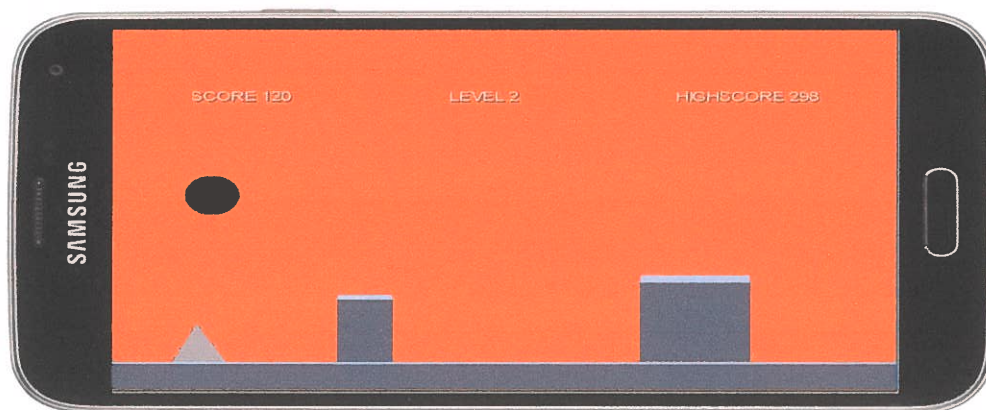


Licenciatura em Informática
Projeto Global

Desenvolvimento de um Jogo Móvel
Kamikaze Ball



ÍNDICE

1 INTRODUÇÃO.....	6
2 ESTADO DA ARTE	8
2.1 DEFINIÇÃO DE JOGOS MÓVEIS	8
2.2 HISTÓRIA DOS JOGOS MÓVEIS	8
2.2.1 Antecedentes	8
2.2.2 O GameBoy.....	9
2.2.3 Os Primeiros Jogos para Telemóveis.....	9
2.2.4 As Plataformas de Desenvolvimento para Telemóveis.....	10
2.2.5 O Papel dos Smartphones e das Lojas Móveis.....	10
2.3 A PESQUISA RELATIVA AOS JOGOS MÓVEIS NA EUROPA.....	11
2.4 O FUTURO DOS JOGOS MÓVEIS.....	12
2.4.1 TENDÊNCIAS	12
2.4.2 OS JOGOS PARA SMARTPHONES	13
2.5 SISTEMAS OPERATIVOS.....	14
2.5.1 Plataforma Android.....	14
2.5.2 Plataforma IOS	15
2.6 PLATAFORMAS DE DESENVOLVIMENTO.....	15
3 DESENVOLVIMENTO DA APLICAÇÃO	17
3.1 OBJETIVOS DO JOGO E ESCOLHA DA APLICAÇÃO.....	17
3.2 ETAPAS DO PROCESSO DE DESENVOLVIMENTO DO JOGO	17
3.3 DESCRIÇÃO DA APLICAÇÃO.....	18
3.3.1 Principais Funcionalidades:	18
3.3.2 Ambiente de Desenvolvimento	20
4 DESCRIÇÃO TÉCNICA DA APLICAÇÃO	22
4.1.1 Estrutura do Jogo.....	22
4.1.2 O Ajustamento das Escalas.....	23
4.1.3 A Construção dos Objetos.....	23
4.1.4 A Criação do Movimento no Jogo	26
4.1.5 Colisão com Objetos.....	27
4.1.6 A Criação das Telas	28
4.1.7 O Gerador dos Blocos/Inimigos e dos Níveis	30
4.1.8 O Som no Jogo	32
4.1.9 A Criação do Highscore	33
4.1.10 Game Over	34

5 CONCLUSÃO	36
BIBLIOGRAFIA	37

LISTA DE FIGURAS

Figura 1 - Tela do Loading.....	18
Figura 2 – Tela do Menu	18
Figura 3 – Tela do Jogo	19
Figura 4 – Tela do GameOver	19
Figura 5 – Android Studio.....	20
Figura 6 – LibGDX	21
Figura 7 – Estrutura.....	22
Figura 8 – Chão do Jogo.....	23
Figura 9 – Bloco Criado	24
Figura 10 - Inimigo.....	25
Figura 11 - Bola	26

Lista de abreviaturas, siglas e acrónimos

APP	<i>Mobile Application</i>
BREW	<i>Binary Runtime Environment for Wireless</i>
GPS	<i>Global Positioning System</i>
IDE	<i>Integrated Development Environment</i>
IPERG	<i>Integrated Project in Pervasive Gaming</i>
ITU	<i>International Telecommunication Union</i>
JDK	<i>Java Development Kit</i>
MGAIN	<i>Mobile Entertainment and Industry and Culture</i>
MVC	<i>Model-view-controller</i>
NES	<i>Nintendo Entertainment System</i>

PC	<i>Personal Computer</i>
RFID	<i>Radio Frequency Identification</i>
SMS	<i>Short Message Service</i>
SNES	<i>Super Nintendo Entertainment System</i>
SVN	<i>Apache Subversion</i>
WAP	<i>Wireless Application Protocol</i>

RESUMO

O presente projeto teve como objetivo desenvolver um jogo denominado *Kamikaze Ball* para dispositivos móveis, nomeadamente smartphones com o sistema operativo Android.

A linguagem utilizada na programação do jogo foi o Java e a plataforma de desenvolvimento do jogo foi o *Android Studio*, sendo auxiliado por uma biblioteca externa denominada *libGDX* que facilitou o desenvolvimento desta aplicação.

O *Kamikaze Ball* é um jogo de fácil utilização, mas ao mesmo tempo desafiador e divertido, direcionado a uma população alvo diversificada, que abrange desde crianças a adultos de idade avançada.

O jogo contempla níveis de dificuldade progressivos que requerem proficiência acrescida do jogador para poder avançar. Inclui som que acompanha as diferentes etapas do jogo podendo ser opcional. Existem mecanismos automáticos de gravação dos resultados.

O jogo foi testado junto a uma amostra da população alvo tendo-se verificado que os objetivos foram atingidos uma vez que o mesmo funcionou.

ABSTRACT

The project objective was the development of a game entitled Kamikaze Ball for mobile devices, namely smartphones with the Android operating system.

The programming language used was Java and the game development platform was Android Studio, being supported by an external library called libGDX that facilitated this application development.

Kamikaze Ball is an easy-to-use, yet challenging and entertaining game addressed towards a diverse target population ranging from children to old adults.

The game includes progressive levels of difficulty that require increased proficiency of the player to move forward. It includes optional sounds that follows the different stages of the game. There are automatic mechanisms for score recording.

The application functionality was tested by a sample of the target population and it can be said that the objectives were successfully reached as it has worked.

1 INTRODUÇÃO

Os telemóveis estão cada vez mais enraizados nas nossas vidas. Em casa ou no trabalho, a correr ou no carro, todos interagimos com esta nova realidade tecnológica. De facto, nesta nova era da revolução digital os telemóveis desempenham um papel chave, permitindo o acesso à informação e à comunicação à escala global.

Com o desenvolvimento da tecnologia digital os telemóveis que, no início do seu aparecimento se limitavam a chamadas de voz e a *SMS*, nos dias de hoje permitem ao utilizador todo o tipo de utilizações sem a necessidade de um *desktop* ou computador portátil. Este facto proporcionou oportunidades para muitos sectores de negócios e a indústria de jogos *online* não foi a exceção.

O *boom* dos jogos online em dispositivos móveis é uma realidade. Relativamente a este tema, é reconhecido no meio digital que, enquanto os dispositivos fixos, tais como computadores continuam como tecnologia dominante, para os jogadores e apostadores, o crescimento no sector das comunicações móveis é inevitável”.

