

CSS3 (II): posicionamento, Flexbox, Grid Layout

QXD0020 - Desenvolvimento de Software para Web

Universidade Federal do Ceará - *Campus Quixadá*

Prof. Francisco Victor da Silva Pinheiro
victorpinheiro@ufc.br

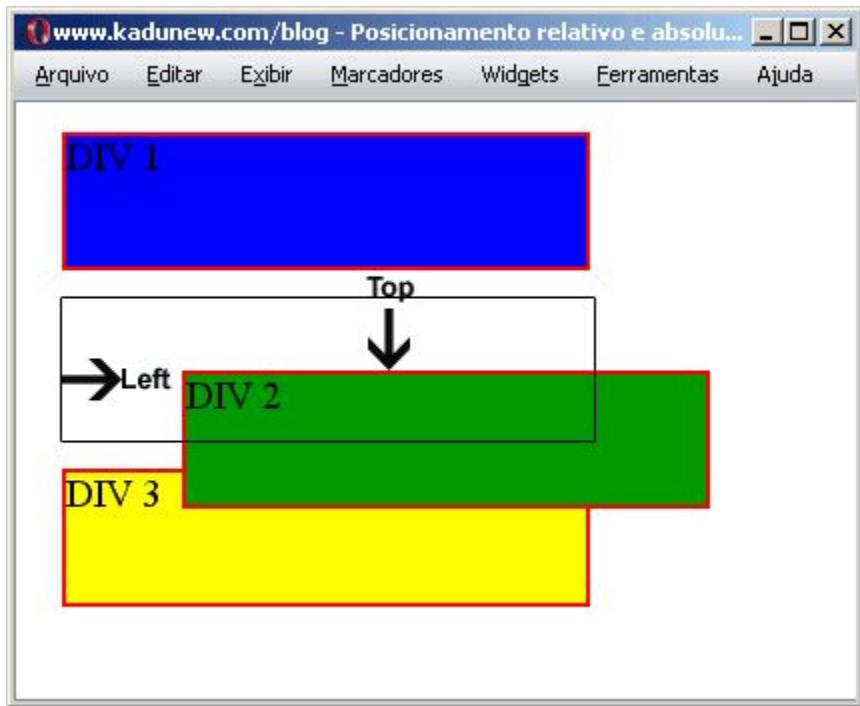


Agenda

- Posicionamento em CSS
- Tipos de posicionamento
- Flexbox (Flexible Box Layout)
- Eixos do Flexbox
- Propriedades do container
- Propriedades dos itens
- Grid Layout
- Propriedades do container
- Propriedades dos itens
- Comparação Flexbox x Grid
- Boas práticas

Posicionamento em CSS

- Conceito
 - O posicionamento define como os elementos se organizam no espaço da página e como interagem com o fluxo normal do documento. É a base para criar layouts controlados.



Tipos de posicionamento

- static (padrão)
 - Segue o fluxo natural da página.
 - Não reage a top, left, etc.

```
.box {  
  position: static; /* valor default */  
}
```

Tipos de posicionamento

- relative
 - Mantém espaço reservado no fluxo normal.
 - Permite deslocamento em relação à posição original.

```
.box {  
  position: relative;  
  top: 10px;  
  left: 20px;  
}
```

Tipos de posicionamento

- absolute
 - Sai do fluxo normal.
 - Posiciona-se em relação ao primeiro ancestral posicionado (não-static).

```
.box {  
  position: absolute;  
  top: 0;  
  right: 0;  
}
```

Tipos de posicionamento

- fixed
 - Fica fixo em relação à viewport.
 - Não se move ao rolar a página.

```
.menu {  
    position: fixed;  
    bottom: 0;  
    width: 100%;  
}
```

Tipos de posicionamento

- sticky
 - Mistura de relative + fixed.
 - "Gruda" em uma posição definida quando o scroll atinge um ponto.

