FRAMEWORKS PARA DESENVOLVIMENTOS DE JOGOS: UMA ABORDAGEM VANTAJOSA NO DESENVOLVIMENTO DE JOGOS ELETRÔNICOS

ANDRÉ LUIZ KUCZNER PEREIRA, ANDREI RICARDO RODRIGUES, ELIANE CRISTINA AMARAL, ELINEY SABINO, MÁRIO SÉRGIO DE ALMEIDA MUNIZ, NARUMI ABE

RESUMO

Nos dias atuais, existem diversas *frameworks de jogos* disponíveis no mercado e há uma carência grande de informações abertas e práticas sobre as mesmas. Este artigo aborda o assunto de maneira geral, focando demonstrar algumas das principais vantagens em usar um *framework* para o desenvolvimento de jogos digitais, tais como o ciclo de vida e as principais funções comumente presentes de forma geral em *frameworks*. Finalmente, foi descrito um pequeno jogo para ilustrar o assunto abordado.

Palavras Chaves: *Frameworks,* Jogos, Desenvolvimento

ABSTRACT

Nowadays, there are many game frameworks available in the market and there is a great lack of open and practical information about them. This paper approaches the subject in a general way, focusing on demonstrating some of the main advantages of using a framework for the digital games development, such as the life cycle and the main functions commonly present in frameworks. Finally, we also described a small game in order to illustrate the concepts for better understanding.

KEYWORDS: *Frameworks, Games, Development*

INTRODUÇÃO

Jogos eletrônicos possuem uma série de elementos de implementação não triviais, mas que são comuns a todos os jogos. Para facilitar a codificação destes elementos, existem algumas ferramentas que visam auxiliar o trabalho de desenvolvimento, como os frameworks. Em geral, *frameworks* funcionam bem e possuem boa aceitação no mercado de desenvolvimento, demonstrando a valia desta tecnologia, pois sem o uso de ferramentas como os frameworks, a tarefa de desenvolvimento torna-se difícil e custosa, demandando mais tempo no desenvolvimento de jogos, podendo frustrar o desenvolvedor e fazendo com que o mesmo desista dessa área ou tenha redundância de tarefas.

Este artigo propõe verificar a usabilidade de um framework de jogos eletrônicos, fazendo uma abordagem geral sobre o tema. O trabalho não possui como objetivo entrar em detalhes de cada *framework*, mas sim avaliar o que podem proporcionar ao desenvolvimento dos mesmos. Ainda através deste mesmo artigo, observar a estrutura básica de um *framework* especializado em jogos junto aos elementos que compõem um jogo eletrônico.

Além disso, propõe-se verificar a viabilidade do uso de *frameworks* no desenvolvimento de qualquer tipo de jogo, focado nos critérios de custo e tempo de desenvolvimento. Também tem a pretensão de ajudar a fomentar a comunidade de desenvolvimento de jogos independentes, visando agregar mais conteúdo e conhecimento sobre o assunto impulsionando o mercado de jogos na Brasil.

JOGOS DIGITAIS

Segundo Schell (2008), um jogo possui quatro elementos básicos como ilustrado na Figura 1: (i) a mecânica que descreve as regras e procedimentos do jogo, bem como as metas a serem alcançadas pelo jogador; (ii) a história que descreve a sequência de eventos do jogo que pode ser linear, ramificada, previamente roteirizada de acordo com que o desenvolvedor estabeleceu; (iii) a estética que descreve o tudo que interage como o jogador por meio de seus sentidos de percepção (imagens, sons, sabores e impressões), sendo extremamente importante desde que haja uma relação direta com a experiência de jogo do jogador e (iv) as tecnologias que descrevem o plano de desenvolvimento do jogo, como ambiente de execução (computadores, smartphones, tablets, videogames), sistemas operacionais (Windows, Android, iOS etc.), decisão a respeito dos mecanismos de desenvolvimento (nativo, híbrido, web). Ainda segundo o autor, cada um desses elementos possui a mesma ordem de importância, e possuem um equilíbrio ou idealmente deveriam possuir.

