



# Introdução a Web Design

Ewerton Mendonça



**Curso Técnico em Informática**

Educação a Distância

2019





## EXPEDIENTE

### **Professor Autor**

Ewerton Mendonça

### **Design Educacional**

Deyvid Souza Nascimento

Renata Marques de Otero

### **Revisão de Língua Portuguesa**

Letícia Garcia

### **Diagramação (2.ed. 2019)**

Jailson Miranda

### **Catálogo e Normalização**

Hugo Cavalcanti (Crb-4 2129)

### **Coordenação**

Anderson Elias

### **Coordenação Executiva**

George Bento Catunda

Terezinha Mônica Sinício Beltrão

### **Coordenação Geral**

Paulo Fernando de Vasconcelos Dutra

Conteúdo produzido para os Cursos Técnicos da Secretaria Executiva de Educação Profissional de Pernambuco, em convênio com o Ministério da Educação (Rede e-Tec Brasil).

**Maior, 2017**



**Dados Internacionais de Catalogação na Publicação (CIP) de acordo com ISDB**

---

M963i

Mendonça, Ewerton.

Introdução a Web Design: Curso Técnico em Informática: Educação a distância / Ewerton Mendonça. – 2.ed. – Recife: Escola Técnica Estadual Professor Antônio Carlos Gomes da Costa, 2019.

56 p.: il.

Inclui referências bibliográficas.

Material produzido em março de 2017 através de convênio com o Ministério da Educação (Rede e-Tec Brasil) e a Secretaria de Educação de Pernambuco.

1. Sites da Web - Projetos. 2. HTML (Linguagem de marcação de documentos). 3. Computação gráfica. I. Título.

CDU – 004.92

---



## Sumário

Introdução .....	6
1. Competência 01   Conhecer os Fundamentos de Web Design e HTML5 .....	8
1.1 Introdução a web design.....	8
1.1.1 Breve história da internet .....	8
1.1.2 World Wide Web.....	10
1.1.3 Padrões web.....	11
1.1.4 Design.....	12
1.1.5 Front-end e Back-end.....	13
1.1.6 A estrutura da internet .....	14
1.1.7 URL .....	16
1.1.8 URL Absoluta e Relativa .....	19
1.1.9 Ide.....	20
1.1.10 W3C e os padrões web.....	21
1.1.11 HTML .....	22
1.1.12 CSS.....	25
2. Competência 02   Formatar um Site Usando Técnicas Avançadas em HTML5 .....	28
2.1 Tag .....	29
2.2 Atributos .....	30
2.3 DOCTYPE.....	30
2.4 Estrutura básica do documento HTML .....	31
2.5 Visualizando as páginas.....	34
2.6 Nomenclatura dos arquivos HTML.....	35
2.7 Layout HTML .....	36
2.8 Atributos Id e Class.....	37
2.9 Novas TAGS HTML 5 .....	38



2.10 Iframe .....	40
2.11 TAGS textuais .....	41
2.12 Listas ordenadas e não ordenadas.....	42
2.13 Tabelas .....	43
2.14 Imagens .....	44
2.15 Formulário .....	45
2.16 Links.....	46
2.17 Finalizando .....	48
3.Competência 03  Formatar um Site com Recursos Multimídia .....	50
3.1 Vídeo .....	50
3.2 Áudio .....	52
3.3 Youtube .....	53
4.Competência 04  Conhecer os Fundamentos do CSS 3.....	56
4.1 Regras CSS .....	57
4.2 Seletores.....	58
4.3 Cores .....	60
4.4 Background .....	61
4.5 Text.....	62
4.6 Font .....	63
4.7 Links.....	64
4.8 List .....	65
4.9 Border .....	67
5.Competência 05  Planejar Layouts com CSS 3.....	69
5.1 Modelo de caixa .....	69
5.2 Width e Height .....	70
5.3 Position.....	71



5.4 Magin e Padding.....	73
5.5 Display .....	76
5.6 Finalizando .....	77
Conclusão .....	80
Referências .....	81
Minicurrículo do Professor .....	82





## Introdução

Manoel acorda cedo para ir para o trabalho e enquanto toma café consulta a web para saber como está o tráfego até lá. Infelizmente, houve um acidente e pelo visto não será possível chegar até a empresa para ler os relatórios, preparar uma apresentação e se encontrar com importantíssimos investidores. Ele, então, fala com sua secretária utilizando um aplicativo WEB e pede para preparar a sala de videoconferência. Depois, entra no sistema web da empresa e, através de uma conexão segura, consulta os relatórios procurando as informações importantes que precisa para convencer os investidores. O trânsito parece não mudar. O que será que aconteceu? Uma consulta em um site de notícias revela um acidente com um caminhão-tanque. O congestionamento deve durar horas. Paciência. Manoel, então, utiliza um aplicativo WEB para fazer uma apresentação online. Prepara alguns gráficos, coloca algumas imagens pesquisadas em um site de busca, acrescenta alguns textos, alguns efeitos, não muitos para não ficar cafona. A secretária entra em contato, informando que a sala está pronta. Manoel pergunta se precisa instalar algum programa para fazer a apresentação de casa, ela explica que não, o navegador comum é o suficiente, apenas será necessário baixar um *plug-in* e pronto. Em dois minutos tudo está funcionando. Em meia hora Manoel está na sala realizando, pela web, a apresentação que alavancará a empresa a um nível internacional.

Para você essa história pode não ser nada demais. Você até poderia reconhecer a maioria dos aplicativos que ele utilizou mesmo sem que eu lhe diga os nomes. Mas na década de 1990 a maioria destas facilidades nem eram imaginadas. Até o próprio conhecimento do que era a internet e o que ela poderia fazer pela gente era difícil de imaginar. Porém, temos dois tipos de usuários na rede: os consumidores e os criadores.

Independente da forma como utilizamos a internet, somos consumidores de seus serviços oferecidos. Seja para pesquisarmos no Google, solicitar um serviço como um transporte Uber, comprar um produto como um livro sobre WEB na Amazon e até arrumar uma companhia no Tinder. Eu, poderia utilizar as cem próximas páginas para descrever o que já existe e mais cem para o que poderá vir a surgir e não seria o bastante para descrever a riqueza da internet e da WEB. Internet e WEB? Não é a mesma coisa? Não é, e vamos entender melhor esta história. Antes disso, eu tenho que avisar que você está estudando para entrar no segundo tipo de usuário, o dos criadores. Aquele pessoal que faz o que você vê nos sites que anda. Chega de apenas consumir o que existe. Vamos aprender como fazer o que as pessoas vão consumir.



Preparados?

A primeira competência, **Conhecer os fundamentos de Web Design e HTML5**, tem a finalidade de embasar o que virá a seguir. Descobrir a diferença de internet e WEB. Conhecer um pouco da história de como tudo se desenrolou. Como funciona a arquitetura básica da internet e o que é o HTML 5.

A segunda competência, Formatar um site usando técnicas avançadas em HTML5.

A terceira competência, Formatar um site com recursos multimídia.

A quarta competência, Conhecer os fundamentos do CSS 3.

A quinta e última competência, **Planejar layouts com CSS 3**, finaliza comandos.

Espero que você se divirta muito aprendendo e sempre tenha em mente que a qualidade do profissional está nas horas dedicadas na produção. Então, repita os exemplos à medida que vai descobrindo, faça os exercícios propostos e modifique porque, afinal, você será um criador.



## 1. Competência 01 | Conhecer os Fundamentos de Web Design e HTML5

Esta competência tem o objetivo de fundamentar o conhecimento relativo à área de internet, especificamente web. Nela vamos descobrir o que é a internet, a diferença de internet e WEB, conhecer um pouco da história de sua criação, desvendar os meandros de sua arquitetura e descobrir o que é os padrões WEB na marca HTML5.

Então, não vamos mais perder tempo e embarcar, ou eu deveria dizer, surfar neste mar de conhecimento.

### 1.1 Introdução a web design

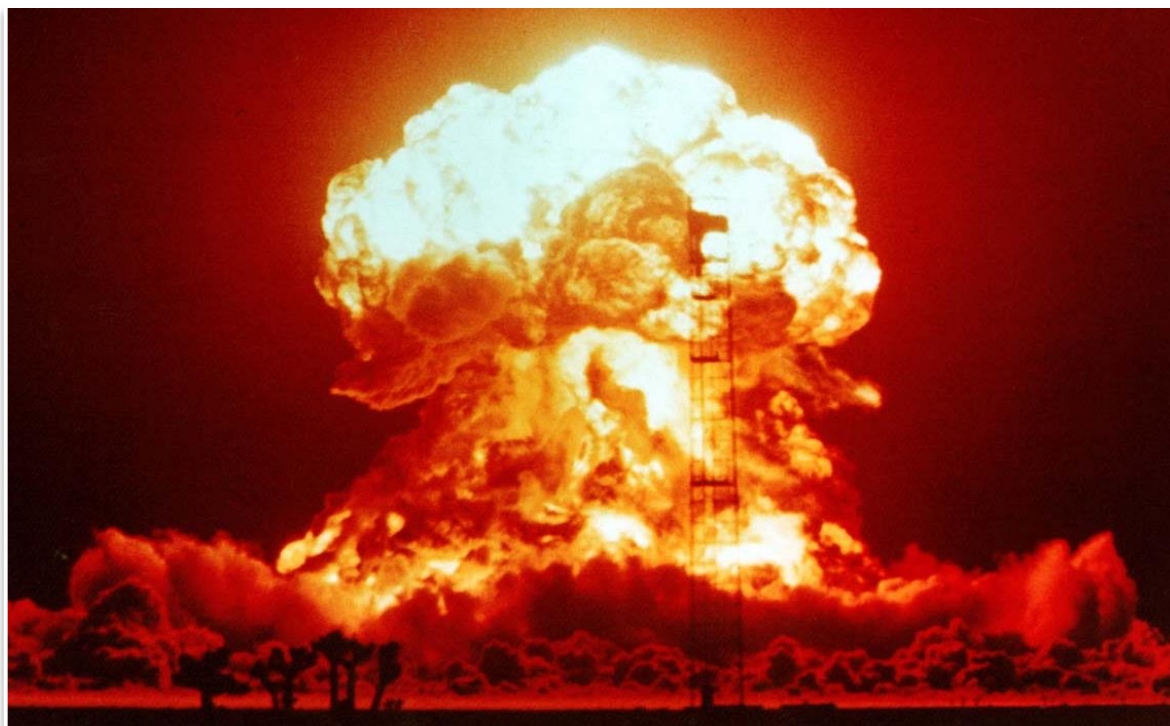
A internet forneceu um suporte para o mundo mudar. Essa tecnologia é considerada uma revolução de várias formas e seu impacto é bem mais profundo do que percebemos. Os mais novos podem até se perguntar: como era possível viver sem internet? É um novo mundo. Um mundo de bits, a menor unidade de informação do computador, que circula de computador para computador na chamada rede mundial de computadores.

Para entrarmos nesse mundo, vamos precisar conhecer um pouquinho de sua história. Caso você tenha curiosidade, deixaremos links para você se aprofundar nos assuntos.

Espero que se divirta aprendendo.

#### 1.1.1 Breve história da internet

Na década de 1960, o mundo passava pela chamada Guerra Fria. De um lado os Estados Unidos, líder do bloco capitalista, e do outro a União Soviética, líder do bloco socialista. Esses dois blocos possuíam armas atômicas e o receio na época era de que algum deles começasse uma guerra nuclear, daí o nome Guerra Fria.



**Figura 1 - Teste nuclear realizado em 18 de abril de 1953 na área de testes de Nevada.**

**Fonte:** [https://upload.wikimedia.org/wikipedia/commons/7/79/Operation\\_Upshot-Knothole\\_-\\_Badger\\_001.jpg](https://upload.wikimedia.org/wikipedia/commons/7/79/Operation_Upshot-Knothole_-_Badger_001.jpg)

**Descrição:** a imagem é uma fotografia de uma explosão nuclear resultado de um teste em Nevada.

No meio disso, os militares dos Estados Unidos tinham receio de que, caso houvesse uma explosão no território americano, ela impedisse a comunicação entre as bases militares. Era necessário um sistema de comunicação entre as bases que continuasse funcionando mesmo que uma fosse destruída. Assim, começou a pesquisa da internet na década de 60.

A internet é um sistema de comunicação entre computadores ao redor do planeta. Ela dá o suporte para a troca de dados entre os computadores que estão conectados a ela. E hoje temos computadores em casa, nos carros e constantemente conosco, através de smartphones.



Assista a quantidade de bombas atômicas que foram explodidas de 1945 a 1998 nesta animação. Você vai se impressionar.

**[www.youtube.com/watch?v=jfpQNfcRE1o](http://www.youtube.com/watch?v=jfpQNfcRE1o)**



Descubra mais sobre a internet.

Acesse [http://pt.wikipedia.org/wiki/Hist%C3%B3ria\\_da\\_Internet](http://pt.wikipedia.org/wiki/Hist%C3%B3ria_da_Internet)

## 1.1.2 World Wide Web

A World Wide Web ou teia mundial, também é conhecida por web ou www. Não é a mesma coisa de internet. A web foi criada em 1980, por um engenheiro do CERN chamado Tim Berners-Lee. Ele trabalhava em um lugar que possuía uma rede de computadores ligados à internet com vários cientistas trocando documentos diferentes entre si, só que os sistemas operacionais e, na maioria das vezes, um documento de uma máquina não abria em outra. Não existia uma forma comum, padronizada, para acessar os diferentes documentos.



**Figura 2 - Tim Berners Lee no Campus Party Brasil.**

**Fonte:** <http://revistagalileu.globo.com/Revista/Galileu2/foto/0,,45208415,00.jpg>

**Descrição:** Tim Berners Lee apresentando uma palestra em um evento no Brasil.

Para resolver esse problema, nas horas vagas, ele começou a escrever um projeto que usava a internet como base. Então, concluímos que internet e web não são as mesmas coisas. A web é um formato de documento HTML, com uma forma de endereçamento URL, e uma forma de entrega





desse documento, através de protocolos. Ex.: HTTP, HTTPS, entre outros. A internet é por onde isso tudo é utilizado.



**Assista à excelente palestra de Tim no TED. A palestra está em inglês, mas possui legendas em português. Vale muito a pena.**  
**[www.ted.com/talks/tim\\_berners\\_lee\\_a\\_magna\\_carta\\_for\\_the\\_web?language=pt-br#t-391210](http://www.ted.com/talks/tim_berners_lee_a_magna_carta_for_the_web?language=pt-br#t-391210)**

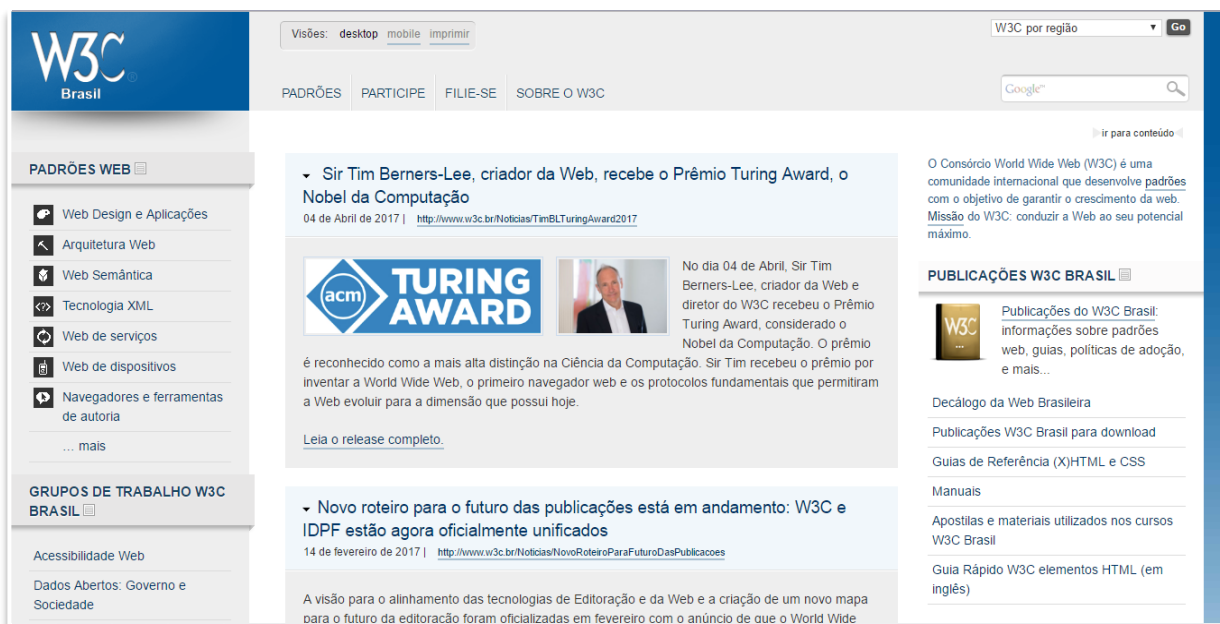


**Descubra mais sobre a web.**  
**Acesse [http://pt.wikipedia.org/wiki/WWW;./](http://pt.wikipedia.org/wiki/WWW;/)**

## 1.1.3 Padrões web

Perceba que o maior problema eram documentos de formatos diferentes vindos de computadores diferentes, a web padronizou tudo. Mas quando a web ficou popular no mundo, as pessoas começaram a modificá-la para adaptá-la às respectivas necessidades. Seria uma bagunça se não fosse uma instituição que cuida da uniformidade na web, garantindo, assim, que o documento que foi produzido do outro lado do mundo seja exibido da mesma forma em qualquer parte dele.

O W3C (World Wide Web Consortium) é a principal organização de padronização da web. Ela não é uma empresa, é um consórcio internacional com quase 400 membros, agrega empresas, órgãos governamentais e organizações independentes com o objetivo de estabelecer padrões para a criação e a interpretação de conteúdos para a Web. Hoje, a W3C cuida da reformulação do HTML em um projeto divulgado como HTML 5.



**Figura 3 - Site da W3C Brasil.**

**Fonte:** [www.w3c.br/Home/WebHome](http://www.w3c.br/Home/WebHome)

**Descrição:** captura de tela do site da W3C Brasil em 23 de abril de 2017.

## 1.1.4 Design

A maioria das pessoas entende de maneira errada a profissão de designer. Design não se traduz como desenho e sim como projeto. A palavra para desenho é draw. O designer projeta não só a parte visual, mas, principalmente, a organização e sua estrutura para a melhor experiência com o artefato. No caso, o web designer é especialista no planejamento e na produção de artefatos conhecidos como sites web.

O trabalho maior do web designer está antes de escrever o HTML ou de desenhar a identidade visual, que são tarefas finais do processo. Fazer um site sem planejar antes é como construir uma casa sem uma planta, sem planejar o encanamento, esgoto, parte elétrica, etc. Dá para perceber que o resultado é bem frustrante.

Então, tenha paciência e faça a parte inicial bem-feita. Assim, aumentaremos as chances de o site ser um sucesso.

## 1.1.5 Front-end e Back-end

A internet possui dois lados: o início e a chegada. Quando abrimos um navegador, aquele programa que visualiza páginas de internet, e solicitamos ver uma página, como a do Google, por exemplo, fazemos um pedido para outro computador, que guarda a página e nos envia.

Os computadores na internet que são especializados em servir as páginas que queremos ler são chamados de servidores. Na verdade, o servidor é um programa que funciona no computador, mas vamos simplificar.

Existem páginas que, não importa quantas vezes as solicitemos, sempre virá o mesmo conteúdo. Classificamos essas páginas como estáticas. A Figura 4 mostra uma página comum a muitos sites que normalmente é estática, a página “Sobre”.



**Figura 4 - página “Sobre” que apresenta informações sobre a que se destina o site.**

**Fonte:** [www.w3c.br/Sobre/](http://www.w3c.br/Sobre/)

**Descrição:** captura de tela da página sobre da W3C Brasil em 23 de abril de 2017.

Mas, às vezes, a página solicitada modifica, dependendo do que estamos fazendo. O carrinho de compra em um site de vendas online é uma página em que o conteúdo dela modifica dependendo de nossa interação no site de comércio online. São chamadas de páginas dinâmicas.





**Figura 5 - sites de e-commerce são sistemas que utilizam a WEB para oferecer produtos e serviços.**

**Fonte:** [www.amazon.com.br/](http://www.amazon.com.br/)

**Descrição:** página inicial do sistema de venda de livros da Amazon em que aparecem alguns livros disponíveis, bem como uma família feliz, representada por pais e filhos.

