1 CONTEXTUALIZAÇÃO

1.1 JOGOS

1.1.1 Benefícios

São variadas as pesquisas que apontam os benefícios dos jogos eletrônicos para as pessoas, coordenação motora, maior concentração.

Muitos jogos ajudam a desenvolver habilidades práticas, servem como uma forma de exercício, ou de alguma forma executam um papel educacional, de simulação ou psicológico. Jogos são uma parte universal do experiência humana, presentes em todas as culturas.

1.1.2 O Mercado

Com a ascensão dos dispositivos inteligentes a massiva quantidade de dispositivos se tornou atraente para a indústria e consumidores de jogos.

A maior dificuldade em capturar uma base de usuários é que o mercado de dispositivos móveis é muito fragmentado e não existe uma única plataforma popular. (HASAN, 2012)

1.2 JOGOS E MULTIPLATAFORMA

É laboriosa a tarefa dos produtores de software em um panorama tão diversificado como o atual, existem muitas plataformas, muitas versões e hardwares diferenciados.

Uma alterativa para mitigar os problemas oriundos da multiplicidade de plataformas é o HTML.

> Rather than enabling the user to control their own experience, just as the web does, native apps are made to lock users in. They may lock us into a hardware model, meaning we don't have access to all of the functionality we want.
>
> They may make us consistently update our software or upgrade to the newest smartphone to get the latest content. Rather than deliver an innovative experience, native apps limit it.
>
> Standards-based technologies, including HTML5 and CSS3, make it possible for web apps to run on pretty much any platform via a modern, standards-compliant browser. Web apps are adaptive

> and responsive, affording developers with more time to innovate as less time is required to learn new coding skills or 'wrap' an app for delivery in a native environment.

1.3 HTML E MULTIPLATAFORMA

Desenvolvedores de jogos web podem rapidamente satisfazer as necessidades de seus jogadores, mantendo-os leais a tecnologia HTML5 (ZHANG, 2012).

> A maioria dos desenvolvedores demonstra interesse para o HTML5

A OWP (Open Web Platform), uma coleção de tecnologias livres, amplamente utilizadas e padronizadas. O HTML Hypertext Markup Language é a peça fundamental da OWP, seu objetivo é especificar o conteúdo das páginas web. Trata-se de uma linguagem de marcação que define a estrutura de elementos que uma página deve ter de modo a fornecer conteúdo aos usuários.

O HTML5 é um conceito guarda chuva para designar as tecnologias da web JavaScript, CSS3 e HTML. Em termos práticos significa que quando uma tecnologia se torna muito popular através da adoção de algumas grandes empresas e dos desenvolvedores, ela se torna candidata a adoção.

A interatividade necessária para a construção de jogos animados em HTML é algo recente, obtido através dos esforços direcionados no HTML5, anteriormente só se obtinha iteratividade de tamanho proporção na Web com a utilização de ferramentas proprietárias como o Adobe Flash e Microsoft Silverlight.

Entretanto, o HTML em sua especificação e implementações atuais costa com algumas limitações que precisam ser compreendidas por aqueles interessados em criar jogos em HTML5.

> O tempo de desenvolvimento de uma aplicação em HTML5 é 67% menor que aplicações nativas. Isso mostra o custo efetivo de aplicações baseadas em HTML5. A real vantagem de aplicações em HTML5 é o suporte horizontal entre as plataformas - que é a maior razão por trás do custo efetivo. (HASAN et al, 2012)

1.4 LIMITAÇÕES DE JOGOS MULTIPLATAFORMA COM HTML5

O HTML vem sendo desenvolvido por muitos anos e por pessoas que não conheciam umas as outras, muitas funcionalidades foram construídas de maneiras inconsistentes.

> Funcionalidades foram disponibilizadas de diversas fontes e não foram construídas de forma especialmente consistente com as demais. Além disso, devida a única característica da Web, BUGs de implementação se tornam frequentes, e muitas vezes se tornam o padrão, pois outras funcionalidades dependem destas primeiras antes que elas estejam estáveis. (W3C manual)

Enquanto o HTML é desenvolvido muitas das funcionalidades disponibilizadas são testadas em apenas um pequeno conjunto de navegadores para um pequeno conjunto de versões (referência 2). Isso acarreta em suporte inconsistente. A forma mais segura de garantir suporte é testando em todas as versões alvo, todavia essa solução não é prática. (REF 2)

Os desenvolvedores de navegadores podem interpretar/implementar as especificações erroneamente aumentando os problemas de compatibilidade.

Nem todos os recursos disponíveis através das SDK's nativas estão presentes através do HTML5.

1.5 ESTE TRABALHO

Este projeto propõe analisar as limitações do HTML5 quanto relativo a construção de jogos multiplataforma. Através de revisão bibliográfica e da criação de um protótipo de jogo multiplataforma.

Um tratado completo sobre o assunto requiriria um comparativo entre jogos desenvolvidos nativamente e jogos em HTML5.

Não é objetivo deste trabalho demonstrar onde o HTML5 se sobressai, apenas suas limitações. Também não é objetivo deste trabalho comparar o HTML com outras tecnologias de desenvolvimento de jogos, como Flash Player, Silverlight ou alternativas Desktop.

1.5.1 O JOGO

Para a análise prática das limitações foi escolhido um jogo de matemática simples. Consistindo na geração de equações com um candidato de resposta. Cabe ao usuário informar se o resultado apontado pelo jogo está correto ou não.

<!!– Porquê escolhi esse tipo de jogo? –>

2 PROBLEMA

A carência de definições concretas sobre a viabilidade da atual versão do HTML5 - quando utilizado no desenvolvimento de jogos e o senso comum, acabam por monopolizar à construção de jogos nativos as plataformas alvo.

