MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS
CAMPUS SÃO JOÃO EVANGELISTA

Curso de Bacharelado em Sistemas de Informação

Bruno Dias De Andrade
Thiago Oliveira Reis

HTML5 x FLASH:
Um estudo exploratório sobre a capacidade do HTML5 em tornar obsoleto o Adobe Flash Player.

São João Evangelista
2013

Bruno Dias De Andrade
Thiago Oliveira Reis

HTML5 x FLASH:
Um estudo exploratório sobre a capacidade do HTML5 em tornar obsoleto o Adobe Flash Player.

Monografia apresentada ao Curso de [Bacharelado em] Sistemas de Informação do Instituto Federal de Minas Gerais – Campus São João Evangelista, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Rosinei Soares Figueiredo
Coorientador: Fernando Henriques Mafra

São João Evangelista
2013

Bruno Dias De Andrade
Thiago Oliveira Reis

HTML5 x FLASH:
Um estudo exploratório sobre a capacidade do HTML5 em tornar obsoleto o Adobe Flash Player.

Monografia apresentada ao Curso de Bacharelado em Sistemas de Informação do Instituto Federal de Minas Gerais – Campus São João Evangelista, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

_____
Rosinei Soares Figueiredo (Orientador) – IFMG

_____
Fernando Henriques Mafra (Coorientador) – IFMG

_____
Ricardo Pimentel Bittencorut (Membro Convidado) – IFMG

São João Evangelista, 14 de Novembro de 2013.

RESUMO

Este estudo realizou alguns experimentos na construção de páginas Web utilizando o HTML5 e o Flash. Seu objetivo foi analisar se a quinta versão da Linguagem de Marcação de Hipertexto, HTML5, é capaz de substituir o uso do Flash. Foi realizado um estudo sobre as duas tecnologias, suas linguagens de marcação e programação, e realização de testes utilizando ambas para o propósito de desenvolver páginas para Web. O trabalho analisou alguns parâmetros de comparação, como, quantidade de código gerado e uso de memória física e virtual do computador. Os resultados mostram que o uso do HTML5 ainda não pode tornar a ferramenta do Adobe obsoleta, por não possuir uma IDE de desenvolvimento para animações, além de ser complexo o seu desenvolvimento, gerando um código consideravelmente maior em relação ao Flash, que por sua vez, possui uma IDE com linguagem de programação própria, que torna o desenvolvimento mais fácil, intuitivo e reduz o tamanho do código.

Palavras Chave: Páginas Web. HTML5. Flash.

ABSTRACT

This study performed some experiments in building Web pages using HTML5 and Adobe Flash. Their goal was to examine whether the fifth version of Hypertext Markup Language, HTML5, is able to replace the use of Flash. A study on the two technologies, their markup languages and programming, and testing using both for the purpose of developing Web pages for The paper analyzed some benchmarks, such as, amount of generated code and physical memory usage was conducted and virtual computer. The results show that the use of HTML5 cannot make obsolete Adobe tool, has no development to an IDE animations, and its development is complex, generating a considerably larger compared to the Flash code, which in turn, an IDE with its own programming language, which makes development easier, intuitive and reduces code size.

Keywords: HTML5. Web pages. Flash.

LISTA DE FIGURAS

FIGURA 1 - Cliente-Servidor 14
FIGURA 2 - Estrutura DOM 17
FIGURA 3 - Estrutura HTML 17
FIGURA 4 - Página HTML 17
FIGURA 5 - Esquema de uma página em HTML5 20
FIGURA 6 - Criação do Elemento Canvas 22
FIGURA 7 - Código em JavaScript 22
FIGURA 8 - Código em ActionScript 3.0 24
FIGURA 9 - Ilustração do programa Adobe Flash Professional CS5 Versão de Avaliação ... 25
FIGURA 10 - Criação de um quadro em Flash 29
FIGURA 11 - Código em HTML com inserção de um arquivo em Flash 29
FIGURA 12 - Quadro em Flash 30
FIGURA 13 - Código em HTML5 para inserção de um quadro na página 31
FIGURA 14 - Quadro em HTML5 31
FIGURA 15 - Animação em Flash 33
FIGURA 16 - Código da Animação em Flash 34
FIGURA 17 - Animação em HTML5 35
FIGURA 18 - Código de Áudio em Flash 37
FIGURA 19 - Áudio em Flash 37
FIGURA 20 - Código de Áudio em HTML5 38
FIGURA 21 - Áudio em HTML5 39
FIGURA 22 - Código de Vídeo em Flash 41
FIGURA 23 - Vídeo em Flash 41
FIGURA 24 - Código de Vídeo em HTML5 42
FIGURA 25 - Vídeo em HTML5 42
FIGURA 26 - Jogo Árcade 44
FIGURA 27 - Código em Flash do Jogo Árcade 1 - Continua 45
FIGURA 28 - Código em Flash do Jogo Árcade 2 - Final 45
FIGURA 29 - Código em HTML5 do Jogo Árcade 1 - Continua 46
FIGURA 30 - Código em HTML5 do Jogo Árcade 2 - Final 47

LISTA DE GRÁFICOS

GRÁFICO 1 – Quadro.....32
GRÁFICO 2 – Animação.....36
GRÁFICO 3 – Áudio.....40
GRÁFICO 4 – Vídeo.....43
GRÁFICO 5 – Jogo.....48

SUMÁRIO

1 INTRODUÇÃO 10

2 FUNDAMENTAÇÃO TEÓRICA 12
2.1 Internet 12
2.1.1 *A World Wide Web* 13
2.2 HTML 14
2.2.1 *História do HTML* 15
2.2.2 *Plug-ins* 15
2.2.3 *HTML5* 16
2.2.4 *Funcionamento do HTML5* 16
2.2.4.1 Elementos Estruturais 19
2.2.4.2 Elementos de Conteúdo 20
2.2.4.3 Elemento Canvas 21
2.2.5 *JavaScript* 21
2.3 Flash 23
2.3.1 *ActionScript* 23
2.3.2 *Funcionamento* 24

3 REVISÃO BIBLIOGRÁFICA 26

4 METODOLOGIA E PROCEDIMENTOS 27
4.1 Descrições dos Experimentos 27
4.2 Parâmetros de Comparação 28
4.3 Procedimentos 28
4.3.1 *Quadro* 28
4.3.1.1 Flash 29
4.3.1.2 HTML5 30
4.3.1.3 Comparações Básicas e Resultados 32
4.3.2 *Animação* 32
4.3.2.1 Flash 33
4.3.2.2 HTML5 34
4.3.2.3 Comparações Básicas e Resultados 35
4.3.3 *Áudio* 36
4.3.3.1 Flash 36
4.3.3.2 HTML5 38
4.3.3.3 Comparações Básicas e Resultados 39
4.3.4 *Vídeo* 40
4.3.4.1 Flash 40
4.3.4.2 HTML5 42
4.3.4.3 Comparações Básicas e Resultados 43
4.3.5 *Animação com Interatividade* 44
4.3.5.1 FLASH 44

4.3.5.2 HTML5.....46
4.3.5.3 Comparações Básicas e Resultados.....48

5 CONCLUSÕES.....49

REFERÊNCIAS 51

1 INTRODUÇÃO

A internet surgiu na década de 1960 e foi popularizada em meados da década de 1980, nesta época, o físico britânico chamado Tim Berners-Lee propôs um projeto denominado ENRIQUE, inicialmente foi desenvolvido em linguagem pascal, com o qual Berners-Lee consegui implementar a primeira comunicação bem sucedida entre um cliente HTTP (*HyperText Transfer Protocol*) e um servidor através da internet.

Segundo o portal W3C (2013), o HTML - *Hyper Text Markup Language* é responsável por organizar e formatar páginas que são lidas pelos navegadores web. Esta linguagem surgiu em 1990 e sofreu várias revisões e alterações na sua especificação. Em 1995, o padrão passou a ser regularizado pela *W3C – Word Wide Web Consortium* e, em 1999, teve a sua versão 4.01 aprovada. Em 1998, o W3C decidiu que a especificação do HTML ficaria congelada na versão 4.01, abrindo espaço para uma nova especificação, chamada XML ou XHTML. Em 2009, a W3C parou de trabalhar na especificação do XHTML 2.0 e colocou mais recursos para o desenvolvimento do HTML5.

A proposta dessa nova versão é conseguir estruturar melhor uma página web, permitindo assim que os motores de busca Web possam identificar as áreas da página a partir de suas *tags* específicas, além de eliminar os *plug-ins* externos como o Adobe Flash Player, responsável por reproduzir mídias que são necessárias para o enriquecimento de informações nas páginas web, o que acarreta em uma melhor navegabilidade, maior simplificação de código, melhor interoperabilidade e redução de custos de desenvolvimento.

Segundo o portal Adobe (2013), o Adobe Flash Player é o padrão para fornecimento de conteúdo de alto impacto e rico para a Web. As interfaces do usuário de designs, animação e do aplicativo são implantadas imediatamente em todos os navegadores e plataformas, atraindo e envolvendo os usuários com uma rica experiência para a Web, com o surgimento do HTML5 esse recurso tende a se tornar obsoleto uma vez que o próprio HTML5 irá fazer o trabalho que antes era feito somente pelo Adobe Flash Player.

O HTML5 busca beneficiar tanto o usuário quanto o desenvolvedor, já que torna as páginas mais leves e dinâmicas, simplificando a navegabilidade, além de ser mais seguro, uma vez que não necessita de *plug-ins* instalados, isso busca tornar o comportamento dos diversos tipos de *browsers* interoperáveis, desse modo os navegadores desempenhariam a função de renderizadores e não somente a função de exibição.

O objetivo deste trabalho é realizar um estudo analítico sobre a capacidade do HTML5 em tornar obsoleto o Adobe Flash Player, para alcançar este objetivo será feito um estudo de ambas as tecnologias citadas e implementações, como forma de exemplificação do processo de exploração das tecnologias, com isso, busca-se: comparar a tecnologia do HTML5 e do Adobe Flash Player, analisar a compatibilidade destas tecnologias com os *browsers* atuais e comparar a viabilidade de implementação destas tecnologias.

Esta pesquisa justifica-se com a necessidade de buscar novas implementações e comparações para as novas tecnologias do mercado web, pois é um mercado que cresce significativamente a cada ano e exige melhores tecnologias para seu desenvolvimento e navegação, como é o caso do HTML5 e do Adobe Flash Player, tecnologias que são as bases da programação web e recursos de multimídia na atualidade.

2 FUNDAMENTAÇÃO TEÓRICA

Segundo Cristina (2010), a internet surgiu através da reação do governo norte-americano ao Projeto Sputnik, programa espacial que produziu o primeiro satélite artificial da antiga União das Repúblicas Socialistas Soviéticas (URSS), capitaneadas pela Rússia, durante a Guerra Fria, em 1957. O nascimento da Internet está diretamente relacionado ao trabalho de peritos militares norte-americanos que desenvolveram a ARPANET, rede da Agência de Investigação de Projetos Avançados dos Estados Unidos, durante a disputa do poder mundial com a URSS, em 1962, segundo Turner e Muñoz (2002, p. 27).

2.1 Internet

A Internet é uma rede mundial de computadores que utiliza os protocolos de rede padrão TCP/IP, TCP (*Transmission Control Protocol*) Protocolo de Controle de Transmissão e o IP (*Internet Protocol*), Protocolo de Internet. Para compreendê-la melhor, é necessário introduzir o conceito de Rede de Computadores. Uma rede de computadores é um conjunto de dois ou mais computadores interligados a partir de um meio de comunicação, com qual é possível a troca de informações. Para ser capaz de comunicar em uma rede, um computador deve ser capaz de acessar um meio de comunicação comum a todos e trocar informações utilizando o mesmo "protocolo de comunicações"[1] das outras máquinas. Existem diversos meios de comunicação orientados (fios de cobre e fibra ótica, por exemplo) e meios de comunicação nãos orientados (ondas de rádio e micro-ondas, por exemplo).

É possível acessar a Internet utilizando qualquer um desses meios, desde que se utilizem os protocolos adequados. Atualmente, a estrutura física da Internet compreende cabos de fibra ótica intercontinentais, a própria estrutura física da telefonia pública e a comunicação sem fio. Assim, para se conectar a Internet, basta ter acesso à rede telefônica pública, cabos ou comunicação sem fio interligados a uma Espinha Dorsal (estrutura principal) da Rede Mundial.

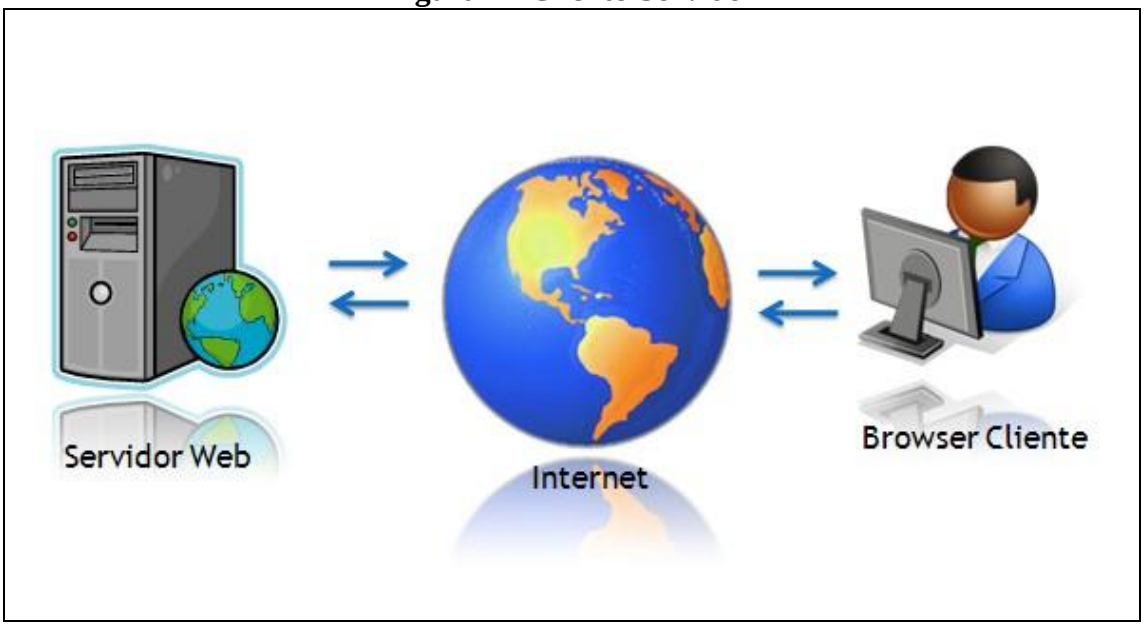
[1] No domínio das redes de computação, um protocolo é um conjunto de especificações objetivas que os computadores entendem.

