Bibliotecas e Componentes para Interatividade de Jogos em Flash

Leonardo Serra Faria

Orientador: Bruno Feijó

Dept. de Informática, PUC-Rio

PIBIC
Programa Institucional de Bolsas de Iniciação Científica

PUC-Rio
CNPq

2007

Bibliotecas e componentes para interatividade de jogos em Flash

Aluno: Leonardo Serra Faria
Orientador: Bruno Feijó

Introdução

A plataforma Flash vem sendo muito utilizada atualmente em aplicações multimídia, web e jogos. Pode ser executada em diversas plataformas e possui uma interface boa para o desenvolvimento. Ademais, esta plataforma é excelente para o design.

Não há ainda no mercado uma ferramenta para desenvolvimento de jogos 2D em Flash que facilmente programe efeitos especiais, a dinâmica dos objetos e o comportamento inteligente dos personagens. Apesar de já existirem vários jogos desenvolvidos em Flash, podemos observar uma grande dificuldade por parte dos desenvolvedores. É comum observar códigos repetidos e algoritmos não eficazes.

O presente projeto pretende dar andamento às investigações e implementações já realizadas no Laboratório VisionLab/PUC-Rio em motores de Jogos, com especial foco em jogos educativos para web e TV digital interativa. A PUC-Rio passou a ser considerada líder na pesquisa e desenvolvimento de jogos digitais. O presente projeto fortalece esta nova área.

Objetivos

Este projeto visa desenvolver uma série de bibliotecas de funções e componentes encapsulados, gerando um motor de jogo 2D em Flash eficiente e amigável. A meta é maximizar a produção dos designers (que geralmente têm pouco conhecimento de programação em Flash) no desenvolvimento de jogos educativos para web e televisão digital interativa. Faz parte dos objetivos estender e consolidar trabalhos anteriores nesta área conduzidos pelo Laboratório VisionLab/IGames da PUC-Rio, melhorando as questões de modularidade e de reuso. Os principais pontos a serem atendidos no final do projeto são os seguintes: melhoria no controle de animação de personagens, percurso em mundo isométrico, construção de fases e carregamento de objetos externos dinamicamente.

Metodologia e Desenvolvimentos

A metodologia baseia-se em desenvolvimento modular, construindo-se primeiramente as bibliotecas de função e posteriormente os componentes. O desenvolvimento de cada módulo deve acontecer através de planejamento, desenvolvimento, testes e documentação. Numa primeira fase o pesquisador deve se familiarizar com as técnicas de engenharia de software e com as arquiteturas de motores de jogos 2D. Na segunda fase, o pesquisador deve estudar as técnicas mais adequadas para os objetivos propostos. Estas duas fases são acompanhadas de implementações e validações.

Os desenvolvimentos iniciais do projeto são descritos a seguir.

Estudo de técnicas de engenharia de software - busca-se definir um modelo de classes e a definição de suas propriedades e métodos que organizem os elementos necessários para a produção de jogos em Flash.

Estudo de arquiteturas de motor de jogo 2D – um primeiro resultado deste estudo é que as seguintes características de um motor de jogos 2D já se encontram na estrutura do Flash: carregamento de imagens, "render" de gráficos vetoriais e a habilidade de reproduzir sons e

vídeos. Uma funcionalidade que não se encontra tão fácil é o disparo de eventos por tempo de jogo, por colisão, por alteração em uma variável e por estado do jogo.

Definição dos módulos e funções - Foram esboçados os principais focos e metas para cada um dos módulos e especificadas as principais funções.

Biblioteca de animação de personagens – devem-se construir duas bibliotecas, uma de animação em geral, que controla o fluxo, a velocidade e o sequenciamento das animações, usando como base os Movie Clips do Flash. Depois deve ser construída uma biblioteca especializada para animação de um personagem de plataforma 2D.

Biblioteca de máquina de estado – primeiramente deve ser construída uma biblioteca de grafos para uma máquina de estados genérica. Depois, esta máquina deve ser aplicada na construção da biblioteca especializada para animação de personagem. Esta biblioteca mostrou dificuldades maiores dos que as inicialmente esperadas. Os resultados ainda não estão aceitáveis.

A cena de teste e a interface para definição de parâmetros básicos seguem os exemplos das Figs. 1 e 2.

Fig. 1 Cena de testes

A cena da Fig. 1 utiliza funções previamente criadas e encapsuladas por outros pesquisadores, demonstrando assim a importância da biblioteca de funções para maior facilidade na criação de jogos tornando dispensável, para futuros usuários, o conhecimento da linguagem ActionScript.

