Interfaces dinamicamente adaptáveis

Layout baseado em proporções, layout responsivo, formatação condicional com *media queries*

Desenvolvimento Web - Prof. Eduardo Mangeli

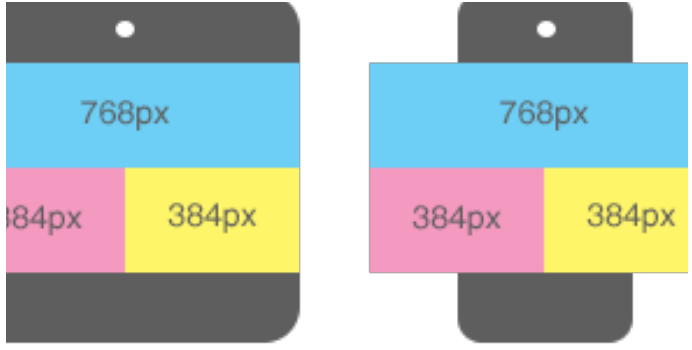
Entendendo o desafio

- A web não é feita “apenas” de telas
- Telas não têm o mesmo tamanho
- O usuário pode não usar toda a tela
- Os elementos não devem mudar de tamanho de forma linear indiscriminadamente
  - Um botão pode precisar ser proporcionalmente maior em telas menores

Enfrentando o problema

Layout fixo

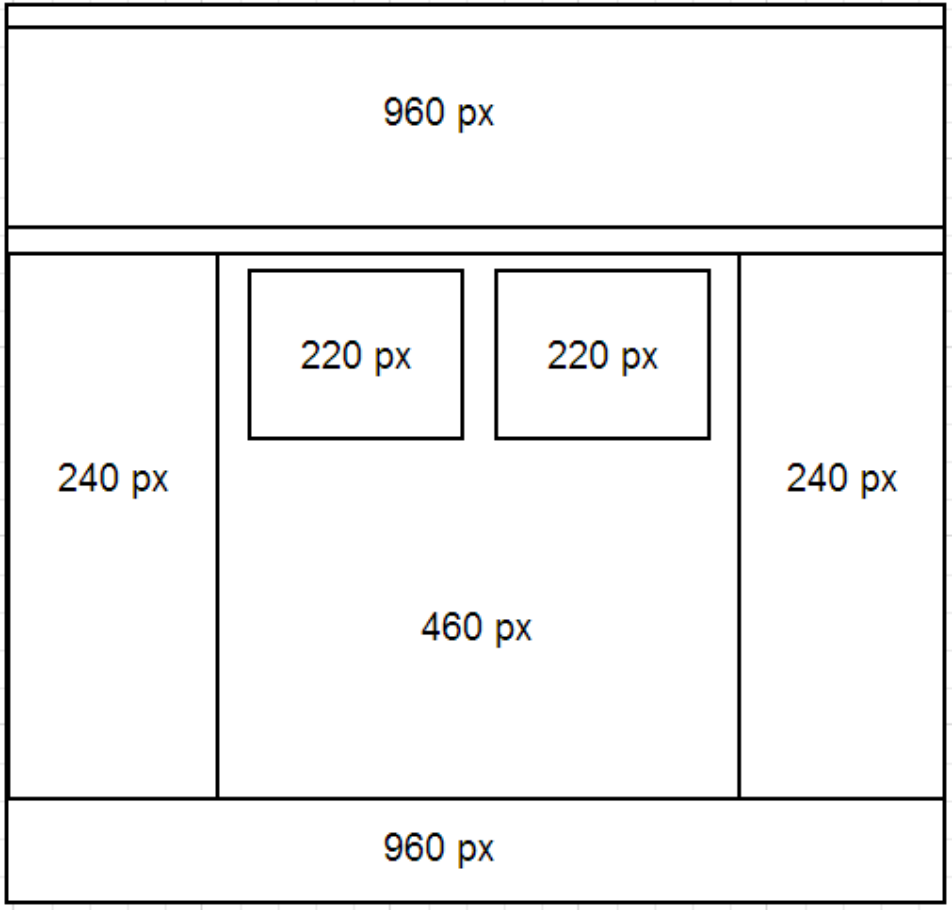
- Estipula uma largura fixa para a página
- Força o tamanho dos elementos
- Não aproveita a tela
- Pode deixar elementos ilegíveis
- Altamente inadequado para dispositivos móveis

