



Curso Técnico em Desenvolvimento de Sistemas Online

DESENVOLVIMENTO WEB

**GEEaD - Grupo de Estudos de
Educação a Distância
Centro de Educação Tecnológica
Paula Souza**

GEEaD – CETEC
GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
PROGRAMAÇÃO WEB I

Autores:
Paulo Eduardo Cardoso Andrade

Revisão Técnica:
Eliana Cristina Nogueira Barion
Lilian Aparecida Bertini

Revisão Gramatical:
Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:
Flávio Biazim

São Paulo – SP, 2019

APRESENTAÇÃO

Este material didático do Curso Técnico em Desenvolvimento de Sistemas modalidade EaD foi elaborado especialmente por professores do Centro Paula Souza para as Escolas Técnicas Estaduais – ETECs.

O material foi elaborado para servir de apoio aos estudos dos discentes para que estes atinjam as competências e as habilidades profissionais necessárias para a sua plena formação como Técnicos em Desenvolvimento de Sistemas.

Esperamos que este livro possa contribuir para uma melhor formação e aperfeiçoamento dos futuros Técnicos.

AGENDA 12

CSS -CASCADING STYLE SHEETS





MERGULHANDO NO TEMA...

CSS

O **CSS** (Cascading Style Sheets - Folhas de Estilo em Cascata) é a linguagem oficial para estilização de páginas **HTML**. Com ela é possível definir todas as características relacionadas à aparência dos elementos presentes em uma página, customizando cores, formas, fontes etc. Em conjunto ao HTML, o **CSS** compõe a base para o desenvolvimento de aplicações web, o que reforça a sua importância.

Quando o HTML foi criado, a intenção não era de formatar informação, dar cor ou dividir uma página em setores, então logo que o HTML foi se tornando popular, com as novas versões, foram sendo inseridas formas para alterar e controlar algumas aparências para o documento, como as cores, por exemplo, fazendo com que a linguagem ficasse mais complexa, e, portanto, mais difícil de entender e manter.

Outro problema era que os navegadores tinham diferenças de implementações, tornando diferente a visualização dos sites de navegador para navegador.

HåkonWium Lie, visualizando toda essa dificuldade, resolveu desenvolver um jeito mais fácil para formatar a informação, propondo então a criação do **CSS** ou Cascading Style Sheets. A partir de então, o **HTML** ficou focado para o que ele faz de melhor, que é a “marcação” e o **CSS** pelo estilo da página.

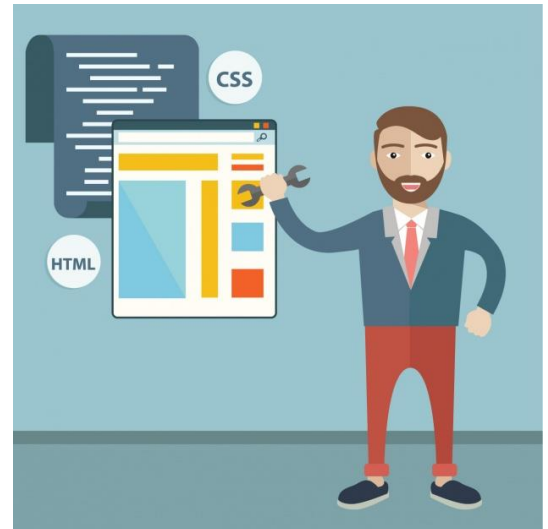
Tanto é que no **HTML 5** algumas tags foram retiradas de uso por terem como função a estilização da página como, por exemplo, a tag ``. Outras tag's retiradas do **HTML 5** podem ser vistas por meio dos links:

DevMedia

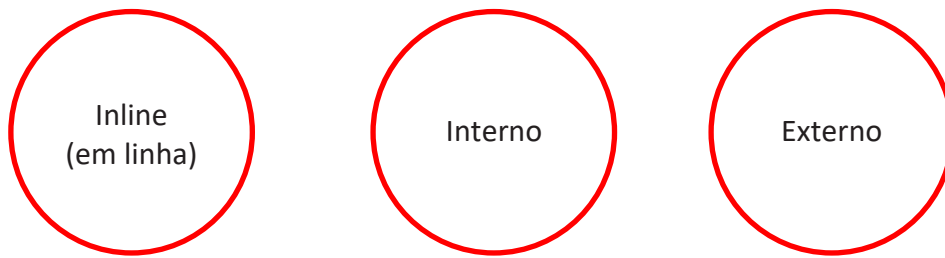
<https://www.devmedia.com.br/tags-e-atributos-depreciados-na-html/28042>

W3 (em inglês)

<https://www.w3.org/TR/html5-diff/#obsolete-elements>



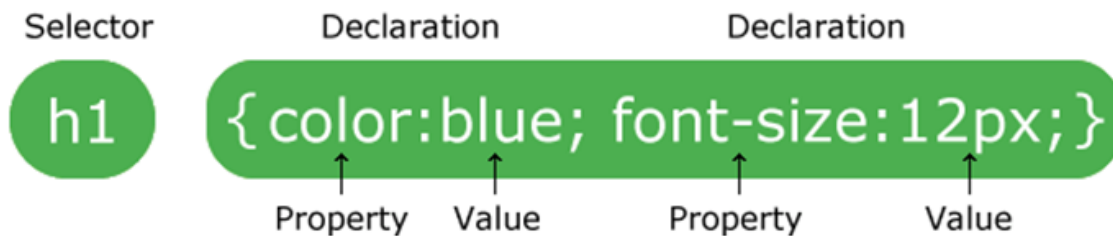
Existem **três maneiras de incluir códigos CSS** em um site:



Antes de começar a programar, vamos entender um pouco mais sobre a **sintaxe do CSS** e **seletores**.

Sintaxe e Seletores CSS

Basicamente a sintaxe do **CSS** consiste em um seletor e um bloco de declaração que contém propriedades (cor, alinhamento do texto) e valores (azul, centralizado), definindo, assim o estilo dos seletores. A imagem a seguir ilustra essa sintaxe:



- **H1** - o seletor da tag h1;
- **Chaves** - referenciam o início e o fim das declarações das propriedades valores;
- **Color** - propriedade;
- **Blue** - valor da propriedade.

Essa sintaxe muda um pouco quando o **CSS** é utilizado em linha.

Agora, vamos parar de teoria e vamos aprender programando, começando por **CSS INLINE**.

Inline

Para entender melhor essa maneira de incluir códigos CSS, crie um arquivo **HTML** e salve com o nome de "**css Inline**", não se esquecendo de criar a estrutura básica.

Como o nome diz, o código **CSS** é inserido na tag **em linha**.

