HTML e CSS

Front End Web

Msc. Lucas G. F. Alves
e-mail: lucas.g.f.alves@gmail.com

Senac

Planejamento de Aula

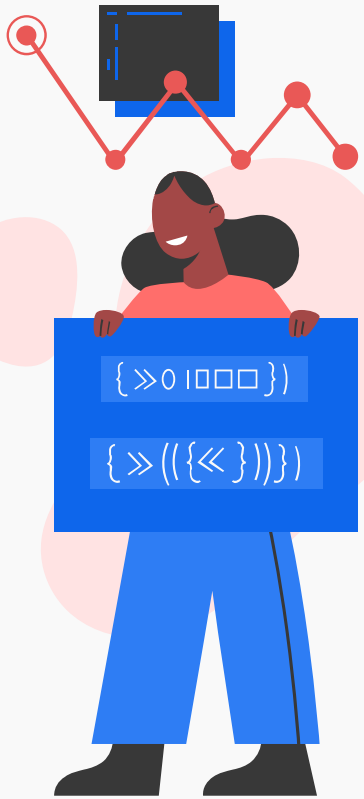
Revisão

Cascata e Herança

Display: Inline Block

Seletor de Atributo no CSS3

Exercícios



Revisão

HTML semântico

As tags que usamos antes - header, section e footer - são tags novas do HTML5.

Antigamente, seria criado apenas três div, uma para cada parte da página, e talvez colocando ids diferentes para cada uma.

A função do HTML é fazer a marcação do conteúdo da página, representar sua estrutura.

Já o CSS é cuidar da apresentação final e dos detalhes de design.

O HTML precisa ser claro e limpo, focado em marcar o conteúdo.

HTML semântico

Um HTML semântico carrega significado independente da sua apresentação.

Um usuário cego poderia usar um leitor de tela para ouvir sua página. Neste caso, a estrutura semântica do HTML é essencial para ele entender as partes do conteúdo.

É muito comum usar um <h1> com um texto que represente o título da nossa página.

Mas e pra colocar uma imagem de logo?
Quando o texto for lido para um cego, queremos essa mensagem lida. Quando o Google indexar, queremos que ele associe nossa página com o texto escrito e não com uma imagem.

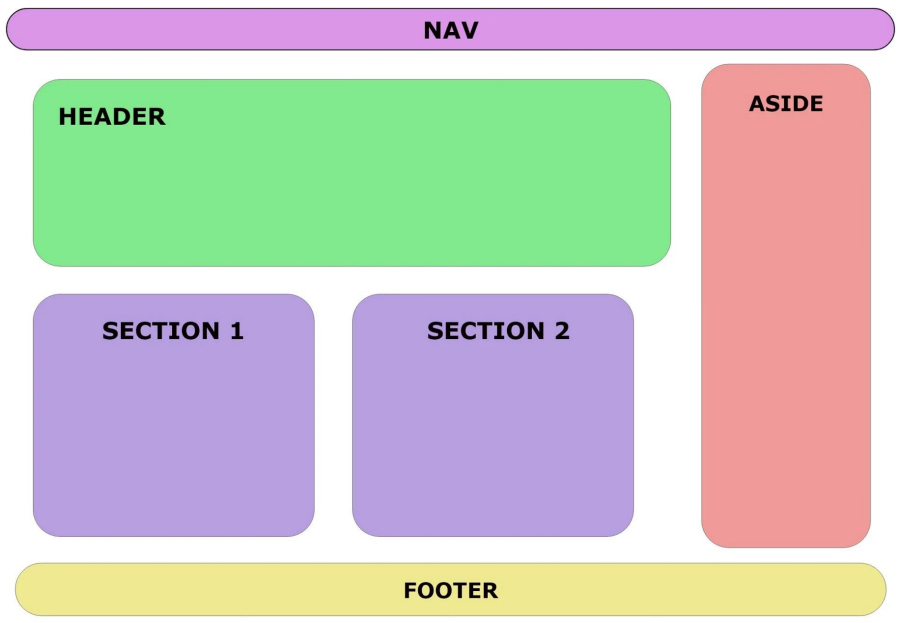
```
<h1><img src="img/logo.png" alt="site-tal"></h1>
```

Tags Semânticas

Tags semânticas são tags que possuem um significado, que dão sentido a informação de texto ao navegador e buscadores.