A diferença é que nas páginas estáticas fazemos uma vez seu conteúdo e ele será copiado toda vez que for requisitado. Nas páginas dinâmicas o conteúdo vai ser determinado por um software que produz a página a cada requisição.

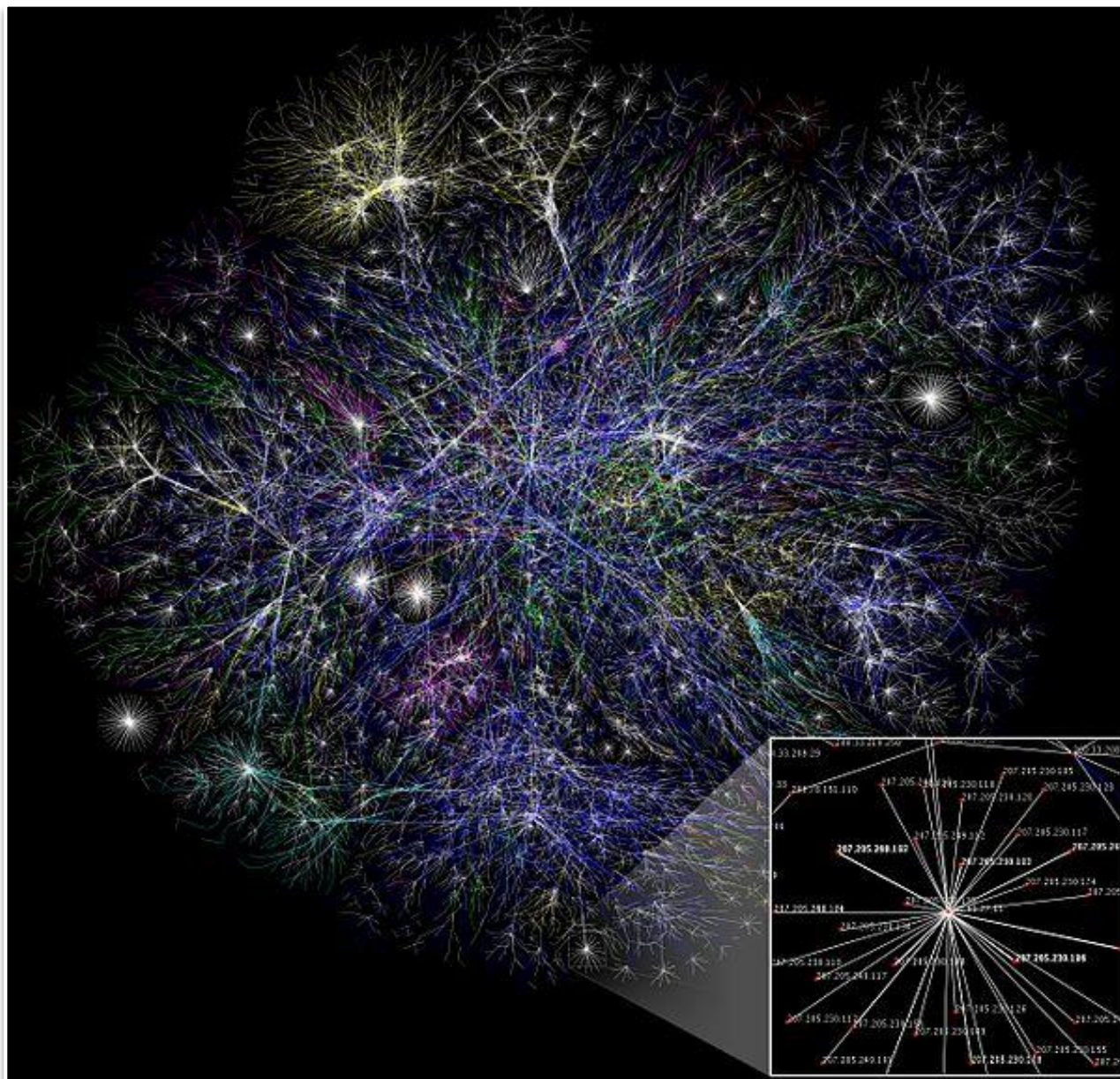
Esta disciplina é focada na criação de páginas de conteúdo estático, e a isso classificamos como desenvolvimento front-end. Quando temos que desenvolver uma aplicação que construirá as páginas de forma dinâmica, temos então o desenvolvimento back-end. Este tipo de desenvolvimento é mais difícil e para fazê-lo é muito recomendável que você conheça bem o desenvolvimento front-end primeiro.

## 1.1.6 A estrutura da internet

Agora, precisamos saber, por alto, como ela funciona. Precisamos entender como é que escrevemos um endereço de um site e como posteriormente a página correta aparece em nosso navegador. O que será que acontece no meio disso tudo?

Sabemos que a internet é uma rede de redes, e que cada rede que forma a internet pode possuir computadores, celulares, impressoras, ou até outras redes, como em uma lanhouse. É um

emaranhado de conexões que se assemelham a rodovias que ligam cidades e levam produtos e serviços de um lugar a outro.



**Figura 6 - mapa parcial da Internet em 15 de janeiro de 2005, baseado em dados da opte.org.**

**Fonte:** [www.opte.org/maps/](http://www.opte.org/maps/)

**Descrição:** a imagem mostra um gráfico de linhas coloridas conectadas.





**Para mais informações sobre a internet, acesse**  
**<http://pt.wikipedia.org/wiki/Internet>**

A estrutura da internet é classificada como cliente/servidor. É assim chamada porque aplicações, que são clientes, solicitam algo a aplicações servidoras. Tanto uma como outra funcionam em computadores, que estão em locais diferentes, mas conectados pela internet. Para que eles possam se comunicar precisam de duas informações: um endereço único e uma forma de comunicação. Assim, cada máquina possui um IP, Internet Protocol, como um CEP; e o TCP, Transmission Control Protocolo, as regras de como os dados circulam na rede.

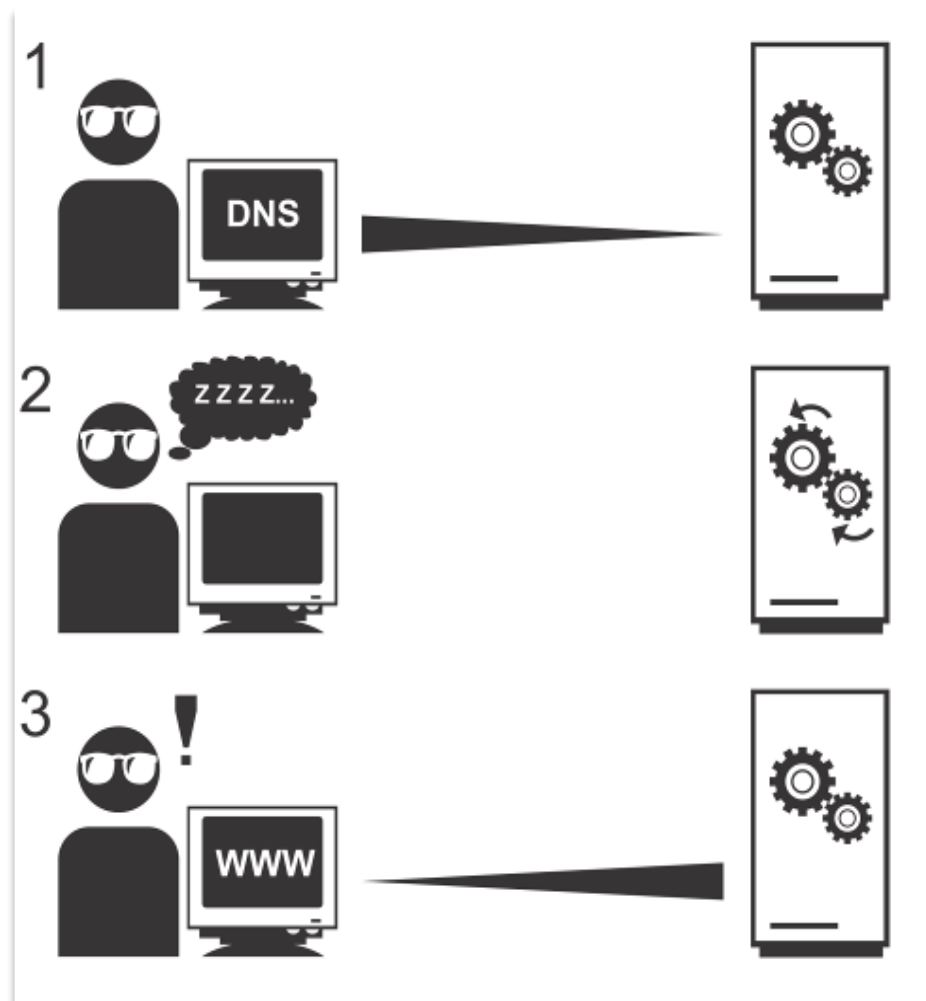


**Assista no vídeo a uma explicação sobre a estrutura da internet**  
**[www.youtube.com/watch?v=HNQD0qJ0TC4](http://www.youtube.com/watch?v=HNQD0qJ0TC4)**

Dessa forma, vários serviços podem utilizar a internet para funcionar. O e-mail é um deles. O e-mail é a troca de mensagens de forma assíncrona. Isso quer dizer que a pessoa que recebe um e-mail não precisa estar de prontidão, esperando em frente ao computador, para receber a mensagem, como acontece no telefone, que é síncrono. O serviço de e-mail tem aplicações clientes e aplicações servidoras que gerenciam a troca de mensagens.

## 1.1.7 URL

Vimos também, na competência 1, que a WWW não é igual à internet. A WWW utiliza a internet como meio de tráfego para serviços que servem às páginas dos sites. Assim, nós solicitamos uma página através de um endereço, acionamos um protocolo de comunicação com o aplicativo que fornece a página e nosso navegador lê o documento recebido e monta seus elementos para que possamos ver. Observe na Figura 19 o passo a passo.



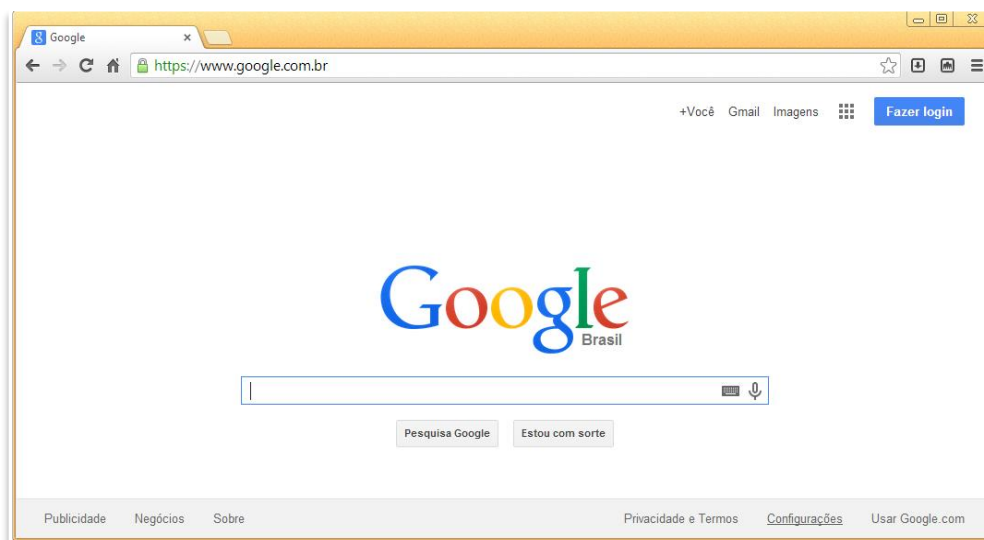
**Figura 7 - Ilustração que mostra os passos simplificados da solicitação de uma página web para uma aplicação.**

**Fonte:** Ewerton Mendonça.

**Descrição:** a ilustração, representada por bonequinhos e computadores, é dividida em três fases: a requisição do usuário para o servidor, o processamento do servidor e a entrega do resultado do processamento ao usuário.

As páginas ficam hospedadas em um servidor web. O mais popular é o Apache. Devemos contratar um servidor web para que possamos disponibilizar nosso site para o mundo. Além disso, precisamos de um endereço único para ele. Este endereço chama-se URL, **Uniform Resource Locators**.

Você percebeu que apesar do IP ser um número, não digitamos números no navegador para solicitar uma página a um servidor web, e sim a nomes. Bem mais fácil, né? Esse nome é a URL, um endereço para um determinado serviço.



**Figura 8 - observe a URL do Google na barra de navegação.**

**Fonte:** Ewerton Mendonça.

**Descrição:** navegador exibindo a página de busca da Google. A URL está na barra de navegação.

Observe na Figura 8 a URL do serviço de busca do Google para o Brasil. O endereço que está lá é o `https://www.google.com.br` e é composto de várias partes. Estas partes são como o nome de países, estados, cidades, bairros e ruas que o serviço dos Correios utiliza para entregar cartas e pacotes.

Vamos conhecer melhor estas partes?

- **https://** é o protocolo para tráfego de páginas web de forma segura. Quando não precisa de segurança é utilizado o “http://”;
- **www.** é o nome do host relacionado ao site. Existe uma discussão sobre a futura remoção desta parte. Você pode encontrar alguns URLs que não possuem esta parte;
- **google.** é o nome do domínio. Domínio é um conjunto de recursos relacionados;
- **com.** é a classificação do propósito do serviço. Podemos ter edu. para serviços de educação, org. para organizações não governamentais, entre muitos outros;
- **br/** refere-se à localização do serviço. No caso br. é Brasil e quando não possui localização refere-se aos Estados Unidos.

Estas partes são separadas por ponto e ao final temos uma barra. Após a barra temos o recurso. O recurso é algum arquivo do site, podendo ser uma página html, imagem, arquivo de vídeo, entre muitos outros, ou uma pasta onde está localizado o recurso.



É muito importante que você entenda muito bem essas partes. Observe alguns endereços web e identifique-as. Você terá que escrever muitos desses endereços nas páginas que fará.

## 1.1.8 URL Absoluta e Relativa

Como vimos anteriormente, um recurso é qualquer arquivo que o servidor envia para nosso navegador. Pode ser o próprio arquivo HTML, mas ele pode necessitar de arquivos CSS, imagens, JavaScript, vídeo etc. Cada um deles é um recurso e cada um deve possuir um endereço único.

Então, um arquivo HTML pode estar em:

- [www.meusite.com.br/menu.html](http://www.meusite.com.br/menu.html)

Que solicita um recurso de imagem que está em:

- [www.meusite.com.br/imagens/marca.jpg](http://www.meusite.com.br/imagens/marca.jpg)

Estas URLs são absolutas, ou seja, o endereço que localiza o recurso está completo. Agora, imagine se temos para este recurso menu.html dezenas de imagens, cada uma com uma URL diferente. O trabalho que vai dar repetir todo o início da URL, que é igual?! E se o cliente mudar o DNS do site para:

- <http://www.melhorsitedomundo.com.br/>

Você perderia seu fim de semana para fazer todas as modificações. Sem falar que não conseguiria testar seu site em sua máquina, que possui DNS:

- <http://localhost/projeto>

Para resolver isso, temos o conceito de URL relativa. Nesse caso, utilizamos o endereço do recurso atual, ou seja, [www.meusite.com.br/](http://www.meusite.com.br/) no caso do recurso menu.html, como base para o próximo recurso.

Desse jeito, colocamos apenas o que muda. Em nosso exemplo o recurso marca.jpg ficaria assim:

- imagens/marca.jpg

A parte anterior será copiada de menu.html. Bem mais simples.



E quando você for testar em casa, irá digitar o endereço:

- <http://localhost/projeto/menu.html>

Automaticamente, todos os recursos utilizarão a nova localização. Muito mais prático.

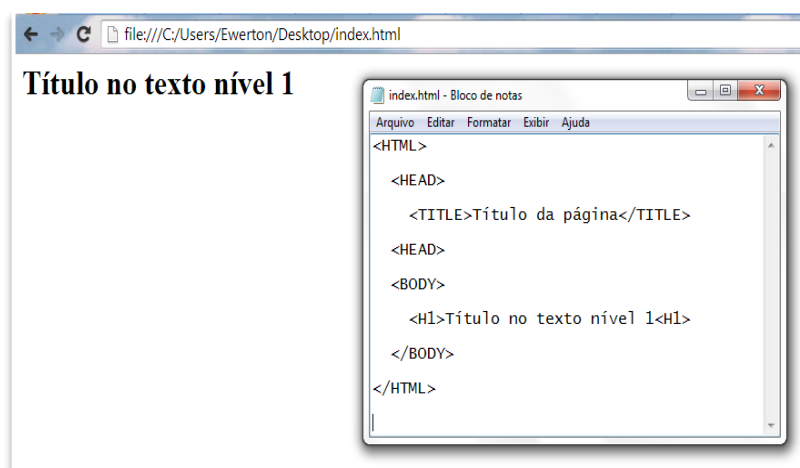
Este conhecimento é de extrema importância. Você deve entendê-lo perfeitamente para prosseguir nas próximas competências, porque é através dessas URL que ligamos uma página a outra com os hiperlinks. Se colocar o endereço errado, a página não será encontrada e o usuário receberá uma mensagem de erro.

## 1.1.9 Ide

Agora que você já conhece um pouco sobre a web, vamos colocar a mão na massa. Mas que programa podemos utilizar para construir páginas web?

Você pode fazer um site web completo utilizando apenas o Bloco de Notas, Notepad do Windows ou qualquer editor de texto, figura 9. Não utilize o Word. Nesse caso, é melhor utilizar o Bloco de Notas por ser um editor mais simples.

Um arquivo HTML é um texto que o navegador interpreta e monta para você visualizar. Como uma receita, são as instruções para preparar um bolo, por exemplo.



**Figura 9 - documento HTML e a exibição correspondente no navegador.**

**Fonte:** Ewerton Mendonça

**Descrição:** página HTML escrita no Bloco de Notas do Windows e sua exibição no navegador.



Escrever todas as páginas no Bloco de Notas funciona, mas outras aplicações são especializadas para este serviço. Chamamos de IDE, Integrated Development Environment, ou seja, um ambiente integrado para desenvolvimento. Temos várias IDEs gratuitas e pagas disponíveis para download. Veja uma lista com algumas gratuitas.

<http://www.sublimetext.com/>

Sublime Text 2 é uma das mais conhecidas e populares do Mercado;

<http://aptana.com/>

Aptana é uma IDE baseada em um ambiente de programação completo, que possui mais recursos e consequentemente é mais complicado.

Existem muitas outras, mas o mais importante é saber que elas funcionam como um editor de texto. Você pode escolher aquele que mais lhe agrada. Só atente que, no caso no Bloco de Notas, você deve salvar com a extensão .html, para o navegador saber do que se trata.

Até aqui falamos muito sobre arquivos HTML, porém o que são estes arquivos? Antes de explicarmos como eles são escritos, vamos expô-lo de modo abrangente. Nas próximas competências eles serão abordados de forma melhor.

## 1.1.10 W3C e os padrões web

A web só cresceu ao ponto que vemos hoje por conta dos padrões e do fato de que eles são abertos, não possuem uma empresa como dona. No entanto, um corpo sem cabeça não sabe para onde ir. Isso quer dizer que alguma organização precisa cuidar dos padrões para que eles possam cumprir seu propósito e evoluir cada vez mais.

Quem define e cuida dos padrões web é uma organização chamada W3C, WWW Consortium. Quem mantém esta organização são grandes empresas que têm interesse que a web cresça e evolua, como a IBM, Microsoft, Google, entre outras. Assim, a W3C define as regras dos padrões e as empresas que desejam seguem. O que isso quer dizer?

Vamos ver no caso do padrão HTML. A W3C estuda e define como o HTML funciona para construir documentos HTML. Então, distribui um documento, como um PDF, contendo as regras do padrão. As empresas que desejam construir um navegador web utilizam o documento para compor





um navegador que sabe ler documentos HTML. Assim, todo navegador web mostrará o HTML da mesma forma, pois todos seguem o mesmo padrão.

O HTML 5 é um conjunto de novos padrões compostos pelas versões mais atuais do HTML, CSS e JavaScript. Foi utilizado o nome HTML 5 para esse conjunto por ser mais popular e facilitar a publicidade. Existem muitos outros padrões relacionados à web, mas vamos focar nesta tríade.

Ainda nesta competência gostaria que você tivesse um primeiro contato com as tecnologias. Nada muito complicado. Apenas o início de qualquer site. Assim, poderemos iniciar o aprendizado de cada um dos padrões já com um conhecimento prévio. Será um exemplo bem simples.

