Introdução ao Desenvolvimento de Games com GWT e HTML5

Ely Fernando do Prado

Departamento de Computação, Universidade Federal de São Carlos (UFSCar) - São Carlos, SP - Brasil

Figura 1: Logotipos do Google Web Toolkit e do HTML 5.

Resumo

O advento da tecnologia do HTML 5 tem aberto um novo mercado de jogos para internet, onde os usuários podem interagir com o game através de diferentes equipamentos, como computadores, tablets e celulares sem a necessidade de instalação prévia da aplicação ou mesmo algum plug-in. Por outro lado o framework Google Web Toolkit tem se mostrado uma boa alternativa para desenvolvimento de aplicações ricas para internet, utilizando a linguagem Java para gerar códigos HTML, CSS e JavaScript. Assim este trabalho tem por objetivo apresentar o framework GWT como solução para o desenvolvimento de jogos para internet em HTML5, demonstrando todos os passos necessários para codificação de um game loop, animações e interação com o usuário.

Keywords: GWT, HTML5, Jogos, Canvas

Authors' contact:
ely.prado@dc.ufscar.br

1. Introdução

Atualmente estamos presenciando um grande crescimento na demanda por jogos para internet. Outro acontecimento que está em bastante evidencia hoje é o surgimento e amadurecimento do HTML 5, que tem possibilitado a criação de jogos que rodam direto no navegador de maneira leve e prática.

A principal motivação para este tutorial é o grande crescimento no mercado de jogos para internet. O surgimento do HTML 5 permitiu que passássemos a desenvolver aplicações complexas para internet, sem ter que depender de algum plug-in específico. Além disso, aplicações desta natureza podem ser executadas em qualquer dispositivo que possua internet, como computadores, tablets e celulares de qualquer sistema operacional atual.

Um bom exemplo que tem alcançado bastante sucesso entre o público são os jogos adicionados no logotipo do Google, chamados *doodles*. Os *doodles games* são adicionados ao site de pesquisa do Google em comemoração a alguma data ou evento especial, e são jogáveis no próprio site.

Graças a grande experiência alcançada pelos engenheiros da Google no setor de aplicações ricas para internet, foi criado por eles o framework Google Web Toolkit (GWT), que tem facilitado muito criação de aplicações complexas na web, incluindo os jogos.

O objetivo deste tutorial é apresentar o framework GWT como uma alternativa para o desenvolvimento de jogos para internet. Para isso será apresentado como se dá o desenvolvimento de um jogo nesta tecnologia, sendo um jogo com poucas funcionalidades, porém o suficiente para dar os primeiros passos neste framework.

2. HTML 5

O padrão HTML5 complementa as capacidades das normas existentes no HTML com vários novos recursos. Embora o HTML5 seja um padrão web de propósito geral, muitos dos novos recursos são destinados diretamente para tornar a Web um lugar melhor para aplicações web com estilo desktop.

Dentre os novos recursos estão a capacidade das aplicações executarem em modo off-line e de armazenar dados localmente no computador ou dispositivo. Um recurso importante, especialmente quando se trata de desenvolvimento de jogos é o elemento Canvas, que oferece uma tela de desenho 2D, permitindo o desenho de formas gráficas, imagens e texto em tempo de execução. Outros recursos disponibilizados pelo HTML5 são para permitir que arquivos de mídia (áudio e vídeo) sejam executados no navegador sem necessidade de plug-in externo, também há elementos para carregamento de dados de forma assíncrona e apoio para eventos de arrastar e soltar. Além dos recursos citados, a especificação HTML5 define inúmeros outros acréscimos, mas muitas destas especificações, bem como a especificação do HTML5 em si, estão ainda em definição, de modo que na versão final os seus detalhes podem variar. (Taivalsaari e Mikkonen, 2011)

Para o desenvolvimento de jogos o HTML por si só não é suficiente. O HTML é uma linguagem de marcação, que permite incluir elementos em uma página, como campos de formulário, texto, imagens, canvas, etc. Mas todos esses elementos são estáticos. Para superar as limitações do HTML podemos utilizar o Javascript, pois a ação toda precisa ser escrita em uma linguagem de programação. Javascript é uma linguagem de programação poderosa, com sintaxe baseada em C++, porém com suporte apenas parcial à orientação a objetos. Javascript é uma linguagem interpretada, sendo assim sua velocidade de execução e sua compatibilidade depende da máquina interpretadora que o navegador possui. (Nörnberg, 2011)