Na Fig. 2 está ilustrado o conceito de interação para definição de movimentos complexos. A idéia é facilitar a elaboração do jogo, fazendo com o que o usuário preencha

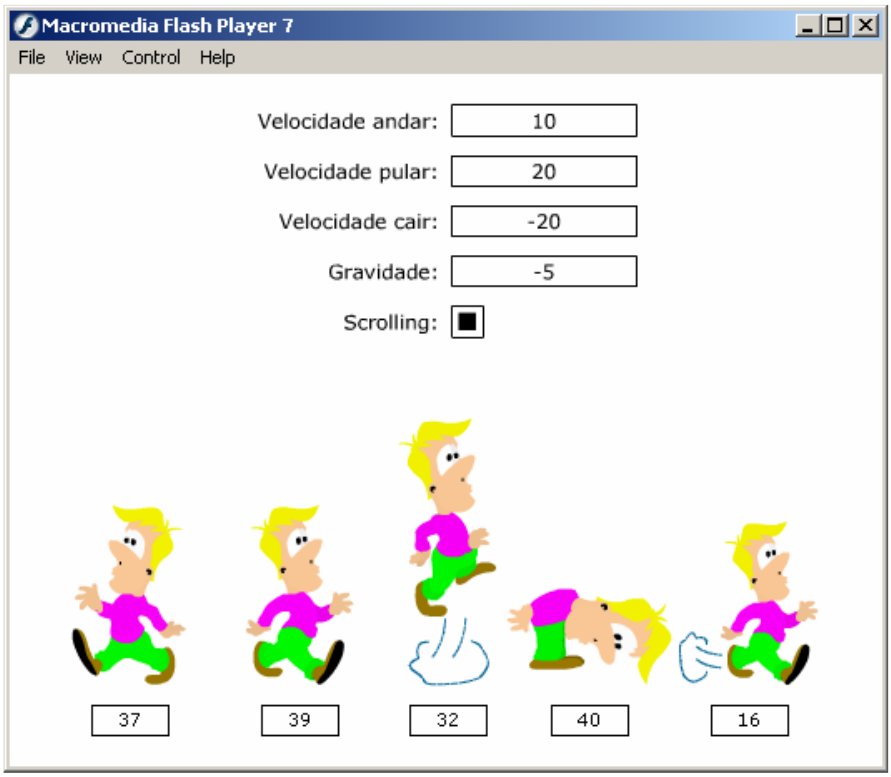

Fig.2 Interface para definição de parâmetros

nos campos determinados os valores que ele deseja para o seu jogo em relação à gravidade e à gravidade, entre outros parâmetros. Os seguintes parâmetros são facilmente definidos para a definição de caminhar: tempo de cada componente de movimento, velocidade horizontal do deslocamento, velocidade vertical para cima, velocidade vertical para baixo e gravidade.

Alguns exemplos de códigos previamente criados para a biblioteca de funções e aplicados no jogo da Fig. 1 estão listados a seguir:

Execução da Música

```
#initclip 1
function Musica() {
 this.Inicializar();
}
Musica.prototype = new Object();
Musica.prototype.Inicializar = function() {
 this._visible = false;
 this.Som = new Sound();
 this.Som.attachSound(this.varmusica);
 this.Som.start();
 this.Som.onSoundComplete = function() {
       this.start();
 }
}
Object.registerClass("Musica", Musica);
#endinitclip
```

Tempo no Jogo

```
#initclip 1
function Tempo() {
 this.Inicializar();
}
Tempo.prototype = new Object();
Tempo.prototype.Inicializar = function() {
 this.MovieBase = eval(this._targetInstanceName);
 if (!(this.MovieBase instanceof MovieClip)) {
       trace("Erro: " + this);
       this.enabled = false;
       return;
 }
 this._visible = false;
 this.ssomtempo = new Sound();
 this.ssomtempo.attachSound(this.somtempo);
 this.atual = 0;
 this.incremento = this.intervalo/this.tempofinal;
 _root.parametros.controletempo = setInterval(this.Funcao,
             this.intervalo, this);
 this.MovieBase.stop();
}
Tempo.prototype.Funcao = function(obj) {
 obj.atual += obj.incremento;
 obj.MovieBase.gotoAndStop(Math.floor(obj.MovieBase._totalframes*obj.a
tual));
 if (obj.atual >= 1) {
       stopAllSounds();
       obj.ssomtempo.start();
       _root.gotoAndStop(obj.goto);
       clearInterval(_root.parametros.controletempo);
 }
}
Object.registerClass("Tempo", Tempo);
#endinitclip
```