## 1.1.11 HTML

O HTML é um acrônimo para **Hipertext Markup Language**, que significa linguagem de marcação para hipertexto. Hipertexto nada mais é que do que um texto fragmentado em páginas conectadas por hiperlinks e pode ser lido de forma não linear, ou seja, você começa a ler em um lugar e se quiser pode parar de ler e seguir para outra página, fazendo seu próprio fluxo de leitura.

O HTML serve para estruturar, hierarquizar e agora, com o HTML 5, contextualizar o conteúdo daquela página. No caso, você terá um conteúdo que será exibido na página. Formataremos uma estrutura com uma cabeça e um corpo, depois vamos hierarquizá-lo marcando os níveis externos dos internos e, em seguida, poderemos contextualizá-lo para, por exemplo, o navegador saber que aquele pedaço de texto é um menu e não um título.

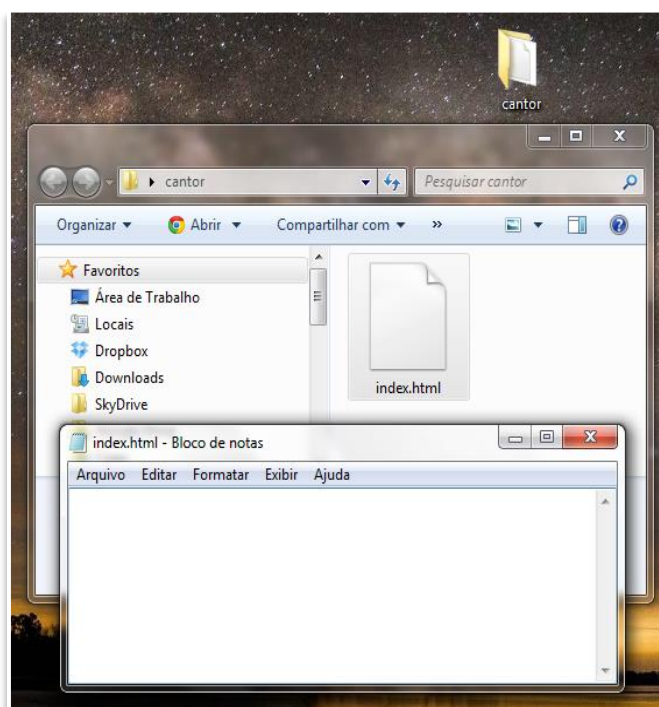
Observe que não falei nada sobre cores, imagens ilustrativas, brilhos e truques. Isso se deve porque “estilizar” não é o trabalho do HTML. O trabalho pertence a outro padrão, o CSS. Mas, vamos com calma. Faremos nossa primeira página, a página inicial. Aquela que vemos pela primeira vez quando entramos em um domínio.

Esta primeira página comumente é chamada index.html, mas pode ser default.html ou até possuir outra extensão, index.htm e default.htm. Mas o comum, e a que vamos utilizar aqui, é index.html. Tenha cuidado com a extensão. Se você utilizar o Bloco de Notas do Windows, ele salvará com a extensão .txt, mesmo que você coloque .html. Então, verifique se seu arquivo não está escrito index.html.txt. Se tiver, você terá que renomear para index.html.



Comece criando uma pasta com o nome de seu projeto em um lugar fácil, pode ser na área de trabalho. Depois abra seu editor de texto preferido e salve o arquivo dentro desta pasta com o nome index.html. Quando colocamos uma URL no navegador e não colocamos o nome da página, o servidor nos envia a index.html. É bem prático.

Observe na Figura 10 a pasta e o arquivo no Bloco de Notas.

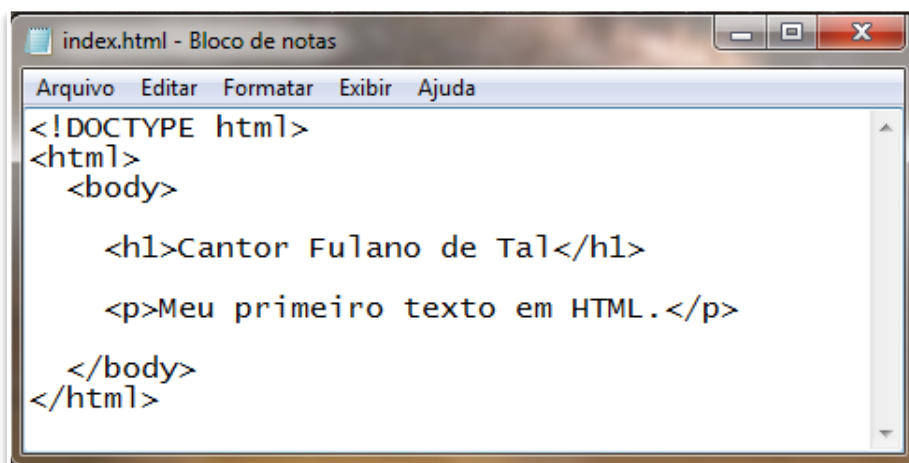


**Figura 10 - página inicial de nosso site de exemplo, index.html. Por enquanto, vazia.**

**Fonte:** Ewerton Mendonça.

**Descrição:** pasta do projeto 'cantor' e página index.html na pasta. Observe que ainda não escrevemos nada no documento.

Agora, copie o código da figura 11 no arquivo index.html. Explicaremos mais tarde cada uma delas. Tenha cuidado de copiar exatamente igual. Caso erre alguma letra, a apresentação pode não aparecer do mesmo jeito. Salve o arquivo.



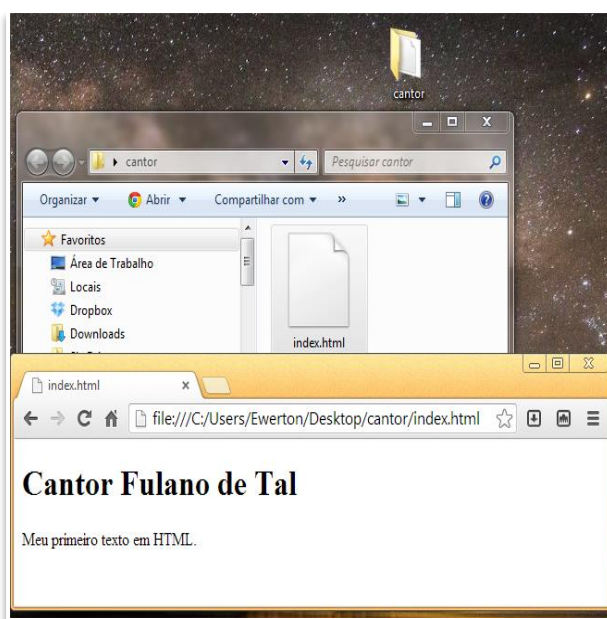
```
index.html - Bloco de notas
Arquivo  Editar  Formatar  Exibir  Ajuda
<!DOCTYPE html>
<html>
  <body>
    <h1>Cantor Fulano de Tal</h1>
    <p>Meu primeiro texto em HTML.</p>
  </body>
</html>
```

**Figura 11 - código inicial do arquivo index.html.**

**Fonte:** Ewerton Mendonça.

**Descrição:** bloco de notas exibindo o código HTML inicial.

Vamos ver o resultado. Arraste o arquivo index.html para seu navegador. Isto abrirá esta página nele. Observe a figura 12, se houver algo diferente pode ser algum problema de digitação. Na maioria das vezes, quando estamos aprendendo, os erros são de digitação. Então, quando algo não sair como planejado, procure o erro letra por letra.



**Figura 12 - resultado da página inicial no navegador.**

**Fonte:** próprio autor.

**Descrição:** navegador exibindo o resultado do código da figura 11.



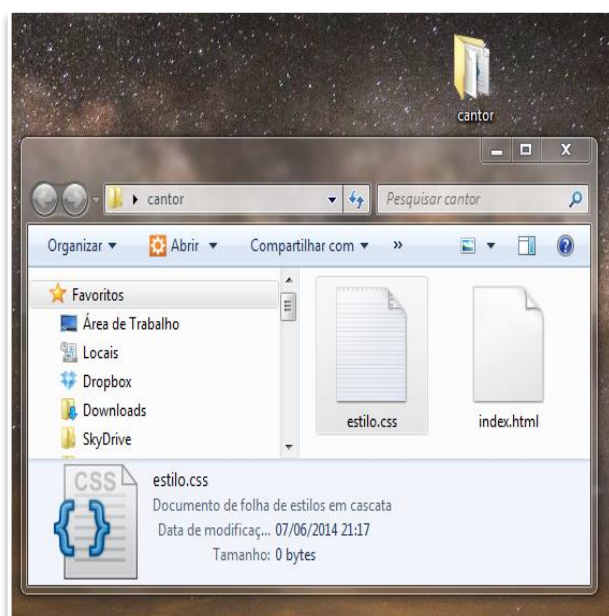
Observe que o documento foi apresentado com uma estilização. A fonte de título está em negrito e em um tamanho maior, com alinhamento à esquerda. O fundo da página é branco e o texto do parágrafo é menor. O título do texto também apresenta uma distância do começo da página até a linha, e uma distância à esquerda e abaixo. Todas as características estéticas foram definidas pelo navegador. Quando ele não encontra um “estilo” configurado para a página, ele utiliza esse padrão.

Para mudarmos o estilo, podemos fazer no próprio HTML, mas isto não seria profissional. Vamos utilizar outro padrão criado especialmente para este objetivo, o CSS.

## 1.1.12 CSS

CSS é o acrônimo de Cascading Style Sheets, que em uma tradução livre seria folhas de estilo em cascata. O motivo de ser em cascata é porque uma folha se sobrepõe a outra apenas nos pontos que utilizamos, e sempre podemos acrescentar folhas por cima.

Vamos demonstrar isso. Faça um arquivo, estilo.css. Tenha cuidado com a extensão. Salve esse arquivo na pasta de nosso projeto. Observe a figura 13 do meu exemplo.



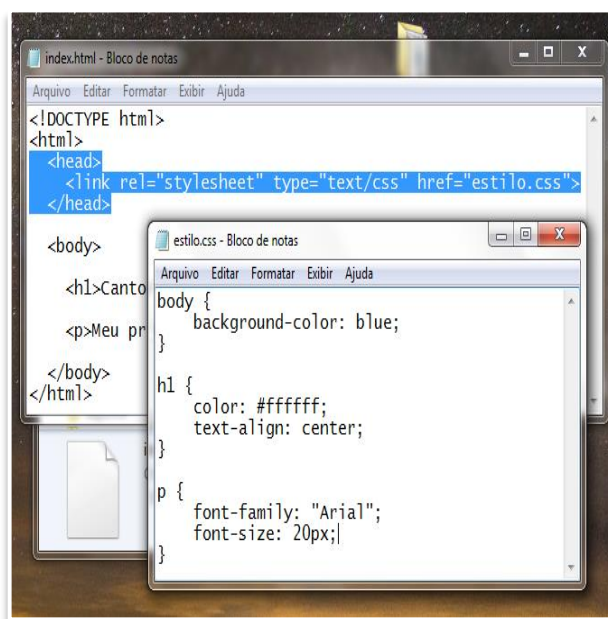
**Figura 13 - arquivo estilo.css que guarda a configuração do estilo utilizado para exibir o documento.**

**Fonte:** Ewerton Mendonça.

**Descrição:** pasta do projeto com o arquivo index.html e estilo.css.



Vamos configura a cor, o tamanho, a forma, a posição dos elementos do arquivo HTML. Copie o texto da figura 14 exatamente igual. Se algo sair errado, procure por erros de digitação. O CSS é bem mais rigoroso do que o HTML com relação a erros, fique atento também às letras maiúsculas e minúsculas no CSS.



**Figura 14 - código destacado na imagem diz que aquele documento deve ser exibido de acordo com as regras CSS que estão em outro documento.**

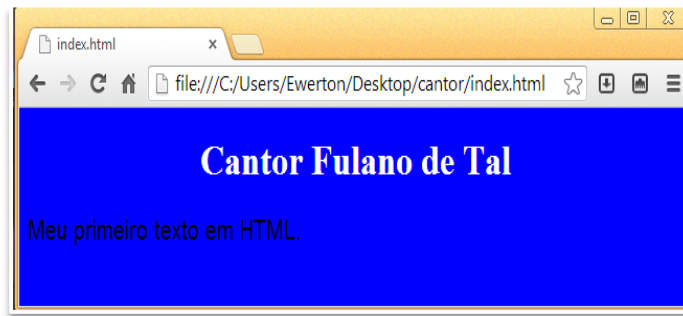
**Fonte:** Ewerton Mendonça

**Descrição:** conteúdo dos arquivos estilo.css e index.html.

Observe que também alteramos o arquivo index.html. Colocamos três linhas que estão destacadas em azul na Figura 14. Elas servem para “amarrar” aquela página àquele estilo. Assim, várias páginas podem receber o mesmo estilo. Se mudarmos ele, todas recebem as mudanças.

Compare o resultado com a imagem da figura 15. Explicaremos como isso aconteceu em outra competência.





**Figura 15 - página estilizada pelo CSS.**

**Fonte:** Ewerton Mendonça.

**Descrição:** navegador exibindo o documento HTML de acordo com as regras CSS.

Por enquanto já está bom. Se ainda não fez, faça este mesmo exemplo, para sentir o processo de desenvolvimento. Não demonstraremos o JavaScript, mas ele funcionará bem parecido com o CSS. Por ele ser uma linguagem de programação, vai exigir uma atenção maior.



Lembre-se de que seres humanos aprendem repetindo. Então, para exercitar, comece com o site de seu amigo. Aquele que nos ajudou desde a primeira competência. Faça a primeira página e o primeiro estilo para a página dele. Utilize os exemplos desta competência, mas descubra como modificar o texto do título e o texto do parágrafo.



## 2. Competência 02 | Formatar um Site Usando Técnicas Avançadas em HTML5

Agora que já sabemos o que são os padrões, de onde vieram e até já experimentamos o gostinho de fazer o primeiro site, vamos mergulhar para valer no primeiro padrão, o HTML.

Utilizaremos a versão mais atual do HTML na construção do site que trabalhamos até aqui, competência por competência. Se por acaso você pulou uma, terá que voltar e ler atentamente para entendê-la e depois continuar. Isso é necessário porque todo o caminho que percorremos até aqui depende do anterior.



HTML 5 é o conjunto de tecnologias: HTML, CSS e JavaScript em suas versões mais recentes. A utilização do termo HTML 5 não significa que tudo está apenas em um padrão. Trata-se de um truque publicitário para melhorar sua adoção.

Nos próximos passos explicarei como construir as páginas através de marcações. Cada marcação pode parecer simples, mas o entendimento delas depende de que você exercite. Então, copie o exemplo, faça os exercícios que propusermos e experimente outras formas. A internet está abarrotada de tutoriais e vídeos tutoriais que podem lhe ajudar a ter o domínio do padrão. Muita gente até desenvolveu truques interessantes com os padrões. Então, tenha dedicação e curiosidade que o resultado será muito bom.

Vamos começar explicando que o HTML não deve ser utilizado para estilizar uma página. Por estilizar, quero dizer colocar cores, imagens, definir fontes e posições, ou seja, dar uma aparência agradável ao conteúdo. Ela é uma linguagem responsável por estruturar, hierarquizar e, agora em sua nova versão, contextualizar a informação da página.

Estruturar significa que o documento terá um começo, um meio e um fim. Hierarquizar quer dizer que teremos um conteúdo organizado, como se fossem caixas dentro de caixas, dentro de caixas, e assim por diante... E por contextualizar temos que uma informação é um título e não um parágrafo, nem um endereço. Assim, nosso documento ficará bem construído para ser utilizado pelo mundo.



## 2.1 Tag

Conseguimos estas qualidades (estruturação, hierarquização e contextualização) através de marcações. Uma marcação também é conhecida por “tag”. As tags marcam o início e o fim de um conteúdo. Veja o exemplo da figura 16. Nela é mostrada a estrutura da tag <p>. Esta tag define um parágrafo. Observe que ela começa com uma tag de início <p> e uma tag de fechamento </p>. Todas as tags de fechamento possuem uma barra comum à frente do nome da tag. Porém, nem toda tag tem tag de início e tag de fim, por exemplo: <br />, <img />, entre outras.

**<p>**Meu primeiro parágrafo.**</p>**

**Figura 16 - estrutura de uma tag.**

**Fonte:** Ewerton Mendonça.

**Descrição:** estrutura da tag p responsável pela semântica de parágrafo.



**Tenha muito cuidado com a escrita das tags. Um erro de digitação pode comprometer completamente a estrutura. Então, preste atenção nos símbolos < , > e a barra /. Muita gente, quando está aprendendo, troca ou esquece. Quando algo der errado, verifique caractere por caractere, até achar o erro**

Vimos uma tag que possui um início e um fim, classificadas como tags de bloco. Mas também existem tags que não precisam de um fim. Elas podem marcar um local do documento. Por exemplo, a tag <img />. Ela marca o local onde será exibida uma imagem. Classificamos estas tags como tags de linha. Tags em linha começam e terminam em cada linha, colocando a barra ao final da tag.





## 2.2 Atributos

Mas do jeito que está, a tag não mostrará nada. Ela precisa de outras informações, como a URL de onde está a imagem. Temos, então, os atributos da tag. Eles possuem uma palavra-chave, o símbolo de igual e seu valor entre aspas duplas. Observe na figura 17, o atributo src que possui a URL de onde está a imagem. A tag <img> tem outros atributos, que veremos mais adiante. Cada novo atributo é colocado logo a seguir, sejam quantos forem. Observe também na figura 31 que a tag <img> é em linha, ela começa e termina na mesma tag, não possuindo uma tag de fechamento.

```

```

**Figura 17 - tag <img> com dois atributos, src que diz o endereço da imagem no servidor, e o alt com um texto, que é exibido se a imagem não puder ser mostrada.**

**Fonte:** Ewerton Mendonça.

**Descrição:** atributos src e alt da tag img.

Assim, um documento HTML nada mais é que um texto misturado com as tags.

## 2.3 DOCTYPE

Como dissemos, estamos aprendendo a última versão do padrão HTML, chamada de HTML 5, mas existiram outras que eram diferentes em algum aspecto. Na internet existem todas estas versões. Para não ficar confuso enquanto lê o documento, temos uma declaração que diz exatamente isso.

Version	Year
HTML	1991
HTML+	1993
HTML 2.0	1995
HTML 3.2	1997
HTML 4.01	1999
XHTML	2000
HTML5	2012

**Figura 18 - tabela com as versões do HTML e o ano de lançamento.**

**Fonte:** [www.w3schools.com/html/html\\_intro.asp](http://www.w3schools.com/html/html_intro.asp)

**Descrição:** tabela com as versões do HTML e o ano de lançamento, que vão de 1991 a 2012.



O DOCTYPE é uma declaração que informa à aplicação que lê o documento HTML qual seu tipo para, assim, ser lido corretamente. A declaração deve ser escrita em uma das três formas da figura 19, e deve ser exatamente à primeira linha do documento.

```
<!DOCTYPE html>  
  
<!DOCTYPE HTML>  
  
<!DOCTYPE Html>
```

**Figura 19** - três opções de declaração. A mais comum é a primeira. Não deve haver de forma alguma qualquer outra linha acima desta.

**Fonte:** Ewerton Mendonça.

**Descrição:** as três formas de declaração do doctype no HTML5.



DOCTYPE não é uma marcação nem faz parte do HTML. Ela é uma declaração de que tipo e versão de documento ele é.

## 2.4 Estrutura básica do documento HTML

Todo documento HTML se divide em duas partes. A primeira parte informa alguma coisa sobre o documento, como o seu título. A segunda parte é o conteúdo do documento. Existem tags específicas para cada uma dessas partes, mas caso você erre e coloque tags erradas nos lugares errados o navegador, quando ler, vai ignorar. Mas não faça isso. Alguns softwares podem ler páginas para deficientes visuais e, em alguns casos, os erros podem impedir a leitura correta.