Os jogos continuam a acompanhar as tecnologias inovadoras para os consumidores para que os mesmos estejam na vanguarda das atualizações tecnológicas.

Verifica-se cada vez mais que as pessoas querem produtos e serviços que estejam disponíveis 24 horas por dia, e portanto, as restrições dos países sobre esta disponibilidade não são uma opção viável a longo.

Face à importância que os jogos móveis desempenham na evolução da tecnologia digital da atualidade, decidiu-se, escolhê-lo como tema de projeto de licenciatura.

O presente projeto teve como objetivo desenvolver um jogo denominado *Kamikaze Ball* para dispositivos móveis, nomeadamente smartphones com o sistema operativo *Android*.

A linguagem utilizada na programação foi o Java e a plataforma de desenvolvimento do jogo foi o *Android Studio*, sendo auxiliado por uma biblioteca externa denominada *libGDX* que facilitou a construção desta aplicação.

Utilizou-se a linguagem *Java* por ser uma das linguagens ensinadas durante o curso e pela facilidade da instalação das bibliotecas que apoiam a programação. Possui suporte para janelas, gráficos, 2D, som, *threads*, rede, *mouse* e teclado.

Pretendeu-se desenvolver um jogo que fosse de fácil utilização, mas ao mesmo tempo desafiador e divertido, sendo dirigido a uma população alvo diversificada, que abrange desde crianças a adultos de idade avançada.

Para além da Introdução, no primeiro Capítulo, este projeto engloba mais quatro capítulos.

No segundo Capítulo descreve-se o “Estado da Arte”. Inclui uma breve revisão bibliográfica relativa à evolução dos jogos móveis, desde o seu início até aos dias de hoje assim como as metodologias mais utilizadas no desenvolvimento de jogos sustentadas, em opiniões de especialistas nestas matérias.

No terceiro Capítulo apresenta-se a estrutura de desenvolvimento da aplicação, bem como a apresentação das telas principais que o compõem, permitindo a visualização do produto final.

No quarto Capítulo, faz-se a descrição técnica da aplicação evidenciando as partes mais importantes do código do programa, justificando as etapas de desenvolvimento do mesmo e os resultados obtidos.

No último Capítulo apresenta-se a conclusão sumarizando os resultados decorrentes da realização do projeto face aos objetivos inicialmente pretendidos, destacando os pontos fortes e as dificuldades envolvidas no desenvolvimento da aplicação. Incluem-se ainda algumas oportunidades de melhoria sobre possíveis evoluções do jogo.

2 ESTADO DA ARTE

2.1 DEFINIÇÃO DE JOGOS MÓVEIS

Apesar de ser um termo bastante comum nos dias de hoje, nas sociedades orientadas para as novas tecnologias, não é necessariamente evidente o que constitui e define exatamente um "jogo móvel". A literatura sobre jogos móveis é muitas vezes tecnicamente focada e direciona-se para discutir a implementação de jogos para telemóveis e outros dispositivos móveis sem esclarecer o conceito-chave (Hamer, 2007).

A forma mais comum de entender os jogos móveis está relacionada com duas linhas distintas de desenvolvimento e publicação de jogos. A primeira relaciona-se com jogos móveis e a segunda com jogos eletrónicos portáteis e consolas de videojogos.

Os videojogos portáteis têm muitas ligações com os fabricantes de equipamentos, enquanto que os criadores de jogos móveis precisam de ter em consideração as características de diferentes modelos de telemóvel assim como os serviços de operadoras móveis que são também distintos entre si.

A forma como se faz a colocação no mercado dos jogos móveis e dos videojogos é diferente. Os videojogos chegam ao consumidor através de vendas em lojas enquanto que a distribuição de jogos para telemóveis se faz através de uma pré-instalação no aparelho ou são configurados pelo utilizador. Alguns fabricantes de dispositivos móveis recorreram a cartões de memória *add-on* como um meio de distribuição de jogos, mas não tiveram um grande sucesso.

2.2 HISTÓRIA DOS JOGOS MÓVEIS

2.2.1 Antecedentes

O início dos jogos móveis é posterior ao dos jogos eletrónicos. Os jogos eletrónicos desenvolveram-se no final da década de 1970 e começaram por ser dispositivos eletrónicos simples, como o Merlin da Parker Bros (1978).

A ideia de ter jogos para passar o tempo já tem mais tempo do que à partida se pode imaginar, pois existem dados que dizem que o imperador romano Claudius durante as suas viagens já usava jogos de tabuleiro (Joannou, 2007).

Dá que tenha sido natural que a ideia de jogos tivesse por base atividades que pudessem ser realizadas durante uma viagem como um baralho de cartas ou um jogo de tabuleiro, porque eram fáceis de usar e tinham uma portabilidade. Estes factos tiveram um papel importante na sua evolução e popularidade

Podem-se identificar duas origens para o jogo móvel. A primeira tem a ver com os primeiros videojogos *arcade* que passaram para jogos eletrónicos portáteis. A segunda está ligada ao telemóvel como um tipo de plataforma de aplicações e jogos. Em termos de adequação para a jogabilidade, um dispositivo dedicado para jogos de mão beneficia de um fator de forma e controlos otimizados para jogos. Os dispositivos móveis, por outro lado, foram desenvolvidos para facilitar a comunicação por isso o seu foco e prioridades iniciais foram nas capacidades de enviar mensagens e efetuar chamadas.

2.2.2 O GameBoy

O *Game Boy*, criado em 1989, foi a primeira consola de jogos alimentada a baterias que podiam ser recarregáveis. Para o mercado dos Estados Unidos, o dispositivo foi empacotado com o jogo Tetris, uma combinação que fez com que se tornasse muito popular entre os jogadores casuais e jovens entusiastas de videojogos.

O facto de ser um dispositivo portátil fez com que ambos os géneros jogassem e atingiram números de jogadores quase iguais. Um estudo da Nintendo referiu que em 1995, 46% dos jogadores de Game Boys eram mulheres, em contraste com 29% da consola NES e 14% das consolas SNES. As vendas totais do *Game Boy* ultrapassaram os 200 milhões de euros.

Embora a evolução de jogos para consolas portáteis tenha desfrutado de muitas condições para o desenvolvimento dos seus jogos, no caso dos dispositivos móveis foi muito diferente.

2.2.3 Os Primeiros Jogos para Telemóveis

Entre as décadas de 1970 a 1980, houve muitos fabricantes de telemóveis no mercado, cada um com o lançamento de modelos diferentes que suportavam diversos conjuntos de recursos. Os fatores-chave que contribuíram para que o desenvolvimento de um jogo fosse suportável pelos telemóveis foram algumas características do hardware e software como o tamanho da tela, teclado, memória, processador e sistema operacional.

O jogo de telemóvel mais popular de sempre foi uma versão arcade do jogo *Snake* em 1997, que foi pré-instalado em aparelhos *Nokia* e, portanto, acabou por estar disponível em mais de 400 milhões de dispositivos móveis (Wright, 2008).

2.2.4 As Plataformas de Desenvolvimento para Telemóveis

Antes de serem lançados os Smartphones, havia várias plataformas de desenvolvimento concorrentes para jogos móveis como *Macromedia Flash Lite*, *Doja* de *NTT DoCoMo*, *BREW* por *Qualcomm* e *Java ME* da *Sun*.

Através do protocolo inicial da internet móvel (WAP), estas tecnologias possibilitaram, no final da década de 1990, a venda, o *download* e a instalação de um jogo para telemóvel através de uma rede de operadoras sem fio. Além disso, as mensagens de texto (SMS) foram usadas para implementar jogos simples, como questionários, onde o preço de cada mensagem de texto foi incluído na conta do dispositivo móvel (Feijoo, 2012).

