

Dell IT Academy

HTML e CSS

Instrutor: Júlio Pereira Machado (julio.machado@pucrs.br)



Cascading Style Sheets



Cascading Style Sheets (CSS)

- **Cascading Style Sheets (CSS)** é uma linguagem que define como os elementos HTML são estilizados
 - Em outras palavras, HTML estrutura um documento, enquanto CSS o formata
- CSS pode ser aplicado a elementos HTML individuais (diretamente), ou pode ser mantido em um arquivo separado e aplicado através de referências
 - Arquivos que contêm CSSs são chamados de “folhas de estilos” e usam a extensão .css
- Ao contrário do HTML, CSS utiliza **regras** ao invés de tags

example.css

```
h1, h2, h3 {  
    font-family:  
    Arial;  
    color: yellow;  
}  
  
p {  
    font-family:  
    Arial;  
    font-size: 12px;  
}
```

CSS3

- CSS3 é a versão do CSS que foi definida em conjunto com HTML5
- CSS3 é compatível com versões anteriores
- CSS3 apresenta um conjunto de novos efeitos, incluindo transformações 2D e 3D, bem como animações

Prefixos Específicos

- Assim como HTML5, CSS3 pode não ser compatível com todos navegadores
- Novas propriedades são adicionadas o tempo todo, enquanto outras são modificadas
- Muitos navegadores oferecem propriedades alternativas
- O nome dessas propriedades deve aparecer com um prefixo específico
- Um prefixo específico é simplesmente uma palavra-chave circundada por traços

NAVEGADOR	PREFIXO
Internet Explorer, Edge (antigo)	-ms-
Edge	-webkit-
Firefox	-moz-
Opera (antigo)	-o-
Chrome	-webkit-
Safari	-webkit-

Ligando CSS com HTML

- Existem vários modos de incorporar CSS ao HTML:

1. Estilo *inline*
2. Use do elemento `<style>` aninhado ao elemento `<head>`
3. Elemento `<link>` referenciando arquivo CSS separado

- Colocar CSS em um arquivo separado possui vantagens:

- Troca do estilo permite alterar um documento inteiro
- Times podem trabalhar em responsabilidades separadas: conteúdo e design visual

1 `<p style="color: red">`

2 `<style>
 h1 {
 font-family: 'Segoe
UI';
 color: #808080;
 }
</style>`

3 `<link href="StyleSheet.css"
 rel="stylesheet"
 type="text/css">`

Como ligar HTML ao arquivo CSS?

- Arquivos CSS são ligados ao arquivo HTML via elemento **<link>**
- O atributo **href** aponta para o local do arquivo CSS
 - É extremamente importante que o nome e localização do arquivo estejam corretos ou nenhum estilo será aplicado
- O atributo **rel** deve ser configurado com "stylesheet", enquanto o atributo **type** deve ser configurado com "text/css"
 - O atributo type é opcional e pode ser entendido pelo contexto

```
<link href="StyleSheet.css" rel="stylesheet" type="text/css">
```

Ordem de Aplicação (“cascata”)

- Geralmente:

- Estilo in-line (dentro do elemento HTML)
- Folhas de estilo internas ou externas (indicadas no elemento head)
- Valor padrão do navegador

Valores

Cada propriedade de um elemento HTML possui diferentes tipos de valores

Exemplos:

- https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Values_and_units

Seletores e Declarações



Seletores e Declarações

- Existem duas partes de uma regra CSS:
 - seletores
 - declarações
- Um **seletor** referencia o elemento HTML a ser estilizado
- Uma **declaração** é o estilo que se deseja aplicar ao elemento
 - declarações possuem duas partes: uma propriedade seguida por dois-pontos (:), um espaço, e um valor seguido por um ponto-e-vírgula (;)
 - declarações aparecem entre chaves { }