2.1.1 *A World Wide Web*

A Web, ou Rede de Alcance Mundial em português, representado pela sigla "WWW", é um sistema de informações de escala global que realiza conexões de computadores através de hipermídia (hiper-ligações em forma de texto, vídeo, som entre outras formas de multimídia), pois é possível realizar uma comunicação de leitura e escrita através dos computadores que estejam conectados á internet, logo, a Web permite ao usuário acessar uma infinidade de conteúdos disponibilizados na internet através do navegador (*browser*), responsável pela visualização do conteúdo das páginas web. O termo Web ainda é usado erroneamente como sinônimo da própria Internet, no entanto a Web é apenas um serviço que utiliza a internet, assim como o servidor de e-mail.

O funcionamento da Web pode ser apresentado através de um modelo chamado cliente-servidor, o servidor web funciona com um software que tem o propósito de servir informações necessárias quando há uma requisição de um documento (página de internet) feito pelo cliente web. O cliente web é um software que serve de interface para o usuário (*browser*), além de realizar pedidos de documentos a um servidor especificado pelo usuário, o cliente web faz uma conexão com outro computador através de um endereço especificado pelo usuário, sendo que este endereço pode estar em qualquer lugar da internet, a partir de então, o cliente web pode solicitar ao servidor o documento que deseja. O servidor web responde ao usuário enviando o conteúdo do documento e, junto com ele, qualquer outro elemento de mídia existente (figuras, sons, ou filmes) trazendo a visualização do conteúdo que é mostrado na tela do usuário. A Figura 1 representa o funcionamento do cliente-servidor.

Figura 1 - Cliente-Servidor



Fonte: RIBBENS (2010).

Essa comunicação entre cliente e servidor é, na grande maioria das vezes, feita através de um protocolo denominado de HTTP (*Hypertext Transfer Protocol*), responsável pelo envio e recebimento de documentos de hipermídia, para isso[,] tanto o servidor quanto o cliente precisam usar o protocolo HTTP para a troca de informações, geralmente os servidores web são chamados de servidores HTTP. A criação e reconhecimento de documentos de hipermídia e feita utilizando-se uma linguagem padrão chamada HTML, que será abordado na próxima seção.

2.2 HTML

A tecnologia HTML é a utilizada na construção de páginas web. A estrutura das páginas de internet que o usuário está acostumado a ver é escrita nessa linguagem. Essa linguagem de marcação é utilizada para formatação de documentos, e é responsável pela organização das páginas com relação ao posicionamento do conteúdo, estilo de textos, títulos de documentos, layout, cores, parágrafos, listas, hiperlinks entre outras funções. Algumas dessas funções do HTML exigem um programa externo para sua execução, por exemplo, os vídeos que serão visualizados pelo usuário via browser precisam de um *plug-in* externo para executar esta função. O HTML passou por várias versões e hoje se encontra na versão 5, que será abordada nas próximas seções.

2.2.1 *História do HTML*

De acordo com W3C (2010), entre 1993 e 1995, o HTML ganhou as versões HTML+, HTML2.0 e HTML3.0, e foram propostas diversas mudanças para enriquecer as possibilidades da linguagem. Contudo, até essa versão o HTML ainda não era tratado como um padrão. Apenas em 1997, o grupo de trabalho do W3C, responsável por manter o padrão do código, trabalhou na versão 3.2 da linguagem, fazendo com que ela fosse tratada como prática comum. Desde o começo, o HTML foi criado para ser uma linguagem independente de plataformas, browsers e outros meios de acesso. O desenvolvedor cria apenas um código HTML e este código pode ser lido por diversos meios, ao invés de versões diferentes para diversos dispositivos. Dessa forma, evitou-se que a Web fosse desenvolvida em uma base proprietária, com formatos incompatíveis e limitada.

Por isso, o HTML foi desenvolvido para fazer com que a informação publicada por meio deste código fosse acessível por dispositivos e outros meios com características diferentes, não importando o tamanho da tela, resolução, variação de cor, se são próprios para deficientes visuais e auditivos ou dispositivos móveis e portáteis. Segundo Ferreira e Eis (2013), O HTML deve ser entendido universalmente, dando a possibilidade para a reutilização dessa informação de acordo com as limitações de cada meio de acesso.

2.2.2 *Plug-ins*

Segundo o portal Mozilla (2013), um *plug-in* é um programa instalado no navegador que permite a utilização de recursos não presentes na linguagem HTML, com a qual são criadas as páginas. Este estudo aborda o *plug-in* Flash Player, que é um visualizador de programas escritos em flash. Este *plug-in* é usado pelos *browsers* quando é necessário executar um programa em flash no navegador (como banners animados, jogos ou os vídeos do YouTube). Como exemplo, outro *plug-in* muito comum é o desenvolvido pela empresa Oracle, o Java, que é um programa que permite executar aplicações Java, muito usada pelos *sites* de bancos para criar teclados virtuais e por outros sites para fazer sistemas de notícias. Além destes, há uma variedade enorme de outros *plug-ins* como o instalado pelo Adobe Reader para exibição de arquivos PDF dentro do Firefox, o Windows Media Player e Quick Time para exibição de vídeos.

2.2.3 *HTML5*

De acordo com W3C (2010), ao contrário das versões anteriores, o HTML5 fornece ferramentas para a CSS (*Cascading Style Sheets*)[2], folha de estilo, que é utilizada para definir a aparência em páginas web, e o Javascript, linguagem que pode ser utilizada juntamente com o HTML e que pode inserir vários efeitos, permitindo que a página web fique mais dinâmica e com uma maior interação com o usuário, o JavaScript será abordado na sessão 2.2.5. O HTML5 permite, por meio de suas APIs (*Application Programming Interface*), a manipulação das características destes elementos de forma que o website ou a aplicação continue leve e funcional. O HTML5 também cria novas etiquetas e modifica a função de outras. As versões anteriores ao HTML5 não continham um padrão universal para a criação de seções comuns e específicas como rodapé, cabeçalho, sidebar, menus e etc. Não havia um padrão de nomenclatura de IDs[3], Classes[4] ou etiquetas. O HTML5 modifica a forma organizacional da escrita do código. A versão do HTML5 facilitou o desenvolvimento, pois há mais semântica com menos código, incorpora a funcionalidade de vários *plug-ins* na própria estrutura, assim várias tecnologias funcionam direto do HTML5.

O HTML5 melhora a linguagem e passa a oferecer suporte a recursos de multimídia, diminuindo a dependência de *plug-ins* que antes eram utilizados, como o Flash do Adobe. Mantem o código legível para desenvolvedores e consistentemente compreendido pelos computadores e dispositivos e, ainda, com vantagem de ser uma plataforma de código aberto.

2.2.4 *Funcionamento do HTML5*

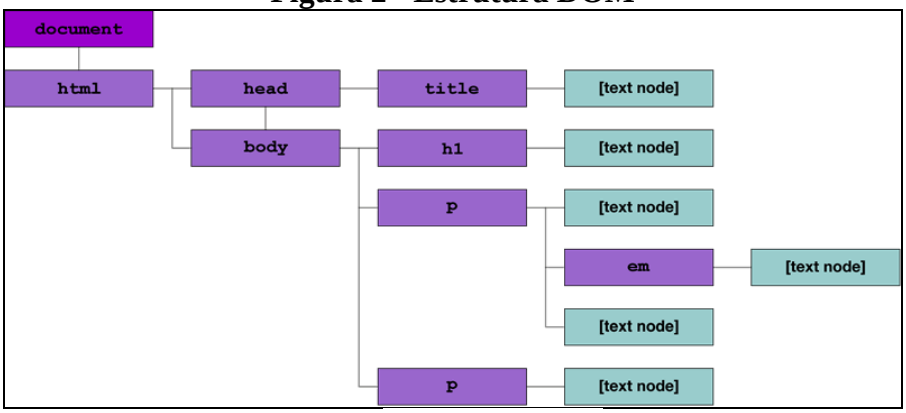
O HTML5, assim como as versões anteriores, é uma linguagem que baseia sua sintaxe em um elemento de base chamado etiqueta (tag). A etiqueta apresenta frequentemente duas partes: uma abertura de forma geral <etiqueta> e um fechamento do tipo </etiqueta>, tudo que estiver incluído no interior desta etiqueta sofrerá as modificações que caracterizam a esta etiqueta.

O HTML5 funciona segundo a estrutura DOM (*Document Object Model*) assim como suas versões anteriores. A estrutura DOM é um modelo de organização dos elementos HTML, a Figura 2 demonstra a estrutura DOM.

---

[2] Tecnologia usada para formatar documentos HTML, XML e XHTML.
[3] É um identificador ou seletor para um elemento no HTML.
[4] Estrutura que abstrai um conjunto de objetos com características similares.

Figura 2 - Estrutura DOM



Fonte: FRANKLIN (2013).

A estrutura DOM representa a hierarquia que deve ser seguida para a confecção de uma página em HTML, nota-se que a estrutura é construída dentro da tag “html”, que por sua vez possui subclasses, utilizadas na criação de uma página.

O código em HTML que irá gerar a página web pode ser escrito em vários softwares, desde softwares mais simples como o Bloco de Notas até IDEs complexas para desenvolvimento de softwares, como o NetBeans. É preciso que os arquivos sejam salvos como .html ou .htm, para que os navegadores saibam interpretá-los, a Figura 3 mostra uma estrutura básica de um código em HTML escrito no Bloco de Notas.

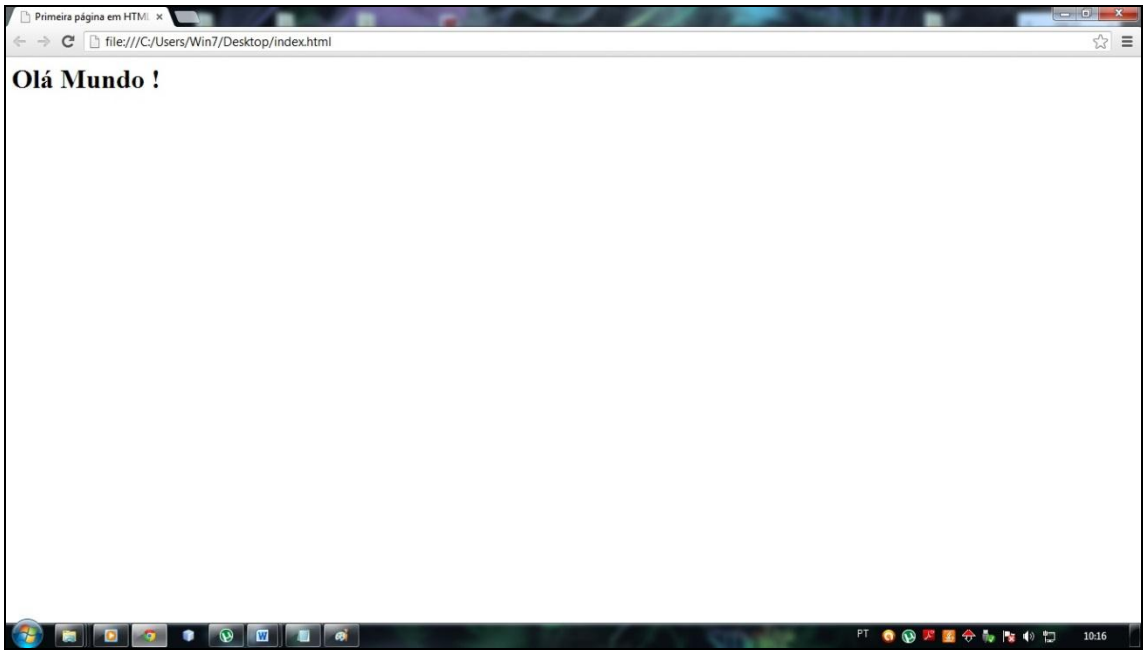
Figura 3 - Estrutura HTML

```
<! DOCTYPE html>
<html>
   <head>
        <title> Primeira página em HTML 5 </title>
   </head>
<body>
   <h1> Olá Mundo !</h1>
</body>
</html>
```

Fonte: Elaborado pelo autor

O resultado do código demonstrado na Figura 3 será demonstrado na Figura 4.

Figura 4 - Página HTML



Fonte: Elaborado pelo autor

A primeira linha a ser declarada em um documento HTML é o <*doctype*>, ele é necessário para que o navegador interprete o documento corretamente, logo abaixo do <*doctype*>, existe a tag <*html*>, esta tag representa a página em si, todo o código dentro da tag <*html*> estará referenciando a página construída pelo código, se alguma tag ou conteúdo estiver fora desta tag, não fará parte da estrutura padrão da página. Em seguida existe a tag <*head*>, que corresponde às informações sobre o documento, como o autor, título, *scripts* e declaração de folhas de estilo CSS. A tag <*title*> corresponde ao título da página que é exibido na barra superior do navegador, geralmente representa o nome do site ou o assunto de um blog.

