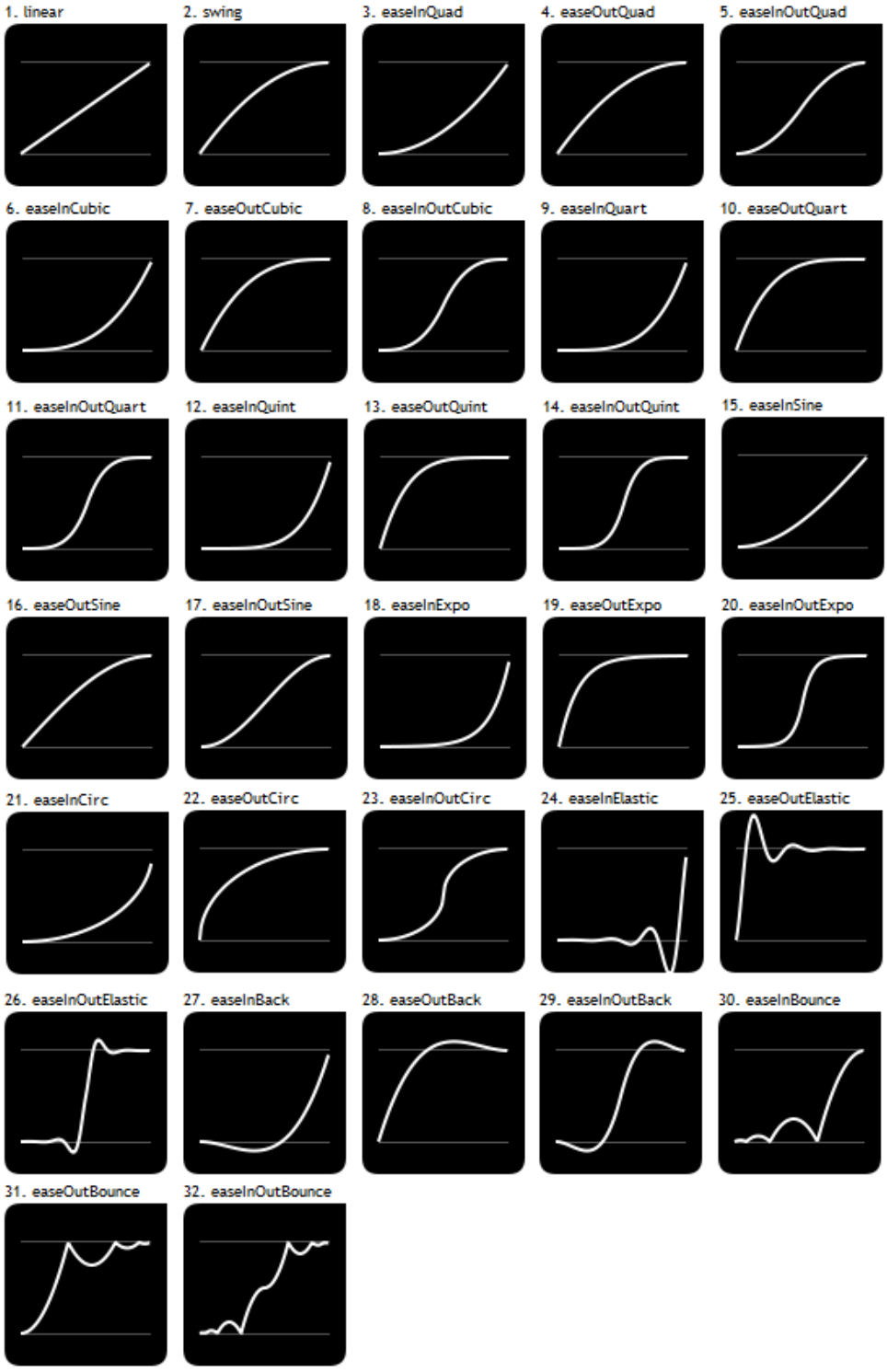
9. Referências

- Isbister, Katherine e Schaffer, Noah, *Game Usability: Advancing the Player Experience*
- Korhonen, Hannu e Koivisto, Elina M.I., *Playability Heuristics for Mobile Games*
- Desurvive, Heather e Wiberg, Charlotte, *Game Usability Heuristics (PLAY) For Evaluating and Designing Better Games: The Next Iteration*
- Craft, Chris e McElveen, Jamey, *iPhone Game Development (Developer Reference)*
- In-App Purchase Programming Guide

(https://developer.apple.com/library/ios/#documentation/NetworkingInternet/Conceptual/StoreKitGuide/Introduction/Introduction.html)

