THALISSON CHRISTIANO DE ALMEIDA

ESTÁGIO CURRICULAR I e II

IMPLEMENTAÇÃO E OTIMIZAÇÃO DE JOGOS DIGITAIS

EMPRESA: FUNDAÇÃO SOFTVILLE
SETOR: CÉU GAMES
SUPERVISOR: ADEMIR ALBINO ROSSI
ORIENTADOR: DANIELA GORSKI TREVISAN
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO
CENTRO DE CIÊNCIAS TECNOLÓGIAS - CCT
UNIVERSIDADE DO ESTADO DE SANTA CATARINA – UDESC

JOINVILLE
SANTA CATARINA - BRASIL
JUN/2009

APROVADO EM ...../...../.....

_____
Professora Daniela Gorski Trevisan
Titulação: Doutora em Ciências Aplicadas
Professora Orientadora

_____
Professor Edson Murakami
Titulação Doutor em Engenharia Elétrica

_____
Professor Salvador A. dos Santos
Titulação Mestre em Ciências da Computação

_____
Ademir Albino Rossi
Supervisor da Fundação Softville

Carimbo da Empresa

UNIDADE CONCEDENTE			
Razão Social: Fundação Softville			CGC/MF: 00.724.082/0001-17
Endereço: Rua Otto Boehm, 48			Bairro: Centro
CEP: 89201-700	Cidade: Joinville	UF: SC	Fone: 3422-7077
Supervisor: Ademir Albino Rossi			Cargo: Gerente Executivo

ESTAGIÁRIO			
Nome : Thalisson Christiano de Almeida			Matrícula: 211010618
Endereço: Rua Rezende, 553			Bairro: Bom Retiro
CEP: 89222-330	Cidade: Joinville	UF: SC	Fone: 9974-4553
Curso de : Bacharelado em Ciência da Computação			

Título do Estágio: Implementação e Otimização de Jogos Digitais
Período: 19/01/2009/ a 08/05/2009 Carga horária: 360 Horas

AVALIAÇÃO FINAL DO ESTÁGIO I E II PELO CENTRO DE CIÊNCIAS TECNOLÓGICAS

Representada pelo Professor Orientador: Daniela Gorski Trevisan

CONCEITO FINAL DO ESTÁGIO I E II	**NOTA ETG I (Média do Processo)**	**NOTA ETG II (Média do Processo)**
Excelente (9,1 a 10)		
Muito Bom (8,1 a 9,0)		
Bom (7,1 a 8,0)		
Regular (5,0 a 7,0)		
Reprovado (0,0 a 4,9)		

Rubrica do Professor da Disciplina

Joinville

____/____/_____

Nome do Estagiário : Thalisson Christiano de Almeida

QUADRO I

AVALIAÇÃO NOS ASPECTOS PROFISSIONAIS	Pontos
QUALIDADE DO TRABALHO: Considerando o possível.	
ENGENHOSIDADE: Capacidade de sugerir, projetar, executar modificações ou inovações.	
CONHECIMENTO: Demonstrado no desenvolvimento das atividades programadas.	
CUMPRIMENTO DAS TAREFAS: Considerar o volume de atividades dentro do padrão razoável.	
ESPÍRITO INQUISITIVO: Disposição demonstrada para aprender.	
INICIATIVA: No desenvolvimento das atividades.	
SOMA	

QUADRO II

AVALIAÇÃO DOS ASPECTOS HUMANOS	Pontos
ASSIDUIDADE: Cumprimento do horário e ausência de faltas.	
DISCIPLINA: Observância das normas internas da Empresa.	
SOCIABILIDADE: Facilidade de se integrar com os outros no ambiente de trabalho.	
COOPERAÇÃO: Disposição para cooperar com os demais para atender as atividades.	
SENSO DE RESPONSABILIDADE: Zelo pelo material, equipamentos e bens da empresa.	
SOMA	

PONTUAÇÃO PARA O QUADRO I E II

Sofrível - 1 ponto, Regular - 2 pontos, Bom - 3 pontos, Muito Bom - 4 pontos, Excelente - 5 pontos

AVALIAÇÃO FINAL	Pontos
SOMA do Quadro I multiplicada por 7	
SOMA do Quadro II multiplicada por 3	
SOMA TOTAL	

LIMITES PARA CONCEITUAÇÃO

De 57 a 101 - SOFRÍVEL
De 102 a 147 - REGULAR
De 148 a 194 - BOM
De 195 a 240 - MUITO BOM
De 241 a 285 - EXCELENTE

Nome da Empresa: Fundação Softville

Representada pelo Supervisor: Ademir Albino Rossi

CONCEITO CONFORME SOMA TOTAL	Rubrica do Supervisor da Empresa Local: Data :	**Carimbo da Empresa**

ESTAGIÁRIO

Nome: Thalisson Christiano de Almeida **Matrícula: 211010618**
Endereço (Em Jlle): Rua Rezende, 553 **Bairro: Bom Retiro**
CEP: 89223-350 Cidade: Joinville UF: SC Fone: 3435-3423
Endereço (Local estágio): Rua Otto Boehm, 48 **Bairro: América**
CEP: 89201-700 Cidade: Joinville UF: SC Fone: 3422-7077

Regularmente matriculado no semestre: 7º Curso: Bacharelado em Ciência da Computação
Formatura (prevista) Semestre/Ano: 01/2010

UNIDADE CONCEDENTE

Razão Social: Fundação Softville
CGC/MF: 00.724.082/0001-17
Endereço: Rua Otto Boehm, 48 **Bairro: América**
CEP: 89201-700 Cidade: Joinville UF: SC Fone: 3422-7077
Atividade Principal: Incubadora Tecnológica
Supervisor: Ademir Albino Rossi **Cargo: Gerente Executivo**

DADOS DO ESTÁGIO

Área de atuação: Programação de Jogos Eletrônicos
Departamento de atuação: Céu Games **Fone: 3422-7077**
Ramal: 220
Horário do estágio: 17:30 às 22:00 **Total de horas do Estágio: 360**
Período: 19/01/2009 até 08/05/2009 **Total de horas semanais: 22,5 horas**

Nome do Professor Orientador: Daniela Gorski Trevisan
Departamento: Ciência da Computação

Disciplina(s) simultânea(s) com o estágio
Quantas: 5
Quais: COF (Contabilidade e Finanças)
SMU (Sistemas Multimídia)
TCC-I (Trabalho de Conclusão de Curso I)
TOCI-07 (Desenvolvimento de Aplicações Web)
TOCI-20 (Tópicos em Programação Avançada)

OBJETIVO GERAL

Atuar na programação de jogos eletrônicos para celulares (Java) e para computadores (Flash), adquirindo conhecimento tanto técnico quanto organizacional de uma empresa.

ATIVIDADES	OBJETIVO ESPECÍFICO	HORAS
Programação de Jogos para Celulares	Manutenção e Desenvolvimento dos Jogos Eletrônicos para Celulares	100
Porting dos Jogos para Celulares	Adaptar os jogos para funcionar corretamente nas diversas marcas de celulares existentes no mercado.	100
Programação de Jogos em Flash	Manutenção e Desenvolvimento dos Jogos Eletrônicos em Flash	160

Rubrica do Professor Orientador	Aprovação do Membro do Comitê de Estágio	Rubrica do Coordenador de Estágio	Rubrica do Supervisor da Empresa	
				Carimbo da Empresa
Data:	**Data:**	**Data:** Prof César Malutta	**Data:**	

CRONOGRAMA FÍSICO E REAL

PERÍODO (20 horas) ATIVIDADES		**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
Programação de Jogos para Celulares	**P**	■	■	■	■	■															
	R	■	■	■	■	■															
Porting de Jogos para Celulares	**P**														■	■	■	■	■	■	■
	R														■	■	■	■	■		
Programação de Jogos em Flash	**P**						■	■	■	■	■	■	■	■							
	R						■	■	■	■	■	■	■	■							

Aos meus pais
Liberato José de Almeida
e Nerci de Almeida
pelo amor que me dão
em tudo que eu faço

AGRADECIMENTOS

Muitas pessoas e empresas tornaram-se merecedoras do nosso reconhecimento, pelo muito que colaboraram para a realização deste trabalho, dentre elas destacam-se:

- A minha família, que sempre me deram apoio e amor em tudo que eu faço;
- Daniela Gorski Trevisan por aceitar em ser a minha orientadora;
- Fundação Softville e seu gerente executivo Ademir Rossi pela supervisão;
- Céu Games por ter dado essa oportunidade de ouro e pelas experiências valiosas que recebi nesse período que estamos unidos;
- Aos meus amigos e parceiros Fabiano Naspolini de Oliveira e Santiago Viertel pelo apoio e ajuda nesta fase final de faculdade;
- Aos meus colegas do Taka no Dan, pelo apoio que recebo desde o Ensino Médio;
- Deus, que me guiou de forma inacreditável para que eu pudesse atingir meus verdadeiros objetivos e permitindo que eu conhecesse todas as pessoas aqui mencionadas.

SUMÁRIO

LISTA DE FIGURAS

RESUMO

Os jogos eletrônicos estão ganhando mais espaço no mercado, mas para o seu desenvolvimento exige um conhecimento multidisciplinar. Esse estágio ocorreu na empresa de jogos digitais Céu Games, com o objetivo de desenvolvimento e manutenção de jogos para celulares e jogos em Flash. Esse relatório contém informações sobre a empresa, as descrições das atividades realizadas pelo estagiário e os desafios encontrados, assim como os relacionamentos das disciplinas durante o curso na prática.

Palavras-Chave: jogos digitais, celulares, internet, *porting*, otimização

INTRODUÇÃO

A principal finalidade deste trabalho é descrever as atividades realizadas durante a fase final do jogo Sperm Race, dentre elas a implementação, manutenção e otimização do jogo, além das atividades de desenvolvimento do jogo Quebracabeça, lançado no Portal Diversão no site da Céu Games.

O estágio foi realizado na Fundação Softville no setor da Céu Games, desenvolvedora de jogos eletrônicos culturais. Durante o estágio, foram realizadas as atividades contidas no Plano proposto.

1.1. OBJETIVOS

1.1.1. Geral

Atuar na programação de jogos eletrônicos para celulares (Java) e para computadores (Flash), adquirindo conhecimento tanto técnico quanto organizacional de uma empresa.

1.1.2. Específicos

Os objetivos específicos são:

- Manutenção e desenvolvimento dos jogos eletrônicos para celulares (100 horas);
- Adaptar os jogos para funcionar corretamente nas diversas marcas de celulares existentes no mercado (100 horas);
- Manutenção e desenvolvimento dos jogos eletrônicos em Flash (160 horas).

1.1.3. Justificativa

A importância desse estágio deve-se ao fato do estagiário ter dificuldades em entrar no mercado de trabalho na área de jogos, cujas ofertas são escassas na região. A empresa escolhida para a realização do estágio, Céu Games, criou a oportunidade de ingresso e através de um processo seletivo, foi aceito como estagiário. Muitas experiências foram adquiridas durante o estágio, tanto no conhecimento técnico quanto em questões de relacionamento com pessoas.

Cada atividade citada neste relatório é divida em tarefas, que são distribuídas para os membros da equipe. Essas atividades são: Programação de Jogos para celulares, *Porting* dos jogos para celulares e Programação de jogos em Flash.

A programação de jogos para celulares foi realizada com o objetivo de programar as funcionalidades do jogo Sperm Race e fazer a correção dos erros encontrados no mesmo.

O *porting* dos jogos para celulares tem como objetivo fazer com que o jogo seja executado em diversos aparelhos celulares. Existem diferenças de *hardware* e ambiente de *software* entre um modelo e outro. Portanto é necessário fazer adaptações no jogo de forma que as diferenças entre os modelos não altere as suas funcionalidades.

A programação de jogos em Flash foi realizada devido ao projeto Portal Diversão, um portal de jogos eletrônicos que pode ser acessado na Internet, cujo objetivo é atrair o público para o site e divulgar a marca da Céu Games.