**Figura 19 - site da W3C que verifica seus arquivos HTML em busca de erros. Você pode utilizá-lo para conferir seu trabalho antes de publicar.**

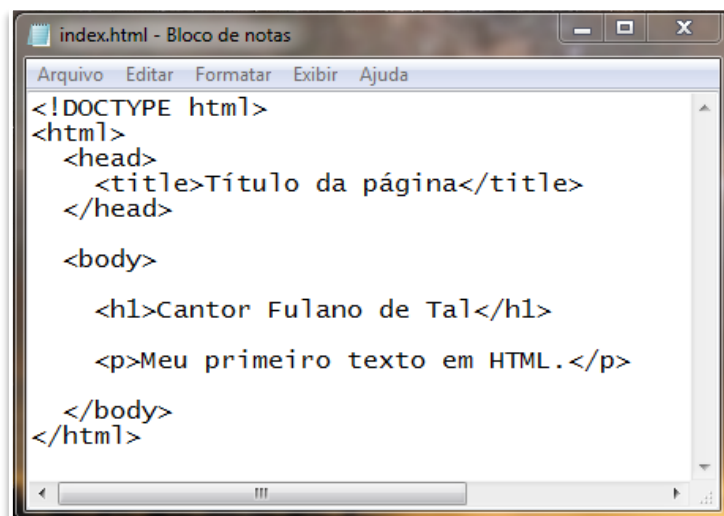
**Fonte:** [http://validator.w3.org/#validate\\_by\\_upload](http://validator.w3.org/#validate_by_upload)

**Descrição:** captura de tela da página de validação da W3C.



A W3C fornece uma página onde você pode enviar seus arquivos HTML e verificar se existe algum erro. [http://validator.w3.org/#validate\\_by\\_upload](http://validator.w3.org/#validate_by_upload)

A estrutura básica de um documento HTML começa pela tag <HTML>. Ela englobará duas tags: a <head> (cabeçalho) e a <body> (corpo). A tag <head> guarda tags que dizem alguma coisa sobre o documento, como a tag <title> que dá um título para o documento. A tag <body> engloba o conteúdo e as tags que estruturam esse conteúdo. Isso fica melhor de ser entendido na figura 20.

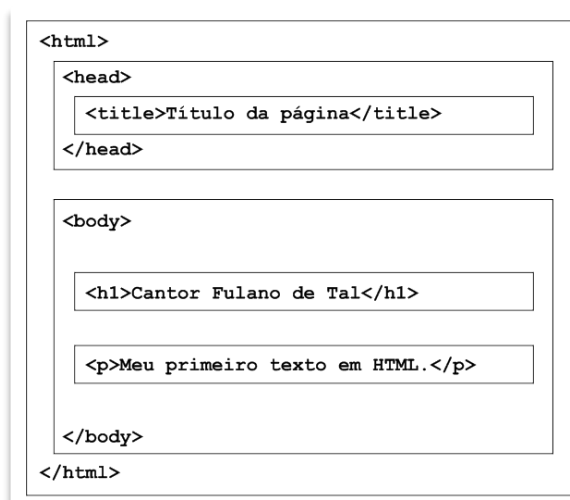


**Figura 20 - estrutura básica de um documento HTML.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código HTML contendo a estrutura básica de toda página HTML.

Para entender melhor esta história de hierarquização e caixas dentro de caixas, e dentro de caixas, você tem que enxergar o documento de acordo com a figura 21. Observe que a tag <title> pertence à tag <head>, que pertence à tag <html>. Já as tags <h1> e <p> pertencem à tag <body>, que pertence à tag <html>.



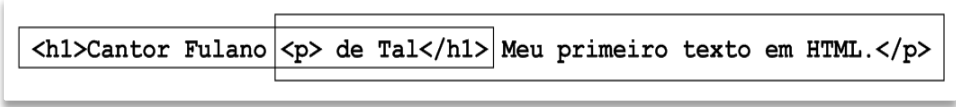
**Figura 21 - a hierarquia de uma página HTML se assemelha a caixas dentro de caixas.**

**Fonte:** Ewerton Mendonça.

**Descrição:** documento HTML onde as tags estão escritas dentro de caixas para demonstrar visualmente a hierarquia.



Você deve prestar bastante atenção a esta hierarquia para não errar misturando as tags. A figura 22 mostra uma forma errada de tags. Quando algo de estranho acontecer na página, pode ser um erro nessa hierarquia.



```
<h1>Cantor Fulano <p> de Tal</h1> Meu primeiro texto em HTML.</p>
```

**Figura 22 - forma errada. As tags de abertura e fechamento se misturam.**

**Fonte:** Ewerton Mendonça.

**Descrição:** duas tags entrelaçadas com desenhos de caixas que se cruzam, demonstrando que estão erradas.

Toda página deve ter as tags `<html>`, `<head>` e `<body>` da forma como foram mostradas.



Tags HTML não diferenciam marcações em maiúsculas ou minúsculas, ou seja, `<P>` significa o mesmo que `<p>`. Nós utilizamos tags minúsculas porque o World Wide Web Consortium (W3C) recomenda minúsculas em HTML 4 e exige etiquetas minúsculas em XHTML.

## 2.5 Visualizando as páginas

O propósito de um navegador (como o Google Chrome, Internet Explorer, Firefox, Safari) é ler documentos HTML e exibi-los como páginas da web. O navegador não exibe as tags HTML, mas usa as tags para determinar como o conteúdo da página HTML deve ser apresentado para o usuário.

Para testar as páginas que você construir, basta arrastar o arquivo em um navegador aberto, ou digitar na barra de endereço o endereço do arquivo. Como se fosse uma URI. Veja na figura 23 a nossa página de exemplo sendo exibida.



**Figura 23 - exibição da página de exemplo da figura 20.**

**Fonte:** Ewerton Mendonça.

**Descrição:** navegador exibindo o resultado do código da figura 20.

Observe que a aba da janela, destacada em vermelho na figura 23, mostra o texto escrito dentro da tag <title>, “Título da página”. Já no espaço de exibição da página, está nosso título e parágrafo. Mas observe que eles estão formatados. Mesmo que não tenhamos definido um estilo, os navegadores possuem um estilo CSS padrão que dá margens, define cores (letras pretas em fundo branco), fontes, tamanhos, etc. Como o navegador se preocupa com a apresentação, então, caso o desenvolvedor não tenha se preocupado em definir um estilo, ele usa o estilo padrão presente em todos os navegadores.

A construção da aparência de uma página seguindo o HTML e o CSS chama-se renderização da página.



Documentos HTML ignoram o pulo de linha e quando há mais de um espaçamento na renderização. Utilizamos tags e códigos para isso.

## 2.6 Nomenclatura dos arquivos HTML

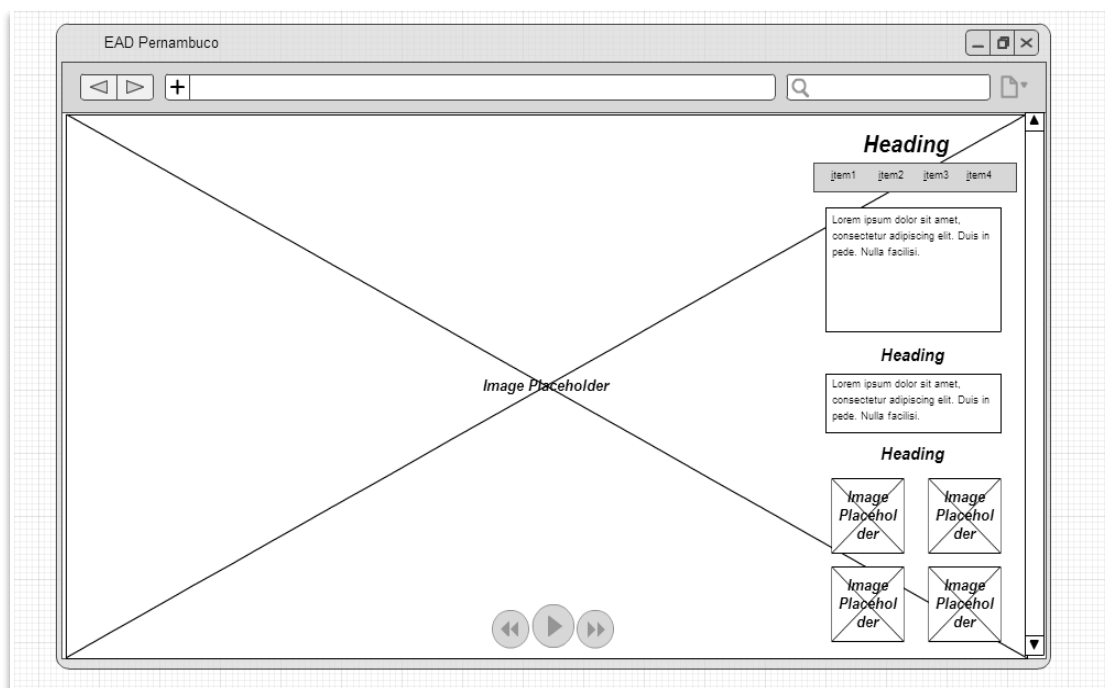
Os arquivos HTML são armazenados nos servidores aguardando que alguém os solicite. O servidor é uma aplicação, como seu navegador, e possui algumas regras para os nomes dos arquivos que estão hospedados nela.

- NUNCA utilize caracteres especiais como letras acentuadas e símbolos (!, @, #, \$, %, ", &, \*, (, ), entre outros.). Mesmo que ele possa ser visualizado em seu navegador, o servidor pode não aceitar. Então, apenas utilize nomes simples;
- NUNCA utilize espaço para nomes compostos. As URIs não podem ter espaçamento em seus nomes. Coloque tudo junto, se utilizar nomes compostos;
- Não aconselhamos a utilização de letras maiúsculas. Alguns servidores mais antigos não as aceitam. Se garanta e utilize apenas letras minúsculas e/ou números.

## 2.7 Layout HTML

O layout HTML pode ser produzido através de tags <div>. Elas isolam áreas e podem ser configuradas pelo CSS para terem a aparência que você quiser.

O wireframe irá nos ajudar a organizar o layout HTML. A figura 24 mostra o wireframe de exemplo do site do cantor sertanejo e aproveitamos para “encaixotar” todos os grupos em tags <div>. Observe que colocamos todo o conteúdo em uma coluna que sempre fica à direita e “encaixotamos” o título e o menu e embaixo deixamos uma área para as mídias sociais e mais alguma informação.



**Figura 24 - planejamento do layout HTML baseado no wireframe de exemplo.**

**Fonte:** Ewerton Mendonça.

**Descrição:** diagrama da página inicial de nosso site de exemplo.





Vamos agora estruturar todo o conteúdo do site com as tags no documento HTML.

Nossa primeira página é a index.html. A figura 25 mostra o código básico de qualquer página HTML, com todas as <div> que estruturam nosso exemplo de acordo com o wireframe. Observe e tente entender. Cada <div> corresponde a uma área do wireframe. Quando você for construir o seu projeto, estude a silhueta dele e estruture o documento. Quanto mais exercício fizer, melhor vai entender. Comece sempre pelos mais simples, depois você pode buscar layouts mais complexos.

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title>Fulano de Tal</title>
5    </head>
6
7    <body>
8      <div id="container">
9        <div id="cabecalho">
10
11        </div>
12
13        <div id="menu">
14
15        </div>
16
17        <div id="conteudo">
18
19        </div>
20
21        <div id="rodape">
22
23        </div>
24      </div>
25    </body>
26  </html>
```

**Figura 26 - construção da estrutura através de tags <div>.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código HTML demonstrando a estrutura de tags básicos para a maioria das páginas.

## 2.8 Atributos Id e Class

Na Figura 27 as tags <div> estão com um atributo, chamado id. Este atributo identifica aquela tag com um nome único. Não devemos utilizar, no mesmo documento HTML, o mesmo id em tags diferentes. Lembre-se de que o id é como um número de identidade, só deve haver um no mesmo documento.





Outra forma de identificar é utilizando o atributo class. No entanto, esse atributo identifica um grupo de tags. O class classifica aquele grupo com uma identidade. Assim, o class faz mais sentido quando temos mais de uma tag com aquele identificador.

Tanto o id quanto o class não servem para nosso HTML. Vão servir muito para a estilização com CSS. Mas faz sentido eu mostrar para vocês agora, por causa de nosso próximo tópico.



**Coloque sempre nomes simples em minúsculas e sem acentos ou caracteres especiais. Não utilize nomes compostos.**

## 2.9 Novas TAGS HTML 5

O HTML 5 foi lançado para atender vários objetivos. Entre eles está a semântica das informações do documento. Por exemplo, boa parte dos sites da internet tem uma informação chamada endereço. Utilizando id e class podemos identificar e classificar, mas como cada um pode colocar o identificador que quiser para elas, ficamos na mesma situação. Assim, o HTML 5 provê novas tags para dar significado à informação de forma padronizada.

A figura 27 mostra uma tabela com as tags mais usuais e sua finalidade.

Tag	Descrição
<header>	É o início de alguma sessão. Pode ser do próprio documento onde fica a marca do site.
<footer>	É a finalização daquela sessão. Em uma página de produto em uma loja virtual, o footer pode englobar dicas de produtos que outras pessoas compraram junto daquele produto.
<nav>	Engloba tags referentes a algum sistema de navegação. Como aqueles que vimos em uma competência anterior.
<section>	Define uma sessão do documento.
<article>	É mais compreensível com a comparação de um blog que possui vários artigos. Esta tag informa qual parte do documento é um artigo. Este artigo ainda pode ter um <header> e um <footer> referente ao artigo.

Figura 27 - tabela com as tags HTML 5 para contextualização mais utilizadas.

Fonte: Ewerton Mendonça.

Descrição: tabela com o significado das tags básicas.



O HTML 5 provê muitas outras tags e como um web design profissional é sua obrigação conhecer todas elas. Acesse o link abaixo para aprender mais. [https://developer.mozilla.org/pt-BR/docs/HTML/HTML5/HTML5\\_element\\_list](https://developer.mozilla.org/pt-BR/docs/HTML/HTML5/HTML5_element_list)

Podemos utilizar <div> sem problema, mas era assim que se fazia antes do HTML 5. Então, vamos atualizar nosso documento com as novas tags. A figura 28 mostra como ela ficou.

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Fulano de Tal</title>
5   </head>
6
7   <body>
8     <section>
9       <header>
10
11       </header>
12
13       <nav>
14
15       </nav>
16
17       <div id="conteudo">
18
19       </div>
20
21       <footer>
22
23       </footer>
24     </section>
25   </body>
26 </html>
```

**Figura 28 - página formatada com as novas tags HTML 5.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código HTML básico com as novas tags HTML 5.

Se você renderizar o documento em um navegador, não irá ver nada, porque não há conteúdo. Apenas definimos a estrutura, a área de cada informação.

Observe que deixei uma div, mas vamos modificá-la no próximo tópico.



## 2.10 Iframe

A tag <iframe> cria uma área em nosso documento onde podemos renderizar outra página e depois ficar trocando o conteúdo. Você não precisa utilizar <iframe> em seu site. Estou mostrando esta técnica porque ela se encaixa perfeitamente no projeto de exemplo, onde um vídeo ficará passando no fundo da página principal, enquanto o usuário pode navegar por outras páginas sendo exibidas no <iframe>. Então, vamos construí-lo.

A figura 29 mostra o <iframe>. Ele possui um atributo name, para podermos apontá-lo posteriormente, e um src com uma uri relativa para outra página web, que é a primeira página exibida no iframe. Note que troquei a <div> pelo <iframe>.

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Fulano de Tal</title>
5   </head>
6
7   <body>
8     <section>
9       <header>
10
11       </header>
12
13     <nav>
14
15     </nav>
16
17     <iframe name="pagina" src="cantor.html">
18
19     </iframe>
20
21     <footer>
22
23     </footer>
24   </section>
25 </body>
26 </html>
```

**Figura 29 – utilização da tag <iframe>.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código HTML utilizando a tag iframe.



## 2.11 TAGS textuais

Vamos criar outro documento HTML da mesma forma como fizemos antes. É melhor você digitar tudo de novo para decorar. Será um bom exercício. Esta página contém informações sobre o cantor e vamos nomeá-la de cantor.html. Ela possui muito texto e será um bom exemplo de como estruturamos um texto.

O HTML ignora pulos de linhas e mais de um espaçamento entre as palavras. Para pularmos linhas devemos marcar cada linha com a tag **<p>** que significa um parágrafo. Caso queiramos apenas quebrar a linha, ou seja, seguir com o texto na próxima linha, podemos interromper o fluxo utilizando a tag **<br />**. Procure as tags no exemplo da figura 44.

Os títulos são marcados com as tags **<h1>**, **<h2>**, **<h3>**, **<h4>**, **<h5>** e **<h6>**. A numeração mostra os níveis de título, subtítulo e assim por diante. Em nosso exemplo, como vamos utilizar o **<h1>** para o título principal do site, utilizaremos o seguinte, **<h2>** para as sessões do site.

Utilizamos ainda a tag **<span>** nos títulos. Esta tag não faz nada além de marcar. Podemos utilizar ela para um truque de CSS.

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Fulano de Tal :: Cantor</title>
5
6     <meta name="author" content="Ewerton Mendonça">
7     <meta name="description" content="Site demonstração do curso de Web Design do EAD Pernambuco.">
8     <meta charset="utf-8">
9
10  </head>
11  <body>
12    <h2><span>Biografia</span></h2>
13    <p>Lorem ipsum dolor sit amet, consectetur <a href="#">adipiscing</a> elit.
14      Quisque rutrum nibh imperdiet suscipit vehicula.</p>
15    <p>Donec lacinia dolor dictum ante tempus varius. Nulla consequat accumsan magna,
16      ut rutrum nisl vehicula sit amet. Nam at euismod quam. Nulla laoreet justo in
17      risus molestie, vitae dignissim turpis hendrerit. Cras aliquet at orci a egestas.
18      Duis consequat molestie interdum. Mauris quis cursus orci, et auctor metus.</p>
19    <p>Cras nisi massa,<br /> malesuada ac est id, cursus mollis metus. Pellentesque
20      habitant morbi tristique senectus et netus et malesuada
21      fames ac turpis egestas.</p>
22  </body>
23 </html>
```

Figura 30 - documento HTML da página cantor.html.

Fonte: Ewerton Mendonça.

Descrição: código HTML demonstrando as tags textuais.



Agora podemos visualizar a página index.html que mostrará o conteúdo de cantor.html.

A figura 31 mostra o resultado.

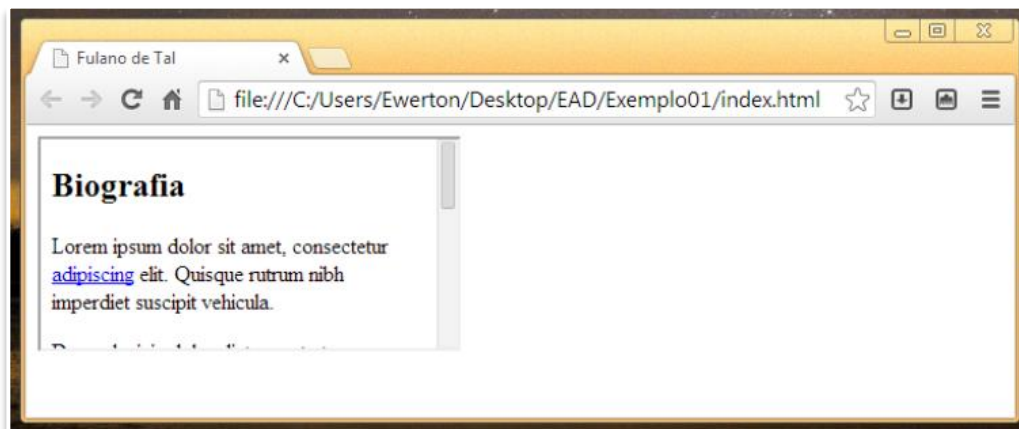


Figura 31 - exibição da página index.html mostrando no <iframe> o conteúdo da página cantor.html.

Fonte: Ewerton Mendonça.

Descrição: Navegador exibindo o resultado do código da figura 30.