10. Anexos

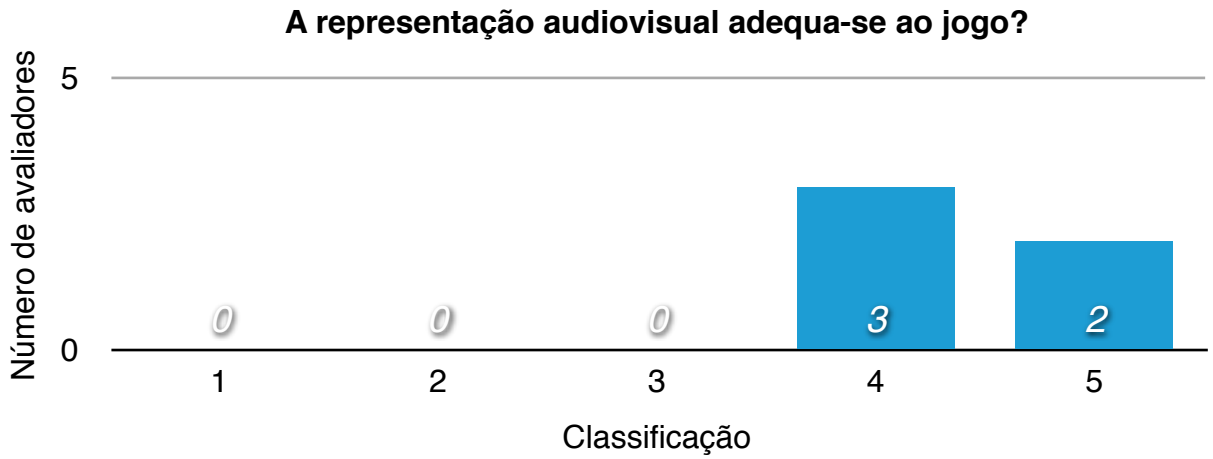
Anexo 1 - Funções de easing

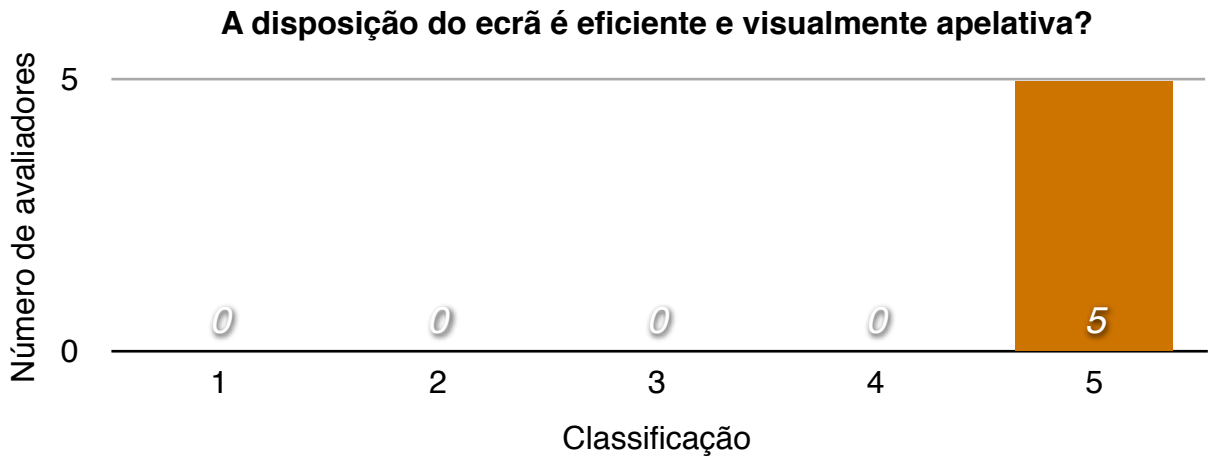


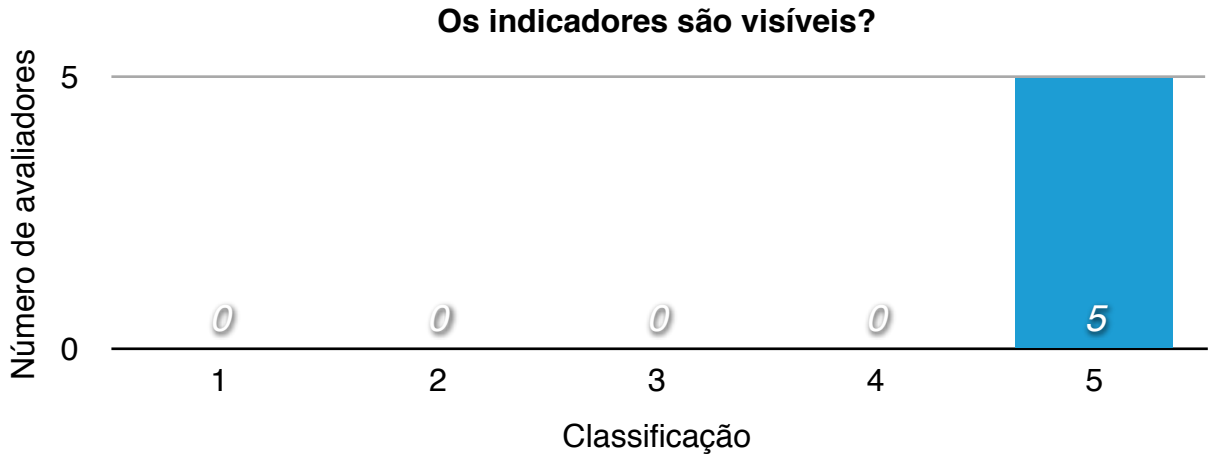
Anexo 2 - Ecrãs do jogo Falcao VS Aliens

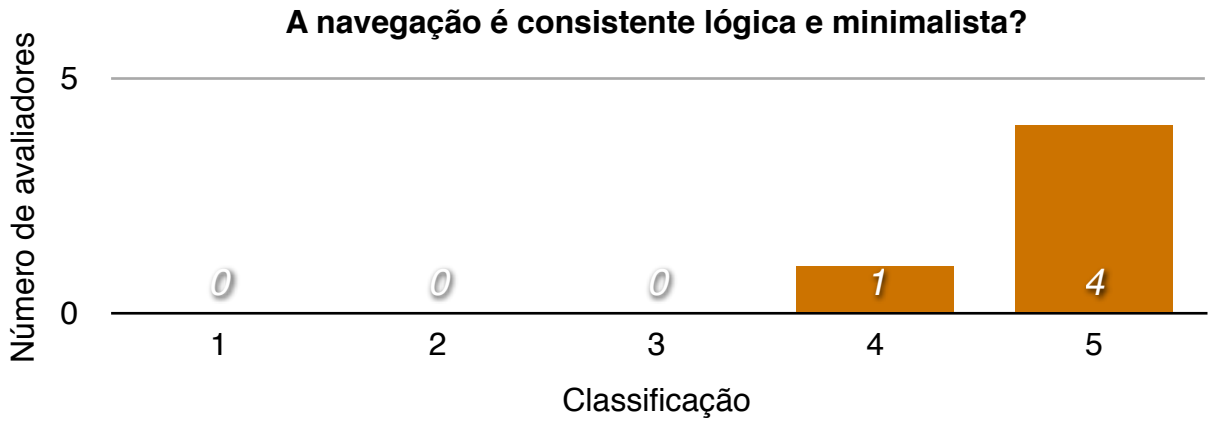

GAME WORLD
ZONE 1
TAP TO CONTINUE

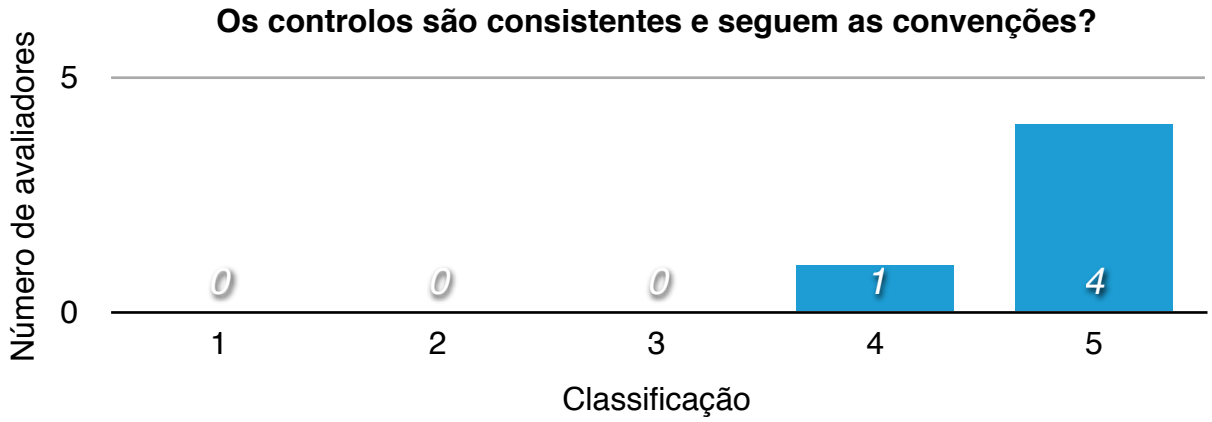
Anexo 3 - Avaliação da jogabilidade (Falcao VS Aliens)