A tecnologia é baseada onde aquele framework trabalha e qual sua linguagem de desenvolvimento. a mecânica envolve as regras e o procedimento de jogo onde ela é mais válida.

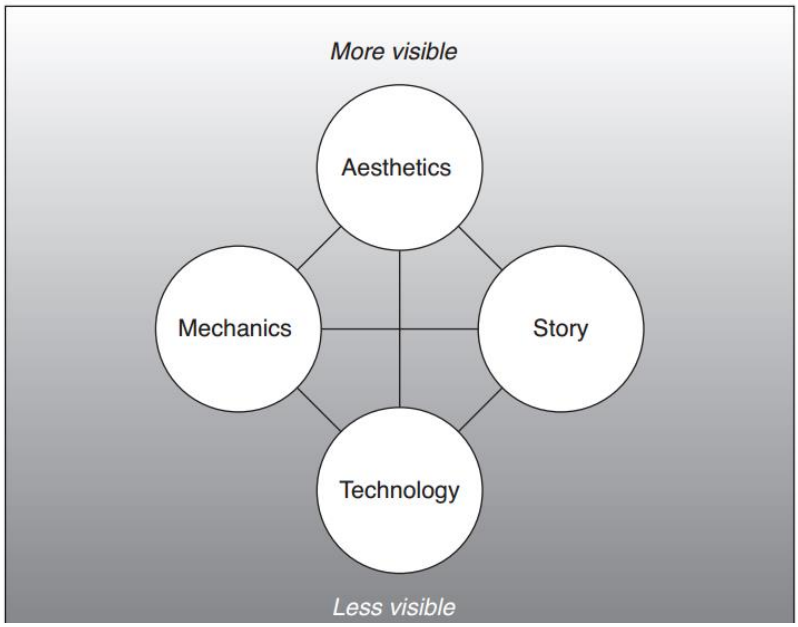

Figura 1 - Estrutura de elementos de um jogo

Fonte: Schell (2008, P.42)

Segundo Lemes, Tomaseli e Camarotti (2012), o mercado de jogos tem crescido muito nos últimos anos, como exemplo, jogos AAA, que demandam muito tempo de desenvolvimento e alta lucratividade, de última geração podem superar o valor de 100 milhões de dólares para seu desenvolvimento e com advento das novas tecnologias computacionais e os esportes eletrônicos, cada vez mais empresas e desenvolvedores estão migrando para essa área.

Segundo Newzoo (2016), o mercado de jogos em 2016 foi de 99,6 bilhões de dólares e pode chegar a 118 bilhões de dólares até 2019 como mostra a Figura 2.

