Faculdade de Tecnologia de Americana
Curso Superior de Tecnologia em Desenvolvimento de Jogos Digitais

UMA ABORDAGEM DO MERCADO ANDROID PARA DESENVOLVIMENTO DE JOGOS

CAROLINE OLIVEIRA LOPEZ

Americana, SP
2011

Faculdade de Tecnologia de Americana
Curso Superior de

UMA ABORDAGEM DA PLATAFORMA ANDROID PARA DESENVOLVIMENTO DE JOGOS

CAROLINE OLIVEIRA LOPEZ
carol.lopezo@gmail.com

Trabalho de Conclusão de Curso desenvolvido em cumprimento à exigência curricular do Curso Superior de Tecnologia em Desenvolvimento de Jogos Digitais, sob a orientação do Prof. Me. Jose Alberto F. Rodrigues Filho.

Área: Jogos Digitais

Americana, SP
2010

BANCA EXAMINADORA

-

-

-

AGRADECIMENTOS

Agradeço aos professores e ao meu orientador por toda orientação que me foi dada para concluir esta etapa final do curso. Agradeço também ao espírito opensource da internet e da Google que incentiva e espalha o conhecimento para quem quer.

DEDICATÓRIA

Ao amor da minha vida que me aguenta e está comigo nessa batalha de vida e ao meu pai que sempre me apoiou nas minhas escolhas.

RESUMO

Android é considerada a bola da vez no que se diz a mobilidade. A área de jogos então ganhou um novo e extremamente promissor nicho de mercado, que é o dos smartphones. Embora haja outras plataformas de grande valor para dispositivos móveis, este estudo foca exclusivamente na plataforma Android e desenvolvimento de jogos. Espero um bom apreciamento e utilidade do conteúdo.

Palavras Chave: Android, Mercado, Jogos, Desenvolvimento.

ABSTRACT

Android now have high visibility when it says mobility. The games market then won a new and extremely promising market niche, which is the smartphones. Although there other valuable platforms for mobile devices. This study focuses exclusively on the android platform for market and games development. I hope a good pricing and usefulness of content.

Keywords: Android, Market, Games, Development.

SUMÁRIO

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

2D	Duas Dimensões
3D	Três Dimensões
ADC	Android Developer Challenge
API	Application programming interface
APPS	Abreviação de Aplicativos
GB	Gygabyte
GPS	Global Positioning System
H.264	Padrão para compressão de vídeo
HTTP	Hypertext Transfer Protocol
HSVGA/HVGA	Half-Size Vídeo Graphics Array
I/O	In/Out (Entrada/Saída)
INC	Incorporated
J2ME	Java Platform, Micro Edition
J2SE	Java Platform, Standard Edition
JPEG	Joint Photographic Experts Group
LCD	Liquid crystal display
MB	Megabyte
MP3	MPEG-2 Audio Layer III
MPEG-4	Compressão de formato audio e vídeo
OHA	Open Handset Alliance
PC	Personal Computer
PNG	Portable Network Graphics
QVGA	Quarter Video Graphics Array
RAM	Random-access memory
S.O.	Sistema Operacional
SD	Secure Digital
SDK	Software development kit

TFT	Thin film transistor
XML	eXtensible Markup Language
W3C	World Wide Web Consortium

1 INTRODUÇÃO

Android é uma tecnologia nova que de imediato foi muito bem aceita em geral. Sua aceitação pode vir tanto por ter o nome da poderosa Google por trás quanto pela sua real eficiência ou mesmo pelos dois motivos. O fato é que os números apresentados são muito positivos e crescentes no mercado em geral. A venda de smartphones é alta e progressiva assim como o número de aplicações para *Android*.

A plataforma *Android* é *opensource* e possui além da *Open Handset Alliance* um exército de desenvolvedores que se apoiam e liberam material de estudo, e suporte. Quando se fala em plataforma, que fique claro que, está se falando em um Sistema Operacional. *Android* é um Sistema Operacional baseado em Linux, mas com muitas diferenciações que o torna muito diferente de um Linux, e não mais uma distribuição como muitos pensam.

Dado seu grande aceitamento entre os consumidores, explora-se neste estudo não só a situação do mercado *Android* para os desenvolvedores que querem iniciar essa área e publicar suas aplicações como também a área de jogos. Além do S.O. *Android* ser muito poderoso, o hardware dos aparelhos que carregam o *Android* possui muitos recursos, fazendo com que, o usuário tenha agradáveis experiências em jogos. Esse fato é muito animador para aquele que quer desenvolver jogos e publicar.

Desenvolver jogos é muito mais que apenas codificar centenas, milhares ou milhões de linhas. É preciso conhecer o que está programando, e conhecer a plataforma *Android* é muito importante, visto que deve ser muito bem definido o projeto e tipo de jogo. Seguindo um cuidado antes de começar a escrever o jogo e ter um certo aprendizado da plataforma é essencial para que o sucesso da aplicação em si seja grande. Outros pontos a se somar para um pleno sucesso entre os consumidores são as idéias, criatividade, jogabilidade e marketing do jogo que fica por conta do próprio desenvolvedor ou equipe de desenvolvedores.

2 UMA ABORDAGEM DO MERCADO PARA *ANDROID*

De acordo com Pereira e Silva (2009) a quantidade de telefones celulares vem aumentando massivamente. Atualmente o celular é o produto de consumo considerado como mais utilizado no mundo, cuja quantidade de aparelho corresponde à metade da população do mundo, cerca de 3,3 bilhões (dados extraídos de 2007). Estima-se que até o final de 2013, esse número poderá chegar a 5,6 bilhões.

Na Figura 1 é possível acompanhar a evolução do celular como mostra o gráfico.

Figura 1 – Gráfico.

Gráfico com Quantidade de Celulares (Pereira, 2009).

Contabilizando que a quantidade de celulares na internet gira em um número de três bilhões e neste ano 2011 muito mais, não resta dúvida que este mercado tão promissor tende às inovações constantes para manter o público em dinamismo com as mudanças, visto que cerca de 300 milhões de aparelhos são descartados anualmente, número também em crescimento.

Para se ter uma idéia, segundo Ghandavi e Shah (2010) existe 3.5 vezes mais telefones móveis no mundo do que PC's, e um dos grandes motivos além de todas funcionalidades na mão, está não só a atrativa plataforma mas também o preço dos aparelhos decrescendo, e a gradual melhora da qualidade do aparelho, telas sensíveis ao toque maiores, processadores melhores tornando o dispositivo mais rápido, baixo preço do armazenamento de dados.

E tendo como motivação a inovação na área de telefonia móvel, a Google em uma parceria com outras empresas do ramo Mobile originou a *Open Handset Alliance* dando continuidade ao projeto da *Android Inc*., para competir com outros grandes nomes de plataformas já solidificadas nesse competitivo mercado, como Apple (iPhone OS), Nokia (Symbian), Microsoft (Windows Mobile), RIM (RIM OS for Blackberry). Assim, nasce oficialmente a tecnologia de plataforma móvel *Android*.

Android não é uma linguagem de programação e sim uma plataforma livre cujo crescimento segue de maneira progressiva, que oferece um rico ambiente para que desenvolvedores terceiros possam criar suas aplicações, e trazer mais inovações em utilizando a sua API (*Application Programming Interfaces*). Ou seja, Android é uma plataforma completa para operadoras de telefonia móvel, desenvolvedores e fabricantes de celulares para produção em massa de dispositivos móveis, softwares e serviços.

2.1 Segurança

Ainda seguindo as referências de Pereira (2009), o autor explica que Android embora tenha sido criado em uma base de Linux, este não é como um ambiente Linux e nem suporte diversas bibliotecas nativas aos Linux. Neste ponto entra a questão de segurança do Android, afinal o usuário se pergunta o quão seguro será o ambiente utilizado em seu telefone móvel, em que muitas informações e transações serão realizadas, uma vez que o Smartphone deixa de ser um telefone armazenador de poucos dados de uma agenda, e pequenos programas e passa a ser um multitarefa móvel em plataforma *opensource*, extremamente interativo e com muitos dados do cliente armazenados – dados extremamente confidenciais. Definitivamente, o smartphone é a evolução do telefone móvel associada à evolução dos computadores de mão.

Neste caso podemos dizer que o Android é tem seu grande nível de segurança dada sua maneira de programar, baseada em permissões na estrutura do código do *AndroidManifest*.xml (será citado posteriormente). A inicialização do telefone é similar a do Linux, que primeiro carrega do seu core (*Kernel*) e carrega

usuários isolados para cada aplicativo, impedindo invasões entre aplicativos. Um programa malicioso neste caso não consegue invadir os dados pertencentes de outro programa ou conteúdo nativo do telefone.

2.2 Proposta de liberdade

De acordo com greatereader.org (2011) quando a Google comprou o Sistema Operacional Android em 2005, a proposta foi aperfeiçoá-lo e libera-lo para um uso comum, de maneira que fosse totalmente customizável, mas sem perder funções nativas. Analogamente, se pensarmos no *Kernel* do Linux, este é o mesmo em várias *releases* do Linux, Fedora, Red Hat, Ubuntu, Kurumin, Slackware etc. Cada um possui a sua característica que o torna mais amigável visualmente ou não, porém a base não acessível ao usuário final ou desenvolvedor de aplicações de plataforma alta, é a mesma.

Sendo assim, o Android foi liberado para ser utilizado como base de aplicativos compatíveis. E assim as operadoras de telefonia foram aderindo, sendo o primeiro telefone Android um HTC lançado em 2008.

O que mais faz entusiastas da tecnologia louvar a plataforma é o quão aberto ele é para que fabricantes desenvolvam suas ferramentas e usuários instruídos também o façam.

2.3 Qualquer um pode fazer o seu Android *apps*.

Embora não seja simples o desenvolvimento de aplicativos, visto que utiliza uma linguagem de programação orientada a objetos, *xml* e outros conceitos (que serão abordados detalhadamente nos próximos tópicos abordando o ambiente e desenvolvimento de jogos), a Google pretende simplificar para os mais leigos.

Sabe-se que nesse mercado aquecido dos *smarthphones*, o grande sucesso tanto do iPhone OS quanto do Android são os *apps*. *Apps* são nada mais que aplicativos relacionados a qualquer fim, business, entretenimento, educacional,

jogos etc. Os *Apps* são todos comercializados ou não, na loja virtual Android, conhecida como Android Market. Podem ser *free* ou pagos, e essa movimentação que chega a bilhões de downloads torna o cenário muito atrativo. Pensando nisso, a Google quis facilitar o desenvolvimento de *apps* para usuários que não possuem vasto conhecimento em programação, trata-se de um aplicativo chamado *App Inventor* for *Android*. Para desenvolver *apps* e inclusive poder ganhar dinheiro com isso, faz o usuário se sentir muito atraído com a plataforma e certamente vai querer um smartphone para manusear e conhecer melhor a tecnologia, nisto há grande chance de ter um usuário fiel à tecnologia que sempre estará consumindo produtos e aparelhos.