3. Google Web Toolkit

O framework GWT (Google Web Toolkit) foi criado para facilitar o desenvolvimento de aplicações ricas para web fornecendo uma camada de abstração, que esconde os detalhes do Javascript e também as diferenças entre os ambientes específicos dos navegadores. Toda aplicação é escrita utilizando a linguagem Java, e o framework GWT traduz este código em JavaScript, DHTML e CSS. Ao efetuar esta compilação são geradas versões especificas da aplicação para cada tipo de navegador, tornando a aplicação compatível com os mais variados ambientes, e também com as diferentes versões desses navegadores. (Smeets, Boness e Bankras, 2009)

Seu objetivo é permitir o desenvolvimento produtivo de aplicações Web de alto desempenho sem que o desenvolvedor necessite ser um especialista nas peculiaridades de cada navegador, XMLHttpRequest e JavaScript

GWT é um framework essencialmente para o lado do cliente (*cliente-side*) e dá suporte à comunicação com o servidor através de RPCs (*Remote Procedure Calls*). Ele não é um framework para aplicações clássicas da web, pois deixa a implementação da aplicação web parecida com implementações em desktop. Para quem está habituado a desenvolver aplicações desktop, especialmente na linguagem Java se sente familiarizado com o uso do framework GWT. (Geary, 2008)

O GWT é utilizado por muitos produtos do Google, incluindo o Google Wave e Google AdWords. Tem sido utilizado também para a construção de jogos para internet, como por exemplo, a versão web do jogo Angry Birds.

GWT é de código aberto, totalmente gratuito, e utilizado por milhares de desenvolvedores ao redor do mundo. Está disponível sob a Licença Apache v. 2.0, concedendo-lhe uma licença perpétua, mundial, não exclusiva, sem nenhum custo, isenta de royalties, direitos autorais irrevogáveis para reproduzir, preparar trabalhos derivados, publicamente exibir, executar publicamente, sublicenciar e distribuir o trabalho.

Já que GWT compila código Java para JavaScript, é importante questionarmos quais são as vantagens de se desenvolver a aplicação em Java com GWT, ao invés de escrever diretamente em código JavaScript. A vantagem mais óbvia está no fato de GWT criar JavaScript perfeitamente compatível com os diferente navegadores, sendo assim não precisamos escrever estruturas condicionais para cuidar das diferenças do navegadores. Mas Dwyer, 2008, ainda faz a seguinte afirmação: "há três áreas específicas em que o GWT supera o JavaScript: escalabilidade, suporte à refatoração, e familiaridade.". Isso se deve ao fato de GWT utilizar a linguagem Java, que apesar de seu nome sugerir o contrário, Java tem mais diferenças com JavaScript do que igualdades.

3.1 Ambiente de Desenvolvimento

Como o framework GWT executa sobre a plataforma Java, você pode preparar seu ambiente de desenvolvimento nos principais sistemas operacionais disponíveis (Windows, Linux ou MacOS), porém é necessário ter instalado o Kit de Desenvolvimento Java em seu computador. Além disso, é requisito básico ter feito download e descompressão do Google Web Toolkit SDK. Ambas ferramentas citadas podem ser encontradas nos links a seguir:

- Java SE Development Kit (JDK):

http://java.sun.com/javase/downloads/

- Google Web Toolkit SDK:

https://developers.google.com/web-toolkit/download

Apesar de não ser um pré-requisito, é bastante interessante utilizar uma IDE de desenvolvimento. Dentre as principais IDEs utilizadas para desenvolvimento de aplicações Java, podemos destacar o Eclipse que possui um plug-in oficial da Google para trabalhar com GWT. Tanto a IDE como o plug-in para desenvolver em GWT podem ser encontrados no link a seguir:

- IDE Eclipse:

http://www.eclipse.org/downloads/

- Google Plugin for Eclipse:

http://dl.google.com/eclipse/plugin/4.2

Para este tutorial foi utilizado a versão 4.2 do Eclipse, apelidada de Juno.

Para configurar o plug-in do GWT no Eclipse, basta clicar no menu do Eclipse "*Help"*, logo em seguida em *"Install New Software"*. Depois clique no botão "*Add"* e digite no campo *"Location"* o endereço http://dl.google.com/eclipse/plugin/4.2 e clique em *"Ok"*. Marque os itens *"Google Plugin for Eclipse", "GWT Designer for GPE"* e *"SDKs"*.Clique no botão *"Next"*, aguarde o download ser terminado e clique em *"Finish"*.