São elas:

<body>
    NAV
    HEADER
    MAIN
        SECTION
    FOOTER
    ASIDE
</body>



Estilização com classes

Para estilizar os elementos é necessário definir o CSS de cada coisa.

Já vimos **seletor de tag** e por **ID**. Ou seja, pra estilizar nosso menu <nav>, podíamos fazer:

nav { ... }

Mas imagine que podemos ter muitos NAV na página e queremos ser mais específicos. O ID é uma solução:

<nav id="menu-opcoes"> </nav>

E, no CSS:
#menu-opcoes { ... }

Estilização com classes

Classes.

O código é semelhante mas usa o atributo class no HTML e o ponto no CSS:

<nav class="menu-opcoes"> </nav>

E, no CSS:

.menu-opcoes { … }

Mas quando será usar ID ou CLASS?

Estilização com classes

Classes.

Ambos fariam seu trabalho nesse caso.

Mas IDs são mais fortes e devem ser únicos na página.

Embora o menu seja único agora, no futuro, o mesmo menu pode ser criado em outro ponto da página, mais pra baixo?

Usar classes facilita reuso de código e flexibilidade.

Estilização com classes

Um elemento pode ter mais de uma classe, aplicando estilos de várias regras do CSS:

```
<nav class="menu-opcoes menu-cabecalho"> ... </nav>
.menu-opcoes {
// código de um menu de opcoes
// essas regras serão aplicadas ao nav
}
.menu-cabecalho {
// código de um menu no cabeçalho
// essas regras TAMBÉM serão aplicadas ao nav
}
```

No caso do ID, não. Cada elemento só tem um id, único.

Estilização com classes

Utilizar classes é reaproveitar aquele elemento em mais de um ponto depois.

Vamos fazer isso no carrinho também:
 Nenhum item no carrinho de compras

Pode ser interessante criar uma classe que determina a centralização horizontal de qualquer elemento, sem interferir em suas margens superior e inferior, como no exemplo a seguir:

```
.container {
margin-right: auto;
margin-left: auto;
}
```

Estilização com classes

Agora, é só adicionar a class “container” ao elemento, mesmo que ele já tenha outros valores para esse atributo:

```
<div class="info container">

    <p>Conteúdo que deve ficar centralizado</p>

</div>
```

Exercícios

1) No arquivo index. Criar o cabeçalho utilizando as tags semânticas <header>, <nav>, <ul>, <li>, etc. Crie links para as páginas no menu. E use <h1> para representar o título da página com o logo acessível.

```
<header class="container">
    <h1><img src="img/logo.png" alt="nome_voces"></h1>
    <p class="carrinho"> Nenhum item no carrinho de compras </p>
    <nav class="menu-opcoes">
        <ul>
            <li><a href="conteudo.html">Conteúdo</a></li>
            <li><a href="blog.html">Blog</a></li>
            <li><a href="contato.html">Contato</a></li>
        </ul>
    </nav>
</header>
```

Exercícios

2) Ajuste as cores e alinhamento dos textos. Coloque o ícone da sacola com CSS através de uma imagem de fundo do parágrafo:

.carrinho { background: url(../img/carrinho.png) no-repeat top right; font-size: 14px; padding-right: 35px; text-align: right; width: 140px; }

.menu-opcoes ul { font-size: 15px; }

.menu-opcoes a { color: #003366; }

body { color: #333333; font-family: "Lucida Sans Unicode", "Lucida Grande", sans-serif; }

CSS Reset

O navegador utiliza uma série de estilos padrão, que são diferentes em cada um dos navegadores.

Para evitar problemas de quebra de layout por navegador alguns desenvolvedores e empresas criaram alguns estilos que chamamos de CSS Reset.

- Setar um valor básico para todas as características do CSS, sobrescrevendo totalmente os estilos padrão do navegador.

Começando sempre do mesmo ponto para todos os casos, o que nos permite ter um resultado muito mais sólido em vários navegadores.

CSS Reset

A opções para resetar os valores do CSS.

HTML5 Boilerplate

O HTML5 Boilerplate fornecer um ponto de partida para quem desenvolve um novo projeto com HTML5.