1.2. ORGANIZAÇÃO DO ESTUDO

O estudo está organizado em quatro partes: a primeira contém uma introdução, resumo, os objetivos gerais e específicos, as motivações e a forma como que está organizado o relatório.

A segunda parte contém as informações sobre a empresa contratante, contendo seu histórico, principais clientes e principais produtos.

A terceira parte é a descrição detalhadas das atividades realizadas durante o estágio, assim como os treinamentos realizados, que são divididos em três

principais atividades: Programação de jogos para celulares, *Porting* de jogos para celulares e Programação de jogos em Flash

A quarta parte são as considerações finais que demonstra o benefício do estágio para o acadêmico, assim como dificuldades encontradas e uma conclusão sobre o estágio.

Após a quarta parte, encontra-se um glossário, com as definições de algumas palavras utilizadas durante o relatório, assim como anexos, que complementam o relatório.

2. A EMPRESA

2.1. HISTÓRICO

Figura 2.1 Marca da Fundação Softville (SOFTVILLE,2009)

A Fundação Softville (Figura 2.1) surgiu de uma iniciativa de empresas de Tecnologia da Informação da Região de Joinville em abril de 1993. Tendo como apoio associações de Classe, Entidades de Ensino e o Poder Público, foi criado o Projeto Softville e estabelecido o Núcleo do Programa SOFTEX.

Em agosto de 1995, a Fundação Softville foi criada como entidade de caráter técnico-científico, privada, sem fins lucrativos, com estrutura organizacional para atender seus usuários da Incubadora Tecnológica do Sistema Compartilhado de Serviços.

A nova Softville, em julho de 2001, passou a ser mantida pelas instituições de ensino superior de Joinville: Univille, Udesc e Sociesc, Sindicato das Empresas de P.D. e Informática e Prefeitura de Joinville, com foco no empreendedorismo para a criação de empresas, geração de emprego e renda (SOFVILLE, 2009).

Figura 2.2 Marca da Céu Games (CÉU GAMES, 2009)

A Céu Games (Figura 2.2) é uma empresa residente na Fundação Softville desde 28 de março de 2007. A idéia surgiu na disciplina de empreendedorismo do curso de Bacharelado em Ciência da Computação na UDESC, sendo desenvolvida através de um Plano de Negócios.

Os estudantes analisaram a viabilidade da empresa devido ao mercado promissor dos games, as iniciativas governamentais em fomentar a indústria de entretenimento interativo e a pouca presença de empresas do ramo no Brasil para atender este mercado.

Quanto aos produtos, desenvolveram dois jogos eletrônicos para celulares convencionais, ambos em processo de comercialização. Foi um passo importante para a caracterização da qualidade dos produtos da Céu Games, além de validar que a empresa tem competência no desenvolvimento de games.

A Céu Games participou de eventos empresariais como a Expogestão e outros relacionados a jogos digitais, como o Simpósio Brasileiro de Games e Entretenimento Digital, cujo evento reúne as maiores autoridades de pesquisa no ramo de jogos eletrônicos, além de empresários nacionais e internacionais.

Além de participar, a empresa promoveu eventos de games como mini-cursos para desenvolvedores de games e o Joinville Games, evento que reuniu os principais desenvolvedores de games de Santa Catarina – em parceria com o pólo de games do estado – para discutir assuntos relacionados ao desenvolvimento de jogos eletrônicos em Joinville (CÉU GAMES, 2009).

2.2. PRINCIPAIS PRODUTOS

A Softville oferece instalações físicas na região do Centro equipada com mais de 1.200 m^2, equipada com laboratórios de informáticas e auditórios para eventos. Possui uma equipe capacitada para atender aos usuários da incubadora, além de orientar os empreendimentos nascentes para o ingresso no mercado e apoio a empresas já consolidadas (SOFTVILLE, 2009).

A Céu Games trabalha com jogos culturais, ou seja, jogos que tem como objetivo passar algum tipo de conhecimento. Partindo dos jogos culturais, a Céu Games trabalha em três linhas de produtos atualmente:

- *Edutainment*: Jogos que tem como objetivo ensinar algum tipo de conteúdo de forma lúdica, sendo ótimos materiais complementares para o ensino;
- *Advergames*: São jogos publicitários, que tem como objetivo divulgar empresas e produtos;
- *Business Games*: Jogos que simulam negócios, sendo muito utilizado em treinamento gerencial.

Os jogos da Céu Games são feitos para as plataformas móveis (telefones celulares), utilizando a tecnologia J2ME (Java 2 Micro Edition) e para a Internet, utilizando Flash.

Atualmente, os principais produtos da Céu Games são dois jogos para celulares: Memobot e Sperm Race. O primeiro (Figura 2.3) é um jogo de memorização de sequências de sinais mostradas na tela. O jogador deve repetir a sequência através do teclado, onde quanto mais acertar, mais pontos serão concedidos.

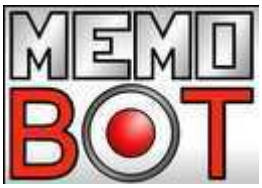


Figura 2.3 Marca do Memobot da Céu Games (CÉU GAMES, 2009)

O segundo jogo é o Sperm Race (Figura 2.4), este um jogo de corrida de espermatozóides, cujo objetivo do jogador é ajudar o seu competidor a alcançar o óvulo, assim vencendo a corrida mais importante da vida de qualquer ser vivo (CÉU GAMES, 2009).

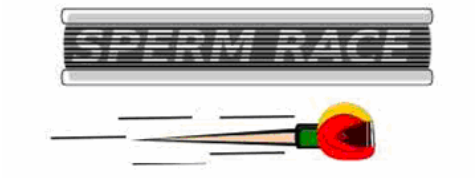


Figura 2.4 Marca do Sperm Race da Céu Games (CÉU GAMES, 2009)

2.3. PRINCIPAIS CLIENTES

Os principais clientes da Fundação Softville são novas empresas compostas tanto por pessoas físicas como jurídicas e empresas já existentes e que desejam se transferir para a incubadora ou seus departamentos. Dentre elas, pode-se citar a Céu Games, Lean Solutions, Quality Test, P4 Soluções entre outras empresas.

Existem dois tipos de público que a Céu Games atende: o público de massa e o público corporativo. O primeiro são os jogos para dispositivos móveis, cujo enfoque são os celulares. Já o público corporativo são empresas que contratam a Céu Games para fazer um jogo digital de acordo com a sua necessidade. A principal cliente da Céu Games atualmente é a Cartão Um, uma empresa com escritórios nas cidades de Salvador e São Paulo.

Figura 2.5 Marca da Empresa Cartão Um (CARTÃO UM, 2009).

2.4. CONSIDERAÇÕES GERAIS

A Fundação Softville é uma incubadora de empresas cujo objetivo é oferecer oportunidades para os empresários, através de oferecimentos de salas, laboratórios de informática, secretárias entre outros recursos, ajudando a diminuir as despesas de empresas iniciantes.

A Céu Games este ano está focada em divulgar seus serviços com o público corporativo. Atualmente está com uma equipe de três pessoas no desenvolvimento de jogos e se encontra em transição de pré-incubada para incubada. O enfoque comercial também compreende novos clientes, mas também a promoção da empresa com iniciativas como o Portal Diversão, onde estarão unidos vários jogos em Flash com intuito de divulgar seus produtos. Além disto, a participação e promoção de

eventos ligados à área de games também será considerada pela empresa neste plano comercial de 2009.

3. DESENVOLVIMENTO

O processo de desenvolvimento da Céu Games utiliza a metodologia SCRUM para basear as atividades que serão dadas aos funcionários de acordo com a prioridade estabelecida, além do Game Design, processo adotado pela maioria das empresas de jogos digitais do mercado. Como auxílio ao projeto, são utilizadas diversas ferramentas, sendo as principais: Mantis, FreeVCS, e GanttProject

Dentro da programação, etapa que o estágio dá maior enfoque devido à atuação do estagiário, utilizou-se da linguagem de programação Java e do Action Script, linguagem própria do Flash.

Neste capítulo serão apresentados os treinamentos realizados internamente na empresa, além do trabalho realizado em algumas etapas do desenvolvimento do jogo Sperm Race, bem como a portabilidade do *game* para os diversos modelos de celulares. Por fim, será demonstrada a atuação em um novo jogo digital, cujo enfoque era o Portal Diversão da empresa: o Quebracabeça Céu Games.

3.1. TREINAMENTOS EM LINGUAGEM DE PROGRAMAÇÃO

Ocorreram dois treinamentos durante o estágio para a capacitação: o primeiro treinamento ocorrido foi a capacitação para utilização da tecnologia J2ME e outra é a capacitação para utilização da tecnologia Flash. Também houve um treinamento que foi dado pelo estagiário sobre a tecnologia Flash para um dos funcionários da própria empresa. No Anexo A, pode ser encontrado a lista de exercícios utilizada durante os treinamentos.

3.1.1 TREINAMENTO EM J2ME

O objetivo desse treinamento é a aprendizagem sobre a tecnologia J2ME (http://java.sun.com/javame/index.jsp) para desenvolvimento de jogos para celulares. Segundo (MATSUMOTO, 2005), Java 2 Micro Edition é uma edição do Java

desenvolvido especialmente para eletrodomésticos e dispositivos embarcados. Os tópicos aprendidos nos treinamentos foram: MIDlets, os executáveis JAR e JAD, as limitações dos dispositivos móveis e o desenvolvimento de uma MIDlet.

O primeiro item são os MIDlets. MIDlets são aplicações desenvolvidas utilizando a tecnologia J2ME utilizado nos telefones celulares. Os MIDlets são controlados pelo sistema operacional do telefone celular de acordo com a necessidade, gerando um ciclo de vida para o MIDlet com três estados:

- *Started* – chamado pelo método startApp(), significa que o aplicativo está em uso, podendo alocar os recursos necessários para a sua execução;
- *Paused* – chamado pelo método pauseApp(), significa que aplicativo está suspenso, liberando recursos não-vitais para o aplicativo e entrando em um modo de espera. A aplicação entra nesse estado quando ocorre um evento externo ao aplicativo que necessita de atenção. Um exemplo é o chamado do telefone;
- *Destroyed* – chamado pelo método destroyApp(), o aplicativo está finalizado e seus recursos serão totalmente liberados.

O ciclo de vida de um MIDlet percorre como mostra a figura 3.1:

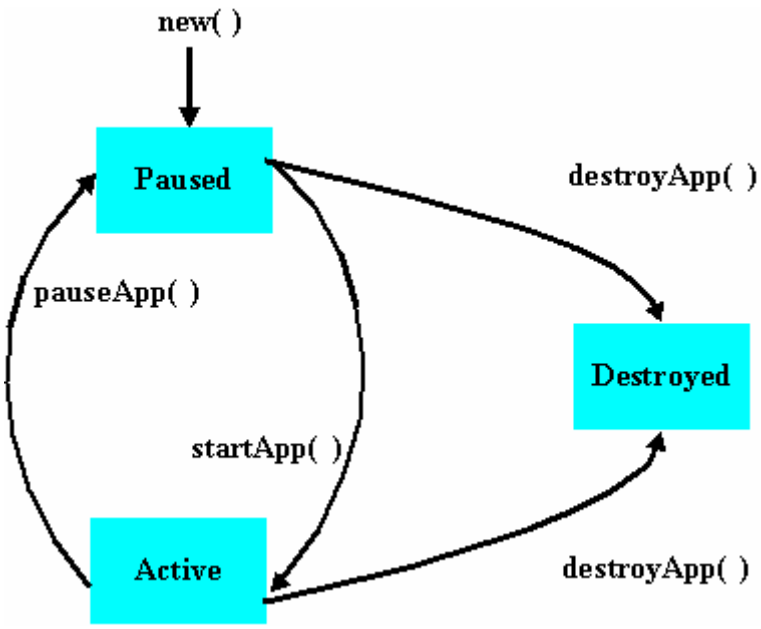


Figura 3.1 Ciclo de vida de um MIDlet (SUN, 2009)

O segundo item a ser aprendido foi saber a diferença entre um arquivo JAR e um arquivo JAD. O Arquivo JAR é o executável do Java propriamente dito, onde estão todos os códigos e recursos como imagens e músicas que estarão no jogo. O arquivo JAD é um arquivo descritor, que contém a descrição de um MIDlet e seus parâmetros, sendo necessário para a inicialização do MIDlet. Entretanto, recentemente os arquivos JAR estão incluindo um arquivo Manifest, que contém as mesmas informações do JAD, dispensando a sua necessidade nos modelos de celulares mais novos.