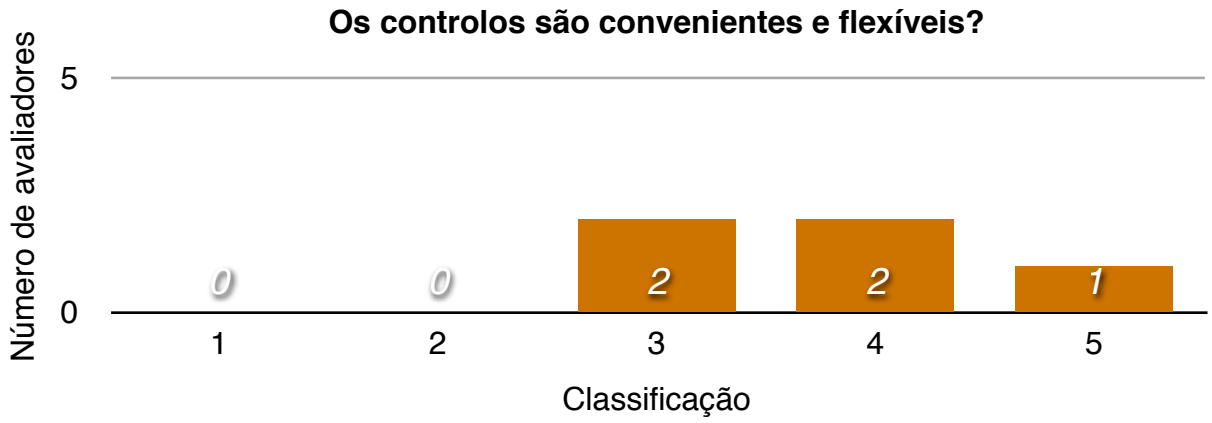


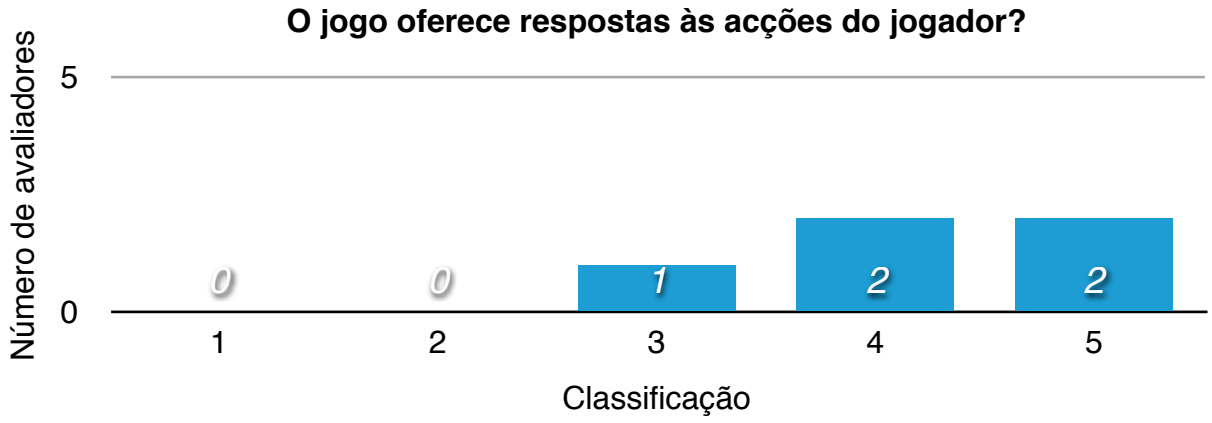


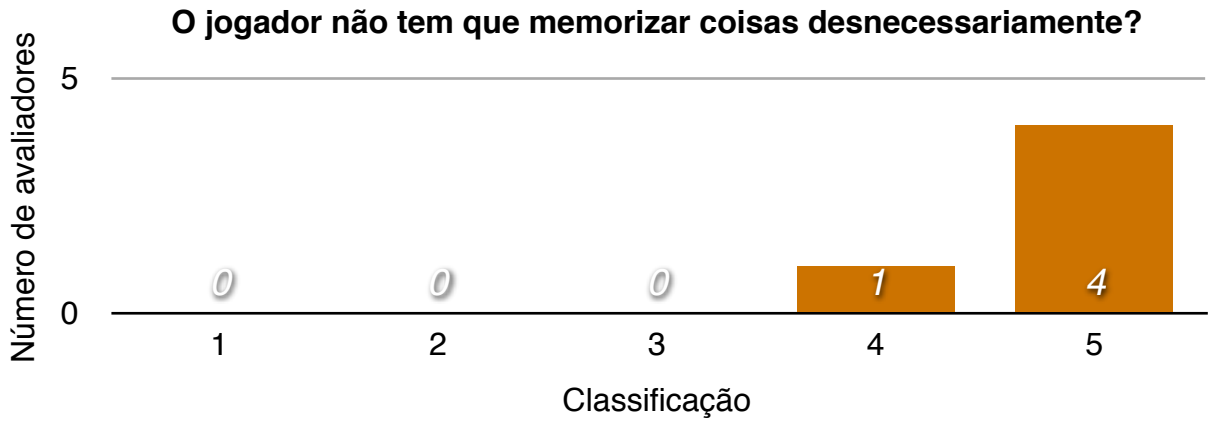


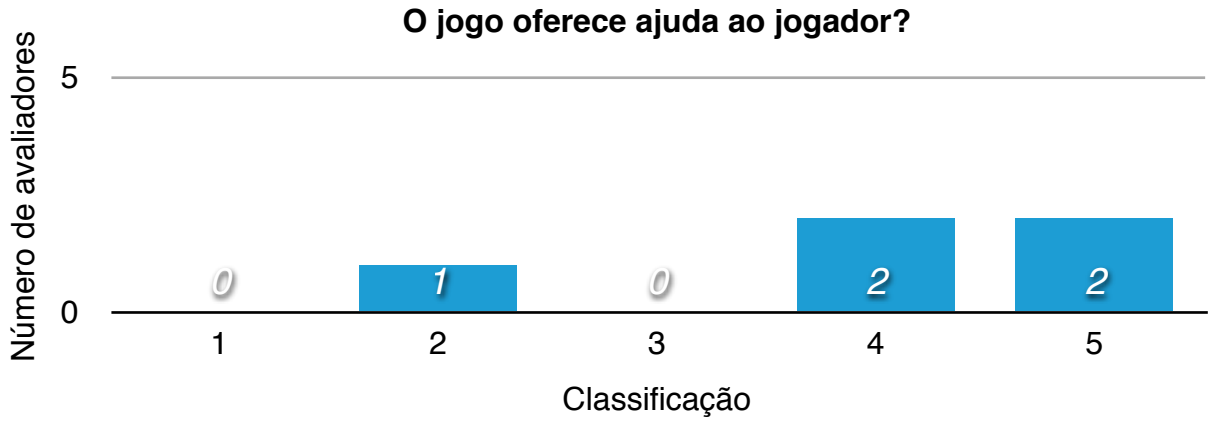

A navegação é consistente lógica e minimalista?
Número de avaliadores
5
0
0
0
0
1
4
1
2
3
4
5
Classificação