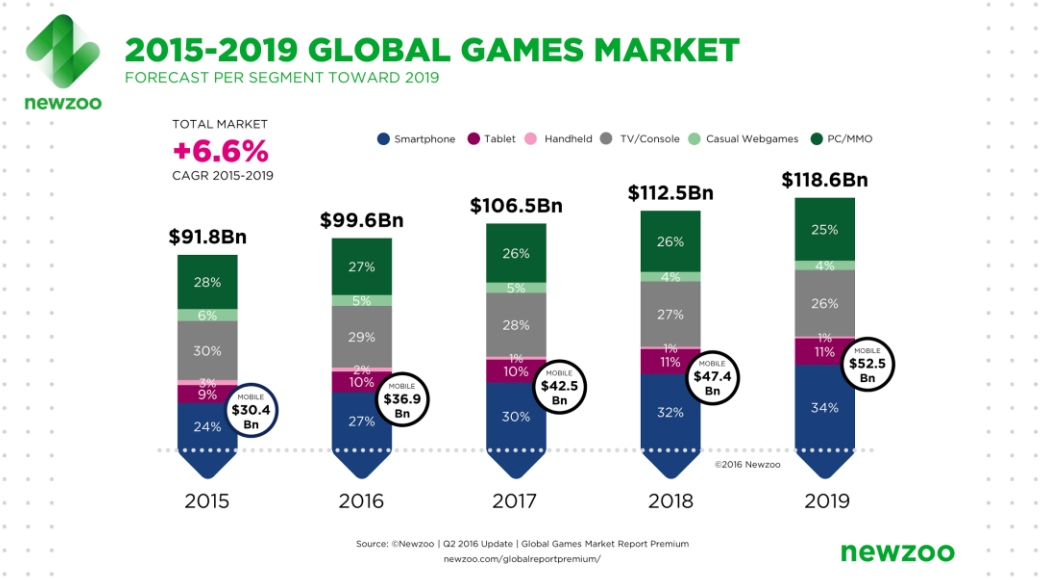

Figura 2 - Mercado de jogos - Newzoo
Fonte: Newzoo (2016, S/N)

Com este nível de expansão no mercado, é natural que entusiastas da tecnologia da informação desenvolvam interesse nessa área devido a oportunidades financeiras e de satisfação pessoal.

Para Superdatasearch (2016), sobre *BigData* especializado em jogos, no ano de 2016, consumidores gastaram 41 bilhões de dólares em jogos *mobile*[1] guiados por *blockbusters*[2] como *clash royale* e *pokémon GO*.

O QUE É UM FRAMEWORK?

Mattson (1996) define *Framework* como uma arquitetura desenvolvida com o objetivo de atingir a máxima reutilização, representada como um conjunto de classes abstratas e concretas, com grande potencial de especialização.

[1] Do Inglês, significa móvel e refere-se a dispositivos móveis como: smartphones, tablets, etc...

[2] Do Inglês, termo que se refere a grandes produções de mídia como cinema, teatro e música.

Segundo Silva e Price (1997), um framework de aplicações orientada a objetos é uma estrutura de classes inter-relacionadas, que constitui uma implementação inacabada, para um conjunto de aplicações de um domínio.

Segundo Sauvé (2000), os frameworks interagem suas estruturas de classes para que elas "conversem" umas com as outras a fim de resolver a atividade para o qual foram desenvolvidas, diferente das biblioteca que nada mais é que um conjunto de classes sem uma estrutura especializada como mostra a figura 3.

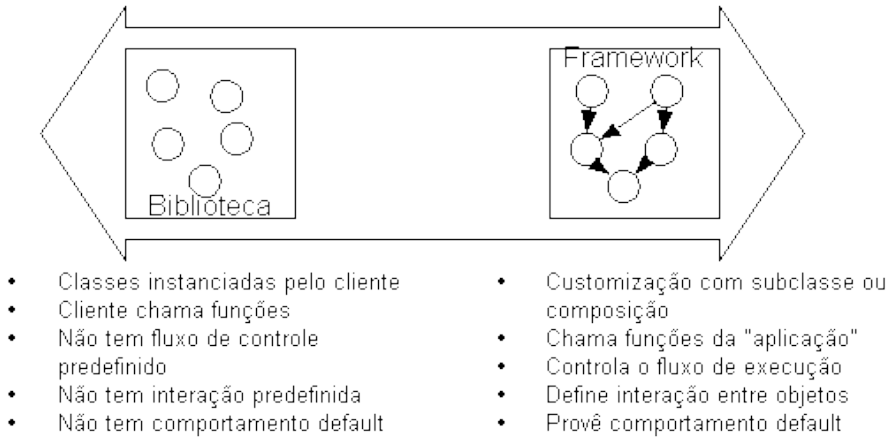

Figura 3 - Diferenças de uma biblioteca e uma *framework*

Fonte:Jacques Sauvé (2000. S/N)

PORQUE USAR UM FRAMEWORK?

O desenvolvimento de jogos a partir do zero não é uma tarefa trivial e nem tampouco desejada, pois conforme visto na revisão, ferramentas como *frameworks* existem para minimizar a codificação de problemas já resolvidos e maximizar a reutilização de códigos já testados por outros desenvolvedores.