O terceiro item do treinamento fala sobre as limitações dos celulares. Celulares são dispositivos que possuem pouca memória e pouco poder de processamento. Desta forma, os algoritmos devem ser desenvolvidos para que seja o mais eficiente possível. Também existe a limitação de tamanho da tela, o que exige um projeto tão trabalhado quanto os algoritmos desenvolvidos, para que a visualização do jogo não seja prejudicada.

Por fim, foram realizados exercícios práticos, que envolveram a construção de um MIDlet, manipulação de formulários prontos do J2ME, aplicação gráfica utilizando a classe Canvas, *Sprites* e *Tiled Layers* e leitura e gravação de dados no celular.

A classe Canvas foi desenvolvida para aplicações que precisam manipular a saída gráfica, assim como eventos que possam ocorrer como a tecla ser apertada. Para renderizar uma saída gráfica, é definido um método chamado paint(), que receberá uma classe gráfica (Graphics), que conterá todos os métodos de configuração e renderização suportadas pelo J2ME. Para a leitura de teclas, o Canvas possui os métodos keyPressed(), keyRepeat() e keyReleased(). Os três retornam o valor de uma tecla, mas respectivamente, o primeiro método é chamado quando aperta a tecla, o segundo quando a segura e o terceiro quando ela é solta.

A classe Sprite é usada para se gerar uma animação. Utilizando uma imagem contendo todos os movimentos da animação como a Figura 3.2., a imagem é dividida em partes iguais, com tamanhos definido pelo programador, e geram uma sequência de frames, onde cada um corresponde a uma parte da animação, que pode ser controlada com a classe Sprite.

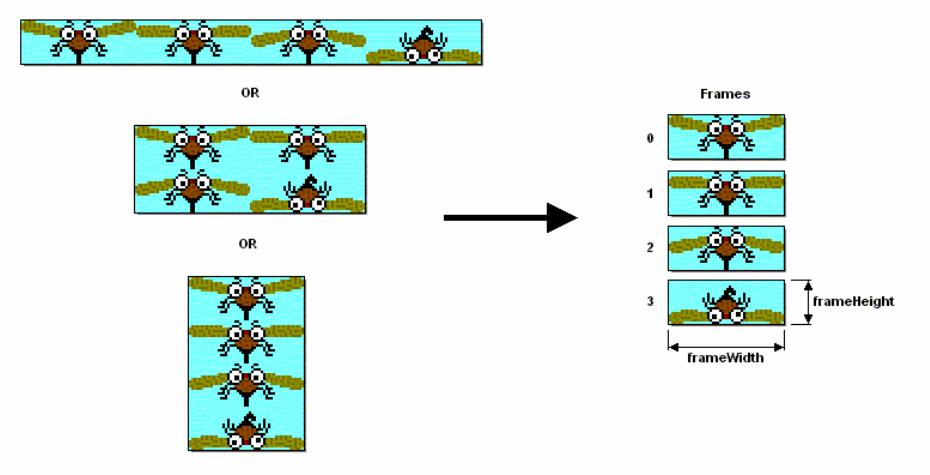


Figura 3.2 Imagens de um Sprite e sua divisão em Frames (SUN, 2009).

De forma semelhante ao Sprite, a TiledLayer também utiliza uma imagem, mas com o objetivo de construir cenários demasiadamente grandes, reaproveitando partes de uma mesma imagem. Por exemplo, a imagem abaixo contém os elementos de um cenário. A imagem é dividida em pedaços iguais chamados Tiles, com dimensões definido pelo programador, e a cada pedaço é atribuído um número (indexação) como mostra a figura 3.3:

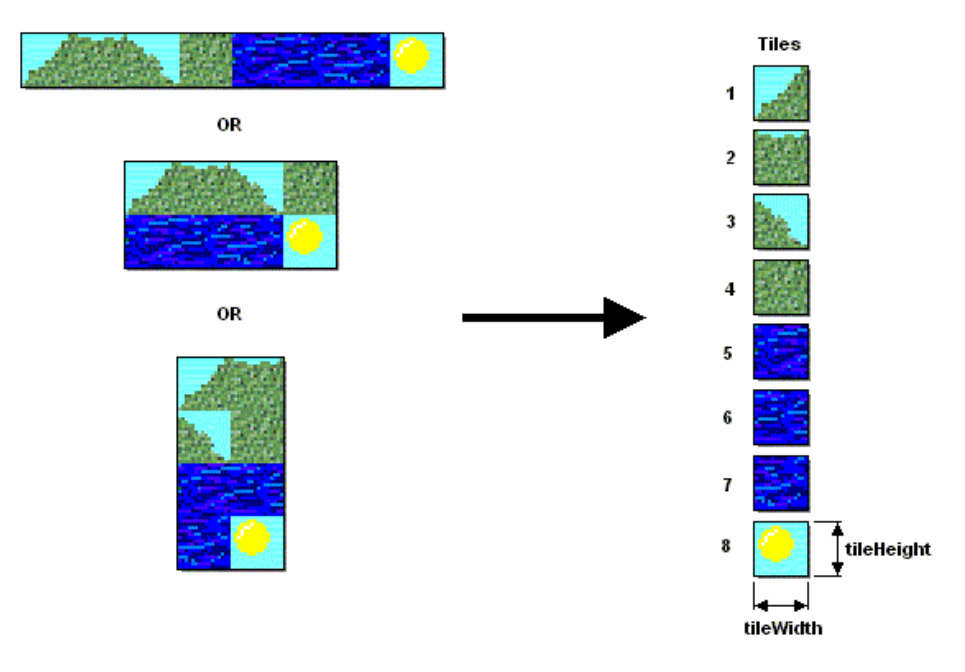


Figura 3.3 Uma imagem de TiledLayer e como ficará dividido.

Com as Tiles já indexadas, utiliza-se uma matriz que conterá o valor das Tiles desejadas, de forma que o cenário seja montado conforme a Figura 3.4:

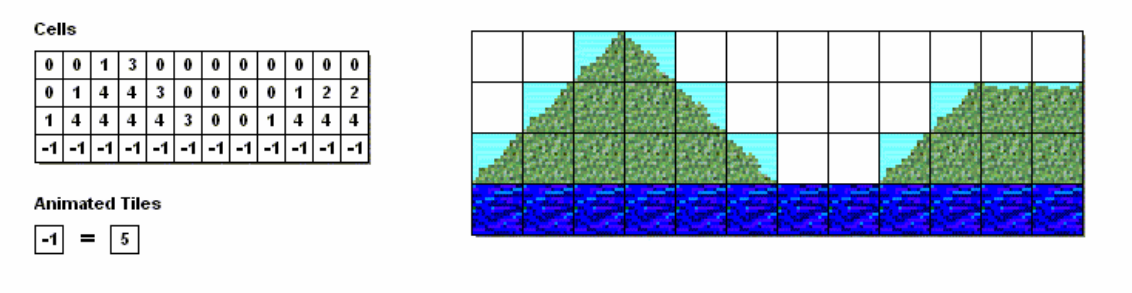


Figura 3.4 Matriz de Tile e como será renderizado (SUN, 2009)

O último item visto no treinamento prático foi a gravação e leitura de dados na memória permanente dos aparelhos celulares. Essa manipulação é feito pela classe *RecordStore*. Para gravar algum dado é necessário primeiramente transformar o dado em um vetor de bytes, e assim passar como parâmetro do método addRecord(). Para ler os dados, utiliza-se o método getRecord(), que retornará os dados em vetores de bytes, sendo a necessária a conversão para o tipo de dado original.

Todos os exercícios realizados, tanto os práticos quanto os teóricos, encontram-se no ANEXO A deste relatório de estágio.

3.1.2 Treinamento em *Flash*

O treinamento em *Flash* ocorreu em dois módulos: ferramentas, cujo objetivo era conhecer o ambiente e seus elementos, como a linha de tempo, os objetos e seus tipos e outras ferramentas; o último, a programação, cujo enfoque era aprender a linguagem *ActionScript*.

A ferramenta utilizada é a *Adobe Flash CS3 Professional* (Figura 3.5), da empresa *Adobe System Incorporated* (ADOBE, 2009), muito utilizada em desenvolvimento de jogos e sites para WEB. A principal funcionalidade do *Flash* é a possibilidade de criar animações que, aliadas a capacidade de programação em *ActionScript*, pode-se adicionar mais interatividade entre a animação e o usuário.

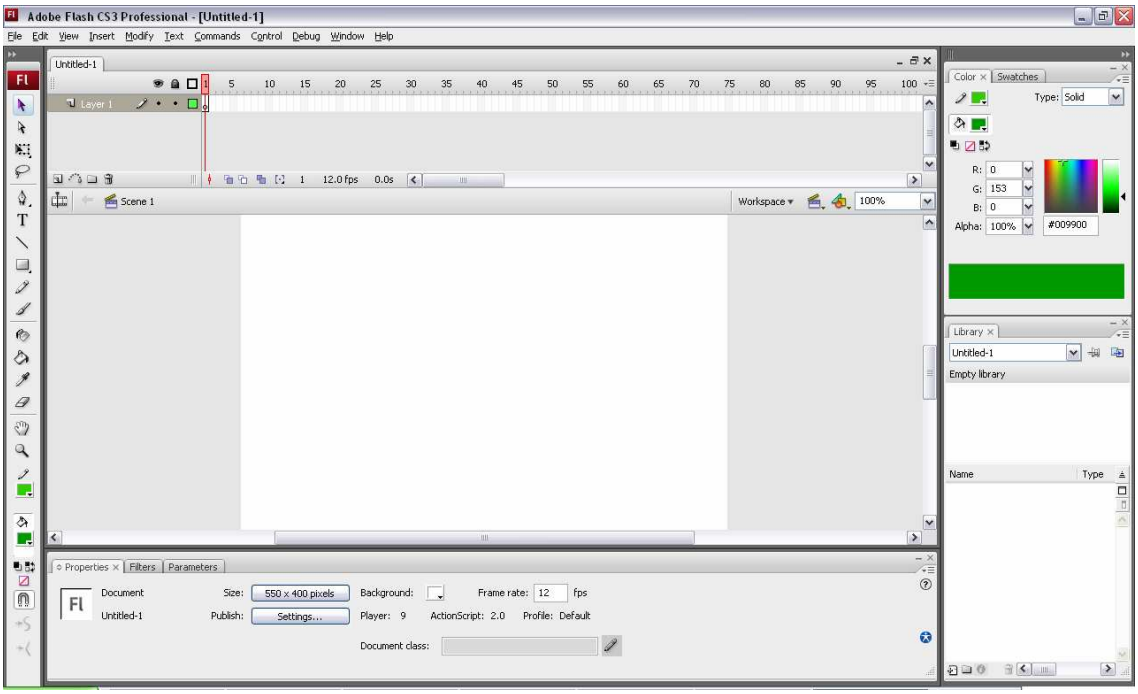


Figura 3.5 Tela Principal do Adobe Flash CS3 Professional

A linha do tempo é a parte fundamental do *Flash*, pois é nela que são feitas as animações. Selecionando um frame da linha do tempo, é possível criar scripts (programas pequenos) que serão executados quando a animação chegar ao determinado frame.

Existe também o elemento chamado *Symbol* (Símbolo) que pode ser *Graphic* (Gráficos), *Movie Clip* (Clipe de Vídeo) e *Button* (botões). O elemento Graphic é uma imagem estática. O elemento *Movie Clip* cria um outro vídeo, com a linha de tempo totalmente independente da linha do tempo original, ou seja, a animação que ocorre dentro do *Movie Clip* não afetará nenhum elemento do vídeo principal. O elemento *Movie Clip* pode também conter outros *Symbols*, criando uma hierarquia de *Symbols*. Os *Buttons* são utilizados para a criação de botões, podendo ser animados ou não.