Toda a estrutura do conteúdo fica contida dentro da tag <*body*>. Os elementos estruturais contidos dentro da tag <*body*> representam o conteúdo do documento, na versão anterior do HTML, o HTML 4.01, usava-se uma tag especial para a divisão das informações que são disponibilizadas no documento, esta tag é declarada como <*div*>, que através do atributo *id*, podia nomear as partes divididas do documento, mas era somente para efeito de estilização e organização do documento, quando bem organizado, facilita os motores de busca assim como a indexação. O HTML5 possui novos elementos que foram introduzidos com a finalidade de facilitar a manutenção do código e a sua compreensão, alguns desses elementos surgiram da necessidade de padronização da maneira de se publicar conteúdo, uma evolução da tag <*div*>, as principais novas tags dessa versão são divididas em elementos de estrutura e elementos de conteúdo que serão abordadas nas próximas seções.

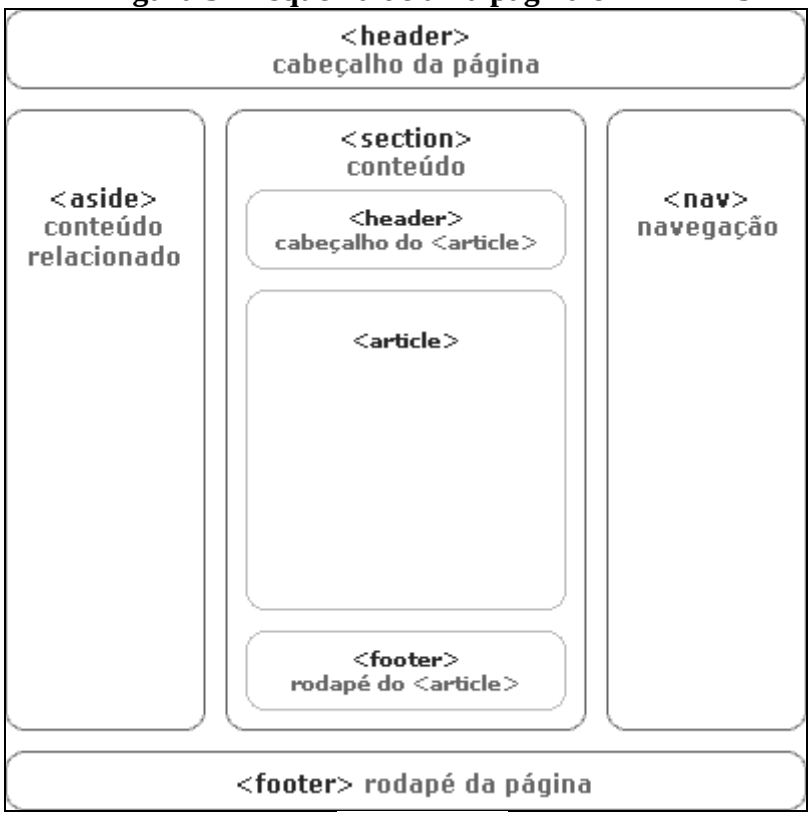
2.2.4.1 Elementos Estruturais

As novas tags que representam os elementos estruturais são:

- <*header*> - representa o cabeçalho da página ou de uma seção, diferente da tag <*head*>, que escreve o que vai estar no cabeçalho da página;
- <*section*> - representa cada seção do conteúdo;
- <*article*> - representa algum item do conteúdo dentro de uma página ou de uma seção;
- <*footer*> - representa o rodapé da página ou de uma seção;
- <*nav*> - representa um conjunto de links que formam a navegação, por exemplo, o menu principal;
- <*aside*> - representa um conteúdo relacionado ao artigo.

A Figura 5 mostra o modelo do corpo de uma página com os novos elementos estruturais do HTML5.

Figura 5 - Esquema de uma página em HTML5



Fonte: MAKIS (2013).

2.2.4.2 Elementos de Conteúdo

As novas tag que representam os elementos de conteúdo são:

- <*figure*> - está tag é utilizada para se associar uma legenda a um vídeo, imagem ou arquivo de áudio;
- <*canvas*> - é utilizada para renderizar imagens 2D dinâmicas que poderão ser usadas em jogos, é utilizada por meio de uma API gráfica;
- <*audio*> e <*video*> - são usadas para a transmissão pela a internet de áudio e vídeo;
- <*dialog*> - é utilizada para formatar um diálogo;
- <*time*> - representa a data e/ou hora;
- <*meter*> - utilizado para representar medidas.

Algumas das *tags* que eram usadas na versão anterior do HTML foram retiradas, uma vez que entraram em desuso ou foram substituídas por declarações em CSS, como: <*font*>, <*s*>, <*strike*>,<*basefont*>,<*big*>, <*center*>, <*tt*> e <*u*>. Algumas outras tags foram tiradas, pois afetam a navegabilidade e a acessibilidade do site como: <*frame*>, <*frameset*> *e* <*noframes*>.

As tags <*b*> e <*i*> ainda serão reconhecidas e renderizadas pelo[s] navegador[es] para fins de formatação, mas devem ser substituídas pelas tag <*strong*>. Outros elementos que foram retiradas são: *target* na tag <a>, *align* nas tags <*table*>, <*iframe*>, <*img*>, <*input*>, <*hr*>, <*div*>, <*p*>, os elementos *background*, *bgcolor* em <*body*>, o elemento *border* em <*table*> e <*object*>, os elementos *cellpadding* e *cellspacing* em <*table*>, *height* em <*td*> e <*th*>, *width* nos elementos <*hr*>, <*table*>, <*td*>, <*th*> e <*pre*>, *hspace* e *vspace* em <*img*> e <*object*>, *noshade* e *size* em <*hr*>.

O HTML5, segundo alguns autores, pode tornar obsoleto o *plug-in* Flash com a nova forma de estruturas de elementos para a multimídia, o que é o objeto de investigação deste trabalho.

2.2.4.3 Elemento Canvas

Segundo o portal W3C (2013), Canvas API permite desenhar na tela do navegador via Javascript. O único elemento HTML existente para isso é o elemento canvas, o todo resto é feito via Javascript. O Elemento Canvas permite especificar uma área da página onde se pode, através de scripts, desenhar e renderizar imagens, o que possibilita criar páginas dinâmicas, permitindo assim fazer coisas que até então estavam limitadas aos desenvolvedores em Flash.

Oficialmente, o canvas é uma tela de bitmap dependente de resolução que pode ser usada para renderizar gráficos, animações de jogos e outras imagens visuais em tempo real. O canvas é um novo elemento do HTML5 que permite desenhar gráficos usando o JavaScript. Pode ser usado para renderizar textos, imagens, gráficos, retângulos, linhas, gradientes e outros efeitos de forma dinâmica. O desenho é feito através da API 2D do canvas. Essa API contém inúmeras funções que permitem desenhar praticamente qualquer coisa na tela. Atualmente, o canvas tem suporte para superfícies 2D, não 3D. A Figura 6 mostra um código de criação do canvas em uma página HTML.

Figura 6 - Criação do Elemento Canvas

```
<!DOCTYPE html>
<html>
    <head>
        <script type="text/javascript">
            window.onload function(){
                var canvas = document.getElementById('exemplo'),
                c = canvas.getContext('2d');

                    c.fillStyle = "Black";
                    c.fillRect (0, 0, 150, 150);
                }
            }
        </script>
    </head>
    <body>
        <canvas id="exemplo" width="240" height="240">
            Use o Chrome!
        </canvas>
    </body>
</html>
```

Fonte: Elaborado pelo autor

2.2.5 *JavaScript*

Apesar de não ser nativo da linguagem HTML, o JavaScript é usado como uma ferramenta poderosa para o desenvolvimento de aplicações Web. Segundo o portal Java (2013), a linguagem de programação JavaScript foi desenvolvida pela Netscape e não faz parte da plataforma Java. Diferente do Java, o JavaScript não cria applets e nem aplicações independentes. É muito comum a utilização do JavaScript em documentos HTML, oferecendo níveis de interatividade com páginas da Web que não podem ser obtidos com HTML simples.

O exemplo apresentado na Figura 7, exibe um código em JavaScript. O código irá apresentar na tela a seguinte mensagem: “HTML5 vs Flash! JavaScript”.

Figura 7 - Código em JavaScript

```
1  <html>
2    <head>
3    </head>
4    <body>
5      <script type="text/javascript">
6          document.write ("HTML5 vs Flash! JavaScript");
7      </script>
8    </body>
9
10 </html>
```

Fonte: Elaborado pelo autor

O código da Figura 7 mostra o funcionamento da criação de uma mensagem escrita na etiqueta “script”.

2.3 Flash

Segundo Pacievitch (2013), Flash é ferramenta para produção e reprodução de multimídia criada nos anos 90 pela empresa Futurewave. O Flash pode ser considerado uma IDE ou um Player. Seu nome era, inicialmente, Smartsketch. Posteriormente a empresa mudou o programa, criando um player em Java, e o programa passou a se chamar Cellanimator. Este programa foi feito após centenas de pedidos dos usuários da internet por algo para criar animações pra Web. A Macromedia comprou a Futurewave e trocou o nome do programa para Flash. O Flash continuou evoluindo, nas mãos da Macromedia, que estava falindo. A empresa foi comprada pelo Adobe em 2005. Logo após a compra os técnicos do Adobe trabalharam e, no mesmo ano, criaram o Adobe Flash Player CS3. Este foi um enorme avanço para o software, nas mãos da maior empresa de design gráfico do mundo. A extensão dos arquivos a partir da versão CS3 é SWF. O Flash, desde 1999, passou a ter uma linguagem totalmente nova, o ActionScript, que permite ao usuário criar loops, usar variáveis, condições, ou seja, aumentou em muito as possibilidades no programa.

2.3.1 *ActionScript*

Actionscript é uma linguagem de programação da plataforma Adobe Flash que foi desenvolvida como um meio para os desenvolvedores programarem dinamicamente, melhorando a eficiência do desenvolvimento de aplicações na plataforma Flash, desde uma imagem simples até complexas animações. O Actionscript está na sua terceira versão, existem duas versões lançadas que são as versões 1.0 e 2.0 (a versão 1.0 foi nomeada somente quando a versão 2.0 foi lançada). No exemplo da Figura 8 é exibido um código em ActionScript 3.0. O código irá apresentar na tela a seguinte mensagem: “HTM5 vs Flash! ActionScript 3.0”.

Figura 8 - Código em ActionScript 3.0

```
1  package {
2      import flash.display.Sprite {
3
4      public class saudacao extends Sprite{
5          public function saudacao()
6          {
7              trace("HTML5 vs Flash! Action Script 3.0");
8          }
9      }
10     }
11 }
```

Figura: Elaborado pelo autor

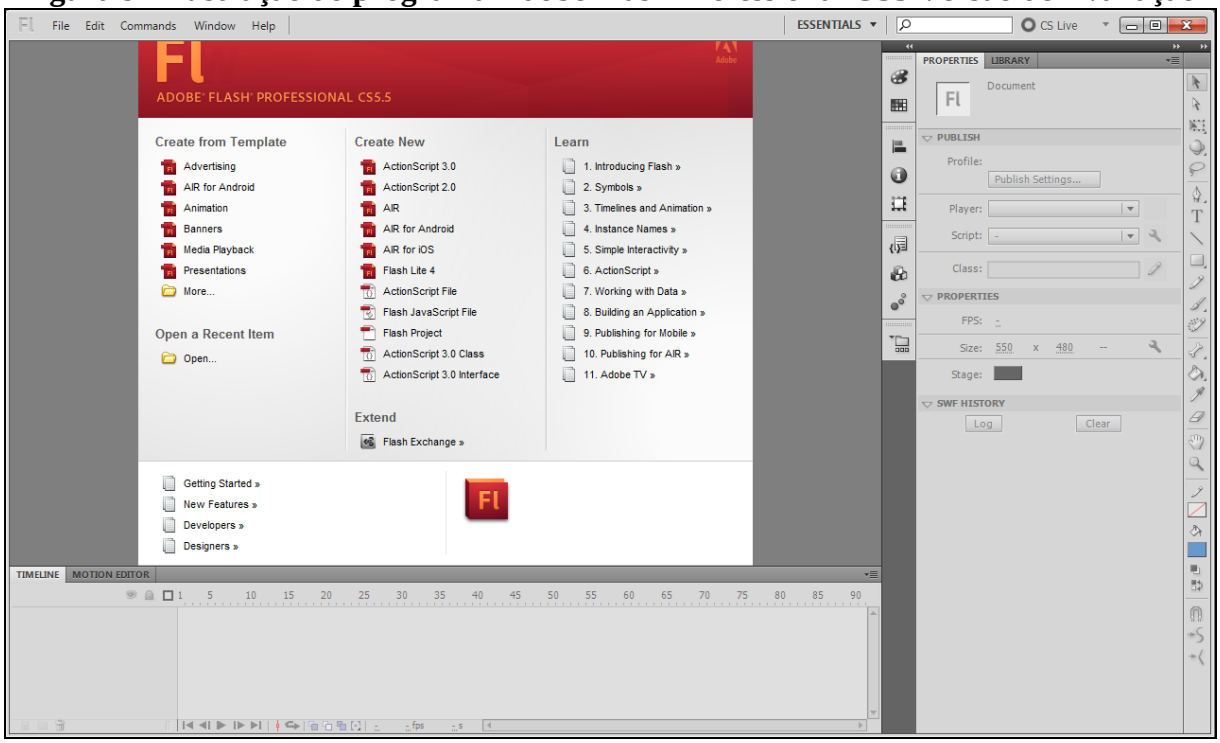
2.3.2 *Funcionamento*

Segundo o Pacievitch (2013), o ActionScript é uma linguagem relativamente fácil, e pode ser usada para criar desde aplicações simples, até efeitos visuais surpreendentes, podendo também suavizar e melhorar os detalhes nos efeitos.