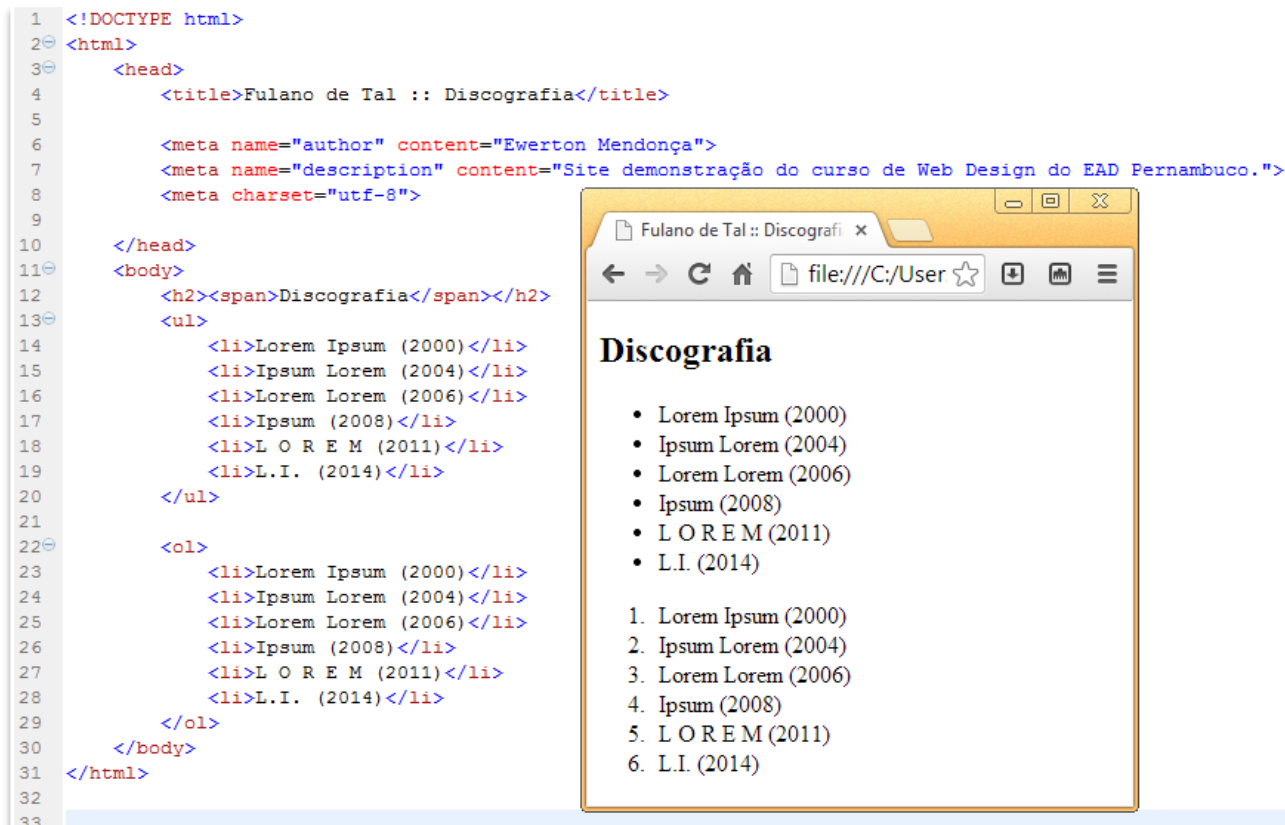
## 2.12 Listas ordenadas e não ordenadas

Vamos agora criar outra página chamada discografia.html. Nela teremos uma listagem dos álbuns musicais lançados pelo cantor. Para estruturarmos uma lista, utilizamos uma tag para agrupar os itens, que podem ser <ol> para listas ordenadas e <ul> para listas não ordenadas.

A diferença é que as listas ordenadas são numeradas e as listas não ordenadas possuem um símbolo, normalmente um ponto. Tanto a forma de numeração quanto o símbolo utilizado podem ser estilizados através de CSS.

Independente da lista que você escolha, os itens são marcados pela tag <li>.

A figura 32 mostra o HTML da página discografia.html e o resultado no navegador. Como as páginas são independentes, resolvi mostrar no navegador apenas a página discografia.html.



**Figura 32 - código HTML de listas ordenadas e não ordenadas e exibição no navegador da página discografia.html.**

**Fonte:** Ewerton Mendonça.

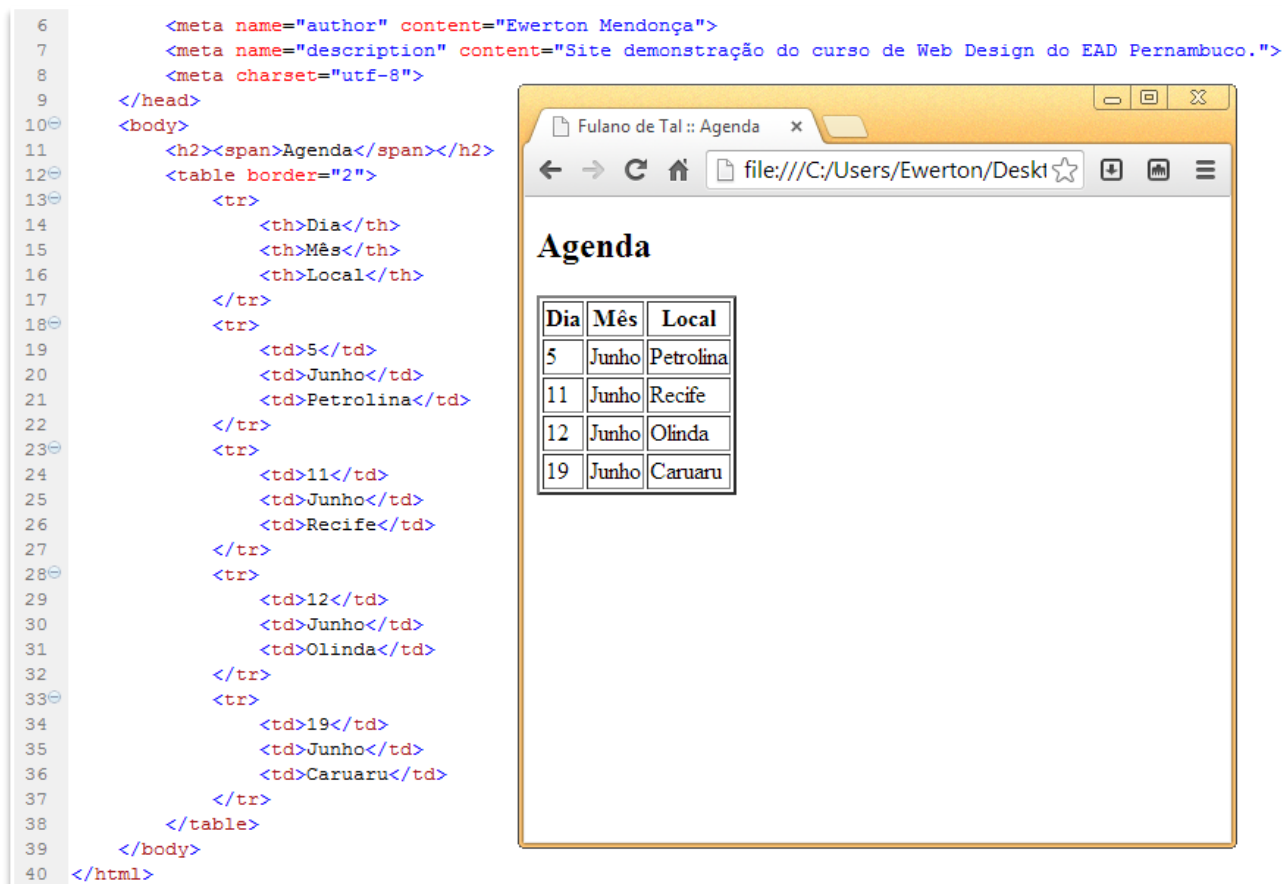
**Descrição:** código de demonstração de listas no Bloco de Notas e o resultado da exibição no navegador.

## 2.13 Tabelas

Antigamente se utilizavam tabelas para juntar as imagens que estilizavam o site. Isto era um problema sério, porque o conteúdo, a estrutura e o estilismo ficavam misturados. O CSS resolveu isto e a utilização de tabelas se resumiu à finalidade de descrever dados tabulares.

Em nosso exemplo, na página agenda.html, temos uma tabela que apresenta de forma organizada os dias, meses e locais de show do cantor. A estrutura de uma tabela começa com uma tag **<table>** que engloba toda a tabela. Depois, temos que criar as linhas com a tag **<tr>** que englobarão as tags **<td>** que são as células. Ainda podemos fazer com que as células superiores da tabela sejam o cabeçalho das colunas com a tag **<th>**. Veja o exemplo na figura 33.





**Figura 33 - marcação para formatação de dados tabulares.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código de demonstração de tabela no Bloco de Notas e o resultado da exibição no navegador.

## 2.14 Imagens

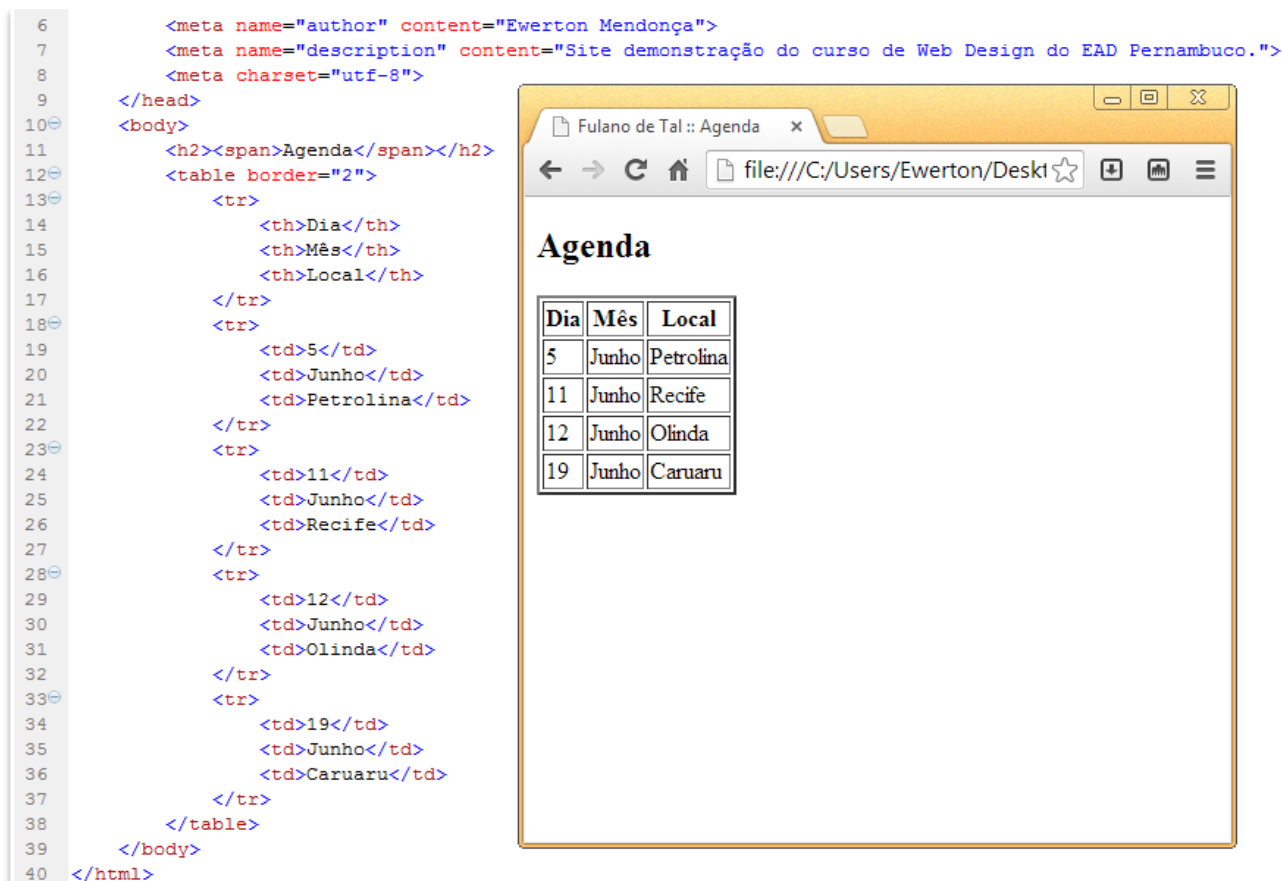
Imagens inseridas no HTML servem para ilustrar o assunto de que se está falando. Não devem ser utilizadas para estilizar, para isso temos o CSS. No caso de nosso exemplo, o “assunto” são fotos dos shows do cantor, então, as fotos podem ser utilizadas no conteúdo do site.

Não podemos inserir uma foto no código HTML porque o documento é apenas de texto, mas podemos marcar um local onde será exibida uma imagem quando o documento for renderizado no navegador. Por conta disso, quando levamos uma página que contenha imagem em um pendrive, por exemplo, devemos nos lembrar de levar todas as imagens juntas.

Marcamos o local onde as imagens serão inseridas utilizando a tag **<img>**, figura 34. Entre os atributos dela temos o **src** que indica a URI da imagem. No exemplo, na pasta do site, criamos uma



pasta “imagens” para guardar todas as imagens e dentro dela outra pasta chamada “album”, para separar de outras imagens. As imagens foram preparadas antecipadamente para serem apresentadas no tamanho correto. Para isso, foi utilizado um programa de edição fotográfica como o Gimp.



**Figura 34 - código HTML da página fotos.html e renderização no navegador.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código de demonstração de inserção de imagens no Bloco de Notas e o resultado da exibição no navegador.

## 2.15 Formulário

Um formulário é uma forma de comunicação com o usuário. Ele fornece campos para coletar informação. Essas informações vão para o servidor para serem processadas. Mas ele não é responsável por esse trabalho, o servidor apenas serve páginas, ele passa os dados para alguma outra aplicação. Normalmente um serviço de hospedagem também oferece uma aplicação que possa processar os dados, no caso, enviar um e-mail com os dados do formulário. Existem várias aplicações



que fazem isso e todas de formas diferentes. Você precisa se informar com seu serviço de hospedagem para saber como proceder.

Independente do serviço que você utilizará, todos vão ter um formulário HTML para coletar do usuário e enviar a informação. O formulário é construído iniciando com a tag **<form>** que precisa de dois atributos. O atributo **action** aponta para a URI onde os dados vão ser processados e o atributo **method** especifica a forma de envio, normalmente utilizamos o método **post**.

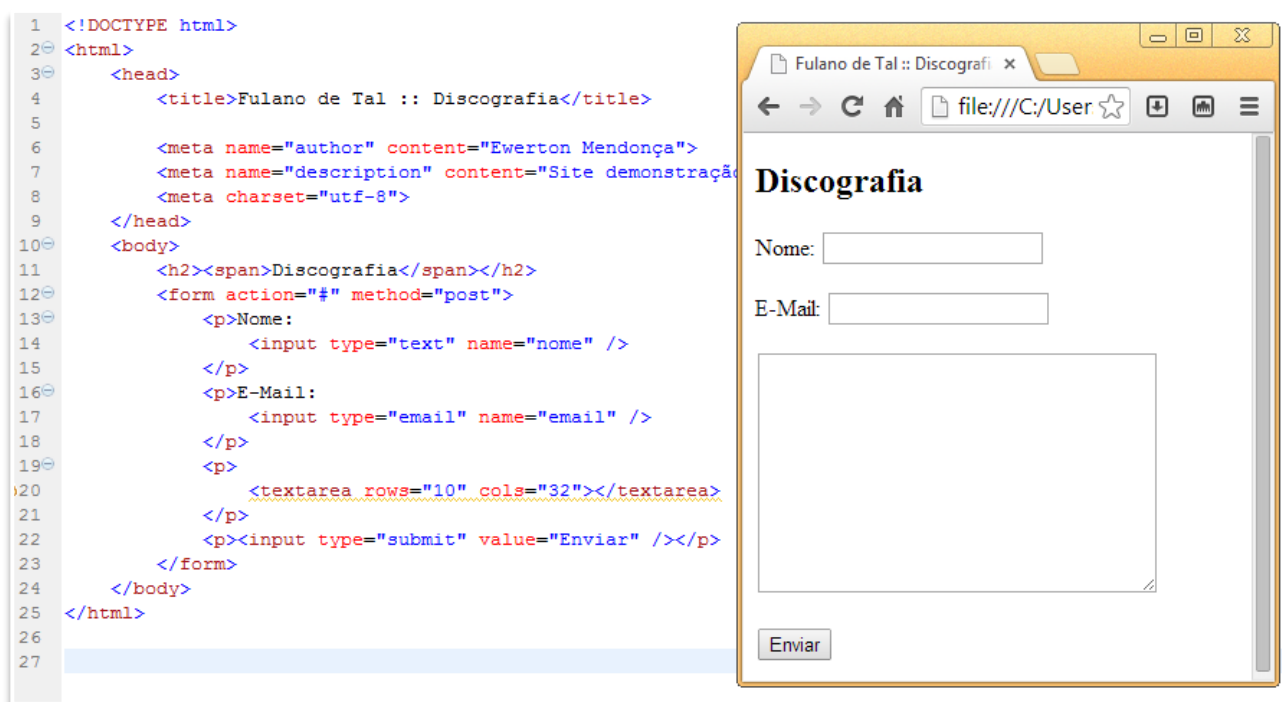


Figura 35 – código do formulário HTML e exibição no navegador.

Fonte: Ewerton Mendonça.

Descrição: código de demonstração de formulário no Bloco de Notas e o resultado da exibição no navegador.

## 2.16 Links

Já temos todas as páginas de nosso site de exemplo. O conteúdo foi distribuído nas páginas e temos a página principal, index.html. Através dela é que vamos ter acesso a todas as outras páginas.

Como este layout utiliza o **<iframe>** não temos necessidade de ter navegação nas outras páginas, mas provavelmente seu layout deve ser diferente e neste caso você terá que ter um sistema de navegação em cada página. Lembre-se disso.



Voltamos para a `index.html` e vamos construir um sistema de navegação para os usuários poderem ter acesso ao restante do conteúdo. Para isso, vamos utilizar a tag `<a>` e vamos colocar na tag `<nav>` que é destinada a englobar tags de navegação. Entre os atributos da tag `<a>`, temos o **href** que deve ter uma URI como valor, apontando para a página que o link leva, e o atributo **target**, que diz onde será exibida a página. Os valores mais utilizados para **target** estão abaixo:

- **blank**: abre o documento apontado em uma nova janela ou aba;
- **self**: abre o documento apontado na própria janela ou aba. Esta opção é a padrão, se não colocarmos o atributo **target**;
- **framename**: podemos colocar o valor de um atributo **name** de um `<iframe>`, para que o documento seja exibido naquela área.

Não queremos substituir a página `index.html` porque ela está exibindo o vídeo do cantor. Então, vamos colocar o nome do `iframe` como valor do **target**. Assim, quando alguém clicar no link, a página da URI de **src** será exibida no `<iframe>`. Observe na figura 36 destacadas em vermelho as tags `<a>` do menu que engloba o texto do link. O nome do `<iframe>` é “pagina”. Lembre-se de sempre evitar acentos e caracteres especiais, assim como palavras separadas por espaço, quando nomear algo em HTML. Por isso, “pagina” ficou sem acento.



```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Fulano de Tal</title>
5     <meta name="author" content="Ewerton Mendonça">
6     <meta name="description" content="Site demonstração do curso de Web Design do EAD Pernambuco.">
7     <meta charset="utf-8">
8   </head>
9   <body>
10    <section>
11      <header>
12        <h1><span>Fulano de Tal</span></h1>
13      </header>
14      <nav>
15        <a href="cantor.html" target="pagina">CANTOR</a> |
16        <a href="discografia.html" target="pagina">DISCOGRAFIA</a> |
17        <a href="fotos.html" target="pagina">FOTOS</a> |
18        <a href="agenda.html" target="pagina">AGENDA</a> |
19        <a href="contato.html" target="pagina">CONTATO</a>
20      </nav>
21      <iframe name="pagina" src="cantor.html">
22
23      </iframe>
24      <footer>
25        <nav>
26          <a href="#"></a>
27          <a href="#"></a>
28          <a href="#"></a>
29          <a href="#"></a>
30        </nav>
31        <address>Produzido para EAD Pernambuco</address>
32      </footer>
33    </section>
34  </body>
35 </html>

```

**Figura 36 - tags que formam o sistema de navegação do site em index.html.**

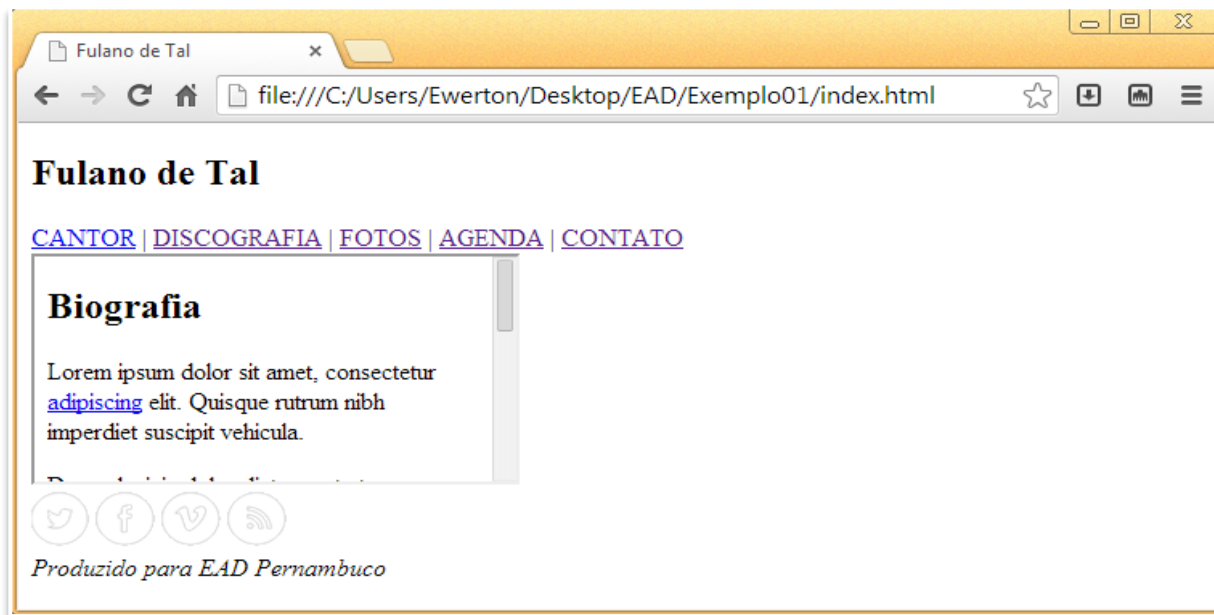
**Fonte:** Ewerton Mendonça.

