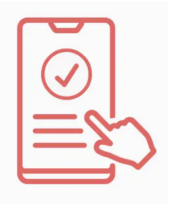
INSTITUTO FEDERAL
Santa Catarina

Curso Técnico em Desenvolvimento de Sistemas
Desenvolvimento para Dispositivos Móveis
Prof. Bruno Panerai Velloso
Prof. Glauco Cardozo

Mobile First



Mobile First é uma abordagem de design e desenvolvimento web que prioriza a criação de experiências otimizadas para dispositivos móveis antes de adaptar o design para telas maiores, como tablets e desktops. Essa metodologia surgiu em resposta ao aumento significativo do uso de dispositivos móveis para acessar a internet, garantindo que os sites e aplicações ofereçam uma experiência eficiente e agradável nesses dispositivos.

O conceito de Mobile First foi elaborado por Luke Wroblewski em 2009 e 2010 no seu blog e no ano seguinte ele publicou através da A Book Apart o livro Mobile First que é uma leitura obrigatória para aqueles que queiram compreender a fundo a técnica e suas aplicabilidades.

Por que Adotar o Mobile First?

1. **Crescimento do Uso de Dispositivos Móveis:**
   - A maioria dos acessos à internet agora ocorre por dispositivos móveis. Priorizar esses dispositivos garante que a maior parte dos usuários tenha uma experiência satisfatória.
2. **Desempenho e Velocidade:**
   - Dispositivos móveis geralmente têm limitações de processamento e conexões de internet menos estáveis. Projetar para esses ambientes exige otimização desde o início, resultando em aplicações mais rápidas e eficientes.
3. **Melhor Organização de Conteúdo:**
   - Começar pelo mobile força a priorização do conteúdo essencial, evitando sobrecarregar a interface com elementos desnecessários.
4. **SEO e Indexação:**
   - Motores de busca, como o Google, favorecem sites responsivos e otimizados para dispositivos móveis em seus resultados de pesquisa.

Princípios do Mobile First

1. **Design Progressivo:**
   - Começa com uma base sólida para dispositivos móveis e, em seguida, adiciona melhorias para telas maiores usando media queries e outras técnicas de CSS.
2. **Conteúdo Prioritário:**

    - Foco no conteúdo essencial. Determine quais informações são cruciais para o usuário móvel e destaque-as.
3. **Performance Otimizada:**
    - Minimização de recursos, como imagens e scripts, para reduzir o tempo de carregamento e melhorar a experiência do usuário.
4. **Usabilidade e Acessibilidade:**
    - Garantir que a interface seja intuitiva e acessível em telas menores, com elementos interativos adequadamente dimensionados e espaçados.

Melhores Práticas para Mobile First

1. **Planejamento de Conteúdo:**
    - Identifique quais informações são mais importantes para os usuários móveis e organize o conteúdo de forma hierárquica.
2. **Design Responsivo:**
    - Utilize layouts flexíveis e unidades relativas (como porcentagens, `em`, `rem`) para garantir que os elementos se ajustem proporcionalmente a diferentes tamanhos de tela.
3. **Imagens Otimizadas:**
    - Use imagens adaptativas e formatos otimizados para reduzir o tempo de carregamento em dispositivos móveis.
4. **Tipografia Legível:**
    - Escolha fontes claras e tamanhos adequados para facilitar a leitura em telas pequenas.
5. **Interações Intuitivas:**
    - Botões e links devem ser suficientemente grandes e espaçados para evitar toques acidentais.
6. **Teste em Diversos Dispositivos:**
    - Realize testes em diferentes dispositivos e resoluções para garantir a consistência e a usabilidade.

Desafios do Mobile First

1. **Complexidade Inicial:**
    - Pode exigir um planejamento mais detalhado e uma abordagem diferente em comparação com o design tradicional desktop-first.
2. **Compatibilidade de Navegadores:**
    - Garantir que os estilos e funcionalidades funcionem corretamente em uma variedade de navegadores móveis.
3. **Gerenciamento de Recursos:**
    - Necessidade de otimizar recursos como imagens e scripts para diferentes dispositivos sem comprometer a qualidade.

Responsividade

Responsividade refere-se à capacidade de um site ou aplicação web de se adaptar de forma fluida a diferentes tamanhos e tipos de tela, garantindo uma experiência de usuário consistente e otimizada em dispositivos móveis, tablets e desktops. O objetivo é que o conteúdo seja facilmente acessível e utilizável, independentemente do dispositivo utilizado.

Por Que a Responsividade é Importante?

1. **Experiência do Usuário (UX) Melhorada:**
   - Usuários esperam que os sites funcionem bem em qualquer dispositivo. Um design responsivo atende a essas expectativas, aumentando a satisfação e a retenção do usuário.
2. **SEO (Search Engine Optimization):**
   - Motores de busca como o Google favorecem sites responsivos nos resultados de pesquisa, melhorando a visibilidade e o tráfego orgânico.
3. **Manutenção Simplificada:**
   - Em vez de manter versões separadas para desktop e mobile, um design responsivo unifica o desenvolvimento e a manutenção.
4. **Aumento no Alcance:**
   - Com a diversidade de dispositivos no mercado, garantir que seu site seja acessível a todos amplia seu público-alvo.