No ambiente de desenvolvimento do Flash, dentro do painel de programação existem várias dicas de programação, facilitando ainda mais. As animações do flash são feitas em camadas, para fazer o movimento de um objeto, pode-se desenhar o objeto no estado inicial, e o mesmo no estado final, então se acrescenta um comando, e o programa faz o percurso, mesclando as duas imagens.

O Flash é um programa leve e fácil de usar, e é usado tanto por amadores quanto por profissionais, pois seus controles são únicos, mesmo os outros programas como o *Moonlight* e o *Silverlight*, não conseguem superar a qualidade do Flash. Atualmente milhões de sites usam Flash em animações, banners, menus, entre outros. A seguir a Figura 9 mostra a interface do Adobe Flash Professional Player CS5, que é uma IDE para desenvolvimento de aplicações multimídia em Flash.

Figura 9 - Ilustração do programa Adobe Flash Professional CS5 Versão de Avaliação



Fonte: Elaborado pelo autor

3 REVISÃO BIBLIOGRÁFICA

Nessa seção serão citados alguns autores cujos temas das pesquisas se relacionam diretamente com o tema abordado neste trabalho. Serão abordados os trabalhos de Fuverki (2010), com o "Estudo de viabilidade do HTML5 para desenvolvimento web" e o trabalho de Alighieri (2009), com o "Estudo do Flash, Usabilidade e Web 2.0".

Segundo Fuverki (2010), que apresenta um estudo de viabilidade do HTML5 para desenvolvimento web, há uma falta de mecanismos nativos na versão 4 do HTML, o que trouxe a necessidade do uso de tecnologias auxiliares que são responsáveis por tarefas que não poderiam ser executadas somente pelo HTML, como por exemplo, a tecnologia Flash, utilizada para a apresentação de recursos de multimídia em uma página HTML. Em sua pesquisa, o autor diz que "é plausível considerar uma análise de viabilidade para desenvolvimento HTML5 em futuras aplicações, relevando sobre as diferentes situações para uso de ferramentas de programação mais adequadas". Fuverki (2010) busca descrever os novos elementos estruturais de marcação do HTML5 com seu funcionamento e conteúdo, mostrando também que não será mais necessário o uso de tecnologias auxiliares, pois nesta versão do HTML não será utilizada as tecnologias que até então eram necessárias para a reprodução de recursos de multimídia. Através da análise de motores de renderização web, Fuverki (2010) visa que as tendências para futuras atualizações e versões dos navegadores apontam para um suporte aos novos elementos, sendo que atualmente 46% de todo tráfego da web já é capaz de visualizar o conteúdo do HTML5, e que muitas funcionalidades já estão sendo empregadas em grandes aplicativos e serviços, como Google, Youtube e Vimeo, mas com potencial ainda grande para aprimoramento e estabilidade.

A pesquisa de Alighieri (2009) trata de aspectos de usabilidade em sites desenvolvidos com a ferramenta Adobe Flash. A finalidade é apoiar pesquisadores, desenvolvedores e estudantes da área a projetarem interfaces que propiciem aos seus usuários uma interação fácil e transparente. O autor trata das duas vertentes que são enxergadas por profissionais e pesquisadores em relação ao Flash, a vertente dos que apoiam a ferramenta pelas suas inúmeras utilidades e a vertente dos que abominam seu uso devido questões de usabilidade e acessibilidade.

Fuveki (2010) e Alighieri (2009) colaboram significativamente para o desenvolvimento desta pesquisa, principalmente para a análise das funções de multimídia que vem competindo forte com as tecnologias do HTML5.

4 METODOLOGIA E PROCEDIMENTOS

Este projeto trata-se de uma pesquisa experimental. Busca-se construir uma comparação entre as tecnologias HTML5 e Flash, um estudo exploratório sobre a capacidade do HTML5 em tornar obsoleto o Adobe Flash Player. As principais atividades que compõem este estudo são:

- Fazer uma revisão da literatura: esta etapa serve para selecionar o material necessário para o estudo de ambas as tecnologias e é também uma fase relacionada à confecção do documento do TCC;
- Definir os parâmetros de comparação: esta etapa refere-se ao estudo e definição dos parâmetros que serão utilizados para comparar as duas tecnologias;
- Definir, modelar e projetar um estudo de caso: esta fase é responsável pela definição da implementação que será realizada nas tecnologias abordadas;
- Construir o estudo de caso: esta fase busca a implementação do estudo de caso definido anteriormente;
- Comparar as tecnologias: esta fase é responsável pelas comparações das tecnologias em relação aos critérios que serão definidos;
- Analisar e divulgar os resultados: esta fase busca mostrar qual foi o resultado alcançado.

4.1 Descrições dos Experimentos

As tecnologias utilizadas no projeto serão o HTML5 e o Flash. O desenvolvimento das páginas HTML será através do software NetBeans IDE 7.3.1 (netbeans.org), que nesta versão trás suporte a HTML5, já o desenvolvimento dos recursos em Flash se dará pela IDE Adobe Flash CS5 Professional versão de avaliação (adobe.com). Ambas as tecnologias serão testadas nos navegadores: Google Chrome, Firefox, Internet Explore e Opera, tanto para a execução quanto para as comparações necessárias. O programa utilizado para a medição e comparação da quantidade de memória necessária a cada processo será o “Monitor de Recursos” do Windows, disponível na opção de desempenho nas ferramentas do sistema.

4.2 Parâmetros de Comparação

As comparações envolvidas serão de utilização de memória, tanto virtual, que refere-se ao espaço no disco rígido reservado para ajudar a armazenar os dados da memória RAM, quanto a física, que é a própria memória RAM utilizada pelo computador, dando foco à quantidade de memória reservada pelo sistema operacional para o processo, com isso pode-se analisar o custo de memória utilizado e qual a melhor escolha para uma implementação mais adequada, sendo que, uma utilização de pouco espaço na memória, irá refletir em um processamento mais rápido, também serão avaliados o tamanho das páginas geradas, para que seja analisado o custo de armazenamento e quantidade de código, que é necessário para a construção dos projetos propostos.

Serão construídas aplicações experimentais que envolvam imagens simples, animação, áudio, vídeo e uma animação com interatividade referente à criação de um jogo árcade simples que pode ser executado em uma página da Web.

4.3 Procedimentos

Esta seção apresenta os experimentos realizados, bem como os resultados alcançados com tais experimentos. Para cada experimento realizado há uma gráfico com os resultados obtidos com a comparação de memória. São cinco categorias propostas: Imagem, Animação, Áudio, Vídeo e Animação com interatividade, o navegador Google Chrome foi o primeiro a ser testado, assim, as imagens de ilustração deste projeto são da execução nesse navegador, quanto aos demais navegadores será mostrado somente os resultados.

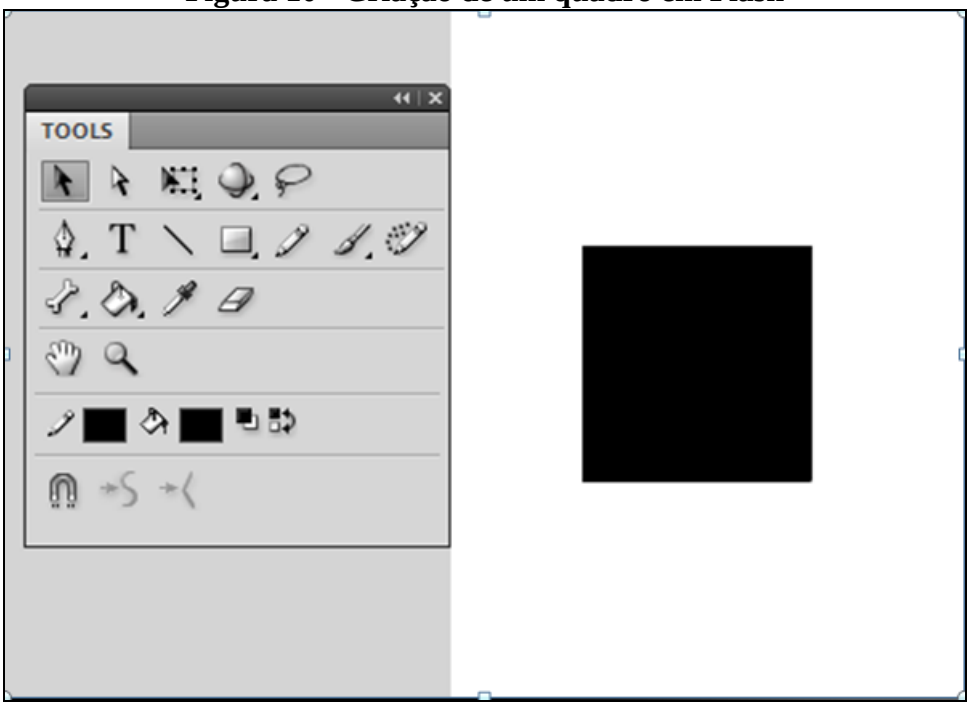
4.3.1 *Quadro*

O estudo começa com o desenvolvimento de um simples quadro desenhado na janela do navegador, com o tamanho de 150px de altura por 150px de largura.

4.3.1.1 Flash

No flash, utilizando o programa Adobe Flash CS5, foi gerado um arquivo swf (extensão para arquivos em flash) com o tamanho de 587 Bytes, que foi introduzido em uma página HTML simples através da tag <*embed*>, o quadro foi de simples desenvolvimento e não gerou muito código, pois foi utilizada a interface gráfica que facilita a criação de imagens e gráficos para páginas Web. A interface de criação do quadro em Flash é demonstrada na Figura 10 e o código de inserção do arquivo swf na página HTML é mostrado na Figura 11.

Figura 10 - Criação de um quadro em Flash



Fonte: Elaborado pelo autor

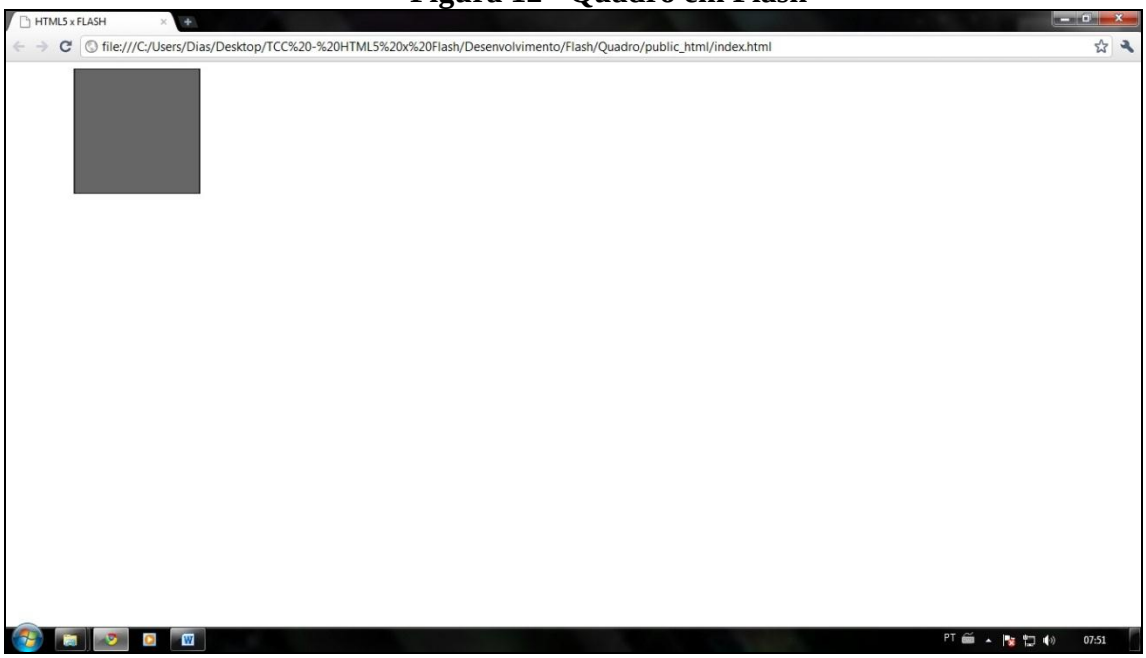
Figura 11 - Código em HTML com inserção de um arquivo em Flash

```
<!DOCTYPE html>
<html>
    <head>
        <title>HTML5 x FLASH</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    </head>
    <body>
        <div><embed src="quadro.swf"> </embed></div>

    </body>
</html>
```

Fonte: Elaborado pelo autor

Quando o código da Figura 11 é executado, na janela do navegador é mostrado um quadro como demonstrado na Figura 12.

Figura 12 - Quadro em Flash



Fonte: Elaborado pelo autor

4.3.1.2 HTML5

No HTML, a criação do quadro foi desenvolvida em uma tag nativa do próprio HTML5 chamada <*canvas*>, primeiramente é necessária a criação de uma função na tag <*script*>. Nesta função é definido um retângulo com o método (context.fillRect), como demonstrado na Figura 13 a seguir, onde os dois primeiros parâmetros da função "context.fillRect" fazem referência a posição onde o retângulo será exibido, a terceira e quarta coordenada fazem referência ao tamanha da figura.