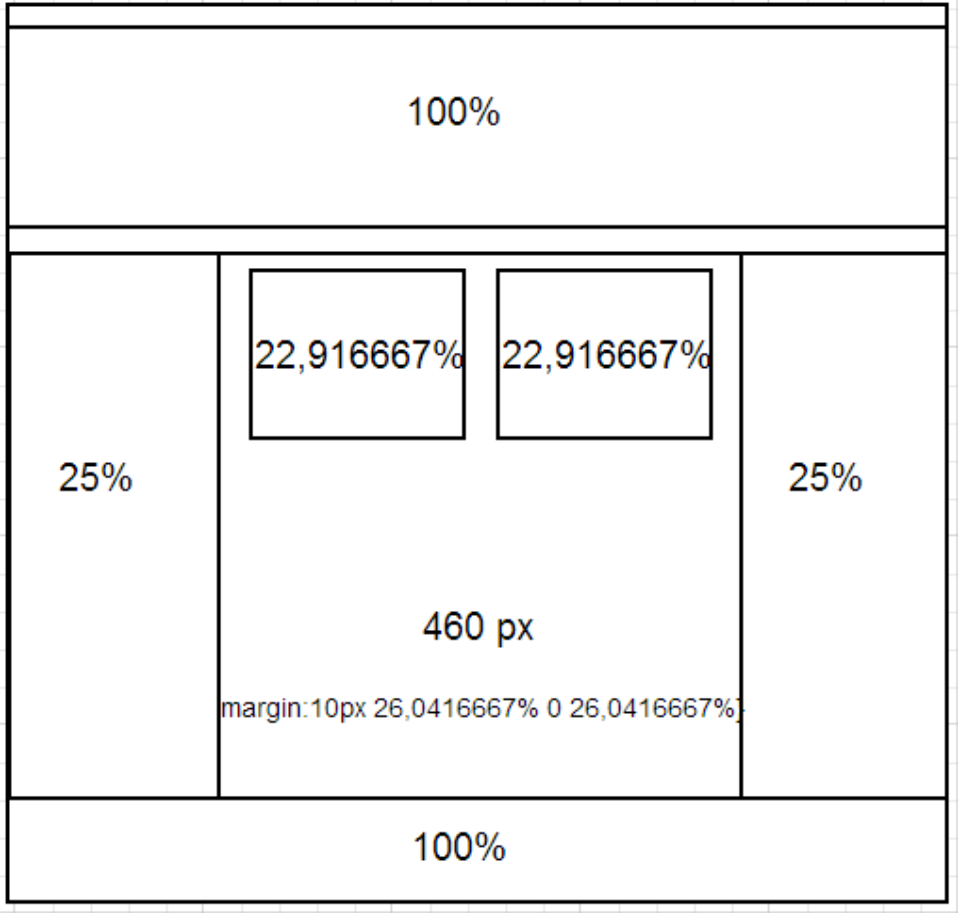


Layout proporcional

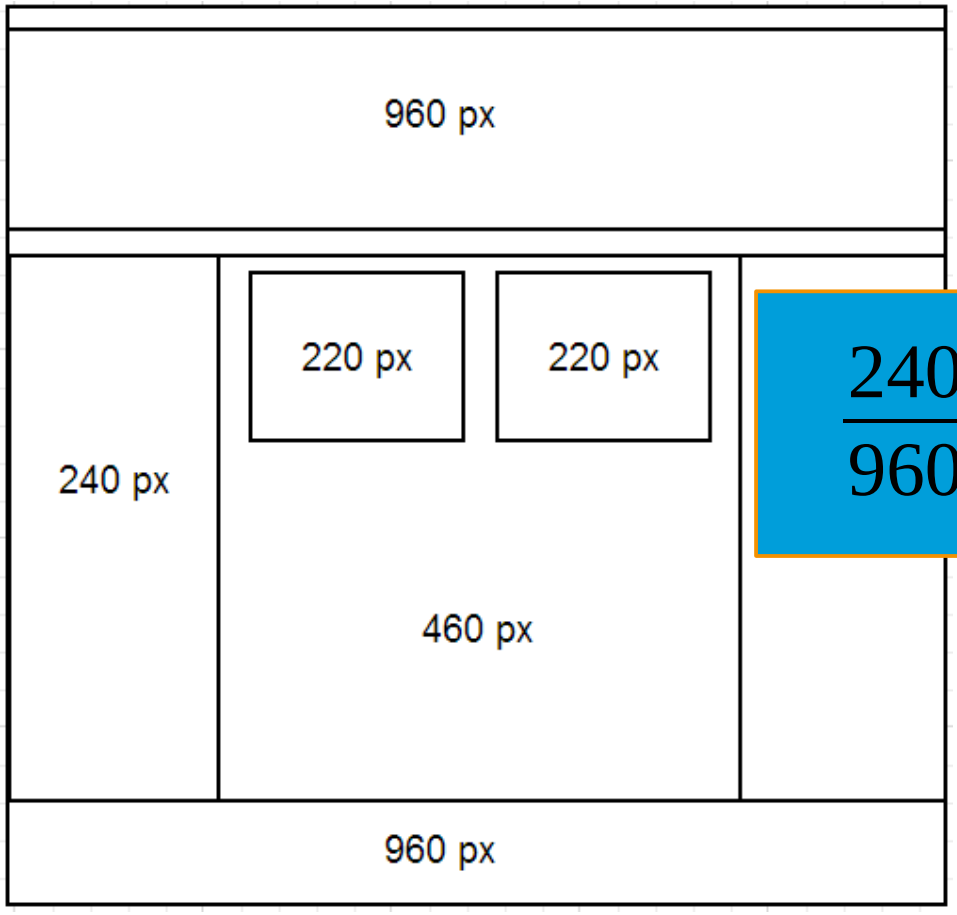
- Usa unidades de medida proporcionais
  - percentuais, em, rem, vw, vh
- Uma resposta a tendência anterior de design fixo