Técnicas para Implementar Responsividade

1. **Media Queries**

As media queries são a base do design responsivo. Elas permitem aplicar estilos CSS específicos com base nas características do dispositivo, como largura, altura, resolução, orientação, etc.

Exemplo Básico de Media Queries:

```CSS
/* Estilos para dispositivos móveis */
body {
  font-size: 16px;
  padding: 10px;
}
```

```css
/* Estilos para tablets (largura mínima de 768px) */
@media (min-width: 768px) {
  body {
    font-size: 18px;
  }
}

/* Estilos para desktops (largura mínima de 1024px) */
@media (min-width: 1024px) {
  body {
    font-size: 20px;
  }
}
```

2. **Grid Layout**

O **CSS Grid Layout** é um sistema de layout bidimensional que permite criar designs complexos com facilidade. Ele é especialmente útil para estruturar conteúdo em linhas e colunas.

Exemplo de Grid Layout:

CSS

```css
.container {
  display: grid;
  grid-template-columns: 1fr; /* Uma coluna para mobile */
  gap: 20px;
}

@media (min-width: 768px) {
  .container {
    grid-template-columns: repeat(2, 1fr); /* Duas colunas para tablets
*/
  }
}

@media (min-width: 1024px) {
  .container {
    grid-template-columns: repeat(3, 1fr); /* Três colunas para desktops
*/
  }
}
```

3. **Imagens e Mídias Flexíveis**

Para garantir que as imagens e outros elementos de mídia se ajustem ao tamanho da tela, utilize unidades relativas e propriedades CSS específicas.

Exemplo de Imagem Responsiva:

```CSS
img {
  max-width: 100%;
  height: auto;
}
```

4. **Unidades Relativas**

Utilize unidades como `%`, `em`, `rem`, `vw`, `vh` para definir tamanhos que se adaptam ao contexto da tela.

Exemplo de Uso de Unidades Relativas:

```CSS
.container {
  width: 90%; /* 90% da largura do elemento pai */
  padding: 2em; /* Espaçamento relativo ao tamanho da fonte */
}

h1 {
  font-size: 2.5rem; /* 2.5 vezes o tamanho da fonte raiz */
}
```

Implementando o Mobile First com CSS

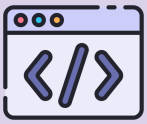
A abordagem Mobile First pode ser implementada utilizando **media queries** no CSS, onde os estilos base são definidos para dispositivos móveis e os estilos adicionais são aplicados para telas maiores.

```CSS
/* Estilos para dispositivos móveis */
body {
  font-size: 16px;
  padding: 10px;
  display: flex;
```

```
  flex-direction: column;
}

.container {
  width: 100%;
  padding: 10px;
}

.nav {
  display: none; /* Menu simplificado para mobile */
}

/* Estilos para tablets */
@media (min-width: 768px) {
  body {
    font-size: 18px;
  }

  .nav {
    display: flex; /* Menu completo para tablets */
    justify-content: space-around;
  }
}

/* Estilos para desktops */
@media (min-width: 1024px) {
  body {
    font-size: 20px;
  }

  .container {
    width: 80%;
    margin: 0 auto;
  }
}
```

Vantagens dessa Abordagem:

- **Simplificação do CSS:**
  - Menos sobrecarga de estilos para dispositivos móveis, já que os estilos para telas maiores são incrementais.
- **Manutenção Facilitada:**
  - Código mais organizado e fácil de manter, pois a progressão é linear do mobile para o desktop.



MÃO NA MASSA

Vamos exercitar os conhecimentos aprendidos até aqui.

Para testar nosso código podemos usar um editor online. Uma sugestão é o html-css-js, mas você pode usar qualquer uma de sua preferência.

Segue abaixo um exemplo de código para você testar. Fique a vontade para fazer alterações:

HTML - CSS - Javascript

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Curso Técnico em Desenvolvimento de Sistemas - IFSC
Florianópolis</title>
  <style>
    /* Estilos para dispositivos móveis */
    body {
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 10px;
      background-color: #f9f9f9;
    }

    header, footer {
      background-color: #004080;
      color: #fff;
      text-align: center;
      padding: 15px 0;
    }

    nav {
      display: none; /* Menu simplificado para mobile */
      flex-direction: column;
      gap: 10px;
    }

    nav a {
      color: #fff;
      text-decoration: none;
      padding: 10px;
```

```
    background-color: #0066cc;
    border-radius: 5px;
  }

  .menu-toggle {
    display: block;
    background-color: #0066cc;
    color: #fff;
    padding: 10px;
    text-align: center;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    margin-bottom: 10px;
  }

  main {
    display: flex;
    flex-direction: column;
    gap: 20px;
  }

  .section {
    background-color: #fff;
    padding: 15px;
    border-radius: 5px;
    box-shadow: 0 2px 5px rgba(0,0,0,0.1);
  }