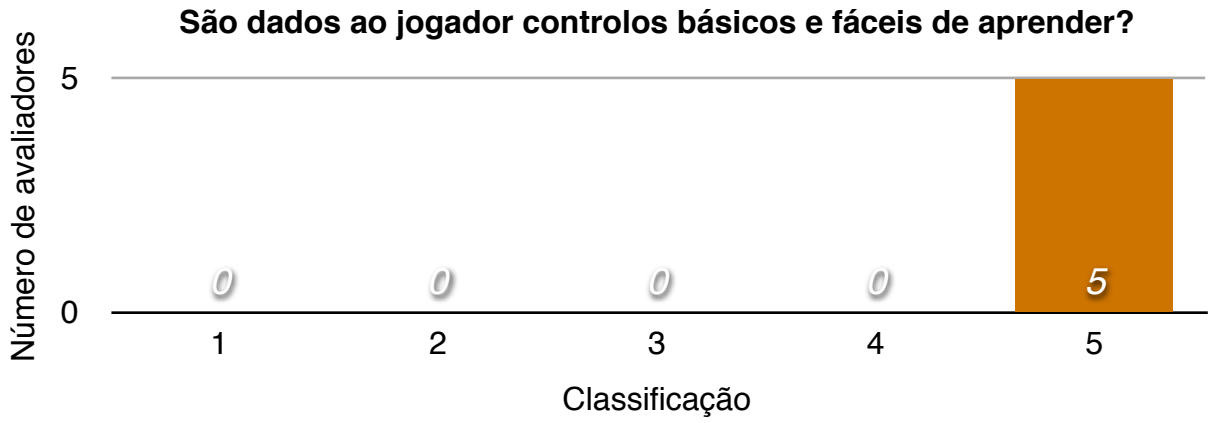

Os controlos são consistentes e seguem as convenções?
Número de avaliadores
5
0
0
0
0
1
4
1
2
3
4
5
Classificação

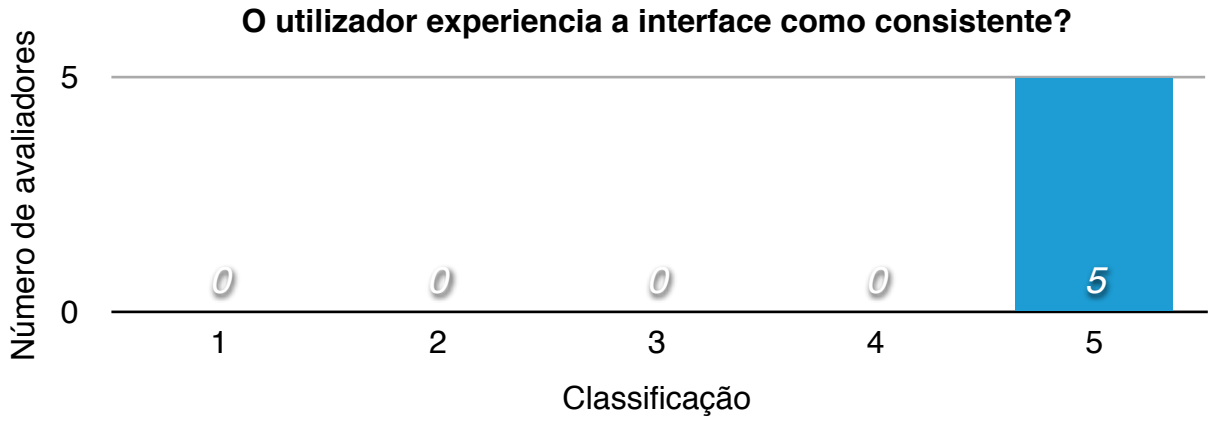

Os controlos são convenientes e flexíveis?
Número de avaliadores
5
0
0
0
2
2
1
1
2
3
4
5
Classificação

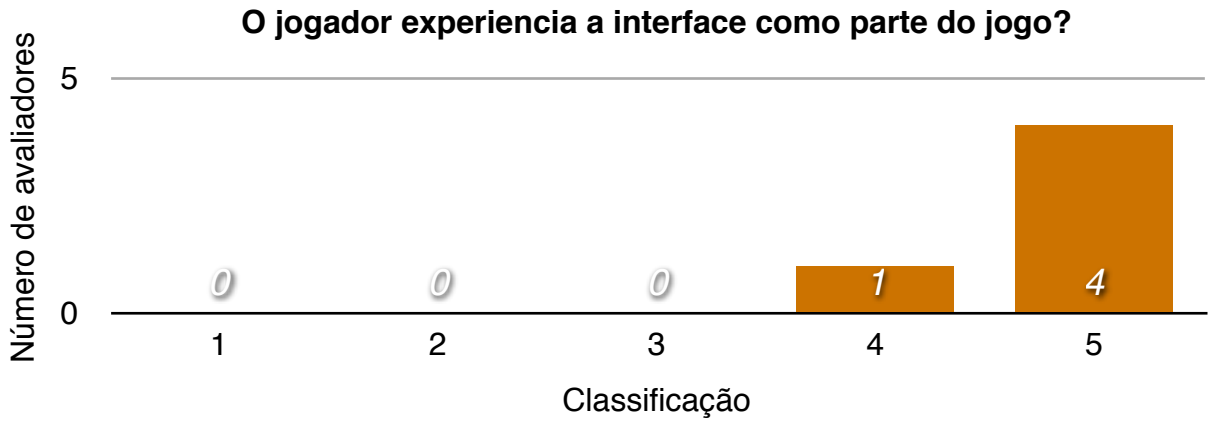

O jogo oferece respostas às acções do jogador?
Número de avaliadores
5
0
0
0
1
2
2
1
2
3
4
5
Classificação

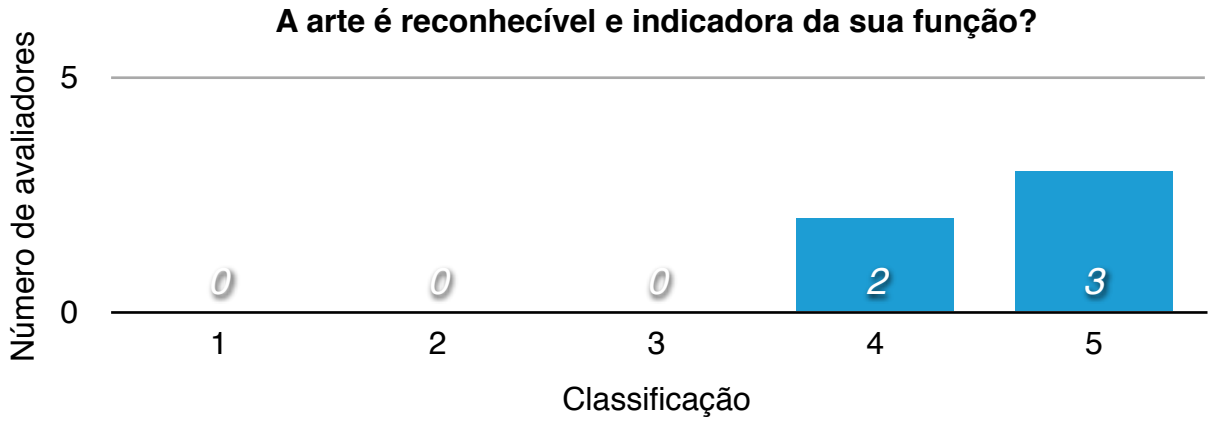

O jogador não tem que memorizar coisas desnecessariamente?
Número de avaliadores
5
0
0
0
0
1
4
1
2
3
4
5
Classificação