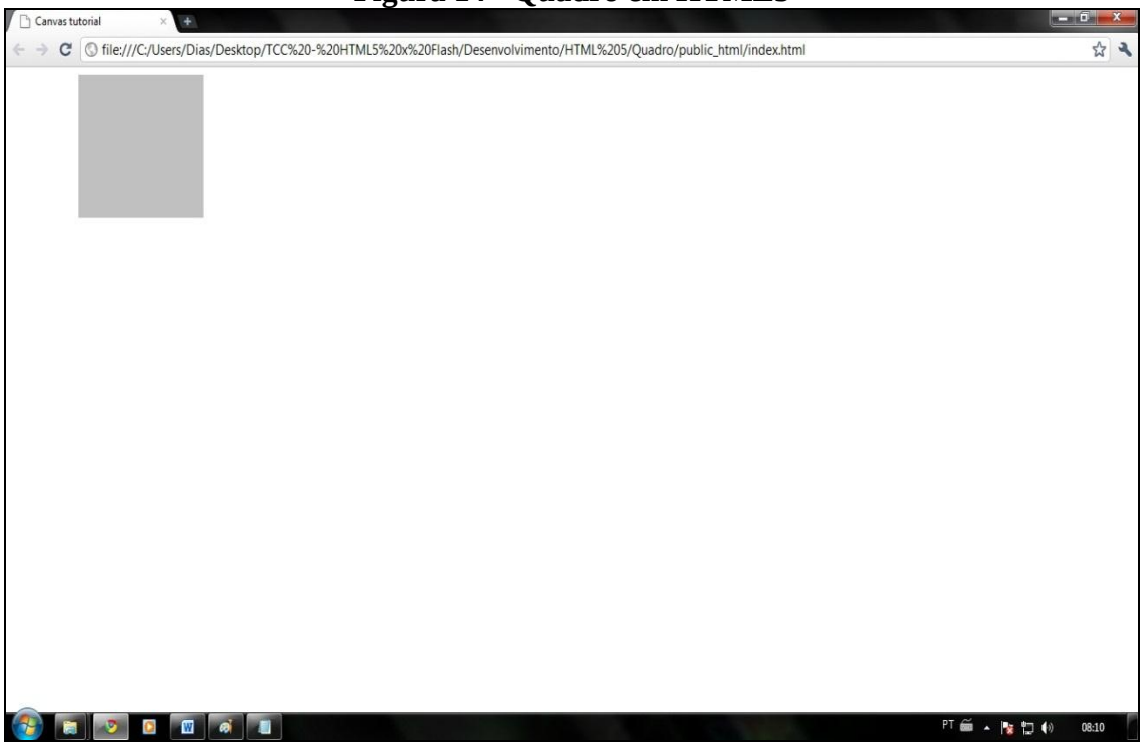
Figura 13 - Código em HTML5 para inserção de um quadro na página

```
<!DOCTYPE html>
<html>
    <head>
        <title>Canvas tutorial</title>

        <script type="text/javascript">
            function draw(){
                var canvas = document.getElementById('icon');
                if (canvas.getContext){
                    var context = canvas.getContext('2d');

                    //retangulo
                    context.fillStyle = "silver";//cor do preenchimento
                    context.fillRect (80, 0, 150, 150);//background


                }
            }
        </script>
    </head>
    <body onload="draw();">
        <canvas id="icon" width="240" height="240"></canvas>
    </body>
</html>
```

Fonte: Elaborado pelo autor

Quando o código da Figura 13 é executado, na janela do navegador é mostrado um quadro como demonstrado na Figura 14.

Figura 14 - Quadro em HTML5



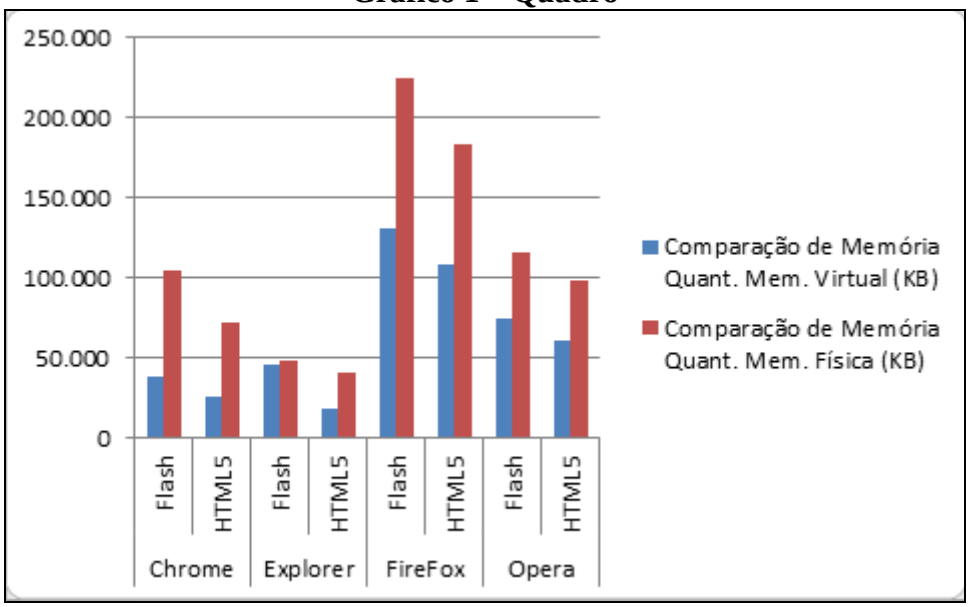
Fonte: Elaborado pelo autor

4.3.1.3 Comparações Básicas e Resultados

Como pode ser observado para o desenvolvimento em HTML5 é necessário um maior conhecimento ao ser aplicado o seu código, pois o elemento Canvas trabalha com JavaScript, entretanto ele gasta menos espaço de armazenamento. O HTML5 apresentou maior quantidade de código sendo que, para a criação do quadro é utilizada uma função e uma sequência de códigos para definir a cor e a forma do objeto. Em Flash, através de sua interface, o arquivo que representa o quadro não gerou código, uma vez que não há interação com a imagem.

Com relação ao tamanho dos projetos, o desenvolvimento em flash apresentou um maior tamanho possuindo 1,65KB, sendo que em HTML5 o projeto obteve o tamanho de 1,27KB. O Gráfico 1 mostra os resultados obtidos com relação aos testes de comparação de memória.

Gráfico 1 – Quadro



Fonte: Elaborado pelo autor

É importante observar que quando é aberto um novo processo para a página em flash o sistema adiciona também o processo do plug-in que adiciona valores ao processo principal. De acordo com o Gráfico 1, o navegador que apresentou o menor consumo de memória foi o Explorer, e o navegador que apresentou o maior consumo de memória foi o Firefox.

4.3.2 *Animação*

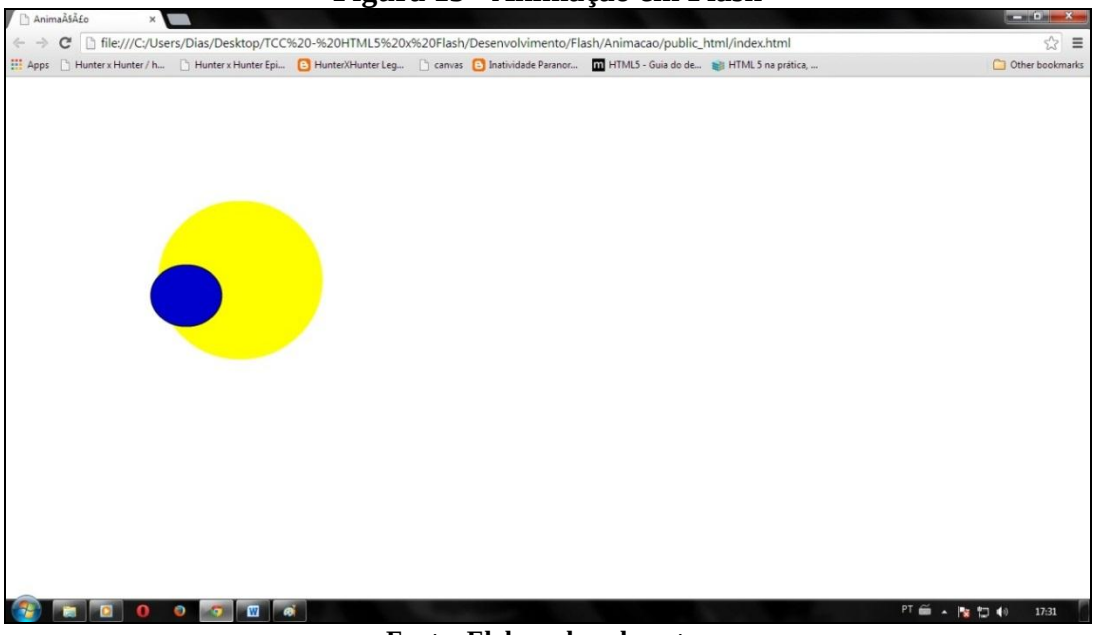
Para o estudo de animações, foram desenvolvidas em ambas as tecnologias uma animação de uma simples representação do sistema solar tendo o planeta terra girando em torno do sol.

4.3.2.1 Flash

No flash, utilizando o programa Adobe Flash CS5, foi gerado um arquivo swf com o tamanho de 987 Bytes, que foi introduzido em uma página HTML simples através da tag *<embed>*, o processo de desenvolvimento desta animação foi feito em duas camadas que funcionam com uma interpolação clássica (recurso do flash para dar movimento a animações), sendo que o caminho percorrido pela bola azul foi traçado em uma camada distinta que tem a função de linha guia, para que a animação ganhe movimento. O código para a inserção da animação em uma página HTML é feito da mesma maneira que a apresentada no exemplo anterior da Figura 11.

A execução do código, com o arquivo flash inserido, mostra uma simples animação de representação do sistema solar demostrada na Figura 15.

Figura 15 - Animação em Flash



Fonte: Elaborado pelo autor

4.3.2.2 HTML5

Em HTML5, para a criação de animações é utilizado o elemento <*canvas*>, que foi explicado na seção 4.2.1. Por se tratar de um código extenso, é mostrada apenas uma parte do código da animação proposta na Figura 16.

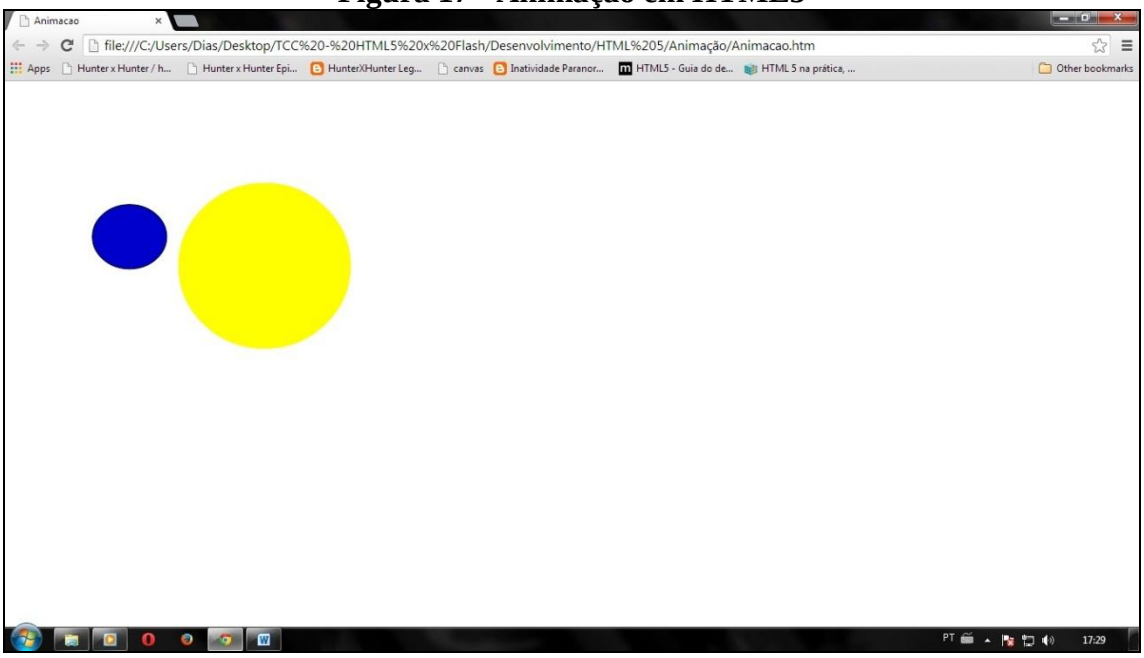
Figura 16 - Código da Animação em Flash

```
<script>
    window.onload = function(){
        var canvas = document.getElementById("exemplo"),
        c = canvas.getContext("2d");


        var posX = 5;
        setInterval(function(){
            posX +=1;

            c.fillStyle = "black";
            c.fillRect(posX, 10, 150, 150);
        },30);

        }
</script>
```

Fonte: Elaborado pelo autor

É importante observar que os parâmetros do objeto “fillRect” foram trocados por variáveis, que por sua vez, são alteradas de acordo com a estrutura de repetição em que se encontra. Os valores numéricos são entendidos pelo navegador como pixel da tela, isso faz com que o objeto ganhe movimento durante a execução da função “setInterval”, que serve para repetir as instruções contidas em sua estrutura em milissegundos. A animação em HTML5 será demostrada na Figura 17.