Basicamente, a integração de classes e métodos se dá por meio de dependências, uma classe precisa de outras para que complete uma determinada atividade.

AVALIAÇÃO DAS FRAMEWORKS

A melhor *framework* para desenvolvimento de jogos digitais é que melhor atende as especificações do desenvolvedor, tais como compatibilidade com a linguagem usada, a existência de recursos desejados dentro da implementação do framework e o tempo de aprendizado.

Os frameworks possuem características principais muito semelhantes entre elas, a Xna (C#), LibGDX (JAVA) e PhaserJs (JavaScript) são exemplos disso, possuem muitas semelhanças em sua estrutura de desenvolvimento, ainda que escritas em linguagens diferentes.

Todas possuem métodos para iniciar, carregar, atualizar, renderizar e descarregar recursos que não são mais necessários ou inativos como ilustram as Figuras 4-6.

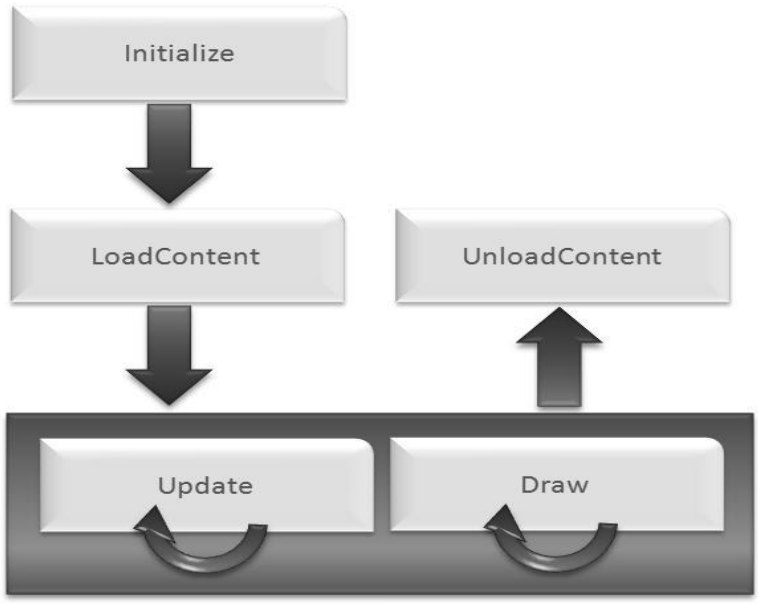

Figura 4 - Ciclo de vida do XNA - **Technical article, into a game**

Fonte: Microsoft (2010, P.5)

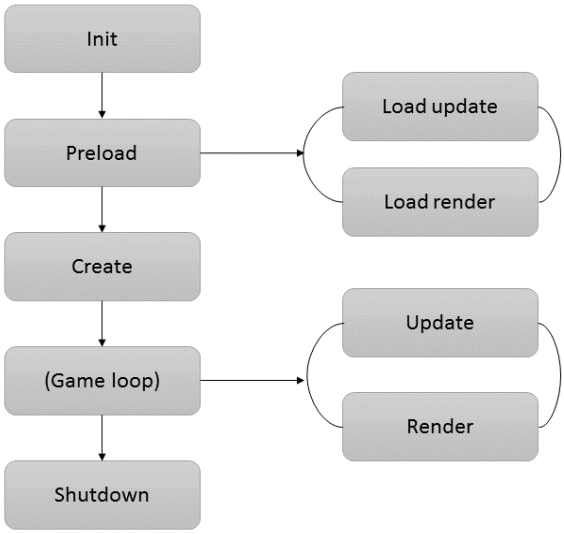

Figura 5 - Ciclo de vida do Phaser.js

Fonte: Travis Faas (2016, P.61).