Convertendo Layout Fixo para Proporcional

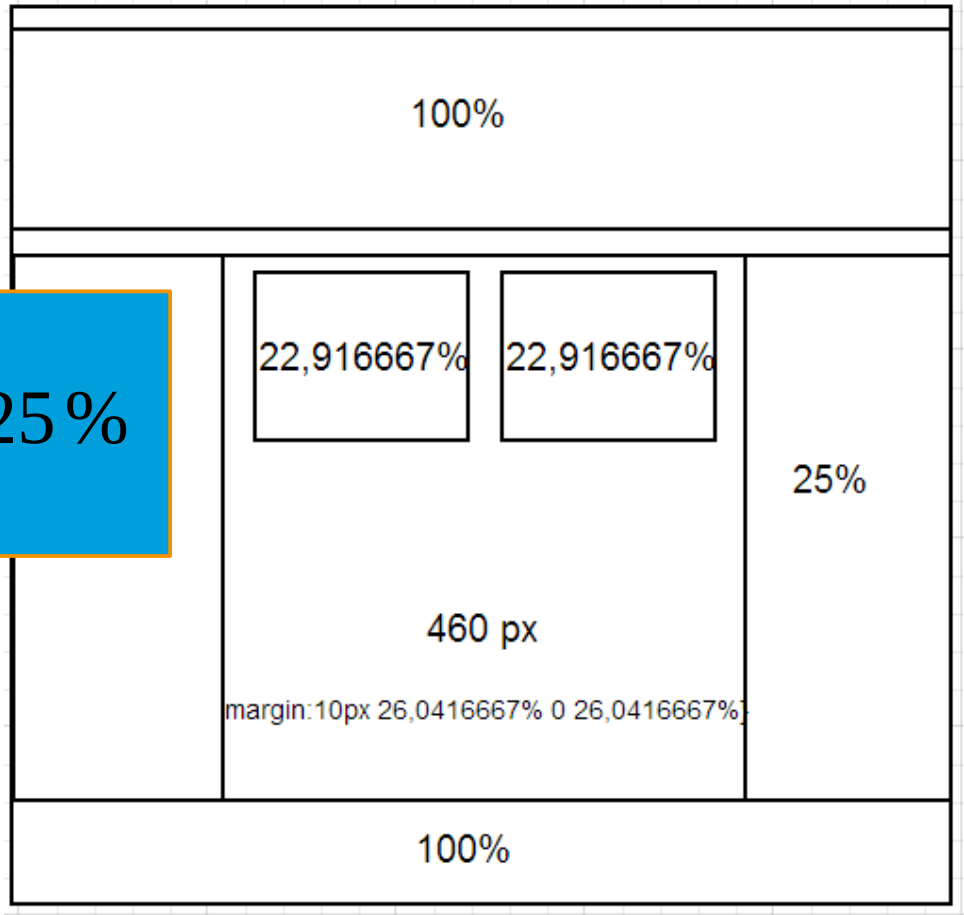




Convertendo Layout Fixo para Proporcional

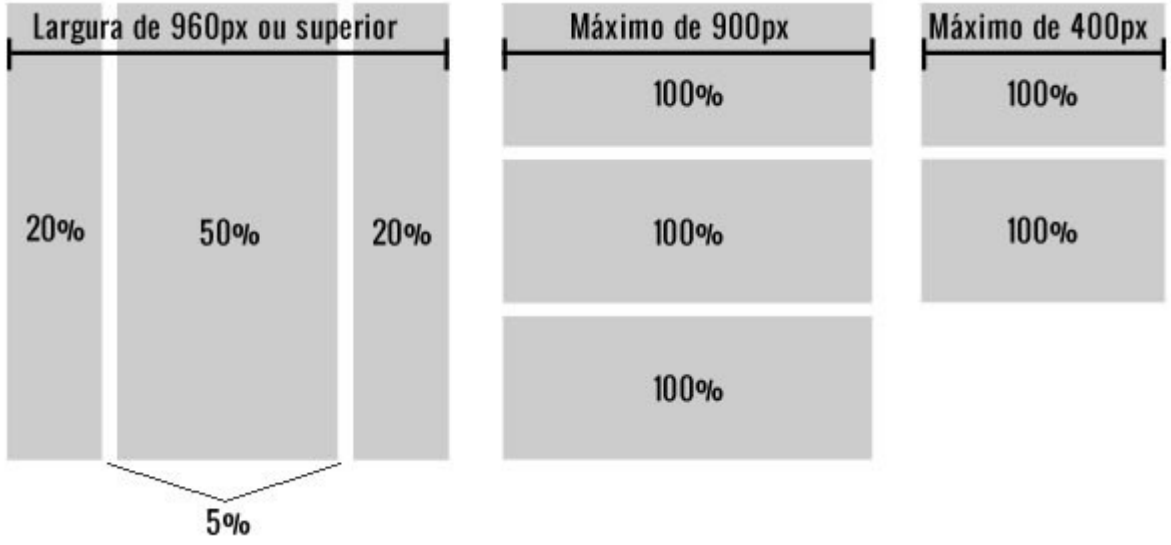


$$\frac{240\ px}{960\ px} = 25\%$$



Layout Fluido

- Elementos se adaptam ao espaço disponível
- Layout se adequa o tamanho da tela
- Disposição dos elementos pode mudar