Com essas etapas, o Eclipse está preparado para ser utilizado com ferramente de desenvolvimento para o framework GWT.

3.2 Objeto Canvas

O objeto Canvas representa o elemento de mesmo nome do HTML 5, e pode ser utilizado para processamento de gráficos em tempo de execução. Uma característica importante deste elemento é que ele consegue redimensionar seu conteúdo para adequar à resolução do navegador do usuário. Pode ser executado na maioria dos navegadores atuais, porém em versões antigas de navegadores ele não é aceito, por se tratar de uma tecnologia recente.

3.2.1 Principais métodos

setCoordinateSpaceWidth (int width)
Argumento:
- width: largura em pixels

Descrição:
- Define a largura do espaço interno do Canvas, fazendo com que independente da resolução do dispositivo do usuário, as coordenadas dos objetos sejam sempre as mesmas.

setCoordinateSpaceHeight(int height)
Argumento:
- height: altura em pixels

Descrição:
- Define a altura do espaço interno do Canvas.

setWidth (String width)
Argumento:
- width: largura do objeto, em unidades de CSS (por exemplo: "10px", "1em")

Descrição:
- Define a largura do objeto Canvas na página.

setHeight(String height)
Argumento:
- height: altura do objeto, em unidades de CSS (por exemplo: "10px", "1em")

Descrição:
- Define a altura do objeto Canvas na página.

setFocus(boolean focused)
Argumento:
- focused: indica se o objeto receberá o foco ou se perde o foco.

Descrição:
- Indica o objeto como focado ou não focado. Apenas um objeto da página pode ter o foco por vez, e este é o que receberá todos os eventos de teclado.

3.3 Objeto Context2d

O objeto Context2d faz referência ao elemento de mesmo nome do HTML 5. Refere-se ao contexto de renderização, podendo ser utilizado para desenhar formas e imagens no objeto Canvas.

3.3.1 Principais métodos

setFillStyle(String fillStyleColor)
Argumento:
- fillStyleColor: cor como uma String no formato CssColor.

Descrição:
- Define a cor de preenchimento dos elementos que forem desenhados em seguida.

fillRect (double x, double y, double w, double h)
Argumentos:
- x: posição horizontal do canto superior esquerdo do retângulo
- y: posição vertical do canto superior esquerdo do retângulo
- w: largura do retângulo
- h: altura do retângulo

Descrição:
- Desenha um retângulo.

drawImage (ImageElement image, double dx, double dy)
Argumentos:
- image: imagem no formato de um objeto ImageElement
- dx: posição horizontal do canto superior esquerdo da imagem
- dy: posição vertical do canto superior esquerdo da imagem

Descrição:
- Desenha uma imagem previamente carregada.

3.4 Objeto Timer

O objeto Timer é um temporizador, que tem por finalidade fazer chamada automática a um método qualquer em um tempo pré-programado.

3.4.1 Principais métodos

scheduleRepeating (int periodMillis)
Argumento:
- periodMillis: o tempo de espera em milissegundos entre cada repetição da chama do método.

Descrição:
- Ativa um temporizador que decorre repetidamente com um tempo determinado.

4. Desenvolvimento de Projeto

Para facilitar a compreensão das ferramentas aqui citadas, será descrito neste capitulo um guia passo a passo de como se dá o desenvolvimento de um jogo em HTML 5 utilizando o framework GWT.

4.1 Projeto Inicial

Para criar uma nova aplicação GWT no Eclipse, basta clicar no menu "*File -> New -> Project"*. Selecione o tipo de projeto "*Google -> Web Application Project"* e clique em "*Next"*. Digite o nome do projeto e o nome do pacote padrão. Desmarque a opção "*Use Google App Engine"* e também *"Generate project sample code"* para que não seja criada nenhuma classe de exemplo no projeto, deixando assim o projeto apenas com as configurações iniciais básicas. Por fim clique no botão "*Finish*".

O próximo passo após a criação do projeto é criar um novo Módulo GWT clicando o botão direito do mouse no projeto e depois na opção *"New->Other"*. Selecione o item *"Google Web Toolkit->Module"* e clique em *"Next"*. Informe o nome do pacote e do modulo, preferencialmente coloque o mesmo nome de pacote mencionado na criação do projeto e clique em Finish.