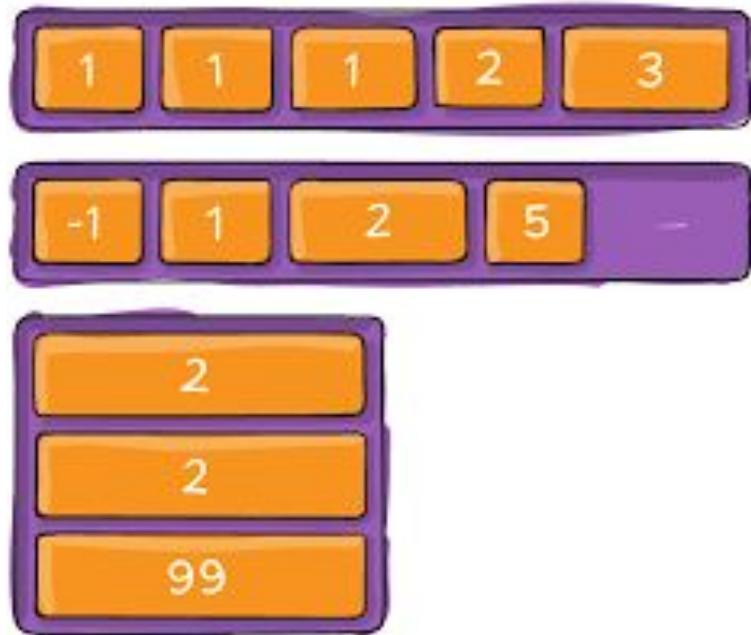
```
.header {  
    position: sticky;  
    top: 0;  
    background: white;  
}
```

Tipos de posicionamento

- Outras propriedades importantes
 - z-index: controla sobreposição de camadas.
 - clip-path: permite cortes/recortes em formas.
 - overflow: controla conteúdo que extrapola limites (visible, hidden, scroll, auto).

Flexbox (Flexible Box Layout)

- Conceito
 - O Flexbox organiza elementos em uma dimensão (linha ou coluna). Foi criado para resolver problemas de alinhamento e espaçamento que antes dependiam de float e table.

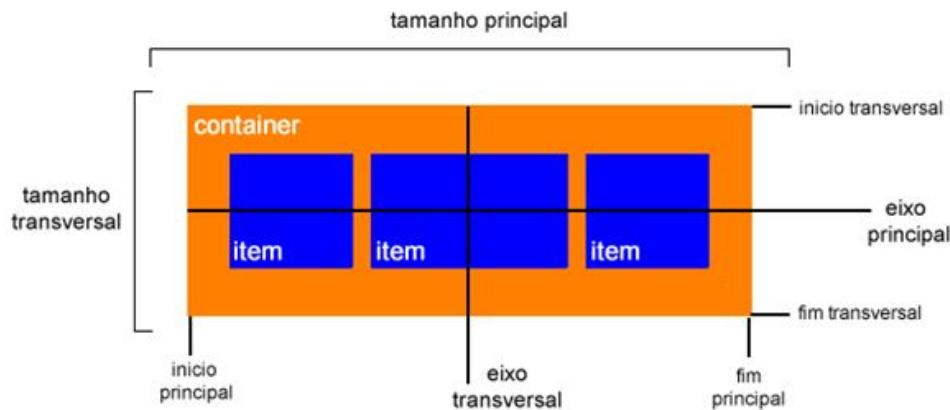


Ativação

```
.container {  
    display: flex; /* ativa o Flexbox */  
}
```

Eixos do Flexbox

- Eixo principal (main axis): definido por flex-direction.
- Eixo cruzado (cross axis): perpendicular ao eixo principal.



Propriedades do container

- **flex-direction:** define direção dos itens.
 - row | row-reverse | column | column-reverse.
- **justify-content:** alinha itens no eixo principal.
 - flex-start, flex-end, center, space-between, space-around, space-evenly.

