

# PROGRAMAÇÃO WEB

Prof. Ms. Wilson Lourenço

wilson.slourenco@sp.senac.br



# HTML SEMÂNTICO E POSICIONAMENTO NO CSS (PARTE 1)

# CSS Reset

- Quando não especificamos nenhum estilo para nossos elementos do HTML, o navegador utiliza uma série de estilos padrão, que são diferentes em cada um dos navegadores.
- Para evitar esse tipo de situação, alguns desenvolvedores e empresas criaram alguns estilos chamados de CSS Reset.

- A intenção é setar um valor básico para todas as características do CSS, sobrescrevendo totalmente os estilos padrão do navegador.
- Assim, podemos começar a estilizar as nossas páginas a partir de um ponto que é o mesmo para todos os casos, o que nos permite ter um resultado muito mais sólido em vários navegadores.
- Existem algumas opções para resetar os valores do CSS.

- HTML5 Boilerplate
  - projeto que pretende fornecer um excelente ponto de partida para quem pretende desenvolver um novo projeto com HTML5.
  - Uma série de técnicas para aumentar a compatibilidade da nova tecnologia com navegadores um pouco mais antigos estão presentes e o código é totalmente gratuito.
  - Em seu arquivo "style.css", estão reunidas diversas técnicas de CSS Reset.
  - Apesar de consistentes, algumas dessas técnicas são um pouco complexas, mas é um ponto de partida que podemos considerar.

- YUI3 CSS Reset
  - Criado pelos desenvolvedores front-end do Yahoo!, uma das referências na área, esse CSS Reset é composto de 3 arquivos distintos.
  - O primeiro, chamado de Reset, simplesmente muda todos os valores possíveis para um valor padrão, onde até mesmo as tags `<h1>` e `<small>` passam a ser exibidas com o mesmo tamanho.
  - O segundo arquivo é chamado de Base, onde algumas margens e dimensões dos elementos são padronizadas.
  - O terceiro é chamado de Font, onde o tamanho dos tipos é definido para que tenhamos um visual consistente inclusive em diversos dispositivos móveis.

- Eric Meyer CSS Reset
  - Há também o famoso CSS Reset de Eric Meyer, que pode ser obtido em <http://meyerweb.com/eric/tools/css/reset/>
  - É um arquivo com tamanho bem reduzido.

# Block X Inline

- Os elementos do HTML, quando renderizados no navegador, podem comportar-se basicamente de duas maneiras diferentes no que diz respeito à maneira como eles interferem no documento como um todo:
  - Em bloco (block)
  - Em linha (inline)



- Elementos em bloco são aqueles que ocupam toda a largura do documento, tanto antes quanto depois deles.
- Um bom exemplo de elemento em bloco é a tag `<h1>`.
- Note que não há nenhum outro elemento à esquerda ou à direita do nosso nome da loja, apesar da expressão "Mirror Fashion" não ocupar toda a largura do documento.

- Entre os elementos em bloco, podemos destacar as tags de heading `<h1>` a `<h6>`, os parágrafos `<p>` e divisões `<div>`.
- Elementos em linha são aqueles que ocupam somente o espaço necessário para que seu próprio conteúdo seja exibido, permitindo que outros elementos em linha possam ser renderizados logo na sequência, seja antes ou depois, exibindo diversos elementos nessa mesma linha.