Estratégias de enfrentamento

- Layout Responsivo
  - Consolida técnicas de layout fluido
    - grids fluidas
    - imagens fluidas (max-width: 100%)
    - media queries
- Mobile First
  - Considera como tela mais importante a dos dispositivos móveis
  - Quebra de paradigma do desenvolvimento feito no desktop

Como implementar

Media Queries

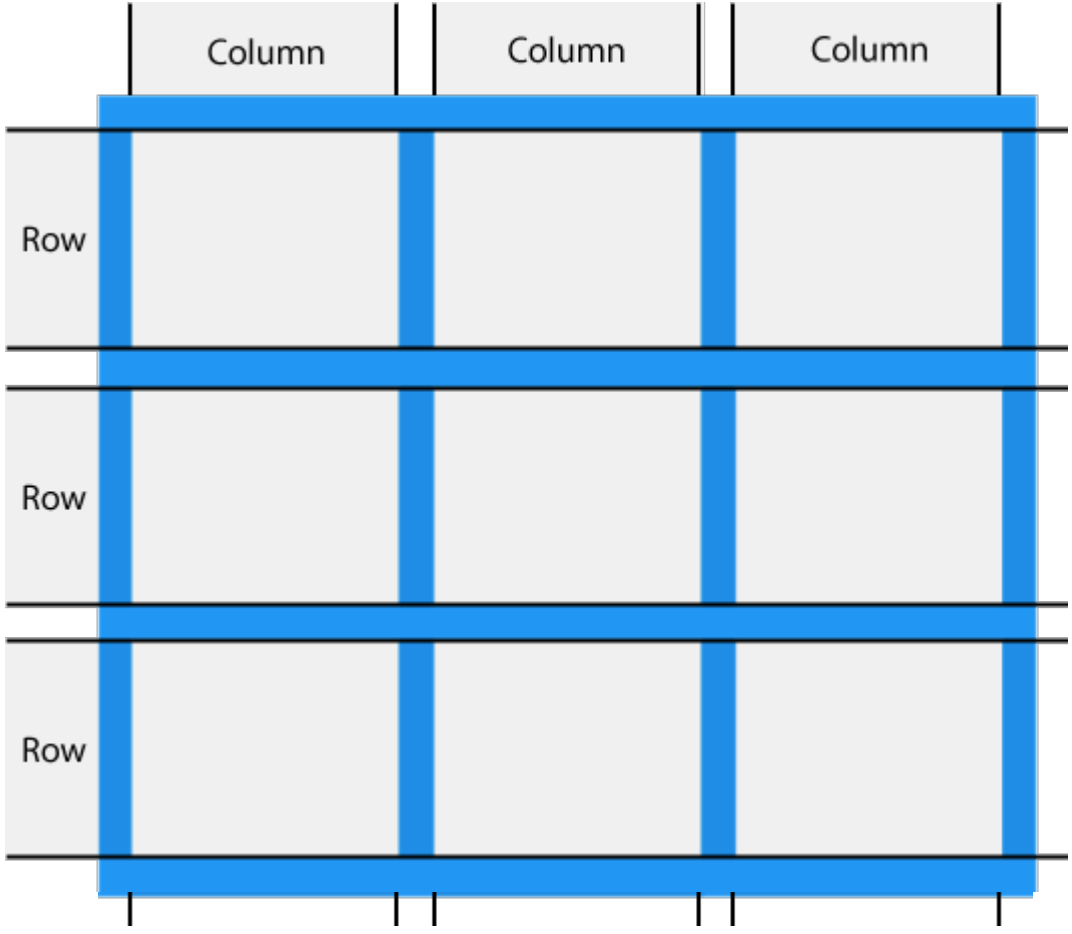
- Aplicação condicional de regras CSS
- Regra @media
- Tipos de mídia: screen, print, all
- Features de Mídia: width, resolution, pointer etc
- Operadores lógicos: and, not, only, vírgula (,)