Figura 17 - Animação em HTML5



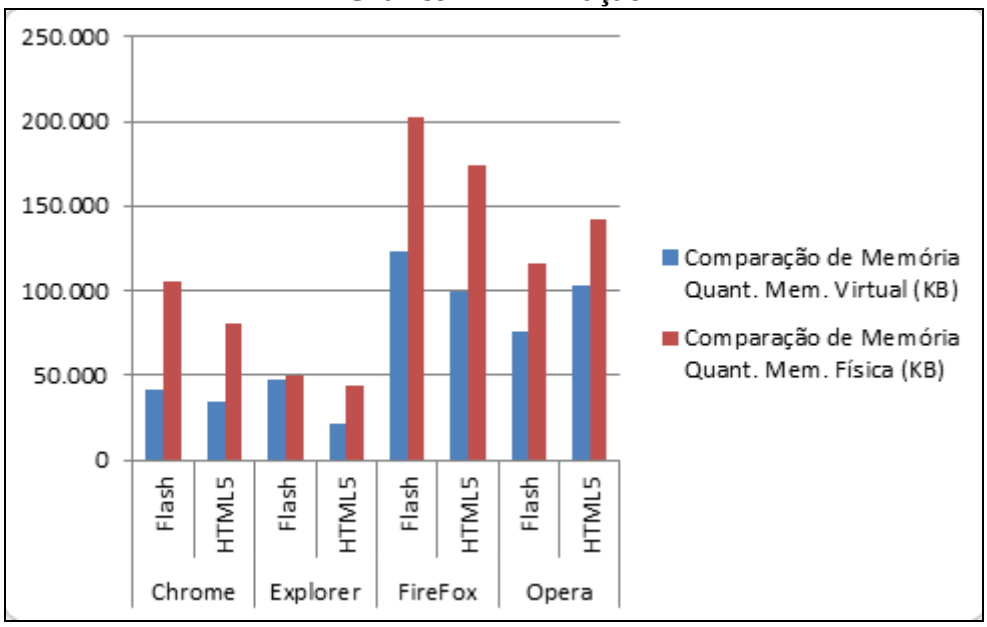
Fonte: Elaborado pelo autor

4.3.2.3 Comparações Básicas e Resultados

Foi observado que a animação proposta gerou uma quantidade excessiva de código no HTML5 em relação ao Flash, e em contra partida foi de simples desenvolvimento no Flash devido a sua interface.

Com relação ao tamanho dos projetos, o desenvolvimento em flash apresentou um menor tamanho, chagando a possuir 2,30KB, sendo que em HTML5 o projeto obteve o tamanho de 2,83KB. O Gráfico 2 mostra os resultados obtidos com relação aos testes de comparação de memória.

Gráfico 2 – Animação



Fonte: Elaborado pelo autor

Foi observado que o navegador Firefox obteve uma taxa elevada de consumo de memória em relação aos demais navegadores, e que, o navegador Explorer obteve as menores taxas de consumo de memória.

4.3.3 *Áudio*

Para as comparações de áudio foi utilizado um arquivo Mp3 com o tamanho de 7,36 MB, introduzido em uma página em branco para análise em ambas as tecnologias.

4.3.3.1 Flash

Para que seja possível inserir um arquivo mp3 em um site construído em HTML simples, é necessário um *plug-in* do flash instalado no computador para que a página possa executar o áudio esperado. Com o *plug-in* instalado, foi inserido o arquivo Mp3 em uma página em branco através da tag *<embed>*, com o conteúdo do *"src"* (*source*, endereço do arquivo no computador), para a reprodução do áudio. O código utilizado neste estudo está demonstrado na Figura 18.

Figura 18 - Código de Áudio em Flash

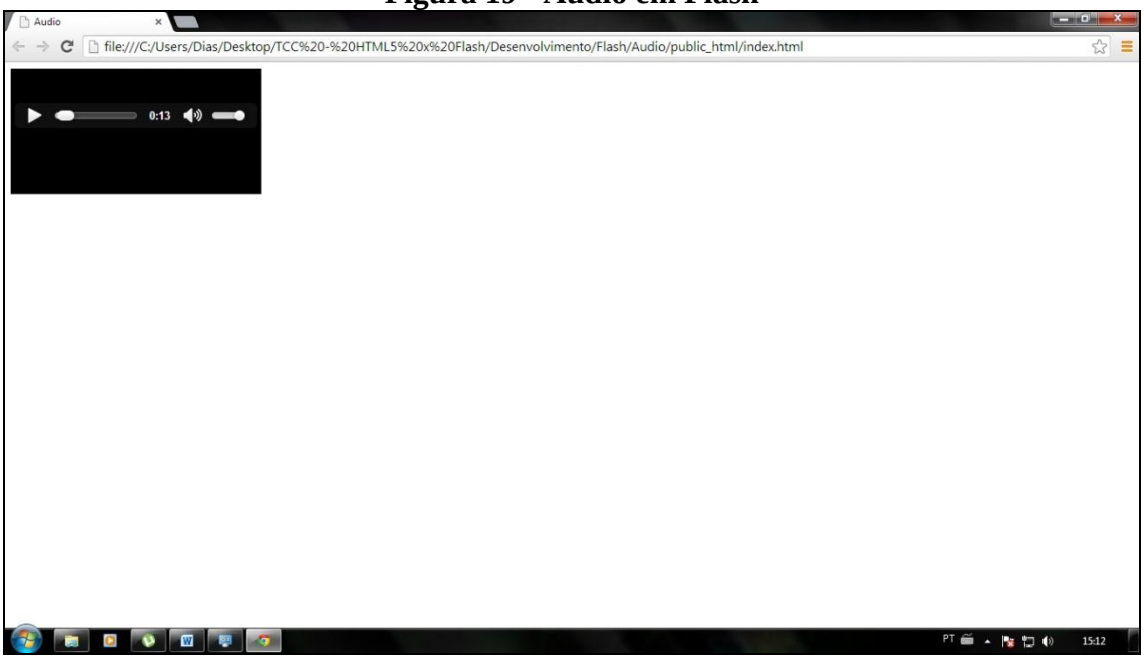
```
<!--
To change this template, choose Tools | Templates
and open the template in the editor.
-->
<!DOCTYPE html>
<html>
    <head>
        <title>Audio</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    </head>
    <body>
        <embed src="Roadhouse Blues.mp3">
    </body>
</html>
```

Fonte: Elaborado pelo autor

Pode-se notar que, por padrão, o flash dimensiona a interface do áudio, fazendo com que exista um espaço que não pode ser utilizado no restante da página.

Como resultado do código da Figura 18 acima, foi obtido a interface padrão demostrada na Figura 19.

Figura 19 - Áudio em Flash



Fonte: Elaborado pelo autor

O áudio, por padrão, está em modo de auto reprodução, com ele estão disponíveis duas opções de navegação, a opção e pausar/tocar e a opção de volume.

4.3.3.2 HTML5

Em HTML5, para que seja inserido um arquivo de áudio em uma página, é utilizado um novo elemento específico desta versão do HTML, a tag <*audio*>, responsável pela reprodução de áudio, possui o mesmo atributo "src" utilizado no flash para endereçamento do arquivo de áudio, este elemento possui alguns atributos que possibilita ao desenvolvedor dimensionar e editar a forma com que o áudio é exibido na página, podendo inclusive, alterar o plano de fundo do áudio. O código utilizado nesta demonstração está na Figura 20 a seguir.

Figura 20 - Código de Áudio em HTML5

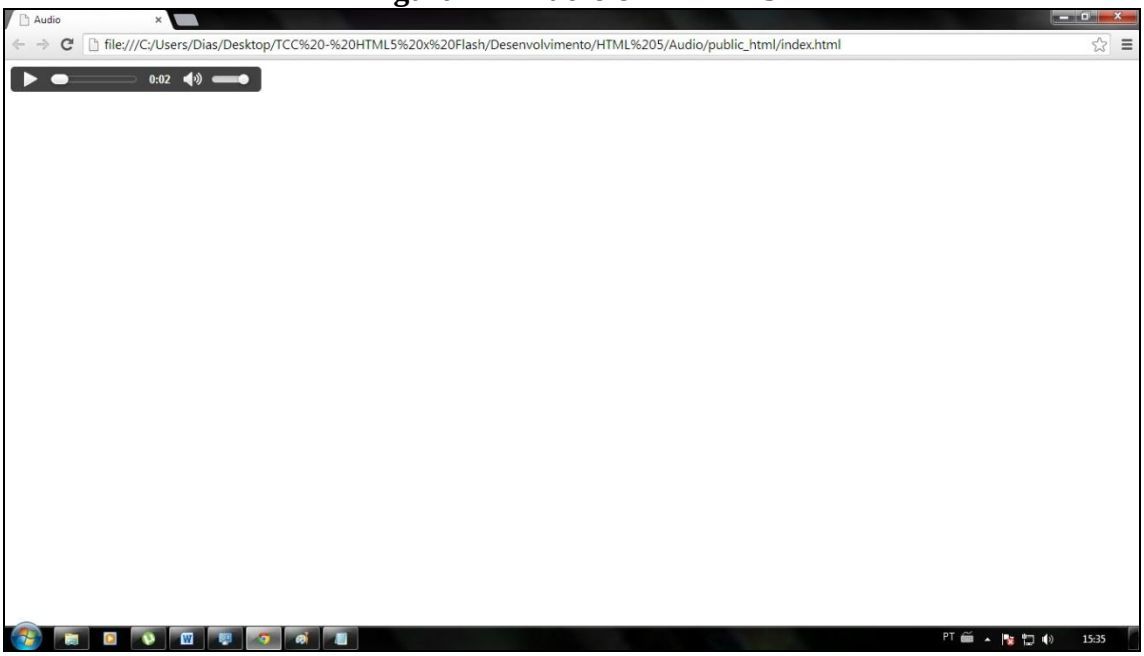
```
<!--
To change this template, choose Tools | Templates
and open the template in the editor.
-->
<!DOCTYPE html>
<html>
    <head>
        <title>Audio</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    </head>
    <body>
        <audio src="Roadhouse Blues.mp3" preload="auto" controls> </audio>
    </body>
</html>
```

Fonte: Elaborado pelo autor

Pode-se observar que, por padrão, a tag <áudio> do HTML5 possui os atributos "preload", para indicar se a reprodução do áudio será automática ou se será necessário à intervenção do usuário para a inicialização do áudio, e o atributo "controls", para a exibição dos controles do player, dando assim, a interação com o usuário.

Como resultado do código da Figura 20 acima, é mostrada a interface na Figura 21.

Figura 21 - Áudio em HTML5



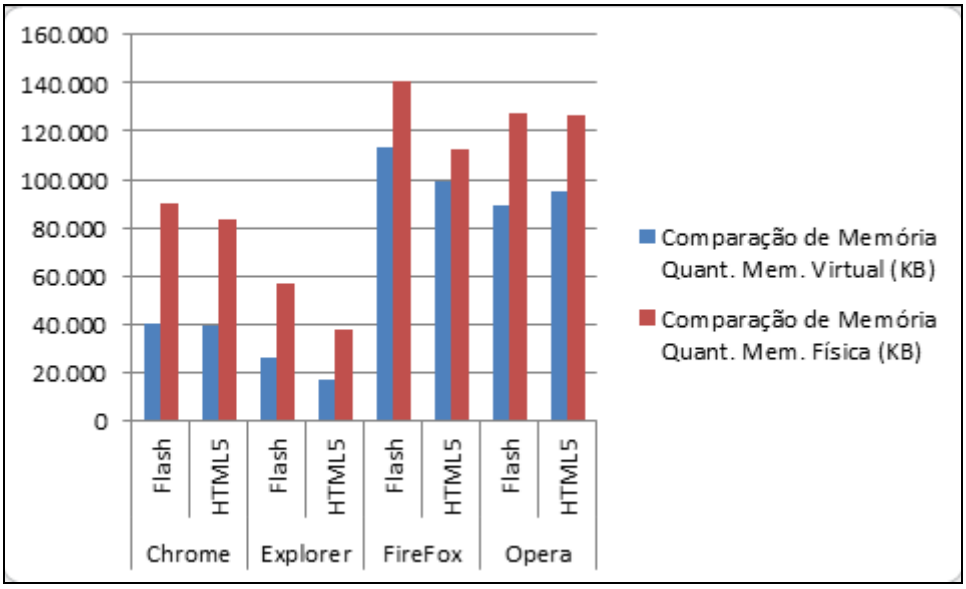
Fonte: Elaborado pelo autor

4.3.3.3 Comparações Básicas e Resultados

Como pode ser observado para o desenvolvimento em HTML5, foi implementado uma tag específica que possibilita a edição e visualização dos controles de áudio em uma página, o que enriquece o conteúdo e aumenta a interoperabilidade. Segundo o portal Adobe (2013), os formatos de arquivos de som suportados pelo Flash são: Adobe Sound (.asnd), Wave (.wav), AIFF (.aif, .aifc), Mp3, Sound Designer II (.sd2), Sun Au (.au, .snd), FLAC (.flac) e Ogg Vorbis (.ogg, .oga). Em HTML5, segundo Eis (2010), os formatos de áudio suportados são: Ogg Vorbis, Mp3 e Wav.

Com relação ao tamanho dos projetos, o desenvolvimento em flash apresentou o tamanho de 1,36KB e o HTML5 apresentou o tamanho de 1,39KB. O Gráfico 3 a seguir mostra os resultados obtidos com relação aos testes de comparação de memória.

Gráfico 3 – Áudio



Fonte: Elaborado pelo autor

Foi observado que o navegador Firefox, juntamente com Opera, obtiveram uma taxa elevada de consumo de memória em relação aos demais navegadores, por outro lado, o navegador Explorer obteve as menores taxas de consumo de memória.

4.3.4 *Vídeo*

Para as comparações de vídeo foi utilizado um arquivo Mp4 com o tamanho de 22,1 MB, introduzido em uma página em branco para análise em ambas as tecnologias.

4.3.4.1 Flash

Foi construída uma página HTML em branco e adicionado a ela um vídeo de 22,1MB com o nome de “linux.mp4”, lembrando que para reprodução de áudio e vídeo, é necessário ter instalado o plug-in do Flash Player. O arquivo de vídeo pode ser introduzido na página através da tag <*embed*>, com o conteúdo do ”*src”* (*source*, endereço do arquivo no computador), para a reprodução do vídeo. O código utilizado neste estudo está demonstrado na Figura 22.

