Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

A IMPORTÂNCIA DOS CUIDADOS COM *DESIGN* RESPONSIVO E A USABILIDADE NO DESENVOLVIMENTO DE APLICAÇÕES *WEB*

José Nelson Cultri [1]

Carlos Alberto Lucas [2]

Resumo

Aplicações *web* devem ser sempre implementadas com as tecnologias mais atuais possíveis para atingir o maior número de usuários possível. Dos conceitos mais conhecidos, dois deles imprescindíveis na atualidade são os conceitos de usabilidade e *design* responsivo. O presente artigo foi elaborado com base no levantamento bibliográfico e dividido em quatro seções, sendo elas a introdução, com uma explanação básica dos conceitos a serem tratados, a estrutura do artigo bem como os objetivos e metodologia de desenvolvimento do artigo, uma seção sobre usabilidade, uma seção sobre o *design* responsivo e uma seção apresentando experiências e estatísticas sobre a utilização de ambos os conceitos. Este artigo é finalizado com a conclusão, sendo feita uma análise sobre a implementação desses conceitos e, em seguida, são apresentadas as referências bibliográficas do artigo.

Palavras-chave: Aplicações *Web,* CSS, *Design* Responsivo, *Layout*, Usabilidade.

Abstract

Web applications should always be implemented with the most updated technologies to reach as many users as possible. Among the most known concepts, two essential ones, nowadays, are the concepts of usability and responsive design. This article was elaborated based on some bibliographical research and divided into four sections, being the introduction, with a basic explanation of the concepts to be treated; the development of the article as well as the objectives and methodology of the article writing process; a section about usability; a section about responsive design and a section presenting experiences and statistics on the use of both concepts. This article ends up with a conclusion by making an analysis of the implementation of these concepts and then the bibliographical references of the article are presented.

Keywords: *CSS, Layout, Responsive Design, Usability, Web Applications*

[1] Graduando em Análise e Desenvolvimento de Sistemas pela Fatec Dr. Thomaz Novelino – Franca/SP, Endereço eletrônico: josefcultri@yahoo.com.br

[2] Docente do curso de Análise e Desenvolvimento de Sistemas pela Fatec Dr. Thomaz Novelino – Franca/SP. Endereço eletrônico: projetos@profcarloslucas.com.br

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

1 Introdução

Inovação sempre será um quesito importantíssimo em sistemas de informação. Em termos de tecnologia da informação inovar traduz-se por sobreviver, e as empresas sobreviverão à medida que inovarem (JOBS, 2012).

O presente artigo tratará das questões de usabilidade em sistemas *web* e será dividido em quatro seções, a primeira sendo uma introdução sobre o artigo em si, a segunda tratando da usabilidade, a terceira tratando sobre o *design* responsivo e a quarta trará estatísticas sobre a importância da utilização da usabilidade e do *design* responsivo no desenvolvimento de aplicações *web*. Por fim serão apresentadas as conclusões sobre o artigo, bem como as referências.

São três os objetivos deste artigo: apresentar os conceitos de usabilidade e *design* responsivo através de exemplos, pontuar a devida importância destes conceitos e, por fim, servir como um guia para desenvolvedores tanto iniciantes quanto experientes sobre os temas que serão tratados.

Segundo Pagani (2011, *online*), “a usabilidade é englobada dentro da qualidade e visa garantir uma parte da eficiência e eficácia do sistema”. Com uma enorme gama de dispositivos sendo utilizados atualmente para acesso a sistemas diversos, faz-se necessário o cuidado de manter o sistema o mais prático e objetivo possível.

Muitos dispositivos diferentes implicam em necessidades distintas de apresentação do sistema. É nesse ponto que entra o *design* responsivo em uma aplicação *web*. Segundo Santana (2015, *online*), “*Design* responsivo é um conceito de otimização estrutural e de design de sites para diferentes tipos de tela de dispositivos”. Com as pessoas utilizando cada vez mais um *smartphone* ou até um *tablet* para acesso à internet e realização da maioria de suas tarefas diárias como leitura de notícias ou pagamento de contas nos *sites* dos bancos, os sistemas *web* precisam ser atualizados e responder bem aos dispositivos pelos quais estão sendo acessados, passando assim, através da usabilidade e de seu *design* responsivo, uma boa impressão para o usuário final, que entenderá que o sistema está funcionando da maneira correta e esperada.

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

2 Usabilidade

Usabilidade está relacionada com a facilidade de navegação, sendo o primeiro estágio da criação de um *site*, visando uma melhor interação entre usuário e interface (RIBEIRO, 2016). Um sistema com uma boa usabilidade é um sistema em que o usuário identifica facilmente quais as ações necessárias para concluir suas tarefas.

"Se uma página falha em mostrar claramente o que uma empresa oferece e o que os usuários podem fazer no *site*, as pessoas saem" (NIELSEN, 2012, traduzido). Como a velocidade e o fácil acesso a serviços são características fundamentais do uso da *internet*, as pessoas precisam de sistemas e páginas que forneçam tais características. Nielsen, considerado o pai da usabilidade, definiu uma série de dez heurísticas, que servem de orientação aos profissionais de UX (do inglês *User Experience*, experiência de usuário) para criação dos *designs* das páginas *web*. Estas heurísticas serão descritas ao longo desta seção.

É importante que o usuário deve estar ciente do estado do sistema em tempo real, ou seja, o sistema deve contextualizar sempre o usuário de quais ações estão acontecendo naquele momento. Um bom exemplo de aplicação desse conceito é o aplicativo *web* do Spotify.

Figura 1 - Interface do aplicativo *web* do Spotify

Fonte: Captura de tela do aplicativo *web* do Spotify

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

Na figura 1 é possível identificar qual a *playlist* de música que está em execução, qual a música que está sendo tocada no momento e ainda verificar quais serão as próximas músicas, tudo de maneira objetiva.