Quando utilizamos o **CSS em linha** o seletor, a tag e a declaração da propriedade que você deseja estilizar é realizada por meio do atributo **style**. Para exemplificar, programe a seguinte linha de código.

```
<h1 style="color:blue;">Aprendendo  
CSS!</h1>
```

O código completo deve ficar como demonstrado a seguir:

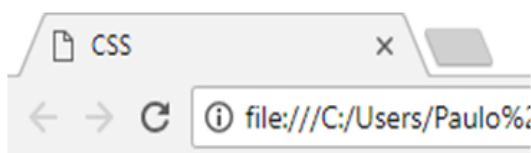
```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title> CSS</title>
</head>
<body>
<h1 style="color:blue;">Aprendendo CSS!</h1>
</body>
</html>
```

Vamos à explicação: a tag `<h1>`, já estudada, está com o atributo `style` atribuído com a declaração `"Color:Blue;"`, ou seja, a propriedade `Color` com o valor `blue` define a cor do texto marcado pela tag `h1` (Seletor). Repare que, nesse exemplo, apenas uma propriedade foi declarada.

Podemos observar um quadro azul após os dois pontos, conforme imagem ao lado. Esse é mais um recurso oferecido pelo `IntelliSense` do Visual Studio Code, demonstrando qual é a cor escolhida para a propriedade.



O resultado pode ser visto na imagem a seguir:



Aprendendo CSS!

Interno

Crie um arquivo `HTML` e salve com o nome de `"cssInterno"`. Não se esqueça de criar a estrutura básica.

Esta forma `interna` ou `em bloco`, codifica o `CSS` em uma tag localizada dentro da tag `head` (Cabeçalho). Trata-se da tag `<style></style>`, então, tudo o que estiver entre esta tag será interpretado pelo navegador como código `CSS`.

Programa entre as tags `<head>` e `</head>`. O código completo deve ficar como demonstrado a seguir:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title> CSS </title>
<style>
h1 {
color:blue;
}
</style>
</head>
<body>
<h1>AprendendoCSS!</h1>
</body>
</html>
```

A explicação é basicamente a mesma em torno da declaração de “**color: blue;**”, ou seja, a **propriedade** Color com o **valor** blue define a cor do texto do seletor **h1** e o resultado é o mesmo como pode ser observado na Imagem a seguir:



Aprendendo CSS!

Nesses exemplos, apesar dos resultados para o uso do **CSS inline** e **interno** serem os mesmos, eles possuem diferenças quando são utilizados mais códigos.

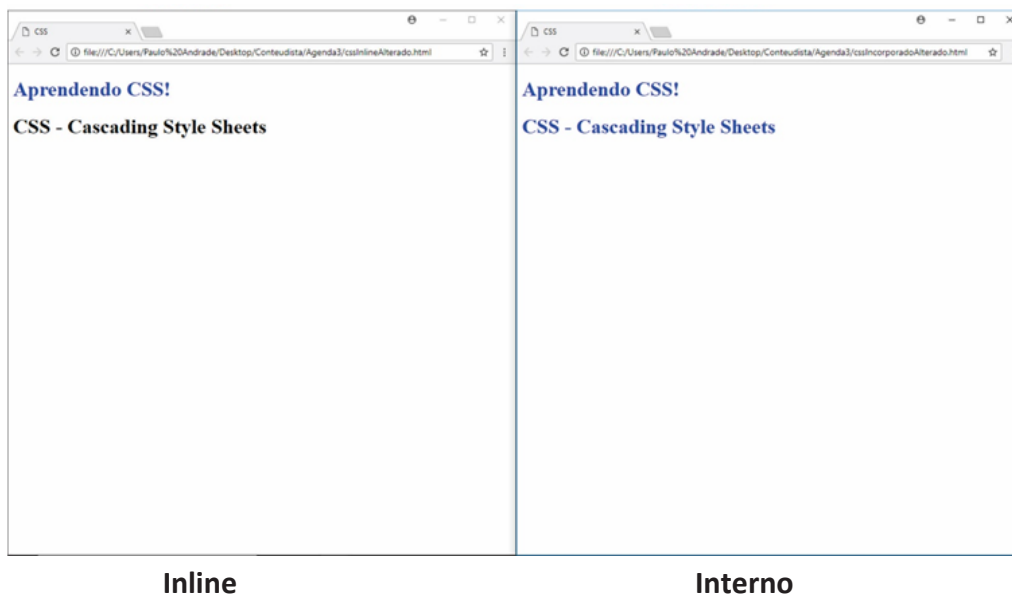
Vamos a um exemplo:

Alterando um pouco os arquivos **css Interno** e **css Inline** conseguimos demonstrar as diferenças.

A seguir o mesmo arquivo **css Inline** com uma tag `<h1>` a mais (`<h1>CSS - Cascading Style Sheets</h1>`).

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>CSS</title>
<style>
h1 {
color:blue;
}
</style>
</head>
<body>
<h1>AprendendoCSS!</h1>
<h1>CSS - Cascading Style Sheets</h1>
</body>
</html>
```

É possível observar no resultado na Imagem 09, que quanto utilizamos o **CSS inline**, a alteração da propriedade acontece apenas na tag onde foi utilizado o código **CSS**, por outro lado, quando utilizamos o **CSS Interno** todas as tags `<h1>` da página recebem a definição de cor do texto como sendo **azul**.



Inline

Interno

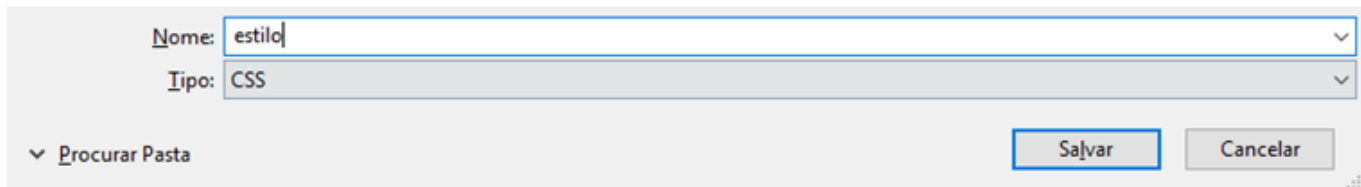
Observação: Você pode usar o **CSS Inline**, **Interno** e **Externo** no mesmo documento, ao mesmo tempo, segundo a necessidade da sua página.

Externo

Crie um arquivo **HTML** e salve com o nome de “**cssExterno**”, não se esquecendo da estrutura básica.

Como o próprio nome diz: “**externo**”, quer dizer que o código **CSS** será programado **fora** do arquivo html (**cssExterno**) recentemente criado, ou seja, um novo arquivo que será codificado apenas com **CSS**.

Crie um arquivo com o nome de “**estilo**” e escolha o tipo CSS para que não seja salvo com a extensão “.html”, como pode ser visto na Imagem 10.



Pronto, agora temos dois arquivos! Vamos codificar o arquivo “**estilo**” com os mesmos códigos **CSS** dos exemplos anteriores. O Código deve ficar como demonstrado a seguir:

```
h1 {
  color:blue;
}
```

Agora precisamos fazer com que a página (**cssExterno.html**) utilize esse código CSS. Para isso, precisamos fazer um link entre os dois arquivos utilizando a tag **<link>** que fica localizada dentro da tag **<head>**.