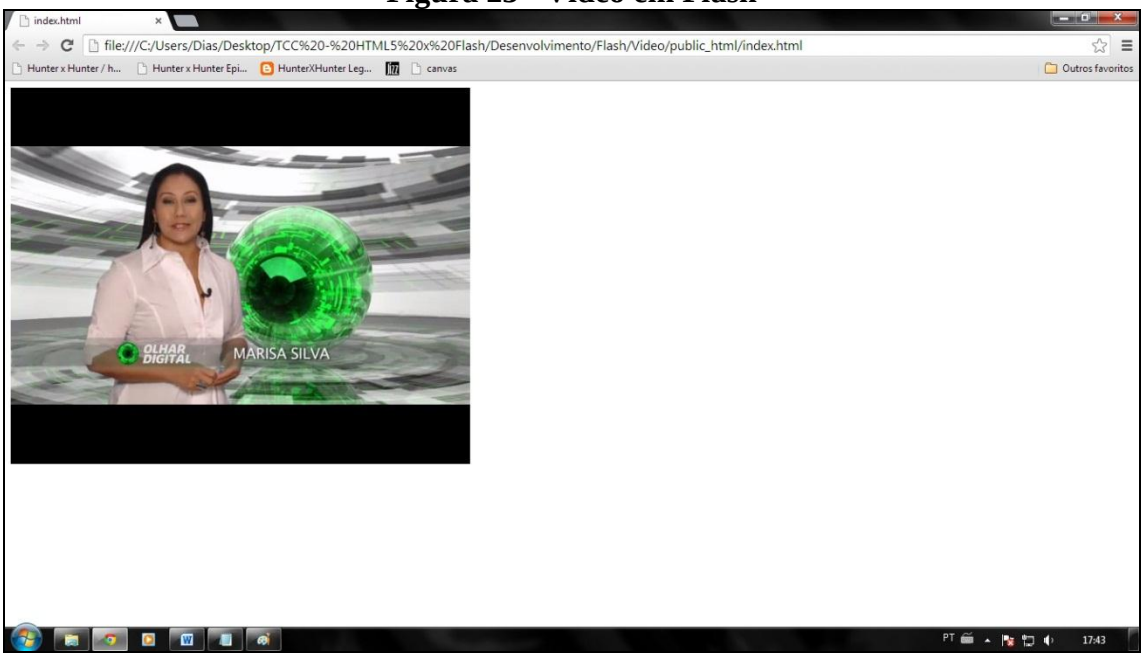
Figura 22 - Código de Vídeo em Flash

```
<!--
To change this template, choose Tools | Templates
and open the template in the editor.
-->
<!DOCTYPE html>
<html>
    <head>
        <title></title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    </head>
    <body>
        <embed src="linux.mp4" width="550" height="450">
    </body>
</html>
```

Fonte: Elaborado pelo autor

O resultado da execução do código da Figura 22 será mostrado na Figura 23.

Figura 23 - Vídeo em Flash



Fonte: Elaborado pelo autor

Dentro da tag *<embed>*, o programador pode aumentar ou diminuir o tamanho do vídeo livremente pelos atributos “width” e “height”. Por padrão, o vídeo está em modo automático, assim que a página é aberta o vídeo é iniciado, também, por padrão, os controles do vídeo ficam ocultados, aparecendo quando o usuário move o mouse pelo vídeo. Os controles do vídeo são: play/pause, volume e aumento de tela.

4.3.4.2 HTML5

Em HTML5, para que seja inserido um vídeo na página é utilizado o novo elemento chamado de <*vídeo*>. Para este estudo foi inserido um vídeo de 22,1MB em uma página em branco do HTML5, como mostra a Figura 24.

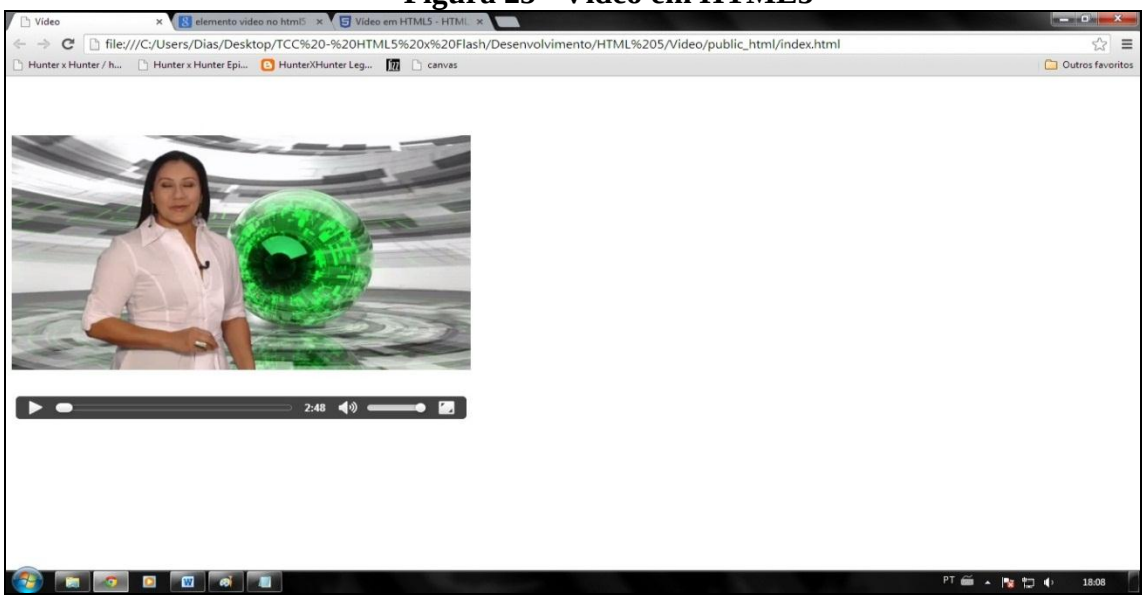
Figura 24 - Código de Vídeo em HTML5

```
<!--
To change this template, choose Tools | Templates
and open the template in the editor.
-->
<!DOCTYPE html>
<html>
    <head>
        <title>Vídeo</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    </head>
    <body>
        <video width="550" height="450" src="linux.mp4" controls>

        </video>
    </body>
</html>
```

Fonte: Elaborado pelo autor

Como resultado do código da Figura 24, foi obtida a página demostrada na Figura 25.

Figura 25 - Vídeo em HTML5



Fonte: Elaborado pelo autor

Pode-se observar que, diferentemente do estudo realizado com o Flash, o vídeo não inicia juntamente com a página, é esperado a interação do usuário para que o vídeo comece.

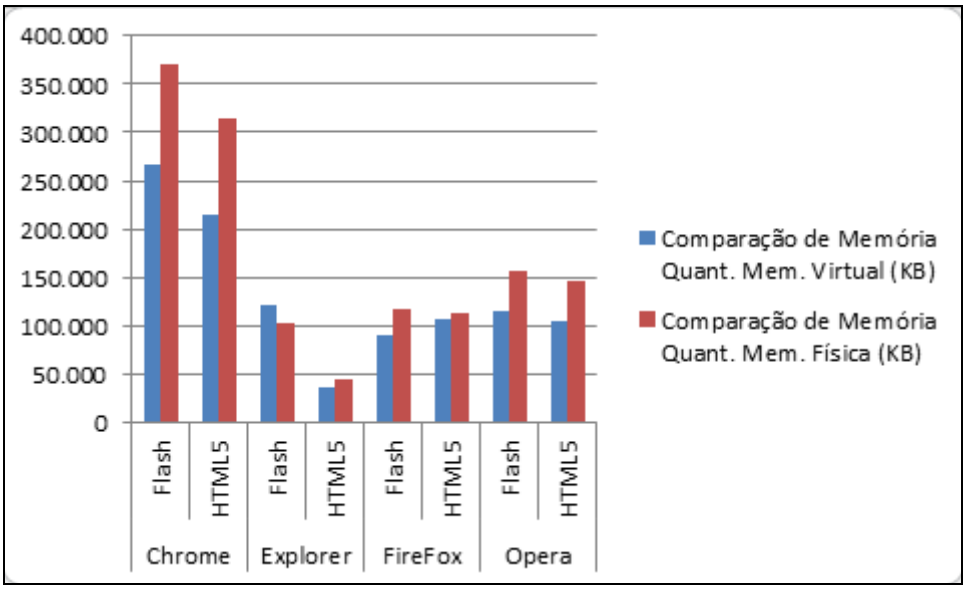
Os controles estão posicionados abaixo do vídeo e possuem os mesmos comandos de play/pause, volume e aumento da tela.

4.3.4.3 Comparações Básicas e Resultados

A principal diferença observada neste estudo são os controles do vídeo e seu posicionamento, foi observado que no flash os controles se ocultam e quando exibidos eles estão posicionados em cima do vídeo, podendo perder um pouco das imagens que estão sendo exibidas, em contrapartida, no HTML5 os controles de vídeo estão abaixo do vídeo e com um espaçamento que traz uma melhor visualização. A tag <*video*> é geralmente apresentada como uma alternativa ao Flash na mídia, sua principal vantagem é a integração natural com as outras camadas da pilha de desenvolvimento da web, como, por exemplo, CSS e JavaScript, bem como as outras *tags* HTML. Segundo o portal Adobe (2013), os formatos de arquivos de vídeo suportados pelo Flash são: Mp3, Mp4, M4V, F4V, 3GPP, MKV, AVI, FLV e WMV. Em HTML5[,] segundo Delgado (2010), os formatos de vídeo suportados são: Ogg Vorbis, Mp4 e Webm.

Com relação ao tamanho dos projetos, o desenvolvimento em flash apresentou o tamanho de 1,37KB e o HTML5 apresentaram o tamanho de 1,42KB. O Gráfico 4 mostra os resultados obtidos com relação aos testes de comparação de memória.

Gráfico 4 – Vídeo



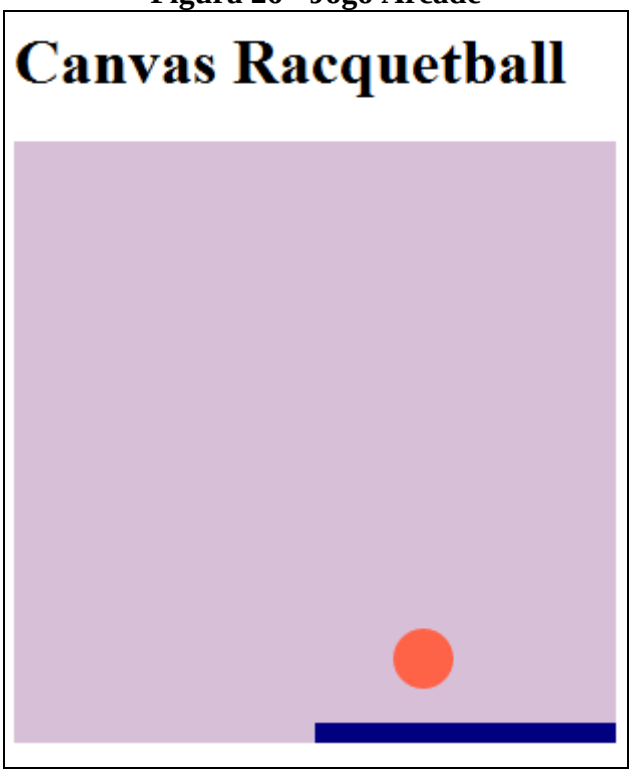
Fonte: Elaborado pelo autor

Foi observado que, o navegador Chrome obteve uma taxa elevada de consumo de memória em relação aos demais navegadores, e que, o navegador Explorer obteve as menores taxas de consumo de memória.

4.3.5 *Animação com Interatividade*

Esta seção compara como usar a Canvas e Flash para criar um jogo árcade simples que pode ser executado em uma página da Web. O objetivo do jogo é acertar uma bola com uma raquete de forma que ela atravesse a quadra até o outro lado. A Figura 26 a seguir mostra como é o jogo proposto.

Figura 26 - Jogo Árcade



Fonte: Elaborado pelo autor

4.3.5.1 FLASH

De acordo com Adobe (2013), o painel *Snippets* de código foi programado para facilitar o uso simples do ActionScript 3.0 rapidamente por usuários que não sejam programadores. Ele permite adicionar código ActionScript 3.0 ao arquivo FLA a fim de ativar a funcionalidade comum. Usar o painel *Snippets* de código não exige conhecimentos do ActionScript 3.0.

O exemplo a seguir explica os princípios básicos de movimento, da bola e da raquete, de criação da ação, de detecção das colisões e de término do jogo. O código de exemplo das Figuras 27 e 28 apresentam o jogo em Flash e mostra como ActionScript usa pixels em uma determinada variável para a movimentação da bola, da raquete e para testes de colisão com as extremidades do palco e da raquete.

