Faculdade Senac Goiás

Curso de Gestão de Tecnologia da Informação

Fundamentos de TI

HTML5

Professora Orientadora: Kelly

Acadêmico: Henrique Sousa e SIlva

. INTRODUÇÃO

O HTML5 é a nova versão do HTML4 e um dos seus principais objetivos é facilitar a manipulação dos elementos, possibilitando o desenvolvedor modificar as características dos objetos de forma não intrusiva, fazendo com que isso fique transparente para o usuário final.

Para se ter uma ideia disso, diferente das versões anteriores, o **HTML5 fornece ferramentas para o CSS e o Javascript fazerem seu trabalho da melhor possível** de forma que um web site ou aplicação continue leve e funcional.

Algumas tags foram modificadas, outras criadas e algumas descontinuadas. As versões anteriores do HTML não eram padronizadas para criação de seções comuns e específicas como rodapé, cabeçalho, slidebar, menus etc.

Houve também modificações na forma em que escrevemos o código e organizamos a página. Ela passou a ser mais semântica com menos códigos, aumentando a interatividade sem a necessidade de instalação de plug-ins, que em alguns casos, causa perda de performance. É um código interpolável, ou seja, pronto para futuros dispositivos, facilitando a reutilização da informação de diversas maneiras.

Mas, caso você esteja pensando “vou ter de refazer todo o meu web site”, a WHATWG tem mantido o foco na retro compatibilidade, ou seja, você não irá precisar refazer todo o trabalho para se adequar aos novos conceitos e regras.

. VANTAGENS E DESVANTAGENS

As vantagens do HTML5 são:

1. Torna a internet mais rápida: desenvolvedores vem usando HTML5 para reduzir o tamanho de arquivos e tornar a experiência do usuário mais limpa.

2. HTML5 tem Canvas que é uma poderosa ferramenta para a criação de conteúdo gráfico. Com Canvas é mais fácil criar animações, desenhos e outros elementos visuais complexos sem usar aplicativos externos.

3. Cada vez mais navegadores dão suporte ao HTML5.

4. Um dos grandes diferenciais do HTML5 é sua capacidade de vídeo. Desenvolvedores podem integrar vídeos e HTML5 sem a necessidade de plugins. E como temos cada vez mais navegadores que são compatíveis ao HTML5, vemos um rápido crescimento na porcentagem de vídeos online.

5. O mesmo acontece com sons. HTML5 vem com uma ferramenta de suporte poderosa para elementos de audio.

6. HTML5 está disponível em diversas plataformas e sua performance é ótima em PC's, dispositivos móveis e tablets.

As desvantagens do HTML5 são:

1. A possibilidade de armazenamento local de dados local do HTML5 melhora imensamente a capacidade de usar aplicativos Web no modo offline. O único problema é a sincronização de dados.
Se um aplicativo Web está conectado à Internet, pode sempre salvar os dados para a nuvem. Quando está offline, as alterações não são sempre armazenados na nuvem. Quando alguém muda de navegador ou usa uma máquina diferente, cópias começam a proliferar e as dificuldades de sincronização também.

Os desenvolvedores devem se preocupar em fornecer a interface para que o usuário possa lidar com a sincronização. A especificação HTML5 não oferece qualquer ajuda.

Programadores gerenciar essas dores de cabeça utilizando sistemas de controle de versão, que se tornaram cada vez mais sofisticados para lidar com esse problema. No entanto, apenas ter a tecnologia não significa que ela seja de fácil uso. A fusão de vários repositórios GIT pode levar tempo.

2. O usuário não pode ter controle sobre os dados armazenados localmente, mas o site central também pode ter problemas por causa com a sincronização e até mesmo com a segurança desses dados.

Não há como o desenvolvedor garantir que o banco de dados local nunca será manipulado pelo usuário. Embora não existam ferramentas que tornem a edição dos dados locais e/ou a atualização de privilégios, tarefas fáceis para os usuários, ainda assim elas podem vir a ocorrer. Brechas de segurança do código JavaScript, por exemplo, são apenas um dos caminhos possíveis para tal.