O usuário deve se sentir ambientado no sistema. Para isso, é necessário que os elementos da interface sejam elementos comuns para ele no que tange ao mundo real, ou seja, devem ser elementos de fácil identificação no dia-a-dia. O YouTube é um excelente exemplo nesse ponto, como visto na figura 2.

Figura 2 – *Site* do YouTube com uma interface correspondente ao mundo real

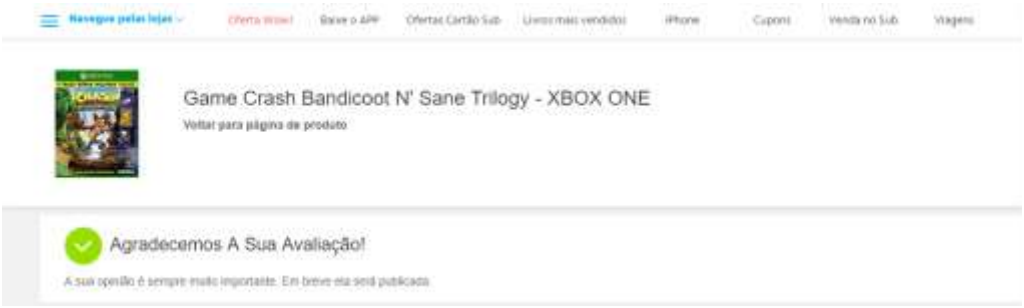
Fonte: Captura de tela da página principal do YouTube

Com a figura 2 é possível identificar facilmente o lugar de busca para que o usuário encontre os vídeos que deseja assistir, o ícone que dispara a ação de gravação de um vídeo para *upload* na plataforma e também o ícone indicando as notificações de vídeos não vistos.

É também importante que o usuário entenda que tem a liberdade para fazer o que bem entender dentro do sistema, exceto quando a ação irá ferir a integridade de outra rotina ou então de quesitos legais. Um exemplo interessante é o *e-commerce* Submarino e sua seção de comentários, exemplo visto na figura 3.

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

Figura 3 – Comentários no Submarino, em um produto comprado

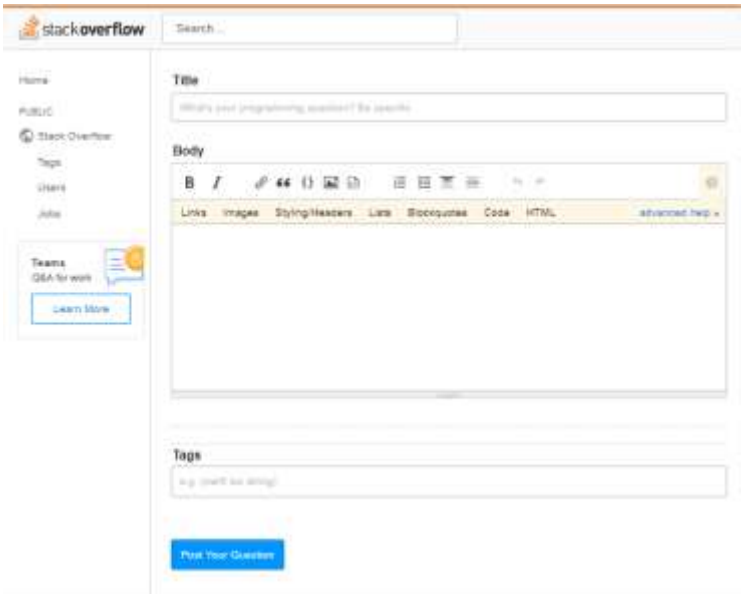


Fonte: Captura de tela da mensagem de alerta do *site* Submarino

Ao postar um comentário no Submarino ele é submetido para análise de uma equipe que exibirá o comentário ou não no *site*, evitando assim *spam* ou então comentários muito ofensivos.

A fim de que o *site* seja interessante para o usuário, é importante manter uma consistência visual, aplicando padrões de fontes e cores ao longo do sistema para que o mesmo consiga identificar facilmente a ação que será realizada por determinado botão ou até mesmo aonde deve digitar determinada informação. Bons exemplos desse ponto são os *sites* do StackOverflow e do StackExchange, representados nas figuras 4 e 5, respectivamente. As páginas para realizar as postagens em ambas as plataformas possuem a mesma identidade visual, o que facilita para os usuários no tocante a operar a caixa de comentários.

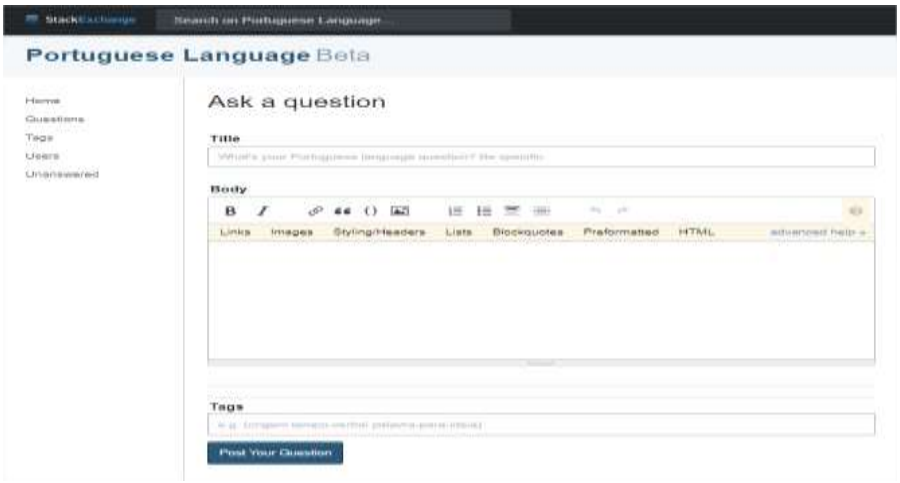
Figura 4 – Caixa de postagens do StackOverflow