Os custos introduzidos no ciclo vida de um jogo, para diversas plataformas, é muito alto para ser considerado trivial. Cerca de 65% mais altos (segundo trabalho 2)

3 OBJETIVOS

Abaixo seguem os objetivos deste trabalho.

3.1 OBJETIVO GERAL

Identificar possíveis limitações no processo de desenvolvimento de jogos multiplataforma oriundas do atual estado de definição e implementação do HTML5.

3.2 OBJETIVOS ESPECÍFICOS

Estudar as limitações de desenvolvimento de jogos nas plataformas de dispositivos inteligentes Android e navegadores Desktop Google Chrome 42 e Firefox 37. Optamos por Android, e não IOS, pois o primeiro contém a vasta maioria do mercado de dispositivos inteligentes, e por termos maior experiência na já mencionada plataforma.

Pretende-se também estudar os seguintes tópicos do desenvolvimento de jogos, relativos ao HTML5: - Debugging - Diferenças em tamanho de tela - Canvas 1. Resizing via Canvas vs DOM 2. Aceleração de GPU 3. API de Audio (reference 2) - Performance - Empacotadores HTML5 - Eventos de entrada - Vibração - Acelerômetro - Storage - Disponibilização de assets (controle de tamanhos, cache, etc) - Aplicações offline - CSS media queries

Elaborar uma lista de limitações e correlacionar os dados de acordo com as plataformas.

4 JUSTIFICATIVA

Tendo em vista que este trabalho busca mapear possíveis problemas do desenvolvimento multiplataforma em HTML ele serve para apoiar e justificar decisões relativas ao desenvolvimento de jogos multiplataforma; Por tratar cientificamente de aspectos importantes do HTML, este trabalho tem potencial apontar os pontos chave que necessitam de melhorias nas plataformas alvo, colateralmente colaborando para a melhoria do próprio HTML. A opinião comum tende para soluções nativas em detrimento do desenvolvimento de jogos, este trabalho pretende desafiar esta concepção. (REFERENCIAR) Muitos desenvolvedores estão familiarizados com as tecnologias da WEB ou apontam interesse na tecnologia. Estimular e avançar o estudo da implementação da Open Web;

5 REVISÃO BIBLIOGRÁFICA

5.1 JOGOS

Segundo LEMES (2009, pg 126) > jogo digital constitui-se em uma atividade lúdica composta por uma série de ações e decisões, limitada por regras e pelo universo do game, que resultam em uma condição final. O game é uma hipermídia por excelência e tem na interatividade mediada por aparatos tecnológicos, seu papel fundamental. Essa característica interativa é a dependência comandos sobre uma interface digital, que faz com que o projeto digital desta natureza não seja um filme ou uma animação, e sim um game.

Quando desenvolvendo qualquer jogo, o desenvolvedor tem que considerar seu usuário. O objetivo é maximizar a satisfação de seu usuário. Jogos em plataformas móveis trazem um novo conjunto de desafios para produtores de jogos. Um destes desafios é fornecer feedback suficiente para o player pois o dispositivo é limitado em proporções, som, tela etc. Já jogos multiplataforma em HTML5 tem a dificuldade adicional de ter que comportar, na mesma base, o feedback adequando para cada plataforma móvel.

A interface tem que ser o mais intuitiva o possível. No caso de dispositivos móveis, quanto menos gestos necessários melhor. Tornar previsível causa e efeito é uma boa característica para os jogos. Os desenvolvedores tem que evitar fazer o jogo para eles mesmos. E pela falta de crítica os designs tendem a ser ruins. Afinal o que os jogadores querem? LEMES (2009, pg XX) aponta alguns fatores procurados pelos usuários de jogos: Desafio, socializar, experiência solitária, respeito e fantasia.

5.2 GÊNEROS

LEMES (2009, p. 43) aponta os seguintes gêneros de jogos.

- Adventure
- Ação
- RPG
- Estratégia
- Simuladores
- Esportes
- Luta
- Casuais
- 'God' Games
- Educacionais
- Puzzle
- Online / Massive Multiplayer

5.3 MECÂNICA DOS JOGOS

A mecânica é composta pelas regras do jogo. Quais as ações disponíveis aos usuários, é fortemente influenciada pela categoria do jogo em questão.

5.4 ARQUITETURA DOS JOGOS

Existem algumas estratégias relativas às plataformas alvo de como efetuar construção de jogos.

5.4.1 DESENVOLVIMENTO DE JOGOS NATIVOS

Habilita a melhor experiência de usuário pois permite utilizar ao máximo os recursos e funcionalidades dos aparelhos. Porém, devido a cada plataforma conter seu próprio sistema operacional, com seus próprios *SDK's* totalmente incompatíveis, os desenvolvedores são forçados a desenvolver uma versão do jogo para cada plataforma alvo. Além da replicação dos fontes, esta abordagem requer mais pessoas, e maior custo com possivelmente parte do mercado não atendido de qualquer forma.

5.4.2 DESENVOLVIMENTO DE JOGOS WEB

Necessitam de apenas uma base de código e pode rodar em todas as plataformas. Contém a mais vasta gama de desenvolvedores e muitos interessados em aprendê-la. Seus custos também são inferiores, aos do desenvolvimento nativo – pois demandam menos trabalhadores/hora devido a inexistência de duplicação da base. Não obstante, esta opção – devido a incompletude da especificação de padrões – carece de alguns recursos e outros não estão completamente implementados. Performance também pode ser um limitador, visto que estas tecnologias são executadas através de um navegador, criando uma camada de abstração superior à das API's nativas que fazem chamadas ao sistema diretamente.