. TIPOS DE APLICAÇÕES QUE POSSUEM O HTML5 COMO BASE

1. Landing Pages e Hotsites

Já ouviu falar em Landing Pages e Hotsites? Ambos os termos se referem a sites que possuem apenas uma página e visam objetivos específicos:

Landing Pages são muito utilizadas em ações de marketing digital. Nesse formato de página, o objetivo final geralmente é fazer com que o usuário preencha um

formulário em troca de algo. Esse algo pode ser uma promoção, um ebook, acesso a um webinar, enfim. Não há limites para a criatividade. Nesse cenário, o usuário fica feliz, pois conseguiu acessar o material desejado e a empresa também atinge o seu objetivo, pois conquistou o usuário para realizar futuras ações de Marketing. Nós, por exemplo, possuímos diversas Landing Pages aqui na Becode, dê uma olhada.

Já, os Hotsites, são páginas temporárias ou pequenos websites que não necessariamente possuem o objetivo de levar o usuário a realizar uma ação (como é o caso das Landing Pages). Atualmente, Hotsites são muito utilizados para a divulgação de eventos e campanhas avulsas de Marketing (marcas grandes como Coca-Cola e Pepsi costumam fazer isso), entre outras aplicações possíveis. Entretanto, é inegável que os Hotsites são, em sua maioria, utilizados para eventos, visando a divulgação de programações e atrações temporais. Veja um exemplo de Hotsite abaixo.

Nos dois casos, o desenvolvimento dessas páginas é mais fácil e rápido, já que não existe muita complexidade em seu processo de criação. O desenvolvimento costuma ter um início, meio e fim descomplicado. Em outras palavras, tendo experiência e prática, é possível criar e finalizar tudo em um dia só. E, como prometido, trata-se de uma aplicação que, em projetos menores, pode ser criada apenas com HTML, CSS e JavaScript.

2. Email marketing

Já ouviu falar em email marketing? Então, trata-se de uma prática de Marketing Digital utilizada para divulgar e compartilhar conteúdo por email. Atualmente, muitas empresas usam isso para manter a comunicação em dia com o seu público-alvo.

Então, para criar um email marketing você precisa saber somente HTML e CSS. Isso mesmo, não é necessário possuir conhecimento em JS. Além disso, em muitos casos, nem HTML e CSS é necessário. Isso porque hoje em dia existem muitas ferramentas que fazem o trabalho de um desenvolvedor front-end (Mailchimp, Getresponse, RD Station…).

3. Blogs, Lojas Virtuais e Sites Completos

Quando estamos falando de lojas virtuais, blogs e sites mais completos, é necessário ter a ajuda extra de um serviço de CMS (Content Management System, ou Sistema de Gestão de Conteúdos), de um SGBD e de uma linguagem de back-end.

Continuando, se você usar um CMS amigável e fácil de instalar como o WordPress, o conhecimento que você precisará ter sobre cada uma dessas tecnologias é muito próximo a zero. Claro, isso para projetos mais simples. Mas sim, se você se considera autodidata, tenho certeza que você não terá problemas para lidar com o WordPress. Além disso, as hospedagens web mais populares já possuem o serviço de instalação automática do WordPress, o que facilita muito!

Resumindo, com a ajuda dessas tecnologias (sabendo muito pouco sobre elas), já será possível criar websites mais completos e até lojas virtuais. Nesses casos, o HTML, CSS e JavaScript irão servir como articuladores daquilo que o seu cliente vê. Todas as telas, ícones e imagens serão preparados através do front-end. E a parte mais técnica, como recebimento de pagamentos e cadastramento de produtos, ficará por conta do backend no sistema de CMS. Pode parecer uma sopa de letrinhas, mas acredite, é relativamente fácil de aprender!.