Seletores e Declarações

- Elementos HTML podem ser referenciados por seletores de diversas maneiras, incluindo:
 - Nome da tag, tal como `p`, `h1`, `table`, etc.
 - Seletor de id, tal como `#navbar`, o qual inclui o símbolo de hashtag (`#`) como prefixo
 - Seletor de classe, como `.happy`, o qual inclui o ponto (`.`) como prefixo
- `id` e `class` são ambos atributos universais do HTML
 - `id` é utilizado para identificar elementos de forma única
 - `class` deve ser utilizado para categorizar elementos em grupos que serão estilizados de forma semelhante

nome da tag

```
h1 {  
  color: red;  
  font-family: sans-serif;  
  text-align: left;  
}
```

seletor id

```
#navbar {  
  background-color: green;  
}
```

seletor class

```
.happy {  
  font-size: 14px;  
}
```

Seletores e Declarações

- Nome do elemento e class podem ser combinados no selector
- Lista de seletores podem ser informados

nome e classe

```
p.center {  
  text-align: center;  
}
```

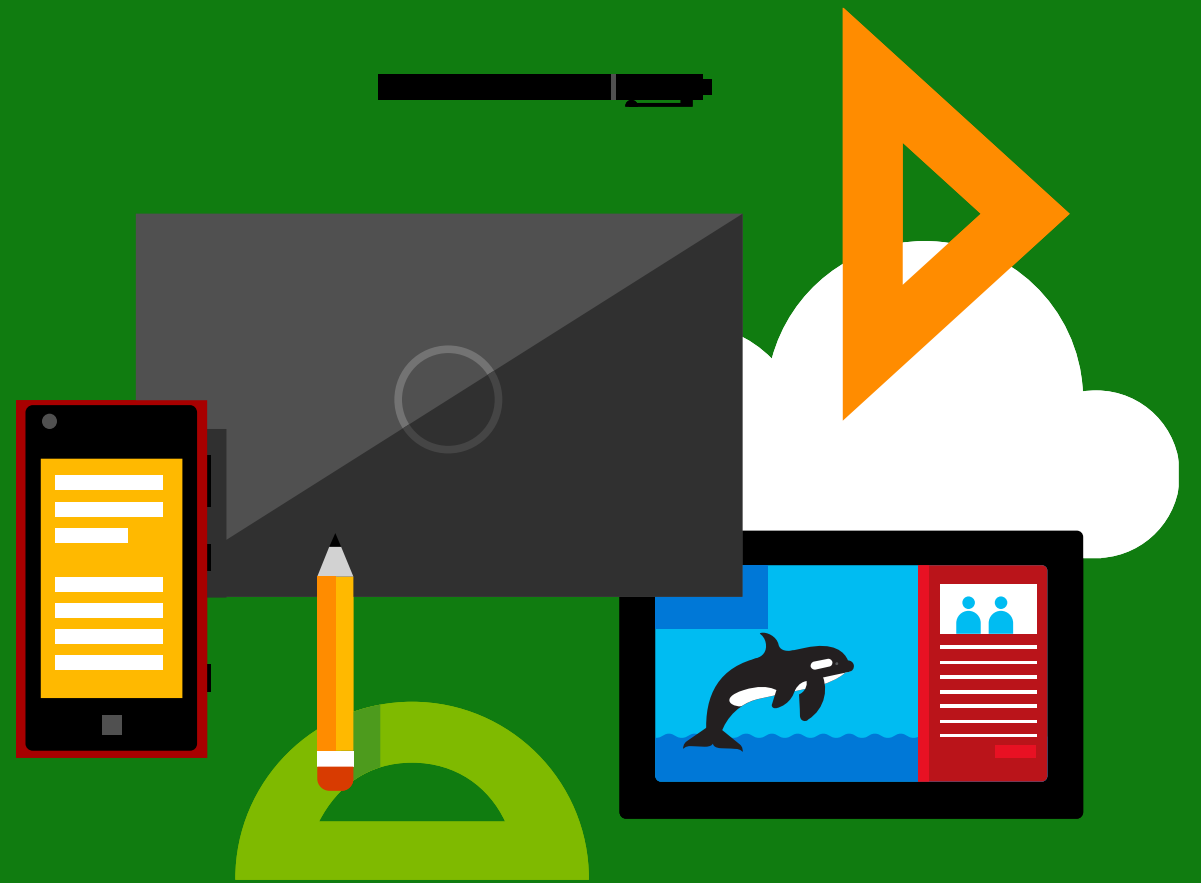
agrupamento

```
p,h1,h2 {  
  text-align: center;  
}
```

Seletores e Declarações

- O conjunto de opções de seletores é bastante extensa e permite obter qualquer elemento da árvore DOM do HTML de formas alternativas
- Especialmente as pseudo-classes que permite selecionar elementos em função do estado atual e os pseudo-elementos que permitem selecionar partes de um elemento
- Consulte a lista em https://www.w3schools.com/cssref/css_selectors.asp