A tag **<link>** deve ser codificada da seguinte forma:

```
<link rel="stylesheet" type="text/css" href="estilo.css">
```

Vamos à explicação: esta **tag** terá a função de ligar diversas informações à página, o atributo **rel** define que será uma folha de estilo do valor “**stylesheet**”, e o atributo **type**, como o próprio nome indica, define qual tipo por meio do valor “**text/css**”; por fim, o atributo **href** define a localização do arquivo CSS.

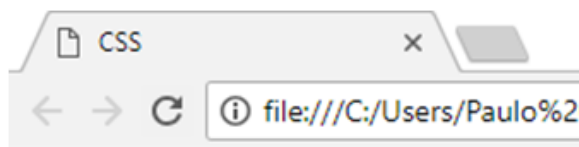
Como nesse caso salvamos ambos (**cssExterno.html** e **estilo.css**) na mesma pasta, somente há a necessidade de colocar o nome do arquivo. O código do arquivo “**cssExterno**” deve ficar como demonstrado a seguir:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>CSS</title>
</head>
<body>
<h1>Aprendendo CSS!</h1>
</body>
</html>
```

Observação: Repare que o nome do arquivo ficou sublinhado. Este é um outro recurso oferecido pelo Visual Code, por meio do **IntelliSense**; caso você pressione o **CTRL** e clique em cima do **estilo.css**, o arquivo será aberto no Visual Code em uma nova aba.

Observe que teremos o mesmo resultado dos outros exemplos como evidenciados na imagem a seguir.



Aprendendo CSS!



O resultado é exatamente o mesmo, então surge a pergunta:

Para que utilizar o código em um arquivo externo? A resposta é simples: para fins de organização e reuso.

Veja:

- **Organização:** ao utilizar um arquivo apenas para css, seu código ficará muito mais organizado e limpo, o que facilita e muito a manutenção do código e site.
- **Reuso:** talvez seja o mais importante, oferece a possibilidade de utilizar o mesmo arquivo css

Vamos exemplificar:

Crie outro arquivo HTML e salve com o nome de “**cssExemplo**” na mesma pasta dos arquivos **cssExterno.html** e **estilo.css**.

Não se esqueça de criar a estrutura básica. Programe a linha a seguir dentro da tag **<body>**:

```
<h1>Aprendendo Folha de Estilo Externa</h1>
```

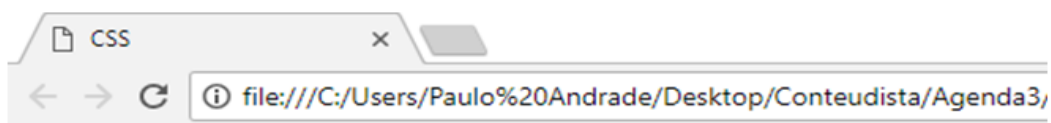
E dentro da tag **<head>** programe a tag **<link>** como no arquivo “**cssExterno**”:

```
<link rel="stylesheet" type="text/css" href="estilo.css">
```

O código do arquivo **cssExemplo** deve ficar como exibido a seguir:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<link rel="stylesheet" type="text/css" href="estilo.css">
<title>CSS</title>
</head>
<body>
<h1>Aprendendo Folha de Estilo Externa</h1>
</body>
</html>
```

O resultado será:



Aprendendo Folha de Estilo Externa

É fácil perceber que você pode reutilizar o código de estilo previamente programado em um arquivo, o que é muito útil para a produção de site com todas as páginas padronizadas, mas vai além disso, teste o seguinte:

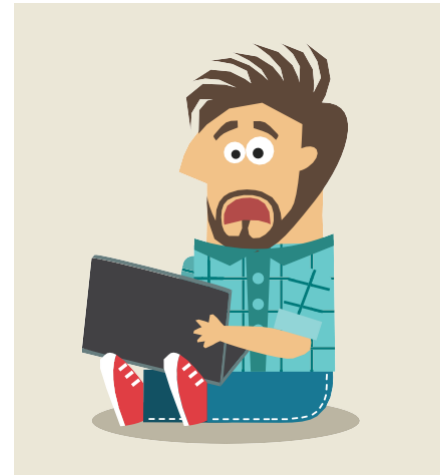
1. Deixe os arquivos **cssExterno.html** e **cssExemplo.html** abertos no navegador;
2. Edite o arquivo **estilo.css** trocando a cor de azul (blue) para vermelho(red);
3. Salve a alteração no arquivo **estilo.css**.
4. Volte ao navegador e atualize ambas as páginas (**cssExemplo** e **cssExterno**) pressionando a tecla **F5**.

Você deve ter notado que trocando a cor apenas no arquivo externo acontece um efeito cascata, alterando a cor de todas as páginas que contêm o arquivo **CSS** linkado nela.

Nesse exemplo, este recurso pode parecer simples; porém, pense em um portal completo com 300 arquivos **HTML**, todos seguindo o mesmo padrão de cor e, um certo dia, o seu cliente fala que a empresa quer mudar a cor padrão site. E aí? Como você resolve isso?

Imaginando essa situação, calcule o trabalho que você teria para mudar a cor de todos os arquivos utilizando a forma: **Inline**, **Interno** e **Externo**.

O mais fácil, sem sombra de dúvida, seria o arquivo externo, porém todos os três têm suas vantagens e desvantagens, sempre utilize o método (inline, externo ou interno), de acordo com sua necessidade, mas sempre pensando em possibilidades futuras de manutenção de código!



Observação: Como mencionado anteriormente, é possível a utilização de todas as formas de **CSS** em uma mesma página, na verdade isso é muito comum.

Surge, então, uma questão:

- Qual estilo será usado quando houver mais de um estilo especificado para um elemento HTML?

A resposta é que no geral, todos os estilos vão “cascatear” em uma nova folha de estilo “virtual” seguindo as regras, na qual o número 1 tem a prioridade mais alta:

1. Estilo **Inline** (dentro de um elemento HTML)
2. Folhas de estilo **externas** e **internas** (ordem de declaração).
3. Padrão do navegador

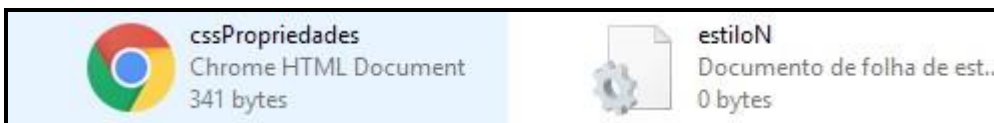
Logo, quando utilizado **CSS Inline**, ele é quem possui a prioridade mais alta, o que significa que substituirá um estilo definido dentro da tag **<head>** ou em uma folha de estilos externa ou ainda um valor padrão do navegador.