Com os recursos de transferência de dados lentos e pequenas telas dos telemóveis disponíveis, as listas de operadores eram de um modo geral bastante limitadas. Por exemplo o operador norte-americano *Verizon Wireless* listou cerca de 350 jogos e o seu concorrente *Sprint* cerca de 250 (Rabowsky, 2009).

A maioria dos utilizadores reconhecia a qualidade de um bom jogo associando a um título que era imediatamente reconhecido. Os lançamentos de jogos com base em séries populares de filmes, televisão ou livros eram, portanto, escolhas dominadoras.

Enquanto as tecnologias móveis de "*middleware*", como o *Java ME*, continuam a ser populares em telemóveis de baixo custo, como aqueles que funcionam com o sistema operacional *Symbian Series 40* da *Nokia*, os smartphones mudaram radicalmente o rosto dos jogos móveis.

Em 2003, a *Nokia* tentou lançar um sistema dedicado a jogos baseado num dispositivo móvel chamado *N-Gage*, mas a quantidade de jogos, preços e a experiência dos utilizadores da *N-Gage* fez com que fosse um total fracasso quando comparado, por exemplo com o *Game Boy* da *Nintendo*.

2.2.5 O Papel dos Smartphones e das Lojas Móveis

Foi o lançamento do *iPhone* pela *Apple* em 2007, seguido do serviço de distribuição da *App Store* em 2008, que teve o maior impacto no *software* móvel e nos ecossistemas de jogos. Em 2013, a *Apple*

informou que os seus utilizadores fizeram *download* de mais de 40 mil milhões de aplicativos da *App Store* e que a loja tinha para download mais de 800 mil aplicações móveis ("*apps*"). Outros canais de distribuição digital semelhantes incluem o *Google Play* (originalmente lançado em 2008 como "*Android Market*") e o *Windows Phone Store* (lançado em 2010 como "*Windows Phone Marketplace*"). Todas estas lojas móveis oferecem aos utilizadores acesso a milhares de aplicações, alguns gratuitos, alguns pagos.

A crescente popularidade dos ecossistemas de aplicações móveis pode ser atribuída à melhor qualidade dos jogos móveis, à melhor experiência do utilizador fornecida pelos smartphones habilitados para tela sensível ao toque e acesso mais rápido, via banda larga móvel (redes 3G e 4G). Estima-se que o número de utilizadores de smartphones em todo o mundo excedesse mil milhões em 2012, ultrapassando os números de qualquer outra plataforma de jogos, exceto jogos em computadores pessoais. Da mesma forma, o desenvolvedor de jogos finlandês *Rovio* informou que a sua popular franquia *Angry Birds* de jogos móveis atingiu o acumulado de mil milhões de *downloads* em 2012.

2.3 A PESQUISA RELATIVA AOS JOGOS MÓVEIS NA EUROPA

A investigação relativa aos jogos móveis não teve por base unicamente o mercado de estudos de jogos contemporâneos. Por outro lado, a pesquisa relacionada com os telefones móveis concentrou-se principalmente na vertente de comunicação e não esteve focada em jogos móveis. No entanto, existem várias vertentes de pesquisa que se relacionam entre estas duas áreas de investigação.

Na Europa, alguns centros de pesquisa realizaram trabalhos direcionados a jogos móveis, uma vez que a União Europeia apoia e promove a pesquisa e desenvolvimento de jogos móveis. Por exemplo, o projeto de pesquisa *MGAIN* enquadrou os jogos móveis no contexto mais amplo de conteúdos móveis e indústrias de entretenimento. Este estudo sugeriu ainda que o jogo móvel continuaria a crescer em popularidade, tal como outras aplicações e serviços móveis, relacionadas com a música para dispositivos móveis, serviços de mensagens, multimédia, jogos e serviços baseados na localização (MGAIN,2003).

O projeto *EU Kids Online* produziu pesquisas que relatam o uso por crianças de tecnologias online na Europa, indicando que jogar é uma das atividades infantis mais populares hoje em dia, mas também que os comportamentos problemáticos, como o *bullying*, se tornaram elementos comuns nas vidas das crianças (Livingstone, Haddon, & Görzig, 2012).

Outro grande projeto de pesquisa europeu *IperG*, centrou-se nas novas oportunidades artísticas, tecnológicas e de negócios relacionadas com a forma como as novas tecnologias móveis permitem a extensão de experiências de jogo em várias dimensões espaciais, sociais e temporais (Montola, Stenros e Waern, 2009).

Em termos sociológicos também houve estudos que se debruçaram sobre o jogo móvel. O trabalho de Larissa Hjorth é particularmente interessante, pois focou-se nas dimensões socioculturais dos jogos móveis na região Ásia-Pacífico. Mostrou como os videojogos e os telemóveis servem como extensões da identidade de um utilizador e como sites para porem em prática a sua criatividade (Hjorth, 2011).

2.4 O FUTURO DOS JOGOS MÓVEIS

2.4.1 TENDÊNCIAS

À medida que a popularidade e as capacidades das tecnologias móveis continuam a aumentar, é muito provável que as aplicações e serviços móveis cresçam e se tornem cada vez mais sofisticados, com capacidades de fazer ligações mais complexas ligando o jogo com outros domínios, como saúde, aprendizagem ou marketing.

Este novo conceito é conhecido por "gamificação", que significa a aplicação de elementos de jogo para fins que não são de entretenimento (Deterding, 2011).

Atualmente e no futuro, os chamados jogos que procuram estar inseridos numa determinada envolvente, incluem múltiplos dados que os tornam mais realistas, tais como dados do calendário, localização e presença de, por exemplo, objetos chamados *RFID*, atividade física que também inclui gestos, dados corporais (por exemplo, adrenalina ou nível de stress), assim como informações contextuais fornecidas por outras pessoas e redes sociais (Tester, 2006).

Todas essas informações também podem ser usadas para "gamificar" as experiências e atividades do dia-a-dia, apoiando a motivação para uma caminhada saudável, em vez de utilizar o carro.

As aplicações da gamificação na aprendizagem móvel (m-learning) também estão a ser consideradas muito atrativas e a gerar muito interesse (Kapp, 2012).

Como uma categoria, os jogos móveis desenvolveram-se em múltiplas direções por conta própria. A convergência de plataformas de jogos também é um desenvolvimento importante: em alguns ecossistemas e usando técnicas como a transmissão de jogos, agora é possível mudar de um tipo de

dispositivo para outro e ainda continuar com o mesmo jogo. Este é um desenvolvimento que contribui para eliminar barreiras entre os jogos móveis. Apesar das passagens entre os vários dispositivos é de realçar que as características-chave do jogo num dispositivo móvel permanecem distintivas e únicas na sua essência.

2.4.2 OS JOGOS PARA *SMARTPHONES*

O jogo *Angry Birds* da *Rovio Entertainment* representa bem o sucesso dos jogos móveis desenhados para o mundo atual dos smartphones.

Com base em fórmulas já testadas em jogos anteriores e, portanto, bastante confiáveis, os jogos atuais usam eficientemente a interface de tela sensível ao toque e as capacidades audiovisuais dos processadores e recursos de memória dos smartphones.

Muitos destes tipos de jogos são lançados pela primeira vez como versões gratuitas para download, tentando posteriormente que os jogadores atualizem para versões completas e pagas das aplicações móveis. Uma vez que existem muitos jogadores, o que aumenta o volume de intervenientes, os preços podem ser acessíveis, às vezes menos de um euro, o que representa um benefício significativo para os consumidores dos jogos móveis.