5.4.3 DESENVOLVIMENTO DE JOGOS HÍBRIDOS

Jogos híbridos são jogos geralmente desenvolvidos com tecnologias da web: beneficiando-se da não necessidade de duplicação. Rodam dentro de um *container* nativo – possibilitando o acesso à chamadas do sistema, recursos de hardware, eliminando muitias das dificuldades da web. Em certo sentido, beneficiam-se do melhor de ambas as metodologias anteriores. Phone game é uma ferramente deste tipo. Permite acessar os dispositivos utilizando sua API JavaScript. Funciona encapsulando todo o código HTML5. Este tipo de abordagem permite acessar câmera, acelerômetro, GPS, etc.

5.5 NAVEGADORES WEB

Aplicações do lado do cliente geralmente se comunicam com um servidor através de documentos em HTTP. Quado o navegador recebe um destes pacotes em HTML ele começa o processo de renderização. A renderização pode requisitar outros arquivos a fim de completar a experiência desenvolvida para o endereço em questão. Nos navegadores os usuários necessitam localizar a página que desejam, sabendo o endereço, ou pesquisando em buscadores. Isso é um processo árduo para a plataformas móveis pois necessitam maior interação do usuários e não são "naturais" se comparado ao modo normal de consumir aplicativos nestas mesmas plataformas – simplesmente adquirindo o aplicativo na loja e abrindo-o no sistema operacional. Alguns contornos para este problema serão descritos nas tecnologias offline.

Para transformar as instruções retornadas pelo servidor em algo útil para o usuário final os navegadores geralmente fazem uso de bibliotecas externas capazes de interpretar HTML5 e gerar o conteúdo iterativo.

Bibliotecas web

O Google Chrome utiliza o Webkit para renderizar seu conteúdo HTML5. O webkit foi criado pela Apple baseando-se no motor de renderização do Konkeror do projeto KDE. Safari e Opera também fazem uso do Webkit. V8 para JavaScript.

O motor de renderização do HTML5 do Firefox é o XXX. O motor de JavaScript é o.

5.6 ANDROID

É um sistema operacional *open-source*, largamente suportado pelo Google. Utiliza o kernel Linux para as tarefas mais básicas como: gerência de memória, processos, etc. Os programas para Android são geralmente escritos em Java e executados através da máquina virtual Dalvik. Dalvik é similar a máquina virtual Java, mas roda um formato de arquivos diferenciado (dex), otimizados para consumir pouca memória, que são agrupados em um único Android Package (apk). Android permite a renderização de documentos HTML através de sua própria API WEBVIEW. Ou através do navegador disponibilizado por padrão, ou outros de terceiros como o Google Chrome, Firefox, Opera, etc.

No quesito jogos para dispositivos móveis é preferível disponibilizar os jogos através da interface nativa pois dá a sensação de continuidade para com os demais aplicativos instalados no dispositivo.

5.7 HTML

Um documento HTML representa conteúdo iterativo de uma forma independente de plataforma. Podem ser enderizados em uma tela, em um sintetizador de voz, etc.

Com o crescimento da demanda de interatividade na internet o HTML foi forçado a evoluir. Em sua quinta versão, o HTML5, grandes melhorias foram adicionadas. Antes do HTML5 para alcançar a interatividade desejada, só te podia recorrer a plugins de terceiros como o Flash Player e o Silverlight, estes necessitam de instalação de plugin e ficam limitados a um distribuidor de software.

O HTML em si trata cruamente da estrutura, para as páginas ficarem agradáveis e seus usuários faz-se necessário o uso de estilos que é proporcionado através do CSS.

A WHATWG começou a construção do HTML5 e a W3C se interessou e entrou no projeto. Mesmo assim, a WHATWG mantém uma versão com licença menos restrita que a W3C contendo recursos a mais dos que os especificados no HTML atual.

5.7 CSS

Uma linguagem de marcação, como o HTML, que utiliza seletores e regras para definir a representação de um documento HTML e seus elementos.

Os navegadores interpretam CSS através da tag `<style>`. O CSS é dividido em módulo, contendo aproximadamente 50 deles.

Sua última versão, o CSS3, introduziu várias funcionalidades multiplataforma como media-queries que possibilitam regras para tamanhos de tela e transformações 3D.

Muitos navegadores também suportam aceleração de GPU (Unidade de processamento gráfico) para elementos que tenham transformações 3d.

Flow de documento, ordem e posição em que os elementos tem que aparecer na página. Modelo de caixa o que encapsula o conteúdo em um elementos.

5.8 JavaScript

EMACScript melhor conhecido como JavaScript foi criado por Brendan Eich na Netscape. O padrão Ecma é a especificação do JavaScript. JScript foi um concorrente do JavaScript também baseado no padrão Ecma. JavaScript tornou tão popular que foi abraçada pela W3C Atualmente a linguagem de programação mais popular do mundo. Todo o navegador atual contém um interpretador de JavaScript tornando-o portável.

O objetivo inicial do JavaScript diverge muito da produção de projetos de larga escala como são feitos atualmente.

Existem vários conversores de código (*transpilers*) para JavaScript que permitem a utilização de outras linguagens durante o desenvolvimento e posterior geração de JavaScript. A nova versão do JavaScript, o JavaScript 6, agrega vários conceitos de orientação à objetos.

Tipicamente JavaScript é criado separadamente do HTML, através da tag `<script>`. Quando os navegadores encontram essa tag eles fazem a requisição para o servidor para injetar o código no documento. Isso pode ser problemático para projetos que incluam vários scripts.