Propriedades do container

- **align-items:** alinha itens no eixo cruzado.
 - stretch, flex-start, flex-end, center, baseline.
- **flex-wrap:** controla quebra de linha.
 - nowrap (padrão), wrap, wrap-reverse.
- **align-content:** organiza linhas múltiplas de itens.

Propriedades dos itens

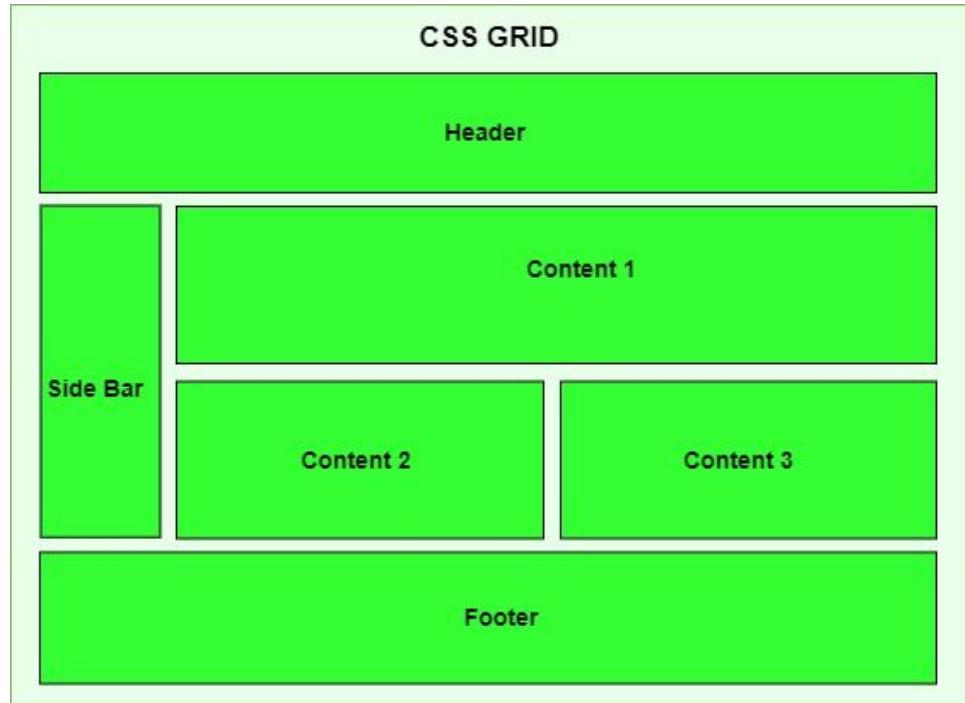
- **flex-grow:** fator de crescimento (quanto expande).
- **flex-shrink:** fator de contração (quanto encolhe).
- **flex-basis:** tamanho inicial do item.
- **order:** altera ordem visual.
- **align-self:** sobrescreve align-items para um item específico.

Exemplo

```
.container {  
    display: flex;  
    justify-content: space-between;  
    align-items: center;  
}  
  
.item1 { flex: 1; } /* ocupa proporcionalmente */  
.item2 { flex: 2; } /* ocupa o dobro do espaço */
```

Grid Layout

- Conceito
 - O CSS Grid organiza elementos em duas dimensões (linhas e colunas).
 - É ideal para layouts de páginas completas.



Ativação

```
.container {  
    display: grid;  
    grid-template-columns: 200px 1fr 1fr;  
    grid-template-rows: 100px auto 80px;  
}
```

Propriedades do container

- grid-template-columns e grid-template-rows: definem estrutura.
 - Valores: px, %, fr (frações).
- gap: espaço entre células (row-gap, column-gap).
- justify-items e align-items: alinham itens dentro da célula.
- grid-template-areas: cria layouts semânticos.

Exemplo

```
.container {  
    display: grid;  
    grid-template-areas:  
        "header header"  
        "sidebar main"  
        "footer footer";  
}  
  
.header { grid-area: header; }  
.sidebar { grid-area: sidebar; }  
.main { grid-area: main; }  
.footer { grid-area: footer; }
```

Propriedades dos itens

- grid-column: 1 / 3; → ocupa colunas 1 a 3.
- grid-row: 2 / 4; → ocupa linhas 2 a 4.
- grid-area: associa item a área nomeada.