Uma abordagem alternativa às vendas, designa-se por modelo "*freemium*" e baseia-se em compras na aplicação de recursos "*premium*", como melhores equipamentos ou níveis adicionais de jogos que permitem outras funcionalidades que o jogo gratuito não permite. Embora comercialmente bem-sucedidas, estas técnicas foram criticadas por jogadores e desenvolvedores. Além da inovação do modelo de negócios, os jogos móveis também se focaram em algumas experiências tecnológicas, exclusivas para dispositivos móveis.

Embora existam várias décadas de história na pesquisa da computação móvel, foi no início dos anos 2000 que os primeiros jogos virados para o mercado comercial foram lançados. Muito antes disso havia vários tipos de jogos de estilo caça ao tesouro que depois foram atualizados e melhorados através de dispositivos de navegação *GPS* (Montola, Stenros, & Waern, 2009).

Milhares de aplicações móveis novas são adicionadas mensalmente às lojas de vendas online e os jogos são a categoria mais popular entre as suas centenas de milhões de clientes. Daí, poder concluir-se que a importância cultural e comercial dos jogos móveis se expandiu muito desde os seus modestos primórdios na década de 1990.

Hoje, os jogos de dispositivos móveis são cada vez mais semelhantes a jogos para o PC e consolas, especialmente se entendermos que os tablets estão incluídos na categoria dos dispositivos móveis. Os

jogos móveis estão cada vez mais integrados com as redes sociais, nomeadamente o Facebook. Há um número crescente de jogos para dispositivos móveis que oferecem um tipo de experiência de jogo social online, incluindo a comparação dos melhores resultados dentro da rede social, o envio de desafios, presentes ou convites para os amigos dentro da aplicação de jogos móveis. De destacar que um estudo específico sobre a composição dos jogadores de jogos móveis sociais indica que uma ligeira maioria dos mesmos são mulheres (Karvinen & Mäyrä, 2011).

2.5 SISTEMAS OPERATIVOS

2.5.1 Plataforma *Android*

A plataforma *Android* foi introduzida pela *Google* em 2008 como um sistema operacional para dispositivos móveis. Suporta a interface com hardware comum encontrado em dispositivos incorporados, bem como bibliotecas de programação de propósito geral para *threads* e redes.

O *Android SDK* (R. Mejer, 2012) possui um amplo suporte para programação e inclui extensos exemplos e documentação. Suporta tecnologias comumente encontradas no desenvolvimento de videojogos, por exemplo, canais de renderização 3D (através de *OpenGL*), gráficos de quadros e interação do dispositivo de entrada (teclado e tela de toque). Ao desenvolver projetos em larga escala, um *IDE* é recomendado para recursos como suporte *SVN*, repositórios, compilação automática, etc.

Os programas *Android* têm que aderir a uma estrutura de programa especial para aplicações interativas, como um jogo móvel. Por isso, qualquer programa no *Android* que solicite uma interface visual deve criar uma atividade principal. Esta atividade tem acesso a uma variedade de *widgets* e estruturas visuais. A natureza gráfica do *Android* torna o padrão de design *MVC* amplamente utilizado. Nele incluem três grandes módulos num projeto de software: O Modelo, que especifica o comportamento interno e lógico da aplicação, a Visão, que implementa o seu aspeto visual, e o Controlador, que sincroniza e comunica a Visão e o Modelo

2.5.2 Plataforma IOS

O *iOS* é um sistema operativo desenvolvido originalmente para o iPhone, e também usado em outros dispositivos móveis. O *iOS* é baseado no conceito de manipulação direta, com a utilização do toque, ou seja, a interação do sistema operativo com o usuário é imediata, pois possibilita ao usuário com toques na tela, ampliar fotos, reduzir imagens, digitar mensagens em teclado virtual entre outros.

No *iOS*, a arquitetura do sistema e as tecnologias são semelhantes às encontradas no *Mac OS X*. O *kernel* no *iPhone OS* é baseado em uma variante do mesmo *kernel* base que é encontrado no *Mac OS X*. No topo deste *kernel* estão as camadas de serviços que são utilizadas para implementar aplicações na plataforma. (HUBSCH, 2012)

2.6 PLATAFORMAS DE DESENVOLVIMENTO

O cenário de desenvolvimento para dispositivos móveis enfrenta um considerável problema de fragmentação de plataformas (Koivisto, 2006), que é um problema real para a popularidade das aplicações móveis e dos jogos móveis em particular. Além disso, uma vez que existe um número significativo de dispositivos móveis diferentes, os desenvolvedores de jogos são praticamente incapazes de lançar uma versão de jogo que sirva todas as plataformas e todos os dispositivos móveis. Na prática, significa que disponibilizar um jogo para todo o mercado de aplicações móveis é quase impossível, já que o custo de portabilidade entre plataformas pode ser maior do que o custo de desenvolvimento do jogo.

Atualmente, *Java ME*, *Brew*, *Symbian* e *Flash Lite* são as plataformas de software mais populares usadas para desenvolver jogos móveis.

Java Me é um conjunto de tecnologias e especificações para desenvolver software para dispositivos com recursos limitados sendo a plataforma de aplicações mais onnipresente para dispositivos móveis. A principal vantagem de usar o *Java ME* é que uma aplicação que pode ser escrita uma vez e pode ser usada em todos os dispositivos compatíveis com *Java ME*. No entanto, a única maneira de garantir que a aplicação desenvolvida seja executada num dispositivo específico é testá-la nesse dispositivo em particular. Além disso, as aplicações de *Java ME* são mais lentas em comparação com outras. Um primeiro passo para soluções de portabilidade é o *alcheMo*, uma solução automatizada de portabilidade *Java ME-to-BREW* desenvolvida pela *Innaworks*.

Symbian é um sistema operacional projetado para dispositivos móveis com recursos limitados. As aplicações *Symbian*, geralmente escritas em C ++, são projetadas para um dispositivo específico e, portanto, são mais confiáveis e rápidas do que as aplicações escritas para dispositivos genéricos. Naturalmente, são também mais complicados de escrever.

Brew é uma plataforma de desenvolvimento criada para executar entre a aplicação e o sistema operacional de chips do dispositivo sem fio. Portanto, *BREW* permite que os programadores desenvolvam aplicações sem se preocuparem com a interface do sistema ou detalhes da rede. No entanto, para desenvolver uma aplicação *BREW* é necessário enviar a aplicação para testes, e isso introduz um custo adicional significativo (tanto em termos de tempo como em valor).

Flash Light é uma versão leve do *Adobe Flash Player* otimizada para telemóveis e dispositivos eletrônicos de consumo. Esta abordagem é ideal para aplicações que usam massivamente recursos de áudio / gráficos e está-se a tornar cada vez mais popular, já que foi adotada por vários operadores de telemóveis, mas atualmente, a principal desvantagem é que as aplicações não são capazes de se comunicar com tecnologias como *Bluetooth* e infravermelho.

3 DESENVOLVIMENTO DA APLICAÇÃO

3.1 OBJETIVOS DO JOGO E ESCOLHA DA APLICAÇÃO

O jogo denominado *Kamikaze Ball*, foi desenvolvido com os seguintes objetivos:

- Ser um jogo que pudesse ser jogado por principiantes ou por jogadores experientes, pelo que se criaram níveis de progressão que aumentam de complexidade, requerendo maior agilidade.
- Poder ser utilizado por crianças, jovens e idosos para melhorarem e treinarem a sua capacidade de concentração, logo melhorando a sua condição e habilidade de pensar e de atuar.
- Permitir ao utilizador passar o tempo de forma agradável e em qualquer hora e lugar, dada a portabilidade dos dispositivos móveis em que o jogo poderá ser utilizado.