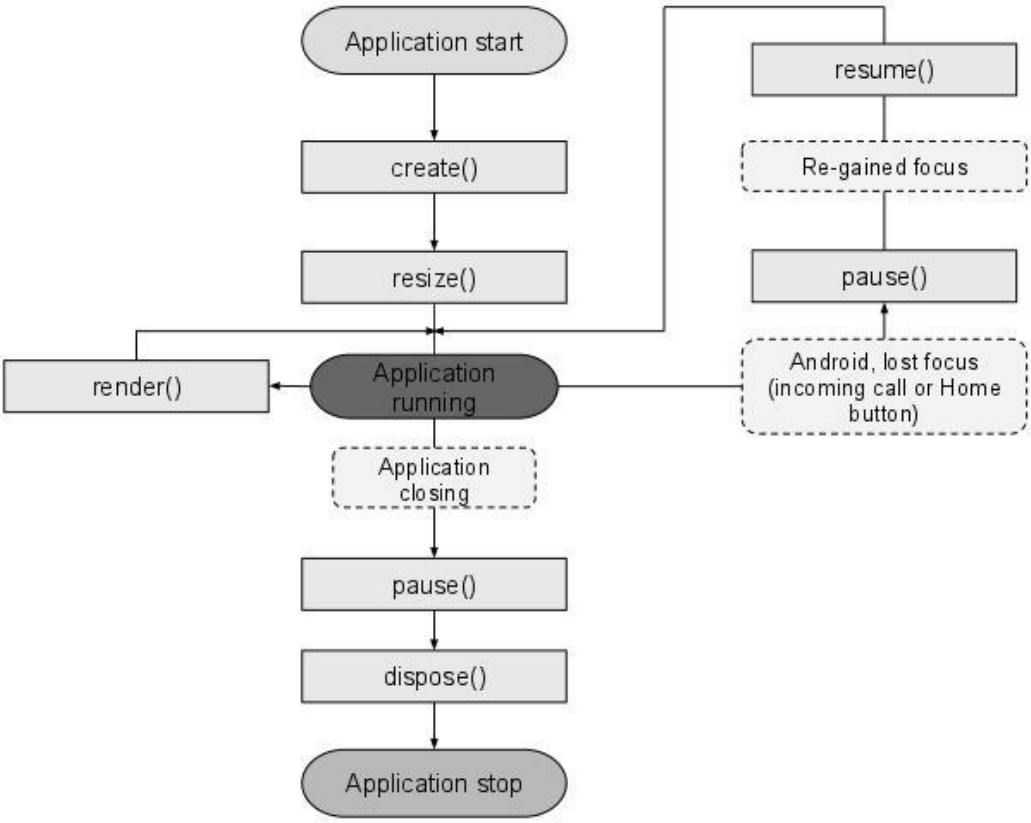

Figura 6 - Ciclo de vida do LibGDX –

Fonte: LibGDX (2015, S/N)

FUNCIONAMENTO DE UMA *FRAMEWORK* PARA JOGOS

Todas os *frameworks* pesquisados possuem um ecossistema de classes com métodos e funções muito bem definidas.

O método "Inicializar" é responsável por iniciar todos os recursos necessários para o desenvolvimento do jogo, tais como recurso para fonte tipográfica, recurso de desenho, recursos de áudio e vídeo, Entrada de comandos (teclado, *joypad*, *touchscreen* etc.), recursos para carregamentos de telas e recursos de câmera.

No método "Carregar" são carregados os recursos tais como, imagens, vídeos, sprites[3] dos personagens, áudio, cenários etc. Esses recursos são alocados na memória para serem processados e exibidos ao usuário (jogador).

No método "Atualizar" são inseridas toda a lógica do jogo que será constantemente atualizado dentro do looping do jogo. Interagindo diretamente com o jogador, exibindo o cenário, música de fundo e sons, estruturados na no logica de jogo aguardando a tomada de decisão do jogador e retornando uma resposta desta decisão. Neste método serão disparados eventos tais como atirar, andar, pular, morrer e a interação do ambiente do jogo com o personagem.

