A21. CSS3:Flexbox & Grid Layout

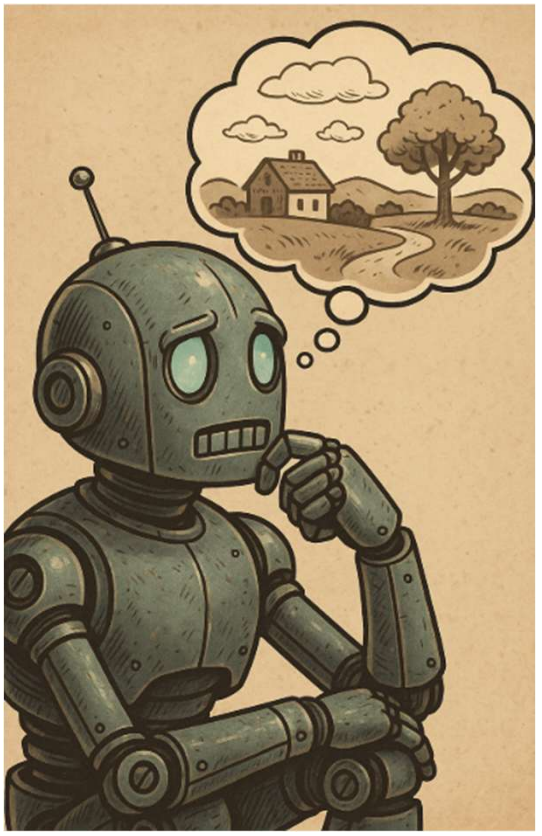
M5. Estilização de Páginas com CSS

Objetivo da aula

- Compreender os conceitos básicos de **Flexbox** e do **Grid Layout**
- Aplicar alinhamento, espaçamento e organização responsiva com Flexbox
- Criar layouts mais complexos e organizados com Grid

Relembrando a última aula...

- ✓ Conceito de Box Model
- ✓ Propriedade Display
- ✓ Posicionamento de Elementos



Nossa aula...

- ✓ Introdução a Layout Responsivo
- ✓ Flexbox
- ✓ Grid Layout
- ✓ Flexbox x Grid

Problema dos layouts antigos

- **Antes do Flexbox/ Grid**
  - Desenvolvedores usavam
  - **float** → para alinhar caixas lado ao lado
  - **inline-block** → para colocar elementos na mesma linha
- **Problemas**
  - Dificuldades em alinhar verticalmente
  - Códigos cheios de *gambiarras*
  - Responsividade complicada, precisava de muitos media queries

Destaques

1. Usa **float: left** para posicionar os blocos lado a lado
2. Precisa de **overflow: auto** ou **clear: both** para evitar que o rodapé suba
3. Esse modelo funciona, mas não é flexível:
   - Se mudar o tamanho da tela, quebra
   - Ajustes responsivos existem muito **media queries**

Conceito de Flexbox

- Flexbox (*Flexible Box Layout*) é um modelo de layout do CSS3 criado para **organizar elementos em linha ou coluna** de forma fácil
- Permite **alinhamento, espaçamento e distribuição de elementos** sem precisar de *gambiarras* como float
- **Estrutura**
  - Ativado com →

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

- **Principais propriedades do container**
  - flex-direction → direção do itens (row, column)
  - justtify-content → alinhamento no eixo principal (horizontal por padrão)
  - align-items → alinhamento no eixo transversal (vertical por padrão)
  - gap → espaço entre os itens

D1.
Vamos demonstrar...

Dimensionamento Flexível

- **Propriedades principais dos itens flex**
  - Cada item dentro de um container display: flex pode ter **3 propriedades** que controlam seu tamanho

1. flex-grow → define **quanto o item pode crescer** quando há espaço sobrando
2. flex-shrink → define **quanto o item pode encolher** quando falta espaço
3. flex-basis → define o **tamanho inicial** do item (antes do Grow/ Shrink)

```
flex: grow shrink basis;

flex: 1 1 100px;
```

D2.
Vamos demonstrar...

Termos e Principais Propriedades Grid

- **Termos importantes**
  - Container Grid: o elemento pai
  - Itens Grid: os elementos filho
  - Células: intersecção entre linhas e colunas
  - Áreas: agrupamento de células que formam regiões (header, sidebar, main, footer)
- **Principais propriedades do container**
  - grid-template-columns: define quantas colunas e seus tamanhos
  - grid-template-rows: define as linhas
  - gap: espaçamento entre células
  - grid-template-áreas: permite nomear regiões do layout

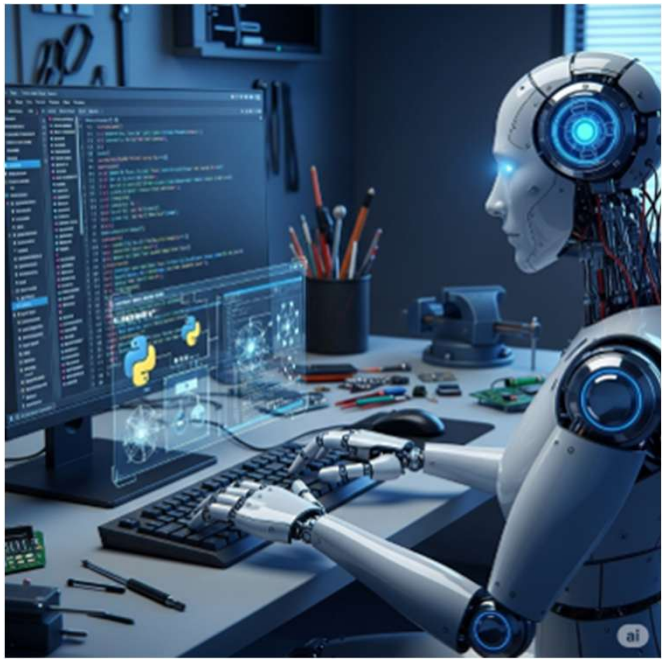
D3.
Vamos demonstrar...

D4.
Um exemplo mais completo

Diferença entre Flexbox e Grid

- Quando usamos display: **flex**
  - Os itens ficam um ao lado do outro, em linha
  - O Flexbox trabalha em um eixo por vez (horizontal OU vertical)
  - Ideal para *menus, listas de cards, botões, barras de navegação*
- Quando usamos display: **grid**
  - Os itens ficam organizados em linha E coluna ao mesmo tempo
  - O Grid trabalha nos dois eixos ao mesmo tempo
  - Ideal para *layouts completos de páginas ou seções complexas*

A1.
Vamos praticar..



Usando Flexbox para organizar elementos lado a lado, com espaçamento uniforme e quebra automática de linha

Agora, refaça a atividade utilizando Grid

Perguntas?! Dúvidas?!?