  .cta-button {
    display: block;
    text-align: center;
    background-color: #28a745;
    color: #fff;
    padding: 15px;
    text-decoration: none;
    border-radius: 5px;
    font-weight: bold;
    margin-top: 10px;
  }

  /* Media query para tablets */
  @media (min-width: 768px) {
    .menu-toggle {
      display: none;
    }
```

```
      nav {
        display: flex;
        justify-content: center;
        gap: 20px;
      }

      main {
        flex-direction: row;
        flex-wrap: wrap;
        justify-content: space-between;
      }

      .section {
        width: 48%;
      }
    }

    /* Media query para desktops */
    @media (min-width: 1024px) {
      main {
        justify-content: space-around;
      }

      .section {
        width: 30%;
      }
    }
  </style>
  <script>
    // Script para alternar o menu em dispositivos móveis
    function toggleMenu() {
      const nav = document.getElementById('nav-menu');
      if (nav.style.display === 'flex') {
        nav.style.display = 'none';
      } else {
        nav.style.display = 'flex';
      }
    }
  </script>
</head>
<body>

  <header>
    <h1>IFSC Florianópolis</h1>
    <h2>Curso Técnico em Desenvolvimento de Sistemas</h2>
```

```
    <button class="menu-toggle" onclick="toggleMenu()">Menu</button>
    <nav id="nav-menu">
      <a href="#sobre">Sobre o Curso</a>
      <a href="#curriculo">Currículo</a>
      <a href="#inscricao">Inscreva-se</a>
      <a href="#contato">Contato</a>
    </nav>
  </header>

  <main>
    <div class="section" id="sobre">
      <h3>Sobre o Curso</h3>
      <p>O Curso Técnico em Desenvolvimento de Sistemas do IFSC
Florianópolis prepara profissionais para atuarem no mercado de
tecnologia, com foco em programação, análise de sistemas e
desenvolvimento de aplicações web e mobile.</p>
      <a href="#inscricao" class="cta-button">Inscreva-se Agora</a>
    </div>

    <div class="section" id="curriculo">
      <h3>Currículo</h3>
      <ul>
        <li>Introdução à Programação</li>
        <li>Desenvolvimento Web com HTML, CSS e JavaScript</li>
        <li>Banco de Dados</li>
        <li>Desenvolvimento de Aplicações Mobile</li>
        <li>Gestão de Projetos de TI</li>
        <li>Estágio Supervisionado</li>
      </ul>
      <a href="#inscricao" class="cta-button">Matricule-se</a>
    </div>

    <div class="section" id="inscricao">
      <h3>Inscrições</h3>
      <p>As inscrições para o próximo semestre já estão abertas! Garanta
sua vaga e inicie sua carreira no setor de tecnologia.</p>
      <a href="https://ifsc.edu.br/inscricao" class="cta-button">Clique
Aqui para se Inscrever</a>
    </div>

    <div class="section" id="contato">
      <h3>Contato</h3>
      <p>Tem dúvidas? Entre em contato conosco:</p>
      <p>Email: <a
href="mailto:admissions@ifsc.edu.br">admissions@ifsc.edu.br</a></p>
      <p>Telefone: (48) 1234-5678</p>
```

```
      <a href="#inscricao" class="cta-button">Fale Conosco</a>
    </div>
  </main>

  <footer>
    <p>&copy; 2024 IFSC Florianópolis. Todos os direitos reservados.</p>
  </footer>

</body>
</html>
```

Flexbox

Flexbox, ou **Flexible Box Layout**, é um módulo de layout do CSS3 que facilita a criação de layouts flexíveis e responsivos. Ele permite alinhar e distribuir espaço entre os itens em um container, mesmo quando o tamanho dos itens é desconhecido ou dinâmico.

Por Que Utilizar Flexbox?

1. **Simplicidade:**
    - Flexbox simplifica o alinhamento e distribuição de elementos, reduzindo a necessidade de cálculos complexos de margem ou posicionamento.
2. **Flexibilidade:**
    - Os itens dentro de um container flexível podem crescer ou encolher para preencher o espaço disponível de forma eficiente.
3. **Alinhamento Facilitado:**
    - Facilita o alinhamento vertical e horizontal dos elementos, algo que era complicado com métodos tradicionais de layout.
4. **Design Responsivo:**
    - Flexbox é ideal para criar layouts que se adaptam automaticamente a diferentes tamanhos de tela.

Conceitos Básicos do Flexbox

1. Flex Container e Flex Items

- **Flex Container:** O elemento pai que possui a propriedade `display: flex;`. Ele define o contexto flexível para os seus filhos.
- **Flex Items:** Os elementos filhos diretos do flex container que serão organizados de acordo com as propriedades do Flexbox.

2. Eixos no Flexbox

- **Eixo Principal (Main Axis):** Define a direção em que os itens são dispostos. Pode ser horizontal (`row`) ou vertical (`column`).
- **Eixo Transversal (Cross Axis):** Perpendicular ao eixo principal. Se o eixo principal for horizontal, o eixo transversal será vertical, e vice-versa.

Propriedades do Flex Container

1. `display: flex;` ou `display: inline-flex;`

Define o elemento como um flex container. `flex` faz com que o container seja um bloco flexível, enquanto `inline-flex` faz com que ele seja um elemento inline flexível.