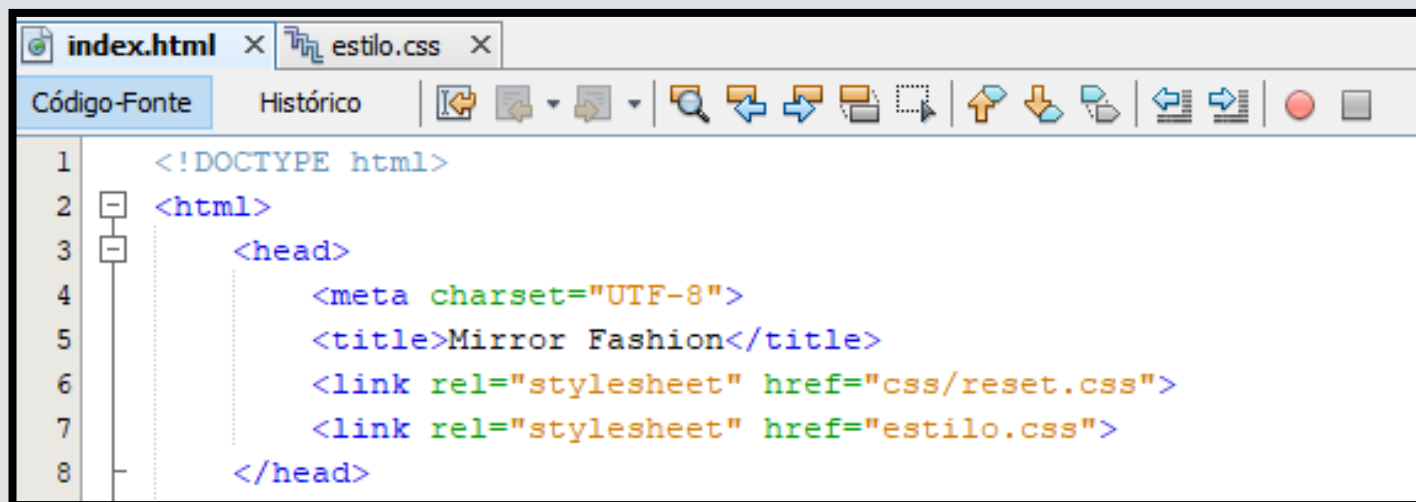
- Entre os elementos em linha, podemos destacar as tags de âncora <a>, as tags de ênfase <small>, <strong> e <em> e a tag de marcação de atributos <span>.
- Saber a distinção entre esses modos de exibição é importante, pois há diferenças na estilização dos elementos dependendo do seu tipo.
- Pode ser interessante alterarmos esse padrão de acordo com nossa necessidade, por isso existe a propriedade display no CSS, que permite definir qual estratégia de exibição o elemento utilizará.

- Por exemplo, o elemento `<li>` de uma `<ul>` tem por padrão o valor `block` para a propriedade `display`.
- Se quisermos os elementos na horizontal, basta alterarmos a propriedade `display` da `<li>` para `inline` :

```
ul li {  
    display: inline;  
}
```

## Vamos Colocar em Prática esses Conceitos de Reset e Display

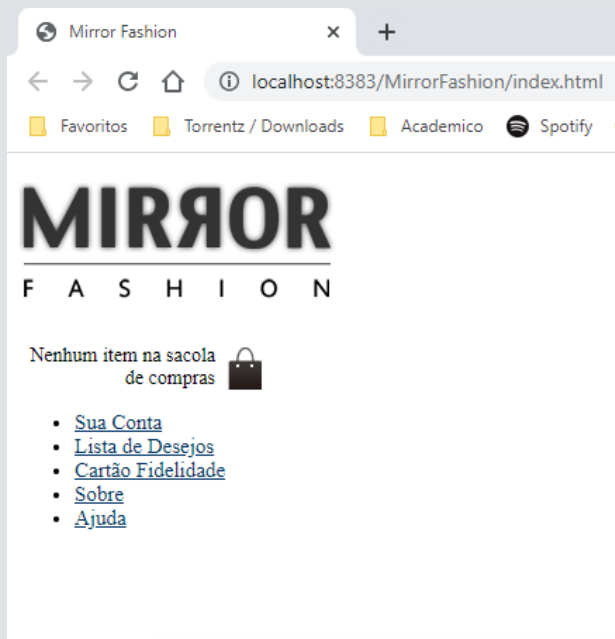
- Utilizaremos o CSS reset do Eric Meyer.
- Vou passar o arquivo reset.css para vocês. Coloquem esse arquivo na nossa pasta principal do projeto, com os demais arquivos.
- Agora, faça a referencia no head de index.html, **antes** de estilos.css