Fonte: Captura de tela da caixa de postagem do StackOverflow

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

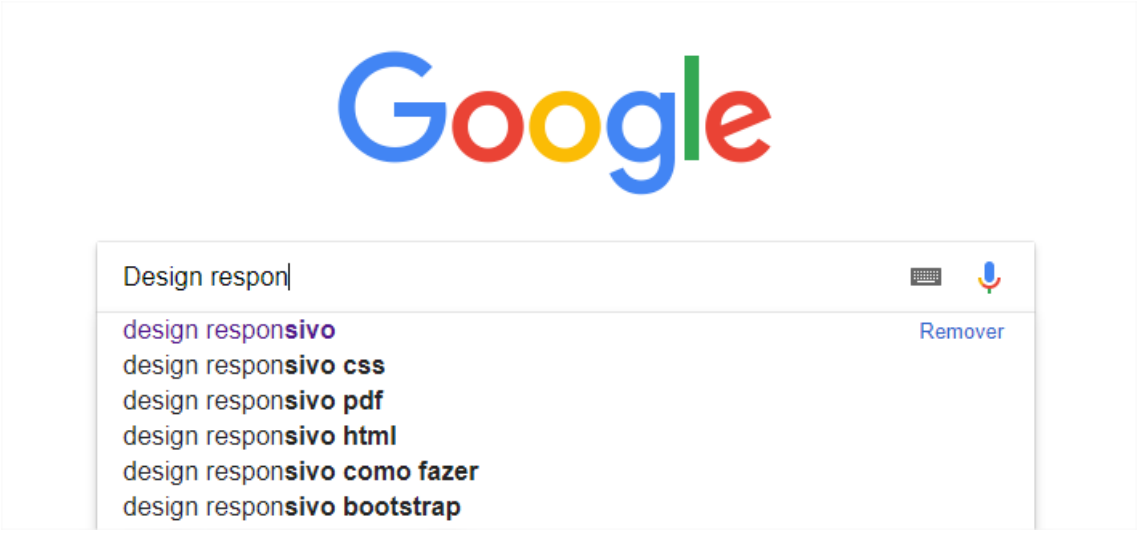
Figura 5 – Caixa de comentários do StackExchange



Fonte: Captura de tela da caixa de comentários do StackExchange

O usuário precisa ter conhecimento de como preencher um formulário a ponto de não receber mensagens de erro após o preenchimento. Segundo o exemplo citado por Bruno (2016) temos o Google, que oferece sugestões de buscas antes de apresentar algum erro para o usuário, em decorrência de uma digitação incorreta do termo de busca, conforme visto na figura 6.

Figura 6 – Sugestões de busca do Google



Fonte: Captura de tela de uma busca do Google

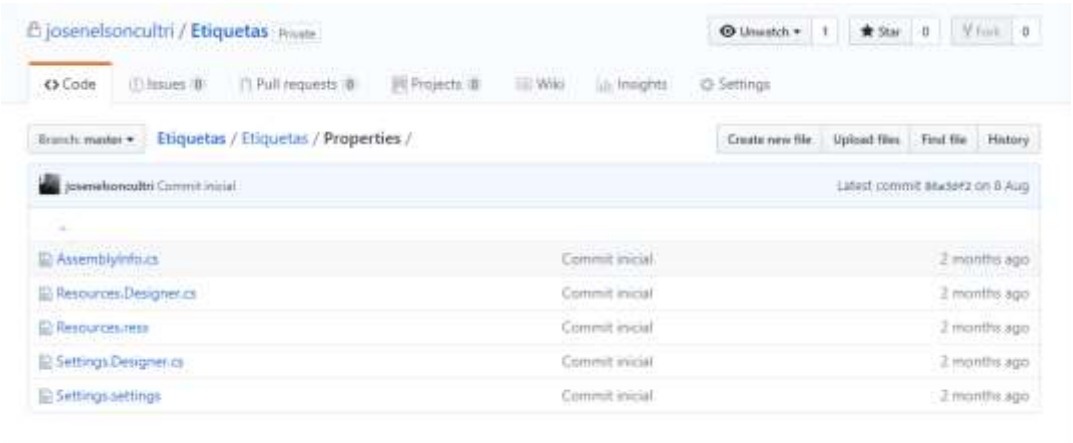
A mesma busca, sendo realizada com algum erro de digitação, retornará a mensagem apresentada na figura 7.

Figura 7 – Busca realizada com erro de digitação

Fonte: Captura de tela de uma busca do Google feita com uma digitação incorreta

Um sistema *web* precisa indicar claramente como o usuário pode navegar entre suas páginas, porém, pode ser que o caminho utilizado por um usuário para chegar a uma página em determinado *site* seja extenso, ou então ele não esteja habituado a fazê-lo por não visitar a página regularmente. É necessário, portanto, que o *site* forneça ferramentas para que a pessoa que o está utilizando no momento saiba aonde está e como chegou naquela página. Para isso é muito comum a utilização de *breadcrumbs*, ou seja, elementos que demonstrem o caminho utilizado para se chegar naquela página. Um excelente exemplo é o GitHub, serviço *online* de hospedagem e versionamento de código. Dentro de um repositório pode-se ver claramente o usuário ao qual aquele repositório pertence e, ainda, qual o diretório atual que está sendo exibido dentro daquele repositório, funcionamento exemplificado na figura 8.

