Considerações para jogos de ação tipo plataforma com base nas experiências do desenvolvimento do jogo Contra Dengue

Paulo Fernando Pereira[1,2*] Rener Baffa da Silva[1,2] Rodrigo Bareato[1,2]
Thiago Correa Camargo[2,3] Thiago Jabur Bittar[3,4] Elson Longo[4]

IFSP, Brasil[1] Aptor Software, Brasil[2] USP, ICMC, Brasil[3] UFG, Brasil[4] Unesp, IQ, Brasil[5]

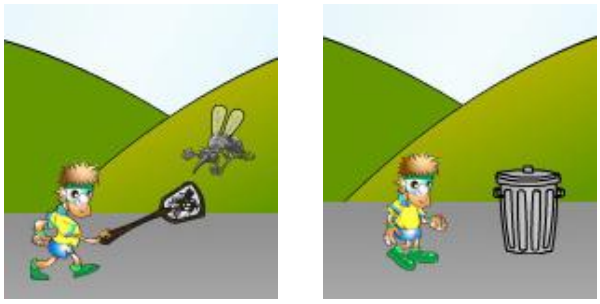

Figure 1: Ilustração do jogo Contra Dengue, jogo de ação para conscientização popular com a temática da dengue.

Abstract

This paper has the main goal presenting the development stages of an action-platform based game, with an easy comprehension and expansion structure, also dealing with the issues found during its development and covering aspects like performance and possible improvements. A game has been developed with a social and educational subject that was able to test its implementation.

Keywords: Dengue, Flash, Games, ActionScript.

Authors' contact:
*pauloaptor@gmail.com

1. Introdução

Um dos maiores sucessos na área de jogos são plataformas do tipo ação. Para esse tipo de plataforma foram criados vários títulos que ainda hoje inspiram desenvolvedores de jogos.

Optou-se, então, pelo desenvolvimento de um jogo desse tipo utilizando o software privado Adobe Flash[1]. O Adobe Flash é um software comumente utilizado para o desenvolvimento de animações embutidas em browsers, celulares, televisores, e outros meios de comunicação, com ênfase para a web.

A linguagem suportada pelo Flash é o ActionScript, cuja versão mais recente, a 3.0, exige a aplicação de conceitos de programação orientada a objetos. Porém, optou-se por utilizar a versão 2.0 porque essa suporta tanto a orientação a objetos quanto desenvolvimento procedural. Para o projeto proposto, utilizou-se a forma procedural para acelerar o processo de criação da plataforma e torná-la simples de ser configurada.

A plataforma sugerida possibilitou a aplicação de um tema e a criação de alguns elementos relacionados a ele que são descritos a seguir.

2. Justificativa do tema

Nos países tropicais, cerca de 2,5 bilhões de indivíduos estão sujeitos aos riscos de uma pandemia de doenças como a dengue, já que a temperatura e umidade propiciam a disseminação dos vetores (Tauil, 2002).

Devido a esse quadro, verificou-se a necessidade da conscientização da população, principalmente jovens e crianças. Constatou-se, então, a possibilidade de usar esse tema em um jogo web, com o intuito de criar uma ferramenta que conscientiza, educa e diverte.

Thompson *et al.* (2010) relatam, nesse sentido, que o uso de informações científicas em jogos é importante e pode oferecer bons resultados educativos.

3. Apresentação do jogo Contra Dengue

O jogo Contra Dengue é baseado em jogos clássicos de ação como o jogo Mario World, no qual o personagem explora o cenário, passando por dificuldades e obstáculos, e seu objetivo é passar pelas fases chegando ao seu final. O uso do jogo é gratuito[2].

No jogo Contra Dengue, para que o jogador passe as fases, é necessário que os focos de dengue espalhados por elas sejam eliminados, evitando, assim, criadouros dos mosquitos transmissores da doença, problema esse comum no Brasil.

No decorrer das fases, o jogador depara-se com seus principais inimigos, que são os mosquitos da dengue. Para combatê-los, o personagem possui dois tipos de armas. A primeira consiste em um mata-moscas, que não possui limite de uso e é uma arma a

[1] http://www.adobe.com/products/flash.html

[2] http://www.ludoeducajogos.com.br/jogo.php?jogo=1

ser utilizada próximo ao inimigo. A segunda consiste em um spray inseticida, que só poderá ser adquirido encontrando o item coletável referente a essa arma. O spray é limitado, mas de alcance maior que o mata-moscas.

Como maior desafio, o jogador tem que encontrar o foco dos mosquitos e realizar as tarefas necessárias para eliminá-lo. Esses focos de dengue funcionam como um criadouro, onde os mosquitos protegem-no, de forma que enquanto ele não seja eliminado, os mosquitos não param de aparecer.