> A scripting language is a programming language that is used to manipulate, customise, and automate the facilities of an existing system. In such systems, useful functionality is already available through a user interface, and the scripting language is a mechanism for exposing that functionality to program control. In this way, the existing system is said to provide a host environment of objects and facilities, which completes the capabilities of the scripting language. A scripting language is intended for use by both professional and non - professional programmers.

JavaScript se tornou portável e isso é um componente chave no desenvolvimento de jogos.

> JavaScript has become one of the most popular programming languages on the Web. Initially, however, many professional programmers denigrated the language because its target audience consisted of Web authors and other such "amateurs", among other reasons.[32] The advent of Ajax returned JavaScript to the spotlight and brought more professional programming attention. The result was a proliferation of comprehensive frameworks and libraries, improved JavaScript programming practices, and increased usage of JavaScript outside Web browsers, as seen by the proliferation of server-side JavaScript platforms.
>
> As in most scripting languages, types are associated with values, not with variables. For example, a variable x could be bound to a number, then later rebound to a string. JavaScript supports various ways to test the type of an object, including duck typing.
>
> JavaScript is almost entirely object-based. JavaScript objects are associative arrays, augmented with prototypes (see below). Object property names are string keys. They support two equivalent syntaxes: dot notation (obj.x = 10) and bracket notation (obj['x'] =

10). Properties and their values can be added, changed, or deleted at run-time. Most properties of an object (and those on its prototype inheritance chain) can be enumerated using a for…in loop. JavaScript has a small number of built-in objects such as Function and Date.

A web browser provides an ECMAScript host environment for client side computation including, for instance, objects that re present windows, menus, pop ups, dialog boxes, text areas, anchors, frames, history, cookies, and input/output. Further, the host environment provides a means to attach scripting code to events such as change of focus, page and image loading, unloading, er ror and abort, selection, form submission, and mouse actions. Scripting code appears within the HTML and the displayed page is a combination of user interface elements and fixed and computed text and images. The scripting code is reactive to user interacti on and there is no need for a main program.

A web server provides a different host environment for server side computation including objects representing requests, clients, and files; and mechanisms to lock and share data. By using browser - side and server side scripting together, it is possible to distribute computation between the client and server while providing a customised user interface for a Web based application.

In a class based object oriented language, in general, state is carried by instances, methods are carried by cla sses, and inheritance is only of structure and behaviour. In ECMAScript, the state and methods are carried by objects, and structure, behaviour, and state are all inherited.

JavaScript includes an eval function that can execute statements provided as strings at run-time.

Some of the facilities of ECMAScript are similar to those used in other programming languages; in parti cular Java, Self, and Scheme
Javascript Estrito

The ECMAScript Language recognises the possibility that some users of the language may wish to restrict their usage of some feat ures available in the language. They might do so in the interests of security, to avoid what they consider to be error prone features, to get enhanced error checking, or for other reasons of their choosing. In support of this possibility, ECMAScript define s a strict variant

> of the language. The strict variant of the language excludes some specific syntactic and semantic features of the regular ECMAScript language and modifies the detailed semantics of some features. The strict variant also specifies additio nal error conditions that must be reported by throwing error exceptions in situations that are not specified as errors by the non strict form of the language.

ASM.js

A versão estrita do JavaScript o asm.js permite grandes aumentos de performance e foi especialmente delineada para a geração automática de código a partir de outras linguagens. asm.js é especialmente importante no contexto dos jogos que usualmente consomem muitos recursos.

> asm.js is not typically written directly: instead, as an intermediate language, it is generated through the use of a compiler that takes source code in a language such as C++ and outputs asm.js.
>
> Much of this performance gain over normal JavaScript is due to 100% type consistency and virtually no garbage collection (memory is manually managed in a large typed array). This simpler model with no dynamic behavior, no memory allocation or deallocation, just a narrow set of well-defined integer and floating point operations enables much greater performance and potential for optimization.[citation needed]

> Games and game engines

Unreal Engine 3: a full version[citation needed] which was ported in 4 days[26][27] Unreal Engine 4 The Unity game engine[28] Doom: the open source Freedoom game assets running on PrBoom, which is based on the open source Doom code[29] SuperTux[30] ScummVM, which supports numerous classic adventure games[31] Dune II via OpenDune[32] BananaBread based on Cube 2[33] Every game in the Humble Mozilla Bundle[34] (Super Hexagon, Aaaaa! for the awesome, Osmos, Zen Bound 2, Dustforce DX, Voxatron, FTL: Advanced Edition and Democracy 3)

5.9 Ajax

Ajax Limitations

> In pre-HTML5 browsers, pages dynamically created using successive Ajax requests did not automatically register themselves with