Finalização

```
#initclip 1
function Fim() {
 this.Inicializar();
}
Fim.prototype = new Object();
Fim.prototype.Inicializar = function() {
 this._visible = false;
 if (_root.parametros == undefined) _root.parametros = new Object();
 _root.parametros.Fim = eval(this._targetInstanceName);
 _root.parametros.somfim = this.varsomfim;
 _root.parametros.gotofim = this.vargoto;
}
Object.registerClass("Fim", Fim);
#endinitclip
```

Além desses códigos, muitos outros códigos previamente criados em ActionScript foram utilizados no exemplo de jogo da Fig.1.

Outro exemplo de jogo criado com funções da biblioteca está ilustrado nas Fig. 3 e Fig. 4. Este jogo tem duas fases. Algumas funções sãoiguais às da Fig. 1, como tempo e finalização.

Fig. 3 Cena teste fase 1

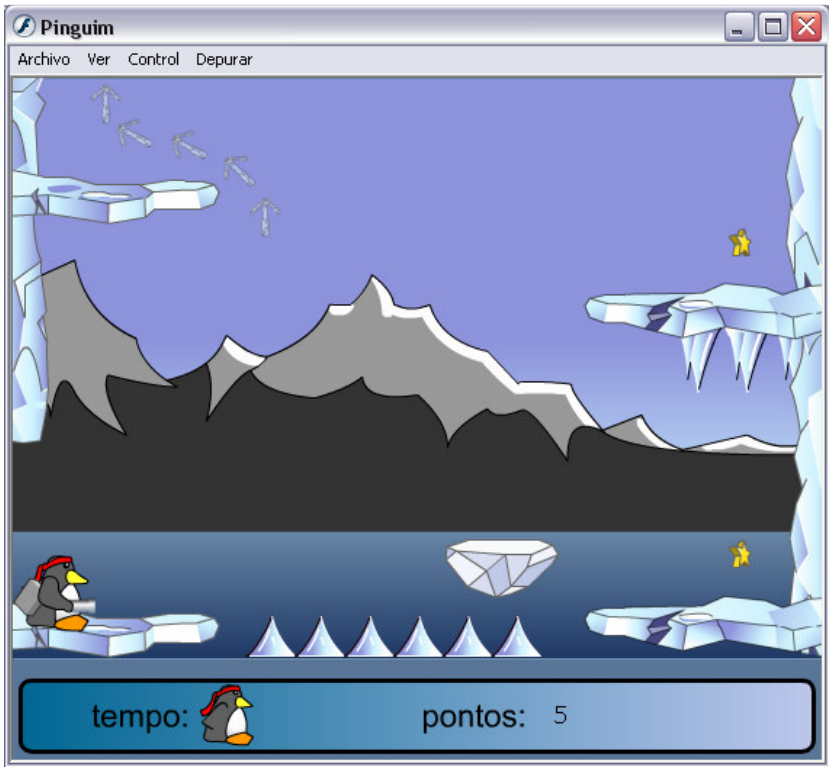

Fig. 4 Cena teste fase 2

Outras funções estão sendo criadas e testadas para melhorar a qualidade dos jogos. Um exemplo é o A* (lê-se "A star") baseado em fases anteriores deste projeto. Trata-se de uma função que calcula o melhor caminho entre um ponto e outro num mapa com obstáculos.