Tem uma série de técnicas para aumentar a compatibilidade da nova tecnologia com navegadores um pouco mais antigos 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.

CSS Reset

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 deles, chamado de Reset, 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, padroniza margens e dimensões dos elementos.

O terceiro é chamado de Font, onde o tamanho dos tipos é definido para que tenhamos um visual consistente inclusive em diversos dispositivos móveis.

CSS Reset

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 apenas um arquivo com tamanho bem reduzido.

Exercícios

1) Utilize o CSS reset do Eric Meyer, coloquem o arquivo reset.css para a pasta css do projeto, referencie no head antes do nosso estilos.css: <link rel="stylesheet" href="css/reset.css">Abra novamente a página no navegador. http://meyerweb.com/eric/tools/css/reset/.

2) Transformar o menu em horizontal e ajustar espaçamentos básicos.
Utilize a propriedade display para mudar os <li> para inline, coloque um espaçamento entre os links com margin, o texto ficará muito pra cima e não alinhado com a base do ícone. Aplique um padding-top.

```
.menu-opcoes ul li { display: inline; margin-left: 20px; }
.carrinho { padding-top: 8px; }
```

Exercícios

3) Para centralizar os elementos, crie uma classe container 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.

4) Altere a tag header e passe o class="container" para ela.

Position

Static.

Para posicionar um elemento na página existem as 4 propriedades, que são top, left, bottom e right. Porém essas propriedades dependem de uma outra propriedade, a position.

A propriedade position determina qual é o modo de posicionamento de um elemento.

Ela pode receber como valor static, relative, absolute ou fixed.

O primeiro valor, padrão, é o static.

Um elemento com posição static permanece sempre em seu local original, o navegador entende como sendo sua posição de renderização.

Position

Relative

Outro valor é o relative. Com ele, as coordenadas que passamos são obedecidas em relação à posição original do elemento. Por exemplo:

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

Será adicionado pixels de distância naquela direção, então o elemento será renderizado mais abaixo e à direita em comparação à sua posição original.

Position

Absolute

Outro valor é o absolute, por definição, o elemento que tem absolute pega como referência qualquer elemento que seja seu pai na estrutura do HTML, onde o valor seja diferente de static (que é o padrão), e obedece às coordenadas de acordo com o tamanho total desse elemento pai.

Quando não há esse elemento, vai aplicar as coordenadas tendo como referência a porção visível da página no navegador.

Position

Exemplo:

Estrutura HTML

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

Estilo CSS

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

Seguindo o exemplo acima, o segundo elemento , que recebe o valor “absoluto”, não tem nenhum elemento “pai”, portanto ele vai alinhar-se ao topo e à direita do limite visível da página, adicionando respectivamente 20px e 30px nessas direções.

Position

Exemplo 2:
Estrutura HTML

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

Estilos CSS

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

Aqui o elemento absolute é “filho” do elemento relative, 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.

Position

Fixed.

O outro modo de posicionamento, fixed, sempre vai tomar como referência a porção visível do documento no navegador.

Mantém essa posição inclusive quando há rolagem na tela.

É uma propriedade útil para avisos importantes que devem ser visíveis com certeza.

Precisa de uma configuração de posicionamento vertical (left ou right) e uma horizontal (top ou bottom).

Exercícios

1) Posicione o menu à direita e embaixo no header. Use position: absolute para isso. E não esqueça: se queremos posicioná-lo absolutamente com relação ao cabeçalho, o cabeçalho precisa estar posicionado.
header { position: relative; }
.menu-opcoes { position: absolute; bottom: 0; right: 0; }

2) O carrinho também deve estar posicionado a direita e no topo. Use position, top e right para conseguir esse comportamento. Adicione as regras de posicionamento ao seletor .carrinho que já tínhamos:
.carrinho { position: absolute; top: 0; right: 0; }

Cascata e herança no CSS

Cascata e herança no CSS

Algumas propriedades de elementos pais, quando alteradas, são aplicadas automaticamente para seus elementos filhos em cascata.
Exemplo:

```
<div id="pai">
        <h1>Sou um título</h1>
        <h2>Sou um subtítulo</h2>
</div>

#pai {
        color: blue;
}
```

No exemplo, todos os elementos filhos herdaram o valor da propriedade color do elemento pai a qual eles pertencem.