Figura 8 – Exemplo de repositório no GitHub



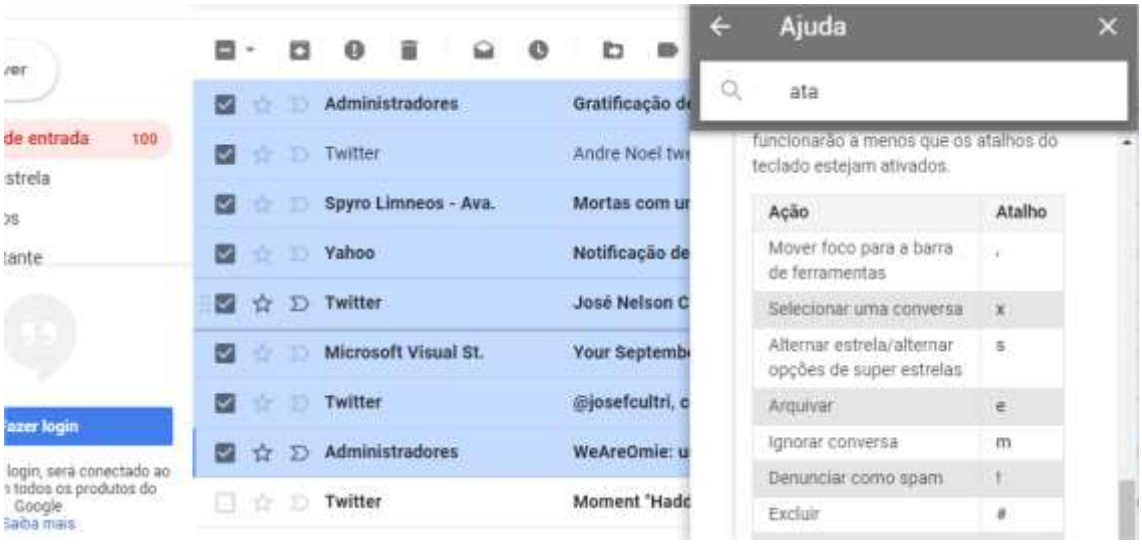
Fonte: Captura de tela de um repositório do GitHub

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

Na figura 8 pode-se visualizar claramente o usuário dono do repositório (“*josenelsoncultri”)*, o nome do repositório (“*Etiquetas”)* e a estrutura de diretórios sendo exibida (“*Etiquetas/Etiquetas/Properties”)*.

Produtividade é uma característica que todas as pessoas buscam possuir no desenvolvimento de suas tarefas diárias. Incluir atalhos no sistema, funções que façam o usuário otimizar seu tempo e concluir suas tarefas rapidamente é a premissa dessa heurística. Um bom exemplo é o Gmail que, através da navegação pelas setas e as teclas X e E do teclado, permite a seleção de vários *e-mails* recebidos e arquivamento, ou ainda selecionar a mensagem com X e realizar a exclusão diretamente digitando #. Os e-mails selecionados dessa maneira estão exemplificados na figura 9.

Figura 9 – Aplicativo Gmail com conversas selecionadas pela tecla X

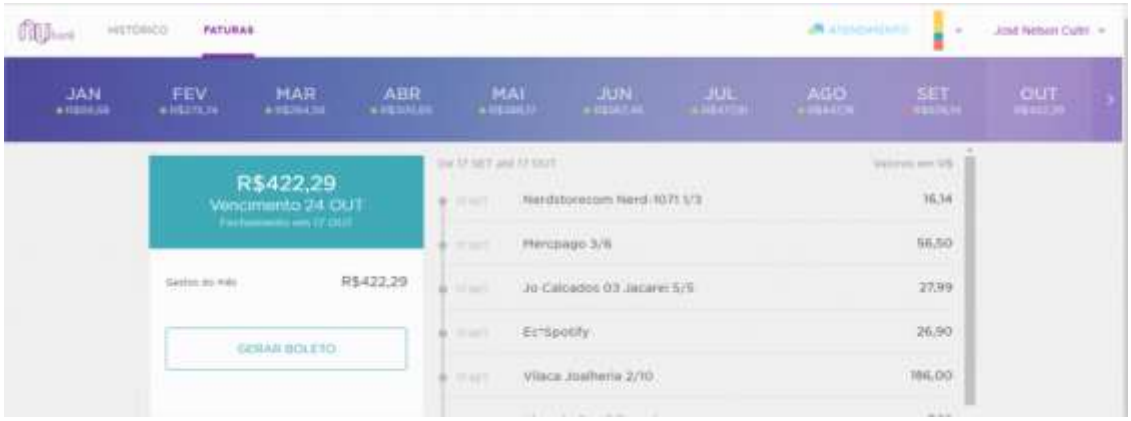


Fonte: Captura de tela de atalhos e *e-mails* do Gmail

Um *site* que possua um *layout* com muitos elementos, ou até com inconsistência quanto às cores utilizadas, pode incomodar os usuários. Quanto mais simples for o *layout*, melhor é para o usuário final. O usuário pode confundir-se com informações colocadas no *site* quando na verdade a intenção do desenvolvedor era apenas auxiliar no processo. Como citado por Bruno (2016), a Nubank é uma empresa que se preocupa bastante com essa heurística. Tomando por base a aplicação *web* de acesso aos dados do cartão de crédito, o processo de geração de um boleto para pagamentos é extremamente simples, como exemplificado na figura 10.

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

Figura 10 – Aplicação *web* do Nubank, tela de geração de faturas



Fonte: Captura de tela da página de gerenciamento de uma conta do Nubank

Com apenas três cliques é possível gerar o código de barras para pagamento da fatura em uma página com o *layout* bem limpo e objetivo, fácil do usuário identificar o que precisa fazer.

Entretanto, por mais simples que o sistema seja, ou por mais que existam elementos que auxiliem o preenchimento de um formulário, o usuário pode realizar um preenchimento incorreto. Neste caso é importante que o usuário saiba quais foram os erros cometidos e tenha o melhor auxílio possível na resolução dos mesmos sem muita poluição visual. Na figura 11 é apresentado o formulário de cadastro do Google e como ele mostra para os usuários que preencheram as informações de maneira incorreta, onde está errado e como corrigir o preenchimento.