the browser's history engine, so clicking the browser's "back" button may not have returned the browser to an earlier state of the Ajax-enabled page, but may have instead returned to the last full page visited before it. Such behavior — navigating between pages instead of navigating between page states — may be desirable, but if fine-grained tracking of page state is required, then a pre-HTML5 workaround was to use invisible iframes to trigger changes in the browser's history. A workaround implemented by Ajax techniques is to change the URL fragment identifier (the part of a URL after the "#") when an Ajax-enabled page is accessed and monitor it for changes.[12][13] HTML5 provides an extensive API standard for working with the browser's history engine.[14] Dynamic Web page updates also make it difficult to bookmark and return to a particular state of the application. Solutions to this problem exist, many of which again use the URL fragment identifier.[12][13] The solution provided by HTML5 for the above problem also applies for this.[14] Depending on the nature of the Ajax application, dynamic page updates may interfere disruptively with user interactions, especially if working on an unstable Internet connection. For instance, editing a search field may trigger a query to the server for search completions, but the user may not know that a search completion popup is forthcoming, and if the internet connection is slow, the popup list may show up at an inconvenient time, when the user has already proceeded to do something else. Excluding Google,[15] most major Web crawlers do not execute JavaScript code,[16] so in order to be indexed by search engines, a Web application must provide an alternative means of accessing the content that would normally be retrieved with Ajax. It has been suggested that a headless browser may be used to index content provided by Ajax-enabled websites.[17] Any user whose browser does not support JavaScript or XMLHttpRequest, or simply has this functionality disabled, will not be able to properly use pages which depend on Ajax. Devices such as smartphones and PDAs may not have support for the required technologies, though this is becoming less of a problem. The only way to let the user carry out functionality is to fall back to non-JavaScript methods. This can be achieved by making sure links and forms can be resolved properly and not relying solely on Ajax.[18] Similarly, some Web applications that use Ajax are built in a way that cannot be read by screen-reading technologies, such as JAWS. The WAI-ARIA standards provide a way to provide hints in such a case.[19] Screen readers that are able to use Ajax may still not be able to properly read the dynamically generated content.[20] The same origin policy prevents some Ajax techniques from being used across domains,[8] although the W3C has a draft of the XMLHttpRequest object that would enable this functionality.[21] Methods exist to sidestep this security feature by using a special Cross Domain

> Communications channel embedded as an iframe within a page,[22] or by the use of JSONP. The asynchronous callback-style of programming required can lead to complex code that is hard to maintain, to debug[23] and to test.[24]

5.10 ALTERNATIVAS AO JavaScript

Abaixo seguem algumas tecnologias que servem de alternativa ao JavaScript.

5.10.1 TYPESCRIPT

Conhecido como uma versão estendida do JavaScript que compila para JavaScript normal. Isso significa que os desenvolvedores podem continuar escrevendo Javascript normalmente. Typescript oferece classes, interfaces e módulos.

5.10.2 DART

Google. DartVM é uma máquina virtual que está embebido no Google Chrome. Significante melhorias em performance quando comparado ao JavaScript. Existe o dart2js que compila código em Dart para JavaScript.

5.11 DOCUMENT OBJECT MODEL (DOM)

É uma plataforma e interface agnóstica a linguagem que permite os programas e scripts dinamicamente acessar e atualizar o conteúdo, estrutura e estilo de documentos. Pode ser novamente processado e o resultado aparecer na tela. O navegador cria um DOM quando ele processa os elementos e tags encontrados em um documento HTML. Gmail é uma aplicação de única página (single-page) que se baseia fortemente no DOM para gerar conteúdo dinâmico e interativo oferecido pelo DOM.

5.12 CANVAS

A nova tag

define um layer gráfico em documentos HTML que pode ser desenhado através de JavaScript. Permite desenhar diagramas, gráficos e animações [7]. É baseado em bitmap. O suporte ainda é escasso. Muitas vezes lento. Algumas soluções tentam arrumar isso através da utilização de GPU. Apache Cordova utiliza o FastCanvas.

CocoonJS é uma aplicativo híbrido que preenche a fraca implementação de OPENGL nos dispositivos móveis possibilitando se desenvolver em WEBGL.

5.13 WEBGL

Baseado no OpenGL.

Web GL não foi utilizada no trabalho apesar de ser de grande relevância no processo de jogos pois ainda não está completamente especificada e a dificuldade e escopo do projeto aumentariam muito se tivessem de incluir um jogos 3D. Versão da especificação atual?

5.14 VIDEO

5.15 AUDIO

Audio é um componente vital para oferecer grande satisfação aos usuários de jogos. Provê feedback e imerge o usuário. Efeitos de som e música podem servir como mecanismo. Jogadores tem baixa tolerância a volume, deve ser utilizado com cautela.

5.14.1 TAG AUDIO

A tag define um som dentro de um documento html. Quando o elemento é renderizado pelos navegadores, ele carrega o conteúdo que pode ser reproduzido pelo player de audio do navegador. Existem muitas discrepâncias entre os formados aceitáveis pelos navegadores. È um tanto limitada quanto comparada ao áudio de múltiplos canais disponibilizados por SDKs nativas.

5.14.2 API DE AUDIO

É uma interface de audio experimental para JavaScript. Provê maior flexibilidade na manipulação de audio. Essa tecnologia é muito mais nova do que a tag audio. FORMATOS DE ÁUDIO

5.15 CAMERA

5.16 ENTRADA DE COMANDOS

Na construção da grande maioria dos jogos é muitas vezes imprescindível alta flexibilidade na gestão de entrada de dados. Este fator muito se amplia na criação de jogos multiplataforma, seja através de teclado, tela sensível ou sensor de movimentos, o importante é oferecer a melhor experiência possível por plataforma. O HTML5 trata todos estes casos abstratamente na forma de eventos, os quais podem ser escutados através de listeners. Os eventos básicos são: keydown (tecla baixa), keyup (tecla solta) e keypress (tecla pressionada).

Para detectar suporte aos mais variados recursos do HTML5 no navegador do cliente existem duas possibilidades. Pode-se implementar testes para cada funcionalidade utilizada abordando os detalhes de implementação de cada uma ou então fazer uso de alguma biblioteca especializada neste processo, o Modernizr é uma opção open-source deste tipo de biblioteca, este gera uma lista de booleanos sobre grande variedade dos recursos HTML5, dentre estes, geolocalização, canvas, áudio, vídeo e local storage.