Em seguida configure uma classe java como ponto de entrada do sistema. Para isso, clique com o botão direito do mouse no pacote "*client"*, que foi criado automaticamente na criação do módulo. Vá em "*New->Other"*, selecione o item *"Google Web Toolkit->Entry Point Class"* e clique no botão "*Next"*. Preferencialmente digite o nome da classe como "*Main*" e clique em "*Finish"*.

Um recurso importante no projeto é a pasta 'war', onde estarão localizados todos os recursos externos do projeto como imagens, folhas de estilo e arquivos HTML. Nesta pasta também deverá estar uma página HTML que irá iniciar a aplicação. Para criar este arquivo clique com o botão direito do mouse na pasta war e depois em "*New->Other"* e selecione o item "*Google Web Toolkit->HTML Page"*. Digite o nome do arquivo como 'index' e clique em "*Finish"*.

Por fim, é necessário fazer uma pequena configuração para que o aplicativo inicialize o módulo corretamente. Supondo que o módulo criado se chame 'jogo', abra o arquivo 'jogo.gwt.xml' localizado no pacote principal e adicione um apelido ao módulo editando sua tag inicial como:

```
<module rename-to="jogo">
```

Para editar o arquivo XML é necessário clicar na aba "source" na parte inferior do Eclipse e editar a linha mencionada, que é a terceira linha do código.

Edite também o arquivo index.html localizado na pasta war, para que sua chamada ao módulo gwt, na sexta linha, fique da seguinte forma:

```
<script type="text/javascript"
language="javascript"
src="modulo/modulo.nocache.js"></script>
```

Após essas tarefas podemos executar o aplicativo, porém não será mostrado nada no navegador ainda, pois teremos que codificar a classe Main.

Ao abrir a aplicação pela primeira vez é solicitada a instalação de um plugin no navegador. Este plugin é utilizado para depuração do código em tempo de desenvolvimento, e não será solicitado quando o usuário acessar a aplicação final compilada.

Todo esse processo de configuração do projeto poderia ser simplificado apenas deixando a opção *"Generate project sample code"* marcada, porém isso faria com que fosse gerada uma série de códigos desnecessários para nossa aplicação.

Como o objetivo deste projeto é desenvolver um jogo em HTML 5, sua codificação deve ser iniciada com o acréscimo de um elemento Canvas e definição de seus parâmetros. Primeiro deve ser feita uma verificação se o navegador do usuário dá suporte ao elemento Canvas do HTML 5. Depois devem ser removidas a margem e a barra de rolagem da página. Por fim são definidos o tamanho e a resolução do elemento Canvas e logo em seguida adiciona-o na página do usuário, conforme mostrado no quadro 1.

```
public class Main implements EntryPoint {
 private Canvas canvas;
 private Context2d context;
 private static final int WIDTH = 800;
 private static final int HEIGHT = 600;

 @Override
 public void onModuleLoad() {
  canvas = Canvas.createIfSupported();
  if (canvas == null) {
    RootPanel.get().add(
new Label("Navegador sem suporte ao
Canvas"));
    return;
  }
  Window.setMargin("0px");
  Window.enableScrolling(false);
  canvas.setWidth(Window.getClientWidth() +
"px");
  canvas.setHeight(Window.getClientHeight()
+ "px");
  //define resolução do objeto canvas
  canvas.setCoordinateSpaceWidth(WIDTH);
  canvas.setCoordinateSpaceHeight(HEIGHT);
  //adiciona o elemento na interface
  RootPanel.get().add(canvas);
  context = canvas.getContext2d();
 }
}
```

Quadro 1 – Configurações Iniciais o Projeto

4.2 Game Loop

Os jogos são dirigidos por um loop que executa uma série de tarefas a cada frame, construindo a ilusão de um mundo animado. No caso de um jogo que roda a 30 frames por segundo, cada execução das tarefas de um frame devem ser feitas em 33,3 milissegundos. (Rabin, 2012).