The image shows a web browser window with two tabs: 'index.html' and 'estilo.css'. The 'index.html' tab is active, displaying the source code in a 'Código-Fonte' (Source Code) view. The code is as follows:

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Mirror Fashion</title>
6     <link rel="stylesheet" href="css/reset.css">
7     <link rel="stylesheet" href="estilo.css">
8   </head>
```

## Antes



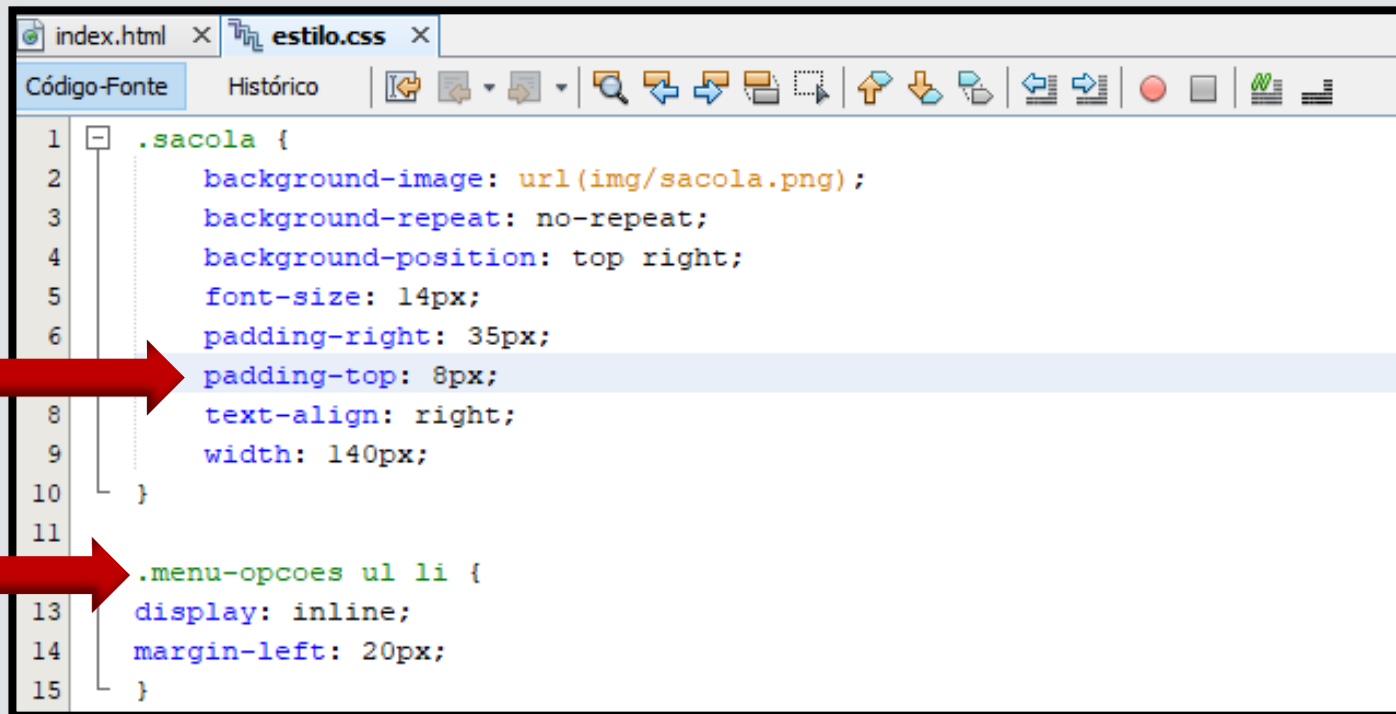
## Depois



Repare na diferença na padronização dos espaçamentos

- Agora precisamos transformar o menu em horizontal e ajustar espaçamentos básicos.
- Mexer no arquivo estilos.css
  - Vamos usar a propriedade display para mudar os <li> para inline
  - Aproveite e já coloque um espaçamento entre os links com margin em tamanho 20
  - É uma boa alinhar a imagem da sacola com o texto deixar como padding-top tamanho 8






The image shows a code editor with two tabs: 'index.html' and 'estilo.css'. The 'Código-Fonte' (Source Code) tab is active. The code is as follows:

```
1  .sacola {  
2      background-image: url(img/sacola.png);  
3      background-repeat: no-repeat;  
4      background-position: top right;  
5      font-size: 14px;  
6      padding-right: 35px;  
7      padding-top: 8px;  
8      text-align: right;  
9      width: 140px;  
10 }  
11  
12 .menu-opcoes ul li {  
13     display: inline;  
14     margin-left: 20px;  
15 }
```

Two red arrows are pointing to specific lines of code: one points to line 7 (`padding-top: 8px;`) and the other points to line 13 (`display: inline;`).

# MIRROR


F A S H I O N

Nenhum item na  
sacola de compras 

- [Sua Conta](#)
- [Lista de Desejos](#)
- [Cartão Fidelidade](#)
- [Sobre](#)
- [Ajuda](#)

# MIRROR

F A S H I O N

Nenhum item na  
sacola de compras 

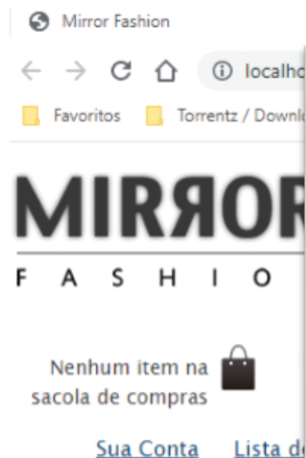
[Sua Conta](#) [Lista de Desejos](#) [Cartão Fidelidade](#) [Sobre](#) [Ajuda](#)

- O header ainda está todo à esquerda da página, sendo que no layout ele tem um tamanho fixo e fica centralizado na página.
  - Aliás, não é só o cabeçalho que fica assim: o conteúdo da página em si e o conteúdo do rodapé também.
- Temos três tipos de elementos que precisam ser centralizados no meio da página. Vamos criar uma classe no HTML a ser aplicada em todos esses pontos e um único seletor no CSS.

```
.container {  
    margin: 0 auto;  
    width: 940px;  
}
```

- Vamos usar essa classe container no HTML também.
- Altere a tag header e passe o class="container" para ela.

```
index.html x estilo.css
Código-Fonte Histórico
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <title>Mirror Fashion</title>
6     <link rel="stylesheet" href="css/reset.css">
7     <link rel="stylesheet" href="estilo.css">
8   </head>
9
10  <body>
11    <header class="container">
12      <h1></h1>
13      <p class="sacola">
14        Nenhum item na sacola de compras
15      </p>
16
17      <nav class="menu-opcoes">
18        <ul>
19          <li><a href="#">Sua Conta</a></li>
20          <li><a href="#">Lista de Desejos</a></li>
21          <li><a href="#">Cartão Fidelidade</a></li>
22          <li><a href="sobre.html">Sobre</a></li>
23          <li><a href="#">Ajuda</a></li>
24        </ul>
25      </nav>
26    </header>
27  </body>
28 </html>
```



## Position: Static, Relative, Absolute

- Existe um conjunto de propriedades que podemos utilizar para posicionar um elemento na página, que são top , left , bottom e right.
- Porém essas propriedades, por padrão, não são obedecidas por nenhum elemento, pois elas dependem de uma outra propriedade, a position .

- A propriedade `position` determina qual é o modo de posicionamento de um elemento, e ela pode receber como valor: `static`, `relative`, `absolute` ou `fixed`.
- O primeiro valor, padrão para todos os elementos, é o `static`.
  - Um elemento com posição `static` permanece sempre em seu local original no documento, aquele que o navegador entende como sendo sua posição de renderização.
  - Se passarmos valores para as propriedades de coordenadas, eles não serão respeitados.



- Outro valor para a propriedade position que aceitam coordenadas é o relative.
  - Com ele, as coordenadas que passamos são obedecidas em relação à posição original do elemento. Exemplo:

```
.logotipo {  
    position: relative;  
    top: 20px;  
    left: 50px;  
}
```

- Os elementos que receberem o valor "logotipo" em seu atributo class terão 20px adicionados ao seu topo e 50px adicionados à sua esquerda em relação à sua posição original.
- Note que, ao definirmos coordenadas, estamos adicionando pixels de distância naquela direção, então o elemento será renderizado mais abaixo e à direita em comparação à sua posição original.

- O próximo modo de posicionamento que temos é o absolute.
  - Existem algumas regras que alteram seu comportamento em determinadas circunstâncias.
  - Por definição, o elemento que tem o modo de posicionamento absolute toma como referência qualquer elemento que seja seu pai na estrutura do HTML cujo modo de posicionamento seja diferente de static (que é o padrão), e obedece às coordenadas de acordo com o tamanho total desse elemento pai.

- Quando não há nenhum elemento em toda a hierarquia daquele que recebe o posicionamento absolute que seja diferente de static, o elemento vai aplicar as coordenadas tendo como referência a porção visível da página no navegador. Por exemplo:

- Estrutura HTML

```
<div class="quadrado"></div>
```

```
<div class="quadrado absoluto"></div>
```

- Estilos CSS

```
.quadrado {  
    background-color: green;  
    height: 200px;  
    width: 200px;
```

```
}
```

```
.absoluto {  
    position: absolute;  
    top: 20px;  
    right: 30px;
```

```
}
```

- Seguindo o exemplo acima, o segundo elemento `<div>`, que recebe o valor "absoluto" em seu atributo class , não tem nenhum elemento como seu "pai" na hierarquia do documento, portanto ele vai alinhar-se ao topo e à direita do limite visível da página no navegador, adicionando respectivamente 20px e 30px nessas direções.
- Vamos analisar agora o exemplo a seguir:

- Estrutura HTML

```
<div class="quadrado relativo">  
    <div class="quadrado  
absoluto"></div>  
</div>
```

- Estilos CSS