A proposta do *App Inventor* é, desenvolvimento de apps Android para não-desenvolvedores, utiliza o conceito de blocos de funções que são agregados como um “lego” em vez de linhas de códigos.

É possível assistir um tutorial do funcionamento do *App Inventor* no *Youtube*, basta procurar por *App Inventor In Action.*

Nisso, um usuário leigo pode desenvolver inclusive jogos simples. E dessa maneira, o mercado de *Apps* se mantém ativo e muito movimentado, seja por *Apps* de desenvolvedores experientes, aprendizes ou mesmo leigos de programação que com a criatividade pode desenvolver *apps* muito interessantes e cheios de atrativos.

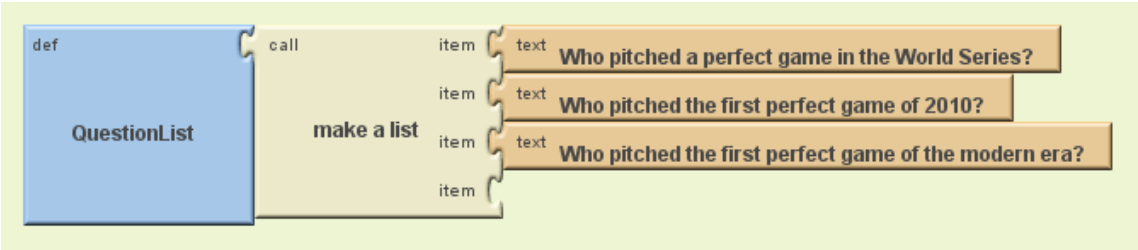

Figura 2 - *App Inventor For Android*..

Desenvolver *apps* sem saber programar. techrepublic.com 2010

3 ANDROID NO SENTIDO DO *OPENSOURCE*.

(Krogh et al., 2006, p. 975) O software livre é normalmente criado dentro do software de código aberto projetos, muitas vezes iniciadas por um indivíduo ou um grupo focado neste interesse, que é desenvolver software ou produto para atender às necessidades de um mercado que vai consumir o produto final.
produto para atender às necessidades do consumidor.

Nas plataformas de código aberto, desenvolvedores de aplicativos podem desenvolver e executar o código através de um kit de desenvolvimento de uma forma eficaz. Da mesma forma, o código fonte desenvolvido pode ser usado por qualquer desenvolvedor e para que este seja modificado e melhorado ganhando versões e *updates* em cima da versão anterior. Um desenvolvedor pode enviar uma atualização rápida para os consumidores através do canal de distribuição. Há também certas regras e procedimentos acordados por todos os membros dessa comunidade em particular em que um padrão deve existir sem que fuja do foco do produto.

.

3.1 Android nos *Smartphones*

A tabela a seguir, é uma fonte da *Gartner Newsroom* que mostra a quantidade de *Smartphones* vendidos para usuários finais. Comparando ainda, com 2008 vê-se que o avanço e a popularidade dos mesmos, e com isso, ganha o mercado de software por trás dessas plataformas.

Worldwide Smartphone Sales to End Users in 2009 (Thousands of Units)				
Company	**2009 Sales**	**Market Share (%)**	**2008 Sales**	**Share (%)**
Nokia	440,881.6	36.4	472,314.9	38.9
Samsung	235,772.0	19.5	199,324.3	16.3
LG	122,055.3	10.1	102,789.1	8.4
Motorola	58,475.2	4.8	106,522.4	8.7
Sony Ericsson	54,873.4	4.5	93,106.1	7.6
Others	299,179.2	24.7	248,196.1	20.3
Total	**1,211,236.6**	**100.0**	**1,222,252.9**	**100.0**

Tabela 1 - Vendas *smartphones*.

Quantidade de Androids vendidos no mundo, 2008, 2009 (Gartner Newsroom, 2010)

Ainda nessa mesma pesquisa da Gartner, constatou que tipos de plataformas – Sistemas Operacionais – que estavam nestes Smartphones, e é muito perceptível o crescimento do uso do Android ainda em seu *debut*. Vide Tabela 3, a seguir:

Company	2009 Units	2009 Market Share (%)	2008 Units	2008 Market Share (%)
Symbian	80,878.6	46.9	72,933.5	52.4
Research In Motion	34,346.6	19.9	23,149.0	16.6
iPhone OS	24,889.8	14.4	11,417.5	8.2
Microsoft Windows Mobile	15,027.6	8.7	16,498.1	11.8
Linux	8,126.5	4.7	10,622.4	7.6
Android	6,798.4	3.9	640.5	0.5
WebOS	1,193.2	0.7	NA	NA
Other OSs	1,112.4	0.6	4,026.9	2.9
Total	**172,373.1**	**100.0**	**139,287.9**	**100.0**

Tabela 2 - S.O.s

Sistemas Operacionais presentes nos Smartphones – 2008/2009 – (Gardner Newsroom 2010).

Segundo Lisa DeLacey, 2008, a viabilidade do Android nos smartphones mostra o quanto à quantidade de aparelhos aumentou em relação aos Computadores Pessoais, um número que no ano de 2008 chega a triplicar em relação aos PCs, como mostra a figura a seguir.

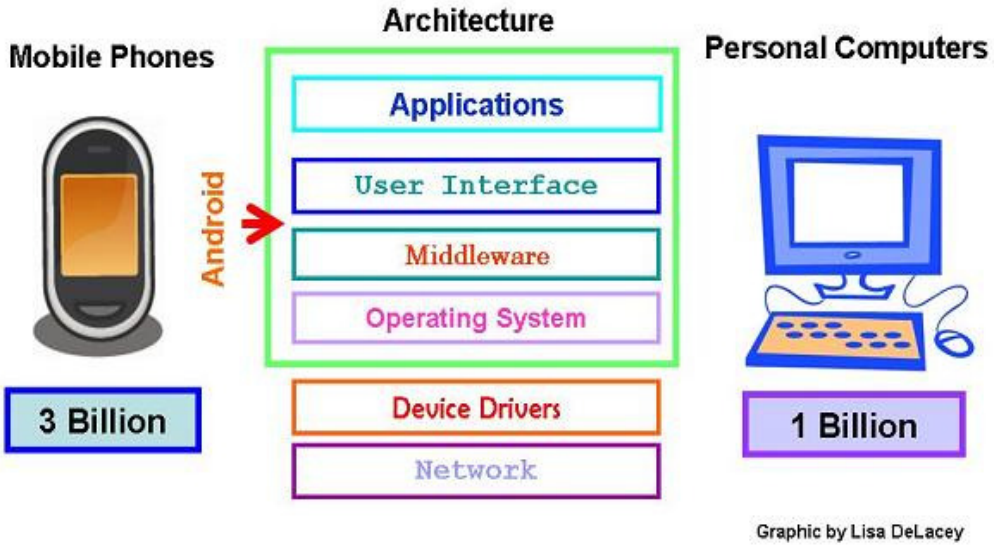

Figura 3 - Arquitetura

Além de ver que a quantidade de smartphones com Android ter triplicado em relação aos PCs, é possível visualizar de maneira simplifica a arquitetura envolvida. Fonte: onlamp.com/onlamp - Lisa DeLacey, 2008.

3.2 Google e Android Marketing

A principal razão da Goggle ter entrado nesse mercado tão emergente foi focando nos anúncios e como e para promover o seu grande sonho, o Sistema Operacional Google que pudesse ser executado em qualquer dispositivo móvel fabricado por diferentes fornecedores tais como Motorola, LG, Samsung, HTC etc.

Dessa forma, os clientes podem comprar aparelhos Android de fabricantes e operadoras distintas, tendo em mãos um dispositivo capaz de rodar aplicações poderosas e em que o cliente tem a plena liberdade de customizá-lo, bem como desenvolver suas próprias ferramentas para uso próprio ou distribuir.

O código fonte do Android está disponível para todos os desenvolvedores de software poder agregar conhecimento, modificar para atualizações futuras, e realizar melhoramentos da versão já existente.

A visão da Google é que para o Android tenha as funções de um computador em um dispositivo móvel, fazendo que com o Pc não faça “falta”. Para que isto seja

possível, a Google criou a já citada Open Handset Alliance, que hoje é um grupo com cerca de 65 empresas do ramo tecnológico que se juntaram para promover o software livre alimentado pela Google. Essas 65 empresas são divididas em três grupos:

1) Fabricantes de celulares como HTC, Motorola, Samsung e etc.
2) Os desenvolvedores de software como o eBay
3) Operadores móveis como a T-Mobile, Sprint, Do Co Mo, etc.
4) Fabricantes de Chips, como a Broadcom, QUALCOMM, Marvell, Intel, etc.

E num acordo comum, uma versão de Android é lançada de modo que sejam compatível todos smartphones contemplados por essa aliança Android. O objetivo é tornar a plataforma viável para móveis e também publicas um código aberto.

A figura a seguir ilustra o processo do sistema da *Open Heandset Alliance*:

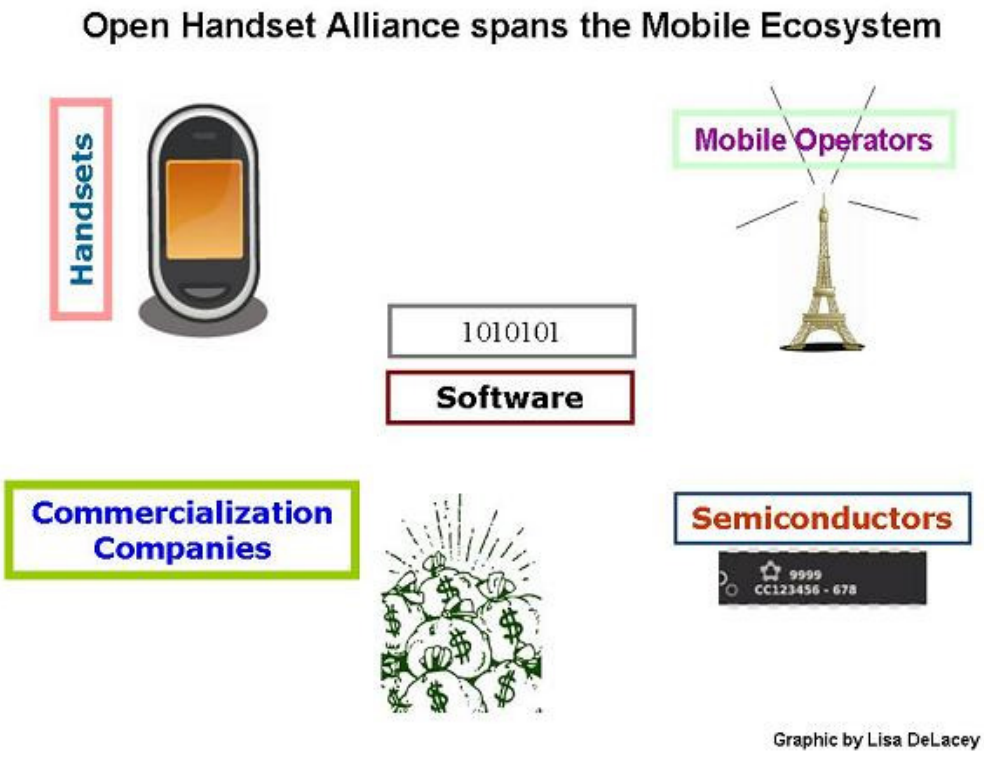

Figura 4 - Arquitetura da OHA

Fonte: http://onlamp.com/onlamp/2007/11/12/google-calling-inside-the-gphone-sdk.html - Lisa DeLacey, 2008