Outras Propriedades

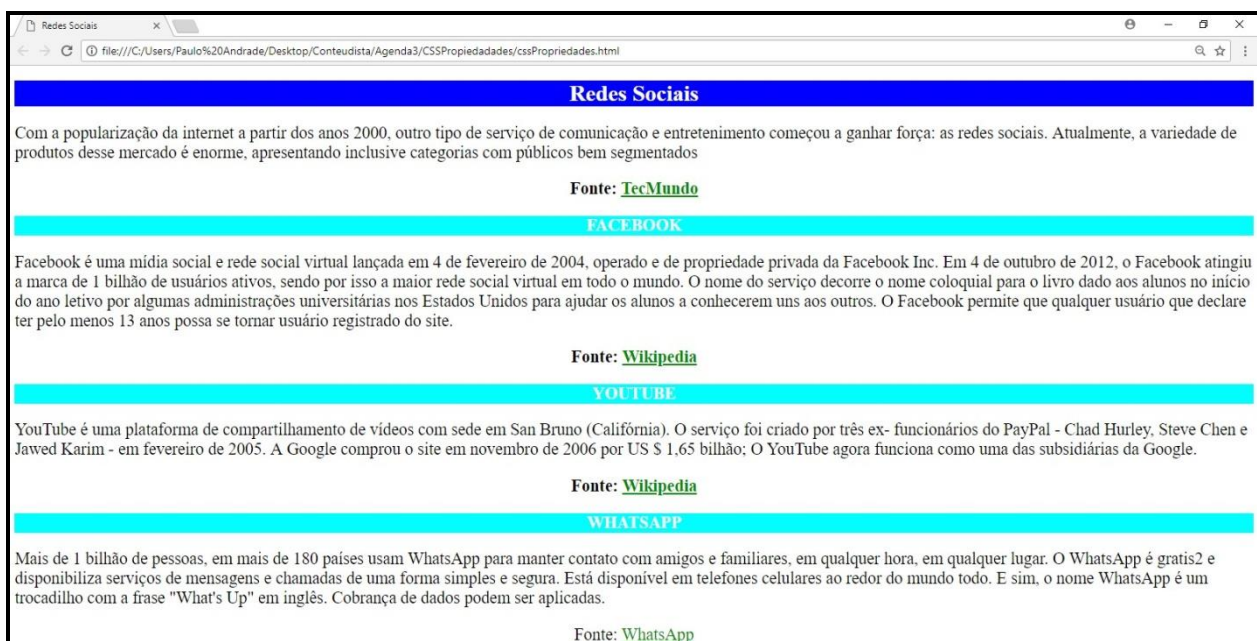
Até agora, para a apresentação dos conceitos, utilizamos basicamente a propriedade cor. Vamos mudar isso e conhecer algumas novas propriedades programando.

Para isso, crie um arquivo **HTML** e outro **CSS**. Salve com os respectivos nomes: “**cssPropriedades**” e “**estiloN**”, não se esquecendo de criar a estrutura básica no arquivo HTML e salvando ambos os arquivos na mesma pasta.

Observe a imagem a seguir:



Nós utilizaremos alguns seletores (**body**, **h1**, **p**, **h2**, **a**) e a página ficará como demonstrado na imagem a seguir:



A partir da estrutura básica criada no documento **HTML** e entre a tag **body**, vamos digitar os códigos e as informações, conforme o exemplo a seguir:

```

<!-- Bloco 1 -->
<h1>Redes Sociais</h1>
<p>
    Com a popularização da internet a partir dos anos 2000,
    outro tipo de serviço de comunicação e entretenimento começou
    a ganhar força: as redes sociais. Atualmente, a variedade de
    produtos desse mercado é enorme, apresentando inclusive
    categorias com públicos bem segmentados
</p>
<p style="text-align:center; font-weight: bold;">
Fonte:
<a href="https://www.tecmundo.com.br/redes-sociais/33036-a-historia-das-redes-sociais-
como-tudo-comecou.htm">
TecMundo
</a>
</p>
<!--FaceBook -->
<h2>FACEBOOK</h2>
<p>
    Facebook é uma mídia social e rede social virtual lançada em 4
    de fevereiro de 2004, operado e de propriedade privada da Facebook
    Inc. Em 4 de outubro de 2012, o Facebook atingiu a marca de 1 bilhão
    de usuários ativos, sendo por isso a maior rede social virtual em
    todo o mundo. O nome do serviço decorre o nome coloquial para o
    livro dado aos alunos no início do ano letivo por algumas administrações
    universitárias nos Estados Unidos para ajudar os alunos a conhecerem
    uns aos outros. O Facebook permite que qualquer usuário que declare
    ter pelo menos 13 anos possa se tornar usuário registrado do site.
</p>
<p style="text-align:center; font-weight: bold;">
Fonte:
<a href="https://pt.wikipedia.org/wiki/Facebook">
Wikipedia
</a>
</p>
<!--Youtube -->
<h2>YOUTUBE</h2>
<p>
    YouTube é uma plataforma de compartilhamento de vídeos com sede em
    San Bruno (Califórnia). O serviço foi criado por três ex- funcionários
    do PayPal - Chad Hurley, Steve Chen e JawedKarim - em fevereiro de 2005.
    A Google comprou o site em novembro de 2006 por US $ 1,65 bilhão;
    O YouTube agora funciona como uma das subsidiárias da Google.
</p>
<p style="text-align: center; font-weight: bold;">
Fonte:
<a href="https://pt.wikipedia.org/wiki/YouTube">
Wikipedia
</a>
</p>
<!-- WhatsApp -->
<h2>WHATSAPP</h2>

```



```

<p>
Mais de 1 bilhão de pessoas, em mais de 180 países usam WhatsApp
para manter contato com amigos e familiares, em qualquer hora,
em qualquer lugar. O WhatsApp é gratis2 e disponibiliza serviços
de mensagens e chamadas de uma forma simples e segura. Está
disponível em telefones celulares ao redor do mundo todo.
E sim, o nome WhatsApp é um trocadilho com a frase "What'sUp"
em inglês. Cobrança de dados podem ser aplicadas.
</p>
<p style = "text-align: center;">
Fonte:
<a href="https://www.whatsapp.com/about/">
    WhatsApp
</a>
</p>

```

Repare que a codificação é pequena perto da quantidade de informações que a página tem. Essa codificação é apenas a parte de **HTML**, se deixarmos assim, o resultado seria como o demonstrado a seguir:



Agora vamos codificar o arquivo **estilo.css**, sem se esquecer de fazer o link no arquivo **cssPropriedades.html**

```
<link rel="stylesheet" type="text/css" href ="estiloN.css">
```

Além do uso apenas do seletor, em alguns casos existe a possibilidade de se usar mais configurações que podem ser realizadas. Nesse exemplo, os links podem ter um estilo diferente, dependendo do estado em que estão: visitado ou com o mouse em cima do mesmo, por exemplo, como pode ser observado a seguir:

```

/*Seletor de link*/
a:link {
color: green;
}
/*Seletor de link visitado*/
a:visited {
color: green;
}
/*Seletor de link quando o mouse passar por cima do link*/
a:hover {
color: blue;
}