5.17 CACHE

Aplicações offline.

Algumas tecnologias desta classe são:

5.18 OFFLINE E ARMAZENAMENTO

> Uma das grades limitações do HTML era a ausência de capacidade de armazenamento de dados. Armazenamento no lado do cliente é um requerimento básico para qualquer aplicação moderna. Essa área era ode as aplicações nativas detinham grande vantagem sobre as aplicações web. O HTML5 solucionou este problema introduzindo várias formas de armazenamento de dados. (HASAN et all, 2012)

5.18.1 LOCAL STORAGe

Também conhecido como WebStorage na especificação do HTML5. Provê uma forma de armazenar os dados como chave valor dentro do navegador. Os dados são persistido mesmo que o navegador seja fechado.

5.18.2 WEB SQL

Simplesmente um banco de dados SQLite embebido no navegador. Permite tabelas relacionais. O tamanho padrão do banco de dados é 5 megabytes e pode ser estendido pelo usuário.

5.19 RECURSOS NATIVOS ATUALMENTE INDISPONÍVEIS PARA O HTML5

- Suporte à câmera;
- Suporte à calendário;

5.20 DEBUG

5.20.1 WEINRE

Debugger remoto depreciado.

5.21 TECNOLOGIAS POLYFILL Acarretando assim, que muitos navegadores não implementam algumas funcionalidades, completa ou parcialmente especificadas, daí surge a necessidade dos polyfills (tecnologias de preenchimento de lacunas) para implementar estas camadas.

Uma das soluções mais promissoras polyfill é o PhoneGap ou Apache Cordova, esta ferramenta é Open-source e possibilita utilizar de inúmeros recursos de hardware da grande maioria das produtoras de dispositivos móveis.

5.22 FERRAMENTAS

5.22.1 NODEJS

Permite rodar JavaScript fora do navegador. Utiliza um modelo dirigido à eventos sem bloqueio, tornando-o rápido e eficiente.

5.22.2 SISTEMAS DE BUILDING

Aquivos JavaScript são requisitados do servidor assincronamente. Isso pode levar a tempos de requisição pouco desejáveis. Uma saída seria escrever o código em apenas um arquivo mais isso leva a gerência de código bagunçada. A saída mais comum entre desenvolvedores é utiliza ruma ferramenta que junta todos os arquivos e disponibiliza apenas um para o usuário.

5.22.3 GRUNT

Aplica as modificações separadamente em cada arquivo.

5.22.4 GULP

Utiliza o conceito de streams para aplicar todas as modificações sobre um arquivo de uma vez só. Minify, obfuscation

5.22.5 SOURCE MAPS

Para encontrar os arquivos minificados a fim de ajudar o desenvolvedor a debugar a aplicação.

5.22.6 MINIFY

Remover caracteres desnecessários do JavaScript como espaços vazios, diminuindo o tamanho dos nomes, fazendo o tempo de loading diminuir.

5.22.7 GERENCIADORES DE PACOTES

5.22.8 BOWER

Package manager para a web

5.22.9 NPM

Package manager para o NODE

5.23 DISPONIBILIZAÇÂO DA APLICAÇÂO

Links com manifestos

5.23.1 INSTALAÇÃO

Este método é benéfico pois possibilita ao usuário a mesma experiência ao adquirir uma aplicação normal. Este tipo de aplicação é comummente referido como "híbrido".

5.23.1.1 CROSSWALK

Crosswalk empacota os fontes juntamente com uma versão do Chromium, a versão Open-source do Google Chrome. Isso faz com que o software se comporte da mesma forma para todas as versões de dispositivos Android.

5.23.1.2 PHONEGAP

5.23.2 PHONEGAP CLOUD

Este serviço possibilita que se faça upload de um arquivo compactado contendo os fontes – ou apontando para um repositório no GitHub – que no tempo desta pesquisa não estava funcionando; e se gere o APK para o Android nativamente.

5.24 O JOGO

Devido ao fato deste trabalho explorar as limitações dos jogos em HTML5, optei por evitar a utilização de plugins e ferramentas de terceiros que pudessem ocultar alguma limitação. Escolhi a simplicidade para não precisar ficar muito tempo aprendendo as coisas em detrimento do refinamento da pesquisa.

5.24.1A mecânica

O jogo consiste em simplesmente em uma tela que apresenta equações e um possível resultado. Cabe ao jogador decidir se o resultado está certo ou errado. O tempo é um fator levado em consideração, quão mais rápido o jogador acertar se a afirmação está correta ou não, mais pontos ele receberá.

Argumentos à favor da escolha do game: Tem profundidade, permite a adição de novos recursos no futuro; É facilmente traduzível em tamanhos de telas diferentes e tipos de entrada de dados diferentes;

5.24.2 IMPLEMENTAÇÃO

Não tenho grande experiência com o desenvolvimento de jogos nem com o desenvolvimento em HTML5. Também para não interferir na pesquisa busquei não me distanciar do que é considerado padrão em ferramentas e métodos. Comecei escrevendo o aplicativo para o Navegador do desktop pois era o que estava mais acessível no momento. Mais tarde descobri que de fato é assim que de desenvolve.