Fontes e Famílias de Fontes



Fontes e Famílias de Fontes

- Uma fonte é um conjunto de caracteres com um estilo e tamanho particulares
- O principal meio de especificar fontes em CSS é utilizando a propriedade **font-family**
- Os três tipos mais comuns de fontes são **serif**, **sans serif**, e **monospace**

Tipo de Fonte	Exemplo	Descrição
Serif	Times New Roman	possui decoração ao final de alguns caracteres
Sans Serif	Arial	não possui decoração ao final dos caracteres
Monospace	Courier	cada letra ocupa o mesmo espaço de tela

Regra **@font-face**

- Anterior ao CSS3, desenvolvedores tinham que utilizar somente fontes disponíveis no dispositivo do usuário
 - Estas fontes são conhecidas como “web-safe”
- Essa característica era um grande limitador
- Regra **@font-face** do CSS3 permite que um desenvolvedor utilize qualquer fonte desde que ela esteja disponível no servidor web

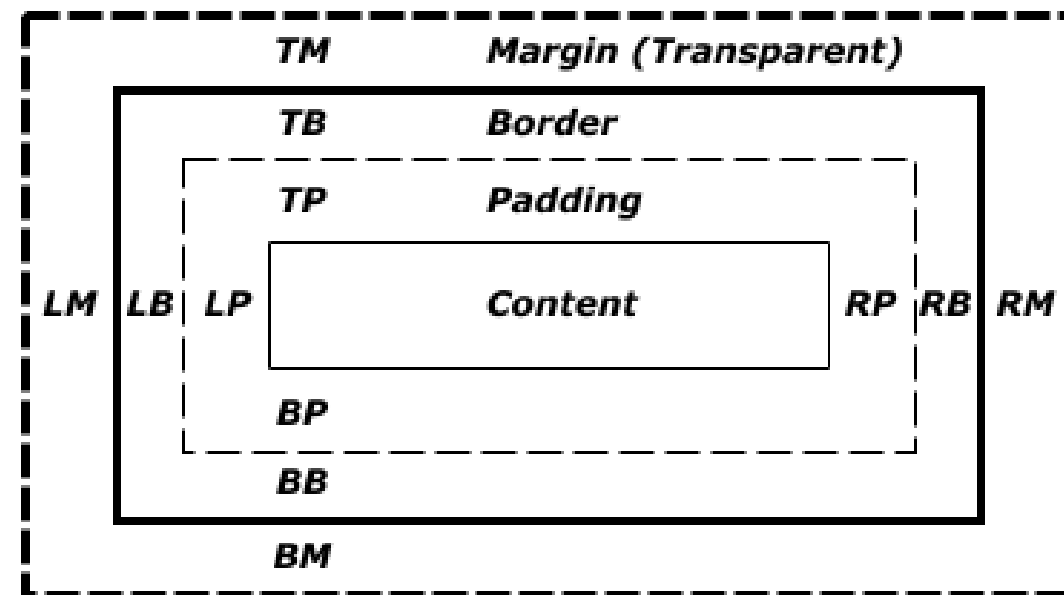
```
@font-face {  
    font-family: "font-family-name";  
    src: url("http://website/fonts/fontfile")  
}
```

CSS Box Model



CSS Box Model

- O **CSS Box Model** define as regras de como um conteúdo é formatado dentro de uma página web
- Cada elemento do HTML está dentro de uma caixa com vários componentes, incluindo padding, border, e margin
 - **Padding** é o espaço entre o conteúdo e sua borda (é transparente)
 - **Border** é uma borda que envelopa a área representada pelo padding
 - **Margin** representa a margem que é o espaço entre a área da borda e outra caixa de outro elemento HTML



- Margin edge
- Border edge
- - - Padding edge
- Content edge

Elementos de Bloco e Inline

- Com o CSS Box Model, existem duas categorias de elementos do HTML
 - block-level
 - inline
- **Block-level** sempre iniciam em nova linha e ocupam todo espaço horizontal
- **Inline** não iniciam uma nova linha e ocupam somente o espaço horizontal necessário