Para que o jogador passe de fase, ele precisa encontrar uma porta, todavia, só poderá passar se todos os focos forem eliminados.

4. Desenvolvimento

O ambiente no Flash é baseado em quadros (*frames*), que constantemente andam de acordo com a freqüência estabelecida. Porém a técnica para o desenvolvimento do jogo é congelar esses quadros, e por meio de código ActionScript 2.0 fazer a física e a interação dos elementos do mesmo.

O *GameLoop*, ou seja, o processo constante do jogo de esperar comandos de entrada, processar o estado atual das variáveis e retornar ao usuário a saída (imagens e sons), é baseado no evento ***onClipEvent(enterFrame)***, nativo do ActionScript 2.0. Esse evento é ativado no tempo referente à execução de um *frame*, portanto, se a animação estiver configurada para rodar em 20 *frames* por segundo, em apenas um segundo esse evento é ativado 20 vezes.

A questão do deslocamento do personagem foi feita com valores que representam sua velocidade, decomposta em velocidade horizontal e vertical.

Uma vez obtidos esses valores, alteramos sua posição relativa acessando as propriedades ***_x*** e ***_y*** do *movieClip* do personagem, dentro da função chamada no *GameLoop*.

Para o desenvolvimento foram aplicados alguns conceitos físicos como gravidade e atrito para criar a jogabilidade.

A simulação da gravidade foi criada como uma força atuante na velocidade vertical do personagem, sempre decrementando a mesma, como se o personagem estivesse sendo atraído para baixo a todo instante. Quando a posição do personagem atinge um valor absoluto mínimo no eixo vertical, que seria o "chão" do cenário, independentemente da velocidade vertical aplicada, ele mantém sua posição vertical fixa.

O evento de pressionar a seta para cima do teclado atribui um valor positivo fixo para a velocidade do personagem no eixo vertical, dando assim o efeito de pulo. A atuação da gravidade simulada cria um comportamento linear sobre a velocidade vertical do personagem, e um comportamento de parábola sobre sua posição vertical.

Quanto à velocidade no eixo horizontal do personagem, o evento de pressionar a seta para direita ou para esquerda do teclado aumenta gradualmente a velocidade no eixo horizontal do personagem. Foi criada então uma simulação de atrito análoga à simulação da gravidade, fazendo com que o personagem parasse aos poucos sua movimentação neste eixo. Quando o personagem não está em contato com o chão, esse coeficiente de atrito é reduzido, porém ainda existente, para fins de melhoria na jogabilidade.

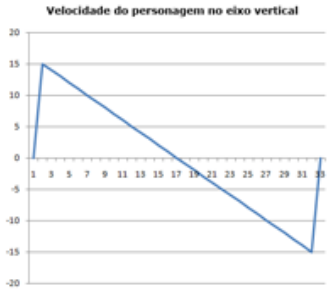

Figura 2: Velocidade do personagem no eixo vertical

Figura 3: Posição do personagem no eixo vertical

Figura 4: Velocidade do personagem no eixo horizontal

Os valores dos coeficientes de atrito e gravidade foram escolhidos empiricamente por meio de testes para se obter uma boa jogabilidade.

Para a implementação da colisão do personagem com os demais objetos foi feito o uso da função ***hitTest()***, nativa do ActionScript 2.0. O objetivo é fazer com que o personagem respeite a área dos demais objetos e junto a isso controlar as variáveis de posição do objeto colidido e posição e velocidade do personagem para que ele não ocupasse o mesmo espaço, por exemplo, de uma parede, e em outras situações, ter o controle se ele está em contato com algum inimigo ou item.

A função ***hitTest()*** utiliza a área retangular mínima que contém o *movieClip*, sendo assim toda colisão é dada pela área delimitadora de cada objeto (*Bounding box*). Esse controle é feito independentemente da forma do *movieClip*.

A verificação de colisão é contínua e tem como objetivo cobrir qualquer objeto que o personagem venha a colidir.

Para os inimigos o sistema de colisão é gerado da mesma forma, porém essa colisão não afeta as variáveis de posição e velocidade do personagem, e o mesmo é válido para os itens coletados na fase.

A interface do Flash apresenta a área onde é exibido o conteúdo da animação. Essa área é fixa, portanto não há como fixar o cenário e caminhar com a área de visualização, como um efeito de movimentação da câmera. Para que se obtivesse o efeito de movimentação, foi feita a proposta de que todos os elementos, exceto o personagem, se movimentem na direção contrária à que ele se movimentaria, enquanto ele fica parado.

Para que o cenário não fique em constante movimento criando uma impressão de artificialidade ao jogo, foram determinadas áreas nos cantos do cenário nas quais a colisão do personagem resulta na movimentação de todos os elementos do cenário no vetor contrário da movimentação do personagem. Essas áreas fazem uma função de câmera que abrange tanto as partes laterais quanto as partes inferiores e superiores.