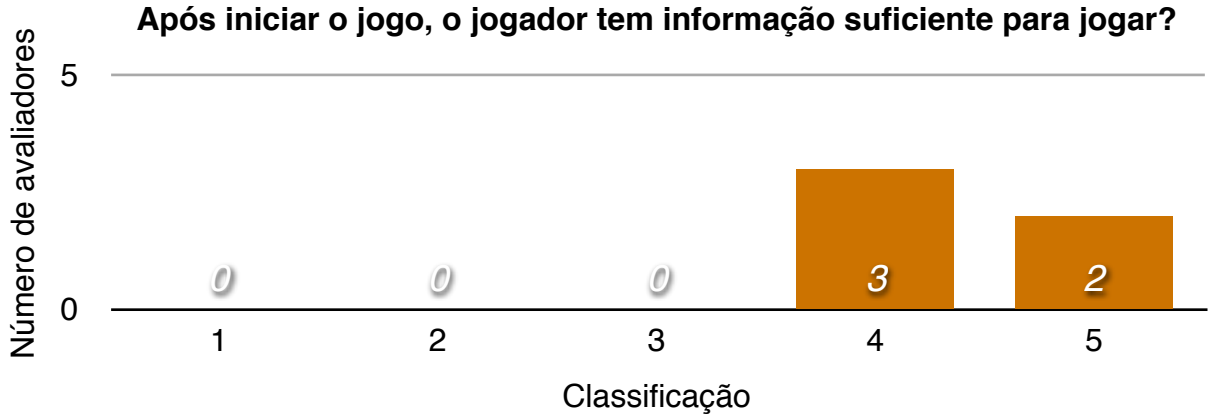

O jogo oferece ajuda ao jogador?
Número de avaliadores
5
0
0
1
0
2
2
1
2
3
4
5
Classificação

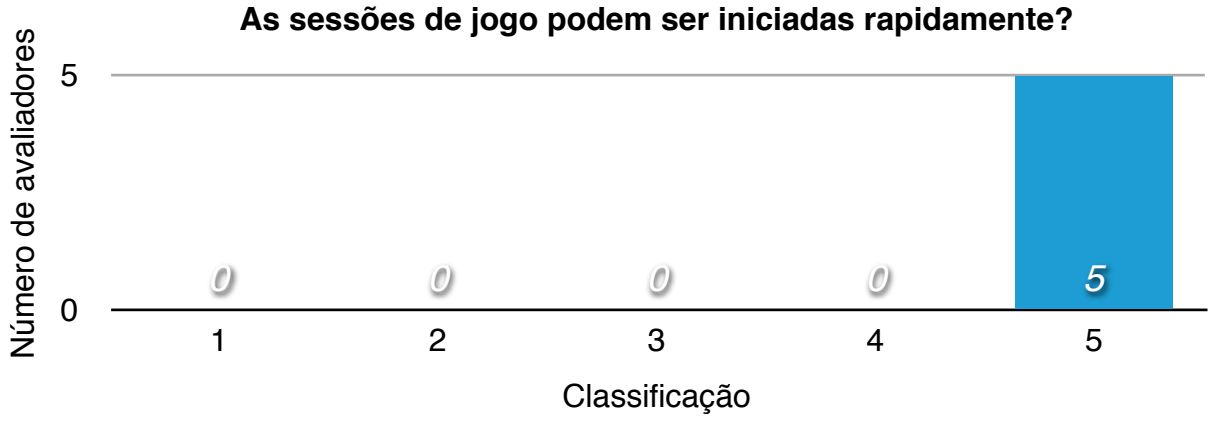

São dados ao jogador controlos básicos e fáceis de aprender?
Número de avaliadores
5
0
0
0
0
0
5
1
2
3
4
5
Classificação

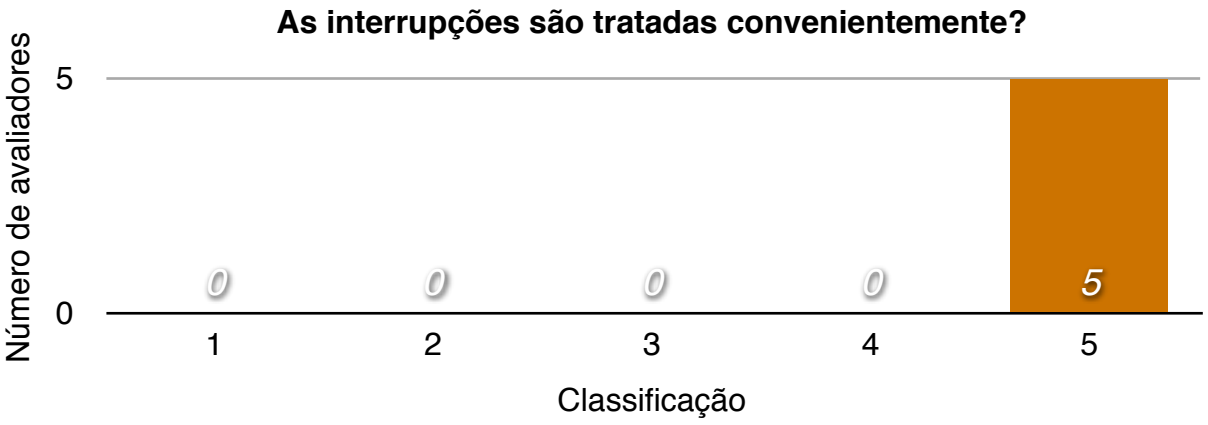

O utilizador experiencia a interface como consistente?
Número de avaliadores
5
0
0
0
0
0
5
1
2
3
4
5
Classificação