```
.quadrado {  
    background-color: green;  
    height: 200px;  
    width: 200px;  
}  
.absoluto {  
    position: absolute;  
    top: 20px;  
    right: 30px;  
}  
.relativo {  
    position: relative;  
}
```

- Nesse caso, o elemento que recebe o posicionamento absolute é "filho" do elemento que recebe o posicionamento relative na estrutura do documento, portanto, o elemento absolute vai usar como ponto de referência para suas coordenadas o elemento relative e se posicionar 20px ao topo e 30px à direita da posição original desse elemento.



- O outro modo de posicionamento, `fixed`, sempre vai tomar como referência a porção visível do documento no navegador, e mantém essa posição inclusive quando há rolagem na tela.
- É uma propriedade útil para avisos importantes que devem ser visíveis com certeza.

## STATIC

- Sua posição é dada automaticamente pelo fluxo da página: por padrão ele é renderizado logo após seus irmãos
- Não aceita um posicionamento manual (left, right, top, bottom)
- O tamanho do seu elemento pai leva em conta o tamanho do elemento static

## RELATIVE

- Por padrão, o elemento será renderizado da mesma maneira que o static
- Aceita posicionamento manual
- O tamanho do seu elemento pai leva em conta o tamanho do elemento relative, porém sem levar em conta seu posicionamento. O pai não sofreria alterações mesmo se o elemento fosse static

# POSITION

## FIXED

- Uma configuração de posicionamento vertical (left ou right) e uma horizontal (top ou bottom) é obrigatória
- O elemento será renderizado na página na posição indicada. Mesmo que ocorra uma rolagem, o elemento permanecerá no mesmo lugar
- Seu tamanho não conta para calcular o tamanho do elemento pai, como se não fosse elemento filho

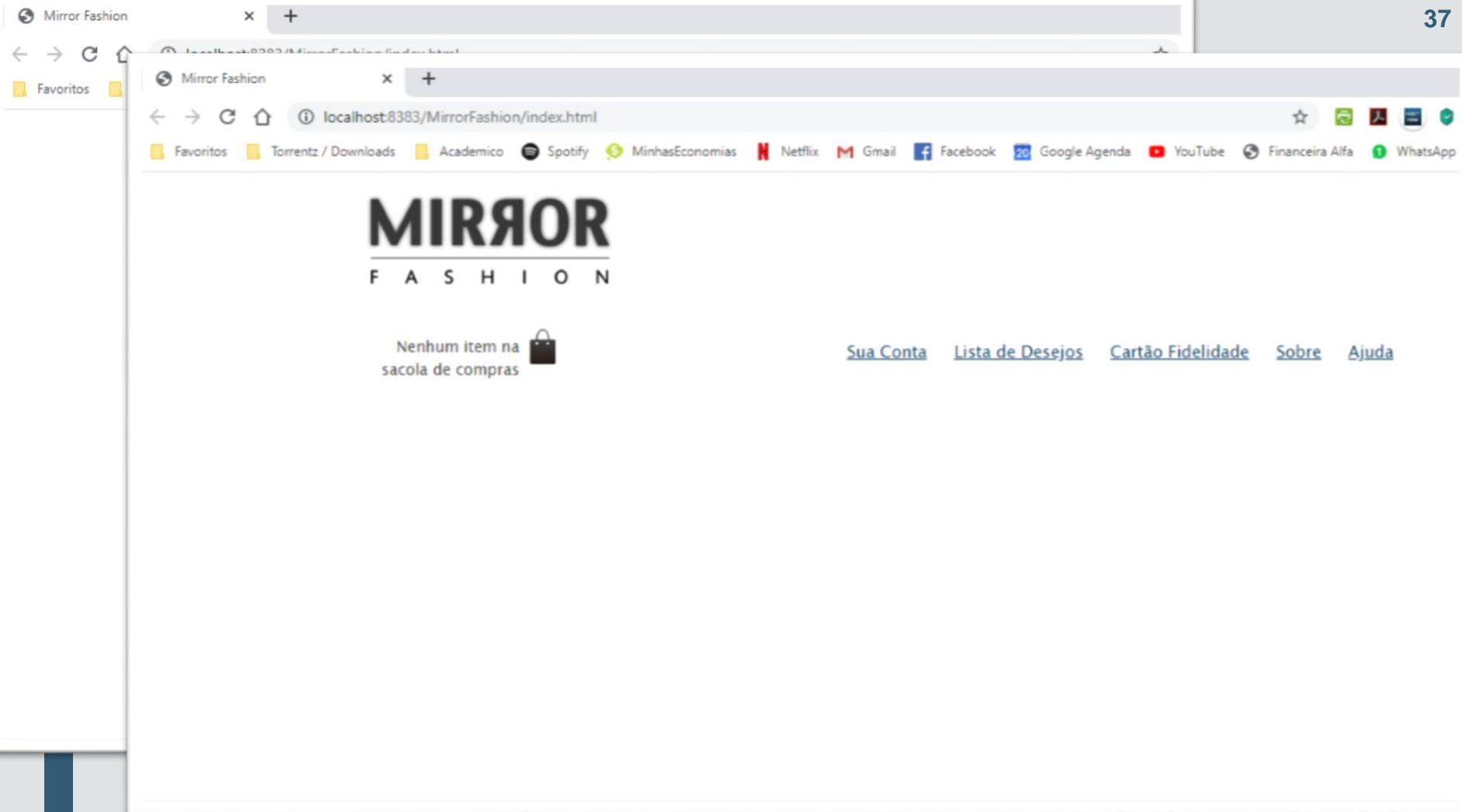
## ABSOLUTE

- Uma configuração de posicionamento vertical (left ou right) e uma horizontal (top ou bottom) é obrigatória
- O elemento será renderizado na posição indicada, porém relativa ao primeiro elemento pai cujo position seja diferente de static ou, se não existir este pai, relativa à página
- Seu tamanho não conta para calcular o tamanho do elemento pai

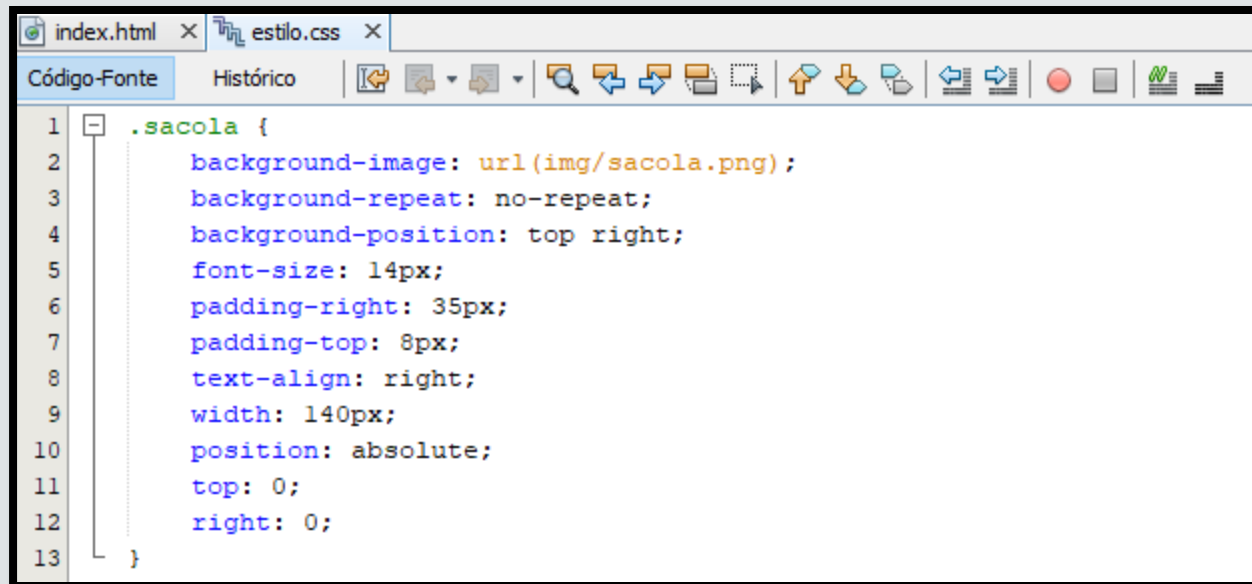
## Colocando Posicionamento em Prática

- Posicione o menu de opções à direita e embaixo no header.
  - Use position: absolute para isso.
  - Não esqueça: se queremos posicioná-lo absolutamente com relação ao cabeçalho, o cabeçalho precisa estar com position: relative