A plataforma Android é composta de várias camadas que proporcionam uma completa pilha de software. A camada inferior é o Kernel do Linux, tendo acima um sistema de bibliotecas incluindo gráficos 2D e 3D, *Dalvik* que é uma máquina virtual, a estrutura do aplicativo, e todos os aplicativos para o usuário final. A parte mais

poderosa da plataforma é o *Dalvik* máquina virtual, que interpreta e executa código Java otimizado para operar na plataforma móvel. Como a maioria das aplicações hoje em dia são relacionadas com a web, as duas primeiras camadas são escritas em Java. Com todas estas funcionalidades, Android é complementada pela camada de aplicação que inclui um navegador da Web, toque mensagens de tela, GPS, instantâneas, câmera de telefone etc. Uma das melhores características deste plataforma é a possibilidade dos desenvolvedores poderem estender de maneira nunca antes pensada. Assim, pode-se dizer que é um recurso completo para o sistema móvel.

Visualizado os downloads de aplicativos Androids baixados para Smartphones, temos a seguintes estatísticas de acordo com a *Androidlib*:

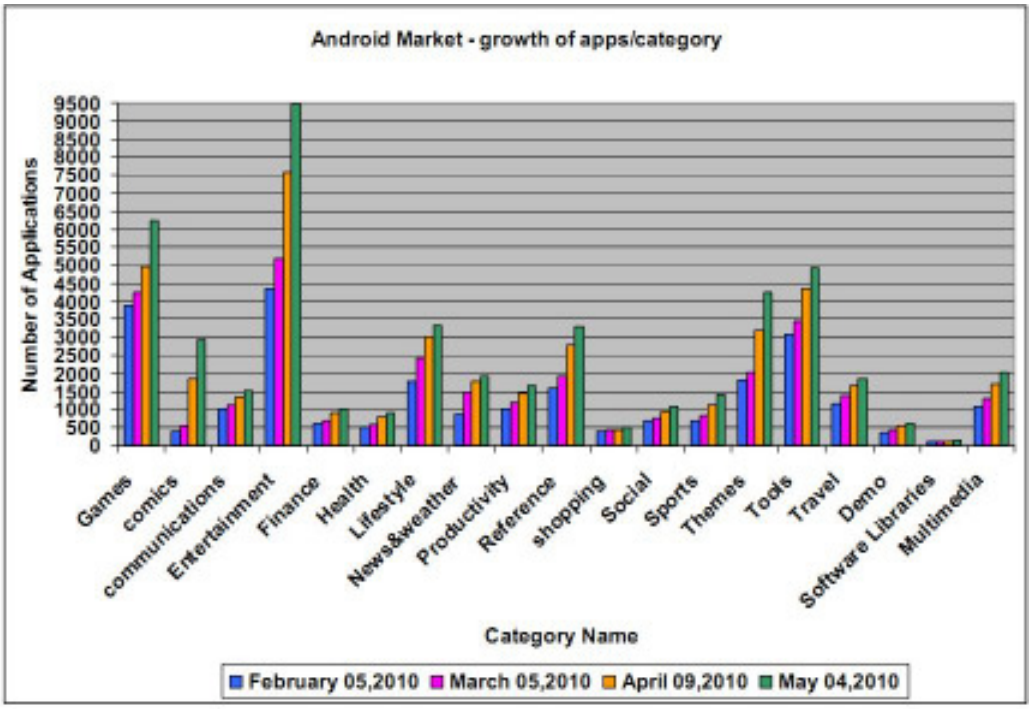

Figura 5 - Android Market

Numero de Aplicações/Categoria/Mês (Androidlibe, 2010).

Percebe-se que Games e Entretenimento lideram a preferência dos usuários.

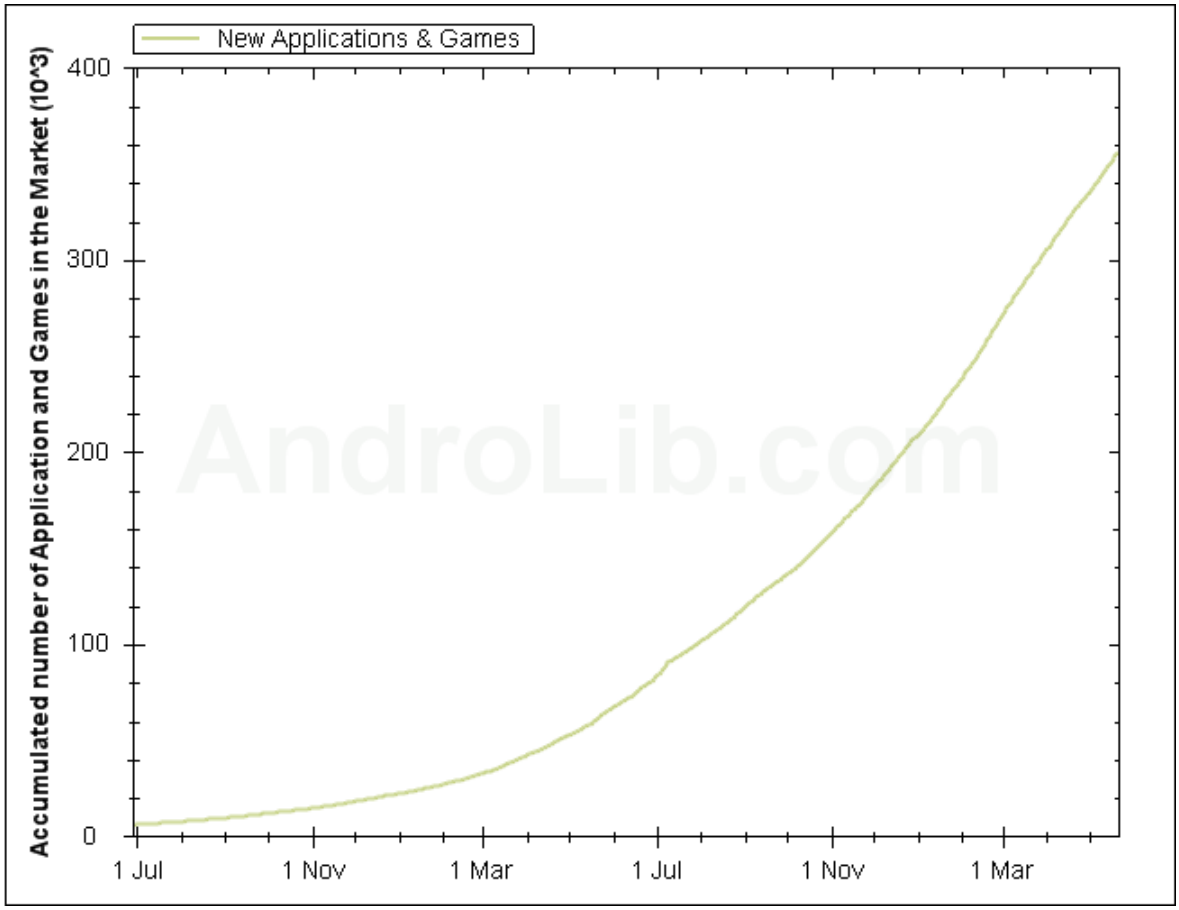

Figura 6 - Android Market

Número acumulado de Aplicações e Games no Android Marketing (Androidlib, 2011)