Todos os elementos do jogo que utilizam algum tipo de evento foram instanciados, ou seja, identificou-se individualmente por um nome e o laço do jogo testa a interatividade entre os elementos usando esse nome da instância. Em exemplo, foi denominado o nome de "boneco" para o objeto que representa o personagem e "bloco1" para um objeto que terá interatividade com o personagem. Dentro do laço do jogo verifica-se se há colisão entre "boneco" e "bloco1", para definir se esses objetos estão em contato, e após o resultado é tratado o posicionamento do personagem para ele respeitar o espaço do objeto. Se adotado um padrão para a nomenclatura, podem ser criados laços de repetição, ***for()***, testando um a um se há colisão.

Quando os inimigos entram em colisão com o personagem, é decrementado o valor de vida do jogo. Como o personagem não é deslocado quando colide com o inimigo, para que o mesmo não ficasse em colisão constante e morresse instantaneamente, é colocado um curto intervalo de tempo em que ele fica imune ao inimigo. Esse intervalo de tempo também é válido quando o contrário acontece, ou seja, o inimigo também fica imune aos ataques do personagem.

Para que o jogador consiga se defender dos inimigos, foram criados dois tipos de armas. Uma arma de alcance curto, do tipo *melee*, e outra de longo alcance, do tipo tiro. A primeira possui uso ilimitado, porém com um baixo dano e a desvantagem de ser necessária a aproximação do personagem aos inimigos. Já a segunda possui dano maior e a vantagem de acertar os inimigos de longe, porém possui uma quantidade de uso limitada e é necessária a coleta de "recargas" para ela.

4.1 Itens coletáveis

Diversos tipos de itens coletáveis podem ser encontrados por todo o cenário. Um deles é a vida coletável.

No momento em que o personagem coleta o item correspondente à vida, a quantidade de vidas do personagem na fase é aumentada, caso essa já não seja a máxima. Do contrário, o item permanece na fase, passível de uma futura coleta. Para se ter esse controle o valor total de vidas do personagem é atribuído na inicialização do jogo.

Outro item que o jogador pode pegar é um *continue* extra, que permite que ele continue da fase que estava caso morra no jogo.

Como mencionado anteriormente, a verificação no caso do personagem morrer acontece na colisão com o inimigo. Se a vida dele chegar a zero, ele "morre". Para sua implementação, o jogo é direcionado para uma nova cena, na qual há uma animação e a quantidade de *continues* é decrementada. Caso não haja mais *continues* o jogo é levado para uma cena de *gameover* e o jogador é direcionado para a tela inicial do jogo.

Outro item coletável são as medalhas. Seu papel é incrementar a quantidade de pontos feitos pelo jogador. Para todos os itens coletáveis, ao haver a colisão do *movieClip* do personagem com o *movieClip* do coletável, é ativada uma animação do *movieClip* correspondente através da função ***play()*** nativa do ActionScript 2.0. Após a animação, o *movieClip* é ocultado com a alteração do parâmetro ***_visible()***.

4.2 Desenvolvimento do jogo Contra Dengue a partir da plataforma

Para o desenvolvimento do jogo, elementos gráficos e sonoros foram criados, além de elementos exclusivos para a proposta do jogo.

Analisando-se o público alvo do jogo, a identidade juvenil do personagem foi definida para atrair mais ainda o interesse dos jovens a utilizarem o jogo, pois o som e aparência possuem uma importância fundamental na interação usuário-computador.

Visando prender a atenção e aumentar a motivação, foram criadas diversas fases, cada fase com um cenário atrativo e diferente. Uma característica que facilitou a implantação dessas fases é que o Flash trabalha com cenas, podendo, portanto, ser implementado cada novo cenário independente dos outros, porém, compartilhando os mesmos códigos do laço de jogo.

4.3 Problemas de desempenho

Durante o desenvolvimento, o jogo apresentou um baixo desempenho mesmo com uma baixa quantidade de elementos no cenário.

A configuração da máquina de teste utilizada foi um computador Intel® *Core™2 Duo P9600* (6M Cache, 2.66 GHz, 1066 MHz FSB), com 4 GB de memória RAM DDR2, Placa de vídeo *NVIDIA® Quadro* NVS *160M*, sistema operacional Windows® 7 Professional 64 bits.

O baixo desempenho era solucionado ajustando a configuração da qualidade gráfica do Flash Player para

média, o que indica que a maior parte do problema era refazer o cálculo dos vetores dos elementos gráficos.

A alteração dessa configuração, entretanto, causa uma grande perda da qualidade estética do jogo, sendo considerada inviável.