Comparação Flexbox x Grid

- A diferença principal é que o Flexbox é unidimensional (foca em uma linha ou coluna), ideal para alinhar e distribuir itens em um único eixo, enquanto o Grid Layout é bidimensional (foca em linhas e colunas), perfeito para criar a estrutura geral de uma página com layouts mais complexos e intersecções de linhas e colunas.
- Ambas as ferramentas são complementares e podem ser usadas juntas para criar designs responsivos.

Comparação Flexbox x Grid

Critério	Flexbox	Grid
Dimensão	1D (linha ou coluna)	2D (linha e coluna)
Melhor uso	Menus, barras, componentes	Layouts completos, dashboards
Controle	Espaçamento e alinhamento	Estrutura de página
Sintaxe	Mais simples	Mais detalhada

Boas práticas

- Use Flexbox para componentes pequenos e alinhamentos.
- Use Grid para estrutura geral da página.
- Combine ambos: Grid para layout global + Flexbox dentro das áreas.
- Prefira fr no Grid (frações) em vez de px fixo para layouts responsivos.
- Teste minmax() no Grid para controle flexível de colunas.



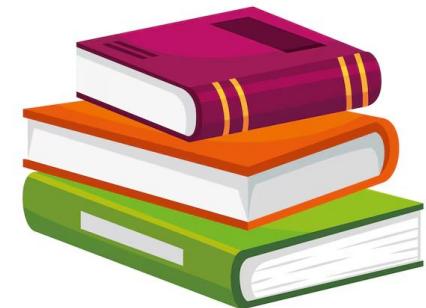
Bibliografia Básica

- LUCKOW, Décio Heinzelmann; MELO, Alexandre Altair de. Programação Java para a Web. São Paulo, SP: Novatec, 2010. 638 p. ISBN 9788575222386.
- GEARY, David; HORSTMANN, Cay. Core JavaServer Faces. 3. ed. Rio de Janeiro, RJ: Alta Books, 2012: ISBN: 9788576086420.
- SILVA, Maurício Samy. HTML 5: a linguagem de marcação que revolucionou a web. 2. ed. São Paulo: Novatec, 2014. 335 p. ISBN 9788575224038.
- FLANAGAN, David. JAVASCRIPT – O Guia Definitivo. Bookman, 6^a ED./2012, 856583719x/9788565837194.



Bibliografia Complementar

- KURNIAWAN, Budi. Java para a Web com Servlets, JSP e EJB: Budi Kurniawan; tradução Savannah Hartmann; revisão técnica Alfredo Dias da Cunha Júnior. Rio de Janeiro, RJ: Ciência Moderna, 2002. xxiv, 807 p. ISBN 8573932104 (broch.).
- FREEMAN, Elisabeth; FREEMAN, Eric. Use a cabeça!: HTML com CCS e XHTML. 2. ed. Rio de Janeiro, RJ: Alta Books, 2008. xxxi, 580 p. ISBN 9788576082187 (broch.).
- URUBATAN, Rodrigo. Ruby on rails: desenvolvimento fácil e rápido de aplicações Web. São Paulo, SP: Novatec, 2009. 285 p. ISBN 9788575221846 (broch.).
- GONÇALVES, Edson. Desenvolvendo aplicações Web com NetBeans IDE 6. Rio de Janeiro: Ciência Moderna, 2008. 581 p. : CD-ROM ISBN 97885739366742.
- BASHAM, Bryan. Use a cabeça!: Servlets & JSP. 2. ed. Rio de Janeiro, RJ: Alta Books, 2008. ISBN 9788576082941.



Obrigado!

Dúvidas?



Universidade Federal do Ceará - Campus Quixadá

Prof. Francisco Victor da Silva Pinheiro
victorpinheiro@ufc.br