Figura 11 – Formulário de cadastro da conta do Google

Fonte: Captura de tela da página de cadastro de uma Conta do Google

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

Todavia, por mais simples que um sistema seja, ainda pode ser necessária alguma explicação sobre alguns elementos da página, seja por complexidade de preenchimento ou por não conhecimento por parte dos usuários do sistema. O interessante é o desenvolvedor não precisar de se preocupar tanto com essa heurística, uma vez que um sistema simples e autoexplicativo não necessita de documentação para preenchimento dos formulários, mas caso realmente faça-se necessária a ajuda, o ideal é que a mesma fique próxima do campo em questão. Na figura 12 é exibida a tela de cadastro das informações de pagamento para a conta da Apple, bem como o auxílio para indicar ao usuário o lugar aonde está localizado o código de segurança do cartão.

Figura 12 – Tela de cadastro das informações de pagamento para a conta da Apple

Fonte: Captura de tela da página de cadastro do cartão de crédito da conta da Apple

3 Responsividade ou *design* responsivo

Atualmente, a gama de dispositivos pelas quais as pessoas acessam a *internet* é muito grande. Devido a isso os desenvolvedores devem ter a cautela de preparar seus *sites* para que sejam acessíveis sem distorções em vários tipos de dispositivos, com tamanho de telas variados. Segundo LePage (2018, *online*), “dispositivos móveis frequentemente são limitados pelo tamanho de suas telas e exigem uma abordagem diferente para o *layout* do conteúdo”. É nesse ponto que entram as preocupações com a chamada responsividade, ou *web design* responsivo.

Responsividade, segundo Teixeira (2011, *online*), implica em “programar um site de forma que os elementos que o compõem se adaptem automaticamente à largura de tela do dispositivo no qual ele está sendo visualizado”, portanto para que um *site* seja responsivo, precisa ter em seu código fonte as instruções necessárias para que suas páginas não fiquem distorcidas nos dispositivos que forem acessá-las.

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

Para construir um *design* responsivo podem ser utilizadas algumas técnicas básicas que serão descritas ao longo desta seção.

3.1 Janela de visualização (ou *viewport*)

Os navegadores de dispositivos móveis renderizam as páginas como se a mesma estivesse sendo acessada de um navegador *desktop*. Isso é feito para que todo o conteúdo seja exibido de uma vez para o usuário (LEPAGE, 2018).

Para que o navegador do *smartphone* (ou outro dispositivo móvel) exiba a página com um ajuste melhor para o usuário é necessário utilizar uma configuração específica para a *viewport*. Segundo Eis (2011, *online*):

> O viewport é a área onde seu website aparece. É a área branca da janela quando você abre o browser. O viewport sempre vai ter o tamanho da janela. Mas a forma como os elementos são renderizados vai depender bastante do dispositivo. Em máquinas desktop nós não precisamos nos preocupar muito, já estamos acostumados com um determinado tamanho de tela e resolução média utilizada pelos usuários. Mas quando começamos a variar muito o tamanho das telas, a largura do viewport começa a ser uma preocupação porque afeta diretamente a forma como o usuário utiliza seu website. O ponto é que em uma tela pequena, com uma resolução muito grande, por exemplo como as telas dos iPhone e da maioria dos smartphones de hoje, o conteúdo pode aparecer muito, muito pequeno. Imagine uma resolução FULL HD dentro de uma tela bem menor que uma TV… Por isso precisamos fazer com que o viewport mostre o conteúdo se baseando pelo tamanho da tela e não da resolução.

O desenvolvedor pode prover informações sobre a página para que o navegador a renderize de maneira correta, ou para que mecanismos de busca possam indexar melhor o *site*. Para isso devem ser utilizadas as chamadas *metatag*s, que são *tags* HTML (do inglês *HyperText Markup Language*, linguagem de marcação de hipertexto) utilizadas para prover algum tipo de informação da página para o navegador, mas sem serem exibidas na página renderizada para o usuário final.

Manipular, portanto, a *metatag* referente à *viewport* é imprescindível para que um *layout* fique bem apresentável. A metatag *viewport* tem um padrão mostrado na figura 13.

Figura 13 – *Metatag viewport*

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Fonte: Informações da *metatag viewport*[3]

[3] Disponível em <https://developers.google.com/web/fundamentals/design-and-ux/responsive/?hl=pt-br/>. Acesso em: 14 set. 2018

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

Quando se informa na *metatag* o atributo “width=device-width”, o navegador entende que deve considerar a largura para exibição da página sendo a mesma do dispositivo, o que evita uma distorção para apresentação do conteúdo todo de uma vez no dispositivo. Além disso, o atributo “initial-scale=1” define uma relação de 1 para 1 entre os *pixels* das unidades do CSS (do inglês *Cascading Style Sheets*, folhas de estilo em cascata) e os *pixels* dos dispositivos, permitindo assim uma melhor adaptação do *layout* para os dispositivos, independente da orientação.

3.2 Unidades relativas

Trabalhar definindo as medidas em *pixels* pode ser um problema para o desenvolvedor, uma vez que o conteúdo fica limitado àquele tamanho, sem ajustar-se mediante o dispositivo e a resolução no qual o site está sendo acessado. Segundo Eis (2012, *online*):

> Antigamente definir unidades de texto em pixels trazia uma desvantagem por causa do Internet Explorer. Quando o usuário tentava mudar o tamanho do texto pelo browser, por algum motivo bizarro o IE não aumentava esse texto pelo simples motivo de que o texto estava definido em pixels. Um problema sério de acessibilidade. É por isso que muitos devs preferiram durante um tempo definir o tamanho do texto utilizando % (porcentagem) em vez de trabalhar com pixels.