BLOCK-LEVEL EXEMPLOS

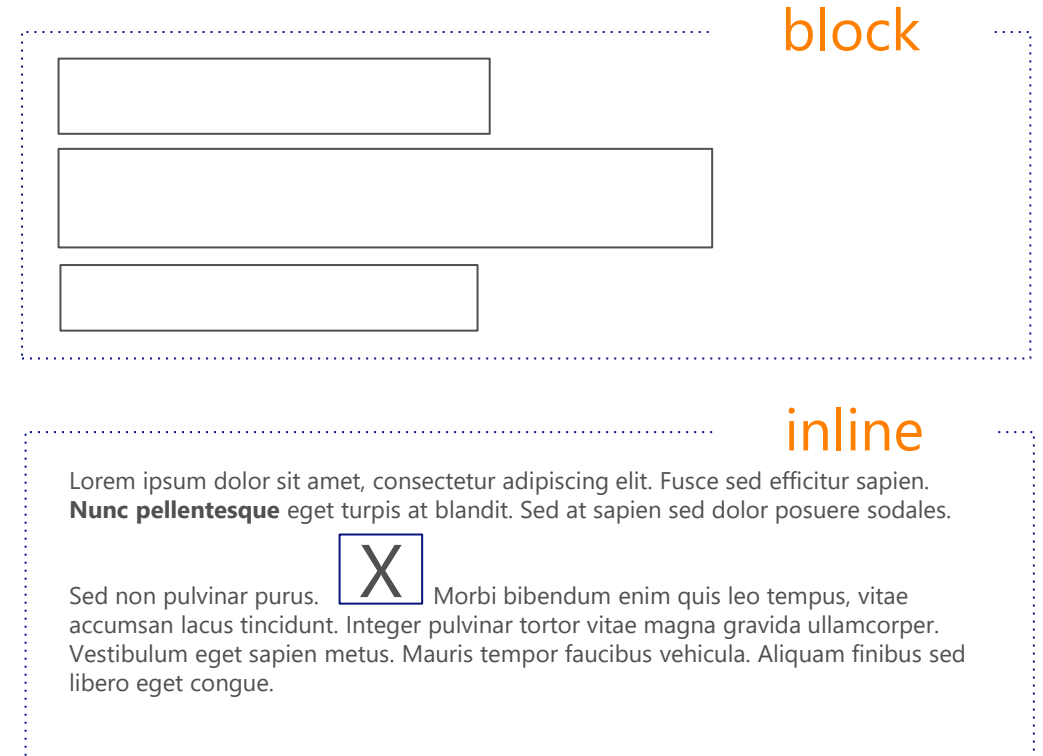
- `<div>`
- `<p>`
- ``, ``, `<dl>`
- `<h1>` - `<h6>`
- `<form>`