```

Assim, o código completo do arquivo [estilo.css](#), deve ficar como o descrito a seguir:

```

body{
font-size: 100%;
}
h1{
text-align:center;
background-color: blue;
color: white;
}
p{
font-size: 1.5em;
color: black;
}
h2{
text-align:center;
background-color: cyan;
color: white;
}
/*Seletor de link*/
a:link {
color: green;
}
/*Seletor de link visitado*/
a:visited {
color: green;
}
/*Seletor de link quando o mouse passar por cima do link*/
a:hover {
color: blue;
}

```

O resultado é que a página agora possui a estilização programada no arquivo `estilo.css`, conforme imagem a seguir:



Observação: Note que no texto "**Fonte: WhatsApp**", última linha da imagem apresentada, não está em negrito e que a palavra fonte está em cinza, enquanto nos outros três a palavra fonte está na cor preta e em negrito. Erros assim acontecem com frequência ao utilizar **Css Inline** e podem ser evitados utilizando **classes**, tema do próximo conteúdo que vamos estudar!

ID e Classe

Em códigos **HTML** e **CSS** existe a possibilidade de se aplicar estilos por meio de '**class**' e '**id**', mas qual a diferença entre elas?

- **Classe** é forma de identificar um **grupo** de elementos (**tags**) que irão compartilhar a estilização por meio de css.
- **Id** é uma forma de identificar um elemento de forma **única**. É como se fosse o CPF, único para cada pessoa. Por meio do id , pode-se atribuir formatação a um elemento em especial.

Colocar um nome de **classe** ou **ID** em um elemento, por padrão, não gera nenhuma alteração a esse elemento, mas para que as **classes** e **IDs** tenham qualquer estilo é necessário codificá-las em **CSS** e então aplicar estilos.

Veja algumas observações:

- Elementos (tags) podem ter ambos, **ID** e **Classes**
- **IDs** têm uma funcionalidade especial nos navegadores.
- Em CSS não existe nada que você possa fazer com **ID** que não possa fazer com classe, mas cuidado! Se não seguir a regra de **ID** único na página, seu código pode não ser validado e você poderá ter problemas com algumas linguagens como o JavaScript, por exemplo.
- Para saber mais sobre validação de código acesse o link:
<https://jigsaw.w3.org/css-validator/about.html.pt-BR>

Certo! Agora que entendemos os conceitos, vamos testá-los ao observar o resultado exibido pelo navegador do arquivo [cssPropriedas.html](#).

Pelo navegador, é possível notar a diferença entre “Fonte: Wikipedia” e última “Fonte: WhatsApp” (imagem ao lado), porém, podemos melhorar o código utilizando o conceito de **classe**, assim ao desenvolver as páginas, erros como esse serão muito raros.

A alteração é bem simples! Vamos criar uma classe chamada **fonte**. Nela, vamos fazer a configuração que, no momento, encontra-se codificada no modelo **CCS** em linha no arquivo [ccsPropriedades.html](#).



```
<p style = "text-align: center; font-weight: bold;">
```

Para isso, basta codificar no arquivo [estiloN.css](#) na classe fonte, utilizando o “.”, seguido do nome que escolhemos para a classe que estamos desenvolvendo que, no caso, será **fonte**.

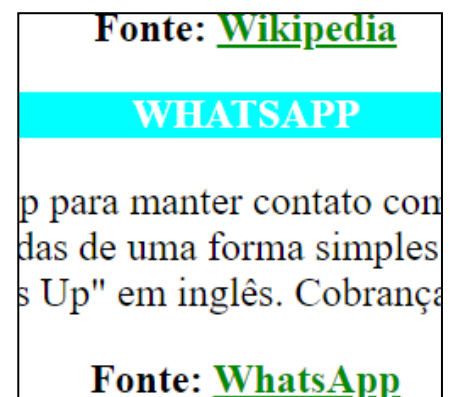
```
.fonte{
  text-align: center;
font-weight: bold;
}
```

Em um arquivo do tipo **css** você sempre observará no código um seletor iniciando com ponto. Trata-se de uma classe desenvolvida para unir um conjunto de estilização para serem utilizadas em determinadas **tags** que, no caso, serão os parágrafos da tag **<p>** que demarcam as fontes das informações da página web.

Mas, agora, precisamos também fazer as alterações no código fonte do arquivo **HTML**, para ser mais preciso, nas tag **<p>** que demarcam as fontes de informações. Com a codificação utilizando a classe, todas as tags **<p>** devem ficar como demonstradas como o exemplo a seguir:

```
<p class="fonte">
```

Note que não é mais necessário o uso do atributo **style**. Utilizamos o atributo **class**, e colocamos o nome da classe que acabamos de desenvolver. Após realizar todas as alterações e salvar, o resultado deve ser como o demonstrado na imagem ao lado.



E o mais interessante é que:

Imagine que desejamos alterar o tamanho da fonte, deixando-a menor do que as informações acima dela. Para isso, basta uma pequena alteração no código da classe e conseguimos que todas as fontes de informações tenham o seu tamanho alterado.

Antes de começar, vamos entender um pouco do código que fizemos, e que ainda não foi abordado. Reparem que a propriedade **font-size** é o código que define e controla o tamanho do texto. O controle e a capacidade de gerenciar o tamanho do texto é muito importante para o desenvolvimento da página. Porém, é necessário ficar atento a alguns detalhes: o valor do tamanho da fonte pode ser um tamanho absoluto ou relativo.

Tamanho absoluto:

- Define o texto para um tamanho especificado.
- Não permite, em nenhum dispositivo, que seja alterado o tamanho do texto nos navegadores, o que para fins de acessibilidade é muito ruim.

Tamanho relativo:

- Define o tamanho em relação aos elementos circundantes.
- Permite que seja alterado o tamanho do texto por navegadores e/ou dispositivos.

Esse conceito também pode ser aplicado a diversos elementos de uma página, não apenas a textos. Foi utilizada a unidade de tamanho que é recomendada pelo **W3C** denominada “**1em**”, que é igual ao tamanho da fonte atual.

Por exemplo, o tamanho do texto padrão nos navegadores é 16px. Quando utilizamos a configuração de **1em** estamos falando que o tamanho da fonte é de **16px**.

O tamanho pode ser calculado em pixels usando a fórmula: pixels / 16 = em.

Porém, se alterarmos o valor padrão para **10px**, então **1em** terá o valor de **10px** e, se utilizarmos **1.5em**, estaríamos dizendo que a fonte teria **15px**.

Vamos ao código!