Exemplos de *media queries*

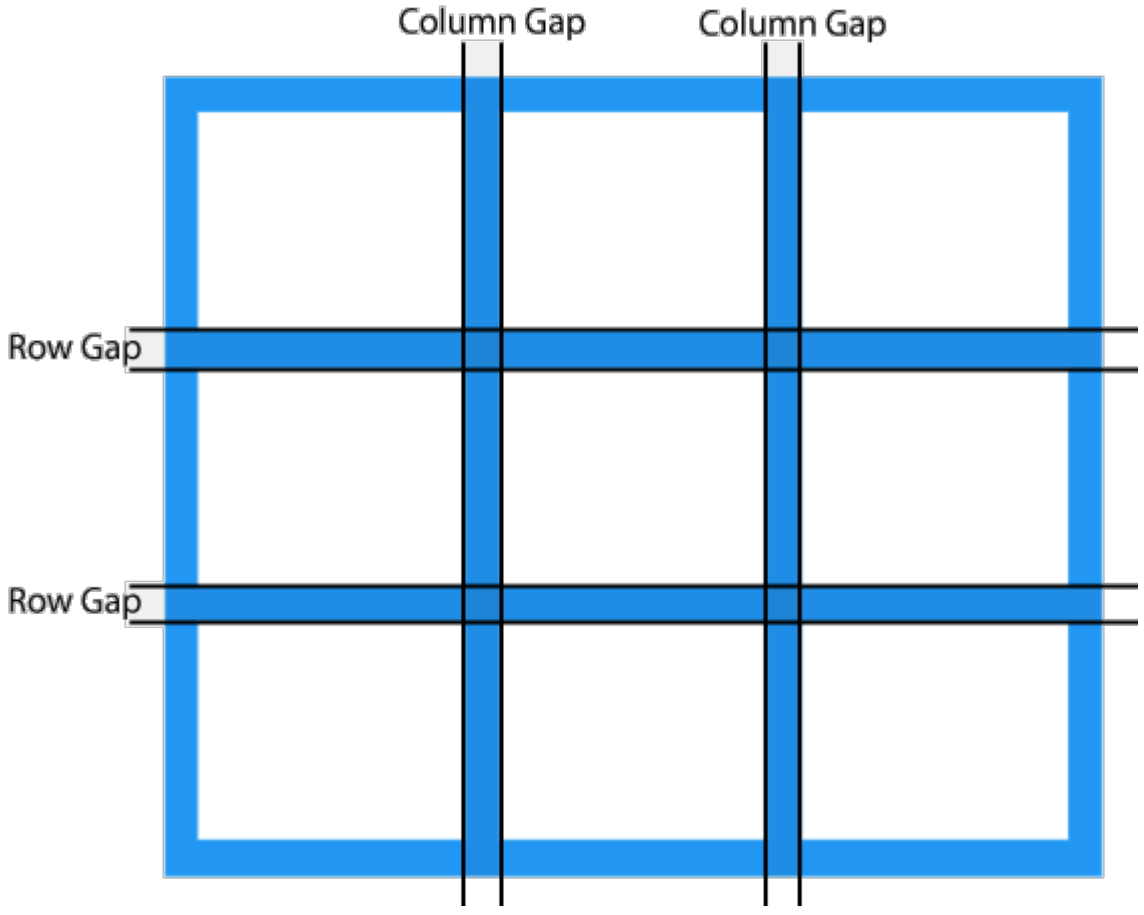
```
@media print {
  body { font-size: 10pt; }
}

@media screen {
  body { font-size: 13px; }
}

@media screen, print {
  body { line-height: 1.2; }
}

@media only screen
  and (min-width: 320px)
  and (max-width: 480px)
  and (resolution: 150dpi) {
    body { line-height: 1.4; }
}
```