**Descrição:** código de demonstração de links no Bloco de Notas e o resultado da exibição no navegador.

Podemos também colocar os hiperlinks em meio ao texto. Na figura 30 existe um tag `<a>` que marca uma palavra do texto como demonstração. Não existe um limite para a quantidade de palavras marcadas. Você pode marcar um texto completo se precisar.

## 2.17 Finalizando

O site em HTML está quase pronto. Para finalizar colocamos um título com `<h1>` em `<header>` e em `<footer>` colocamos mais alguns links para mídias sociais, só que em vez de colocarmos texto como hiperlink, colocamos no lugar imagens com a tag `<img>`. Também em `<footer>` acrescentamos uma tag `<address>` sobre direitos autorais. Tudo isso você pode encontrar no código da figura 50 e visualizar o resultado na figura 37.



**Figura 37 - Exibição do site em HTML.**

**Fonte:** Ewerton Mendonça.

**Descrição:** Resultado da exibição no navegador.

Agora, só falta estilizar, ou seja, tornar a aparência mais agradável para o usuário se sentir estimulado a ler nosso conteúdo.



Como exercício você pode fazer o passo a passo e repetir a construção do site apresentado. Depois, você pode mudar para testar como se comportam os elementos. Quando estiver seguro, pode fazer o site que estamos trabalhando nos exercícios desde a primeira competência. Caso você tenha pulado algum exercício, perceba que cada competência depende das anteriores. Então, você terá que fazer o que deixou para prosseguir. Lembre-se de que a maioria dos erros, quando estamos aprendendo, é de digitação. Então, verifique letra por letra com bastante atenção para saber o que fez de errado.





## 3.Competência 03 | Formatar um Site com Recursos Multimídia

Quando criamos uma página web queremos divulgar uma informação. Podemos utilizar vários meios para divulgar esta informação, como texto ou imagem. Por vezes, a informação pode ser passada de uma forma melhor se utilizarmos mais de um meio, como texto e imagens. A utilização de vários meios para se passar a informação ficou conhecida como multimídia. As páginas da Web geralmente contêm elementos multimídia de diferentes tipos e formatos como som, música, vídeos, filmes e animações.

### 3.1 Vídeo

Os primeiros navegadores da WEB tinham suporte apenas para texto e eram limitados a uma única fonte em uma única cor. Mais tarde eles foram melhorados e deram suporte para cores e várias fontes, além de imagens. Posteriormente deram suporte a áudio, vídeo e animações, mas cada navegador interpretava de uma forma. Era um problema porque era comum uma página ser exibida corretamente em um navegador e não em outro. Para resolver, foram construídos *plug-ins*, pequenos programas incorporados nos navegadores para reproduzir conteúdo multimídia de forma igual. O Flash foi o mais famoso deles. Mas como os *plug-ins* não eram nativos do computador, o que quer dizer que eles não foram construídos juntos ao navegador, eles geravam muitos problemas. Para resolver isso, foram definidas regras nomeadas como HTML5. E o navegador com suporte HTML 5 deve seguir essas regras, chamadas de padrão. Assim, todo mundo segue o padrão e sua página é exibida de forma igual em todos os navegadores e sistemas operacionais. Problema resolvido.

Elementos multimídia (como áudio ou vídeo) são armazenados em arquivos de mídia. A maneira mais comum de descobrir o tipo de um arquivo é olhar para a extensão do arquivo. Os arquivos multimídia têm formatos e extensões diferentes, como: .swf, .wav, .mp3, .mp4, .mpg, .wmv e .avi.

O formato .mp4 até agora é o melhor formato para divulgação de vídeo pela web. A figura 38 mostra uma tabela com os formatos mais populares e uma breve descrição.



FORMATO	EXTENSÃO	DESCRIÇÃO
MPEG	.mpg e .mpeg	O MPEG foi desenvolvido pela Moving Pictures Expert Group, sendo o primeiro formato de vídeo mais popular na web. Suportado por todos os navegadores, mas ele não é suportado pelo padrão HTML5.
AVI	.avi	Desenvolvido pela Microsoft, o AVI, Audio Video Interleave, é geralmente utilizado em hardware de câmeras de vídeo e placas de TV. Reproduz bem no sistema operacional Windows, mas não em navegadores.
WMV	.wmv	Também desenvolvido pela Microsoft, o WMV, Windows Media Video, é um formato substituto do AVI e melhorado, mas com os mesmos problemas para a web.
QuickTime	.mov	Desenvolvido pela Apple para seu hardware e sistema operacional proprietário.
RealVideo	.rm e .ram	Desenvolvido pela Real Media que possuía a capacidade de streaming em baixa largura de banda (internet lenta). Streaming é quando o vídeo pode ser reproduzido mesmo sem terminar de baixar. Alguns formatos possuem esta capacidade, mas a maioria não.
Flash	.swf e .flv	Desenvolvido pela Macromedia, este formato só pode ser reproduzido no navegador com a instalação de seu plug-in.
Ogg	.ogg	O Theora Ogg foi desenvolvido pela Fundação Xiph.org sendo de código aberto e livre. O HTML 5 dá suporte a este formato.
WebM	.webm	O WebM foi desenvolvido pelo financiamento das gigantes da web como a Google, Adobe, Mozilla, Opera etc. para ser o formato de vídeo da web. Possui suporte no HTML 5.
MPEG-4 ou MP4	.mp4	Desenvolvido pela Moving Pictures Expert Group e baseado no QuickTime, o MP4 é um formato que agrada bastante os produtores de vídeo pelo fato de que tanto o hardware como câmera geram diretamente este formato quanto ele é suportado por todos os navegadores modernos e recomendado pelo YouTube. Produção e divulgação em um único formato.

**Figura 38 - diversos formatos de vídeo encontrados na web.**

**Fonte:** Ewerton Mendonça.

**Descrição:** tabela contendo o formato, sua extensão e uma breve descrição.

A tag HTML <video> especifica uma maneira padrão de incorporar um vídeo em uma página da web. Ela é aceita nos navegadores mais populares em suas versões mais atuais.

Na figura 39 temos o código para a inclusão de um vídeo junto com algumas tags opcionais, veremos mais adiante o significado de seus atributos e destas tags opcionais.



```
<video width="320" height="240" autoplay>
  <source src="video.mp4" type="video/mp4">
  <source src="video.ogg" type="video/ogg">
  Seu navegador não suporta a tag video.
</video>
```

Figura 39 - código HTML 5 para inclusão de um vídeo.

Fonte: Ewerton Mendonça.

Descrição: código HTML 5 para inclusão de um vídeo.

O HTML5 define métodos, propriedades e eventos DOM para o elemento `<video>`. Isso permite carregar, reproduzir e pausar vídeos, bem como definir duração e volume. Há também eventos DOM que podem notificá-lo quando um vídeo começa a ser reproduzido, é pausado, etc.



O documento HTML 5 pode ser visto como uma estrutura de dados do tipo árvore e os seus elementos possuem relacionamentos. Isso é definido em uma conversão chamada DOM. Para saber mais, acesse:

[https://pt.wikipedia.org/wiki/Modelo\\_de\\_Objeto\\_de\\_Documentos](https://pt.wikipedia.org/wiki/Modelo_de_Objeto_de_Documentos)

- **<video>** Define um vídeo.
- **<source>** Define múltiplas fontes de formatos diferentes para o elemento de mídia. Caso a primeira não possa ser reproduzida, ela será substituída pela próxima.
- **<track>** Define uma legenda para o vídeo.

Em `<video>` temos o atributo **'autoplay'** que toca o vídeo assim que possível. Sem que o usuário precise apertar o play. Você ainda pode colocar o atributo **'controls'**, da mesma forma que o anterior, para adicionar controladores de vídeo. Em `<source>` o atributo **'src'** guarda a URL do vídeo, já o atributo **'type'** guarda o tipo de vídeo, em HTML 5 temos **'video/mp4'**, **'video/ogg'** e **'video/webm'**.

## 3.2 Áudio

O elemento HTML5 `<audio>` especifica uma maneira padrão de incorporar áudio em uma página da web.



Na figura 40 temos o código para a inclusão de áudio junto com algumas tags opcionais, veremos mais adiante o significado de seus atributos e destas tags opcionais.

```
<audio controls>
  <source src="musica.ogg" type="audio/ogg">
  <source src="musica.mp3" type="audio/mpeg">
  Seu navegador não suporta a tag audio.
</audio>
```

**Figura 40 - Código HTML 5 para inclusão de áudio.**

**Fonte:** Ewerton Mendonça.

**Descrição:** Código HTML 5 para inclusão de áudio em uma página web.

- **<audio>** Define o conteúdo de áudio.
- **<source>** Define múltiplas fontes de formatos diferentes para o elemento de áudio. Caso o primeiro não possa ser reproduzido, ele será substituído pelo próximo.

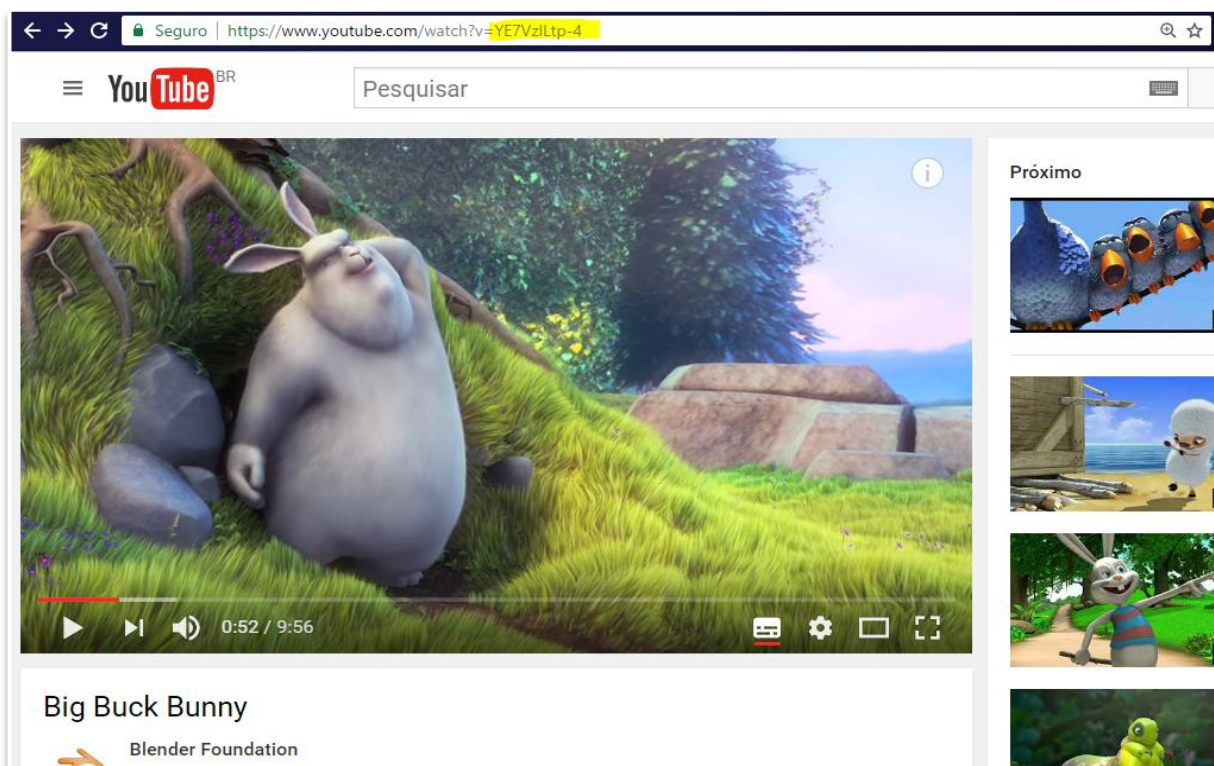
Em **<audio>**, assim como na tag de vídeo, temos o atributo **'controls'** da mesma forma que o anterior, para adicionar controladores de vídeo. Em **<source>** o atributo **'src'** guarda a URL do vídeo, já o atributo **'type'** guarda o tipo de vídeo, em HTML 5 temos **'audio/ogg'**, **'audio/mp3'** e **'audio/wav'**.

## 3.3 Youtube

Colocar áudio e vídeo em sua página pode dar muita dor de cabeça por conta da complexidade existente nos diversos formatos e configurações de vídeo. Pode ser até que você tenha que converter de um formato para o outro, perdendo qualidade com isso, sem nem levar o tempo dessa conversão.

Uma solução mais fácil é permitir que o YouTube reproduza os vídeos em sua página da web.

O YouTube exibirá um ID (como YE7VzlTtp-4), quando você salvar (ou reproduzir) um vídeo. Você pode usar esse id e referir-se ao seu vídeo no código HTML.



**Figura 41 - em destaque o código identificador do vídeo.**

**Fonte:** [www.youtube.com/watch?v=YE7Vz1Ltp-4](https://www.youtube.com/watch?v=YE7Vz1Ltp-4)

**Descrição:** página do YouTube com a exibição de um vídeo e sua URL.

Para reproduzir o vídeo em uma página da Web, faça o seguinte:

- 1. Carregar o vídeo para o YouTube;
- 2. Tome nota da id de vídeo;
- 3. Definir um elemento <iframe> na sua página da Web;
- 4. Deixe o atributo src apontar para o URL do vídeo;
- 5. Use os atributos width e height para especificar a dimensão do player;
- 6. Adicione outros parâmetros ao URL (veja a figura 42);

```
<iframe width="420" height="315" src="https://www.youtube.com/watch?v=YE7Vz1Ltp-4">
</iframe>
```

**Figura 42 - código HTML para inserção de vídeo do YouTube em uma página web.**

**Fonte:** Ewerton Mendonça.

**Descrição:** Código HTML 5 para inserção de vídeo do YouTube.





Você pode fazer com que seu vídeo comece a ser reproduzido automaticamente quando um usuário visita essa página adicionando um parâmetro simples ao seu URL do YouTube. Mas tenha cuidado com isso. A maioria dos usuários se irritam com esse procedimento.

- Valor 0 (padrão): o vídeo não é reproduzido automaticamente quando ele é carregado.
- Valor 1: o vídeo é reproduzido automaticamente quando ele for carregado.

Também podemos adicionar controles ao tocador, da mesma forma que a anterior, utilizando o parâmetro **controls** e os valores anteriores. A figura 43 mostra como colocar as duas coisas ao mesmo tempo.

```
<iframe width="420" height="315"
src="https://www.youtube.com/watch?v=YE7Vz1Ltp-4?autoplay=1&controls=1">
</iframe>
```

Figura 43 - código HTML para inserção de vídeo do YouTube em uma página web com autoplay e controladores.

Fonte: Ewerton Mendonça.

Descrição: código HTML 5 para inserção de vídeo do YouTube.





## 4. Competência 04 | Conhecer os Fundamentos do CSS 3

Agora que estruturamos todo o conteúdo do site em páginas HTML e temos até um sistema de navegação funcional, vamos estilizá-lo utilizando outro padrão da W3C, o CSS.

Alertamos, mais uma vez, que esta competência é a continuação das anteriores. Caso você tenha pulado competências anteriores dificilmente compreenderá esta. Você deve, então, voltar e ler o conteúdo, fazer o exercício proposto para poder seguir adiante.

CSS é um acrônimo para Cascading Style Sheets, traduzindo, significa Folhas de Estilo em Cascata. Este padrão serve para descrever como os elementos HTML são exibidos.

No começo da WWW o HTML também era responsável pela aparência do documento. Com o crescimento, os estilos das páginas ficaram mais complexos e se misturaram à estrutura e ao conteúdo, o que fazia com que os documentos HTML ficassem gigantes e confusos. Mas não era só este problema que existia. Se você tivesse um site com 200 páginas precisava copiar o estilo, ou parte dele, em todas as páginas. Se uma alteração tivesse que ser feita, deveria ser feita em cada uma das páginas.

Com o CSS temos um arquivo com a extensão .css onde ficam as configurações dos elementos. Como o estilo está fora do arquivo HTML, podemos utilizar o mesmo CSS para todas as páginas. Se alterarmos o CSS todas as páginas seguirão as modificações. Isso poupa muito tempo e trabalho.

Outra vantagem do CSS é que podemos ter várias folhas aplicadas no mesmo arquivo HTML. A primeira é sempre a do navegador. Aquele estilo padrão que vemos durante a construção do site de exemplo, quando aplicamos uma folha em uma página. As configurações desta folha se sobrepõem a do navegador. Os elementos que não foram definidos na folha continuam com a configuração da folha do navegador. Se houver mais uma configuração, ela se sobrepõe à primeira e à folha do navegador. E assim por diante. Por isso que é nomeada “em Cascata”.

Vamos começar então. No documento index.html vamos “linkar”, ou seja, criar uma ligação entre esta página e um arquivo CSS que criaremos posteriormente. Para isso, no <head> vamos acrescentar uma tag chamada <link>. Ela possui alguns atributos padrão, então você pode copiar o atributo href que contém a URI com a localização do arquivo CSS. Observe na figura 44.



```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Fulano de Tal</title>
5     <meta name="author" content="Ewerton Mendonça">
6     <meta name="description" content="Site demonstração do curso de Web Design do EAD Pernambuco.">
7     <meta charset="utf-8">
8
9     <link rel="stylesheet" type="text/css" href="estilo.css">
10  </head>
11  <body>
12    <section>
13      <header>
14        <h1><span>Fulano de Tal</span></h1>
15      </header>
```

Figura 44 - tag <link> relacionando uma folha de estilo à página index.html.

Fonte: Ewerton Mendonça.

Descrição: destaque da tag no código HTML que liga uma página HTML ao arquivo com o CSS.

Crie também um arquivo de texto na pasta do projeto com o nome estilo.css. O mesmo nome referenciado na tag <link>. Você pode utilizar o Bloco de Notas do Windows, ou seu editor HTML preferido.

## 4.1 Regras CSS

Como o arquivo estilo.css está vazio, não observaremos, por agora, qualquer modificação na renderização da página index.html. Antes disso, vamos entender como funciona a sintaxe CSS.

Para definir um estilo para algum elemento HTML precisamos dizer qual é esse elemento e suas regras. Assim, o estilo é definido por um seletor e um bloco feito de chaves { }, com as regras que definem a aparência daquele elemento. Uma regra é composta por uma propriedade e um valor. Observe a figura 45 que mostra a estrutura de um estilo, temos como seletor o elemento <body> e duas regras.

```
SELETOR
p {
  background: #7CFC00;
  PROPRIEDADE: VALOR;
  color: #FFFF00;
  REGRAS
}
```

Figura 45 - o que está em preto é a estilização no arquivo CSS. Em azul, está a explicação dos elementos.