Para conseguir executar este loop no tempo previsto, pode-se utilizar o objeto Timer, o qual controlará o tempo de execução de cada frame. A principio as duas tarefas executadas pelo frame serão desenvolvidas nos métodos *"update"* e *"draw"*. O

método *"update"* é responsável pode atualizar a posição dos objetos do jogo, assim como as transformações necessárias para prover os efeitos de animação. O método *"draw"* é responsável por renderizar as imagens no objeto canvas. Pode-se ver a estrutura de um game loop no Quadro 2, onde o método *"update"* ainda não possui nenhuma funcionalidade efetiva e o método *"draw"* apenas pinta uma cor de fundo no objeto canvas. O restante do código será demonstrado nos próximos capítulos.

```
public class Main implements EntryPoint {
 private Canvas canvas;
 private Context2d context;
 private static final int WIDTH = 800;
 private static final int HEIGHT = 600;
 private Timer timer;

 @Override
 public void onModuleLoad() {
   canvas = Canvas.createIfSupported();

   if (canvas == null) {
      RootPanel.get().add(new
Label("Navegador sem suporte ao Canvas"));
      return;
   }
   Window.setMargin("0px");
   Window.enableScrolling(false);
   canvas.setWidth(Window.getClientWidth() +
"px");
   canvas.setHeight(Window.getClientHeight()
+ "px");
   canvas.setCoordinateSpaceWidth(WIDTH);
   canvas.setCoordinateSpaceHeight(HEIGHT);
   RootPanel.get().add(canvas);
   context = canvas.getContext2d();

   timer = new Timer() {
      @Override
      public void run() {
         update();
      }
   };
   timer.scheduleRepeating(33);
 }

 public void update() {
   //lógica do jogo

   draw();
 }
 public void draw() {
   //desenha o fundo
   CssColor cor;
   cor = CssColor.make("rgba(0,50,255,1)");
   context.setFillStyle(cor);
   context.fillRect(0, 0, WIDTH, HEIGHT);
 }
}
```

Quadro 2 – Game Loop

4.3 Animação

Já que GWT permite a codificação da aplicação na linguagem Java, nada mais justo que tirar proveito dos recursos da orientação em objetos para desenvolver seu game. Uma boa metodologia é criar uma classe java para cada tipo funcionalidade, pensando nas técnicas de reuso que a linguagem disponibiliza.

Desta forma, para criar um objeto retangular que será desenhado no jogo, pode-se definir atributos encapsulados para determinar sua posição nos eixos horizontal e vertical, além de declarar um método capaz de fazer a detecção da colisão entre outro retângulo. O resultado desta Classe Java pode ser visto no Quadro 3.

```
public class Retangulo {
 private int x;
 private int y;
 private int height;
 private int width;

 public Retangulo(int x, int y, int height,
int width) {
   this.x = x;
   this.y = y;
   this.height = height;
   this.width = width;
 }

 public int getX() {return x;   }
 public void setX(int x) {
   this.x = x;
 }
 public int getY() {return y;   }
 public void setY(int y) {
   this.y = y;
 }
 public int getHeight() {
   return height;
 }
 public void setHeight(int height) {
   this.height = height;
 }
 public int getWidth() {
   return width;
 }
 public void setWidth(int width) {
   this.width = width;
 }
 public boolean colide(Retangulo c) {
   if ((c.getX()+c.getWidth() >= getX())
   && (c.getX() <= getX() + this.width)
   && (c.getY() + c.getHeight() >= getY())
   && (c.getY() <= getY() + this.height)){
      return true;
   } else {
      return false;
   }
 }
}
```

Quadro 3 – Objeto Retangulo

Desta forma já podemos construir toda a lógica do jogo. Como a ideia é apenas apresentar a tecnologia, iremos demonstrar o desenvolvimento de um jogo simples, apenas com as classes Main e Retangulo. Assim a estrutura do jogo deve ficar igual à mostrada na figura 2.

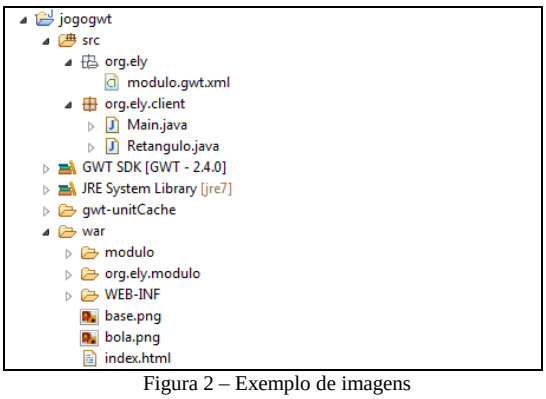

Figura 2 – Exemplo de imagens

O escopo do projeto ficará então como sendo um jogo onde o usuário irá controlar uma plataforma, a qual irá rebater uma bola em movimento. A partir deste exemplo o desenvolvedor poderá facilmente estender suas funcionalidades para tornar o jogo parecido com um Pong ou Breakout mostrados na figura 3. Podem ser utilizados gráficos com ótima qualidade já que as imagens que serão inseridas são no formato PNG, bem diferente do que se tinha na época em que esses dois jogos citados foram criados.