```CSS
.container {
  display: flex;
}
```

2. `flex-direction`

Define a direção dos itens no container.

- **Valores:**
  - `row` (padrão): Horizontal, da esquerda para a direita.
  - `row-reverse`: Horizontal, da direita para a esquerda.
  - `column`: Vertical, de cima para baixo.
  - `column-reverse`: Vertical, de baixo para cima.

```CSS
.container {
  flex-direction: row; /* Itens dispostos horizontalmente */
}
```

3. `flex-wrap`

Determina se os itens devem quebrar para uma nova linha ou coluna quando não houver espaço suficiente.

- **Valores:**
  - `nowrap` (padrão): Todos os itens em uma única linha ou coluna.
  - `wrap`: Itens quebram para múltiplas linhas ou colunas.
  - `wrap-reverse`: Itens quebram para múltiplas linhas ou colunas na direção oposta.

```CSS
.container {
  flex-wrap: wrap; /* Itens podem quebrar para múltiplas linhas */
}
```

4. `justify-content`

Alinha os itens ao longo do eixo principal.

- **Valores Comuns:**
  - `flex-start` (padrão): Alinha itens ao início.
  - `flex-end`: Alinha itens ao final.
  - `center`: Centraliza os itens.
  - `space-between`: Distribui os itens com espaços iguais entre eles.
  - `space-around`: Distribui os itens com espaços iguais ao redor.
  - `space-evenly`: Distribui os itens com espaços iguais entre eles.

```CSS
.container {
  justify-content: space-between; /* Espaço igual entre os itens */
}
```

5. `align-items`

Alinha os itens ao longo do eixo transversal.

- **Valores Comuns:**
  - stretch (padrão): Estica os itens para preencher o container.
  - flex-start: Alinha itens ao início do eixo transversal.
  - flex-end: Alinha itens ao final do eixo transversal.
  - center: Centraliza os itens.
  - baseline: Alinha itens pela linha de base do conteúdo.

```CSS
.container {
  align-items: center; /* Centraliza os itens verticalmente */
}
```

6. `align-content`

Alinha as linhas de itens quando há espaço extra no eixo transversal. Funciona somente se houver múltiplas linhas de itens.

- **Valores Comuns:**
  - `flex-start` (padrão): Alinha itens ao início.
  - `flex-end`: Alinha itens ao final.
  - `center`: Centraliza os itens.
  - `space-between`: Distribui os itens com espaços iguais entre eles.
  - `space-around`: Distribui os itens com espaços iguais ao redor.
  - `stretch`: Estica os itens para preencher o container.

```CSS
.container {
  align-content: space-between; /* Espaço igual entre as linhas */
}
```

Propriedades dos Flex Items

1. `order`

Determina a ordem de exibição dos itens no container. Itens com valores menores aparecem primeiro.

```CSS
.item {
  order: 2; /* Este item aparecerá após itens com order menor */
}
```

2. `flex-grow`

Define a capacidade de um item de crescer para preencher o espaço disponível.

```CSS
.item {
  flex-grow: 1; /* O item pode crescer igualmente para preencher o
espaço */
}
```

3. `flex-shrink`

Define a capacidade de um item de encolher quando necessário.

```css
.item {
  flex-shrink: 1; /* O item pode encolher se necessário */
}
```

4. `flex-basis`

Define o tamanho inicial de um item antes de o espaço restante ser distribuído.

```css
.item {
  flex-basis: 200px; /* Tamanho inicial de 200px */
}
```

5. `align-self`

Alinha um item específico ao longo do eixo transversal, sobrescrevendo `align-items`.

- **Valores Comuns:**
  - stretch (padrão): Estica os itens para preencher o container.
  - flex-start: Alinha itens ao início do eixo transversal.
  - flex-end: Alinha itens ao final do eixo transversal.
  - center: Centraliza os itens.
  - baseline: Alinha itens pela linha de base do conteúdo.

```css
.item {
  align-self: flex-end; /* Alinha este item ao final do eixo transversal
*/
}
```

Exemplos Práticos de Flexbox

Exemplo 1: Alinhamento Horizontal e Vertical

Vamos criar um container flexível que centraliza seus itens tanto horizontalmente quanto verticalmente.

- O container utiliza **`display: flex;`** para ativar o Flexbox.
- **`justify-content: center;`** centraliza os itens horizontalmente.
- **`align-items: center;`** centraliza os itens verticalmente.
- A caixa interna (**`.box`**) também é um flex container para centralizar seu conteúdo.

HTML

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <title>Exemplo Flexbox - Centralização</title>
  <style>
    .container {
      display: flex;
      justify-content: center; /* Centraliza horizontalmente */
      align-items: center; /* Centraliza verticalmente */
      height: 100vh; /* Altura total da viewport */
      background-color: #f0f0f0;
    }

    .box {
      width: 150px;
      height: 150px;
      background-color: #004080;
      color: #fff;
      display: flex;
      justify-content: center;
      align-items: center;
      border-radius: 10px;
    }
  </style>
</head>
<body>

  <div class="container">
    <div class="box">Centralizado</div>
  </div>
```

```
</body>
</html>
```

Exemplo 2: Layout de Cards Responsivo

Criar um layout de cards que se ajusta automaticamente ao tamanho da tela utilizando Flexbox.