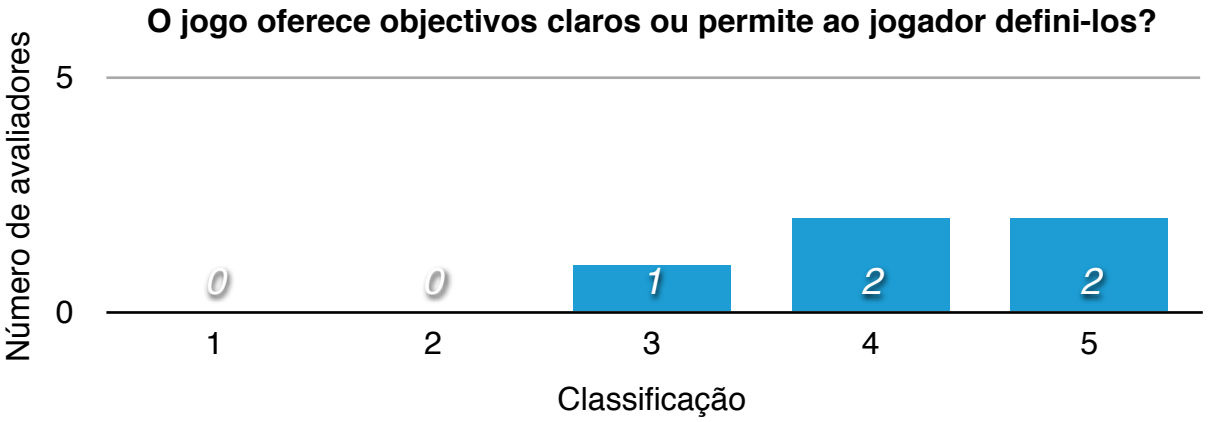

O jogador experiencia a interface como parte do jogo?
Número de avaliadores
5
0
0
0
0
1
4
1
2
3
4
5
Classificação

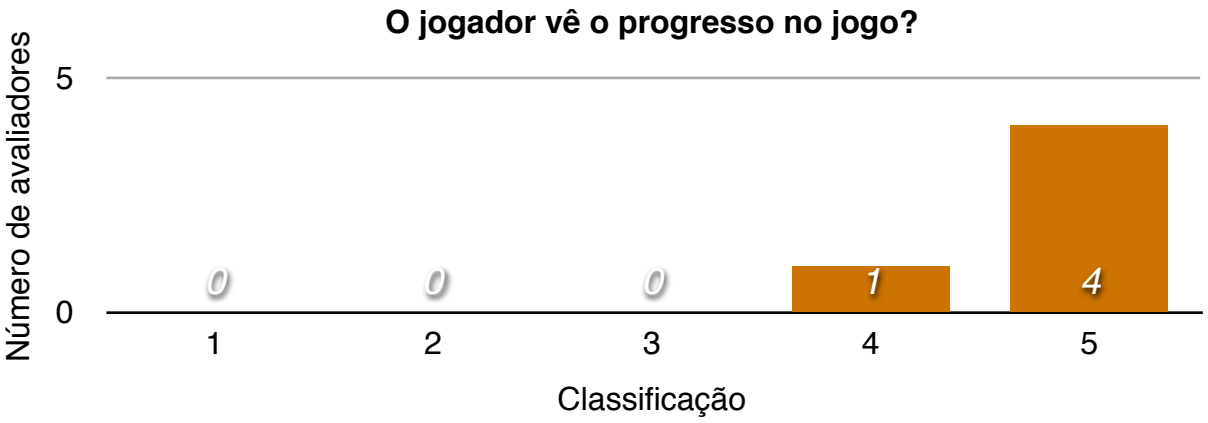

A arte é reconhecível e indicadora da sua função?
Número de avaliadores
5
0
0
0
0
2
3
1
2
3
4
5
Classificação

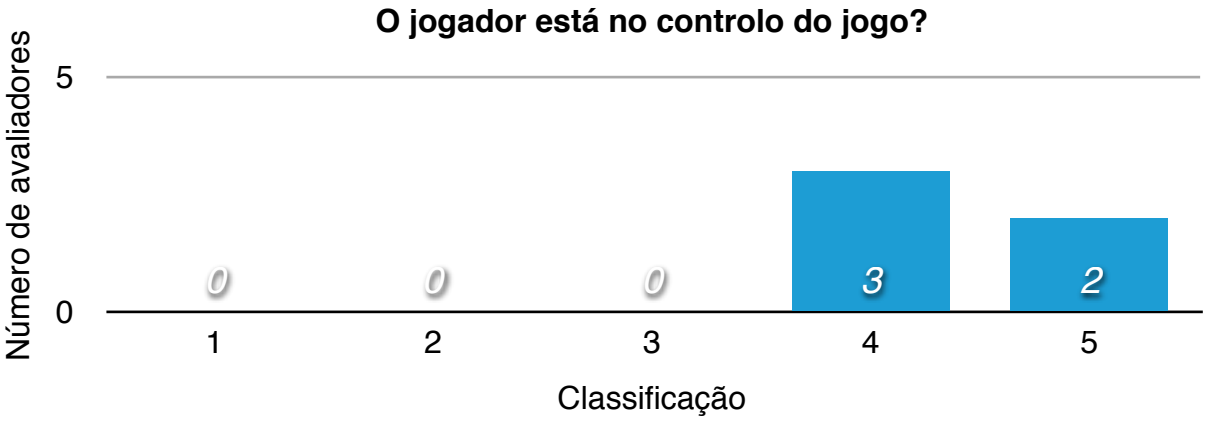

Após iniciar o jogo, o jogador tem informação suficiente para jogar?
Número de avaliadores
5
0
0
0
0
3
2
1
2
3
4
5
Classificação

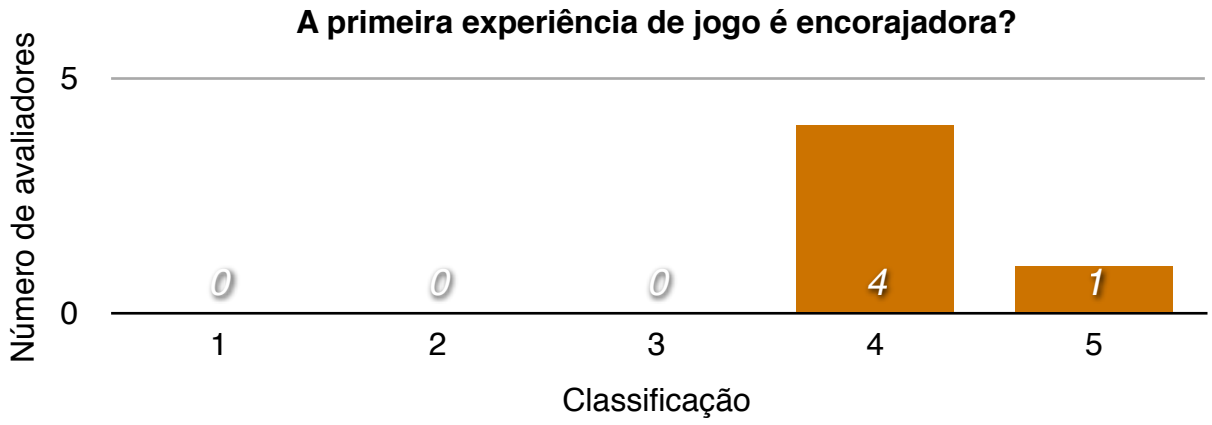

As sessões de jogo podem ser iniciadas rapidamente?
Número de avaliadores
5
0
0
0
0
0
5
1
2
3
4
5
Classificação