INLINE EXEMPLOS

- ``
- `<a>`
- ``
- ``
- `<input>`

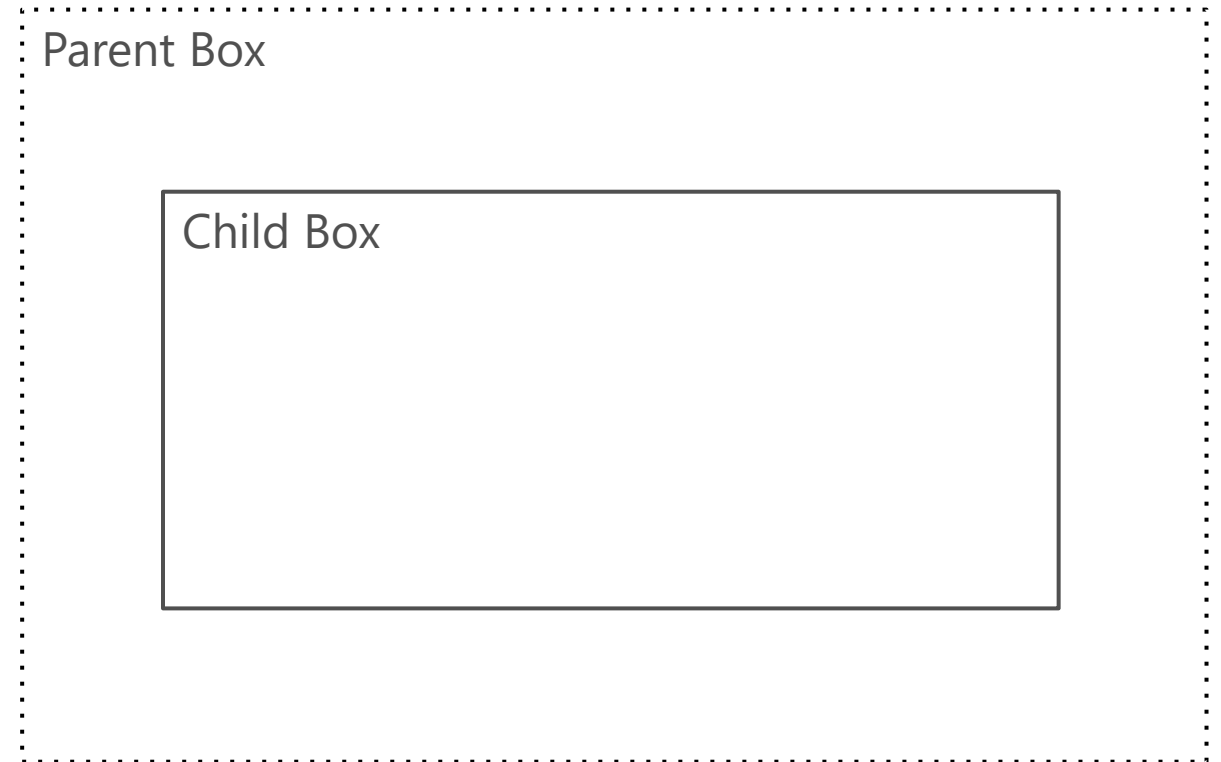
Fluxo de Conteúdo

- **Flow** é um estilo de apresentação do HTML focado em preencher conteúdo em uma página horizontalmente em linhas de cima para baixo
- Existem dois tipos básicos de fluxo para a propriedade **display**:
 - **Block**
 - **Inline**
- No fluxo **block**, o conteúdo é colocado em sua própria linha, acima ou abaixo de outro conteúdo
 - Ele preenche toda linha da esquerda para direita
- No fluxo **inline**, o conteúdo é colocado na mesma linha de outros conteúdos, antes ou depois deles



Relacionamento Pai/Filho

- Com o CSS Box Model, é possível para uma caixa conter outras caixas
 - A caixa mais externa é o pai
 - A caixa interna é o filho
- Uma caixa filha herda o estilo CSS da caixa pai, o que significa que um estilo aplicado a uma caixa pai será aplicado também ao filho
- Contudo, certas propriedades do CSS não suportam herança!



Posicionando Elementos

- A propriedade **position** permite a colocação de um elemento em uma posição específica do documento
- Os valores utilizados são:

VALOR	DESCRIÇÃO
static	Posiciona o elemento dentro de seu fluxo normal
relative	Posiciona um elemento em relação a onde ele normalmente estaria no fluxo de conteúdo
absolute	Garante que o posicionamento do elemento não impacta os demais
fixed	Posiciona um elemento em relação à janela do navegador de maneira fixa
sticky	Mistura de static com fixed




- Exemplos:
 - https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Positioning

Float



Elementos flutuantes

Float Example



Float positioning

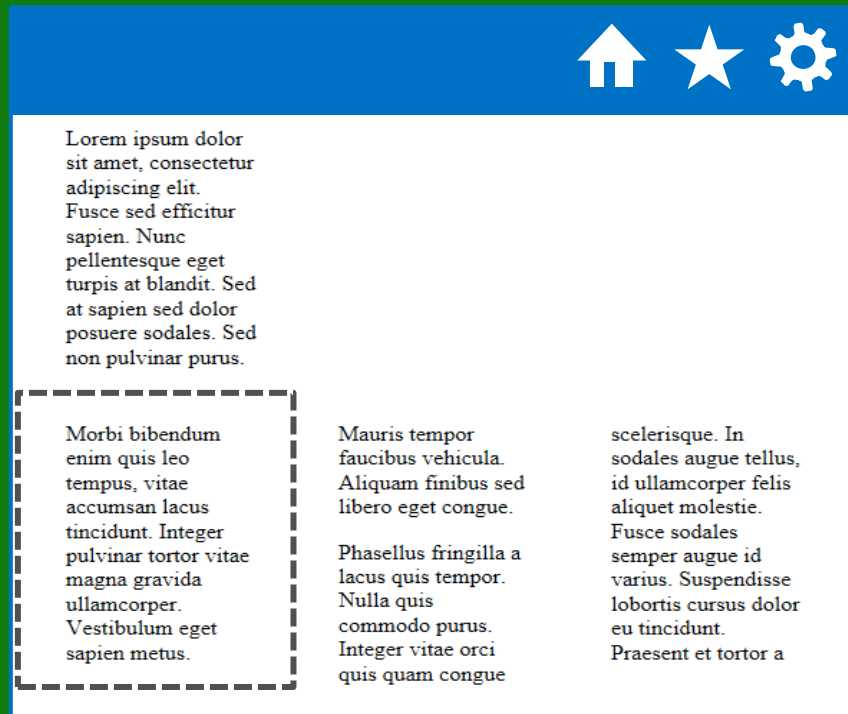
>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sed efficitur sapien. Nunc pellentesque eget turpis at blandit. Sed at sapien sed dolor posuere sodales. Sed non pulvinar purus.