HTML

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <title>Exemplo Flexbox - Layout de Cards</title>
  <style>
    .container {
      display: flex;
      flex-wrap: wrap; /* Permite que os itens quebrem para novas linhas
*/
      gap: 20px;
      padding: 20px;
      background-color: #f9f9f9;
    }

    .card {
      flex: 1 1 calc(100% - 40px); /* Cresce, encolhe e ocupa 100% menos
o gap */
      background-color: #fff;
      padding: 20px;
      border-radius: 5px;
      box-shadow: 0 2px 5px rgba(0,0,0,0.1);
      transition: transform 0.3s;
    }

    .card:hover {
      transform: translateY(-5px);
    }

    @media (min-width: 600px) {
      .card {
        flex: 1 1 calc(50% - 40px); /* Dois cards por linha */
      }
    }
```

```
    @media (min-width: 900px) {
      .card {
        flex: 1 1 calc(33.333% - 40px); /* Três cards por linha */
      }
    }
  </style>
</head>
<body>

  <div class="container">
    <div class="card">Card 1</div>
    <div class="card">Card 2</div>
    <div class="card">Card 3</div>
    <div class="card">Card 4</div>
    <div class="card">Card 5</div>
    <div class="card">Card 6</div>
  </div>

</body>
</html>
```

Entendendo o código:

- **Flex Container (`.container`):**
  - `display: flex;` ativa o Flexbox.
  - `flex-wrap: wrap;` permite que os itens quebrem para novas linhas.
  - `gap: 20px;` adiciona espaçamento entre os itens.
- **Flex Items (`.card`):**
  - `flex: 1 1 calc(100% - 40px);` permite que cada card cresça e encolha, ocupando 100% menos o espaço do gap.
  - **Media Queries:**
    - A partir de 600px, os cards ocupam 50% da largura, resultando em dois cards por linha.
    - A partir de 900px, os cards ocupam 33.333% da largura, resultando em três cards por linha.
- **Interatividade:**
  - Efeito de hover que eleva o card ligeiramente para melhorar a interatividade.

Exemplo 3: Barra de Navegação Responsiva

Criar uma barra de navegação que se adapta a diferentes tamanhos de tela usando Flexbox.

HTML

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <title>Exemplo Flexbox - Barra de Navegação</title>
  <style>
    body {
      margin: 0;
      font-family: Arial, sans-serif;
    }

    header {
      background-color: #004080;
      color: #fff;
      padding: 15px 20px;
      display: flex;
      justify-content: space-between;
      align-items: center;
    }

    .logo {
      font-size: 1.5em;
      font-weight: bold;
    }

    nav {
      display: flex;
      gap: 15px;
    }

    nav a {
      color: #fff;
      text-decoration: none;
      padding: 8px 12px;
      border-radius: 4px;
      transition: background-color 0.3s;
    }

    nav a:hover {
      background-color: #0066cc;
```

```
    }

    /* Menu Mobile */
    .menu-toggle {
      display: none;
      background-color: #0066cc;
      color: #fff;
      padding: 8px 12px;
      border: none;
      border-radius: 4px;
      cursor: pointer;
    }

    @media (max-width: 768px) {
      nav {
        display: none;
        flex-direction: column;
        width: 100%;
        background-color: #004080;
      }

      nav a {
        padding: 12px 20px;
        border-top: 1px solid #0066cc;
      }

      .menu-toggle {
        display: block;
      }

      .nav-open {
        display: flex;
      }
    }
  </style>
  <script>
    function toggleNav() {
      const nav = document.getElementById('nav-menu');
      nav.classList.toggle('nav-open');
    }
  </script>
</head>
<body>

  <header>
    <div class="logo">IFSC Florianópolis</div>
```

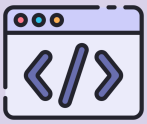
```
    <button class="menu-toggle" onclick="toggleNav()">Menu</button>
    <nav id="nav-menu">
      <a href="#sobre">Sobre</a>
      <a href="#curriculo">Currículo</a>
      <a href="#inscricao">Inscreva-se</a>
      <a href="#contato">Contato</a>
    </nav>
  </header>

  <main>
    <!-- Conteúdo da página -->
  </main>

</body>
</html>
```

Entendendo o código:

- **Header:**
  - `display: flex;` organiza o logo e o menu horizontalmente.
  - `justify-content: space-between;` distribui espaço entre o logo e o botão de menu.
- **Navegação (`nav`):**
  - Em telas maiores, exibe os links de navegação horizontalmente.
  - Em telas menores (até 768px), a navegação é ocultada e pode ser exibida clicando no botão "Menu".
- **Menu Toggle:**
  - Botão visível apenas em dispositivos móveis para alternar a visibilidade do menu.
- **Interatividade:**
  - O JavaScript adiciona ou remove a classe `nav-open` para exibir ou ocultar o menu em dispositivos móveis.