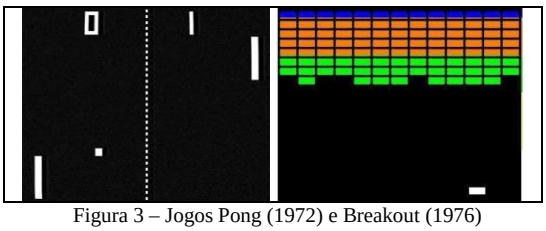

Figura 3 – Jogos Pong (1972) e Breakout (1976)

Para continuar o desenvolvimento, deve ser declarado dois atributos retângulo na classe Main, um para a plataforma base e outro para a bola. Também declare atributos para armazenar a direção a qual a bola deverá se movimentar. A cada frame, incremente a posição da bola para dar a ilusão de movimento, e desenhe os dois retângulos na tela.

Sobre a movimentação da bola, é importante que ela mude de direção sempre que colidir com alguma das extremidades da tela, ou mesmo com a plataforma base também. Algumas estruturas condicionais podem realizar esta tarefa, além de fazerem a validação, caso a bola encontre a extremidade inferior da tela informando ao usuário que ele perdeu o jogo.

Considerando os critérios mencionados, a classe Main deverá ficar conforme está mostrado no quadro 4.

```
public class Main implements EntryPoint {
 private Canvas canvas;
 private Context2d context;
 private static final int WIDTH = 800;
 private static final int HEIGHT = 600;
 private Timer timer;

 private Retangulo base
= new Retangulo(350, 550, 25, 100);
 private Retangulo bola
= new Retangulo(0, 0, 25, 25);
 private int dirX = 3;
 private int dirY = 3;

 @Override
 public void onModuleLoad() {...

 public void update() {
   bola.setX(bola.getX()+dirX);
   bola.setY(bola.getY()+dirY);
   if (bola.getX()+bola.getWidth()>=WIDTH) {
      dirX *= -1;
   }
   if (bola.getX()<=0) {
      dirX *= -1;
   }
   if (bola.getY()<=0) {
      dirY *= -1;
   }
   if (base.colide(bola)) {
      dirY *= -1;
   }
   if (bola.getY()+bola.getHeight()>=HEIGHT)
   {
      Window.alert("Você Perdeu!");
      bola.setX(0);
      bola.setY(0);
      dirX = Math.abs(dirX);
      dirY = Math.abs(dirY);
   }

   draw();
 }
 public void draw() {
   //desenha fundo
   CssColor cor;
   cor = CssColor.make("rgba(0,50,255,1)");
   context.setFillStyle(cor);
   context.fillRect(0, 0, WIDTH, HEIGHT);

   //desenha bola e base
   cor = CssColor.make("rgba(0,255,50,1)");
   context.setFillStyle(cor);
   context.fillRect(bola.getX(),
bola.getY(), bola.getWidth(),
bola.getHeight());
   context.fillRect(base.getX(),
base.getY(), base.getWidth(),
base.getHeight());
 }
}
```

Quadro 4 – Animação

4.4 Interação por Teclado

Em se tratando de interação do usuário com o jogo, uma das formas muito utilizadas é a interação por teclado. No exemplo apresentado neste tutorial, vamos controlar a plataforma base com as setas do teclado para direita e para esquerda. Para esta tarefa foram declaradas duas variáveis booleanas, as quais indicarão o estado de cada tecla. Também foi criado um método chamado *'initKeyHandlers'* responsável por inicializar a captura das ações de pressionar alguma tecla

(*KeyDownHandler*) e de soltar alguma tecla (*KeyUpHandler*).