Cascata e herança no CSS

Existem propriedades que não são aplicadas em cascata, geralmente são as que afetam a caixa (box) do elemento, como width, height, margin e padding.

```
h1 { padding-left: 40px; }
#pai {
        color: blue;
        padding-left: 0;
}
<div id="pai">
        <h1>Sou um título</h1>
        <h2>Sou um subtítulo</h2>
</div>
```

Perceba que o padding do elemento <h1> não foi sobrescrito pelo valor do elemento pai , ou seja, o valor 40px foi mantido.

Cascata e herança no CSS

Inherit

O valor Inherit indica que o elemento filho terá o mesmo valor do elemento pai.

Toda vez que o tamanho do elemento pai for alterado, automaticamente o elemento filho herdará o novo valor, facilitando assim, a manutenção do código.

O valor inherit também afeta propriedades que não são aplicadas em cascata.

Cascata e herança no CSS

Inherit

Exemplo:

```
<div>
    <img src="box-model.png" alt="box model">
</div>
div {
    border: 2px solid;
    border-color: red;
    width: 30px;
    height: 30px;
}
```

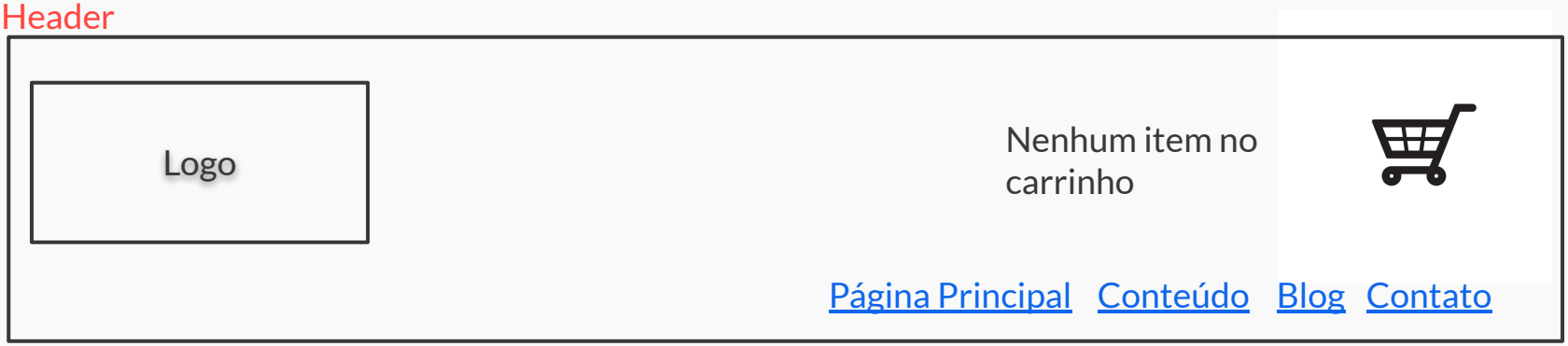
```
img {
    width: 30px;
    height: 30px;
}
```

```
img {
    width: inherit;
    height: inherit;
}
```

Exercício

Exercício – HTML

Header




```
<body>
    <header><!-- Conteúdo do cabeçalho --></header>
    <section id="main"><!-- Conteúdo principal --></section>
    <section id="destaques"><!-- Painéis com destaques --></section>
    <footer><!-- Conteúdo do rodapé --></footer>
</body>
```

Exercício – HTML

1) Criar um elemento destaque e, dentro dele, uma section para busca e outra para o menu, na página index.html.

```
<div class="container destaque">
    <section class="busca">
        <h2>Busca</h2> <form><input type="search"><input
        type="image" src="img/busca.png"></form
    </section> <!-- fim .busca -->
    <section class="menu-conteudo">
        <h2>Menu</h2>
        <nav>
            <ul>
                <li><a href="#">Aula1</a></li>
                <li><a href="#">Aula2</a></li>
            </ul> </nav> </section><!-- fim .menu-conteudo →
</div> <!-- fim .container .destaque →
```