A segunda parte do treinamento foi a aprendizagem da linguagem de programação utilizada pela ferramenta: o *ActionScript*. Esta é uma linguagem orientada a objeto desenvolvida especificamente para o Flash pela *Macromedia*. Os tópicos vistos foram a declaração de variáveis, declaração de funções e como integrar os scripts nos vídeos.

As variáveis no *ActionScript* não necessitam de declaração. Se os valores já são uma primitiva ou um objeto já criado, basta atribuir um valor a uma variável.

Para se declarar uma função, deve-se criar uma variável e atribuir a função para a variável:

```
funcao = function(){
    [Código da Função]
}
```

O terceiro tópico é a integração dos códigos com o vídeo. Cada elemento da animação é um objeto que pode ser chamado durante o momento. Logo, é recomendado que se nomeiem os objetos que serão manipulados pelos códigos.

A última atividade foi outro treinamento de *Flash* ocorrido na empresa, mas com a responsabilidade de tutor, ou seja, passar as explicações sobre o *Flash*, as atividades de fixação e responder as dúvidas.

Todos os exercícios realizados sobre Flash encontram-se no ANEXO A deste relatório de estágio.

3.2. PROGRAMAÇÃO DE JOGOS PARA CELULARES – SPERM RACE

A etapa final da programação de jogos para celulares é importante, pois irá compor a funcionalidade do jogo digital.

Quatro atividades nesta etapa do estágio foram executadas: a programação de som, onde foi criado um tocador para executar a música de fundo do jogo. A segunda atividade foi tratamento de colisão entre o Exterminador e Espermicida, dois personagens cujo objetivo é servirem como obstáculo ao jogador. A terceira foi o sistema de tratamento de imprevistos no celular, para quando ocorrer influências externas durante o jogo que afetam a jogabilidade. A última etapa foi a fase de testes e correções de erros na programação. As referências bibliográficas nesta seção foram tiradas do *site* da Sun, empresa criadora da plataforma Java e J2ME.

3.2.1 Programação do Som

O som é um elemento que ajuda na imersão do jogador. Logo, uma trilha sonora foi necessária ao *Sperm Race*. Com base nesses motivos, é fundamental implementar um algoritmo para executar as músicas de fundo durante o jogo.

Os arquivos de música utilizados no *Sperm Race* são uma mídia do tipo MIDI (*Musical Instrument Digital Interface*). O objetivo foi criar um gerenciador de MIDI para o jogo, de forma que execute a música quando solicitada pelo programa. Para isso, foi implementada a classe *MusicPlayer*, que lê o arquivo e executa a música de forma paralela ao jogo.

A classe *MusicPlayer* possui dois atributos: o objeto *Player* e *VolumeControl*. O primeiro é responsável por ler e reproduzir o arquivo MIDI. Entretanto, a classe *Player* não é capaz de controlar o volume. Logo, é necessário o segundo objeto: o *VolumeControl*. As funcionalidades das duas classes combinadas, formam o *MusicPlayer*.

Para o desenvolvimento da classe *MusicPlayer*, foi necessário o entendimento do ciclo de vida de um *Player* (Figura 3.6), este através da classe já implementada *Player*. Esta classe possui um ciclo de vida com cinco estados: *Unrealized*, *Realized*, *Prefetched*, *Started* e *Closed*. Quando iniciado, a classe *Player* está no estado *Unrealized*. Nesse modo, o *Player* não tem informação suficiente para iniciar a execução de uma mídia. Quando o *Player* termina de ler a mídia, o mesmo passa para o estado *Realized*. Nesse estado, o *Player* já tem a mídia carregada, mas ainda faltam informações de recursos dependentes do sistema operacional, como os recursos de hardware necessários. Com todos os recursos alocados, o *Player* entra no estado *Prefetched*, onde está pronto, mas não está em execução. Quando entra em execução, vai ao estado *Started*. Se a mídia acaba ou é requisitada para parar a execução, volta ao estado de *Prefetched*. Quando não se faz mais a necessidade do *Player*, o mesmo deve ser destruído para o desalocamento dos recursos, sendo este o estado *Closed*.

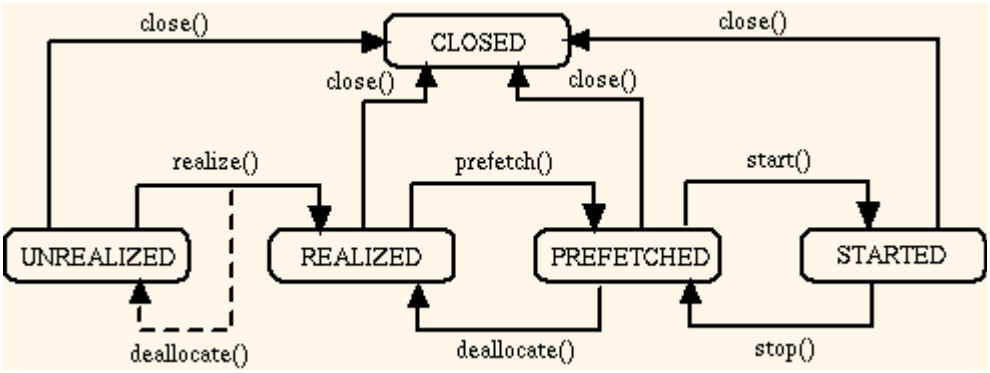


Figura 3.6 Ciclo de vida de um Player (SUN, 2009)

Logo, criou-se a classe *MusicPlayer*, composta por cinco métodos: *musicStart*(), *musicPause*(), *musicContinue*(), *musicVolume*() e *musicGetVolume*(). O método *musicStart*() é o método principal que recebe como parâmetro o local e o nome da mídia e o volume inicial. Caso já existe uma música tocando, ele desalocará o *Player* atual e alocará um *Player* novo com a música atual. Em seguida, o arquivo MIDI será carregado e inicializado no *Player*. O *Player* então é configurado para executar indefinidamente e o volume é atribuído. Por último, é dado o comando ao *Player para* executar a música.

O método *musicPause*() serve para interromper a execução da música, enquanto o método *musicContinue*() é utilizado para retomar a mesma. O método *musicVolume*() faz a mudança do volume da música, enquanto o método *musicGetVolume*(), retorna ao nível de volume, que é necessários em outras partes do aplicativo.

3.2.2 Tratamento de Colisão do Exterminador e Espermicida

O exterminador é um dos vilões do jogo, cuja função é esperar os espermatozóides se aproximarem dele para lançar espermicida, assim deixando os competidores lentos. Essa atividade teve como objetivo fazer com que, caso o jogador fizesse o espermatozóide entrar em contato com algum deles, seus respectivos efeitos ocorram. Para isto, foi necessário o uso de tratamento de colisão.

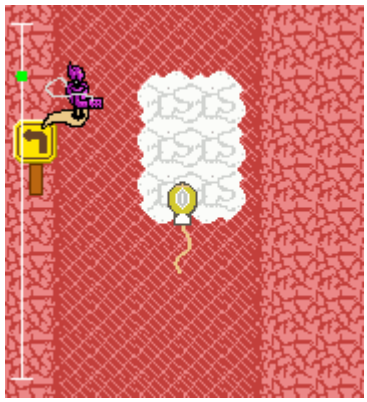


Figura 3.7 Exterminador e Espermicida durante o jogo

Segundo (BOURG, 2002), o processo de tratamento de colisão pode ser divido em duas partes: detecção e resposta de colisão. O primeiro verifica se os dois corpos estão tocando e o último é a aplicação do efeito. A plataforma J2ME já possui um algoritmo de detecção de colisão implementado, fazendo com que o enfoque fosse no tratamento.

No Exterminador, é verificada a posição do competidor em relação ao exterminador (Figura 3.8). Caso a colisão ocorrer na parte frontal do exterminador (Figura 3.8 (a)), o espermatozóide é arrastado até o canto da pista, ficando imobilizado durante esse tempo. Para esse efeito, a velocidade do espermatozóide se modifica, tornando-se igual a velocidade do Exterminador. Caso a colisão ocorre na parte traseira do Exterminador (Figura 3.8 (b)), a velocidade do espermatozóide é reduzido por uma porcentagem.

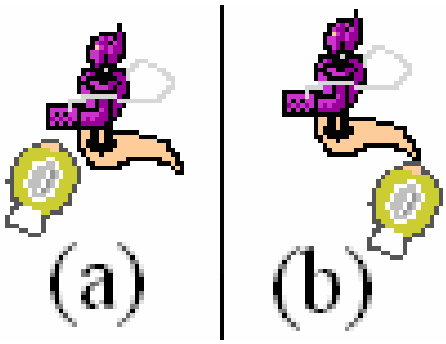


Figura 3.8 Colisão com a parte frontal (a) e com a parte traseira(b)

No caso do espermicida, quando ocorre o contato com um espermatozóide, a velocidade máxima diminui em 70% da velocidade máxima normal e a aceleração máxima também é reduzida em 50%.

3.2.3 Sistema de Tratamento de Imprevistos no Celular

Um aparelho celular está sujeito a imprevistos, como receber uma ligação telefônica, recebimento de uma mensagem ou até mesmo um aviso do aparelho. Esses são eventos externos e não podem ser impedidos pelo aplicativo por ser eventos de alta prioridade do sistema. Para isso é necessário um tratamento que permite suspender o jogo durante esse evento.

Teoricamente, esse tratamento deve ser feito na classe *pauseApp*(), onde é inicializado o MIDlet, pois, quando ocorre um evento externo, o sistema operacional deveria chamar o *pauseApp*(). Embora no emulador, o sistema obteve êxito, nos testes de aparelho celulares demonstrou que, na prática, nenhum dos modelos chamam o método *pauseApp*() quando ocorre eventos externos.

Foi necessária uma mudança de estratégia e, através de pesquisas, percebeu-se que se pode utilizar o método *hideNotify*() da classe *Canvas*, a classe utilizada no desenvolvimento da interface do jogo. O método *hideNotify*() é executado sempre que a tela em questão perde o foco, ou seja, quando o celular exibe outras telas ou, no casos de celulares modelo *flip*, fecharem o celular.

O cálculo de deslocamento dos personagens é feito utilizando como base o tempo. Quanto maior é o tempo que leva entre as atualizações, maior é a distância que percorrem. Quando o jogo perde o foco, não ocorrem atualizações durante o imprevisto. Isto acarreta que o deslocamento do personagem seja demasiadamente grande, durante esse tempo entre as atualizações, fazendo com que o personagem saia da pista. Outro problema que ocorre na falta de tratamento de imprevisto é a música, que não é interrompida quando o jogo perde o foco, fazendo necessária a interrupção explícita.

Para o cálculo de deslocamento, quando o jogo perde o foco, é armazenado o horário em que ocorreu o evento. No momento em que o jogo volta ao foco, é calculado o período de pausa e o mesmo é diminuído no cálculo entre os tempos, fazendo com que o período entre as duas atualizações desconte o tempo interrompido e

o cálculo de deslocamento seja feito corretamente. Por exemplo, a última atualização ocorreu no tempo X quando tocou o telefone. Após 10 minutos, o jogador retoma o jogo. Logo, quando ocorrer a próxima atualização no tempo Y, o tempo percorrido entre a atualização seria (Y+10)-X. A estratégia então é adicionar o tempo suspenso também no valor de X, tendo (Y+10)-(X+10), que é equivalente a Y-X que ocorre nas situações normais.

Para o problema da música, é simplesmente invocado o método *MusicPause*() da classe *MusicPlayer* e quando o jogo voltar ao foco, invocar o método *MusicContinue()*, estes já explorados no tópico 3.2.1.

3.2.4 Testes

Após a conclusão da implementação do *Sperm Race*, foram realizados estes, utilizando celulares reais e emuladores (Figura 3.9), focando principalmente nos emuladores dos celulares da Motorola. Durantes os testes realizados, perceberam alguns erros, entretanto, foi descoberto que esses eles se deviam à limitações do próprio aparelho e não do algoritmo.