Utilizar *pixels* é um processo que exige do desenvolvedor saber exatamente o que será desenvolvido e exige também um teste muito bem apurado em vários dispositivos, com resoluções e comportamentos diferentes. Para isso existem as unidades relativas. Segundo Silva (2016, *online*) as unidades de medidas relativas:

> São medidas calculadas em relação a uma outra unidade de medida. Usar unidades de medidas relativas é mais apropriado para se obter ajustes em diferentes tipos de mídia. (por exemplo: ajustar de uma tela de monitor para uma impressora laser).

Existem diversas unidades relativas que podem ser utilizadas pelos desenvolvedores. Para entende-las é necessário levar em conta a hierarquia das *tags* HTML dentro de uma página. Elemento raiz sempre será o elemento *body* da página, aparecendo uma única vez e englobando todas as *tags* que serão exibidas para o usuário final, elemento pai é aquele imediatamente superior à *tag* a ser analisada e elemento filho é a própria *tag*, ou seja, se a *tag* a ser analisada é um parágrafo que está dentro de uma tag *span*, elemento filho será o parágrafo, elemento pai a tag *span*

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

e elemento raiz será o *body* da página. A seguir estão destacas e exemplificadas algumas delas (SILVA, 2016).

- EM: segundo Lopes (2014, *online*), “O valor é calculado levando sempre em consideração o *font-size* do elemento pai”, portanto se um elemento está com sua propriedade *font-size* definida como 2em e seu elemento pai está com um tamanho de fonte definido como 16 *pixels*, o tamanho da fonte do elemento filho será de 32 *pixels*. Esta unidade de medida é interessante, pois caso seja necessária a alteração das propriedades em todos os elementos basta que a alteração seja feita uma única vez no elemento pai e, a partir dele, todos os elementos filhos terão suas propriedades ajustadas automaticamente.
- REM: trabalhar definindo as medidas na unidade EM é trabalhoso, uma vez que exige do desenvolvedor um cálculo para cada elemento definido. Pensando nisso foi criada a unidade REM, aonde o “r” significa *root*, ou seja, para o cálculo da medida será sempre tomado como referência o valor do elemento raiz do documento, ou seja, será sempre assumida a unidade definida para o *body* da página (EIS, 2012).
- VW e VH: 1vw equivale a 1% da largura da *viewport*. Da mesma forma, 1vh equivale a 1% da altura da mesma. Como a *viewport* varia de dispositivo para dispositivo, a utilização dessas unidades é uma excelente maneira de adaptar larguras e alturas para os vários tamanhos de tela disponíveis.

3.3 *Media queries*

O HTML foi criado com a ideia de ser portátil, ou seja, o mesmo código rodar em diversos tipos de dispositivos diferentes, porém cada dispositivo pode ter uma necessidade específica de adaptação dos estilos da página. Dessa forma surgiu a necessidade da criação de recursos de estilização específica para cada tipo de dispositivo, ou seja, para cada *media type*, sendo alguns deles os seguintes: *print* (quando a saída do HTML for uma impressão), *tv* (caso a saída seja uma televisão) e *screen* (caso o HTML seja renderizado em uma tela de computador), porém é necessário que o navegador de cada um dos tipos de dispositivos diferentes saiba que deve aplicar ou não o código CSS corretamente, e é nesse ponto que entram as *media queries*, que são condições definidas pelo desenvolvedor para aplicação do CSS nos dispositivos determinados. Segundo Teixeira (2013, *online*):

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

> Os problemas das media queries são que cada vez mais surgem dispositivos com diversos tamanhos e também com hardwares bem parecidos aos dos desktops, fazendo com que esses dispositivos tenham uma navegação quase igual aos dos desktops.

Um bom exemplo do uso de *media queries* está ilustrado na figura 14.

Figura 14 – Utilização de media queries

```
<link rel=”styleSheet” hef=”estilo.css” media=”screen and (color)” />
```

Fonte: Informações sobre a utilização das *media queries*[4]

Com o exemplo da figura 14, o arquivo estilo.css somente será aplicado se a página for exibida na tela do dispositivo e o mesmo seja um dispositivo cuja tela permita renderizar as cores além do preto e branco. Existem diversas regras que podem ser utilizadas em uma *media query*, algumas delas são: *max-width* (o código CSS em questão será aplicado caso a largura da área de renderização seja de, no máximo, o valor que estiver especificado nesta regra)*, max-height* (nesta regra o comportamento é o mesmo da regra *max-width,* mas referente à altura)*, max-device-width* (para esta regra o comportamento é o mesmo da regra *max-width*, porém levando em consideração a largura do dispositivo e não apenas a área de renderização disponível)*, max-device-height* (o comportamento é o mesmo da regra *max-device-width*, porém referente à altura) e *orientation* (diz respeito à orientação da *viewport*, ou seja, *portrait* caso a *viewport* seja mais alta do que mais larga, e *landscape,* caso a *viewport* seja mais larga do que mais alta).

3.4 Testes do *site* em vários tamanhos diferentes

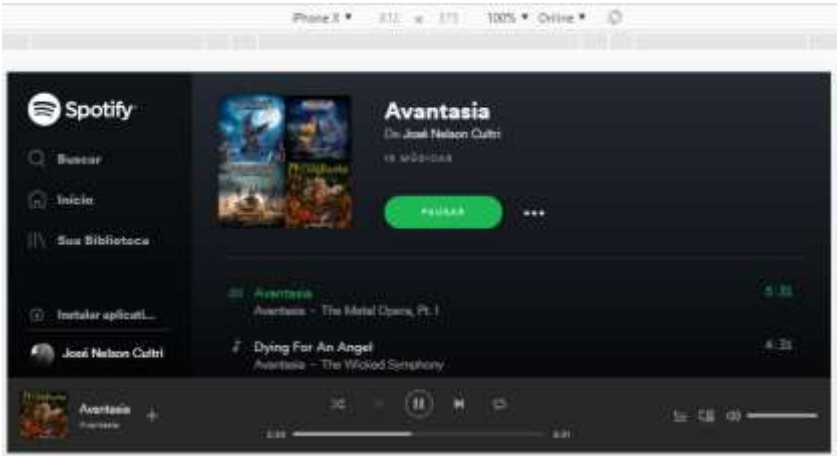
Com vários dispositivos existentes no mercado, acessar todos eles fisicamente para realizar testes reais é praticamente impossível. Por isso se faz necessário o uso de ferramentas que simulem os dispositivos em questão, a fim de tornar o trabalho do desenvolvedor o mais preciso possível. Nativamente alguns navegadores já possuem essa ferramenta de simulação de telas, como o Google