Exercício – CSS

2) Aplicar cor de fundo as caixas de busca e no menu, o texto deve estar em negrito e apresentado em maiúsculas. Aplicar também algumas regras de tamanhos e margens.

```
.busca, .menu-conteúdo {
        background-color: #dcdcdc; font-weight: bold; text-transform: uppercase;
        margin-right: 10px;width: 230px; float: left;
}
.busca h2, .busca form, .menu-conteudo h2 {
        margin: 10px;
}
.menu-conteudo li {
        background-color: white; margin-bottom: 1px; padding: 5px 10px;
}
.menu-conteudo a {
        color: #333333; text-decoration: none;
}
.menu-conteudo { clear: left; }
```

Exercício - CSS

3) Na busca, use a propriedade vertical-align para alinhar o campo de texto à imagem da lupa pelo centro. Aproveite e coloque o tamanho do campo de texto para melhor encaixar no design e use seletores de atributo do CSS para isso.

```
.busca input {  vertical-align: middle;}
.busca input[type=search] { width: 170px;}
```

4) Acerte as margens e posicionamentos no menu lateral e no topo.

```
 .destaque { margin-top: 10px; }
.menu-conteudo { margin-top: 10px; padding-bottom: 10px; }
```

({(({ ≫}))≪ })

Display Inline-Block

Display Inline-Block

Vamos criar um painel com uma lista de pastas-aulas, onde cada aula será representado por uma <li>.

Por padrão uma <li> possui a propriedade display:block. Porém as pastas de cada aula serão lado a lado. Para isso defini-se a propriedade display como inline.

Também é necessário alterar as propriedades width, margin e padding das <li> , mas agora os elementos <li> são inline e este modo ignora alterações de propriedades da box.
Como resolver este problema?

Display Inline-Block

Vamos criar um painel com uma lista de pastas-aulas, onde cada aula será representado por uma <li>.

Por padrão uma <li> possui a propriedade display:block. Porém as pastas de cada aula serão lado a lado. Para isso defini-se a propriedade display como inline.

Também é necessário alterar as propriedades width, margin e padding das <li> , mas agora os elementos <li> são inline e este modo ignora alterações de propriedades da box.
Como resolver este problema?

Uma mistura dos dois, o inline-block.

Os elementos que recebem o valor inline-block aceitam as propriedades height (altura) e width (largura) e ainda seguem o inline.

Display Inline-Block

Exemplo: display: inline-block.

```
.painel li {
        display: inline-block;
        vertical-align: top;
        width: 140px;
        margin: 2px;
        padding-bottom: 10px;
}
```

Exercício

Exercício

1) Criar um elemento para conter os dois painéis. Ele deve receber a classe container, para se alinhar ao meio da tela, e a classe paineis que usaremos depois no CSS.

```
<div class="container paineis"> <!-- os paineis de pastas com aulas --> </div>
```

2) Dentro da div criada acima, crie uma nova <section> para cada painel. A primeira, receberá as classes **painel** e **aula1**. Cada exercicio deve ser representado como um item na lista (<li>) com um link para o exercício e sua imagem (representado por figure, figcaption e img).

```
<section class="painel aula 1">
        <h2>Aula 1</h2>
        <ol>
                <!-- primeiro exercicio -->
                <li> <a href="exercicio1..html">
                <figure><img src="img/aula1/miniatura-aula.png">
                <figcaption>Exercicio 1 HTML</figcaption>
                </figure> </a> </li>
                <!-- coloque mais exercicios aqui! -->
        </ol>
</section>
```

Exercício

3) Criar um segundo painel, para representar as segunda aula. Esse painel deve ficar após o fechamento do painel anterior, mas ainda dentro da div paineis.
O novo painel deve receber as classes painel e aula2. Sua estrutura é idêntica ao do exercício anterior (dica: copie o código para evitar refazer tudo de novo).

```
<section class="painel aula2">
        <h2>Aula 2</h2>
        <ol> <!-- coloque vários exercicios aqui --></ol>
</section>
```

Exercício

4) Posicionar os painéis no css.

```
.painel { margin: 10px 0; padding: 10px; width: 445px; }
.aula1 { float: left; background-color: #f5dcdc;  }
.aula2 { float: right; background-color: #dcdcf5;}
.painel li { display: inline-block; vertical-align: top; width: 140px; margin: 2px;
padding-bottom: 10px; }
.painel h2 { font-size: 24px; font-weight: bold; text-transform: uppercase; margin-bottom:
10px; }
.painel a { color: #333; font-size: 14px; text-align: center; text-decoration: none; }
```

Seletores de atributo do CSS3

Seletores de atributo do CSS3

Além dos seletores de tag, classe e id que vimos anteriormente, existe mais uma série de seletores avançados do CSS.