4.4 Uso de elementos vetor *versus* bitmap

Como proposta de solução para o problema de desempenho apresentado, fez-se a substituição dos elementos gráficos do jogo para bitmaps. Para entender o motivo por trás dessa decisão é necessário o entendimento da diferença entre os elementos bitmap com elementos em vetor.

Um vetor é representado a partir de fórmulas matemáticas que compõem suas curvas. Por exemplo, é possível guardar a informação "*círculo de raio 5 na posição 0 em x, 0 em y*" e o computador se encarregará de exibir isso na tela.

Em contrapartida, um bitmap é o conjunto de informações de cada pixel que compõe a imagem final. A informação já está processada.

Para o jogo, a principal vantagem do uso de bitmap é não necessitar refazer o processamento, portanto ele apresenta melhoras no desempenho, sem apresentar perda visual independentemente da configuração de qualidade colocada no Flash Player.

4.5 Seccionamento de fases

Uma desvantagem do uso de bitmaps sobre o uso de vetores é uma maior quantidade de espaço ocupada. A partir da criação de novas fases, a utilização de bitmaps para animação dos *movieClips* deixou o jogo maior, fazendo com que seja necessário carregar um arquivo maior para a memória do computador.

Já que logo no início o jogo se carregava um arquivo com informações de todas as fases, o carregamento tornou-se mais demorado, obrigando o jogador a esperar um tempo relativamente grande.

Como proposta para a solução desse problema, fez-se um carregamento de casa fase de forma independente, e uma cena criada com a função de carregar os arquivos da fase seguinte para a memória. Portanto, quando o jogo é aberto, apenas os dados necessários para a primeira fase são carregados, enquanto que, ao passar da primeira fase para as demais, os dados necessários para a fase seguinte são carregados.

5. Resultados

A partir de uma plataforma genérica, foi criado o jogo educativo Contra Dengue.

Porém, ao se criar os elementos gráficos do jogo, houve uma perda de desempenho citada em 4.3. Ao se alterar os elementos para bitmap, obteve-se uma grande melhora no desempenho.

Algumas medições de *benchmark* foram realizadas, utilizando um cenário padrão de teste, e se obteve o seguinte resultado:

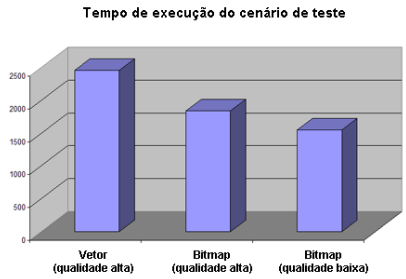

Figura 4: Tempo de execução do cenário de teste

No jogo, ao se realizar testes com beta-players, conferiu-se também que a dinâmica do jogo com a simulação de gravidade e atrito criou um ambiente confortável para o jogador.

6. Conclusão

O uso da ferramenta Adobe Flash para a criação de jogos do tipo plataforma de ação se apresenta bem versátil e fácil de ser configurada, tanto na estrutura de programação com ActionScript quanto com os elementos gráficos, que podem ser divididos em camadas.

Uma desvantagem do uso da ferramenta para o desenvolvimento de jogos é que pelo Flash trabalhar nativamente com elementos em vetor, os jogos podem apresentar desempenho abaixo do esperado mesmo com elementos gráficos relativamente simples.

Uma maneira de contornar este problema é a substituição dos elementos em vetor por elementos em bitmap, que também são suportados pelo Adobe Flash. Um problema acarretado por essa substituição é o aumento do espaço ocupado pelo jogo, que não é interessante quando se trabalha com jogos para web, mas que podem ser solucionados com a secção dos arquivos a serem baixados, conforme o progresso do jogador.

Referências

MATSUI, A. A., VANDERLINDE, F. G., SCALET, G. S., TAKASE, R. D., 2009. Uso de jogos educacionais na conscientização da preservação do meio ambiente. *Revista Ciências do Ambiente On-Line Julho, 2009 Volume 5, Número 1*.

TAROUCO, L. M. R., ROLAND, L. C., FABRE, M. C. J. M., KONRATH, M. L. P., 2004. *Nova Tecnologias na Educação, Rio Grande do Sul, v. 2, n. 1, Março 2004. Disponível em http://www.cinted.ufrgs.br/ciclo3/af/30-jogoseducacionais.pdf. Acesso em: 05 agosto 2011.*

TAUIL, P. L., 2002. Critical aspects of dengue control in Brazil. *Cadernos de Saúde Pública*.

THOMPSON D., BARANOWSKI T, BUDAY R., BARANOWSKI J., VICTORIA, JAGO R., GRIFFITH M. J. 2010. Serious Video Games for Health: How Behavioral Science Guided the Development of a Serious Video Game. *Simul. Gaming 41, 4 (August 2010)*, 587-606.