O método "Renderizar", será chamado 60 vezes por segundo todos os objetos desenhados dentro do método, onde o driver de vídeo irá passar para a placa de vídeo para processar a renderização na tela do dispositivo.

No método de "Descarregar" serão descarregados todos os recursos não utilizados no método de "Atualizar" e "Renderizar". Isso é feito para que o jogo ganhe mais performance, pois todo recurso inicializado é alocado na memória. Alguns desses recursos durante o jogo podem não estar mais sendo utilizados pelo jogo. Por esse motivo, o método "Descarregar" irá descarregar os recursos que estão alocados na memória deixando o processador livre para o processamento de tarefas essenciais do jogo. Outra função é a desalocação total desses recursos para fechamento do jogo, limpando totalmente a memória principal.

DESENVOLVENDO UM JOGO

Neste artigo é proposto um jogo no qual o objetivo é desviar o personagem dos inimigos. Quanto maior o tempo de sobrevivência do jogador, maior será a sua pontuação. O jogador terá 3 vidas, ou seja, 3 chances para se manter vivo o máximo de tempo possível.

Em geral, a classe principal de cada *framework* inicia os recursos que serão necessários para todo o jogo. Na primeira tela, será iniciado o carregamento das imagens e sons de todo o jogo dentro da

[3] sprite é o conjunto de imagens que são sobrepostas por computador a fim de criar um efeito de animação. Estas imagens fazem parte de um mesmo arquivo que é mapeado para separar os trechos que serão sobrepostas.

memória principal. Em seguida, a segunda tela utiliza alguns desses recursos para construir a tela de menu de opções que podem conter as opções "jogar", "pontuação" e "sair". Uma vez escolhida a opção "jogar", o jogador será levado a uma terceira tela onde o jogador irá finalmente jogar. Nesta terceira tela é construído o ambiente gráfico e sonoro da fase contendo toda a lógica do jogo que se baseia na mecânica para o qual foi planejado fazendo uso dos recursos carregados na primeira tela. Durante o processamento do jogo em si, o jogador deve clicar nos cantos inferiores da direita e esquerda ou deslizar o dedo para estas direções. O personagem vai acompanhar esses comandos, fazendo assim com que ele possua a habilidade de se esquivar dos inimigos. Esses objetos são reutilizados constantemente no jogo através de métodos dentro do método "atualizar". Quando a vida do jogador chegar a zero, uma quarta tela será exibida para o jogador mostrando a pontuação alcançada. Esses dados são gravados numa lista de pontuação para que o jogador possa comparar com suas pontuações em partidas anteriores. Na opção "pontuação" do menu principal do jogo, será aberto uma outra tela contendo as melhores pontuações do jogador, no qual o jogador poderá conferir o seu melhor desempenho em suas partidas. Já na opção "sair" do menu principal, todos os recursos do jogo serão colocados dentro do método descarregar que irá desalocar os recursos da memória deixando-os disponíveis para outras aplicações.

RESULTADOS E DISCUSSÃO

Apesar de algumas diferenças entre as diversas *frameworks*, em geral, todas possuem conceitos semelhantes tais como: abertura dos arquivos necessários, criação e instanciação de objetos a partir dos arquivos carregados e seu conteúdo, e aplicação da lógica do jogo. Alguns conceitos também se transformam em parâmetros dentro dos *frameworks* como uma unidade de mundo [4]por exemplo, na qual chega-se a um valor para uma medida abstrata equivalente a unidade de metro ou outro tipo de medida. Outro conceito é o de visão de câmera, implementado em todos os *frameworks* levantados neste artigo. Através desse conceito, pode-se decidir aspectos do comportamento da tela do jogo, como por exemplo, a movimentação do cenário de fundo junto com o personagem ou impedir a movimentação da tela para o personagem explorar o ambiente, ou ainda, permitir a movimentação de forma parcial. Elementos de tela também se encontram presentes nesse conceito como dados de pontuação e hit points. Todos esses recursos aceleram o processo de desenvolvimento do jogo pois evitam a codificação e testes de características inerentes ao desenvolvimento de qualquer jogo.