Dicas para Utilizar Flexbox Eficientemente

1. **Planeje o Layout:**
    - Antes de escrever o código, visualize como os elementos devem se comportar em diferentes dispositivos. Isso facilita a escolha das propriedades Flexbox adequadas.
2. **Utilize as Ferramentas de Desenvolvimento:**
    - Ferramentas como o **Inspector** do navegador permitem testar e ajustar as propriedades Flexbox em tempo real.
3. **Combine Flexbox com Media Queries:**
    - Para layouts mais complexos, combine Flexbox com media queries para adaptar o layout conforme o tamanho da tela.
4. **Conheça as Propriedades:**
    - Familiarize-se com todas as propriedades do Flexbox para aproveitar ao máximo suas funcionalidades.
5. **Evite Sobrecarregar:**
    - Use Flexbox quando ele realmente facilitar o layout. Em alguns casos, Grid Layout pode ser mais apropriado para layouts bidimensionais.



MÃO NA MASSA

Vamos exercitar os conhecimentos aprendidos até aqui.

Para testar nosso código podemos usar um editor online. Uma sugestão é o html-css-js, mas você pode usar qualquer uma de sua preferência.

Segue abaixo um exemplo de código para você testar. Fique a vontade para fazer alterações:

HTML

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <title>Curso de Desenvolvimento de Sistemas - IFSC
Florianópolis</title>
  <style>
    /* Reset básico */
    * {
      margin: 0;
      padding: 0;
      box-sizing: border-box;
    }

    body {
      font-family: Arial, sans-serif;
      background-color: #f0f0f0;
      line-height: 1.6;
    }

    /* Barra de Navegação */
    header {
      background-color: #004080;
      color: #fff;
      padding: 15px 20px;
      display: flex;
      justify-content: space-between;
      align-items: center;
      position: sticky;
      top: 0;
      z-index: 1000;
    }
```

```css
.logo {
  font-size: 1.8em;
  font-weight: bold;
}

nav {
  display: flex;
  gap: 20px;
}

nav a {
  color: #fff;
  text-decoration: none;
  padding: 8px 12px;
  border-radius: 4px;
  transition: background-color 0.3s;
}

nav a:hover {
  background-color: #0066cc;
}

/* Menu Mobile */
.menu-toggle {
  display: none;
  background-color: #0066cc;
  color: #fff;
  padding: 8px 12px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size: 1em;
}

@media (max-width: 768px) {
  nav {
    display: none;
    flex-direction: column;
    width: 100%;
    background-color: #004080;
  }

  nav a {
    padding: 12px 20px;
    border-top: 1px solid #0066cc;
  }
```

```
      .menu-toggle {
        display: block;
      }

      .nav-open {
        display: flex;
      }
    }

    /* Seção Hero */
    .hero {
      display: flex;
      justify-content: center;
      align-items: center;
      height: 60vh;
      background: url('https://via.placeholder.com/1200x600') no-repeat
center center/cover;
      color: #fff;
      text-align: center;
      position: relative;
    }

    .hero::after {
      content: "";
      position: absolute;
      top: 0;
      left: 0;
      width: 100%;
      height: 100%;
      background-color: rgba(0, 0, 0, 0.5);
    }

    .hero-content {
      position: relative;
      z-index: 1;
      max-width: 800px;
      padding: 20px;
    }

    .hero h1 {
      font-size: 3em;
      margin-bottom: 20px;
    }

    .hero p {
```

```
  font-size: 1.2em;
  margin-bottom: 30px;
}

.hero a {
  background-color: #0066cc;
  color: #fff;
  padding: 12px 25px;
  text-decoration: none;
  border-radius: 5px;
  font-size: 1em;
  transition: background-color 0.3s;
}

.hero a:hover {
  background-color: #005bb5;
}

/* Seção Sobre o Curso */
.sobre-container {
  padding: 40px 20px;
  background-color: #fff;
  text-align: center;
}

.sobre-container h2 {
  font-size: 2em;
  margin-bottom: 20px;
  color: #004080;
}

.sobre-container p {
  max-width: 800px;
  margin: 0 auto;
  font-size: 1.1em;
  color: #333;
}

/* Grade Curricular */
.grade-container {
  padding: 40px 20px;
  background-color: #f9f9f9;
}

.grade-container h2 {
  font-size: 2em;
```

```
  margin-bottom: 20px;
  text-align: center;
  color: #004080;
}

.cards-container {
  display: flex;
  flex-wrap: wrap;
  gap: 20px;
  justify-content: center;
}