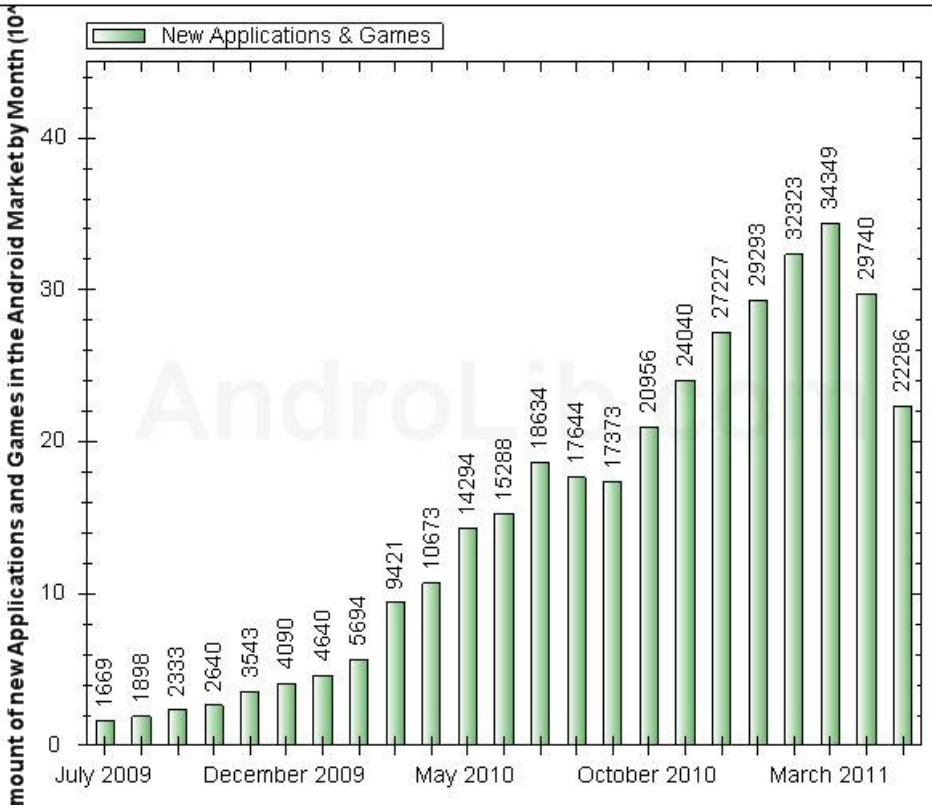

Figura 7 - Novas aplicações

Número de novas aplicações e games por mês – *Androidlib* 2011

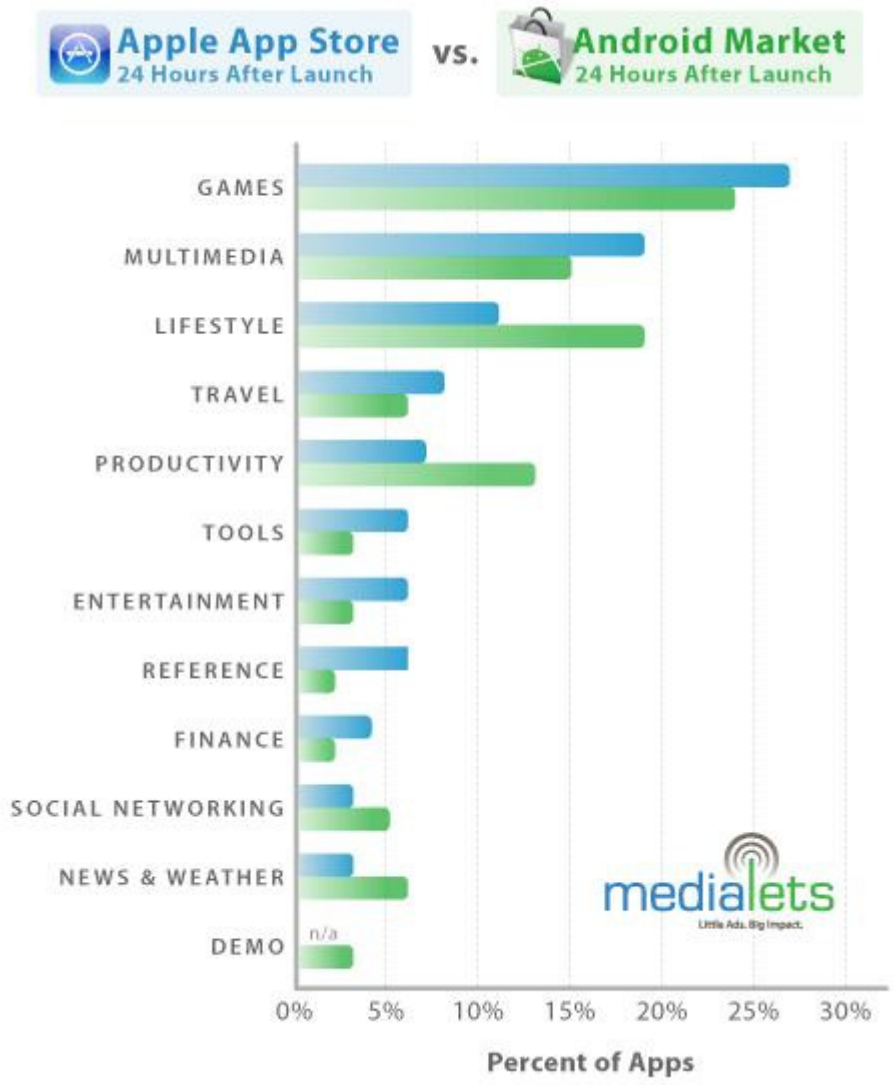

Figura 8 - Comparação

Comparação da *Android Makert* e da *AppStore*, aplicativos. (Androidcentral, 2011).

3.3 Fatores para selecionar e desenvolver na plataforma Android

Discutindo sobre o desenvolvimento do Android, a maioria dos desenvolvedores disse que Android está a crescendo num ritmo mais rápido em comparação com o iPhone e que gostariam de mudar para o Android no futuro próximo. Em suas opiniões, além do código do Android ser aberto este também oferece um pacote SDK completo com arquiteturas, *APIs*, bibliotecas e

muitos outros avançados. Mas apesar de todos esses fatores positivos, muitos desenvolvedores afirmam que as ferramentas do SKD Android não estão bem avançadas e necessitam reforços de recursos para aumentar mais ainda o ritmo de desenvolvimento das aplicações.

3.4 Aprovações

A Google não criou nenhum processo específico ou política de aprovação, portanto qualquer um pode disponibilizar sua aplicação em um mercado online. Para vender um aplicativo Android existem vários mercados disponíveis, ou seja, não há aquela obrigação da aplicação vir do *Android Market*. Qualquer um pode criar o seu próprio mercado para distribuir aplicações Android.

3.5 Fragmentação

Para Richard Taylor, 2011 a grande flexibilidade do Android tem um preço: companhias que optam por desenvolver suas próprias interfaces têm que capturar num ritmo mais rápido assim que as novas versões do Android são liberadas. Isso pode tornar os telefones em questão de meses obsoletos, assim como os fabricantes de celulares e operadoras se recusam a criar atualizações que se incorporam nestes aparelhos.

A fragmentação tem muitas faces. Para o usuário final, significa que ele está impossibilitado de instalar e usar determinadas aplicações e recursos porque está preso a um versão mais antiga de Android. Para os desenvolvedores, significa que alguns cuidados devem ser tomados quando criam aplicações que devem rodar em todas as versões de Android. Enquanto aplicações que são escritas para versões anteriores do Android rodam bem nas versões mais firmes, o contrário não é valido. Alguns recursos adicionados às versões mais novas do Android são obviamente não válidas para versões mais antigas assim como suporte multi-toque. Desenvolvedores são forçados a criar códigos separados para diferentes versões do Android.

3.6 Desafios, Envios de Aparelhos e Google I/O

(Richard Taylor, 2011) Em um contínuo reforço para chamar mais desenvolvedores para a plataforma Android, Google começou a manter desafios. O primeiro desafio, chamado *Android Developer Challenge* (ADC) foi lançado em 2008, oferecendo relativamente altos prêmios em dinheiro para projetos vencedores. O ADC foi realizado no ano subseqüente tendo novamente um enorme sucesso em termos de participação de desenvolvedores. Não houve ADC em 2010, o que pode ser provavelmente atribuído ao Android que possui uma considerável base de desenvolvimento não precisando mais de esforços para absorver novos desenvolvedores. A consolidação do Android para os desenvolvedores tem uma data, e esta é 2010.

A Google também iniciou um programa de envio de aparelhos no início de 2010. Cada desenvolvedor que tivesse uma ou mais aplicações no *Market* com mais de 5.000 downloads e uma média de qualificação dos usuários de 3.5 estrelas ou mais, recebia um novo Motorola Druida, Motorola Milestone ou Nexus One. Isso foi uma ação muito bem recebida dentro da comunidade dos desenvolvedores, visto que inicialmente foi uma idéia recebida com descrédito.

A conferência anual chamada Google I/O é um evento em que cada desenvolvedor Android aguarda com expectativa cada ano. No Google I/O, as últimas e maiores tecnologias e projetos Google são revelados, entre os quais Android tem ganhado um lugar especial nos anos mais recentes. Os tópicos do Google I/O relacionados ao sistema Android podem inclusive ser encontrados no canal *Google Developers* no *Youtube*.

4 ESTRUTURA ANDROID

Um desenvolvedor que queira programar em Android, seja aplicações comerciais, entretenimento ou jogos, devem conhecer bem a estrutura dessa plataforma tão formidável.

(Richard Taylor, 2010) Android não é apenas mais uma distribuição Linux para dispositivos móveis. No lado do desenvolvimento, Android é uma plataforma com abstrações subjacentes do *Kernel* do Linux e é programada via Java, e numa vista de alto-nível, Android possui recursos interessantes.

- Um Framework que provê um rico conjunto de *APIs* para criar vários tipos de aplicações. Também permite a reutilização e a substituição de componentes providos pela aplicação e por aplicações terceiras.
- A máquina virtual *Dalvik* que é responsável por rodar as aplicações Android.
- Um conjunto de bibliotecas para programação 2D e 3D.
- Suporte a mídia áudio, vídeo e imagem nos formatos *Ogg* Vorbis, MP3, MPEG-4, H.264 e PNG. Há inclusive uma API especializada para efeitos sonoros de fundo, que acompanham grande parte dos jogos.
- APIs para acessar periféricos como câmera, GPS (*Global Positioning System*), bússola, acelerômetro, suporte a toque de tela, *trackball* e teclado. Como nem todos aparelhos que carregam o Android possui esses periféricos, vemos que o uso de todos os recursos depende muito do hardware do aparelho.

Android possui muito mais recursos, mas para desenvolvimento de jogos, estes citados acima são os mais relevantes.

A arquitetura do Android é composta por uma pilha de componentes, e cada componente constrói componentes numa camada abaixo. A Figura 9 mostra um resumo dos maiores componentes do Android.

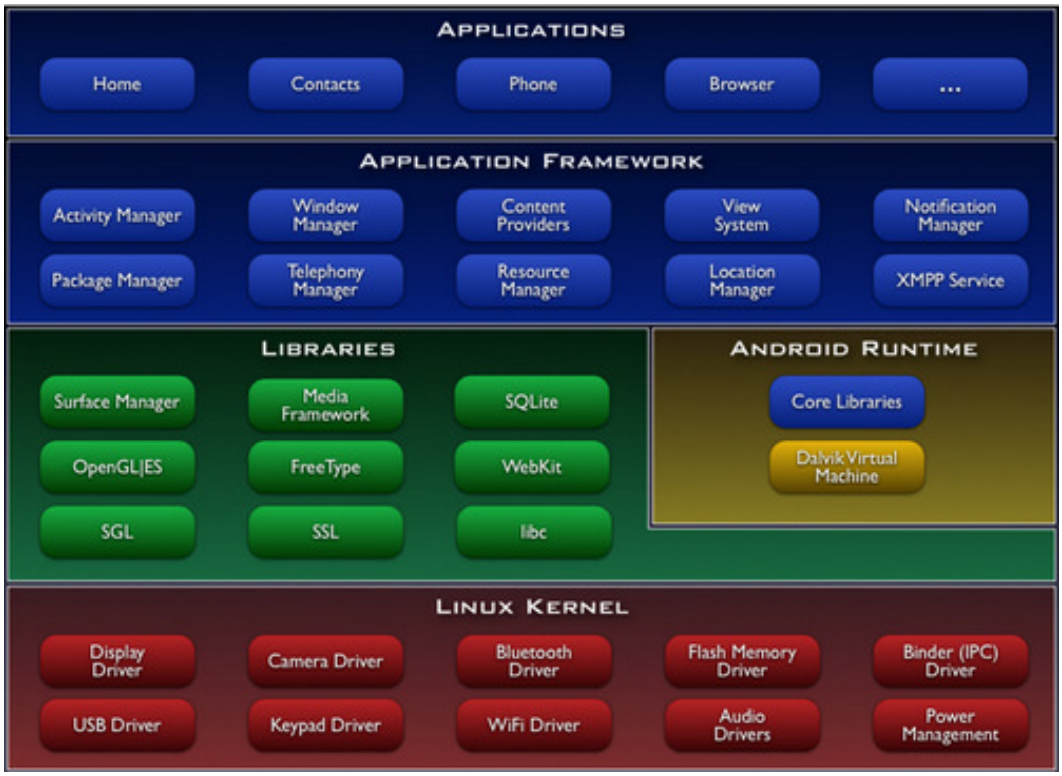

Figura 9 - Arquitetura

Arquitetura Android – Anddev.org- 2007

5 JOGOS PARA ANDROID

De acordo com (Justin Bacon, 2011) sobre jogos em Android é possível afirmar que hoje jogos casuais estão nas grandes ondas do espaço móvel. Num momento da já consolidada existência de dispositivos móveis com a finalidade para jogos o advento dos smartphones que são capazes de suportar jogos teve o efeito de por o *game player* portátil nos bolsos de milhões de pessoas que nunca comprariam um dispositivo dedicado somente a jogos. Por definição, este novo grupo de jogadores é casual, eles jogam jogos enquanto esperam numa fila ou estão no ônibus. Pelo menos à primeira vista, essas pessoas não estão interessadas em grandes títulos caros; elas querem apenas cuidar de suas fazendas virtuais ou resolver pequenos quebra-cabeças. Por conta disto, a maioria dos jogos são 2D.