[4] Disponível em < https://www.devmedia.com.br/utilizando-css-media-queries/27085/>. Acesso em: 19 set. 2018

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

Chrome. A figura 15 ilustra o aplicativo *web* do Spotify, sendo sua abertura simulada para um iPhone X em modo paisagem.

Figura 15 – Abertura do Spotify simulada em um iPhone X



Fonte: Captura de tela do Spotify aberto em uma simulação de tela do iPhone X

Utilizando o recurso das ferramentas de desenvolvedor do Chrome é possível identificar problemas básicos de ajuste dos elementos em dispositivos diferentes.

4 Por que tomar cuidado com responsividade e usabilidade?

A utilização de *smartphones* para tarefas diárias não é mais um fato distante, é uma realidade. Fetter (2016, *online*) apresenta dados sólidos baseados em uma pesquisa conduzida pelo Google sobre o comportamento e as atitudes do consumidor digital:

- 89% dos usuários utilizam smartphones para procurar informações, serviços e produtos de empresas locais;
- 32% dos usuários ficam frustrados quando um site não é exibido corretamente em dispositivos móveis;
- 30% dos consumidores no Brasil têm feito compras utilizando dispositivos móveis.

Nas imagens 16 e 17 são apresentados dois *sites*, um que teve o *design* adaptado para que fosse responsivo e o outro não, respectivamente.

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

Figura 16 – *Site* das Lojas Americanas simulado em um iPhone 7

Fonte: Captura de tela do *site* das Lojas Americanas aberto em uma simulação de tela do iPhone 7

Figura 17 – *Site* do portal Deviantart simulado em um iPhone 7

Fonte: Captura de tela do portal Deviantart aberto em uma simulação de tela do iPhone 7

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

Segundo Fetter (2016, *online*), “13% dos brasileiros usam os celulares para comparar preços, enquanto visitam as lojas físicas”. Esta é uma realidade muito presente na vida das pessoas, visto que várias ainda não confiam totalmente em realizar compras pela *internet* ou então preferem ver os produtos em sua frente na loja, mas não querem ir até ela para conhecer o preço de um produto, ou seja, vão até a loja já sabendo os preços uma vez que já pesquisaram, o que torna a tarefa de compra mais prática.

Portanto, com base nas capturas de tela, nos dados e experiência apresentados, é possível dizer que uma empresa que deseja alcançar o maior público possível e disponibilizar seus serviços com mais eficiência deve levar em conta o desenvolvimento de seu *site* baseado em um *design* responsivo. Fetter (2016, *online*) ainda destaca que “se o seu produto ou serviço é voltado para um público mais jovem, fazer este investimento é essencial – já que 48% dos usuários de internet no Brasil têm entre 18 e 34 anos”.

Em se tratando da técnica de *design* responsivo é necessário tomar cuidado com a implementação da mesma. Uma vez que a ideia é que o *site* rode com o mesmo código em todos os tipos de dispositivos, esse mesmo *site* poderá ficar carregado com arquivos desnecessários, visto que a codificação leva em conta conteúdos que para modelos *desktops* são leves mas onera-se muito o carregamento quando portado para *smartphones* ou *tablets* (TEIXEIRA, 2014a).

Um site que não possua também cuidados com a experiência final do usuário terá um impacto muito prejudicial para o dono do mesmo. De acordo com Teixeira (2014b), “segundos a mais no carregamento de um *site* podem significar frustração para os usuários e menores taxas de conversão para empresas”. Para o usuário final, a demora no carregamento de uma página pode indicar um erro, que enquadraria na heurística de usabilidade referente à produtividade descrito na seção 2 deste artigo, ou seja, seria uma falha na eficiência. Quando um usuário pesquisa algo no Google, o tempo de demora no carregamento da página é levado em conta, ou seja, quanto mais pesada a página for pior o *ranking* recebido e por consequência menos pessoas encontrarão o site daquela empresa (RENKEL, 2017).

Entretanto, não apenas o responsável pela lentidão é o *design* responsivo mal implementado. O código a ser implementado para atingir a responsividade pode ser custoso em termos de tamanho sem a possibilidade de uma redução do mesmo para

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

que os arquivos fiquem menores. Para isso a equipe de desenvolvimento de um *site* e que mantém um portal pode otimizar as configurações dos servidores de aplicação no que tange à *cache*, ou seja, fazer com que esses arquivos pesados sejam distribuídos de uma forma mais rápida e não precisem ser reprocessados constantemente (o que demanda tempo de conexão e impacta na *performance* de acesso do usuário final) e também a melhoria do banco de dados da aplicação, através de técnicas de *tuning*, ou seja, a melhoria do tempo de execução das consultas que o sistema realiza (RENKEL, 2017).