Figura 3.9 Teste sendo realizado com emulador

Os dois principais erros ocorriam pelo fato do celular não conseguir carregar as imagens por falta de memória, mas também pelo fato do próprio aparelho

celular limitar o tamanho do arquivo JAR. Ambas as limitações variam de acordo com o modelo do celular.

O resultado dos testes (ANEXO B), é a motivação que leva para a próxima atividade descrita no tópico 3.3.

3.3. PORTING DE JOGOS PARA CELULARES – SPERM RACE

Um das principais dificuldades no desenvolvimento dos celulares é que os celulares não possuem padrões, o que leva a gerar listas de teclas para cada modelos ou séries. Também existem modelos de celulares que restringem o tamanho de arquivo ou a quantidade máxima para alocação, sendo que esses dois últimos é o foco dessa atividade.

Houve três tipos de otimização: código-fonte, imagens e a otimização de som. Essas três atividades serão vistas neste tópico.

3.3.1 Otimização de Código-fonte

O objetivo principal da otimização do Código-fonte é criar uma forma de gerenciar a memória de forma que se torne mais econômica, além de procurar formas equivalentes de lógica para diminuir o processamento do jogo.

Uma análise no código-fonte foi realizada e com pesquisa percebeu-se que existiam quatro classes que ficavam na memória que carregavam as mesmas imagens, significando que aquelas imagens eram carregadas quatros vezes em instâncias diferentes.

Para resolver esse problema, foi criada uma superclasse que continham as mesmas imagens como atributos e as quatro classes herdavam as imagens. O modificador *static* do Java foi aplicado nessas imagens, significando que essas imagens são atributos da classe e não da instância, ou seja, o conteúdo desses atributos é compartilhado entre todas as classes irmãs instanciadas, fazendo com que as classes referencie as imagens já alocadas na memória, ao invés de cada uma alocar a sua própria imagem.

Um problema semelhante ocorreu na classe Competidores, sendo utilizada a mesma solução para se utilizar as mesmas imagens carregadas na memória do celular.

3.3.2 Otimização de Imagem

O objetivo principal da otimização da imagem é a diminuição do tamanho de memória dela. Inicialmente o tamanho do arquivo JAR era de 250 KB. A meta era atingir 100 Kb. Foi decidido que a melhor forma de diminuir o tamanho final do arquivo era diminuir o tamanho das imagens, pois ocupam a maior parte do espaço.

A primeira atividade foi a pesquisa de otimizadores prontos, sendo encontrados dois open-source: Optipng e Advpng. O primeiro procura a melhor combinatória das configurações dos arquivos png, enquanto o segundo programa otimiza as estruturas de cabeçalho do png. Apesar de reduzir o tamanho da maioria das imagens, fazendo com que o arquivo JAR assumisse o tamanho de 154 KB, ainda não foi suficiente. Logo, foi necessário alterar as imagens propriamente ditas, diminuindo a quantidade de cores.

Para isso, criou-se o algoritmo cuja entrada eram as cores desejadas para a imagem, fazendo uma varredura pixel a pixel e verificando a cor do pixel e a cor mais parecida com a das entradas. O resultado foi que o tamanho do JAR ficou com 105 KB. Atualmente estuda-se uma estratégia para atingir a meta de 100 KB.

3.3.3 Otimização de Som

Com a otimização das imagens, o tamanho do JAR ficou em 105 KB, então foi analisado que as músicas estavam com tamanhos variando entre 6 e 7 KB. Os arquivos de músicas possuem a mesma melodia repetida várias vezes. Logo, foi decidido reduzir as músicas.

As músicas foram diminuídas em 3 ciclos, reduzindo um total de 6 KB, mas se descobriu que o arquivo JAR já comprime os arquivos. Portanto, mesmo possuindo 5 KB cada uma, as músicas na forma comprimida não passavam de 600 bytes, significando que a diminuição do tamanho delas não surtiria o efeito desejado.

3.4. PROGRAMAÇÃO DE JOGOS EM FLASH

Foi desenvolvido na Céu Games um jogo de Quebracabeça, cujo objetivo era divulgar os produtos e a marca Céu Games no Portal Diversão. Este portal foi criado para atrair o público ao *site*, sendo composto atualmente por dois jogos: a versão *Flash* do Memobot e Quebracabeça, este o foco deste tópico.

3.4.1 O Jogo QuebraCabeça

Game Design é uma metodologia para projetar um jogo, onde são definidos os elementos e funcionalidades de forma que a equipe entenda e possa criar o jogo (ROLLINGS, 2003).

O layout (Figura 3.10) consiste basicamente em duas áreas: uma à direita e outra à esquerda. A primeira será onde ficarão as peças a serem montadas e a última onde se deve encaixar as peças. Entre as duas áreas, tem três botões: um para pausar o jogo, onde o tempo percorrido parará assim como impossibilita de mover as peças; o segundo que reinicia o jogo e o terceiro que mostra a solução da fase corrente. Abaixo, possui um botão que mostra o *ranking* dos 10 melhores recordes atuais cadastrados no banco de dados e acima está o logo do QuebraCabeça e o relógio que conta o tempo.

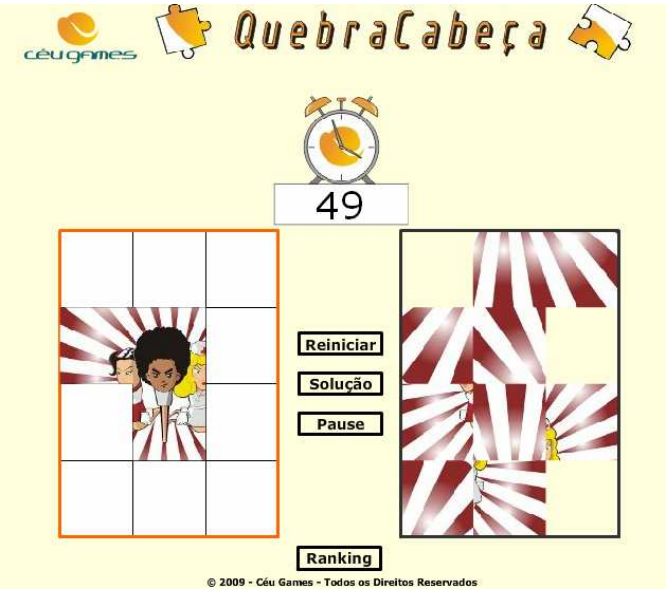


Figura 3.10 Tela do QuebraCabeça

O jogo consiste em 5 fases, onde em cada uma delas vai aumentando o número de peças. Quando o jogador chega ao fim, é feito o cadastro no banco de dados para guardar os recordes obtidos por ele naquela partida. Neste jogo, a pontuação é obtida pelo somatório do tempo obtido nos 5 níveis do *game*.

Mais detalhes respectivos ao *Game Design* encontra-se no ANEXO C deste relatório de estágio.

3.4.2 Programação do Jogo

A grande facilidade é a possibilidade de programar direto no objeto selecionado, diminuindo as chances de desorientação que ocorre em grandes códigos comum na programação tradicional.

A lógica do jogo é a seguinte: para cada peça da fase existe um encaixe correto para a peça. Quando o jogador clicar sobre uma peça e segurar, um evento de clique vai prender a peça ao ponteiro do mouse, utilizando a função *drag*() (Figura 3.11).

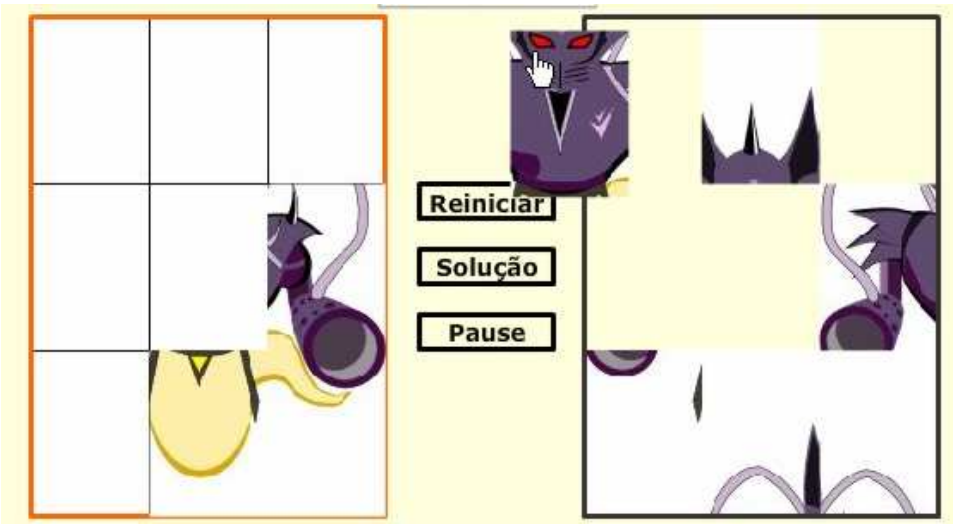


Figura 3.11 Ao clicar na peça é executado a função *drag*()

Ao soltar corretamente a peça sobre um encaixe, será verificado qual encaixe foi colocada a peça. Se a peça está no encaixe correto, o jogador ganhará um ponto (Figura 3.12).

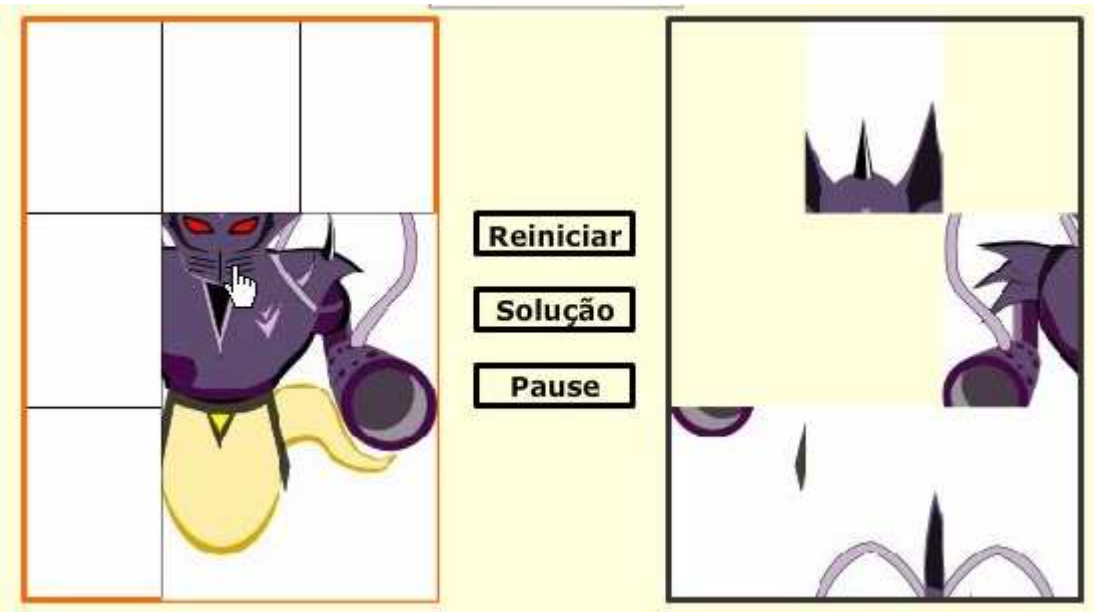


Figura 3.12 Ao soltar a peça na posição correta: pontos+=1

Quando a pontuação for igual ao número de peças, o jogo passará para a próxima fase (Figura 3.13). Caso o jogador retire uma peça previamente colocada no local correto, um ponto será descontado.

Figura 3.13 Se pontos == números de peças, logo significa que passou de fase

A ordenação das peças é feita de maneira aleatória, iniciando na posição 0 e indo até a posição n. Inicialmente cria-se um vetor com o tamanho de quantidade de peças (n) que se deseja, sendo cada posição uma peça e sorteia um número com os

números da posição. Após definir as posições, as peças são colocadas nas posições sorteadas (Figura 3.14).