Fonte: Ewerton Mendonça.

Descrição: regra CSS com sua estrutura destacada.



## 4.2 Seletores

O exemplo da figura 46 mostra a tag <p> como seletor. O seletor serve para achar o elemento HTML que será modificado, no caso do exemplo são todas as tags <p> daquela página. Note que relacionamos apenas uma página ao CSS, então apenas ela será alterada.

Ainda podemos utilizar o atributo id como seletor. A vantagem do atributo id é que seu valor deve ser único naquela página onde se encontra. Para utilizarmos um id como seletor colocamos um símbolo de tralha (#) na frente do valor do identificador, figura 46.

```
#centro {  
    background: #7CFC00;  
    color: #FFFF00;  
}
```

**Figura 46 - utilização do atributo id como seletor.**

**Fonte:** Ewerton Mendonça.

**Descrição:** regra utilizando o id como seletor.

Quando temos vários elementos com a mesma estilização, podemos utilizar o atributo class para formar um grupo. O atributo class pode repetir o mesmo valor agrupando elementos HTML. A forma de selecionar um class é colocar um ponto à frente do valor do class, figura 47.

```
.nomedaclass {  
    background: #7CFC00;  
    color: #FFFF00;  
}
```

**Figura 47 - utilização do atributo class como seletor.**

**Fonte:** Ewerton Mendonça.

**Descrição:** regra utilizando o identificador de class como seletor.



Quando vários seletores possuem as mesmas regras, não precisamos repetir as regras. Separamos os diversos seletores com vírgula, figura 48.

```
p, #centro, .nomedaclass {  
    background: #7CFC00;  
    color: #FFFF00;  
}
```

**Figura 48** - neste exemplo a tag <p>, o elemento com id igual a centro e o elemento com class igual a nomedaclass recebem as mesmas regras.  
**Fonte:** Ewerton Mendonça.  
**Descrição:** código CSS.

Por último, podemos especificar um elemento. Para isso fazemos uma lista sem separação. No exemplo da figura 49, todas as tags <p> englobadas pelo elemento com o id igual a centro são modificadas.

```
#centro p {  
    background: #7CFC00;  
    color: #FFFF00;  
}
```

**Figura 49** - especificação de elementos.  
**Fonte:** Ewerton Mendonça.  
**Descrição:** código CSS.

No CSS você pode colocar quantas regras precisar e até, se for o caso, dividir as regras entre dois seletores iguais. O navegador vai ler o primeiro, fazer as alterações e depois ler o segundo, também aplicando as alterações.



Normalmente quando uma regra não apresenta modificações é porque o nome do seletor está errado.

## 4.3 Cores

Alguns valores de atributos são definições de cores. Em CSS as cores podem ser descritas de várias formas. As mais usadas são através de código hexadecimal e palavras-chaves referentes a uma cor. A figura 50 mostra as duas definições de cores, referentes ao vermelho. O código hexadecimal começa sempre por tralha (#).

```
p {  
    background: red;  
    color: #FF0000;  
}
```

Figura 50 - representação de cores em CSS.

Fonte: Ewerton Mendonça.

Descrição: código CSS para cores.



Você pode ter acesso às palavras-chaves neste link:  
[www.w3schools.com/cssref/css\\_colorsfull.asp](http://www.w3schools.com/cssref/css_colorsfull.asp)



O código hexadecimal possui uma gama de cores bem maior que as palavras-chaves. Este link lhe dá acesso a uma página que pode lhe fornecer o código hexadecimal através de um seletor de cores:  
[http://www.w3schools.com/tags/ref\\_colorpicker.asp?colorhex=000000](http://www.w3schools.com/tags/ref_colorpicker.asp?colorhex=000000)



## 4.4 Background

Através da propriedade background podemos modificar o fundo do elemento. Na figura 51 temos as propriedades relativas ao background.

Propriedade	Descrição
background-color	Define a cor de fundo do elemento.
background-image	Define uma imagem para o elemento.
background-repeat	Define a repetição da imagem.
background-attachment	Define o comportamento da imagem quando rolamos a página no navegador.
background-position	Define a posição da imagem no fundo do elemento.
background	Esta propriedade reúne os efeitos anteriores, desde que na ordem em que foram apresentados nesta tabela, de cima para baixo. Pode ter ausências.

**Figura 51 - representação de cores em CSS.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código CSS para cores.

No site de exemplo vamos colocar o fundo preto, com uma imagem grande o bastante para cobrir todo o fundo. A imagem não irá se repetir, o “attachment” será fixo se o usuário rolar a página, e a posição da imagem será centralizada em relação ao elemento. Observe o código e o resultado na figura 52.







**Figura 52 - CSS do background de <body> e seu resultado no navegador.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código CSS no Bloco de Notas e o resultado do código no navegador, com a exibição de uma imagem de fundo em que aparecem pessoas.

## 4.5 Text

As propriedades referentes ao texto mais usadas estão listadas na figura 53. Algumas propriedades são herdadas pelos seus elementos filhos. Caso isso aconteça e você não queira, terá que redefinir os elementos que mudaram.

Propriedade	Descrição
color	Define a cor do texto no elemento.
text-align	Define o alinhamento do texto no elemento.
text-decoration	Serve para definir e remover a decoração do texto, como o sublinhado.
text-transform	Modifica o texto para maiúsculas, minúsculas, capitulares, etc.
text-indent	Define o recuo do texto no início do parágrafo.

**Figura 53 - propriedades referentes à decoração de texto.**

**Fonte:** Ewerton Mendonça.

**Descrição:** tabela relacionando as propriedades de cor com sua descrição.

Observe em nosso exemplo a última linha. Ela está na tag <address>. Vamos deixar toda em maiúscula e centralizada. Veja o código e o resultado no navegador na figura 54.





**Figura 54 - código da mudança das propriedades do texto e seu resultado.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código CSS no Bloco de Notas e o resultado do código no navegador, com mudança no estilo do texto.

## 4.6 Font

O navegador utiliza as fontes instaladas no computador para exibir o texto. Caso você defina uma fonte que não exista naquele computador, o navegador tentará substituí-la. Quando a fonte que você utilizar não for popular, melhor utilizar uma imagem no lugar. Você pode utilizar um termo genérico como serif, sans-serif e monospace. Nesse caso, você não diz a fonte e sim o seu tipo. E o navegador tenta encontrar a melhor fonte daquele tipo no computador do usuário. Podemos modificar as propriedades da fonte do texto através da propriedade font, figura 55.

Propriedade	Descrição
font-family	Define a família de fontes para aquele elemento. Podemos colocar várias opções separadas por vírgulas. Se houver nomes compostos na fonte, utilizamos aspas duplas. Exemplo: "Times New Roman"
font-style	Define o estilo como normal, itálico ou oblíquo.
font-size	Define o tamanho da fonte do texto daquele elemento. O tamanho pode ser definido de várias formas. A mais utilizada é px (pixel) ou pt (pontos).
font-weight	Define a largura da fonte. Seria o nível de negrito.
font	Configura todas as propriedades acima em uma declaração. Pode haver ausências, mas a ordem acima deve ser respeitada.

**Figura 55 - efeitos relativos à propriedade font para textos.**

**Descrição:** tabela relacionando as propriedades de fonte com sua descrição.

**Fonte:** Ewerton Mendonça.

Vamos configurar a font de vários de nossos elementos. Note na figura 56 que colocamos todo o texto de <body> com a cor branca e todos os seus elementos internos herdaram esta propriedade. Para poder se notar, retiramos temporariamente a imagem de fundo. Também modificamos os links <a> para serem brancos e retiramos a decoração (sublinhado).



**Figura 56 - exemplo de utilização das propriedades de font.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código CSS no Bloco de Notas e o resultado do código no navegador, com mudança no estilo da fonte.

## 4.7 Links

Quando no exemplo eu tirei a decoração (sublinhado) dos links, o usuário perdeu o indicativo de que ali se encontra um hiperlink, que no caso é o sublinhado. Vamos fazer com que, quando o usuário esteja com o ponteiro por cima do link, apareça o sublinhado.

Para isso, vamos modificar um dos quatro estados do link. Abaixo estão todos eles.

- link: link normal, não visitado.
- visited: link já visitado.
- hover: estado quando o ponteiro do mouse está sobre ele.
- active: estado no momento em que é clicado.

Estes estados do elemento <a> só existem para ele. Os estados não são propriedades, é outra forma de seletor para a tag <a>. Para utilizarmos, colocamos o seletor "a" seguido de dois pontos e o estado que desejamos modificar. Veja no exemplo da figura 57.



**Figura 57 - utilização de estados da tag <a> com CSS.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código CSS no Bloco de Notas e o resultado do código no navegador, com mudança no estilo dos links.

Note no destaque em vermelho a modificação do estado hover de <a> para exibir a decoração sublinhada (*underline*). No navegador observe que “DISCOGRAFIA” possui um sublinhado porque o ponteiro está repousado em cima dele.

## 4.8 List

A propriedade list estiliza as listas HTML. figura 58 mostra as propriedades referentes à list.

Propriedade	Descrição
list-style-image	Define uma imagem para o marcador.
list-style-position	Define se o marcador fica dentro ou fora do fluxo.
list-style-type	Define o tipo do marcador. Tipos: circle, square, upper-roman, lower-alpha, entre outros.
List	Configura todas as propriedades acima em uma declaração. Pode haver ausências, mas a ordem acima deve ser respeitada.

**Figura 58 - propriedades relacionadas à formatação de listas.**

**Fonte:** Ewerton Mendonça.

**Descrição:** tabela relacionando as propriedades de lista com sua descrição.



Temos uma lista em nosso site e queremos modificar de círculo para quadrados. No entanto, percebe-se que no exemplo da figura 57 a página exibida no iframe não pode ser vista. Isso acontece porque seu texto pertence a outro HTML e não herda o CSS do index.html. Como queremos demonstrar da melhor forma, vamos criar um estilo chamado **estilopaginas.css** e no <head> de cada página, com exceção da index.html, vamos relacionar a este novo CSS. Assim, a index.html fica relacionada ao **estilo.css** e o restante das páginas ao **estilopagina.css**. Cada página com um CSS diferente.

O código do novo arquivo CSS e o resultado da estilização da list está na figura 59. Aproveitamos e definimos as configurações para <p> e o título em <h2>. Compare com as configurações no outro estilo para entender melhor.



Figura 59 - arquivo estilopaginas.css, com as configurações da lista não ordenada <ul> estão destacadas em vermelho. Trocamos de “disc” para “square”, que é visível no navegador ao lado.

Fonte: Ewerton Mendonça.

Descrição: código CSS com a configuração da lista e seu resultado no navegador.



## 4.9 Border

Configurações de borda podem ser utilizadas em várias tags, vamos demonstrar no <iframe> e nas bordas. Para isso, vamos editar o arquivo estilo.css que define o estilo para index.html, a página que contém o <iframe>; e em estilopagina.css que define o estilo para as outras páginas, incluindo agenda.html, que contém a tabela.

A figura 60 traz uma tabela com as propriedades referentes ao atributo border. Alertamos que a propriedade de estilo é obrigatória para visualização.

Propriedade	Descrição
border-bottom	Define todas as propriedades da borda inferior do elemento em uma única declaração.
border-bottom-color	Define a propriedade de cor da borda inferior.
border-bottom-style	Define a borda inferior com o estilo: solid, dotted, double ou dashed.
border-bottom-width	Define a largura da borda inferior.
border-left	Define todas as propriedades da borda esquerda do elemento em uma única declaração.
border-left-color	Define a propriedade de cor da borda esquerda.
border-left-style	Define a borda esquerda com o estilo: solid, dotted, double ou dashed.
border-left-width	Define a largura da borda esquerda.
border-right	Define todas as propriedades da borda direita do elemento em uma única declaração.
border-right-color	Define a propriedade de cor da borda direita.
border-right-style	Define a borda direita com o estilo: solid, dotted, double ou dashed.
border-right-width	Define a largura da borda direita.
border-top	Define todas as propriedades da borda superior do elemento em uma única declaração.
border-top-color	Define a propriedade de cor da borda superior.
border-top-style	Define a borda superior com o estilo: solid, dotted, double ou dashed.
border-top-width	Define a largura da borda superior.
border-color	Define a propriedade de cor de todas as bordas.
border-style	Define a propriedade do estilo de todas as bordas.
border-width	Define a propriedade de largura de todas as bordas.
Border	Define todas as propriedades da borda do elemento em uma única declaração.

**Figura 60 - propriedades relacionadas à formatação de tabelas.**

**Fonte:** Ewerton Mendonça.

**Descrição:** tabela relacionando as propriedades de tabela com sua descrição.





A figura 61 mostra as configurações de <table> em estilopagina.css. Colocamos uma borda de 2 pixels de largura, na cor cinza e com traço sólido. Ainda definimos a cor de background de <td> para o mesmo tom de cinza do traço. Não alteramos o cabeçalho <th>.

Note que a borda do <iframe> desapareceu. Para isso, definimos **border: 0px** em estilo.css, que governa a aparência de index.html.

```
11 }
12
13 p {
14     font-size: 12px;
15 }
16
17 a {
18     color: #FFFFFF;
19     font-family: Arial, Helvetica, sans-serif;
20     font-weight: bold;
21     font-size: 8pt;
22 }
23
24 a:hover {
25     color: #999999;
26 }
27
28 ul {
29     font-size: 12px;
30     list-style-type: square;
31 }
32
33 table {
34     width: 100%;
35     border: 2px #333333 solid;
36     font-size: 12px;
37 }
38
39 td {
40     background-color: #333333;
41 }
```



Figura 61 - configuração de <table> com uma borda de 2 pixels de largura, na cor cinza e o traço sólido. Ainda definimos a cor de background de <td>.

Fonte: Ewerton Mendonça.

Descrição: código CSS com a configuração da tabela e seu resultado no navegador.



## 5.Competência 05 | Planejar Layouts com CSS 3

Se você está prestando atenção no curso, deve ter percebido uma propriedade na figura 61 relacionada ao seletor `table` chamada `width` que não explicamos. Essa propriedade define a largura do elemento. Mas para entender como determinamos a largura do elemento precisamos compreender melhor como é construído o elemento.

### 5.1 Modelo de caixa

Todo elemento HTML pode ser comparado com uma caixa, daí o nome Modelo de Caixa. O modelo de caixa basicamente é um retângulo que envolve o elemento, seja ele qual elemento for. Ele é composto de alguns elementos, veja a figura 62, que são:

- **Margin:** área transparente ao redor da borda;
- **Border:** borda que circula o preenchimento do conteúdo;
- **Padding:** área transparente que envolve o conteúdo;
- **Content:** conteúdo da caixa, onde o texto ou a imagem aparece.



Figura 62 - modelo de Caixa.

Fonte: [www.w3schools.com/css/box-model.gif](http://www.w3schools.com/css/box-model.gif)

**Descrição:** a imagem é composta de quatro caixas, uma dentro da outra, sendo a mais interna o conteúdo do elemento e subsequentemente o espaçamento, a borda e a margem.



Quando você define a altura ou a largura de um elemento, você está definindo a área disponível para o conteúdo. Então, você deve somar os pixels de margem, da borda, do espaçamento interno, multiplicado por dois ao tamanho do conteúdo. Isso porque temos dois lados horizontalmente ou dois lados verticalmente.

Por exemplo: se o conteúdo tem 100 pixels de largura, a margem de 20 pixels, a borda de 5 pixels e o espaçamento interno de 10 pixels. Temos  $(20 + 5 + 10) * 2 + 100 = 170$ . Width deve ser 170px.

## 5.2 Width e Height

A propriedade width define a largura, e a height define a altura. Elas podem receber valores em pixel ou em porcentagem. A porcentagem é em relação aos elementos que estão dentro.

No caso de nosso exemplo: um width de 100% deixará o elemento com a largura total de onde ele estiver. Como a <table> está dentro do <body>, ele ocupará todo o espaço disponível.

Ainda temos min-width e min-height que definem respectivamente a largura e a altura mínima.

Se você verificar nossa imagem de referência, a layout, veremos que o conteúdo do site é exibido em uma coluna pequena que ocupa toda a altura, 100%. Na figura 63 você pode ver o código de definição de altura e largura do resultado exibido no navegador. Voltamos com a imagem de background em <body> para você ver melhor o efeito.



Figura 63 - definição das propriedades de largura e altura da tag <section>.

Fonte: Ewerton Mendonça.

Descrição: código CSS com a configuração da largura e altura de uma tag e seu resultado no navegador.

No entanto, a altura mínima não foi respeitada. Saberemos o porquê no próximo tópico.

## 5.3 Position

As propriedades de posicionamento CSS permitem que você posicione um elemento. Elas também podem colocar um elemento atrás do outro e especificar o que deve acontecer quando o conteúdo de um elemento é muito grande.

Os elementos podem ser posicionados usando as propriedades top, bottom, left e right. No entanto, essas propriedades não funcionarão a menos que a propriedade position seja definida em primeiro lugar. Eles também funcionam de forma diferente, dependendo do método de posicionamento.



Existem quatro métodos de posicionamento diferentes:

- **Static:** é a padrão. Posiciona-se pelo fluxo normal da página.
- **Fixed:** será posicionado em relação à janela do navegador.
- **Relative:** será posicionado em relação à posição normal.
- **Absolute:** será posicionado em relação ao primeiro elemento estático que encontrar.

Se não houver, será em relação a <html>.

Podemos colocar elementos sobrepostos utilizando a propriedade z-index e especificando uma posição. Normalmente, quando queremos definir que o elemento fique abaixo de todos, colocamos o valor -1000.

Compare a figura 63 com a 64 e veja como a <section> agora chega até embaixo.



Figura 64 - definição da propriedade position de fixed para <section> ativa a propriedade min-height.

Fonte: Ewerton Mendonça.

Descrição: código CSS com a configuração do posicionamento de uma tag e seu resultado no navegador.



O site de exemplo começa a ganhar forma, mas queremos posicioná-lo à direita. Para isso, utilizamos a propriedade `right` em `0px`. Posicionando `<section>` colado com a borda direita da janela. Observe na figura 66.

## 5.4 Magin e Padding

Nosso exemplo ficou junto da borda direita da janela, mas não toca a parte de cima nem a de baixo. Isto acontece por conta da configuração padrão de `<html>` que possui uma margem e um espaçamento interno, definidos.

Todos os elementos seguem o modelo box e possuem uma margem e um espaçamento. Podemos defini-los utilizando as propriedades na tabela da figura 65.

Propriedade	Descrição
<code>margin-top</code>	Configura a margem superior.
<code>margin-right</code>	Configura a margem à direita.
<code>margin-bottom</code>	Configura a margem inferior.
<code>margin-left</code>	Configura a margem à esquerda.
<code>margin</code>	Define todas as propriedades da margem do elemento em uma única declaração. A ordem deve ser a seguida acima, no sentido horário.
<code>padding-top</code>	Configura o espaçamento interno superior.
<code>padding-right</code>	Configura o espaçamento interno à direita.
<code>padding-bottom</code>	Configura o espaçamento interno inferior.
<code>padding-left</code>	Configura o espaçamento interno à esquerda.
<code>padding</code>	Define todas as propriedades de espaçamento interno do elemento em uma única declaração. A ordem deve ser a seguida acima, no sentido horário.

Figura 65 - tabela com as propriedades de margem e espaçamento internos.

Fonte: Ewerton Mendonça.

Descrição: tabela que relaciona a propriedade a sua descrição.

Em todas essas propriedades podemos definir valores em pixels e em porcentagem. A propriedade `margin` ainda tem um valor chamado `auto`, que pega o espaçamento restante e coloca metade em cada lado. Em layouts simples podemos utilizá-la para centralizar elementos.

Em nosso exemplo vamos definir a margem e o espaçamento de `<body>`. Essas propriedades são herdadas pelos elementos que estão dentro, todos para 0 pixel. Em `<section>` vamos definir o espaçamento interno do lado esquerdo como 10 pixels. Veja o código e o resultado na figura 66.