O seletor **body** está com a propriedade **font-size** definida como **100%**. Como não alteramos o tamanho padrão da fonte, todo o texto, por **padrão, terá o tamanho de 16px**. Porém, na classe, definimos que a propriedade **font-size** terá 0.8em, ou seja, ela será menor que o tamanho padrão do texto. Utilizando a fórmula apresentada acima teremos o valor de **12.8px**.



```
.fonte{
text-align:center; font-weight: bold;
font-size: 0.8em;
}
body{
font-size: 100%;
}
```

O resultado dessa alteração pode ser vista na imagem logo a seguir, onde se pode perceber que o tamanho da fonte foi reduzido.

Redes Sociais
Com a popularização da internet a partir dos anos 2000, outro tipo de serviço de comunicação e entretenimento começou a ganhar força: as redes sociais. Atualmente, a variedade de produtos desse mercado é enorme, apresentando inclusive categorias com públicos bem segmentados
Fonte: TecMundo
FACEBOOK
Facebook é uma mídia social e rede social virtual lançada em 4 de fevereiro de 2004, operado e de propriedade privada da Facebook Inc. Em 4 de outubro de 2012, o Facebook atingiu a marca de 1 bilhão de usuários ativos, sendo por isso a maior rede social virtual em todo o mundo. O nome do serviço decorre o nome coloquial para o livro dado aos alunos no início do ano letivo por algumas administrações universitárias nos Estados Unidos para ajudar os alunos a conhecerem uns aos outros. O Facebook permite que qualquer usuário que declare ter pelo menos 13 anos possa se tornar usuário registrado do site.
Fonte: Wikipedia
YOUTUBE
YouTube é uma plataforma de compartilhamento de vídeos com sede em San Bruno (Califórnia). O serviço foi criado por três ex- funcionários do PayPal - Chad Hurley, Steve Chen e Jawed Karim - em fevereiro de 2005. A Google comprou o site em novembro de 2006 por US \$ 1,65 bilhão; O YouTube agora funciona como uma das subsidiárias da Google.
Fonte: Wikipedia
WHATSAPP
Mais de 1 bilhão de pessoas, em mais de 180 países usam WhatsApp para manter contato com amigos e familiares, em qualquer hora, em qualquer lugar. O WhatsApp é grátis ² e disponibiliza serviços de mensagens e chamadas de uma forma simples e segura. Está disponível em telefones celulares ao redor do mundo todo. E sim, o nome WhatsApp é um trocadilho com a frase "What's Up" em inglês. Cobrança de dados podem ser aplicadas.
Fonte: WhatsApp

Agrupamento de Seletores

Imagine uma situação em que você tenha vários **seletores** (elementos) com as mesmas declarações de estilo.

Por exemplo: as tag `<h1>`, `<h2>` e `<p>` estão com a cor de texto definidas como verde.

```
h1 {
color:green;
}

h2 {
color:green;
}

p {
color:green;
}
```


Existe uma maneira para agrupar os seletores para **minimizar o código** e **facilitar alterações e manutenção**. Para agrupar seletores, basta separá-los com uma vírgula. No exemplo a seguir, agrupamos os seletores do código e o resultado foi o mesmo. Veja:

```
h1, h2, p {  
color:green;  
}
```

Comentários

Assim como no **HTML**, em **CSS** também existem notas, informações ou observações que podem ser incluídas no código fonte. Os comentários utilizados não modificam, de maneira alguma, a estilização da página, apenas auxiliam o programador na organização dos seus códigos **CSS**. Observe alguns comentários adicionados no arquivo **estilo.css**:

```
/* Comentário em linha */  
/* Este é um  
comentário utilizando duas ou mais linhas */
```

No código acima, perceba que há uma barra “/” seguida de um asterisco “*”, demarcando o início e um asterisco “*”, seguido de outra barra “/” delimitando o final do comentário.

Desta forma, você pode inserir notificações e lembretes em seu código **CSS**.

```
p {  
color: red;  
/* Comentário em linha */  
}
```

Lembrando, ainda, que outra ótima função para os comentários é a verificação de código, porque ao comentar linhas de seu código, os comentários deixam de ser interpretados e ficam fora da página.

Do mesmo jeito que em **HTML**, repare que o Visual Code deixa todo comentário na cor verde para facilitar a sua identificação.

```
p {  
color: red;  
}  
  
/*  
h1{  
color:green;  
}  
*/
```

CSS Reset

Ao desenvolver páginas web, com certeza, iremos nos deparar com problemas como, por exemplo:

- ✎ a forma com que um mesmo código se comporta ou
- ✎ se será exibido em browsers (navegadores) diferentes, podendo ser difícil obter o mesmo resultado para a página, sendo, portanto, necessário escrever vários arquivos CSS, na tentativa de identificar qual browser está sendo utilizado e para aplicar a melhor formatação.

Para contornar esse problema, foi desenvolvida uma técnica chamada “**Reset CSS**”, que consiste em criar uma folha de estilo, que tenha como objetivo remover toda formatação padrão aplicada pelo navegador. Trata-se de uma folha de estilos (**arquivo CSS**).

Esse código CSS geralmente utiliza a opção **Externa** para ser utilizada em todas as páginas do site. No código encontram-se configurações em CSS, mas para evitar erros e para que o código realize sua função corretamente, o arquivo CSS deve ser inserido antes de qualquer arquivo para que toda a estilização adicional seja feita com base nos elementos “sem estilo”.

Procure sempre utilizar essa técnica no início do desenvolvimento dos projetos para evitar erros de formatação e estilo em um site já codificado e estruturado.

Você pode desenvolver seu próprio arquivo CSS para realizar o reset nas configurações, porém, na internet existem vários disponíveis de uso livre e gratuito. Um dos mais conhecidos e utilizados foi criado por Eric Meyer e está disponível no link a seguir: <https://meyerweb.com/eric/tools/css/reset/>

Basta copiar o código do site e colar em um arquivo do tipo **CSS** e, então, fazer o link com seu arquivo **HTML**.

```
/* http://meyerweb.com/eric/tools/css/reset/
v2.0 | 20110126
   License: none (public domain)
*/

html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary,
time, mark, audio, video {
    margin: 0;
    padding: 0;
    border: 0;
    font-size: 100%;
    font: inherit;
    vertical-align: baseline;
}
/* HTML5 display-role reset for older browsers */
article, aside, details, figcaption, figure,
footer, header, hgroup, menu, nav, section {
    display: block;
}
body {
    line-height: 1;
}
ol, ul {
    list-style: none;
}
blockquote, q {
    quotes: none;
}
blockquote:before, blockquote:after,
q:before, q:after {
    content: '';
    content: none;
}
table {
    border-collapse: collapse;
    border-spacing: 0;
}
```

Vamos testar!

Crie dois arquivos HTML e os salve com os nomes “**comCSSReset**” e “**semCSSReset**”.