Um dos seletores avançados do CSS é o seletor de atributo, com ele é verificado a presença ou valor de um atributo para selecioná-lo. Por exemplo:

```
input[value] { color: #cc0000; }
```

O seletor acima age em todos os elementos da tag <input> que têm o atributo “value”.

Também é possível verificar se o atributo tem um valor específico:

```
input[type="text"] {  border-radius: 4px; }
```

Seletores de atributo do CSS3

Além de verificar um valor, é possível utilizar alguns operadores para selecionar valores em determinadas condições, como por exemplo o seletor de atributo com prefixo:

```
div[class|="menu"] { border-radius: 4px; }
```

O seletor acima vai agir em todas as tags cujo atributo “class” comece com a palavra menu seguida de um hífen e qualquer outro valor na sequência, como por exemplo menu-opções, menu-conteudo etc.
Também é possível buscar por uma palavra específica no valor, não importando o valor completo do atributo. Por exemplo:

```
input[value~="problema"] { color: #cc0000; }
```

Seletores de atributo do CSS3

Com o CSS3 é possível utilizar novos operadores com sinais que se assemelham aos das expressões regulares:

```
/* busca por inputs com valor de "name" iniciando em "usuario" */
input[name^="usuario"] { color: 99ffcc;}

/* busca por inputs com valor de "name" terminando em "teste" */
input[name$="teste"] { background-color: #ccff00; }

/* busca por inputs com valor do atributo "name" contendo "tela" em qualquer posição */
input[name*="tela"] { color: #666666; }
```

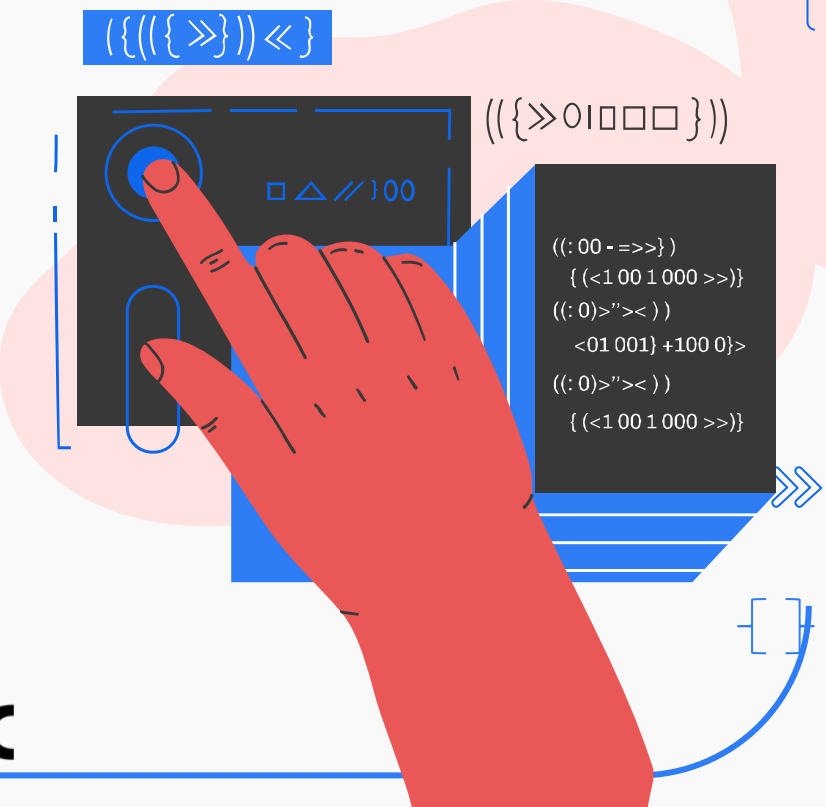
Professor

Lucas G. F. Alves

Senac

Obrigado!

E-mail :lucas.g.f.alves@gmail.com



Senac