```
35  [-] header {  
36      |         position: relative;  
37      |     }  
38  
39  [-] .menu-opcoes {  
40      |         position: absolute;  
41      |         bottom: 0;  
42      |         right: 0;  
43      |     }
```

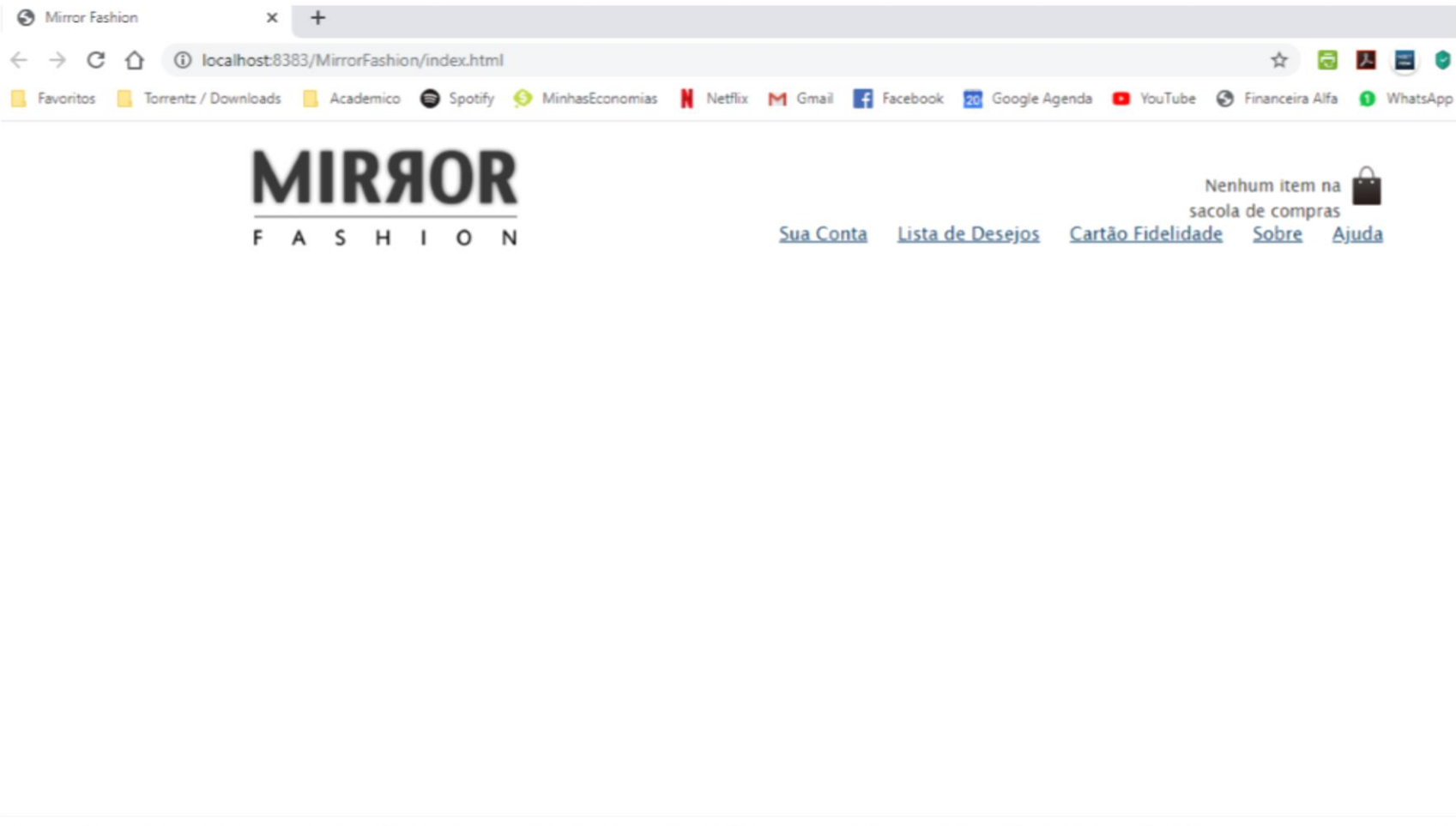


- A sacola também deve estar posicionada a direita e no topo.
  - Use position, top e right para conseguir esse comportamento.
  - Adicione as regras de posicionamento ao seletor .sacola que já tínhamos criado anteriormente



The image shows a web browser window with two tabs: 'index.html' and 'estilo.css'. The 'Código-Fonte' (Source Code) tab is active, displaying the CSS code for the '.sacola' class. The code is as follows:

```
1  .sacola {  
2      background-image: url(img/sacola.png);  
3      background-repeat: no-repeat;  
4      background-position: top right;  
5      font-size: 14px;  
6      padding-right: 35px;  
7      padding-top: 8px;  
8      text-align: right;  
9      width: 140px;  
10     position: absolute;  
11     top: 0;  
12     right: 0;  
13 }
```





# Dicas para Estudo



Seja “CURIOSO”:

Procure revisar o que foi estudado.

Pesquise as referências bibliográficas.



Seja “ANTENADO”:

Leia a próxima aula.



Seja  
“COLABORATIVO”:

Traga assuntos relevantes para a sala de aula.

Participe da aula.

Proponha discussões relevantes sobre o conteúdo.



Prof. Ms. Wilson Lourenço



**Dúvidas?  
Não mais..**