Esses arquivos serão exatamente iguais com o código a seguir:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>Testando CSS reset</title>
</head>
<body>
<h1>
  Teste 1
</h1>
<h2>
  Teste 1
</h2>
<h3>
  Teste 1
</h3>
<h4>
  Teste 1
</h4>
<h5>
  Teste 1
</h5>
</body>
</html>
```

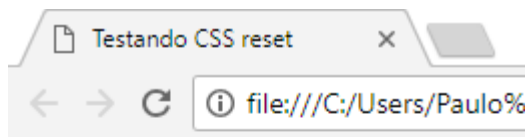
Agora adicione um código a mais no arquivo “**comCSSReset**”:

```
<link rel="stylesheet" type="text/css" href="reset.css">
```

Em seguida, crie um arquivo **CSS** e o salve na mesma pasta dos arquivos **HTML** com o nome de “reset”. Agora basta copiar e colar o código disponibilizado por Eric Meyer.

Depois de concluído, vamos abrir primeiro o arquivo “**semCSSReset**”.

A formatação deve ficar como na imagem a seguir:



Teste 1

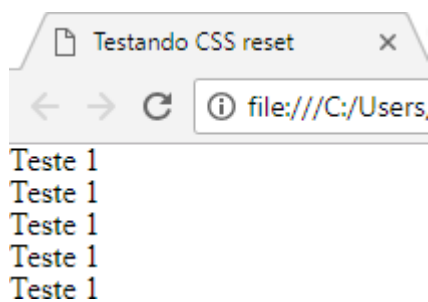
Teste 1

Teste 1

Teste 1

Teste 1

Agora vamos abrir o arquivo “**comCSSReset**”, para perceber como ficou a formatação:



Assim, podemos perceber que toda aquela configuração padrão do navegador para as tag `<h1>`, `<h2>` etc foram removidas, ou seja, será um problema a menos com relação ao desenvolvimento de uma página quando visualizada em diferentes navegadores.

Div's, FlexBox

Vamos à tag `<div>`.

Como o nome dessa tag sugere, é responsável por **dividir** qualquer trecho do código. Essa tag permite criar um bloco - uma divisão e, dentro deste bloco, é possível ter uma imagem, links ou textos.

Mas, por que utilizá-la? Com o auxílio do **CSS**, essa divisão e tudo o que estiver marcado pela tag `<div>`, receberá as configurações de estilo do **CSS**.

Veja um exemplo para utilização de **div**:



Imagine você precise dividir uma página com uma região para inserir o título e uma imagem, outra região para inserir o conteúdo textual e, por fim, uma outra região para as referências bibliográficas.

*Por meio desse exemplo, já é possível perceber que a tag **div** é muito útil para fazer divisões na página e muito importante para definir layouts de páginas.*

Para facilitar as divisões de layout, existe uma propriedade no CSS: o **CSS Flexible Box Layout Model** ou simplesmente **Flexbox**. Sua função é organizar elementos na página, principalmente quando o **layout/design** da página precisa ser visualizado em diversos tamanhos de tela e dispositivos.

O **Flexbox** ajuda a organizar esses elementos, além de sanar diversos problemas que acontecem quando acrescentamos **padding**, **margin** e **border** em alguns elementos.



Vamos agora testar na prática!

Crie um arquivo **HTML** e outro **CSS** e salve com os respectivos nomes de “**div**” e “**estilo**”, não se esquecendo de criar a estrutura básica no arquivo **HTML** e salvar ambos os arquivos na mesma pasta. Em seguida, divida a página em três colunas: esquerda, centro e direita, colorindo o fundo de cada uma com cores diferentes. Para isso codifique desta forma:

```

<div class="container">
  <div class="esquerda">
    Esquerda
  </div>

  <div class="centro">
    Centro
  </div>

  <div class="direita">
    Direita
  </div>
</div>

```

O código completo deve ficar como representado a seguir:

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<link rel="stylesheet" type="text/css" href="estilo.css">
<title>DIVS</title>
</head>
<body>
<div class="container">
<div class="esquerda">
Esquerda
</div>

<div class="centro">
  Centro
</div>

<div class="direita">
  Direita
</div>
</div>
</body>
</html>

```

Repare que para cada **div**, em seu atributo **class**, existe um valor. Essas classes serão programadas no arquivo a seguir, mas é possível notar a criação de quatro divs: uma principal que abriga todas as outras e três do mesmo nível, que serão as colunas: **esquerda**, **centro** e **direita**. Não se esqueça de salvar!

Agora vamos codificar o arquivo **estilo.css**:

```
.container {
width: 100%;
display: flex;
flex-direction: row;
}
.esquerda{

flex-grow: 1;
background:#ff0000;
font-size: 1.5em;
text-align:center;

}
.centro{

flex-grow: 2;
background: #00ff00;
font-size: 1.5em;
text-align:center;

}
.direita
{

flex-grow: 1;
background:#0000ff;
font-size: 1.5em;
text-align:center;

}
```

Vamos entender um pouco desse código:

Classe container:

```
.container {
width: 100%;
display: flex;
flex-direction: row;
}
```

A propriedade **width** determina a largura da **div** principal. Perceba que trabalhamos com o valor **100%** para que, independente do tamanho da tela do dispositivo que for utilizado para a visualização da página, todo o espaço será ocupado.

A próxima propriedade **display**, com o valor **flex**, indica que essa classe se tornará flexível utilizando os recursos disponíveis do **flexbox**. Logo após, utilizando a propriedade **flex-direction** indicamos como os elementos se organizarão e, finalmente, ao aplicar o valor **row**, estes elementos serão organizados de forma horizontal.

As classes: **esquerda**, **centro** e **direita** compartilham a codificação das mesmas propriedades, mas com valores diferentes: A propriedade **flex-grow** define o quanto um item ocupa do espaço total do contêiner em que está alojado em relação ao restante dos itens.

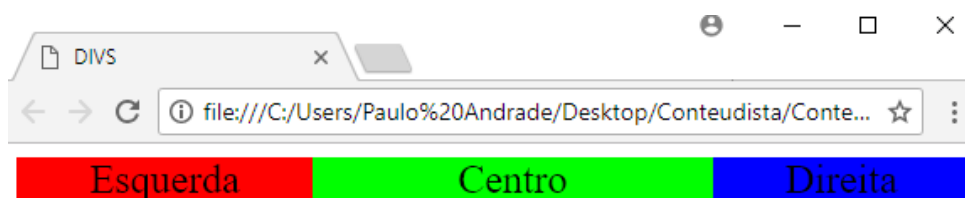
Vamos exemplificar: observe que às classes **direita** e **esquerda** foi atribuído o valor “1” e à classe **centro**, o valor “2”, se somarmos os valores atribuídos, o resultado será “4”, então a propriedade configurada dessa maneira define que a classe do centro ocupará dois quartos do espaço. Assim, a **div** configurada com a classe **container** e as classes **esquerda** e **direita** ficarão com um quarto cada.

A propriedade **background** está definindo a cor de fundo e a propriedade **font-size** define o tamanho da fonte. A propriedade **text-align**, por sua vez, determina o alinhamento do texto.