Uma boa técnica que pode auxiliar na otimização é a minificação dos arquivos JavaScript e CSS. Essa técnica consiste em pegar todos esses arquivos do *site* e remover os espaços em branco, bem como comentários, a fim de reduzir o tamanho do arquivo e, consequentemente, diminuir o tempo de carregamento das páginas para os usuários finais. Por mais simples que esse processo possa parecer, e por mais fácil que seja devido à imensa variedade de ferramentas disponíveis para realização dessa tarefa, todo e qualquer tempo reduzido é importante, visto que para o usuário final a melhora será importantíssima, pois pode significar a conversão de uma visita em uma venda ou um *marketing* negativo para a empresa, uma vez que as pessoas que perguntarem para aquele usuário sobre o *site* ouvirão coisas ruins sobre a empresa e nem chegarão a visitar o *site*.

Considerações Finais

Um dos objetivos desse artigo é servir como um guia inicial de pesquisa para que os desenvolvedores *web* possam saber por onde iniciar suas buscas pelos assuntos centrais (*design* responsivo e usabilidade) do mesmo e também para que pessoas que não sejam necessariamente desenvolvedores possam ter uma noção inicial sobre os assuntos.

Neste artigo foram abordadas características de usabilidade e do *design* responsivo. Além disso foram apresentados alguns dos recursos a serem utilizados para implementação dessas técnicas de desenvolvimento de aplicações *web*.

A perspectiva final a ser observada com este artigo é de que não se pode mais desenvolver um *site* ou aplicação *web* sem levar em conta os conceitos de usabilidade

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

e *design* responsivo. Qualquer desenvolvedor pode encontrar aqui um ponto de partida, devido às informações apresentadas bem como as experiências relatadas.

O futuro está nos dispositivos móveis. A maioria das pessoas utiliza os *smartphones* para muitas tarefas do dia-a-dia. Desenvolvedores que não se preocuparem com a velocidade de suas páginas e a apresentação de seus *sites* não se manterão no mercado.

Referências

BRUNO, Marco. **10 Heurísticas de Nielsen. Uma fórmula para evitar erros básicos de usabilidade**, 2016. Disponível em <http://blog.caelum.com.br/10-heuristicas-de-nielsen-uma-formula-pra-evitar-erros-basicos-de-usabilidade/>. Acesso em: 12 set. 2018

EIS, Diego. **Manipulando a metatag viewport**, 2011. Disponível em <https://tableless.com.br/manipulando-metatag-viewport/>. Acesso em: 16 set. 2018.

EIS, Diego. **Qual unidade utilizar – Pixel, EM ou REM**, 2012. Disponível em <https://tableless.com.br/unidade-pixels-em-rem/>. Acesso em: 17 set. 2018

FETTER, Vanessa. **4 motivos para ter um site responsivo**, 2014. Disponível em < https://www.hostgator.com.br/blog/motivos-para-ter-um-site-responsivo/>. Acesso em: 26 set. 2018.

JOBS, Steve, 2011. Disponível em: <http://www.administradores.com.br/frases/steve-jobs/1040/>. Acesso em: 04 set. 2018.

LEPAGE, Pete. **Princípios básicos do web design responsivo**, 2018. Disponível em: <https://developers.google.com/web/fundamentals/design-and-ux/responsive/?hl=pt-br>. Acesso em: 14 set. 2018.

LOPES, Sérgio. **Porque usar 'em' no seu CSS hoje em dia?**, 2014. Disponível em < http://blog.caelum.com.br/porque-usar-em-no-seu-css-hoje-em-dia/>. Acesso em: 18 set. 2018.

NIELSEN, Jakob. **Usability 101: Introduction to Usability**, 2012. Disponível em <https://www.nngroup.com/articles/usability-101-introduction-to-usability/>. Acesso em: 10 set. 2018.

PAGANI, Talita. **O que é usabilidade?**, 2011. Disponível em: <https://tableless.com.br/o-que-e-usabilidade/>. Acesso em: 04 set. 2018.

RIBEIRO, Alexandra Oliveira. **O que é usabilidade? (Web. UX).** 2016. Disponível em: <https://pt.linkedin.com/pulse/o-que-%C3%A9-usabilidade-web-ux-alexandra-oliveira-ribeiro/>. Acesso em: 07 set. 2018.

Revista EduFatec: educação, tecnologia e gestão
V.1 N.2 julho-dezembro/2018

SANTANA, Flávio. **10 dicas de como começar um design responsivo**. 2015. Disponível em: <https://designculture.com.br/10-dicas-de-como-comecar-um-design-responsivo>. Acesso em: 06 set. 2018.

SILVA, Maurício Samy. **As unidades de medidas CSS.** 2016. Disponível em <http://www.maujor.com/tutorial/unidades-de-medidas-css.php>. Acesso em: 17 set. 2018.

TEIXEIRA, Fabricio. **O que é Responsive Web Design?**, 2011. Disponível em <https://brasil.uxdesign.cc/o-que-%C3%A9-responsive-web-design-ab292eb616b7/>. Acesso em: 15 set. 2018.

TEIXEIRA, Fabricio. **A importância da performance das páginas na experiência do usuário**. 2014a. Disponível em <https://brasil.uxdesign.cc/a-import%C3%A2ncia-da-performance-das-p%C3%A1ginas-na-experi%C3%AAncia-do-usu%C3%A1rio-36c375876e87/>. Acesso em 21 set. 2018.

TEIXEIRA, Fabricio. **Quando o Design Responsivo impacta a performance (e a UX)**. 2014b. Disponível em <https://brasil.uxdesign.cc/quando-o-design-responsivo-impacta-a-performance-e-a-ux-669cab19549e/>. Acesso em 24 set. 2018.

TEIXEIRA, José Ricardo. **Utilizando CSS Media Queries**. 2013. Disponível em <https://www.devmedia.com.br/utilizando-css-media-queries/27085/>. Acesso em: 19 set. 2018.

RENKEL, Gustavo. **Seu e-commerce carrega em menos de 2 segundos?**, 2017. Disponível em <https://www.ecommercebrasil.com.br/artigos/seu-e-commerce-carrega-em-menos-de-2-segundos/>. Acesso em 25 set. 2018.