

# DEUX #4

# CSS & FLEXBOX

Kevin Brain

# CSS + FLEXBOX

**Construindo Interfaces**

Hoje vais aprender:

1. Fundamentos essenciais de CSS
2. Como organizar layout com Flexbox

# FUNDAMENTOS ESSENCIAIS DE CSS

Antes de organizar layout,  
precisas entender o básico.

# O QUE É CSS?

CSS é responsável pela parte visual do site.

HTML → Estrutura

CSS → Aparência e organização

Sem CSS → página desorganizada

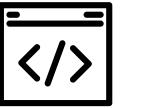
Com CSS → layout profissional

# COMO LIGAMOS CSS AO HTML?



**Dentro do <head>**

```
<link rel="stylesheet" href="style.css">
```



**Criar ficheiro:**

style.css

**Evitar e Nunca usar style inline.**

# ESTRUTURA DE UM CÓDIGO CSS



**Praticar**

```
seletor {propriedade: valor; }
```

p → seletor

color → propriedade

blue → valor



**Praticar**

```
p { color: blue; }
```

# SELETORES ESSENCIAIS



**HTML:**

```
<div class="card" id="header"></div>
```



**Elemento:**

```
p {}  
div {}
```



**Classe:**

```
.card {}
```



**ID (usar pouco):**

```
#header {}
```

# BOX MODEL (MUITO IMPORTANTE)

Tudo no CSS é uma caixa.

Cada elemento tem:

Content

Padding

Border

Margin



## Practicar

```
box {  
    width: 200px;  
    padding: 20px;  
    border: 2px solid black;  
    margin: 10px;  
}
```

# REGRA OBRIGATÓRIA

Sempre colocar no início do CSS:



## Praticar

```
* { box-sizing: border-box; }
```

Evita problemas de tamanho.

# ESPAÇAMENTO CORRETO

Use:

padding → espaço interno

margin → espaço externo

Nunca usar <br> para espaçamento.

# FLEXBOX - ORGANIZAR LAYOUT

Agora vamos organizar de verdade.

# O PROBLEMA SEM FLEXBOX

Difícil alinhar

Difícil centralizar

Layout quebra facilmente

Flexbox resolve isso.

# COMO ATIVAR FLEXBOX



**Praticar**

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

Agora os filhos ficam lado a lado.

# ESPAÇAMENTO CORRETO

Main Axis → Horizontal

Cross Axis → Vertical

Por padrão:

Principal é horizontal

# JUSTIFY-CONTENT



**Alinha no eixo horizontal**

justify-content: center;



**Outras opções:**

flex-start  
flex-end  
space-between  
space-around  
space-evenly

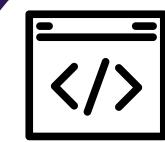
# ALIGN-ITEMS



**Alinha no eixo vertical**

align-items: center;

Muito usado em headers

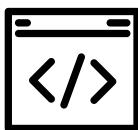


## Centralização Perfeita

```
container {display: flex;justify-content: center;align-items: center;min-height: 100vh;}
```

.Centraliza qualquer elemento.

# FLEX-DIRECTION

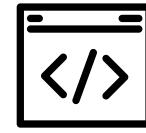


**Muda direção**

`flex-direction: row;`  
`flex-direction: column;`

Column → empilha verticalmente.

# GAP

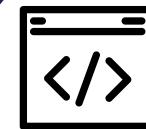


## Espaçamento moderno

gap: 20px;

Mais limpo que usar margin em tudo.

# FLEX-WRAP



## Responsividade

`flex-wrap: wrap;`

Elementos quebram linha quando não há espaço.

# FLEX



## Distribuição proporcional

```
.box { flex: 1; }
```

Todos ocupam espaço igual.

# PROJETO PRÁTICO DA AULA

**Construir:**

Header com logo e menu

Hero centralizado

3 cards lado a lado

Layout responsivo básico

**Entrega obrigatória:**

Desktop

Mobil

# ERROS QUE NÃO PERMITIMOS

Não usar:

position para layout

float

margin-left para empurrar layout

br para espaçamento

# RESULTADO ESPERADO

**Dominares:**

Box Model

justify-content

align-items

flex-direction

gap

flex-wrap

# MISSÃO DE HOJE

- APLICAR CSS FLEXBOX
- ESTILIZAR COM CORES  
SEGUINDO DESIGN