A escolha de desenvolvimento de um jogo para dispositivos móveis teve em consideração:

- Os jogos assentes em plataformas de telemóveis estão entre os mais utilizados, de acordo com os estudos e referências identificadas no capítulo 2 deste projeto.
- A popularidade dos jogos móveis refletida através do crescente número de jogos que aparecem no mercado.
- A possibilidade de pôr em prática os conhecimentos de programação adquiridos no curso.
- O meu gosto por jogos em plataformas móveis desde a infância até aos dias de hoje, passando pelo Game Boy até aos Smartphones.
- A curiosidade de criar um produto final que pudesse combinar os meus conhecimentos em programação com o gosto pelos jogos, sabendo que isso requeria desafios que me obrigaram a aprofundar ainda mais as ferramentas necessárias ao desenvolvimento do jogo.

3.2 ETAPAS DO PROCESSO DE DESENVOLVIMENTO DO JOGO

Tendo definido o objetivo do jogo, seguidamente houve uma segunda etapa em que seleccionei e identifiquei a plataforma Android como aquela que iria utilizar para desenvolver o jogo. Esta plataforma é reconhecida como sendo das mais utilizadas no mercado, além de que pode ser usada sem pagamento

de licenças, permitindo a sua programação por qualquer entidade e eu também possuir um Smartphone Android.

Dado que a minha formação académica foi robusta no que se refere à linguagem Java, naturalmente foi esta a escolha para desenvolvimento desta aplicação.

3.3 DESCRIÇÃO DA APLICAÇÃO

3.3.1 Principais Funcionalidades:

- Carregar o menu do jogo através de uma tela de *Loading*;

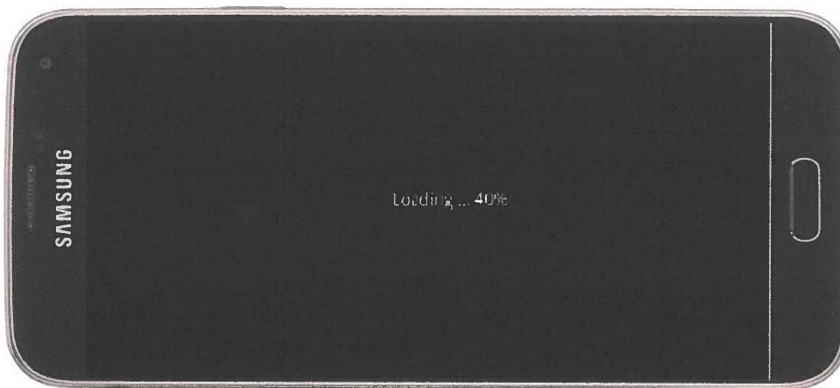


Figura 1 - Tela do Loading

- Aceder ao Jogo através de *touch* do botão Play no Menu Principal;
- Optar por ligar/desligar o som associado ao jogo através de *touch* de botão *Music - On* para *Music - Off* no Menu Principal;



Figura 2 – Tela do Menu

- Gerar o objeto principal que é controlado pelo utilizador, ou seja a bola;

- Gerar a base que serve de percurso para o jogador atuar;
- Gerar inimigos e blocos na base do jogo;
- Permitir que o jogador salte os obstáculos através da gravidade e que se vá movimentado continuamente;
- Gerar de forma aleatória os inimigos e blocos que o jogador irá encontrar, fazendo com que cada jogo seja único;
- Sinalizar o nível em que o jogador se encontra através de texto e mudança de cor da tela;
- Tornar mais difícil o jogo (maior velocidade, mais obstáculos e saltos maiores para ultrapassar os obstáculos);
- Calcular o *Score* ao longo do jogo;
- Registrar o *Score* máximo do jogador que fica automaticamente gravado no telemóvel;



Figura 3 – Tela do Jogo

- Gerar a resposta no caso do jogador perder retornando a um ecrã específico de recomeçar o jogo ou voltar ao menu principal;

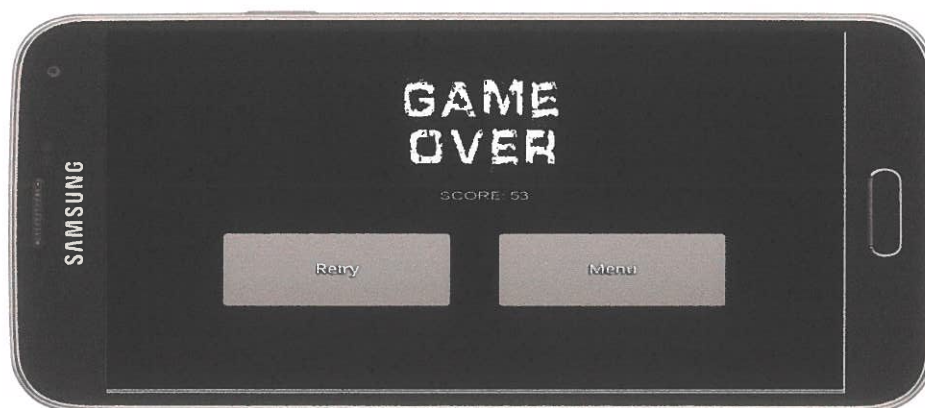


Figura 4 – Tela do *GameOver*

3.3.2 Ambiente de Desenvolvimento

O jogo móvel desenvolvido no âmbito deste projeto teve como alvo dispositivos móveis com o sistema operativo *Android* que tivessem instalado a versão 4 (*IceCreamSandwich*) ou superior.

A aplicação foi desenvolvida num computador com o Windows 10 e com o *Java SE 8* instalado, utilizando o ambiente de desenvolvimento padrão da Google, o *IDE Android Studio*, versão 2.3.3, com o *android SDK* que contém os recursos e *plug-ins* essenciais para programar neste ambiente.



Figura 5 – *Android Studio*

Para além disto, foi utilizada uma biblioteca auxiliar denominada *LibGDX* que tem funcionalidades específicas para o desenvolvimento de jogos, especialmente para jogos em 2D. Na biblioteca *LibGDX* foram utilizadas duas ferramentas, o *Scene2D* e o *Box2D*.

A *LibGDX* é um motor de jogo gratuito e de código aberto, baseado na linguagem de programação Java que permite o desenvolvimento de jogos 2D e 3 D para Android. Embora o seu foco principal seja a componente gráfica, inclui também suporte áudio, gestão de input, bibliotecas matemáticas e de simulação física 2D (*Box2D*).

O *Scene2D* está relacionado com a incorporação dos objetos no jogo (criar a base do jogo, os obstáculos, bem como o agente principal do jogo – a bola).

O *Box2d* serviu para implementar as físicas nos objetos, neste caso, fazer a bola movimentar-se na tela avançar de acordo com as instruções do jogador e até não ultrapassar um determinado obstáculo.



Figura 6 – LibGDX

4 DESCRIÇÃO TÉCNICA DA APLICAÇÃO

4.1.1 Estrutura do Jogo

Para a estrutura do jogo foram utilizados recursos (*assets*) para gerir as imagens, o áudio, as *skins* e o *Highscore*.

Foram também desenvolvidas as entidades do jogo e as regras associadas a cada uma delas, divididas em três classes.

Para o gerador dos níveis foi criada uma classe.

Para a gestão das telas foram criadas sete classes.

E para as constantes foi criada uma classe.