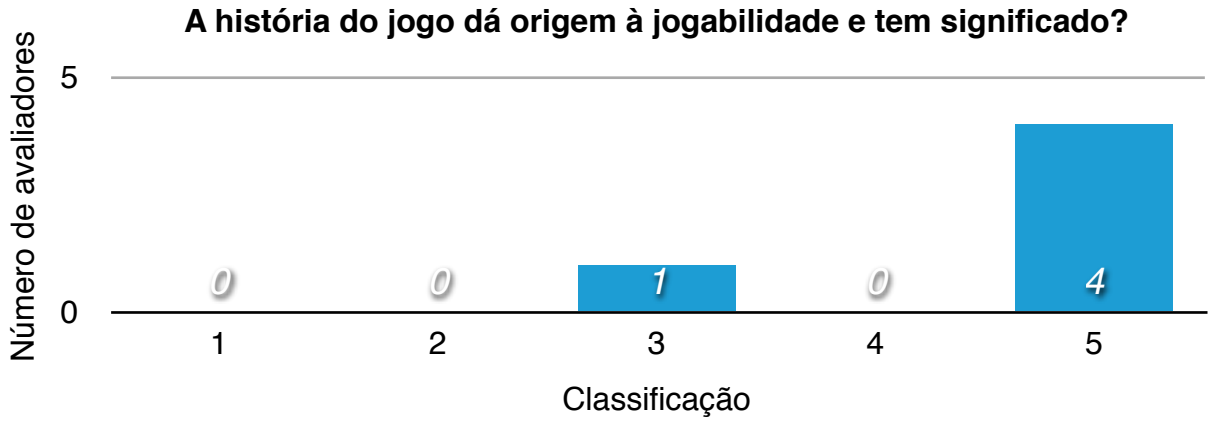

As interrupções são tratadas convenientemente?
Número de avaliadores
5
0
0
0
0
0
5
1
2
3
4
5
Classificação

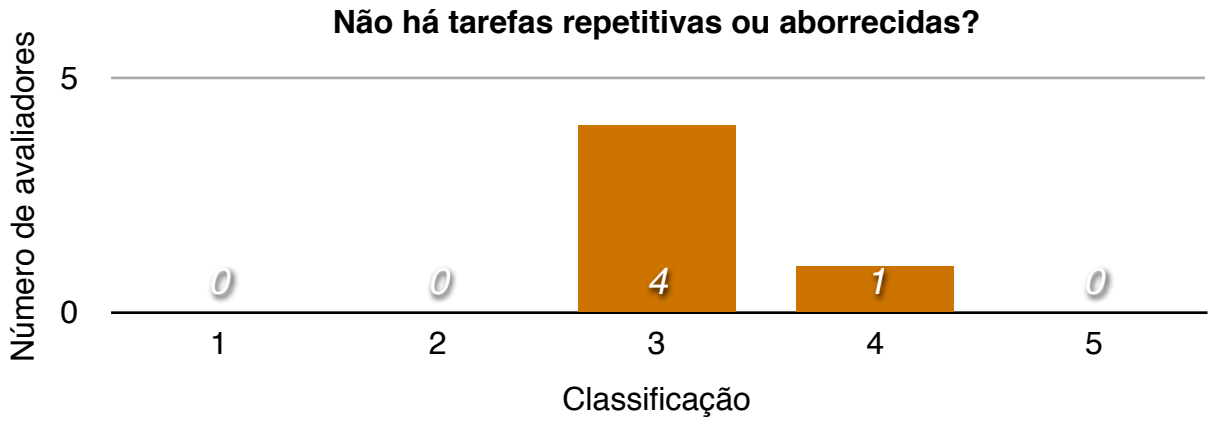

O jogo oferece objectivos claros ou permite ao jogador defini-los?
Número de avaliadores
5
0
0
0
1
2
2
1
2
3
4
5
Classificação

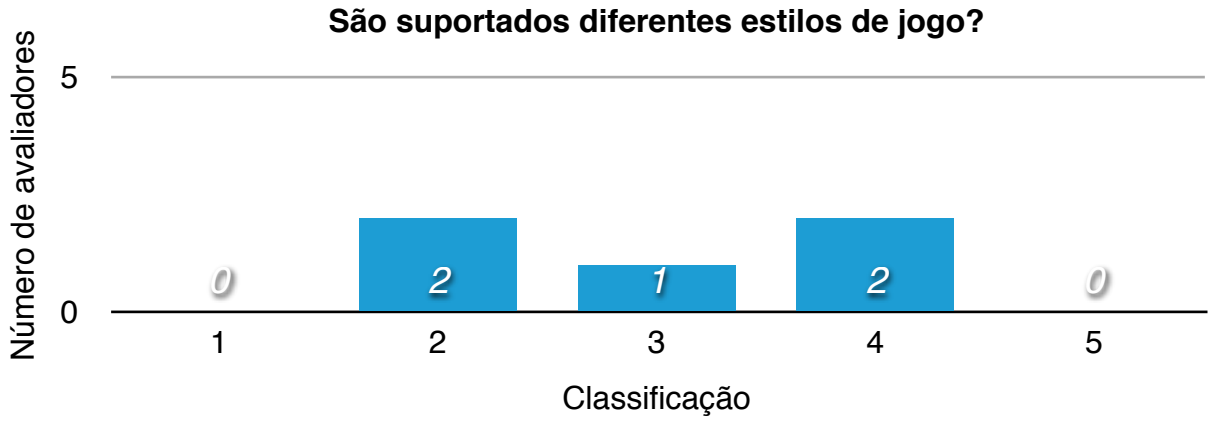

O jogador vê o progresso no jogo?
Número de avaliadores
5
0
0
0
0
1
4
1
2
3
4
5
Classificação

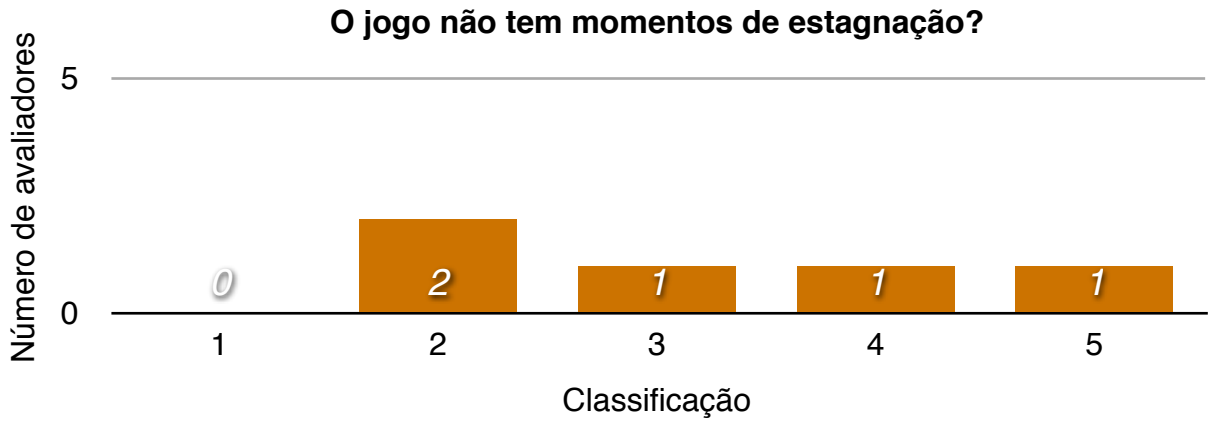

O jogador está no controlo do jogo?
Número de avaliadores
5
0
0
0
0
3
2
1
2
3
4
5
Classificação