Salve as alterações no arquivo **estilo.css** e observe que ao abrir o arquivo **div.html**, o resultado apresentado no navegador deve ser como o exibido na imagem a seguir:



Agora faça um teste: Vá diminuindo a página no navegador e repare que as colunas irão se ajustando proporcionalmente ao novo tamanho, conforme a imagem a seguir:



Observação: Na codificação do arquivo **estilo.css**, pode-se notar que as propriedades **font-size** e **text-align** são exatamente iguais nas três classes (**esquerda**, **centro** e **direita**). Lembre-se que podemos agrupar seletores que dividem as mesmas propriedades!

O código ficaria como demonstrado a seguir e o resultado no navegador seria exatamente o mesmo!

```
.container {
width: 100%;
display: flex;
flex-direction: row;
}
.esquerda, .centro, .direita
{
font-size: 1.5em;
text-align:center;
}
.esquerda{

flex-grow: 1;
background:#ff0000;

}
.centro{

flex-grow: 2;
background: #00ff00;
}
.direita
{

flex-grow: 1;
background:#0000ff;
}
```



Imagine que uma loja de vídeo game precisa desenvolver uma página para fazer propaganda de alguns jogos e, então, lhe fez as seguintes solicitações:

1- Crie uma página com os requisitos:

- Título (aba) da página: Vídeo GameShow.
- Corpo da página:
 - Título: “Vídeo GameShow”;
 - Deverá ser dividido em três colunas de jogos: Ação, Corrida e Esportes;

Em cada uma dessas colunas deverá ter de três a cinco imagens/fotos de jogos de cada determinado tipo.

Dicas:

- Colocar título da página entre as tags `<title>` e `</title>`;
 - Usar a tag `<h1>` para título de cada coluna;
 - Colocar a imagem dos jogos na mesma pasta do arquivo html ou criar uma pasta apenas para imagens;
 - Utilize classes para criar estilizações, não esquecendo, caso as classes tenham estilizações em comum, de utilizar agrupamento de seletores, para evitar códigos repetidos;
 - Link interessante para encontrar fotos de jogos: <https://www.theenemy.com.br/>.
- Obs:** Não se esqueça de colocar a referência das fotos.

Opcional: Se você quiser testar seus conhecimentos, tente colocar o trailer de algum jogo. A seguir, confira se você conseguiu resolver os desafios propostos!

Antes de exibir a resposta, lembre-se que vimos muitos recursos até aqui, logo a resolução mostrada a seguir é apenas uma de muitas possibilidades.

Código HTML:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" type="text/css" href="estilo.css">
  <title>Vídeo Game Show</title>
</head>
<body >
<h1>Vídeo Game Show</h1>
<div class="principal">
  <div class="coluna1">
    <h1>Esportes</h1>
    
    
    
  </div>
  <div class="coluna2">
    <h1>Ação</h1>
    
    
    
  </div>
</div>
```

```




</div>
<div class ="coluna3">
  <h1>Corrida</h1>
  
  
  
</div>
</div>
<h1>Fonte das imagens:<a href="https://www.theenemy.com.br/">
www.theenemy.com.br</a></h1>
</body>
</html>

```

Obs 1: Nesta resposta foi criada uma **div** principal para armazenar as três colunas (**Esportes, Ação e Corrida**), onde serão exibidas as respectivas fotos.

Obs 2: Foram criadas 4 **classes**, uma para cada **div**. Cada uma dessas classes está programada com uma estilização no arquivo **css**.

Obs 3: Note que o caminho das imagens é: "imagens/nomedaimagem.tipo". Isso quer dizer que todas imagens estão dentro de uma pasta denominada imagens.

	Imagens	17/05/2018 11:07	Pasta de arquivos	
	estilo	17/05/2018 11:18	Documento de fol...	1 KB
	index	17/05/2018 11:09	Chrome HTML Do...	2 KB

Código CSS:

```

body{
background-color: #1e1d20;
}
.principal {
width: 100%;
display: flex;
flex-direction: row;
}
.coluna1, .coluna2, .coluna3{

align-items: center;
margin: 1%;
}
.coluna1{
flex-grow: 1;
}

.coluna2{

```

```

flex-grow: 1;
}
.coluna3{
flex-grow: 1;
}
img{
width: 100%;
margin: 1.5%;
}
h1{
text-align:center;
color: rgb(6, 135, 209);
font-size: 4.5em;
}
a:link, a:visited{
color: #fff;
}

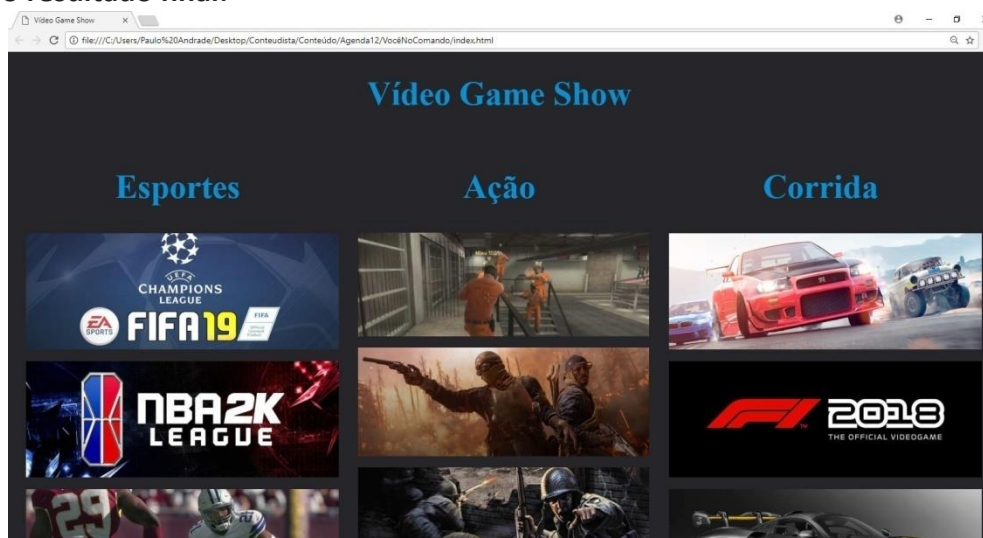
```

Obs 1: Neste código, a definição de cores está em hexadecimal ou em codificação RGB. Para entender melhor acesse este link: <https://www.infowester.com/coreshtml.php>

Obs 2: Note que as classes: coluna1, coluna2 e coluna3 têm algumas propriedades em comum, por esta razão, foi aplicado o agrupamento de seletores para não precisar repetir muito os códigos.

Obs 3: Para a página se adequar a qualquer aparelho e tamanho de telas, foi utilizada apenas porcentagem para definição de largura (**width: 100%;**) e margem (**margin: 1.5%; / margin: 1%;**).

Veja o resultado final:



Imagens: crédito Freepik.com