. ELEMENTOS SEMÂNTICOS

HEADER:

O `<header>` é utilizado para representar o cabeçalho de um documento ouão declarado no HTML. Nele podemos inserir elementos de `<h1>` a `<h6>`, até elementos para representar imagens, parágrafos ou mesmo listas de navegação.

SECTION:

O elemento `<section>` representa uma seção dentro de um documento e geralmente contém um título, o qual é definido por meio de um dos elementos entre `<h1>` e `<h6>`. Podemos utilizar o `<section>`, por exemplo, para descrever as seções/tópicos de um documento.

ARTICLE:

Utilizamos o elemento `<article>` quando precisamos declarar um conteúdo que não precisa de outro para fazer sentido em um documento HTML, por exemplo, um artigo em um blog. É recomendado identificar cada `<article>` com um título.

NAV:

O elemento `<nav>` é utilizado quando precisamos representar um agrupamento de links de navegação, que, por sua vez, são criados com os elementos `<ul>`, `<li>` e `<a>`.

ASIDE:

O elemento `<aside>` é utilizado quando precisamos criar um conteúdo de apoio/adicional ao conteúdo principal. Por exemplo, ao falar de HTML semântico,

podemos indicar ao leitor outros conteúdos sobre a linguagem HTML como sugestão de leitura complementar.

MAIN:

O elemento `<main>` especifica o conteúdo principal e, consequentemente, de maior relevância dentro da página. Para ser considerada bem construída, uma página deve apresentar apenas um conteúdo principal.

FIGURE:

O elemento `<figure>` é uma marcação de uso específico para a inserção de uma figura. Para incluir a descrição dessa figura, podemos utilizar o elemento `<figcaption>`.

FOOTER:

O elemento `<footer>` representa um rodapé de um documento, como a área presente no final de uma página web. Normalmente é utilizado para descrever informações de autoria, como nome e contato do autor, e data de criação do conteúdo.

SEMÂNTICA NO NÍVEL DO TEXTO

Além da semântica estrutural, o HTML nos permite descrever o significado de um conteúdo em nível de texto utilizando um conjunto de elementos semânticos. Assim, é possível, por exemplo, destacar os trechos de texto que devem receber algum tipo de destaque.

A:

A principal função do elemento `<a>` é descrever um link, conectando os diversos documentos de um site e permitindo a navegação por esse conteúdo. Normalmente esses documentos estão relacionados por compartilharem um assunto em comum.

Em:

O elemento `<em>` é utilizado quando desejamos enfatizar um trecho ou palavra no texto, indicando que ela contribui de forma mais relevante para o sentido/compreensão do conteúdo.

Strong:

O elemento `<strong>` também é utilizado para destacar uma parte do texto. Sua principal diferença em relação ao elemento `<em>` é que `<em>` pode alterar o propósito de uma frase, como vimos anteriormente.

Cite e q:

O elemento `<cite>` é utilizado para declarar que naquele trecho há uma citação, isto é, um trecho de texto que não foi escrito pelo autor do conteúdo. Normalmente utiliza-se o `<cite>` em conjunto com o elemento `<q>`, responsável por apresentar o conteúdo retirado de outra fonte.

Time:

O elemento `<time>` é utilizado para representar datas. Assim, caso seja necessário informar a data em que um conteúdo foi escrito, podemos declarar a tag `<time>` e acrescentar a ela o atributo datetime para escrever a data de forma padronizada.

. BIBLIOGRAFIA

https://support.wix.com/pt/article/as-vantagens-do-html5

https://www.devmedia.com.br/o-que-e-o-html5/25820

https://becode.com.br/aplicacoes-que-podem-ser-criadas-aprendendo-html-css-e-javascript/

http://idgnow.com.br/ti-corporativa/2011/08/17/html5-10-duras-verdades/#&panel1-1

https://www.devmedia.com.br/html-semantico-conheca-os-elementos-semanticos-da-html5/38065