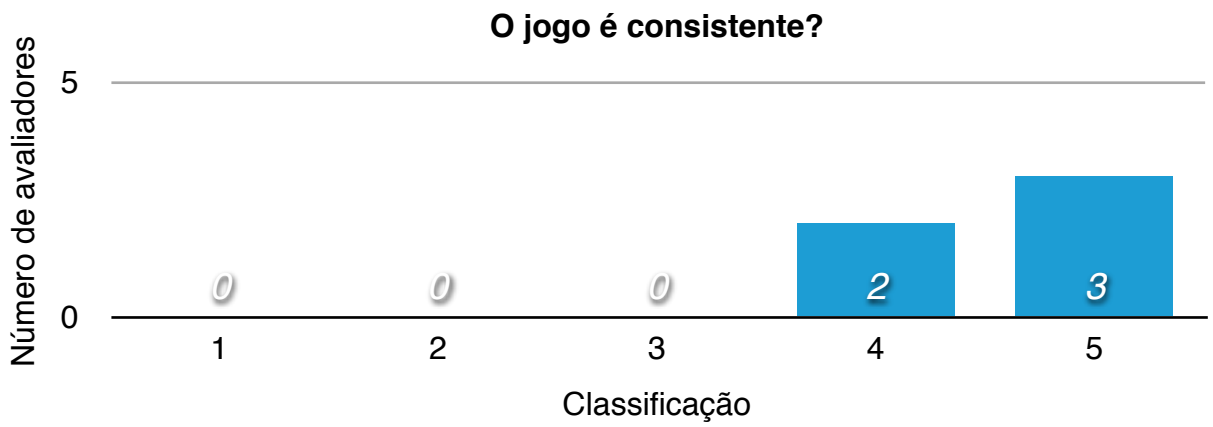

A primeira experiência de jogo é encorajadora?
Número de avaliadores
5
0
0
0
0
4
1
1
2
3
4
5
Classificação

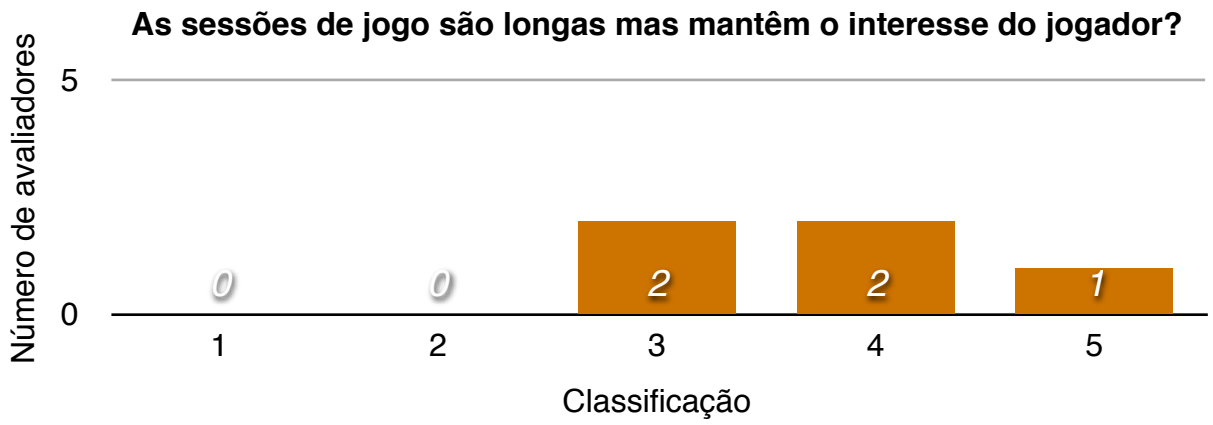

A história do jogo dá origem à jogabilidade e tem significado?
Número de avaliadores
5
0
0
0
1
0
4
1
2
3
4
5
Classificação

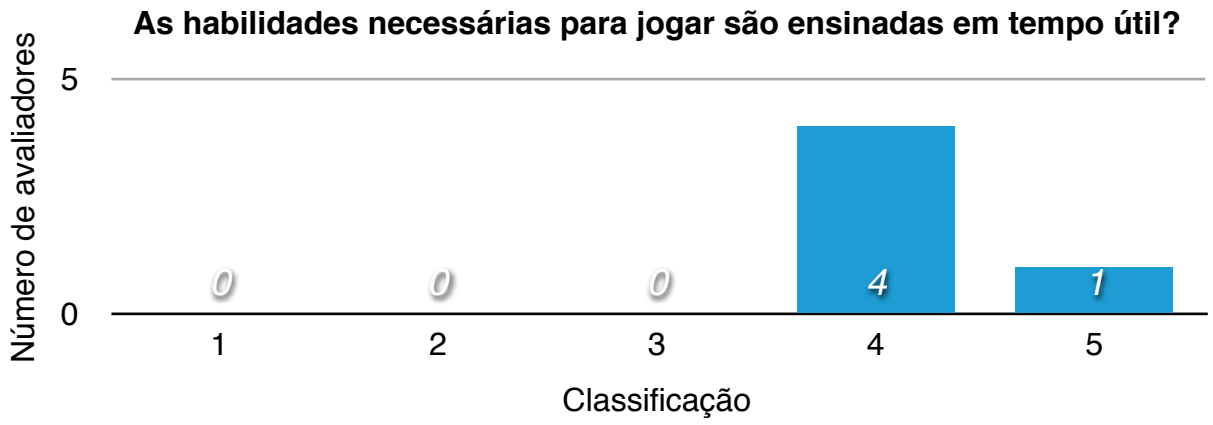

Não há tarefas repetitivas ou aborrecidas?
Número de avaliadores
5
0
0
0
4
1
0
1
2
3
4
5
Classificação


São suportados diferentes estilos de jogo?
Número de avaliadores
5
0
0
2
1
2
0
1
2
3
4
5
Classificação


O jogo não tem momentos de estagnação?
Número de avaliadores
5
0
0
2
1
1
1
1
2
3
4
5
Classificação


O jogo é consistente?
Número de avaliadores
5
0
0
0
0
2
3
1
2
3
4
5
Classificação


As sessões de jogo são longas mas mantêm o interesse do jogador?
Número de avaliadores
5
0
0
0
2
2
1
1
2
3
4
5
Classificação


As habilidades necessárias para jogar são ensinadas em tempo útil?
Número de avaliadores
5
0
0
0
0
4
1
1
2
3
4
5
Classificação