CSS Grid Layout

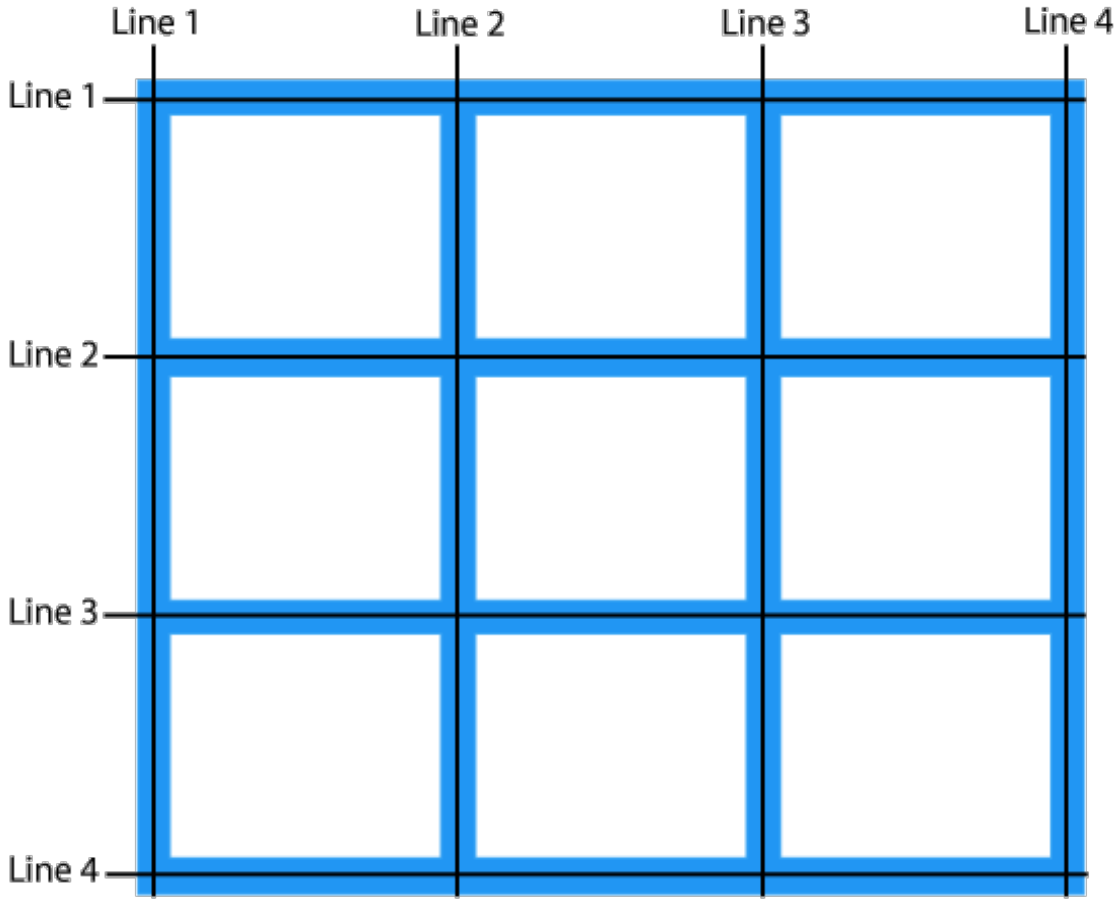
CSS Grid Layout



Grid Layout *gaps*



Grid Lines



CSS Grid Exemple

```
.grid-container {
  display: grid;
  gap: 50px 100px;
}
.item1 {
  grid-column-start: 1;
  grid-column-end: 3;
}
.item3{
 grid-column: 1 / 4;
}
```

https://www.w3schools.com/css/tryit.asp?filename=trycss_grid_lines

Vamos praticar !