Figura 27 - Código em Flash do Jogo Árcade 1 - Continua

```
var leftPressed:Boolean = false;
var rightPressed:Boolean = false;
var velocidadeX:Number = 5;
var velocidadeY:Number = 5;
var str:String = "Game Over!";
player.addEventListener(Event.ENTER_FRAME, fl_MoveInDirectionOfKey);
stage.addEventListener(KeyboardEvent.KEY_DOWN, fl_SetKeyPressed);
stage.addEventListener(KeyboardEvent.KEY_UP, fl_UnsetKeyPressed);
function fl_MoveInDirectionOfKey(event:Event)
{
   if (leftPressed)
    {
        player.x -= 5;
    }
    if (rightPressed)
    {
       player.x += 5;
    }
    bola.x += velocidadeX;
    bola.y += velocidadeY;
    if(bola.x+bola.width/2 >= stage.stageWidth || bola.x - bola.width/2 < 0){
       velocidadeX *= -1;
       }
    if(bola.y+bola.height/2 >= stage.stageHeight || bola.y - bola.height/2 < 0){
       velocidadeY *= -1;
       }
```

Fonte: Elaborado pelo autor

Figura 28 - Código em Flash do Jogo Árcade 2 - Final

```
    if(bola.hitTestObject(player)){
       velocidadeY *= -1;
       }
    if(bola.hitTestObject(lim)){
       game_over.text = str;
       velocidadeX = 0;
       velocidadeY = 0;
       }
```

Fonte: Elaborado pelo autor

As variáveis globais são declaradas. A função fl_MoveIndirectoronKey, quando chamada, realiza os movimentos da raquete pelas teclas seta para direita e seta para esquerda, e o movimento da bola. As colisões são feitas através de *ifs*, verificando colisões com as

laterais, o topo e a raquete. Por fim é feita uma colisão com o objeto *lim*, se a bola colidir com o limite, a mensagem de “Game Over!” é chamada por um tipo texto e a jogo pausa.

4.3.5.2 HTML5

Esse exemplo também desenvolvidos em HTML5 e JavaScript. Ele explica os mesmos princípios utilizado na criação em Flash. O código de exemplo das Figuras 29 e 30 apresentam o jogo em Canvas e mostra como Canvas usa pixels no *immediate mode* (Estilo de interface de programação de aplicativos de bibliotecas gráficas), para desenhar a quadra e a raquete com retângulos e a bola com um arc. Em seguida, o código descreve como colorir a quadra, a raquete e a bola.

Figura 29 - Código em HTML5 do Jogo Árcade 1 - Continua

```
1   function drawGameCanvas(){
2           // Obtendo o elemento canvas.
3           canvas = document.getElementById("gameBoard");
4
5           // Certificando o a escolha de contexto.
6           if (canvas.getContext) {
7             // Especificando o context em 2d.
8             ctx = canvas.getContext("2d");
9
10            // Repetir o jogo em intervalos de 16 milisegundos.
11            gameLoop = setInterval(drawBall, 16);
12
13            // Adicionar sensibilidade de teclas.
14            window.addEventListener('keydown', whatKey, true);
15
16          }
```

Fonte: Elaborado pelo autor

Figura 30 - Código em HTML5 do Jogo Árcade 2 - Final

```
17  function drawBall() { // Função para desenhar a bola
18
19          // Limpar a quadra.
20          ctx.clearRect(0, 0, boardX, boardY);
21
22          // Encher a quadra.
23          ctx.fillStyle = "thistle";
24          ctx.beginPath();
25          ctx.rect(0, 0, boardX, boardY);
26          ctx.closePath();
27          ctx.fill();
28
29          // Desenhar a dimensão da bola e definir sua posição.
30          ctx.fillStyle = "tomato";
31          ctx.beginPath();
32          ctx.arc(ballX, ballY, 15, 0, Math.PI * 2, true);
33          ctx.closePath();
34          ctx.fill();
35
36          // Desenhar a raquete e definer sua posição.
37          ctx.fillStyle = "navy";
38          ctx.beginPath();
39          ctx.rect(paddleX, paddleD, paddleW, paddleH);
40          ctx.closePath();
41          ctx.fill();
42
43          // Mudar a bola de posição.
44          ballX += ballDX;
45          ballY += ballDY;
```

Fonte: Elaborado pelo autor

A função drawball é chamada por um temporizador de evento a cada 16 milissegundos. As tarefas a seguir são executadas sempre que essa função é chamada:

- A quadra do jogo é esvaziada e preenchida. clearRect é chamado para apagar a tela e, em seguida, uma cor é especificada. Um caminho retangular é definido e preenchido;
- A bola é desenhada em sua posição atual especificando fillStyle, beginPath, arc, closePath e fill. A posição da bola é definida por ballX e ballY e mudará com base nas colisões com as paredes e a raquete;
- Em seguida, a raquete é desenhada da mesma maneira, mas com as especificações de paddleX, paddleD, paddleW e paddleH.

4.3.5.3 Comparações Básicas e Resultados

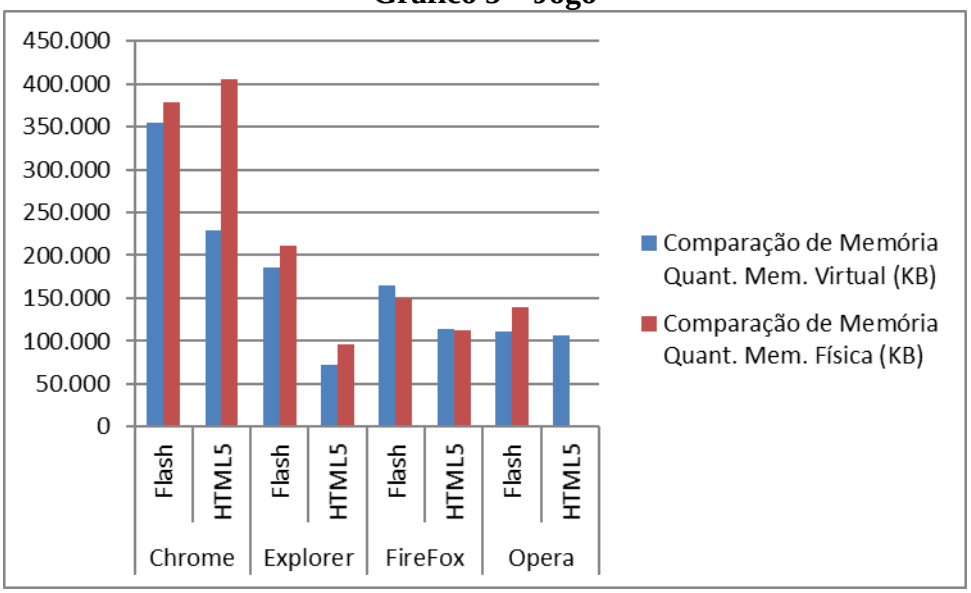
Pode-se observar, no desenvolvimento do jogo, a diferença de implementação dos objetos e das ações. O desenvolvimento em Flash (ActionScript 3.0) é bem prático, pois pode-se desenhar os objetos através de modelos pré-definidos no ambiente de desenvolvimento, como círculos, quadrados e demais figuras geométricas. Em HTML5 o desenvolvimento dos objetos são implementados através de funções e elementos da linguagem de marcação, o que requer um conhecimento prévio da linguagem.

O desenvolvimento das ações dos objetos em Flash são desenvolvidos através do ActionScript e pode-se implementar através de *Snippets* de código, que já provê ações para objetos pré-desenvolvidos ou através do *Script*. Em contrapartida, as ações e eventos para objetos em HTML5 são desenvolvidos através de funções e elementos.

O Flash tem uma facilidade de se desenvolver maior em relação ao HTML5, pois necessita de uma quantidade menor de código para a criação do jogo, já em HTML5, a quantidade de código gerado é consideravelmente maior por se desenvolver toda a estrutura através de códigos.

Com relação ao tamanho dos projetos, o desenvolvimento em Flash incorporado na página, teve 1KB. Em HTML5 chegou ao tamanho de 4KB. O Gráfico 5 mostra os resultados obtidos com relação aos testes de comparação de memória.

Gráfico 5 – Jogo



Fonte: Elaborado pelo autor.

Foi observado que, o navegador Chrome novamente obteve uma taxa elevada de consumo de memória em relação aos demais navegadores, e que, o navegador Explorer obteve as menores taxas de consumo de memória virtual e o navegador Opera obteve as menores taxas de consumo de memória física.

5 CONCLUSÕES

O objetivo desta pesquisa foi realizar um estudo analítico sobre a capacidade do HTML5 em tornar obsoleto o Adobe Flash Player, assim, foi realizado múltiplos testes que descreveram o funcionamento e características específicas de cada tecnologia estudada.

Os testes realizados comprovaram que, por necessitar de um *plug-in*, o Flash utiliza mais memória que o HTML5, já o custo de aprendizado para utilizar o HTML5 é superior ao Flash, vendo-se que, para criar animações e imagens pelo elemento canvas é utilizado o JavaScript, que é uma outra linguagem distinta do HTML5, já em Flash é utilizado a IDE do Adobe Flash CS, que utiliza sua linguagem própria, o ActionScript.

Percebeu-se uma grande diferença na quantidade de código gerado, uma vez que o HTML5 produz uma maior quantidade de código, devido a manipulação do elemento canvas que é feita através do JavaScript, possuindo uma sintaxe mais complexa em relação ao ActionScript 3.0 do Flash.

Foi observado, também, que o tamanho dos projetos em HTML5 apresentou um maior volume de código em alguns casos devido à quantidade necessária para criação do projeto, chegando a possuir, em HTML5 1,47KB contra 1,36KB em Flash.

Com relação à animação com interatividade, percebeu-se uma complexidade maior de desenvolvimento na estrutura do HTML5, pelo fato de gerar mais código em relação ao Flash. O Flash, por sua vez, possui uma excelente IDE para o desenvolvimento de animações e imagens, tornando a construção de projetos gráficos mais intuitivos.

Apesar do HTML5 possuir um consumo de memória menor e um menor tamanho de armazenamento de páginas em relação ao Flash, o HTML5 apresentou uma maior quantidade de código e maior complexidade de desenvolvimento, assim, conclui-se que, por não possuir uma IDE de desenvolvimento para o elemento Canvas, o HTML5 ainda não pode tornar o Flash obsoleto.

REFERÊNCIAS

ADOBE. **Flash Player**. Disponível em: <http://www.adobe.com/br/software/flash/about/>. Acesso em 02 de Abril de 2013.

ADOBE. **Flash Player Noções Básicas sobre Formato de Vídeo**. Disponível em: <http://help.adobe.com/pt_BR/ActionScript/3.0_ProgrammingAS3/WS5b3ccc516d4fbf351e63e3d118a9b90204-7d46.html>. Acesso em 11 de Outubro de 2013.

ADOBE. **Flash Player Professional, Uso de sons no Flash**. Disponível em: <http://help.adobe.com/pt_BR/flash/cs/using/WSd60f23110762d6b883b18f10cb1fe1af6-7ce8a.html>. Acesso em 11 de Outubro de 2013.

ALIGHIERI, Adamo (2009). **Flash, Usabilidade e Web 2.0.** Disponível em: <http://www.aareis.com.br/arquivos_posts/flash_usabilidade_web2.pdf>. Acesso 27 de Maio de 2013.

CRISTINA, Karen (2010). **História e usos da Internet.** Disponível em: <http://www.portalsaofrancisco.com.br/alfa/historia-da-internet>. Acesso em: 08 de abr. 2013.

DELGADO, Ernest (2010), **Vídeo HTML5.** Disponível em: <http://www.html5rocks.com/pt/tutorials/video/basics/>. Acesso em 11 de Outubro de 2013.

EIS, Diego (2010), **HTML5 Elementos de Áudio.** Disponível em: <http://tableless.com.br/elemento-tag-audio/#.UlhcJ9Ksim4>. Acesso em 11 de Outubro de 2013.

FERREIRA, Elcio; EIS, Diego. **HTML5 Curso W3C Escritório Brasil.** Disponível em: <http://www.w3c.br/pub/Cursos/CursoHTML5/html5-web.pdf>. Acesso em: 10 de Abril de 2013.

FRANKLIN, Alysson. **Tenha o DOM.** Disponível em: <http://tableless.com.br/tenha-o-dom/#.UbfPyefU-1d>. Acesso em 20 de Abril de 2013.

FUVERKI, Daniel (2010). **Estudo de viabilidade do HTML5 para desenvolvimento web.** Disponível em: <http://www.munif.com.br/munif/arquivos/HTML5-FINAL.pdf?id=387>. Acesso em 27 de Maio de 2013.

JAVA. **Qual é a diferença entre o JavaScript e o Java?**. Disponível em: <http://java.com/pt_BR/download/faq/java_javascript.xml>. Acesso em 02 de Abril de 2013.

MAKIS, Jhonny. **Introdução ao HTML5.** Disponível em: <http://jhonnymakis.com.br/blog/2010/09/22/introducao-ao-html-5>. Acesso em 20 de Abril de 2013.

MOZILLA. **Plug-ins: para que servem e como instalar?** Disponível em: <http://br.mozdev.org/firefox/plugin>. Acesso em: 10 de Abril de 2013.

PACIEVITCH, Yuri. **Flash.** Disponível em: <http://www.infoescola.com/informatica/flash/>. Acesso em: 25 de abr. 2013.

PEREIRA, Ana. **O que é CSS.** Disponível em: <http://www.tecmundo.com.br/programacao/2705-o-que-e-css-.htm>. Acesso em 15 de Abril de 2013.

PORTAL GEO BRASIL. **Lamina modelo.** Disponível em: <http://www.portalgeobrasil.org/info/material/funcionamento.pdf>. Acesso em: 08 de abr. 2013.

RIBBENS, Jon (2010), **Desenvolvendo em .Net para iniciantes.** Disponível em: <http://eufacoprogramas.com/desenvolvendo-em-net-para-iniciantes>. Acesso em 11 de Outubro de 2013.

W3C. **1ª Visão Geral do HTML5**. Disponível em: <http://www.w3c.br/cursos/html5/conteudo/capitulo1.html>. Acesso em: 02 de Abril de 2013.

W3C. **Canvas API.** Disponível em: <http://www.w3c.br/cursos/html5/conteudo/capitulo14.html>. Acesso em 13 de Abril de 2013.

W3C. **HTML Design Principles.** Disponível em: <http://www.w3.org/TR/html-design-principles>. Acesso em 13 de Abril de 2013.

W3C. **HTML5.** Disponível em: <http://www.w3.org/TR/html5>. Acesso em 13 de Abril de 2013.