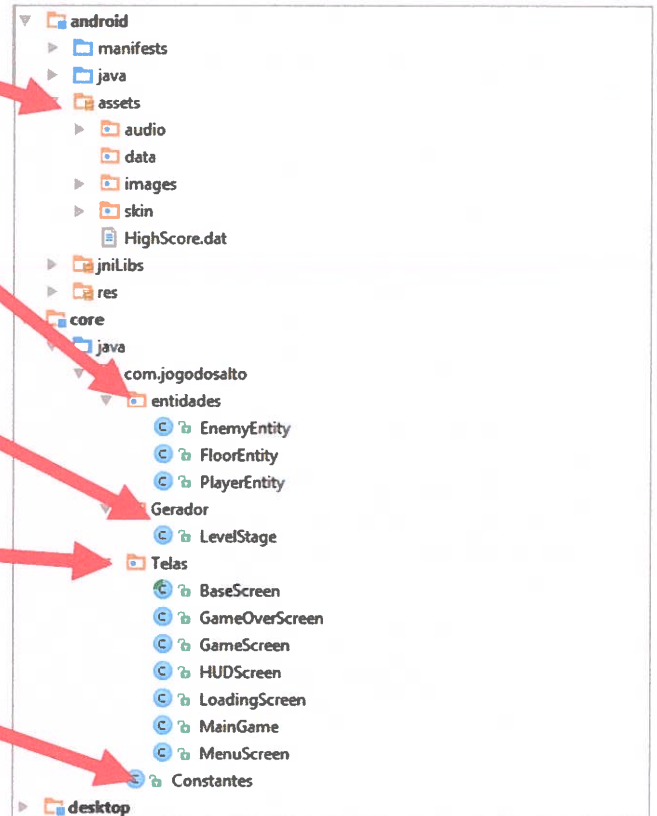


Figura 7 – Estrutura

4.1.2 O Ajustamento das Escalas

A primeira etapa do desenvolvimento do programa foi perceber as ferramentas que estava a utilizar e ver a sua compatibilidade. Assim, como referido no ponto 3.4 recorri às ferramentas, *Scene2D* e *Box2D* sendo que a primeira usa pixéis e a segunda usa metros. Por isso, foi necessário desenvolver um processo que fizesse a conversão entre eles, para que, de acordo com os requisitos de cada função, se fizesse a utilização da medida adequada.

```
public static final float PIXELS_IN_METER = 45f;
```

4.1.3 A Construção dos Objetos

Seguidamente criei quatro tipos de objetos: o chão, os blocos, os inimigos e o objeto principal. Descreve-se cada um dos objetos evidenciando as linhas principais do código desse objeto.

Código do chão – Inclui três coordenadas ($x, y, width$) e serve para os outros objetos assentarem nele e percorrerm-no como um caminho à medida que o jogo se desenrola.

```
public FloorEntity(World world, Texture floor, Texture overfloor, float x, float width, float y){
```

```
    this.world = world;  
    this.floor = floor;  
    this.overfloor = overfloor;
```

```
    //Posição correspondente ao solo  
    BodyDef def = new BodyDef();  
    def.position.set(x + width / 2, y - 0.5f);  
    //def.type = BodyDef.BodyType.DynamicBody;  
    body = world.createBody(def);
```

```
    //Em forma de caixa  
    PolygonShape box = new PolygonShape();  
    box.setAsBox(width / 2, 0.5f);  
    fixture = body.createFixture(box, 1);  
    fixture.setUserData("floor");  
    box.dispose();
```



Figura 8 – Chão do Jogo

```

BodyDef leftDef = new BodyDef();
leftDef.position.set(x , y - 0.55f);
leftBody = world.createBody(leftDef);

PolygonShape leftBox = new PolygonShape();
leftBox.setAsBox( 0.1f,0.45f);
leftFixture = leftBody.createFixture(leftBox,1);
leftFixture.setUserData("enemy");
leftBox.dispose();

setSize(width * PIXELS_IN_METER, PIXELS_IN_METER-10);
setPosition(x * PIXELS_IN_METER, (y-1) * PIXELS_IN_METER);

}

```

Código dos Blocos – Funciona da mesma maneira que o chão, mas tem quatro coordenadas($x,y,width,height$). Possui a particularidade de que quando o objeto principal embate na parte esquerda e na parte inferior de um bloco o jogador perde.

```

public FloorEntity(World world, Texture floor, Texture overfloor, float x, float
width, float y, float height){
    this.world = world;
    this.floor = floor;
    this.overfloor = overfloor;
    //Posição correspondente ao solo
    BodyDef def = new BodyDef();
    def.position.set(x + width / 2, y + height/2 );
    body =world.createBody(def);

    //Em forma de caixa
    PolygonShape box = new PolygonShape();
    box.setAsBox(width / 2, height/2);
    fixture = body.createFixture(box,1);
    fixture.setUserData("floor");
    box.dispose();

    BodyDef leftDef = new BodyDef();
    leftDef.position.set(x+0.05f, y + height/2);
    leftBody = world.createBody(leftDef);

    PolygonShape leftBox = new PolygonShape();
    leftBox.setAsBox( 0.1f, height*0.45f);
    leftFixture = leftBody.createFixture(leftBox,1);
    leftFixture.setUserData("enemy");

```



Figura 9 – Bloco Criado

```
leftBox.dispose();
```

```
BodyDef downDef = new BodyDef();
downDef.position.set(x+width/2, y + 0.05f);
downBody = world.createBody(downDef);
```

```
PolygonShape downBox = new PolygonShape();
downBox.setAsBox( width/2,0.1f);
downFixture = downBody.createFixture(downBox,1);
downFixture.setUserData("enemy");
downBox.dispose();
```

```
setSize(width * PIXELS_IN_METER, height * PIXELS_IN_METER-10);
setPosition(x * PIXELS_IN_METER, (y) * PIXELS_IN_METER);
```

```
}
```

Código do Inimigo – Quando o objeto principal embate no inimigo o jogador perde. O inimigo está definido por três vértices e por duas coordenadas.

```
public EnemyEntity(World world, Texture texture, float x, float y){
```

```
    this.world = world;
    this.texture = texture;
```

```
    BodyDef def = new BodyDef();
    def.position.set(x + 0.5f,y + 0.5f);
    body =world.createBody(def);
```

```
    PolygonShape box = new PolygonShape();
    Vector2[] vertices = new Vector2[3];
    vertices[0] = new Vector2(-0.5f,-0.5f);
    vertices[1] = new Vector2(0.5f,-0.5f);
    vertices[2] = new Vector2(0,0.5f);
    box.set(vertices);
    fixture = body.createFixture(box,1);
    fixture.setUserData("enemy");
    box.dispose();
```

```
    setPosition((x) * PIXELS_IN_METER, (y-0.25f)*PIXELS_IN_METER);
    setSize(PIXELS_IN_METER,PIXELS_IN_METER);
```

```
}
```



Figura 10 - Inimigo

Código da Bola – É o objeto que o utilizador controla durante o jogo com o propósito de percorrer o caminho máximo possível. O objeto vai ter de ser um *Dynamic Body* porque requer movimento.

```
public PlayerEntity(World world, Texture texture, Vector2 position){

    this.world = world;
    this.texture = texture;

    BodyDef def = new BodyDef();
    def.position.set(position);
    def.type = BodyDef.BodyType.DynamicBody;
    body = world.createBody(def);

    PolygonShape box = new PolygonShape();
    box.setAsBox(0.2f,0.2f);
    fixture = body.createFixture(box,3);
    fixture.setUserData("bola");
    box.dispose();

    setSize(PIXELS_IN_METER , PIXELS_IN_METER );

}
```