A plataforma Android possui sua API com grande capacidade para desenvolvimento de jogos. Bibliotecas, e o ambiente de desenvolvimento, bem como suas interações com as camadas de negócios possibilitam a criação tanto de jogos 2D como 3D.

A primeira coisa que o desenvolvedor de jogos deve fazer é selecionar qual tecnologia ele quer usar para implementar o jogo. Basicamente, o desenvolvedor tem que escolher entre 2D e 3D. Se ele escolhe o 3D é requerido para o design do jogo, então ele escolhe o *OpenGL ES* porque esta livraria oferece todas as coisas que ele precisa para desenvolver seu jogo. Se escolher o 2D, ele pode ainda assim utilizar o *OpenGL ES*, mas a complexidade deste biblioteca pode complicá-lo no caminho. A outra escolha para o 2D é usar as capacidades de construir gráficos dentro do próprio SDK do Android.

5.1 Android para Jogos vs. Java ME para jogos

Antes mesmo do Android se popularizar, já existia a versão do Java própria para dispositivos móveis denominada J2ME (*Java Mobile Edition*). A API J2ME que possibilita programar em Java para celulares, smartphones, e inclusive programar Jogos. Android, que também utiliza a base Java foi lançada com o intuito de facilitar. No que se refere a games, Android consegue ser superior ao J2ME?

(Vladmir Silva, 2009) A API do Java Android é diferente da utilizada nas APIs no Java SE e no Java ME, que são as versões Standard e mobile respectivamente. Este fato inclusive defendido por muitos, e publicado na Wikipédia é o fator de grande criticidade em relação ao Android.

- Android não usa um *Kernel* de Linux convencional. Não possui um executável nativo de sistema Windows, e não suporta as bibliotecas padrão do GNU, fazendo com que a reutilização existente nas aplicações Linux ou bibliotecas seja uma dificuldade.
- Android não utiliza os já estáveis padrões de Java (Java SE e Java ME). Isso impede a compatibilidade entre as aplicações Java. Assim como não provê a implementação completa de todas classes, bibliotecas e APIs do Java SE e Java ME.
- Android anterior à versão 2.2 não permitia oficialmente que aplicações seja instaladas ou rodem a partir de um *SD Card* (cartão de memória). Isso é um grande erro pois muitos jogos 3D avançados usam grandes arquivos para armazenar gráficos e sons. Armazenar estes arquivos no sistema principal de arquivos pode rapidamente esgotar precioso espaço em disco. A partir da versão 2.2 também conhecida como *Froyo*, o acesso ao SD card por aplicações e instalações.

Essa deficiência mostrada acima, ainda de acordo com (Vladmir Silva, 2009), são evidentes apenas observando as bibliotecas do Android dentro de sua IDE. Considerando a figura 10 abaixo, que mostra as classes das bibliotecas do android.jar. No lado direito é possível notar que a maioria das *packages* do Java SE estão inclusas, o que é bom. No entanto, algumas classes usuais para jogos como a *java.awt.Polygon* e *java.awt.Dimension,* estão faltando. Embora na versão 1.5 do SDK da Google foi adicionado um importante propriedade do *Java Bean* para troca de suporte (veja a *java.beans.packages*). A Figura do meio mostra onde faz Android e o Java SE serem dois animais diferentes. Nove classes no pacote do Android são parte dos padrões do Java SE e ME. Finalmente, no lado esquerdo, é possível ver que a Google está reutilizando alguma bibliotecas nativas dos Apache

Software Foundation (*Commons HTTP e http client*) e a *W3C XML APIs* (*under org.xml*)., (Vladmir Silva, 2009).

.

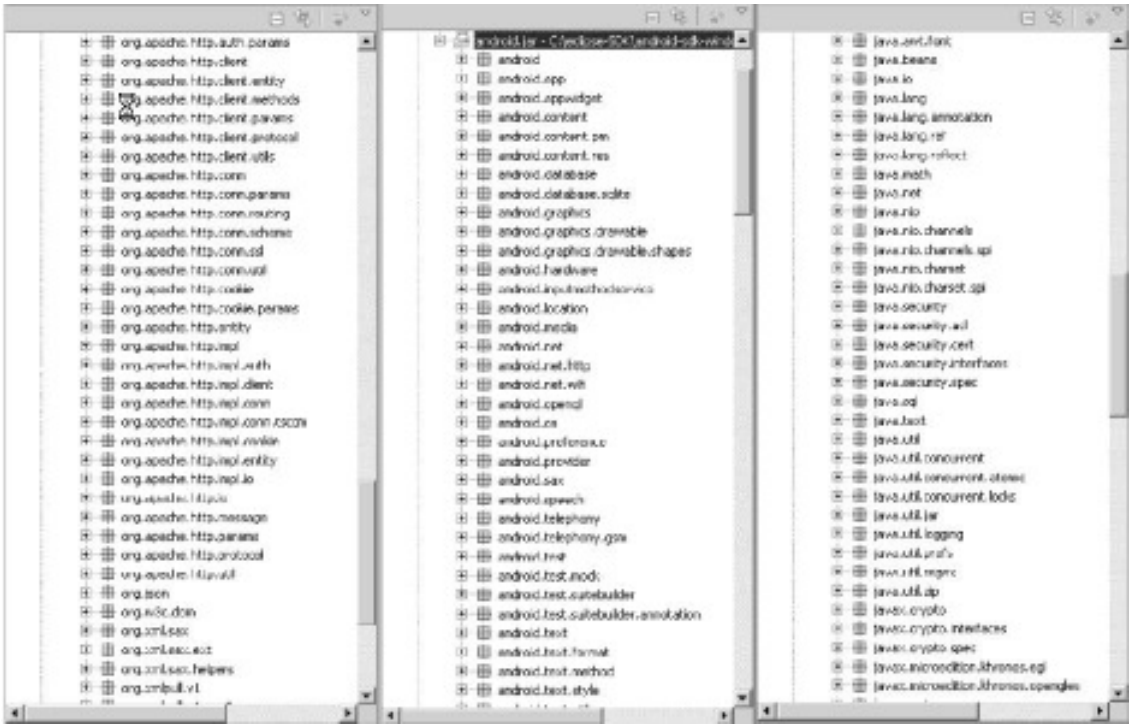

Figura 10 - Bibliotecas

Figura que demonstra as diferenças do Android para o Java Padrão – (Vladimir Silva, 2009).

6 ENTENDENDO O ANDROID PARA JOGOS

A partir do momento que o desenvolvedor se foca em jogos, este além dos conhecimentos da plataforma quanto criação de telas, interação, e conhecimento em Java, o diferencial é o conhecimento quanto à multimídia. Além de ter o computador devidamente preparado com Java, API Java (podendo ser Eclipse ou NetBeans), e o Android SDK devidamente instalado, o desenvolvedor precisa conhecer as bibliotecas 2D e 3D, aplicar sons, fazer uso de imagens, e animação. Como já citado anteriormente, as APIs do Andoid possui muitos recursos e bibliotecas que auxiliam a criação de jogos que demandam um maior apelo visual e interativo, visto que o objetivo, no caso de jogos que serão disponibilizados no mercado é, basicamente, entreter o jogador, e acima de tudo satisfaze-lo para que o jogo seja recomendado e alcance o êxito esperado.

É necessário igualmente ter em mente que tipo de jogo planeja e de tal maneira, conhecer o hardware dos smartphones que irão mais frequentemente ter esse jogo instalado. (Richard Taylor, 2011) Não há requisitos mínimos para um dispositivo que leve Android. Em todo caso, a Google tem recomendado as seguintes recomendações de hardware que praticamente todos os dispositivos Android suprem e que muitas vezes ultrapassam significativamente:

- *ARM-based* CPU: Embora Android também rode em arquitetura x86, os últimos dispositivos *ARM-based* também estão sendo muito utilizado para o recurso dual-core.
- 128MB RAM: Esta é a especificação mínima. Atualmente aparelhos de alta qualidade já possuem 512MB RAM, e 1GB RAM já é algo esperado num futuro bem próximo. Isso significa que, softwares e jogos cada vez mais pesados e complexos podem ser iniciados nesta plataforma.
- 256MB de memória flash: O mínimo de memória flash para armazenamento de imagens do sistema e aplicações. Por um longo tempo, essa pilha de memória foi a maior estranhamento entre os usuários Android porque aplicações terceiras podiam ser instaladas apenas na memória flash. Isso mudou com a nova versão *Froyo*.

- Armazenamento de Mini ou Micro SD: A maioria dos aparelhos vem com poucos gigabytes de armazenamento nos SD card, que podem ser substituídos por SD cards de maior capacidades pelos próprios usuários.
    - 16-bits de cores *Half-Size Vídeo Graphics Array* (HSVGA) TFT LCD com tela de toque (*touch screen*): Versões anteriores a 1.6, apenas telas HVGA (480x320 pixels) eram suportadas pelos sistema operacional. Desde a versão 1.6, baixas e altas resoluções de tela são suportadas. Os atuais aparelhos mais sofisticados possuem tela *Wide Video* ou (852x480 pixels), e alguns aparelhos de menor qualidade suportam telas *Quarter-Size Vídeo Graphics Array* (QVGA) com resoluções mais baixas (320x280 pixels). As telas de toque quase sempre são capacitivas e *single-touch*, ou seja, aceitam apenas um toque de cada vez e não múltiplos, isso na maioria dos aparelhos mais antigos.