5.25 FEEDBACK

6 TRABALHOS SIMILARES

(Referência 2) Faz uma revisão de aspectos do HTML5 através da construção de um jogo. O autor foca muito nos aspectos de criação de jogos e feedback do desenvolvimento. Troca de tecnologias e não especificamente nas limitações conforme o meu trabalho. Em outras palavras seu escopo é mais genérico e não tão preciso quanto este

7 METODOLOGIA

O primeiro passo consiste em definir as plataformas alvo do trabalho; devem ser plataformas mercadologicamente relevantes ao desenvolvimento de jogos, que possibilitem a criação de aplicativos em HTML e que acentuem o antagonismo

de características. Segue-se com a construção de uma lista com os recursos relevantes aos jogos que, empiricamente, sofrem ou são comummente ligados à limitações multiplataforma. Segue-se uma pesquisa para aprofundar teoricamente cada um dos recursos, possivelmente elegendo novos.

Com um baseamento teórico substancial, o próximo passo é a criação do protótipo de um jogo multiplataforma que utilize recursos potencialmente limitados. Para ser considerado pronto, o protótipo deve ser testado, e estar funcional, com adaptações ou não, em cada uma das plataformas alvo definidas.

Com o protótipo concebido, o passo que segue é a enumeração, e descrição das limitações detectadas no processo de desenvolvimento e testes do jogo. Este detalhamento deve responder as seguintes perguntas:

- Quais as limitações foram encontradas no jogo?
- Em quais plataformas?
- Sob quais circunstâncias?
- As limitações puderam ser contornadas?
- Algum efeito colateral das limitações no jogo?
- Qual a categoria do problema: usabilidade, funcionalidade, manutibilidade, portabilidade ou performance? (segundo ISO)

8 RESULTADOS

Abaixo constam as limitações encontradas durante a pesquisa e concepção do jogo

Durante a construção do jogo utilizei a estratégia de declarar todos os objetos relativos ao window e limitar o escopo. Isso se demonstrou uma boa forma de separar as responsabilidades.

8.1 LIMITAÇÕES

> Apesar da grande maioria dos recursos dos dispositivos estar presente em HTML5 ainda existem muitas funcionalidades faltando para este tipo de aplicação. Por exemplo, não podemos mudar a imagem de fundo do dispositivo, ou adicionar toques etc. Similarmente, existem muitas APIs de nuvem como os serviços de impressão do ICloud ou Google cloud que estão disponíveis para aplicações nativas mas não para HTML5. Outros serviços utilitários como o C2DM do Google que está disponível para desenvolvedores Android para utilizar serviços de push também não estão disponíveis para o HTML5. (HASAN, 2012)

1. VERSÕES A grande maioria dos dispositivos atualmente no mercado utilizam obsoletas de seus softwares. Isso dificulta o desenvolvimento. Se a tecnologia de tradução para o navegador utilizar o a classe Webview do Android - como o Apache cordova faz - as versões mais antigas podem ser penalizadas com problemas de performance ou falta de recursos.

2. OFFLINE

Refresh duplo para ver assets cacheados. Ver: http://buildnewgames.com/game-asset-management/

3. AUDIO Api de som quebra quando executado diversas vezes. Os navegadores variam na disponibilização de formatos aceitáveis Somente um áudio pode ser tocado no Navegador do Android Não é possível trocar o volume no IOS. Alguns navegadores favorecem formatos ogg (vorbis) e outros, como o Safari, favorecem o MP3.

   > O maior problema com as API's de áudio e de vídeo do HTML5 é a disputa entre os codecs dos navegadores. Por exemplo, Mozilla e Opera suportam Theora, já o Safari suporta H.264 que também é suportado pelo IE9. Ambos, Iphone e Android suportam H.264 em seus navegadores. A W3C recomenda OggVorbis e OggTheora para áudio e vídeo respectivamente. (HASAN et al, 2012)

3. VIDEO

Codecs

4. ASSETS

Trafegar muitos assets deixa o sistema lento.

Contorno Utilizando páginas de carregamento e/ou cache;

5. UI

É muito custoso desenvolver uma interfaces que pareçam nativas para cada dispositivo sem a utilização de plugins e ferramentas especializadas. Em termos gerais, trabalhar com proporções é positivo. Não obstante há casos, como o dos botões de certo e errado que a proporções ficam exageradas, nesses casos a utilizada de max-width é uma solução conveniente.

6. PERFORMANCE

De acordo com uma pesquisa, para um usuário uma tarefa é instantânea se ele leva até 0.1 segundos para ser executada. Se a tarefa toma aproximadamente um segundo então a demora será notada mas o usuário não se incomodará com ela. Entretanto, se a tarefa leva aproximadamente 10 segundos para terminar o usuário então começa a ficar aborrecido e esse é o limite que algum feedback deve ser dado para um usuário.

ACELERAÇÂO DE GPU

7. Acelerômetro
8. IMPLEMENTAÇÃO INCONSISTENTE DE APIs
9. TAMANHO DE TELA Em alguns casos o tamanho das telas pode ser um fator limitante – como no caso de jogos de estratégia. Jogadores com telas menores podem sair em desvantagem.
10. CAMERA

10 . Javascript Ciclo de vida demorado pois necessita que todos os consumidores da especificação entrem em consenso e implementem-a.

Desktop/Firefox Desktop/Google Chrome Smatphone/Android

9 CONCLUSÕES

Não pude testar todos os métodos e ferramentas e versões à disposição, um trabalho completo demandaria esforços conjuntos de muitos indivíduos ou um período de tempo bem mais extenso. Se uma empresa deseja produzir jogos nativos elas precisarão de vários desenvolvedores. Eu sozinho fui capaz de produzir um jogo em tempo razoável trabalhando apenas com a plataforma web.

Por não utilizar frameworks e bibliotecas estou me distanciando dos casos da vida real. Só poderemos considerar o HTML como uma especificação pronta quando for possível fazer tudo o que se faz nativamente com os dispositivos através de uma API web padronizada.