Através das variáveis de estado para as setas esquerda e direta do teclado, podemos movimentar a plataforma base nesses mesmos sentidos. Esta tarefa é realizada no método *"update"* conforme pode ser visto no Quadro 5.

```
public class Main implements EntryPoint {
 ...

 private boolean keyLeft = false;
 private boolean keyRight = false;

 @Override
 public void onModuleLoad() {
   ...

   initKeyHandlers();
   canvas.setFocus(true);
 }

 public void initKeyHandlers() {
   canvas.addKeyDownHandler(
new KeyDownHandler() {
   @Override
   public void onKeyDown(KeyDownEvent event)
   {
     int key = event.getNativeKeyCode();
     if (key == 37)  {
       keyLeft = true;
     } else if (key == 39)  {
       keyRight = true;
     }
   }
  });
  canvas.addKeyUpHandler(
new KeyUpHandler() {
   @Override
   public void onKeyUp(KeyUpEvent event) {
     int key = event.getNativeKeyCode();
     if (key == 37)  {
        keyLeft = false;
     } else if (key == 39)  {
        keyRight = false;
     }
   }
  });
}

 public void update() {
   ...
   if ((keyLeft) && (base.getX()>0)) {
      base.setX(base.getX()-10);
   }
   if ((keyRight) &&
(base.getX()+base.getWidth()<WIDTH)) {
      base.setX(base.getX()+10);
   }

      draw();
   }
   public void draw() {...
   }
}
```

Quadro 5 – Interação por Teclado

4.5 Interação por Mouse e Toque

Frequentemente o teclado pode não ser a única forma utilizada para interação do usuário com o jogo, podendo este utilizar o mouse ou até mesmo o toque em dispositivos que disponibilizam tal entrada.

Quando estamos tratando de um jogo para internet, temos que ter a consciência de que este aplicativo pode ser acessado por uma diversa quantidade de dispositivos, como computadores, celulares, tablets e até mesmo videogames. Portanto, tratar outros tipos de entrada, como mouse ou toque passa a ser uma exigência do jogo, caso queira ter compatibilidade com todos esses dispositivos.

No framework GWT, o mouse e o toque são tratados por métodos distintos. Sendo assim, vamos criar um método chamado *initMouseHandlers*, para cuidar destes tipos de entrada. A intenção é movimentar a plataforma base do jogo de acordo com a posição do mouse ou do toque. Muito cuidado ao atribuir a posição da base, pois o objeto Canvas estará na resolução 800x600 independente da resolução e tamanho do navegador, porém o mouse ou toque retornarão a posição relativa ao dispositivo. Sendo assim deve ser realizado um calculo de conversão como mostrado no Quadro 6.

```
public class Main implements EntryPoint {
...
 @Override
 public void onModuleLoad() {
   ...
   initMouseHandlers();
 }

 public void initKeyHandlers() {...
 public void initMouseHandlers() {
   canvas.addMouseMoveHandler(new
MouseMoveHandler() {
   @Override
   public void onMouseMove(
MouseMoveEvent event) {
     int x = event.getX();
     x = x*WIDTH/Window.getClientWidth();
     base.setX((x-base.getWidth()/2));
   }
   });
   canvas.addTouchMoveHandler(
new TouchMoveHandler() {
   @Override
   public void onTouchMove(TouchMoveEvent
event) {
     int x =
event.getTouches().get(0).getClientX();
     x = x*WIDTH/Window.getClientWidth();
     base.setX((x-base.getWidth()/2));
   }
   });
 }
 public void update() {...
 public void draw() {...
}
```

Quadro 6 – Interação por Mouse e Toque

4.6 Imagens

A última etapa deste tutorial se refere ao uso de imagens no jogo, ao invés de simplesmente desenhar formas geométricas. É aconselhável que seja usado o formato de imagens PNG, devido ao seu ótimo algoritmo de compressão.

Desenhe no editor de imagens de sua preferência uma figura para a bola, na largura de 25px e altura de 25px. Depois desenhe uma figura para a plataforma base com largura de 100px e altura de 25px. Salve ambas figuras na pasta *'war'* do seu projeto.