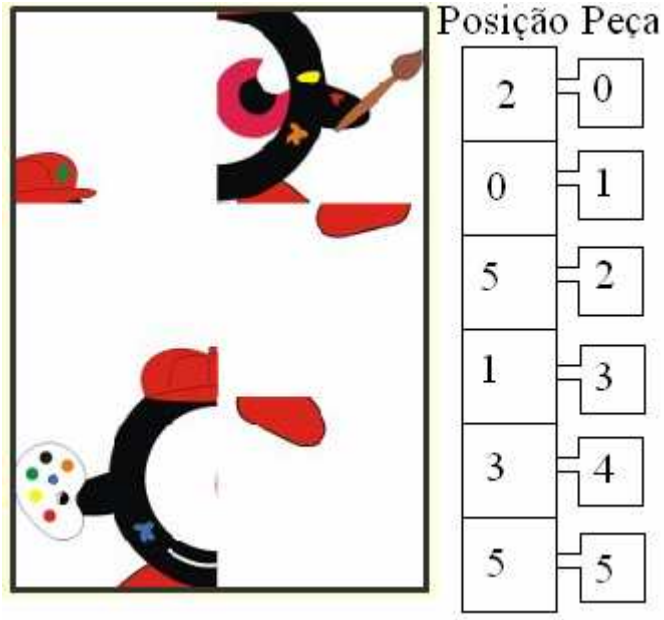


Figura 3.14 Configuração gerada no vetor e a imagem gerada

O relógio foi feito uma função que adiciona o número de segundos, e executados dentro da função *setInterval*(), que recebe dois parâmetros: o primeiro é a função que será executada, seguida pela frequência da execução em milisegundos. No caso do relógio, ele executa a função de adicionar 1 segundo a cada 1000 milisegundos. O relógio é ativado quando o jogo é inicializado e parado por qualquer botão, exceto aquele respectivo a demonstrar a solução do quebracabeça.

Para a inclusão do banco de dados dos recordes é feito uma integração com um servidor PHP. No formulário (Figura 3.15), ao clicar no botão enviar, o aplicativo Flash conecta com uma página PHP e envia os dados pelo método POST. O PHP recebe esses dados e executa a conexão com o banco de dados MySQL, incluindo ou atualizando o mesmo.

Figura 3.15 Formulário de Envio para o banco de dados

Para realizar a leitura, como ocorre na tela dos 10 melhores recordes, o aplicativo Flash conecta com a página PHP responsável pela leitura, executa o comando SQL buscando os 10 melhores recordes no MySQL e envia a resposta para o Flash, sendo tratado e colocado na tela (Figura 3.16). Tanto para receber, quanto para enviar, existe uma latência para o tempo de resposta, o que deve ser tratado, fazendo com que o programa espere até que os dados sejam atualizados ou consultados.

Figura 3.16 Tela de Ranking, com os 10 melhores jogadores.

3.4.3 Implantação do Jogo

O Jogo foi implantado no Portal Diversão da Céu Games, fazendo o *upload* dos arquivos PHP e do executável para Internet do *Flash* (arquivo swf) para o servidor da Céu Games. Um Código HTML incorporado ao *Flash* (Figura 3.17) também é gerado pela própria ferramenta, sendo movido para a página do Portal. Este código permite com que os browsers possam executar o jogo. O jogo pode ser acessado no site da Céu Games, clicando em Diversão e, a seguir, Quebracabeça.

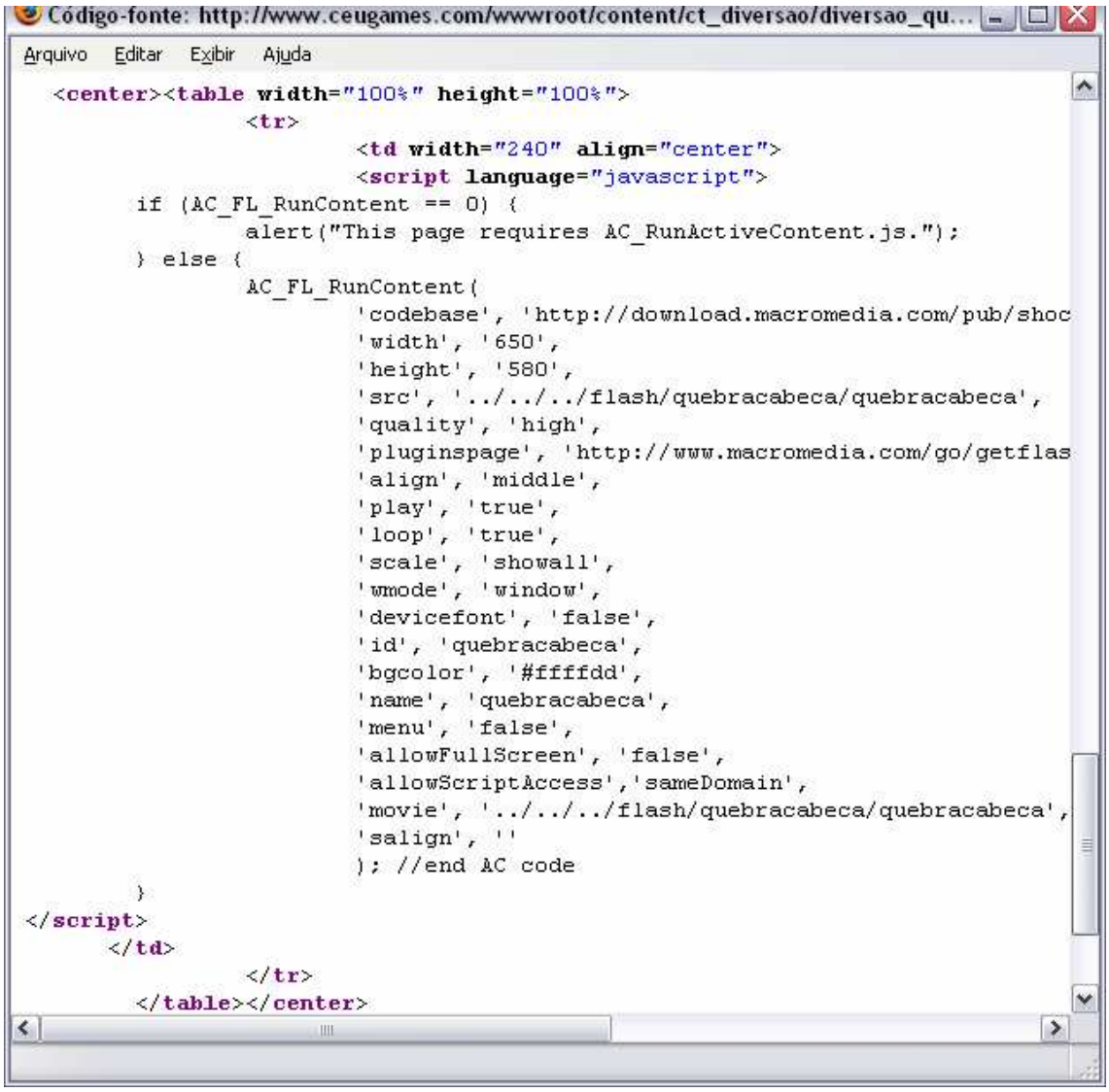


Figura 3.17 Código HTML incorporando o jogo Quebracabeça

3.5. CONSIDERAÇÕES FINAIS

NEste capítulo é descrito as atividades realizadas durante o estágio, tendo três atividades principais:

A primeira atividade é a implementação do jogo, que foi concluída com sucesso e atualmente está pronto. Entretanto devido as limitações dos aparelhos celulares, é necessário realizar adaptações, principalmente na gerência de memória e o tamanho final do jogo.

Dados os resultados dos testes da primeira atividade, foi necessário otimizar o jogo. A primeira otimização foi a de código, que melhorou significativamente as alocações de memória. A segunda otimização foi as de imagem, que reduziu o executável em 105 KB, não atingindo a meta de reduzir para 100 KB. A terceira otimização foi a da música, mas essa estratégia falhou, ao fato de que dentro do executável, as músicas ocupavam menos de 600 bytes. Atualmente estuda-se outras estratégias para diminuir os 5 KB restantes.

A terceira atividade foi a programação do jogo Quebracabeça, que está concluída e pode ser acessada no site da Céu Games. Após a implantação do jogo Quebracabeça no site, percebeu-se que aumentou o número de acessos, o que significa que o jogo está adquirindo resultados em curto prazo.

CONSIDERAÇÕES FINAIS

As etapas desenvolvidas no estágio compreenderam a fase final da implementação do jogo *Sperm Race* e desenvolvimento do jogo em Flash Quebracabeça. As otimizações embora, não tenham obtido êxito total, percebeu-se algumas boas práticas para futuros desenvolvimentos como, por exemplo, as otimizações das imagens serem feitas logo na criação das mesmas e realizar uma análise se os recursos podem ser compartilhados de alguma forma para economia da memória, bem como uma análise de requisitos do projeto mais específica para *porting* de jogos digitais.

Ao longo do estágio foram colocados em prática conhecimentos obtidos em Computação Gráfica com o desenvolvimento da resposta de colisão, Algoritmos e Estruturas de Dados, utilizado para embaralhar as peças no quebracabeça, Linguagem de Programação II, devido a utilização do Java para o desenvolvimento, Engenharia de Software, nas modelagens de como funcionaria a classe *MusicPlayer*, Sistemas Multimídias nas otimizações de imagens, Desenvolvimento de Aplicativos WEB na integração entre Flash, PHP e MySQL e Lógica de Programação na otimização de códigos.

No cronograma, as atividades de Programação de Jogos para Celulares e Programação de Jogos em Flash foram executadas dentro do prazo estimado. Entretanto, o *Porting* de jogos para Celulares ficou pendente devido à falha da estratégia da otimização da música e não pode ser concluído no período de estágio.

Uma nova estratégia estará sendo estudada para que possa atingir a meta de redução do JAR, assim como o desenvolvimento de novos projetos com futuros clientes e o novos projetos para o Portal Diversão da Céu Games.

Não houve dificuldades na implementação do Sperm Race e do QuebraCabeça, devido a documentação detalhada do J2ME fornecidos pela Sun e do Flash fornecido pela Adobe. Entretanto para a otimização do Sperm Race foram encontradas algumas dificuldades devido à falta de conhecimento de como é comprimido os dados no arquivo JAR, assim como as mídias envolvidas.

As atividades da Céu Games contribuíram muito para o desenvolvimento de conhecimento na área da computação, além de aprender e melhorar o trabalho em equipe, este um dos pontos fracos a serem superados pelo estagiário.

ANEXOS

ANEXO A: LISTA DE ATIVIDADES DOS TREINAMENTOS

J2ME TEÓRICO

1. Em que plataformas pode-se aplicar a tecnologia J2ME?

2. O que é um arquivo JAR e JAD? Qual a importância deles?

3. A empresa “Full Games” criou dois jogos para dispositivos móveis, utilizando a tecnologia J2ME. Estes jogos executarão em celulares com MIDP 2.0 e CLDC 1.0. Baseando-se nas especificações técnicas de cada jogo, crie os arquivos JAD para cada um:

a. Nome do jogo: “Pokémon Brilhante”, nome da classe principal: “Pokemon”, nome do arquivo da imagem mostrada no celular: “pkmn.png”, versão 2.8, tem tamanho de arquivo de 66000 bytes e o nome do arquivo JAR é “Pokemon.jar”;

b. Nome do jogo: “Super Mário Bros”, nome da classe principal: “Mario”, nome do arquivo da imagem mostrada no celular: “mario.png”, versão 7,0, tem tamanho de arquivo de 128000 bytes e o nome do arquivo JAR é “Mario.jar”.

4. Mencione as etapas do ciclo de vida de uma MIDlet, explicando os métodos correspondentes a cada etapa e suas respectivas funções.

5. Cite as principais limitações existentes quando se cria aplicações para dispositivos móveis, explicando, através de exemplos, problemas que podem ocorrer, caso o programador não se preocupe com eles.