É particularmente crucial que o desenvolvimento de jogos estejam dedicados a unidades de processamento gráfico (GPU)s. Os aparelhos mais antigos que rodavam Android já possuiam o *OpenGL ES 1.0* complacente a GPU. Os aparelhos mais modernos possuem GPUs com desempenho comparada ao Xbox ou Playstation 2 e suportam *OpenGL ES 2.0*. E se não há processador gráfico disponível, uma reserva na forma de renderização de software chamada *PixelFilinger* é provida por plataforma. Muitos aparelhos de baixo custo dependem do software renderizador, que muitas vezes é suficientemente rápido para telas de baixa resolução.

Juntamente com processadores gráficos, qualquer aparelho Android disponível no momento também possui hardware de som dedicado. Muitas plataformas de hardware também possuem um circuito especial para decodificar diferentes formatos de mídia como H.264 no hardware.

7 JOGOS EM POTENCIAIS PARA ANDROID

(Mario Zencher, 2011) Provavelmente o maior segmento de jogos no Android Market consiste nos chamados jogos casuais. O termo ***jogo "casual"*** *é utilizado para caracterizar jogos digitais (de videogame, jogos de computador ou aparelhos móveis) acessíveis ao grande público. Diferentemente dos jogos tradicionais que são mais complexos e exigem tempo de dedicação do jogador, os jogos casuais são simples e rápidos de aprender. Desta forma podem ser uma opção de diversão para um simples passatempo de alguns minutos. (Wikipedia, 2011).*

A seguir é possível ver as telas de alguns jogos casuais em Android:

(Mario Zencher, 2011) Abduction e Abduction 2 (Figura 11), por Psym Mobile, é um exemplo perfeito de jogo casual. Derivando do subgênero de jogos de pulos. O objetivo do jogo é diretamente fazer a vaca pular de uma plataforma para outra até alcançar o topo do nível. É possível adquirir poderes para alcançar o topo mais rápido, e o controle da vaca vem do jogador, obviamente, influenciando na direção, do pulo ou queda. Controles fáceis de entender, objetivo claro, gráficos concisos fez do jogo um dos primeiro sucessos do Android Market.

Figura 11 - Abduction

Abduction (esquerda) e Abduction 2 (direita), por Psym Mobile.

Antigen (Figura 12), por Battery Powered Games, é um jogo casual em que o jogador é um anticorpo que luta contra diferentes tipos de vírus. O jogo por possuir mais recursos, acaba se tornando mais envolvente que Abduction.

Figura 12 - Antigen

Antigen por Battery Powered Games

Falando em jogos casuais, são muitos subgêneros entretanto sempre há aquele mais populares, como além dos citados acima, os jogos do tipo quebra-cabeça (puzzles) como Tetris não deixam de ser igualmente populares, dispensando introdução.

Super Tumble (Figura 13) é um soberbo exemplo de um quebra-cabeças de Física. O objetivo do jogo é remover os blocos tocando-os, e pegar a estrela acentada nos blocos de maneira segura no fundo da plataforma. Ao mesmo tempo que o jogo parece simples, pode se entender por vários níveis, dificultando a jogada.

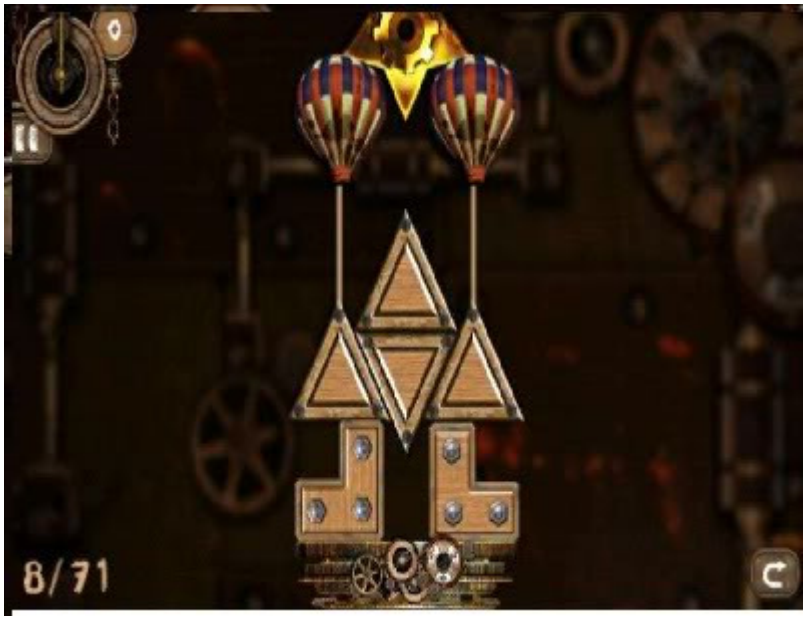

Figura 13 - Super Tumble

Super Tumble, por Camel Games

U Connect (Figura 14), por BitLogik, é um jogo minimalístico mas um divertido quebra-cabeça. O objetivo é conectar todos os pontos do gráfico com uma única linha.

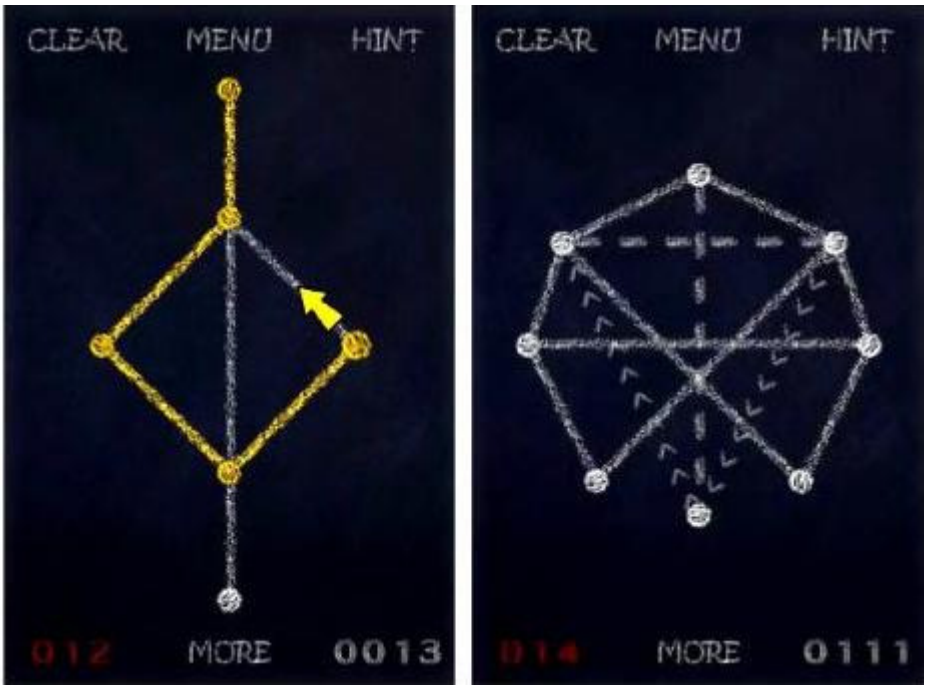

Figura 14 - U Connect

U Connect por BitLogik

7.1 Jogos tipo Arcade e Ação

(Richard Taylor, 2011) Jogos de ação e árcade em geral aproveitaram o grande potencial da plataforma Android. Muitos deles utilizando os recursos dos visuais 3D, demonstrando o que é possível nessa nova e atual geração de hardware. O gênero possui muitos subgêneros, incluindo jogos de corrida, jogos de tiro, e tiros em terceira pessoa. Este segmento no Android Market ainda está pouco desenvolvido, assim como grandes companhias que possuem os recursos para produzir tais títulos hesitam em pular na onda do Android. Alguns desenvolvedores mais “indie” acabaram pegando esse mercado para preencher o nicho.

Exemplos de tais tipos de jogos são muitos, mas demonstrando alguns, tem-se o *Replica Island* (Figura 15) que é provavelmente o de maior sucesso na plataforma Android atualmente. Desenvolvido pelo engenheiro da Google e desenvolvedor de jogos Chris Pruett em uma demonstração de que apenas um pode escrever jogos de alta performance em puro Java no Android. O jogo tenta acomodar todo o potencial das configurações do aparelho oferecendo uma grande variedade de esquemas. Foi tomado um cuidado especial para que o jogo tivesse bom desempenho em aparelhos de baixo custo, com hardware inferior. O jogo em si constitui-se em um robô que é instruído a recuperar um misterioso artefato. O mecanismo do jogo se assemelha aos da plataforma do antigo SNES 16-bit. Em uma configuração padrão, o robô se mexe via um acelerômetro e dois botões, um para pular e outro para acertar os inimigos. O jogo é open source.

Figura 15 - Replica Island

Replica Island, por Chris Puett

Exzeus (Figura 16), por HyperDevBox, é um dos tiros em um espíritos de Starfox no SNES, com alta fidelidade nos gráficos 3D. Os recursos do jogo no todo são: diferentes armas, poderes, lutas como grande chefe, e muitos inimigos para atirar. A principal característica é o controle pela inclinação dos botões na tela. – o esquema de controle do jogo é bem intuitivo para esse tipo de jogo.

Figura 16 - Exzeus

Exzeus, por HyperDevBox

Deadly Chambers (Figura 17), por Battery Powered Games, é um jogo de tiro de terceira pessoa semelhante aos clássicos *Doom* e *Quake*. O Objetivo é, o Dr. Chambers, tenta escapar dos perigos do mago malvado na torre. A produtora do jogo optou por mantê-lo no padrão simples em não elaborar uma história de fundo para o atirador. Mas o princípio do jogo faz com que realmente não precise de história, apenas deve-se atirar nas criaturas que aparecem na frente, que são inimigas e fazer o atirador seguir o seu caminho. O controle principal do jogo se dá por um joystick virtual na tela. Ao contrário de Exzeus, o desenvolvedor tomou cuidado para que o jogo rodasse bem em aparelhos mais baratos. O jogo é considerado um grande feito visto que foi programado por uma pessoa em um período de somente seis meses.

Figura 17 - Deadly Chambers

Deadly Chambers, por Battery Powered Games

Outro jogo interessante que trás um clima nostálgico é o Radiant (Figura 18), desenvolvido pela Hexagon, representando não só o visual *old-school* como também inovando alguns conceitos da jogabilidade. Como exemplo, o controle da nave se dá balançando o aparelho. Destaque também para o visual estilo Atari.