.card {
  flex: 1 1 calc(100% - 40px);
  background-color: #fff;
  padding: 20px;
  border-radius: 5px;
  box-shadow: 0 2px 5px rgba(0,0,0,0.1);
  transition: transform 0.3s;
  max-width: 300px;
}

.card:hover {
  transform: translateY(-5px);
}

.card h3 {
  margin-bottom: 10px;
  color: #004080;
}

.card p {
  color: #555;
  font-size: 0.95em;
}

@media (min-width: 600px) {
  .card {
    flex: 1 1 calc(50% - 40px);
  }
}

@media (min-width: 900px) {
  .card {
    flex: 1 1 calc(33.333% - 40px);
  }
```

```css
}

/* Benefícios */
.beneficios-container {
  padding: 40px 20px;
  background-color: #fff;
  text-align: center;
}

.beneficios-container h2 {
  font-size: 2em;
  margin-bottom: 20px;
  color: #004080;
}

.beneficios-container .beneficios {
  display: flex;
  flex-wrap: wrap;
  gap: 20px;
  justify-content: center;
}

.beneficio {
  flex: 1 1 calc(100% - 40px);
  background-color: #f0f0f0;
  padding: 20px;
  border-radius: 5px;
  max-width: 300px;
}

.beneficio h3 {
  margin-bottom: 10px;
  color: #004080;
}

.beneficio p {
  color: #555;
  font-size: 0.95em;
}

@media (min-width: 600px) {
  .beneficio {
    flex: 1 1 calc(50% - 40px);
  }
}
```

```css
@media (min-width: 900px) {
  .beneficio {
    flex: 1 1 calc(33.333% - 40px);
  }
}

/* Depoimentos */
.depoimentos-container {
  padding: 40px 20px;
  background-color: #f9f9f9;
  text-align: center;
}

.depoimentos-container h2 {
  font-size: 2em;
  margin-bottom: 20px;
  color: #004080;
}

.depoimentos {
  display: flex;
  flex-wrap: wrap;
  gap: 20px;
  justify-content: center;
}

.depoimento {
  flex: 1 1 calc(100% - 40px);
  background-color: #fff;
  padding: 20px;
  border-radius: 5px;
  box-shadow: 0 2px 5px rgba(0,0,0,0.1);
  max-width: 400px;
}

.depoimento p {
  font-style: italic;
  color: #555;
  margin-bottom: 15px;
}

.depoimento .autor {
  font-weight: bold;
  color: #004080;
}
```

```css
@media (min-width: 600px) {
  .depoimento {
    flex: 1 1 calc(50% - 40px);
  }
}

/* Seção de Inscrição */
.inscricao-container {
  padding: 40px 20px;
  background-color: #fff;
  text-align: center;
}

.inscricao-container h2 {
  font-size: 2em;
  margin-bottom: 20px;
  color: #004080;
}

.inscricao-container p {
  max-width: 800px;
  margin: 0 auto 30px auto;
  font-size: 1.1em;
  color: #333;
}

.inscricao-container a {
  background-color: #0066cc;
  color: #fff;
  padding: 15px 30px;
  text-decoration: none;
  border-radius: 5px;
  font-size: 1.1em;
  transition: background-color 0.3s;
}

.inscricao-container a:hover {
  background-color: #005bb5;
}

/* Rodapé */
footer {
  background-color: #004080;
  color: #fff;
  text-align: center;
  padding: 20px;
```

```
      position: relative;
      /* Removido fixed para evitar sobreposição */
    }

    footer a {
      color: #fff;
      text-decoration: underline;
    }
  </style>
  <script>
    function toggleNav() {
      const nav = document.getElementById('nav-menu');
      nav.classList.toggle('nav-open');
    }
  </script>
</head>
<body>

  <!-- Barra de Navegação -->
  <header>
    <div class="logo">IFSC Florianópolis</div>
    <button class="menu-toggle" onclick="toggleNav()">Menu</button>
    <nav id="nav-menu">
      <a href="#hero">Início</a>
      <a href="#sobre">Sobre o Curso</a>
      <a href="#grade">Grade Curricular</a>
      <a href="#beneficios">Benefícios</a>
      <a href="#depoimentos">Depoimentos</a>
      <a href="#inscricao">Inscreva-se</a>
    </nav>
  </header>

  <!-- Seção Hero -->
  <section id="hero" class="hero">
    <div class="hero-content">
      <h1>Curso de Desenvolvimento de Sistemas</h1>
      <p>Prepare-se para o mercado de trabalho com uma formação completa
e atualizada no IFSC Campus Florianópolis.</p>
      <a href="#inscricao">Inscreva-se Agora</a>
    </div>
  </section>

  <!-- Seção Sobre o Curso -->
  <section id="sobre" class="sobre-container">
    <h2>Sobre o Curso</h2>
    <p>O Curso de Desenvolvimento de Sistemas do IFSC Campus
```