6. Qual a função do uso de Threads em uma aplicação J2ME?

7. Explique para que servem os métodos da classe Thread abaixo:

a. start():

b. run():

c. sleep(int delay):

J2ME – PRÁTICO

1. Crie um programa HelloWorldJ2ME e faça as etapas necessárias para se criar uma MIDlet.

Etapas:

a. Implementar o programa e salvar em um arquivo com extensão .java;

b. Compilar;

c. Pré-verificar;

d. Empacotar;

e. Testar;

f. Efetuar as correções necessárias;

g. Distribuir.

2. Faça uma aplicação J2ME em que apareça uma tela inicial como a Figura mostrada e, após acionar o comando “Continuar”, apareça um Alerta avisando o usuário que o arquivo “X.exe” será apagado, ou não.

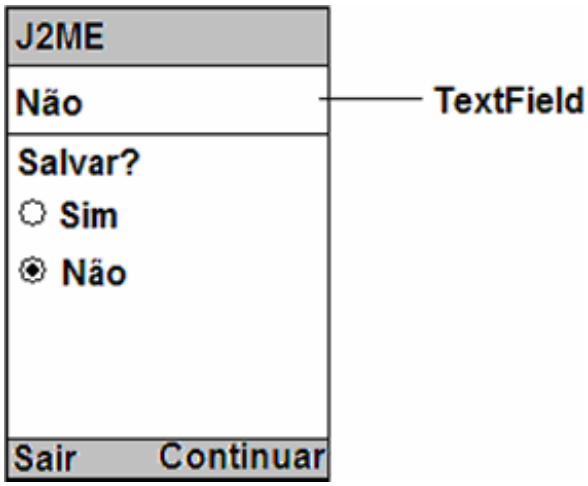
OBS:

a. O TextField não pode ser editado;

b. O TextField deve mostrar a opção selecionada no lista;

c. Após mostrar o Alerta (tempo ou acionamento de comando), o programa deve ser encerrado automaticamente;

d. Ajuda: Tente usar a classe ChoiceGroup.



3. Faça um programa em J2ME que desenha na tela a imagem mostrada na Figura:

OBS:

a. Deverá existir um comando de saída oculto (não aparece na tela).

4. Faça um programa em J2ME que varia a cor de um quadrado existente no centro da tela de acordo com o valor das componentes de cores, mudados pelo usuário.

OBS:

a. A componente Vermelha pode ser variada com os botões 1 e 3 (1: diminui o valor da componente; 3: aumenta o valor);

b. A componente Verde pode ser variada com os botões 4 e 6 (4: diminui o valor da componente; 6: aumenta o valor);

c. A componente Azul pode ser variada com os botões 7 e 9 (7: diminui o valor da componente; 9: aumenta o valor);

d. Deverá existir um comando de saída.

5. Faça uma aplicação gráfica J2ME que faça o tratamento de colisões entre um *Sprite* e um *TiledLayer*.

OBS:

a. O *TiledLayer* não se movimenta;

b. O *Sprite* pode ser movimentado através das setas no teclado;

c. O *Sprite* não pode sair da tela do celular;

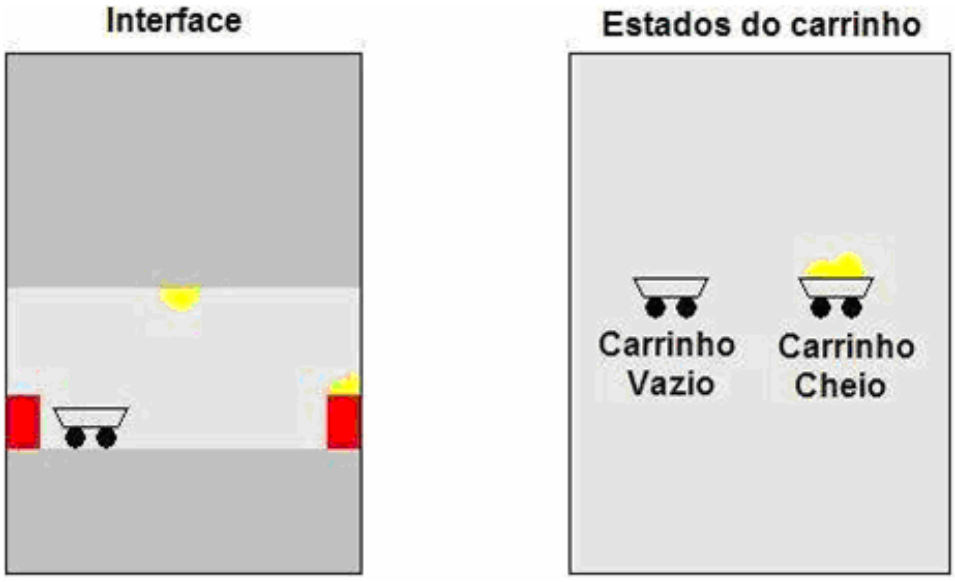
d. O *Sprite* deve ser animado (translação e troca de *frames*) somente ao ser movimentado;

e. Caso ocorra colisão entre o *TiledLayer* o e *Sprite*, não acontecerá movimentação alguma, tanto do *Sprite* como do *TiledLayer*;
f. A cena deve ser construída em *fullscreen*.
g. Em todo o instante deve existir um botão para saída;

6. Faça uma aplicação gráfica J2ME que faça a animação de alguns *Tiles* da camada *TiledLayer*. Em todo o instante deve existir um botão para saída da aplicação.

7. Faça um programa em J2ME que possibilite a leitura e gravação de uma string e um valor inteiro no celular. Caso já exista algo gravado, este deve ser mostrado através de um *StringItem*. Os valores a serem salvos serão passados através de dois *TextField*'s e substituirão os valores gravados anteriormente.

8. Faça um "mini-jogo" em J2ME que possua a interface mostrada na Figura:



a. O carrinho de carregar ouro pode se locomover somente na horizontal através das teclas 4 e 6 do dispositivo;
b. O carrinho fica cheio quando é encostado na caixa vermelha ao lado direito;
c. O carrinho fica vazio quando é encostado na caixa vermelha ao lado esquerdo;
d. O carrinho não pode atravessar nenhuma das caixas (tratamento de colisão);
e. TODA a animação deve ser feita vetorialmente (sem o uso de *Sprites* ou *TiledLayer*);
f. O jogo deve rodar em *fullscreen*;

g. Existirá um botão que possibilite a saída do jogo;

h. Toda a execução principal do jogo deve ser feita através de uma *Thread*.

FLASH

Exercício 1 - Fantasmas

Faça um filme contendo 3 cenas onde:
Na 1ª. cena haja um fantasma. Acionando um botão de “play”, este fantasma deverá mover-se pela tela e transformar-se em um outro objeto qualquer.
Na 2ª cena, deve haver mais 10 fantasmas transmorfos idênticos ao 1º., andando pelo mesmo caminho que o 1º. andava.
Na 3ª cena, além de os fantasmas andarem em fila, eles devem aparecer e desaparecer mudando de cor.
As cenas deverão ser conectadas e “rodadas” nesta seqüência, ao ser pressionado o botão “play”.

Exercício 2 – Animação Logo da Céu Games

Gere uma animação para o site da Céu com a seguinte seqüência de cenas:

Bola da Céu em forma de sol → Cai como uma bola de basquete e faz a cesta → vira a roda de um carro e ele começa a andar → fica a bola sozinha parada na mesma posição do logo da Céu Games → aparece aos poucos o nome ”Céu Games”.

ANEXO B: TABELAS DE TESTES COM OS CELULARES MOTOROLA

Modelos	128x160	240x320	OBS:
A630	Max Allowed 102400b		
A845	Max Allowed 102400b		
C380	Max Allowed 102400b		
C381_C386	Max Allowed 102400b		
C381p	Max Allowed 102400b		
C384_C385_C390_C391	Max Allowed 102400b		
C650_C651	Max Allowed 102400b		
C698p	Max Allowed 102400b		
C975	OK	Null Pointer, mas a tela é menor	OK
C980	OK	Null Pointer, mas a tela é menor	OK
E1000_E1000R	OK	Null Pointer Exception	
E1070	OK	Null Pointer Exception	
E375	Max Allowed 102400b		
E398	Max Allowed 102400b		
E550_V535_V560	Max Allowed 102400b		
E770	OK	Null Pointer, mas a tela é menor	OK
L2_L6_L6i	Null Pointer Exception		
Motokrzr_K1	Null Pointer Exception		
Motokrzr_K3	OK	OK	OK
Motorazr_V3xx	OK	OK	OK
Motorazr_maxx_V6	OK	OK	OK
Motorizr_Z3	Null Pointer Exception		
MotoSLVR_L7i_L7e	OK	OK	OK
MotoSLVR_L9_L72	OK	OK	OK
RAZR_V3	Null Pointer Exception		
RAZR_V3_V3b	Max Allowed 102400b		
RAZR_V3x	OK	Null Pointer Exception	A tela é grande demais para um, mas pequeno demais para outro
ROKR_E1	Null Pointer Exception		
SLVR_L7	Null Pointer		

	Exception		
T725	Max Allowed 102400b		
V1100	OK	Null Pointer Exception	A tela é grande demais para um, mas pequeno demais para outro
V180_V185_V186_V188	Max Allowed 102400b		
V190_V191	Null Pointer Exception		
V195_V196_V197	Null Pointer Exception		
V220_V220i_V226	Max Allowed 102400b		
V230_V235	Null Pointer Exception		
V300_V400_V500	Max Allowed 102400b		
V303p_V400p	Max Allowed 102400b		
V330_V547_V551_V555	Max Allowed 102400b		
V360_V361	Null Pointer Exception		
V365	Null Pointer Exception		
V540_V545_V550	Max Allowed 102400b		
V557_V577p	Null Pointer Exception		
V600	Max Allowed 102400b		
V600i	Max Allowed 102400b		
V620	Max Allowed 102400b		
V635	Max Allowed 102400b		
V80	Max Allowed 102400b		
V975	OK	Null Pointer, mas a tela é menor	OK
V980	OK	Null Pointer, mas a tela é menor	OK
W510	OK	OK	OK

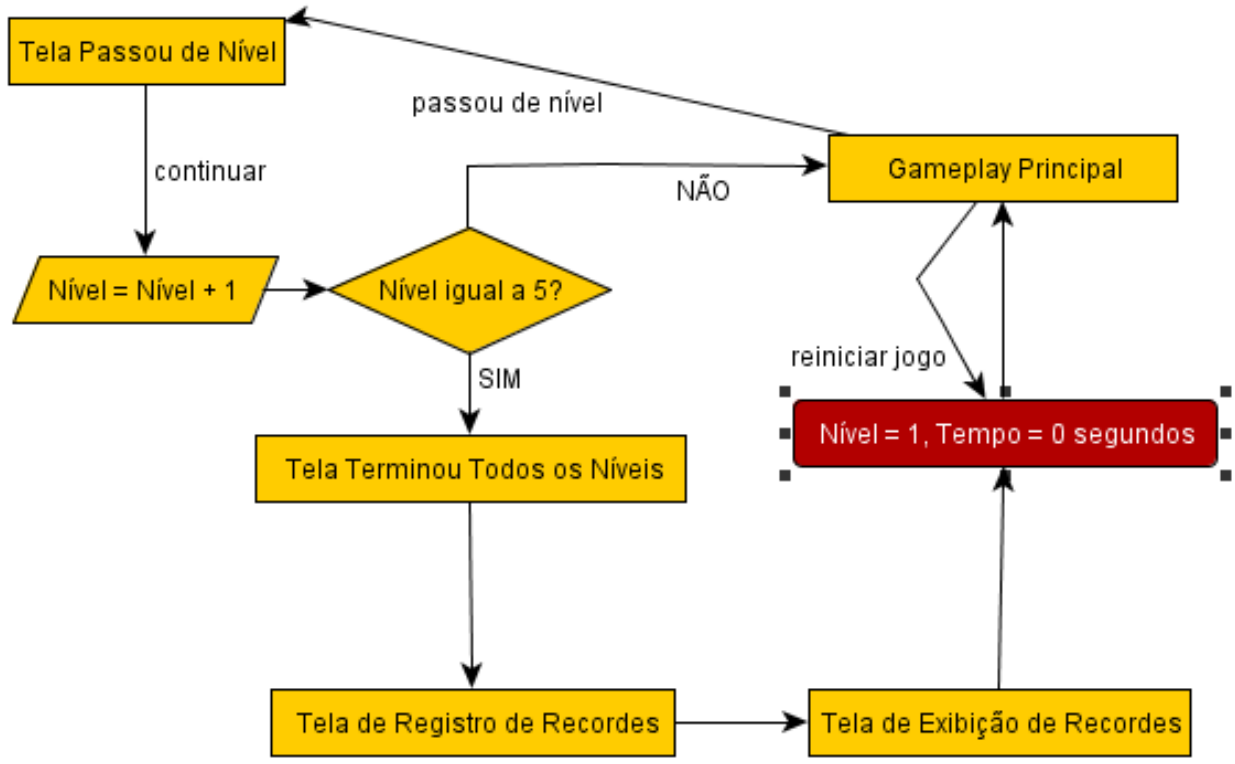
	Quantidade	Proporção
Total de Emuladores de Celulares:	51	100%
Aprovados	11	22%
Null Pointer Exception	16	31%
Max Allowed 102400b	24	47%