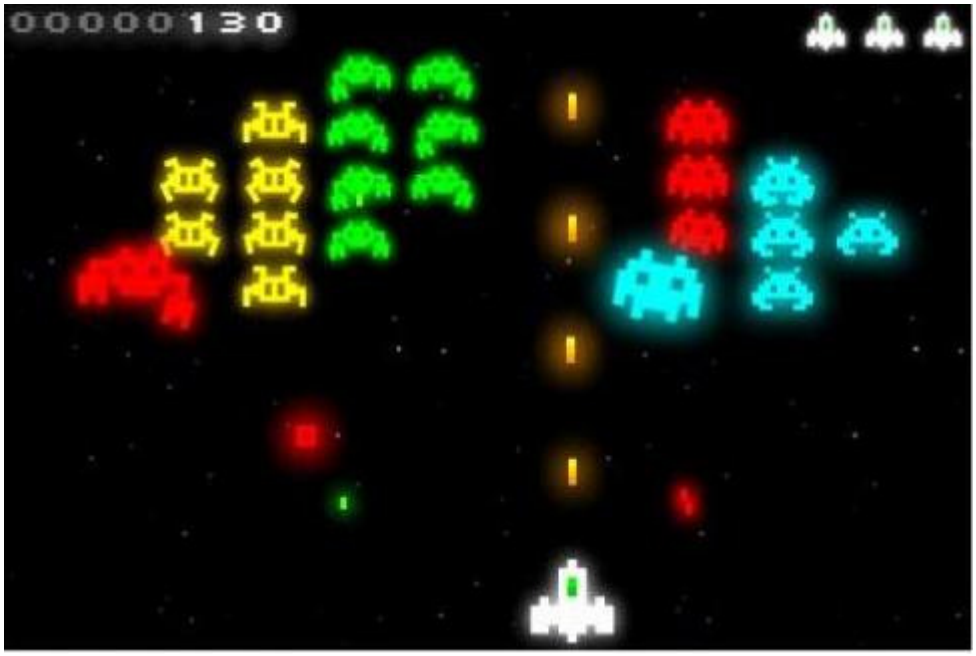

Figura 18 - Radiant

Radiant, por Hexage

7.2 Jogos de corrida

Um exemplo de um jogo aclamado na era Play Station é Need For Speed (Figura 19) que também possui sua versão para Android, produzido pela Eletronic Arts. O jogo utiliza o acelerômetro e pela balança do aparelho é possível controlar o carro na corrida.

Figura 19 - Need For Speed

Need For Speed para Android – Eletric Arts - Tekmobile 2010

7.3 Jogos inovadores

Alguns jogos não podem apenas estar em uma categoria. Eles exploram as novas capacidades dos recursos dos aparelhos Android, como câmera ou o GPS para criar nova variedade de experiências. Esse tipo de jogo introduz elementos da realidade aumentada. (Mario Zechner, 2011).

Um exemplo de jogo assim é o SpecTrek (Figura 20) que foi um dos vencedores da segunda edição do Android Challenge. O objetivo do jogo é vagar com o GPS ativado para encontrar fantasmas e pega-los com a câmera. Os fantasmas são colocados sobre a câmera, e o jogador precisa focá-los e pressionar o botão para fotografá-los e assim pontuar, (Mario Zechner, 2011).

Figura 20 - SpecTrek

SpecTrek, por SpecTrekking.com

8 DESIGN DE UM JOGO PARA ANDROID

De acordo com Richard Taylor 2010, é muito importante que antes de começar a codificar o jogo, plenejá-lo no papel. Para o autor, os melhores passos iniciais para o jogo é esboçar os seguintes itens:

- O núcleo do mecanismo do jogo.
- Uma história de fundo simplificada com as principais características.
- Um esboço dos estilos dos gráficos baseados na história do jogo e suas características.
- Esboço de todas as telas, assim como diagramas de transição entre as telas, interligando a tela de game over.

Exemplos de esboços de telas podem ser vistas na Figura 21 abaixo:

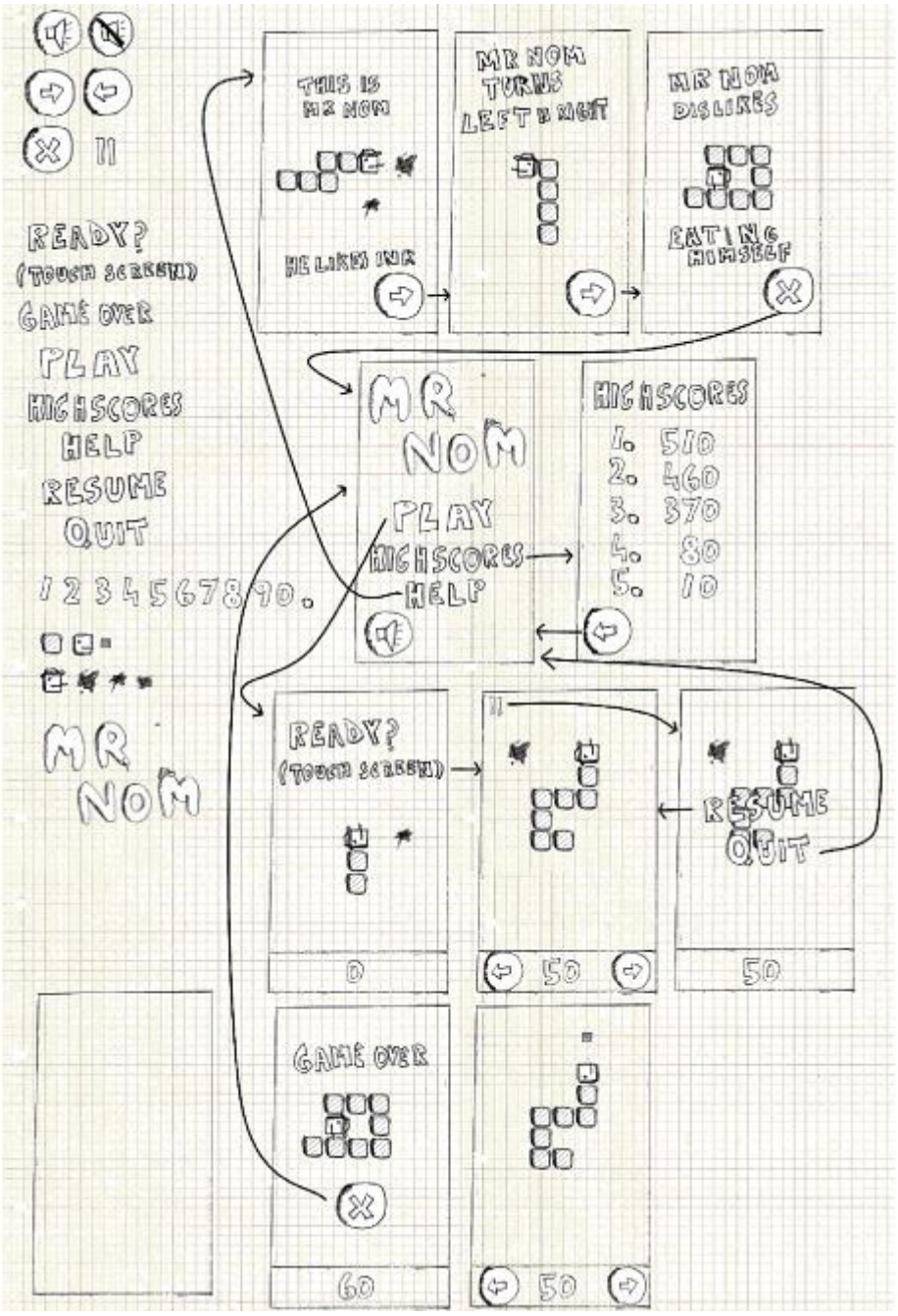

Figura 21 - Design

Design de um jogo tipo Snake, todos os desenhos de telas, botões e as transições - Richard Taylor 2010.

Embora esse seja uma maneira muito interessante de iniciar o projeto de um jogo, este modelo de esboço no papel é mais apropriado para jogos pequenos. Jogos grandes e complexos normalmente são feitos em equipe com todo um gerenciamento mais profissional adotando inclusive alguma metodologia ágil, em que cada um possui o seu papel no time.

9 CODIFICAÇÃO

Depois de conhecer as APIs necessárias do Android para programar jogos, ter uma ideia do design, é hora de transformar toda idéia no jogo em si, ou seja, é preciso começar a codificar o jogo. Apesar de todo misticismo em volta do significado de programar o jogo, muitas vezes pela falta de conhecimento e experiência, a interface Android se torna incrível por duas razões: permite que o programador se concentre na semântica sem precisar conhecer os detalhes da implementação, e permite, também fazer uma trocar de implementação (por exemplo, em vez de usar CPU com renderização 2D, pode-se usar OpenGL ES).

De acordo com Richard Taylor o framework de um game é dividido nos seguintes módulos:

- Gerenciamento de janelas
- Inputs
- File I/O
- Gráficos
- Audio

Cada um desses módulos é composto por uma ou mais interfaces. Cada interface terá pelo uma implementação concreta que implementa semântica da interface baseada no básico que plataforma de fato provê para o desenvolvedor.

9.1 Gerenciamento de janelas

O gerenciamento de aplicações e janelas tem a finalidade de gerenciar a aplicação quanto ao seu ciclo de vida e o seu funcionamento se é em *background* ou *foreground*. Os eventos são:

Create: chamado quando a janela (e portanto a aplicação) é inicializada.

Pause: chamado quando a aplicação é pausada por algum mecanismo.

Resume: chamado quando a aplicação é resumida, ou analogamente, quando a aplicação sai do *background* (ou minimizada) para o *foreground* (maximizada e priorizada).

9.2 Inputs

Richard Taylor, 2010. O usuário naturalmente precisa e quer interagir no jogo de alguma maneira, e é nesse momento que o módulo de inputs entra em ação. Eventos de inputs como toques de tela e pressionamento de botões são o focados para a janela ativa da aplicação. Esses eventos de toques e pressionamento dos botões podem ser gravados e tratados para uso do interesse do jogo.

Android possui três métodos de inputs relevantes: *touchscreen*, teclado/*trackball* e acelerômetro.

9.2.1 Touchscreen

Pode-se gerar três eventos:

- *Touch down*: Acontece quando o dedo toca a tela.
- *Touch drag*: Acontece quando o dedo se arrasta pela tela, seguido sempre de um *touch down*.
- *Touch up*: Acontece quando o dedo solta a tela.

9.2.2 Teclado

O teclado pode gerar dois eventos:

- *Key down*: Acontece quando a tecla é pressionada para baixo.
- *Key up:* Acontece quando a tecla é solta, sempre precedido do *key down.*

9.2.3 Acelerômetro

O acelerômetro reporta a aceleração exercida pela gravidade pelo planeta em um dos três eixos do acelerômetro. Os eixos são x, y e z como mostra a figura x. A aceleração em cada eixo é medida em metros por segundo ao quadrado (m/s^2).