> Conforme JavaScript vai ganhando importância rápido progresso é feito por diferentes empresas a fim de prover boas ferramentas de debug e inspecionamento para JavaScript.

9.1 TRABALHOS FUTUROS

EMACSCRIPT 7

10 CRONOGRAMA

O cronograma foi especificado de acordo com o detalhado na metodologia, suas datas estão especificadas de acordo com dias úteis disponíveis no calendário. Etapa Indicador físico Previsão

BIBLIOGRAFIA

LEMES, David Oliveira. Games independentes. Pontifícia Universidade Católica de São Paulo 2009.

Referência 3 é uma ótima fonte de explicação de como tecnologias relacionadas ao assunto funcionam.

Referência 2 é uma ótimo trabalho para pegar ideias sobre como escrever as coisas.

Referência 7 is a good font from feature detection systems.

HTML5: A blessing or a curse http://www.develop-online.net/tools-and-tech/html5-a-blessing-or-a-curse/0117393

5 common mistakes that html5 developers make http://www.toptal.com/html5/top-5-mistakes-that-html5-developers-make

GLOSSÁRIO

API BUG

ANEXOS

10.1 CONVERSORES PARA HTML5

Além da possibilidade de escrever em HTML, pode-se optar pela alternativa de utilizar-se um conversor de linguagens.

10.2 METODOLOGIA DE DESENVOLVIMENTO DE SOFTWARE PARA A CONSTRUÇÃO DE GAMES

Como o jogo é um software complexo demanda-se a utilização de metodologias de engenharia de software, dentre os processos de software mais conhecidos academicamente destacamos:

- OpenUP: este é bem detalhado e de característica iterativa e incremental. Gerando assim, um levantamento mais apurado dos riscos, requisitos e outros detalhes do sistema e a criação incremental do sistema, com requisitos maleáveis;
- Cascata: processo antigo, caracteriza-se por ser pouco maleável aos requisitos mapeados posteriormente ao processo de análise;
- Processo ágil - SCRUM: sua utilização é flexível e sendo um método ágil especifica pouca documentação, ou como dizem, somente a documentação necessária, este processo é bem conhecido e aceito na comunidade de desenvolvimento de software. Suas principais características são: divisão do processo de desenvolvimento através uma série de iterações chamadas sprints. Cada sprint consiste tipicamente em duas a quatro semanas. É bem aplicado a projetos que mudam constantemente e que demandam rápidas adaptações;
- Processo ágil – XP: tem muitas características similares ao SCRUM por este também ser um processo ágil. Dentre suas especifidades destaca-se: versões frequentes, pequenos ciclos de desenvolvimento que buscam aumentar a produtividade, introduzem checkpoints onde os clientes podem agregar novas funcionalidades;

10.3 AMBIENTES PARA DESENVOLVIMENTO HTML5

Na pesquisa efetuada sobre estes frameworks full stack foram identificadas as seguintes tecnologias: - segundo (PRADO, 2012) o GWT é um framework essencialmente para o lado do cliente (client side) e dá suporte à comunicação com o servidor através de RPCs Remote Procedure Calls (ou procedimento de chamadas remotas). Ele não é um framework para aplicações clássicas da web, pois deixa a implementação da aplicação web parecida com implementações em desktop. Este é utilizado em muitos produtos de grande porte como o Google Adwords e Google Wallet. Outra característica interessante é que a plataforma opera sobre a licença Apache versão 2; - construct 2 - é um editor na nuvem focado para usuários sem conhecimento prévio em programação orientado a comportamento; - PlayCanvas - é uma plataformas para a construção de jogos 3D na nuvem, desenvolvida com foco em performance. Permite a hospedagem, controle de versão e publicação dos aplicativos nela criados, possibilita também a importação de modelos 3D de softwares populares como: Maya, 3ds Max e Blender; - o ambiente HTML5 Development Environment (ambiente de desenvolvimento HTML5) da Intel, este fornece uma solução na nuvem, completa para o desenvolvimento em plataforma cruzada, com serviços de empacotamento, serviços para a criação e testes de aplicativos com montagem de interfaces drag and drop (Intex XDK) e bibliotecas para a construção de jogos utilizando aceleração de hardware, o que garante até duas vezes mais performance que aplicativos mobile baseados em Web tradicionais. Esta solução é free, open source e funciona através de um plugin para o Google Chrome, ou seja, o desenvolvimento também é multiplataforma e devido ao fato de os binários ficarem hospedados na

nuvem, possibilitou a Intel criar compiladores para cada uma das plataformas disponibilizadas pelo PhoneGap, que é o framework polyfill utilizado na solução.

10.4 HTTP

FRAMEWORKS DE DESENVOLVIMENTO DE JOGOS EM HTML5;

10.5 FRAMEWORKS PARA DESENVOLVIMENTO DE JOGOS HTML5

Com o intuito de simplificar o processo para os desenvolvedores, auxiliando-os a focarem-se apenas nas soluções que estão desenvolvendo, foram criados os frameworks para desenvolvimento de jogos. Não obstante, o intuito deste trabalho é desenvolver um jogo sem auxilio de frameworks pois estes muitas vezes escondem possíveis limitações, desenvolvendo *workaroundslog* próprios.

- enchant.js: dentre suas funcionalidades constam: orientação à objetos, orientado à eventos, contém um motor de animação, suporta WebGL e Canvas, etc;
- three.js: considerada leve, renderiza WebGL e Canvas, arquitetura procedural;
- quintus: bom para plataformas 2D
- limeJs: bom para 2d

10.6 INTERFACE E ESCOLHAS DE DESIGN

Mobile first