[4] unidade de mundo é uma medida utilizada para facilitar o desenvolvimento do universo de um jogo.

CONCLUSÕES

O desenvolvimento de jogos utilizando *frameworks* demonstrou-se viável conforme demonstrado nos resultados. Ainda que demande um determinado tempo de aprendizagem, o uso deste tipo de tecnologia é especialmente útil para pequenos desenvolvedores, pela facilidade, agilidade e baixo custo.

REFERÊNCIA

BOSE, Juwal. **LibGDX Game Development Essentials, Make the most of game development features powered by LibGDX and create a side-scrolling action game**, Thrust Copter. ed. Packt Publishing Ltd, Livery Place 35 Livery Street Birmingham B3 2PB, UK, 2014.

FAAS, Travis **An Introduction To HTML5 Game Development With Phaser.js**. ed. CRC Press Taylor & Francis Group, 6000 Broken Sound Parkway NW, Suite 300, Boca Raton, FL 33487-2742, USA, 2016.

LEMES, David O., Fernando C. Tomaselli, and S. R. B. Camarotti. "**A economia digital e o mercado de jogos para dispositivos móveis**." SBC-Proceedings of SBGames 1 (2012): 1-5

LIBGDX, **LibGdx Documentation**, Disponível em <https://libgdx.badlogicgames.com/documentation.html>. Acessado em 15 de abril de 2017.

LIBGDX, **LibGDX The Life Cycle**. Data em 20 de dezembro de 2015. Disponível em <https://github.com/libgdx/libgdx/wiki/The-life-cycle>. Acesso em 15 de abril de 2017.

MICROSOFT, **Technical Article, into a game**. Data em 24 de setembro de 2010. Disponível em <xbox.create.msdn.com/downloads/?id=529>. Acesso em 10 de abril de 2017.

NEWZOO, **The Global Games Market Reaches $99,6 Billion in 2016, Mobile Generating 37%**. Data 21 de abril de 2016 Disponível em <https://newzoo.com/insights/articles/global-games-market-reaches-99-6-billion-2016-mobile-generating-37/> Acesso em 13 de maio de 2017.

PHASER>JS, **API Documentation**, Disponível em <https://photonstorm.github.io/phaser-ce/>. Acessado em 17 de abril de 2017.

SCHELL, Jesse. T**he Art of Game Design, a book of lens.** ed. Morgan Kaufmann Publishers is an imprint of Elsevier. 30 Corporate Drive, Suite 400, Burlington, MA 01803, USA, 2008.

SILVA, Ricardo Pereira, and Roberto Tom PRICE. **"O uso de técnicas de modelagem no projeto de *frameworks* orientados a objetos."** Proceedings of 26th International Conference of the Argentine Computer Science and Operational Research Society (26th JAIIO)/First Argentine Symposium on Object Orientation (ASOO 97). Buenos Aires, Argentine. 1997.

SUPERDATASEARCH, **Market Brief - Year in Review 2016**, Disponível em:<https://www.superdataresearch.com/market-data/market-brief-year-in-review/ >. Acessado em 13 de maio de 2017

STEMKOSKi, Lee. **Beginning Java Game Development with LibGDX**. Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. 2015.

UFCG, Computação, Jaques Suave. **O que é uma framework?** Disponível em <http://www.dsc.ufcg.edu.br/~jacques/cursos/map/html/frame/oque.htm>. Acessado em 04 de abril de 2017.

YANG, Hsin-Chang. **"A general framework for automatically creating games for learning." Advanced Learning Technologies**, 2005. ICALT 2005. Fifth IEEE International Conference on. IEEE, 2005.

XNA, **Framework Class Library**, Disponível em <https://msdn.microsoft.com/en-us/library/bb203940.aspx>. Acesso em 10 de abril de 2017.