Figura 11 - Bola

4.1.4 A Criação do Movimento no Jogo

O jogo envolve vários movimentos da bola e para ultrapassar os obstáculos vai necessitar de efetuar um salto de determinada altura para não colidir com os blocos nem com os inimigos. Também requer que a bola desloque continuamente ao longo das várias telas que vão aparecendo. Por parte do jogador espera-se uma determinada velocidade e força ajustada a cada nível do jogo.

```
public static int IMPULSE_JUMP = 9;
public static float SPEED_PLAYER = 2f;
```

Em relação ao movimento horizontal e vertical expõe-se de seguida o respetivo código.

```
public void act(float delta) {

    //Iniciar um salto se tocarmos na tela
```

```

if(mustJump){
    mustJump=true;
    jump();
}
//se a meio do salto volta ao chao
if(jumping){
    body.applyForceToCenter(0,-Constantes.IMPULSE_JUMP * 1.15f, true);
}
//Fazer o jogador avançar se estiver vivo
if(alive){
    float speedY = body.getLinearVelocity().y;
    body.setLinearVelocity(Constantes.SPEED_PLAYER, speedY);
}

}
public void jump(){
    if(!jumping && alive) {
        jumping = true;
        Vector2 position = body.getPosition();
        body.applyLinearImpulse(0, Constantes.IMPULSE_JUMP, position.x, position.y, true);
    }
}

```

4.1.5 Colisão com Objetos

Neste programa existem três tipos de colisões:

- Entre o objeto principal e os blocos;
- Entre o objeto principal e o inimigo;
- Entre o chão e o objeto principal;

No momento da colisão, a aplicação emite um som a assinalar esse evento.

```

world.setContactListener(new ContactListener() {

    private boolean areCollided(Contact contact, Object UserA, Object UserB){

        return (
            contact.getFixtureA().getUserData().equals(UserA)
                &&
            contact.getFixtureB().getUserData().equals(UserB))
    }
}

```

```

    ||
    (contact.getFixtureA().getUserData().equals(UserB) &&
contact.getFixtureB().getUserData().equals(UserA));
}

```

@Override

```

public void beginContact(Contact contact) {

    if (areCollided(contact, "bola", "floor")) {

        player.setJumping(false);
        if (Gdx.input.isTouched()) {
            if (game.musicOff == false) {
                jumpSound.play();
            }
            player.setMustJump(false);
        }
    }
}

if(areCollided(contact, "bola", "enemy")){
    death();
}
}

```

4.1.6 A Criação das Telas

Foram criadas quatro telas: *LoadingScreen*, *MenuScreen*, *GameScreen* e *GameOver*. A título de exemplo apresenta-se a tela correspondente ao *LoadingScreen*.

```

public class LoadingScreen extends BaseScreen {

    private Stage stage;
    private Skin skin;
    private Label loading;

    public LoadingScreen(MainGame game){
        super(game);

        stage = new Stage(new FitViewport(640,360));
        skin = new Skin(Gdx.files.internal("skin/uiskin.json"));
    }
}

```

```

loading = new Label("Loading...",skin);
loading.setPosition(320 - loading.getWidth() / 2, 180 - loading.getHeight() / 2);

stage.addActor(loading);
}

@Override
public void show() {

    Gdx.input.setInputProcessor(stage);

}

@Override
public void hide() {
    Gdx.input.setInputProcessor(null);
}

@Override
public void dispose() {
    stage.dispose();
}

@Override
public void render(float delta) {
    Gdx.gl.glClearColor(0,0,0,1);
    Gdx.gl.glClear(GL20.GL_COLOR_BUFFER_BIT);

    if (game.getManager().update()){
        game.finishLoading();

    }else{
        int progress = (int) (game.getManager().getProgress() * 100);
        loading.setText("Loading ... " + progress + "%");
    }

    stage.act();
    stage.draw();
}
}

```

4.1.7 O Gerador dos Blocos/Inimigos e dos Níveis

Foi criado um gerador automático para a criação dos blocos e dos inimigos. Este gerador tem em conta o nível em que o jogador está tornando-se mais complexo de nível para nível. A velocidade, o salto, e a cor da tela modificam-se à medida que se vão passando os níveis. A dificuldade vai aumentando à medida que o desafio vai avançando e como resultado o jogador obtém maiores pontuações.

A geração dos blocos assenta em regras de espaçamento entre eles e de dificuldade progressiva. O código permite criar o emparelhamento dos blocos.

```
public void Generate(int dificuldade, float posx){

    float gapX,aux = posx + 5;
    int numeroInimigos, blocosY;
    //int lvlSize = 14*5;
    this.dificuldade = dificuldade;

    Constantes.SPEED_PLAYER = (float) dificuldade + 2;

    AddBlock(posx,0,lvlSize + 15,1);

    if(dificuldade < 4 ){
        Constantes.IMPULSE_JUMP = 7 + (int) Math.round(Math.random());
    }

    else {

        Constantes.IMPULSE_JUMP = 7 + (int) Math.round(Math.random() * 3 );

    }

    for( aux = posx + 5;aux < lvlSize*dificuldade; aux = aux + gapX) {
        if (dificuldade < 4) {

            gapX = (15 - 2 * dificuldade) - (int) Math.round(Math.random()); // Math.Random vai entre 0
            e 1 e dá um número aleatório neste intervalo e é convertido para numero inteiro.

        } else {
```

```

        gapX = 4 + (int) Math.round(Math.random() * 4);
    }
    AddBlock(aux + gapX, 0.6f, 1 + (int) Math.round(Math.random() * 1), 1f + (float)
(Math.random() * 2));
}

for( aux = posx + 5; aux < lvlSize*dificuldade; aux = aux + gapX) {
    if (dificuldade < 4) {

        gapX = (17 - 2 * dificuldade) - (int) Math.round(Math.random()); // Math.Random vai entre 0
e 1 e dá um número aleatório neste intervalo e é convertido para numero inteiro.

    } else {

        gapX = 3 + (int) Math.round(Math.random() * 6);

    }
}

```

Para os inimigos não ficarem sobrepostos nem ligados aos blocos foi necessário também fazer uma série de regras para que ficassem independentes uns dos outros.

```

boolean createBlock = true;
float x = aux + gapX;
float y = 1f;
float we = 1f;
float he = 1f;

for(FloorEntity chao : blockList) {
    float xb = chao.getX() / Constantes.PIXELS_IN_METER;
    float wb = chao.getWidth() / Constantes.PIXELS_IN_METER;
    float yb = chao.getY() / Constantes.PIXELS_IN_METER;
    float hb = chao.getHeight() / Constantes.PIXELS_IN_METER;

    if ((x > (xb-we) && x < (xb + wb) && (y > (yb -he)) && (y < (yb + hb)))) {

        createBlock = false;
    }
}

if(createBlock)

```

```
{
  AddEnemy(x, y);
}
```

4.1.8 O Som no Jogo

Para marcar momentos chave do jogo, como por exemplo o salto da bola e a colisão com os obstáculos foram adicionados sons ao jogo. No código abaixo demonstro as opções de ligar e desligar o som.

```
musicOn = new TextButton("Music - On",skin);
musicOff = new TextButton("Music - Off",skin);
```

```
musicOn.addCaptureListener (new ChangeListener() {
  @Override
  public void changed(ChangeEvent event, Actor actor) {
    game.musicOff = false;
    menuMusic.play();
    menuMusic.setPosition(10);
    musicOff.setVisible(true);

    musicOn.setVisible(false);
  }
});
```

```
musicOff.addCaptureListener (new ChangeListener() {
  @Override
  public void changed(ChangeEvent event, Actor actor) {
    game.musicOff = true;
    menuMusic.stop();

    musicOff.setVisible(false);
    musicOn.setVisible(true);
  }
});
```