```
Florianópolis oferece uma formação abrangente, combinando teoria e
prática para capacitar os estudantes a desenvolver soluções inovadoras
em tecnologia da informação. Com professores altamente qualificados e
infraestrutura moderna, o curso prepara os alunos para enfrentar os
desafios do mercado e contribuir significativamente para a indústria de
tecnologia.</p>
  </section>

  <!-- Grade Curricular -->
  <section id="grade" class="grade-container">
    <h2>Grade Curricular</h2>
    <div class="cards-container">
      <div class="card">
        <h3>Programação Web</h3>
        <p>Aprenda a desenvolver aplicações web dinâmicas utilizando as
mais recentes tecnologias e frameworks.</p>
      </div>
      <div class="card">
        <h3>Banco de Dados</h3>
        <p>Entenda o design, implementação e manutenção de bancos de
dados eficientes e seguros.</p>
      </div>
      <div class="card">
        <h3>Desenvolvimento Mobile</h3>
        <p>Crie aplicativos para dispositivos móveis, explorando as
melhores práticas de design e usabilidade.</p>
      </div>
      <div class="card">
        <h3>Engenharia de Software</h3>
        <p>Adquira conhecimentos sobre processos de desenvolvimento,
metodologias ágeis e gestão de projetos.</p>
      </div>
      <div class="card">
        <h3>Inteligência Artificial</h3>
        <p>Explore os fundamentos e aplicações da inteligência
artificial e aprendizado de máquina.</p>
      </div>
      <div class="card">
        <h3>Segurança da Informação</h3>
        <p>Aprenda a proteger sistemas e dados contra ameaças e
vulnerabilidades.</p>
      </div>
    </div>
  </section>

  <!-- Benefícios -->
```

```
  <section id="beneficios" class="beneficios-container">
    <h2>Benefícios do Curso</h2>
    <div class="beneficios">
      <div class="beneficio">
        <h3>Infraestrutura Moderna</h3>
        <p>Laboratórios equipados com tecnologia de ponta para um
aprendizado prático e eficiente.</p>
      </div>
      <div class="beneficio">
        <h3>Corpo Docente Qualificado</h3>
        <p>Professores com vasta experiência no mercado de trabalho e em
pesquisa acadêmica.</p>
      </div>
      <div class="beneficio">
        <h3>Parcerias com Empresas</h3>
        <p>Oportunidades de estágio e networking com empresas líderes do
setor de tecnologia.</p>
      </div>
      <div class="beneficio">
        <h3>Projetos Práticos</h3>
        <p>Desenvolvimento de projetos reais que permitem aplicar os
conhecimentos adquiridos.</p>
      </div>
      <div class="beneficio">
        <h3>Certificação</h3>
        <p>Certificados reconhecidos que valorizam seu currículo no
mercado de trabalho.</p>
      </div>
      <div class="beneficio">
        <h3>Ambiente Colaborativo</h3>
        <p>Espaços e atividades que incentivam a colaboração e o
trabalho em equipe.</p>
      </div>
    </div>
  </section>

  <!-- Depoimentos -->
  <section id="depoimentos" class="depoimentos-container">
    <h2>Depoimentos</h2>
    <div class="depoimentos">
      <div class="depoimento">
        <p>"O curso de Desenvolvimento de Sistemas no IFSC me
proporcionou as habilidades necessárias para iniciar minha carreira na
área de TI. A qualidade dos professores e a infraestrutura são
excelentes."</p>
        <div class="autor">- João Silva, Ex-Aluno</div>
```

```
      </div>
      <div class="depoimento">
        <p>"As parcerias com empresas me permitiram realizar estágios
que foram fundamentais para meu desenvolvimento profissional. Recomendo
a todos que buscam uma formação sólida."</p>
        <div class="autor">- Maria Oliveira, Aluna Atual</div>
      </div>
      <div class="depoimento">
        <p>"Os projetos práticos realizados durante o curso me deram
confiança para desenvolver minhas próprias aplicações e contribuir para
projetos inovadores."</p>
        <div class="autor">- Pedro Santos, Ex-Aluno</div>
      </div>
      <div class="depoimento">
        <p>"A abordagem prática do curso, aliada ao conhecimento
teórico, me preparou para enfrentar os desafios do mercado de trabalho
com competência."</p>
        <div class="autor">- Ana Paula, Aluna Atual</div>
      </div>
    </div>
  </section>

  <!-- Seção de Inscrição -->
  <section id="inscricao" class="inscricao-container">
    <h2>Inscreva-se Agora</h2>
    <p>Não perca a oportunidade de transformar sua carreira com o Curso
de Desenvolvimento de Sistemas do IFSC Campus Florianópolis. As vagas
são limitadas!</p>
    <a href="https://www.ifsc.edu.br/inscricao" target="_blank">Faça sua
Inscrição</a>
  </section>

  <!-- Rodapé -->
  <footer>
    <p>Entre em contato: <a
href="mailto:contato@ifsc.edu.br">contato@ifsc.edu.br</a> | Telefone:
(48) 3341-XXXX</p>
    <p>&copy; 2024 IFSC Florianópolis. Todos os direitos reservados.</p>
  </footer>

</body>
</html>
```