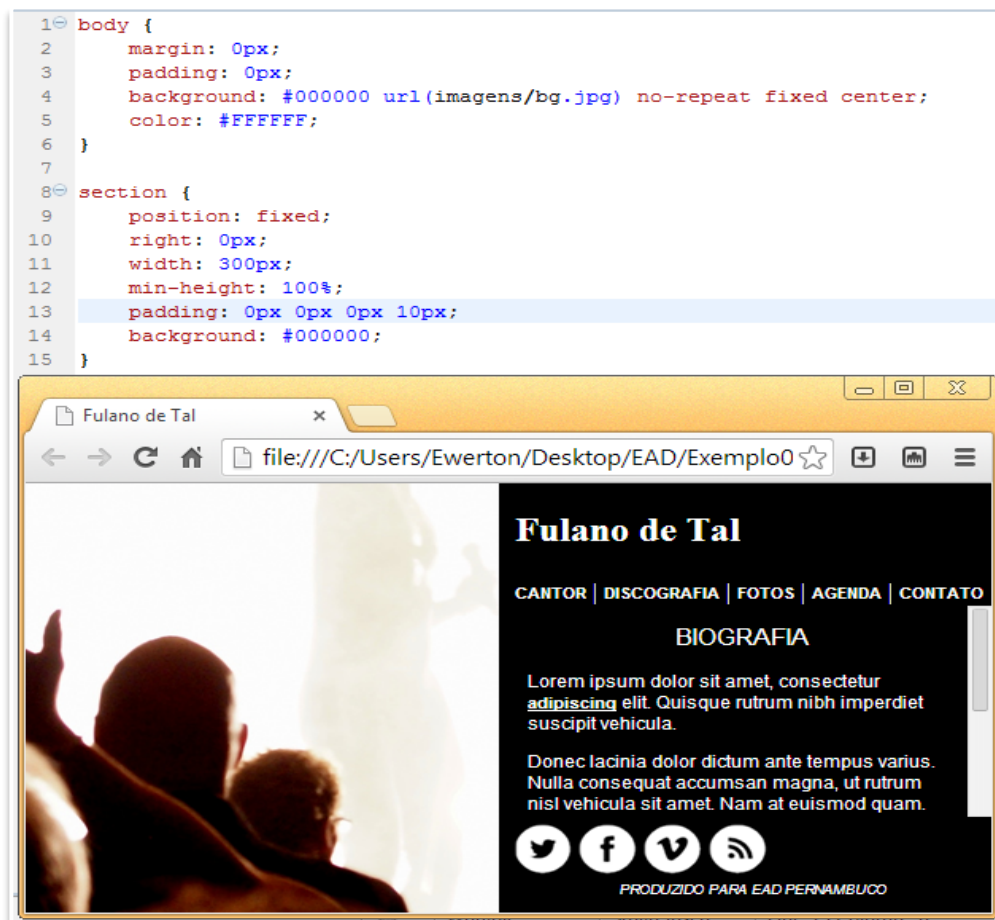
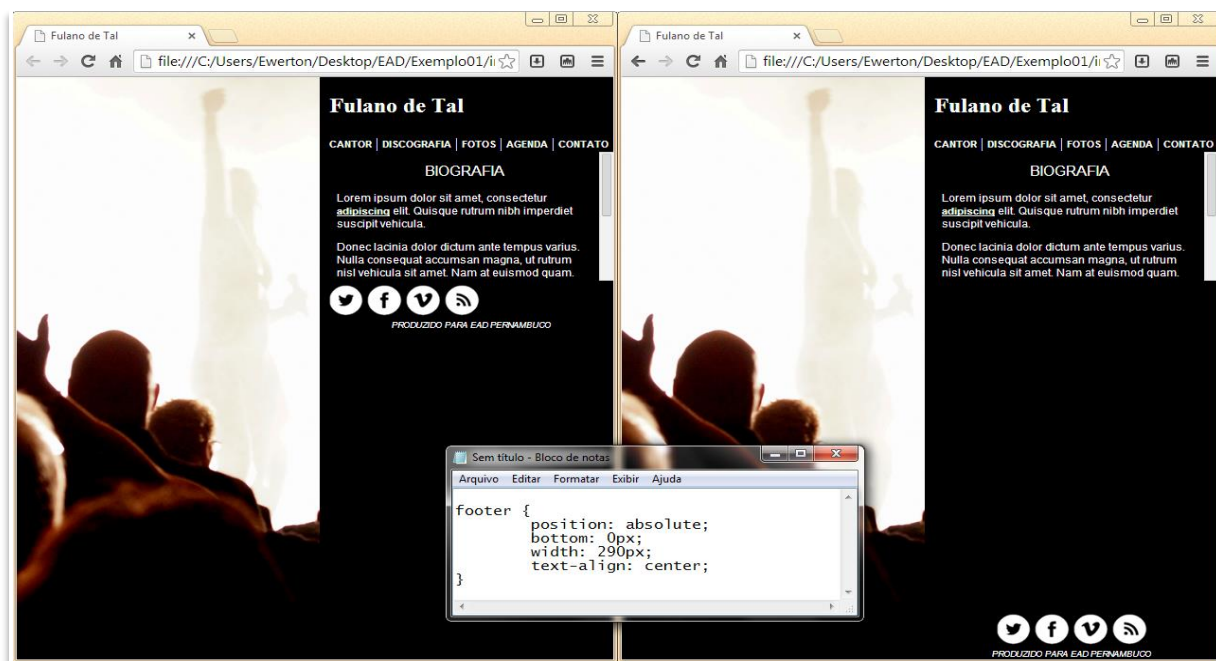


Figura 66 - configuração da margem e espaçamento interno dos elementos.

Fonte: Ewerton Mendonça.

**Descrição:** código CSS com a configuração de margem e espaçamento interno de uma tag e seu resultado no navegador.

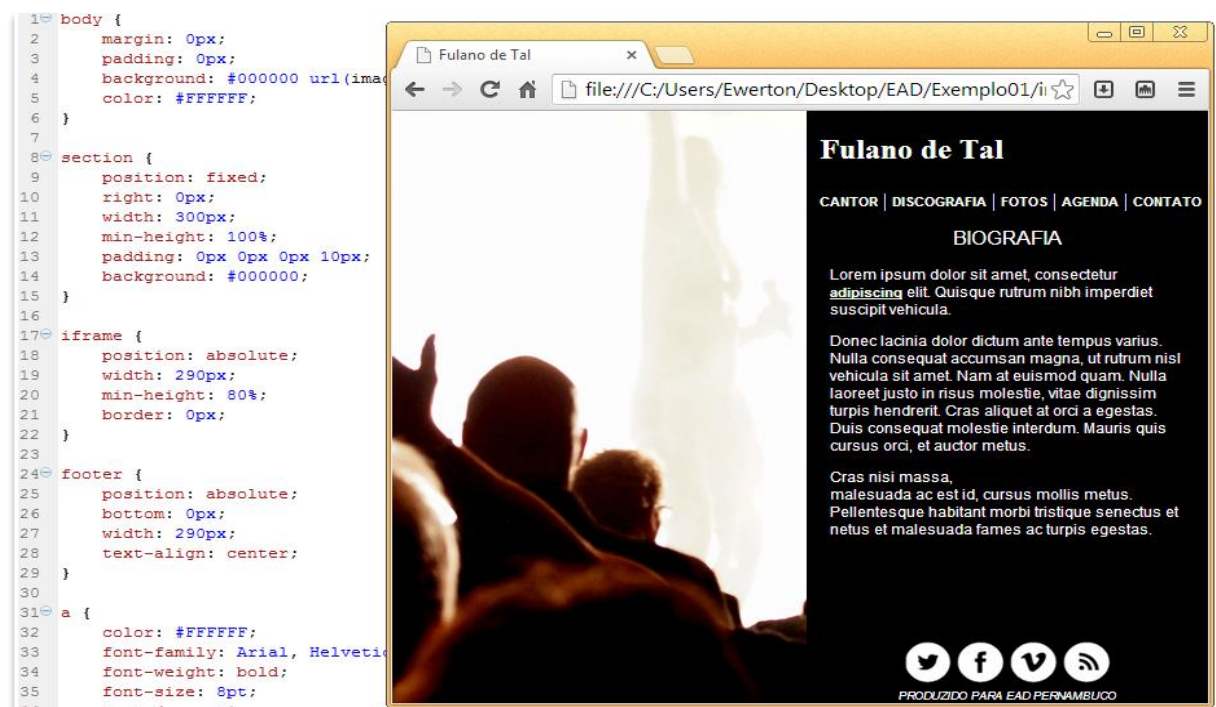
Parece que ele está funcionando bem, mas se esticarmos a janela o <footer> não fica encostado no fundo da página. Observe na figura 67, no navegador da esquerda, o problema. Vamos definir a configuração de <footer> como no código na figura 67. As configurações nós já explicamos anteriormente. Escreva e teste uma a uma das propriedades para ver o efeito de cada. O navegador da direita na figura 67 mostra o efeito resultante.



**Figura 67 - configuração do elemento <footer> para ficar a 0 pixel da borda inferior da janela. Aproveitamos e definimos um tamanho e centralizamos os elementos.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código CSS com a configuração de margem e espaçamento interno de uma tag e seu resultado em dois navegadores para comparação do antes e depois.



**Figura 68 - Configurações de <iframe> para esticar o elemento.**

**Fonte:** Ewerton Mendonça.

**Descrição:** Código CSS com a configuração do iframe de uma tag e seu resultado no navegador.



## 5.5 Display

Para finalizarmos, vamos ensinar um truque. Mas para isso temos que aprender sobre a propriedade display. Ela define como o elemento será exibido, e se será exibido. Abaixo temos os valores e a explicação:

- **None:** retira o elemento da página. Ele existe, mas não é renderizado.
- **Block:** define o elemento como bloco.
- **Inline:** define o elemento em linha.

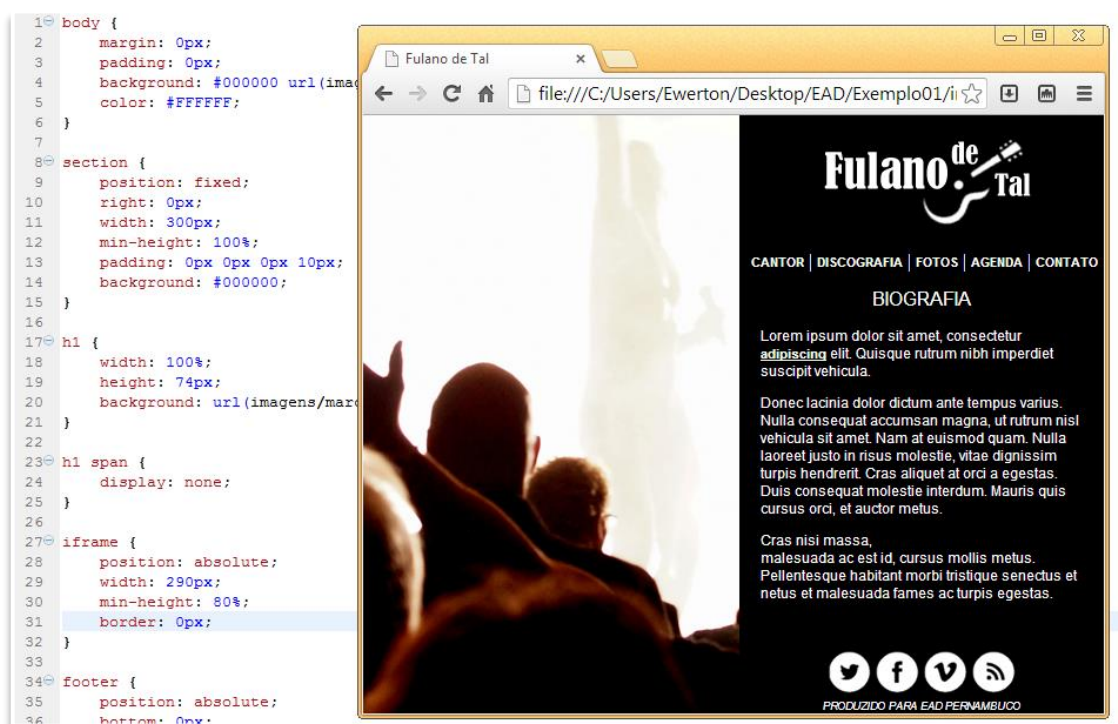
Os elementos em bloco como `<h1>` não permitem elementos ao seu lado. O próximo elemento após um elemento bloco será renderizado abaixo. Os elementos inline, como `<a>`, posicionam o próximo elemento à direita, se houver espaço. Como palavras em um texto.

Em nosso `index.html` colocamos o texto do título de `<h1>` englobado em uma tag `<span>` que serve apenas para marcar. Fizemos isso para esse truque.

Dissemos que imagens para “embelezar” não deveriam ser inseridas com a tag `<img>` e sim por CSS. Mas como fazemos isso?

Temos uma imagem com a marca do cantor do tamanho já correto. Vamos configurar a tag `<h1>` que possui o título para o tamanho da imagem e definir o background para exibir a imagem com a marca. O problema é que o texto ainda fica sendo exibido. Para fazer desaparecê-lo vamos configurar a tag `<span>` com `display none`. Como podemos ter vários `<span>` em outros lugares de nosso documento HTML especificaremos apenas as tags `<span>` dentro de `<h1>`.

O código e o resultado podem ser vistos na figura 69.



**Figura 69 - configuração de imagens por CSS.**

**Fonte:** Ewerton Mendonça.

**Descrição:** código CSS com a configuração de uma imagem e seu resultado no navegador.

## 5.6 Finalizando

Anteriormente colocamos vídeo em nosso site de exemplo. Podemos utilizar o CSS para configurar a forma de exibição deste vídeo.

O vídeo está sendo exibido, mas não está ocupando toda a tela. Vamos voltar para o arquivo estilo.css e definir o estilo de exibição do vídeo. Veja na figura 70 a configuração que utilizamos e na figura 71 o site em tela cheia. Utilizamos um id só para você ver a utilização de um seletor id.





```
41 h1 {  
42     width: 100%;  
43     height: 74px;  
44     background: url(imagens/marca.png) no-repeat center;  
45 }  
46  
47 h1 span {  
48     display: none;  
49 }  
50  
51 a {  
52     color: #FFFFFF;  
53     font-family: Arial, Helvetica, sans-serif;  
54     font-weight: bold;  
55     font-size: 8pt;  
56     text-decoration: none;  
57 }  
58  
59 a:hover {  
60     text-decoration: underline;  
61 }  
62  
63 #video_background {  
64     position: absolute;  
65     bottom: 0px;  
66     right: 0px;  
67     min-width: 100%;  
68     min-height: 100%;  
69     width: auto;  
70     height: auto;  
71     z-index: -1000;  
72     overflow: hidden;  
73 }
```

Figura 70 - configuração do estilo de exibição do vídeo em estilo.css.

Fonte: Ewerton Mendonça.

Descrição: código CSS que configura a exibição do elemento vídeo na página.

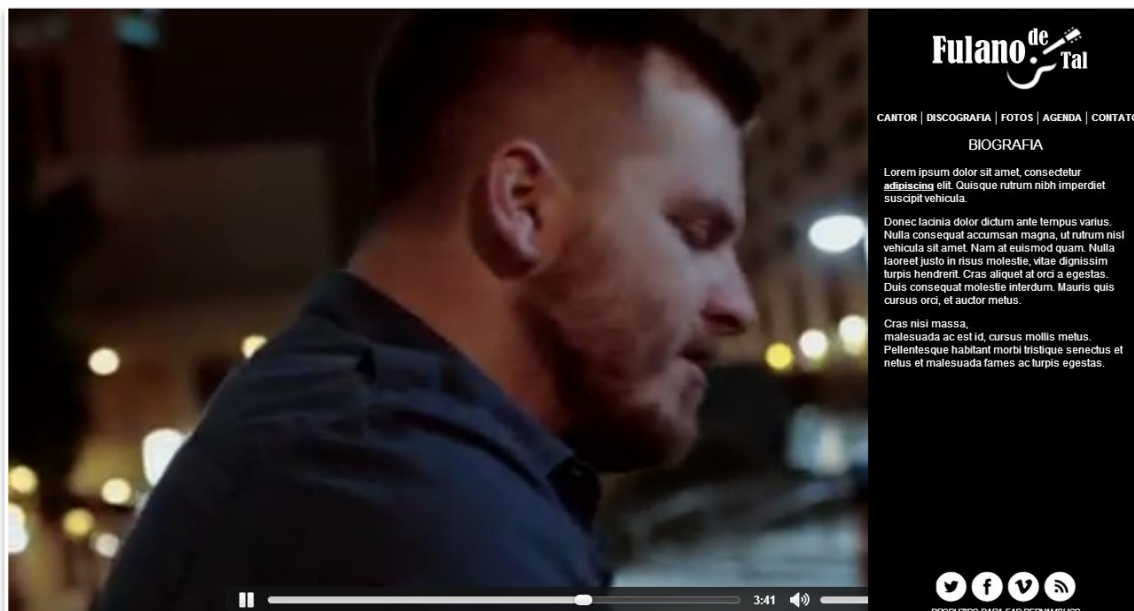


Figura 71 - site com o estilo configurado exibindo o vídeo como background.

Fonte: Ewerton Mendonça.

Descrição: exibição final da página no navegador.



Para dar um efeito de transparência em `<section>`, em `estilo.css` no seletor `section` vamos acrescentar a regra `'opacity: 0.9;'`. A propriedade `opacity` deixa o elemento translúcido. Seus valores vão de 0.0, completamente transparente, até 1.0, completamente opaco.

Não mostrarei como ficou para deixá-lo curioso. Depois de ler e fazer o exemplo, você já deve poder fazer a configuração facilmente.



Como sempre digo, é através da repetição que aprendemos. Quanto mais fizer, mais fácil será. Faça o layout da competência anterior com os conhecimentos adquiridos. Caso o site que você imaginou tenha alguma parte muito complicada, a internet é cheia de tutoriais e truques que irão lhe ajudar a realizar sua arte. O importante é você tentar.





## Conclusão

Ufa!

Foi muita coisa, não é mesmo? Mas lembre-se de que você sempre pode voltar atrás e ver novamente algum ponto que ficou confuso. Na web temos muitos tutoriais e até videoaulas sobre qualquer um desses pontos.

Foi um excelente investimento de tempo que você fez ao se dedicar a este curso, mas para se tornar um profissional será necessário mais pesquisa e investimento, afinal esta área de criação para web é gigantesca.

O autor do livro Fora de Série – Outliers, Malcolm Gladwell, cita uma pesquisa sobre pessoas que são consideradas profissionais em suas áreas. Na pesquisa ele explica que para chegar a um nível profissional são necessárias 10.000 horas dedicadas de forma apaixonada naquilo que se faz.

Gostando do que faz, você presta mais atenção e entende bem melhor do que pessoas que estudam por obrigação. Dessa forma, é fácil chegar às 10.000 horas. Pois é. Sei que são muitas horas estudando, testando código, fuçando e pesquisando sobre os assuntos vistos, mas se você se apaixonar pelo tema, não será um martírio, será sim, bem divertido.

Espero que você tenha gostado, tanto quanto eu, de fazer este curso.

Boa sorte ao entrar nesta maravilhosa área da profissão de web designer.



## Referências

ROBBINS, Jeninifer Niederst. **Aprendendo Web Design**: Guia para iniciantes. 3.ed. Porto Alegre: Bookman, 2010.

KALBACH, James. **Design de Navegação Web**: Otimizando a experiência do usuário. Porto Alegre: Bookman, 2009.

ZEMEL, Tácio. **Web design responsivo**: páginas adaptáveis para todos os dispositivos. São Paulo: Casa do Código, 2012.

SILVA, Mauricio Samy. **HTML 5**: A linguagem de marcação que revolucionou a WEB. São Paulo: Novatec Editora, 2011.

SILVA, Mauricio Samy. **CSS3**: Desenvolva aplicações web profissionais com o uso dos poderosos recursos de estilização das CSS3. São Paulo: Novatec Editora, 2012.

SILVA, Mauricio Samy. **Construindo sites com CSS e (X)HTML**: Sites controlados por folhas de estilo em cascata. São Paulo: Novatec Editora, 2008.

SILVA, Mauricio Samy. **Criando sites com HTML**: Sites de alta qualidade com HTML e CSS. São Paulo: Novatec Editora, 2008.

SILVA, Mauricio Samy. **JQuery**: A Biblioteca do Programador Javascript. 2.ed. São Paulo: Novatec Editora, 2010.

GLADWELL, Malcolm. **Fora de série**: outliers. Rio de Janeiro: Sextante, 2008.



## Minicurrículo do Professor



### Ewerton Mendonça

Formado em Sistemas de Informação pela UPE e Design pela UFPE, com mestrado em Ciência da Computação pela UFPE. Atualmente, é professor da Faculdade de Ciências e Letras de Caruaru – FAFICA e Unifavip Devry Brasil. Possui experiência na área de desenvolvimento WEB e design gráfico desde 1998.