Para utilizar as imagens, deverá ser declarado na classe *Main*, objetos do tipo *ImageElement*, os quais terão funcionalidade de carregar as imagens para que sejam desenhadas no objeto *Canvas*. O código para tal tarefa pode ser visto no Quadro 7.

```
public class Main implements EntryPoint {
 ...

 private ImageElement imgBola =
ImageElement.as(new
Image("bola.png").getElement());
 private ImageElement imgBase =
ImageElement.as(new
Image("base.png").getElement());

 @Override
 public void onModuleLoad() {...
 public void initKeyHandlers() {...
 public void initMouseHandlers() {...
 public void update() {...

 public void draw() {
   //desenha fundo
   CssColor cor;
   cor = CssColor.make("rgba(0,50,255,1)");
   context.setFillStyle(cor);
   context.fillRect(0, 0, WIDTH, HEIGHT);

   //desenha bola e base
   context.drawImage(imgBase, base.getX(),
base.getY());
   context.drawImage(imgBola, bola.getX(),
bola.getY());
 }
}
```

Quadro 7 – Renderização e imagens

Após a inserção das imagens o jogo fica semelhante ao que está mostrado na Figura 4. Caso queira fazer animações, basta trocar as imagens a cada frame, o que torna o visual do jogo bem mais agradável. Para isso pode ser utilizada a técnica de sprites. Outra sugestão interessante é o uso de uma imagem de fundo, que pode ser desenhada da mesma forma que as imagens da base e da bola.

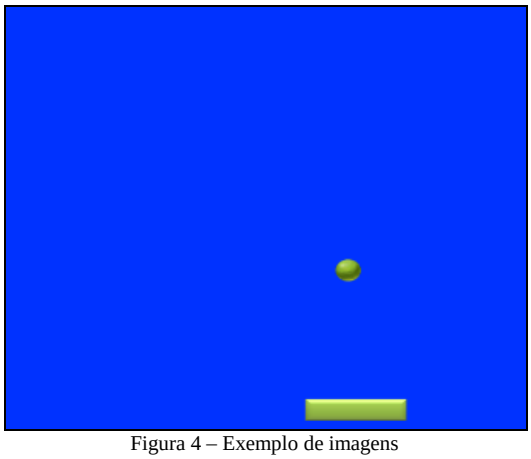

Figura 4 – Exemplo de imagens

5. Perspectivas

O objetivo deste tutorial foi apresentar o framework GWT como alternativa para o desenvolvimento de jogos para internet. Percebemos que este jogo pode ser implementado com facilidade e que o framework GWT proporciona um bom ambiente de desenvolvimento e uma boa organização do código.

A partir dos conceitos demonstrados aqui, este jogo pode ser estendido, aumentando sua complexidade e melhorando sua jogabilidade. Com os recursos proporcionados pela programação orientação a objetos, pode-se criar uma aplicação com muitas linhas de código, sem perder a organização do projeto. Estes recursos também proporcionam a utilização de técnicas de reuso do código, como encapsulamento, herança e outros.

Apesar do HTML 5 ser uma tecnologia recente já é possível desenvolver jogos com qualidade e a tendência é melhorar ainda mais, tanto na compatibilidade com os navegadores, quanto na velocidade de renderização proporcionado pelos mesmos.

Ainda poderemos ter alterações no que se refere ao HTML 5 e até mesmo nas técnicas para desenvolvimento de jogos para internet, já que o consórcio internacional W3C ainda não homologou oficialmente esta linguagem. Mas o framework GWT tem se adaptado bem às ultimas mudanças e é de interesse da Google continuar a manter o framework o mais atualizado possível, continuando a nos proporcionar compatibilidade garantida com os mais diversos navegadores da internet.

Referências

SMEETS, B., BONESS U. E BANKRAS R., 2009. Programando Google Web Toolkit, Rio de Janeiro: Alta Books, 1a edição.

GEARY, D., 2008. Google Web Toolkit solutions. Boston: Pearson.

DWYER. J., 2008. Pro Web 2.0 Application Development with GWT. New York: Apress

TAIVALSAARI, A. E MIKKONEN, T., 2011. The Web as an Application Platform: The Saga Continues. 37th EUROMICRO Conference on Software Engineering and Advanced Applications

NÖRNBERG, L. HTML 5 – futuro dos jogos para web. Disponível em <http://abrindoojogo.com.br/html-5-futuro-dos-jogos-para-web> Acessado em setembro, 2011.

Google Web Toolkit API Reference. Disponível em <http://google-web-toolkit.googlecode.com/svn/javadoc/2.4/index.html> Acessado em setembro, 2011.

Google Web Toolkit. Disponível em <https://developers.google.com/web-toolkit/> Acessado em setembro, 2011.

RABIN, S., 2012. Introdução ao desenvolvimento de games, São Paulo: Cengage, 2ª Edição