4.1.9 A Criação do *Highscore*

Foi feito um contador que permite registar a pontuação obtida à medida que o jogador vai jogando, isto é, esse score vai crescendo à medida que se ultrapassam os obstáculos e interrompe-se a contagem no momento em que o jogador perde.

```
public void update(float dt,int lvl){
    timeCount += dt;

    if(timeCount >= 0.2f && isCounting()){

        score++;

        countdownLabel.setText("SCORE " + String.format("%03d", score));

        if(score > highscore){

            highscore = score;

            BufferedWriter writer = null;

            try {

                FileWriter fw = new FileWriter(hslog);
                writer = new BufferedWriter(fw);
                writer.write((highscore + ""));

                writer.close();
            }
            catch (Exception e) {
                e.printStackTrace();
            }
        }

        timeCount = 0;
    }

    lblvl.setText("LEVEL " + String.format("%01d", lvl));

    lblhighscore.setText("HIGHSCORE " + String.format("%03d", highscore));

    stage.act();
}
```

```
stage.draw();
}
```

Sempre que o jogador ultrapassa o valor que tinha no registo do seu *Highscore* , novo valor é guardado num ficheiro, que se altera sempre que o jogador atinge um valor superior.

```
protected void onCreate (Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    AndroidApplicationConfiguration config = new AndroidApplicationConfiguration();
    BufferedWriter writer = null;
    File logFile=null;
    try {

        logFile = new File(getContext().getFilesDir(), "HighScore.dat");
    }
    catch (Exception e)
    {
        e.printStackTrace();
    }
    initialize(new MainGame(logFile), config);
}
}
```

4.1.10 *Game Over*

Quando o jogador embate num obstáculo o jogo acaba e volta para o menu *GameOver* podendo recomeçar o jogo, a partir da posição inicial.

```
public void death(){
    if (player.isAlive()) {
        player.setAlive(false);

        bgMusic.stop();
        if (game.musicOff == false) {
            dieSound.play();
        }
        stage.addAction(
            Actions.sequence(
                Actions.delay(1.5f),
```

```
        Actions.run(new Runnable() {  
            @Override  
            public void run() {  
                game.setScreen(game.gameOverScreen);  
            }  
        })  
    )  
);  
hudScreen.stopCounting();  
}  
}
```

5 CONCLUSÃO

No contexto nos dias de hoje há uma procura crescente por jogos que possam ser utilizados de forma simples e em qualquer lugar, ou seja, com portabilidade que foi evoluindo de dispositivos de maior dimensão ou peso até chegar aos jogos móveis suportados pelos telemóveis da atualidade.

Por esse facto, neste projeto o foco foi o desenvolvimento de um jogo, *Kamikaze Ball*, para o sistema operativo *Android* correspondente aos dispositivos móveis que se encontram entre os mais utilizados.

O projeto permitiu aprofundar os conhecimentos de programação em particular da linguagem *Java* e da utilização de bibliotecas auxiliares do processo de programação. Permitiu também, a introdução de um gerador de níveis, a aplicação de som, a utilização de conversores para a harmonização de funções, entre outros.

O jogo foi testado com uma amostra do tipo de população alvo a quem se dirigia, ou seja, desde crianças até adultos de idade avançada. Verificam-se que os mesmos aderiram com interesse ao jogo e todos jogaram mais do que uma vez com o objetivo de fazer melhor pontuação.

Futuramente, o jogo poderá ser objeto de melhorias com uma qualidade gráfica superior e com a introdução no menu principal de instruções que indiquem o propósito e regras do jogo, apesar de ser muito evidente o objetivo do mesmo, pelo que deverá ser opcional.

Como forma de divulgação do jogo a um maior número de utilizadores, o mesmo poderá ser colocado na *Play Store*, permitindo que haja uma avaliação mais abrangente das funcionalidades a alterar ou acrescentar, para corresponder aos desejos dos jogadores que irão ser questionados através de um inquérito específico *online*.

Em suma, foi um projeto bastante enriquecedor e que exigiu bastante trabalho, tendo sido proveitoso, uma vez que pertencendo à geração dos jogos móveis tive possibilidade de conseguir concretizar uma aspiração pessoal de fazer um jogo.

BIBLIOGRAFIA

- Böhmer, M., Hecht, B., Schöning, J., Krüger, A., & Bauer, G. (2011). Falling asleep with Angry Birds, Facebook and Kindle: A large scale study on mobile application usage. Proceedings of the 13th international conference on human computer interaction with mobile devices and services, (pp. 47–56). MobileHCI'11. New York, NY: ACM.
- Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: Defining “gamification.” Proceedings of the 15th International Academic Mindtrek Conference: Envisioning Future Media Environments, Tampere. Retrieved from <http://dl.acm.org/citation.cfm?id=2181037.2181040>.
- Feijoo, C. (2012). An exploration of the mobile gaming ecosystem from developers' perspective. In P. Zackariasson & T. L. Wilson (Eds.), *The video game industry: Formation, present state, and future* (pp. 76–98). Routledge Studies in Innovation, Organization, and Technology 24. New York, NY: Routledge.
- Google Inc. The android platform. <http://developer.android.com/design/index.html>, 2012.
- Hamer, C. (2007). *Creating mobile games: Using Java ME platform to put the fun into your mobile device and cell phone*. Berkley, CA: Apress.
- Hjorth, L. (2011). *Mobile media in the Asia-Pacific: Gender and the art of being mobile*. London, UK: Routledge.
- HUBSCH, Eduardo. *Uma Abordagem Comparativa do desenvolvimento de aplicações para dispositivo Móveis*.
- Joannou, J. (2007). Have chess set – will travel. A journey in four parts. Part 1: The early years. *The Chess Collector*, 16(2), 12–18.
- Kapp, K. M. (2012). *The gamification of learning and instruction: Game-based methods and strategies for training and education*. San Francisco, CA: Pfeiffer.
- Karvinen, J., & Mäyrä, F. (2011). *Pelaajabarometri 2011: Pelaamisen muutos*. TRIM Research Reports 6. Tampere: University of Tampere. Retrieved from <http://urn.fi/urn:isbn:978-951-44-8567-1>.
- Koivisto E.M.I., 2006. *Mobile games 2010*. In *CyberGames '06: Proceedings of the 2006 international conference on Game research and development*. Murdoch University, Murdoch University, Australia, Australia. ISBN 86905-901-7, 1–2.
- Livingstone, S. M., Haddon, L., & Görzig, A. (Eds.) (2012). *Children, risk and safety on the internet: Research and policy challenges in comparative perspective*. Bristol, UK: Policy Press.
- MGAIN (2003). *Mobile entertainment in Europe: Current state of the art*. Mobile Entertainment Industry and Culture.

Montola, M., Stenros, J., & Waern, A. (2009). *Pervasive games: Theory and design*. San Francisco, CA: Morgan Kaufmann.

R. Mejer. *Professional Android 4 Application Development*. Wrox, 2012.

Rabowsky, B. (2009). *Interactive entertainment: A videogame industry guide*. Oxnard, CA: Radiosity Press.

Tester, J. (2006). *All the world's a game: The future of context-aware gaming*. Technology Horizons Program. Palo Alto, CA: Institute for the Future. Retrieved from http://www.iff.org/uploads/media/SR-997_Context_Aware_Gaming.pdf.

Wright, C. (2008). *A brief history of mobile games*. PocketGamer, December 22. Retrieved from <http://www.pocketgamer.biz/r/PG.Biz/A+Brief+History+ of+Mobile+Games/feature.asp?c=10618>.