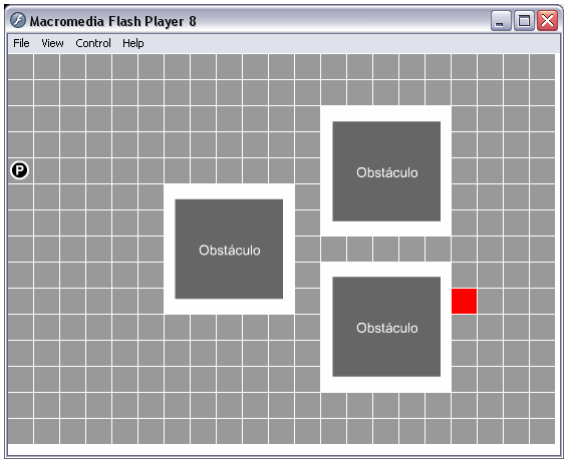

Fig. 5 Cena teste Astar, parte 1

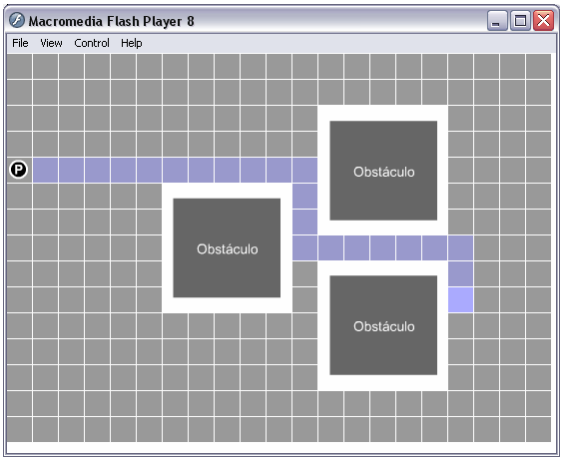

Fig. 6 Cena teste Astar, parte 2

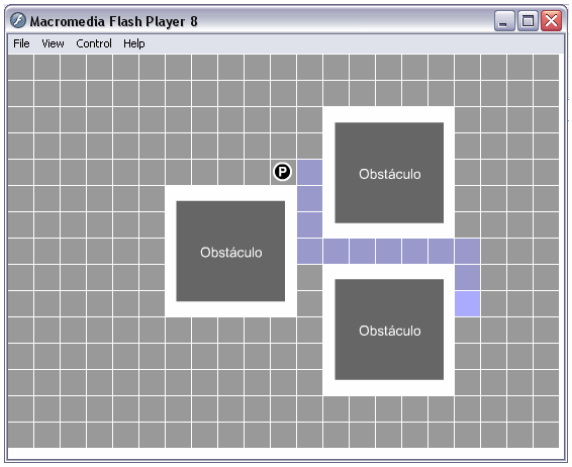

Fig. 7 Cena teste Astar, parte 3

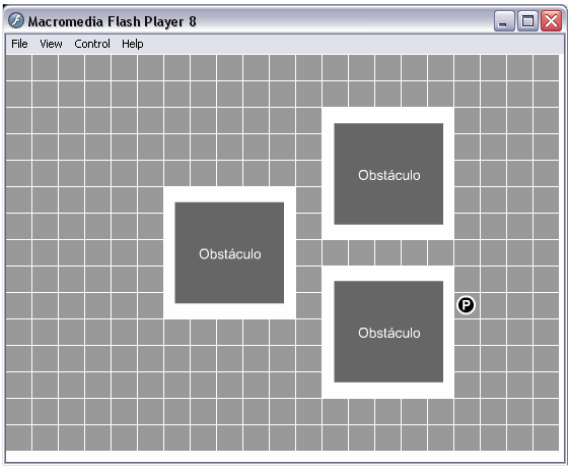

Fig. 8 Cena teste Astar, parte 4

O quadrado em vermelho da Fig. 5 trata-se da posição desejada pelo usuário e selecionada através do mouse. O traçado em azul na Fig. 6 representa o caminho calculado pela máquina de menor trajetória. A Fig. 7 mostra o desenvolvimento do suposto Player pelo caminho e a Fig. 8 exibe, por fim, a chegada ao seu destino.

Esta função pode, inclusive, calcular a trajetória para um novo ponto no processo demonstrado na Fig. 7. Neste caso, a máquina pararia o desenvolvimento do Player, calcularia seu novo caminho (de onde ele foi interrompido até o novo ponto selecionado) e se locomoveria para o novo ponto conforme os passos descritos acima.

Todas essas funções estão em processo de implementação e teste para que se prossiga com uma nova etapa do projeto.

Conclusões

Flash é um interpretador de aplicações multimídia muito utilizado atualmente na internet. Possui a grande vantagem de ser multiplataforma, além de muitas facilidades no desenvolvimento de aplicações interativas. O Flash, porém, não tem um bom suporte ao desenvolvimento de jogos que dependa de programação em ActionScript. O presente projeto pretende desenvolver um motor para jogos em Flash com foco em jogos educativos para web e TV digital interativa. O desenvolvimento até o momento contemplou estudos e esboços em

cima da experiência anterior do Laboratório VisionLab com *frameworks* em Flash. As dificuldades com a biblioteca de máquina de estados devem ser superadas na segunda fase do projeto. Os esforços devem também se concentrar na definição de mundos isométricos e em carregamento de objetos externos dinâmicos.

Bibliografia

Reinhard,R. & Lott,J. Macromedia Flash MX 2004 ActionScript Bible, John Wiley & Sons, 2004.

Moock,C. Essential ActionScript 2.0, O'Reilly, 2004.