Quando um ponto do eixo se move do centro da Terra, a aceleração máxima é aplicada nele. Pode exemplo se o telefone é segurado para cima em orientação retrato, então o eixo y reportará uma aceleração de 9.8 m/s². A figura 22, o eixo z reportaria uma aceleração de 9.8 m/s², e o eixo x e eixo y reportariam uma aceleração nula.

Figura 22 - Acelerômetro

Os eixos do acelerômetro em um aparelho de telefone Android. O eixo z aponta para fora. Richard Taylor, 2010.

9.3 File I/O

Leitura e escrita de arquivos é essencial para o desempenho no desenvolvimento do jogo. Geralmente passa a ser necessário escrever em arquivos quando se deseja manter configurações de placar ou mesmo salvar o jogo de algum ponto para que o jogador possa continuar de onde parou.

Lembra-se que é muito importante o uso de Exceções caso dê algo errado no arquivo e não prejudique o jogo no geral. (Richard Taylor 2010).

9.4 Audio

O áudio de um jogo, em muitos caso é algo essencial. Se o jogo em si necessita de áudio para que a vida do mesmo tenha mais sentido, então a sonoridade no todo deve ser tratada com muito cuidado e dedicação. Apesar de as chamadas do evento em si não serem complexas no Android, alguns pontos devem ser levados em consideração, na questão de tratamento de áudio.

(Richard Taylor, 2010) Sabe-se que uma música de 3 minutos pode demandar muito espaço na memória. Quando uma música de fundo é tocada no jogo, ela é tocada em amostras em tempo real ao invés de ser pré-carregada por completa na memória. Normalmente, num jogo é apenas uma música tocando, sendo um acesso ao disco apenas.

Para pequenos efeitos sonoros, como explosões, tiros de armas, a situação é um pouco diferente. Neste caso é preciso tocar um efeito sonoro múltiplas vezes simultaneamente. Transmitindo amostras sonoras do disco para cada instância de efeito sonoro não é uma boa idéia. Entretanto, sons curtos não tomam muita memória. Deve-se, portanto, ler todas as amostras de um efeito sonoro para a memória, de onde pode-se diretamente e simultaneamente tocá-los.

9.5 Gráficos

O ultimo modulo é o dos gráficos. As imagens obviamente devem ser desenhadas, para a tela, o que parece fácil. Entretanto para um alto desempenho dos gráficos deve-se conhecer o básico de programação gráfica (cores, pixels, compressão, buferização etc), o que pode ser um assunto bastante extenso visto que, o recurso visual é riquíssimo tanto pela capacidade da plataforma Android, quanto a capacidade hardware dos aparelhos.

Dessa maneira é possível trabalhar tanto com 2D quanto 3D, inclusive imagens JPEG ou PNG.

10 ANDROID PARA DESENVOLVEDORES DE GAMES

(Mario Zechner, 2011) Uma vez conhecida a API e para que cada uma pode ser utilizada em Android, deve-se saber que a aplicação é constituída por muitos componentes sendo os principais:

- Activities: é a interface com a qual o desenvolvedor fará os componentes se interagirem na tela.
- Serviços: São processos que trabalham em segundo plano e não são visíveis.
- Provedores de conteúdo: São os componentes que fazem parte dos dados disponíveis da aplicação do desenvolvedor para que outras aplicações possam acessar.
- Intents: São mensagens criadas pelo sistema operacional ou pelas próprias aplicações, que são passadas para qualquer aplicação interessada. Intentes notificam o usuário os eventos de sistema como a remoção do SD card ou o cabo USB sendo conectado. Intents também são utilizadas pelo sistema para iniciar componentes da aplicação que está sendo desenvolvida, como as activities. Pode-se fazer inclusive com que as Intents da aplicação em questão executem ações, como abrir uma galeria de fotos para mostrar imagens ou aplicação de Camera para bater uma foto.
- Broadcast receivers: Este reaje a específicas intents, a pode executar ações como iniciar uma determinada activity ou enviar algum outra intente para o sistema operacional.

Uma aplicação Android não possui um único ponto de entrada, como se é comumente utilizado sistema operacional de desktop (por exemplo na forma do método Java main()). Ao invés disso, os componentes de uma aplicação Android são iniciliazados ou solicitados para executar uma determinada ação por intents específicas.

Quais componentes ou aplicações são compostas e que intents tais componentes reagem são definidos no arquivo manifest da aplicação. O sistema Android utiliza este arquivo manifest para conhecer o que a aplicação é feita, como a activity principal que será mostrada quando a aplicação iniciar.

O arquivo manifest serve para muitos propósitos além de definir os componentes de uma aplicação. A lista a seguir resumo as partes relevantes do arquivos manifest no contexto de desenvolvimento de jogos:

- A versão da aplicação exibida e utilizada no Android Market.
- As versões de Android que a aplicação pode ser executada.
- Especificações de hardware que a aplicação requer (por exemplo, multitouch, resoluções de tela, ou suporte a OpenGL ES 2.0).
- Permissões para uso de determinados componentes, como escrita no cartão de memória ou acesso a internet.

11 PROJETO DE UM JOGO EM ANDROID EM 10 PASSOS

(Richard Taylor, 2010) Sumarizando todo o “como fazer” de um projeto de jogo em Android, utilizando o Eclipse devidamente configurado espera-se o seguinte do projeto:

- Deve está apto a utilizar os recursos da versão mais recente do SDK enquanto mantém compatibilidade com as versões mais antigas que alguns aparelhos ainda rodam. Significa que deve suportar Android 1.5 pra cima.
- Deve ser instalado no cartão de memória quando possível a fim de evitar o uso do armazenamento interno do aparelho.
- Deve ser debugável.
- Deve ter uma única activity principal que manipulará todas configurações feitas pelo desenvolvedor e que não são destruídas quando o teclado do hardware aparece ou a orientação do dispositivo muda.
- A activity deve ser fixa tanto para modos retrato ou paisagem.
- Deve permitir acesso ao cartão de memória.
- Deve permitir o acesso normal assim que é desbloqueada a tela do aparelho.

Tendo isso, eis os dez passos:

1. Criar um novo projeto Android no Eclipse abrindo a tela de diálogo “New Android Project”.
2. Neste diálogo, especificar o nome do projeto e determinar a versão com a qual este será desenvolvido.
3. No mesmo diálogo, determinar o nome do jogo, o pacote em que todas as classes serão armazenadas (package) e o nome da principal activity. Setar a mínima versão do SDK que é 3. Pressionar Finish para que o projeto se torne realidade.
4. Abrir o arquivo AndroidManifest.xml.
5. Fazer com que o jogo possa ser instalado no cartão de memória quando este estiver disponível. Adicionar o atributo installLocation no <manifest> elemento e configurá-lo para preferExternal.

6. Tornar o jogo debugável, adicionar o atributo debuggable para o elemento <application> e deixa-lo como true.
7. Fixar a orientação da activity, adicionar o atributo screenOrientation para o elemento <activity> e especificar a orientação desejada (portrait ou landscape). Informar também ao Android o controle do teclado, mudanças nas configurações dos atributos keyboardHidden e orientation, configurando o atributo configChanges para o elemento <activity> nos estados keyboard|keyboardHidden|orientation.
8. Adicionar dois elementos <user-permission> para o elemento <manifest> e especificar os nomes dos atributos android.permission.WRITE_EXTERNALSTORAGE e android.permission.WAKE_LOCK.
9. Adicionar o atributo targetSdkVersion para o elemento <uses-skd> e determinar o SDK alvo.
10. Iniciar a programação e testes dos jogo.

12 CONSIDERAÇÕES FINAIS

Toda essa breve análise do mercado da plataforma Android faz com que quem quer seguir nessa linha no que se diz a desenvolvimento este muito otimista. Os números mostram que a adesão do mercado é significativamente crescente, mesmo diante de concorrentes de gande peso e qualidade como as plataformas do iPhone e Blackbarry.

Além de todo o otimismo pelos números do mercado, há o fato da plataforma Android oferecer um ambiente de desenvolvimento muito organizado e com vasto material para aprender, bem como a própria Google mantém o seu repositório de códigos (code.google.com).

Tendo todo um conhecimento em mãos, mesmo o iniciante pode se organizar para desenvolver os seus próprios jogos, e este mercado tem muito a se explorar e desenvolver. A plataforma oferece muitos recursos e o hardware cada vez mais poderoso dos smartphones torna tudo propício a investir em jogos. Jogos é entretenimento, e entretenimento sempre está no topo das listas de consumo virtual, tanto que os dados mostram que jogos está entre os mais procurados e baixados no Android Market.

13 REFERÊNCIAS BIBLIOGRÁFICAS

ABLESON, F & SEN, R.(2011).**Android in Action**.Manning Publications

FOX, V.(2010).**Marketing in the Age of Google**.John Wiley and Sons

HOLZNER s. & FELKER D.(2010**).Android Application Development For Dummies**.For Dummies

PEREIRA C. & SILVA M.(2009).**Android Para Desenvolvedores**.Brasport

SHAWN, E.(2010).Pro Android Media: **Developing Graphics, Music, Video, and Rich Media Apps for Smartphones and Tablets**.Appress

SILVA, V.(2010).**Pro Android Games**.Appress

TAYLOR, R & ZECHNER M.(2011).**Beginning Android Games**.Appres

KHUSHBU, S. & BIMAL, G. (2010).**Analysis Of The Emerging Android Market**.San José State University

ADMIN (2011) March 29th: **Why Create Android Apps? – to Make Money – Android App Article**.

Disponível em: http://www.greatereader.org/?p=21540

Acesso em: Abr 2011

DeLacey, B.(2007).**Google SDK**

Disponível em: http://onlamp.com/onlamp/2007/11/12/google-calling-inside-the-gphone-sdk.html

Acesso em: Maio 2011

jogosparaandroid.com.(2010).**Need For Speed Shift Multilinguage Full para Android**

Disponível em: http://jogosparaandroid.com/need-for-speed-shift-multilanguage-full-para-android/

Acesso em: Jun 2011

MORTON, S. (2010) July 14th: **Google wants us all to be mobile app developers**.

Acesso em: abr 2011

disponível em: http://www.techrepublic.com/blog/smartphones/google-wants-us-all-to-be-mobile-app-developers/1281

PLUSMINUS.(2007).**Android - System Architecture (In Words)**

Disponível em: http://www.anddev.org/open-news-f1/android-system-architecture-in-words-t7.html

Acesso em: Maio 2011

WIKIPEDIA.(2011): **Android**

Disponível em: http://pt.wikipedia.org/wiki/Android

Acesso em: Maio 2011

WIKIPEDIA.(2011): **Open Handset Alliance**

Disponível em: http://pt.wikipedia.org/wiki/Open_Handset_Alliance

Acesso em: Maio 2011