ANEXO C: GAME DESIGN DO QUEBRACABEÇA

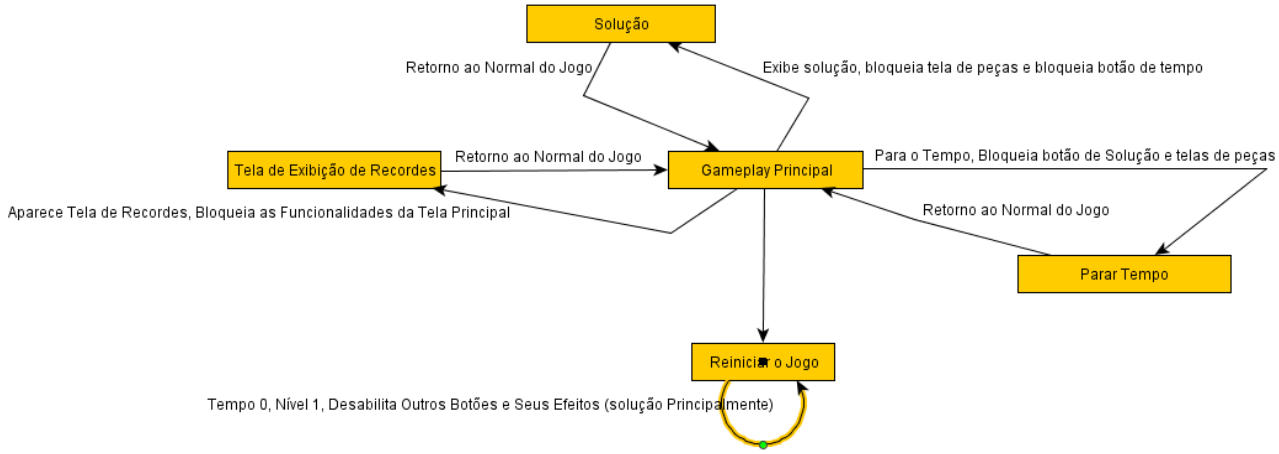
Game Design – Portal Diversão – Quebracabeça

A) *Gameflow* do Jogo

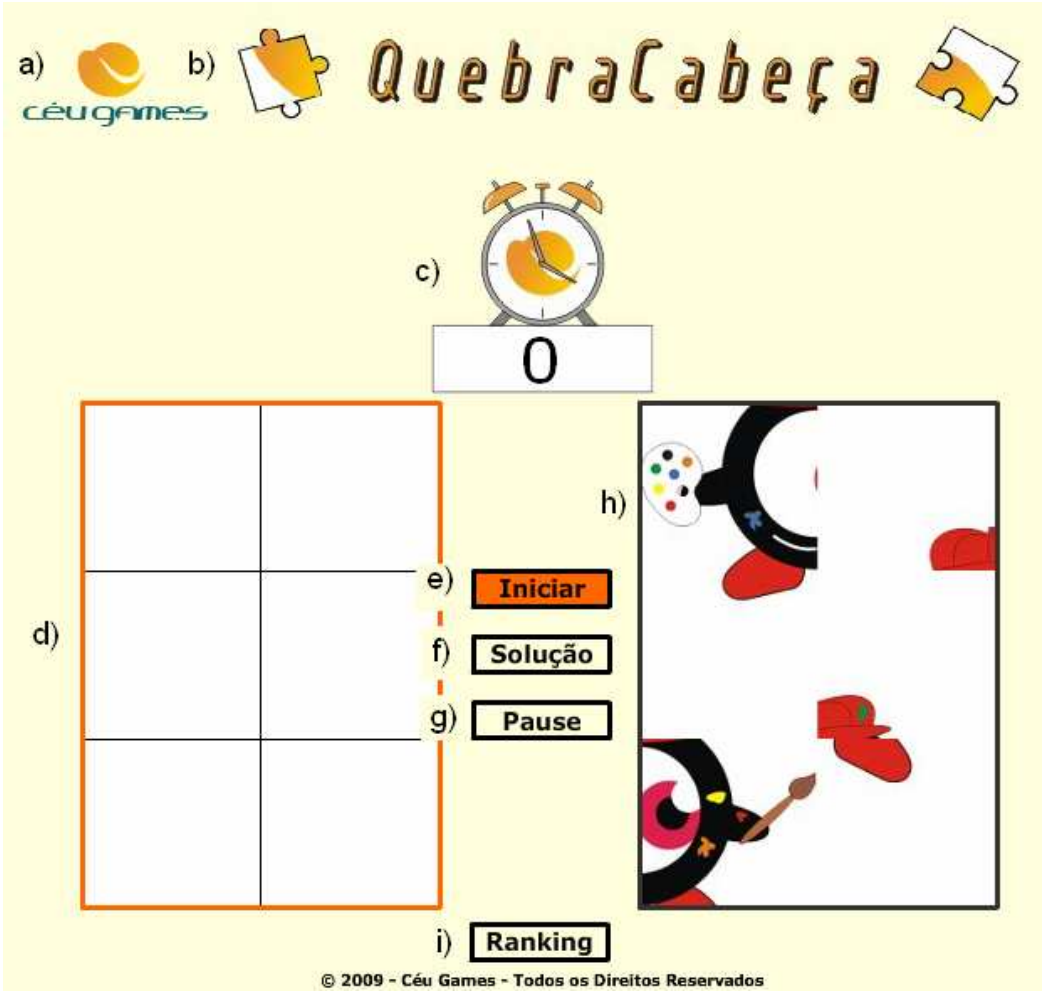
- Principal



- **Interação dos Botões**



B) *Hand Up Display* (Tela Principal)



- **Tela Principal**
  a) Logo da empresa Céu Games
  b) Logo do Jogo Quebracabeça (555 x 110 px)
  c) Controle do Tempo do Jogo (relógio) – deve ser crescente o tempo
  d) Onde as peças serão montadas
  e) Botão Iniciar/Reiniciar:
    - Iniciar: só aparece no início do jogo ou no seu reinício. Responsável por fazer o tempo iniciar no jogo e o jogo em si;
    - Reiniciar: fazer o jogo iniciar novamente (nível 1, tempo 0)

  f) Botão de Mostrar solução (bloquear botão de tempo e telas de montagem).
  g) Botão de Pause do Jogo (parar o tempo)
  h)
    - Onde as peças estarão, sendo apresentadas de forma aleatória e sem que as peças fiquem sobrepostas.
    - Onde será exibida a solução do jogo

  i) Botão de Ranking do Jogo (exibição Top 5)

- **Telas secundárias**

a) Seguem as mesmas do Jogo Memobot em Flash, porém com cor de fundo modificada (igual a principal).

C) Level Design

A dificuldade estão relacionada as imagens escolhidas e a quantidade de secções.

Nível	Quantidade de Secções	Imagem a Ser Utilizada
1	6	Mascote GRUJ
2	9	Exterminador
3	12	Mike, Sammy e Beth
4	16	Banner Sperm Race
5	20	Quebracabeça Céu

D) Registro de Recordes

- Pontuação:

  Soma (i=1 até i=5) t_i , onde i = nível do jogo, t = tempo do jogo.
- Registrar todos, mesmo não sendo Top 5;
- Os melhores serão aqueles com menor tempo realizado nos 5 níveis (tempo total);
- O registro terá o nome da pessoa e o tempo final obtidos nos 5 níveis.

E) Mensagens do Jogo

- Passou de Nível: Parabéns! Você passou de nível! (continuar);
- Fim de Jogo: Fim de Jogo! Tente novamente! (continuar);
- Fechou todos os níveis: Parabéns! Você passou todos os níveis! (continuar).

GLOSSÁRIO

Os conceitos básicos apresentados a seguir estão baseados em diversas fontes bibliográficas.

ACTIONSCRIPT

Linguagem de programação desenvolvida pela Macromedia para ser utilizada no *Flash*.

FLASH

Programa de autoria da Adobe para criação de animação para Internet.

FRAME

Imagem que faz parte de uma animação gráfica.

FREEVCS

Gerenciador de versões, utilizado para evitar que várias pessoas alterem um mesmo arquivo ao mesmo tempo.

GAME DESIGN

Metodologia de desenvolvimento de jogos eletrônicos.

GANTTPROJECT

Programa de Gerenciamento de Projeto.

INCUBAÇÃO

Fase de uma empresa quando está dentro de uma incubadora. É quando a empresa está apta a entrar no mercado, mas ainda não está consolidada.

JAVA

Linguagem de Programação desenvolvida pela SUN.

J2ME

Java 2 Micro Edition. Edição do Java desenvolvido para sistemas embarcados e aparelhos de baixo porte.

JOGABILIDADE

É a forma de interação do jogador durante o jogo digital. Equivalente a usabilidade em *softwares* tradicionais.

MANTIS

Gerenciador de Erros utilizado para o controle de erros, atribuindo responsável para correções e relatórios.

MIDI

Musical Instrument Digital Interface, tipo de mídia de som, muito utilizada na comunicação com instrumentos musicais e equipamentos eletrônicos.

MYSQL

Banco de Dados utilizado frequentemente na Internet.

PHP

Linguagem de Programação para Internet.

PRÉ-INCUBAÇÃO

Fase de uma empresa que está dentro de uma incubadora, onde a empresa se prepara com desenvolvimento de produtos e capacitação para entrar no mercado.

SCRUM

Metodologia ágil de gerenciamento de projetos, fazendo analogia com a formação *Scrum* do *Rugby*.

SPRITE

Imagem ou animação que é inserida num cenário maior; Classe do Java que permite o controle de um *Sprite*.

TILED LAYER

Imagem segmentada que contém elementos para a formação do cenário; Classe do Java utilizado para a montagem de Cenário.

REFERÊNCIAS BIBLIOGRÁFICAS

ADOBE. **Site da Instituição**. Disponível em: http://www.adobe.com. Acesso em 30 de maio de 2009.

BOURG, David M. **Physics for Game Developers.** Sebastopol: O'Reilly, 2002.

CARTÃO UM. **Site da Instituição.** Disponível em: http://www.cartaoum.com.br. Acesso em 31 de maio de 2009.

CÉU GAMES. **Site da Instituição**. Disponível em: http://www.ceugames.com . Acesso em 30 de maio de 2009.

MATSUMOTO, Patricia M. **Sobre J2ME.** Disponível em: www.linux.ime.usp.br/~patty/mac499/tecnica/tecnologias/j2me.html. Acesso em 30 de maio de 2009.

ROLLINGS, Andrew; ADAMS, Ernest. **Andrew Rollings e Ernest Adams on Game Design.** New Riders, 2003.

SOFTVILLE. **Site da Instituição**. Disponível em: http://www.softville.org.br . Acesso em 30 de maio de 2009.

SUN. **Java ME Reference.** Disponível em: http://java.sun.com/javame/reference/index.jsp. Acesso em 29 de maio de 09