>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sed efficitur sapien. Nunc pellentesque eget turpis at blandit. Sed at sapien sed dolor posuere sodales.

>Lorem ipsum consectetur adipiscing sit amet felis. Integer ultrices viverra velit.

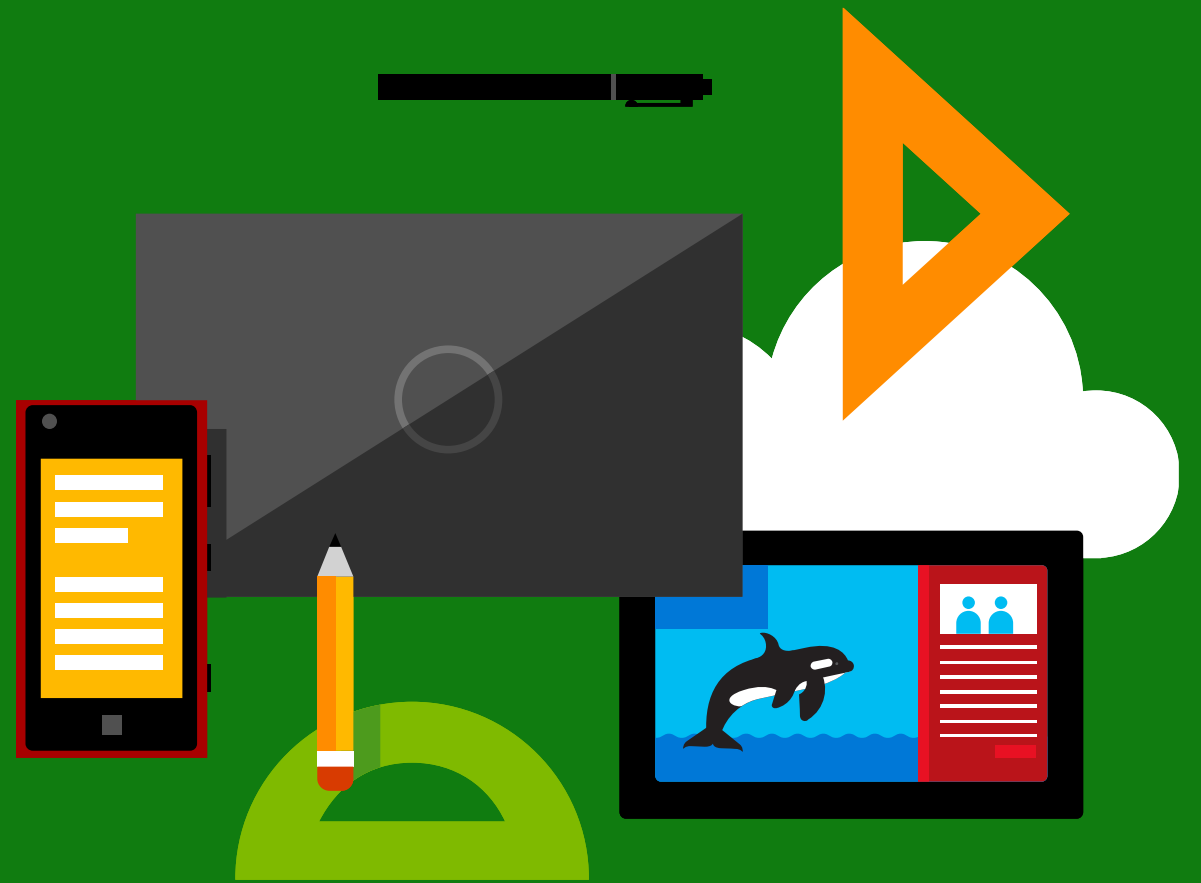
- Em adição à modificação do fluxo de uma página, HTML e CSS permitem que o desenvolvedor posicione elementos HTML de forma individual
- Exemplos:
 - https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Floats

Elementos flutuantes



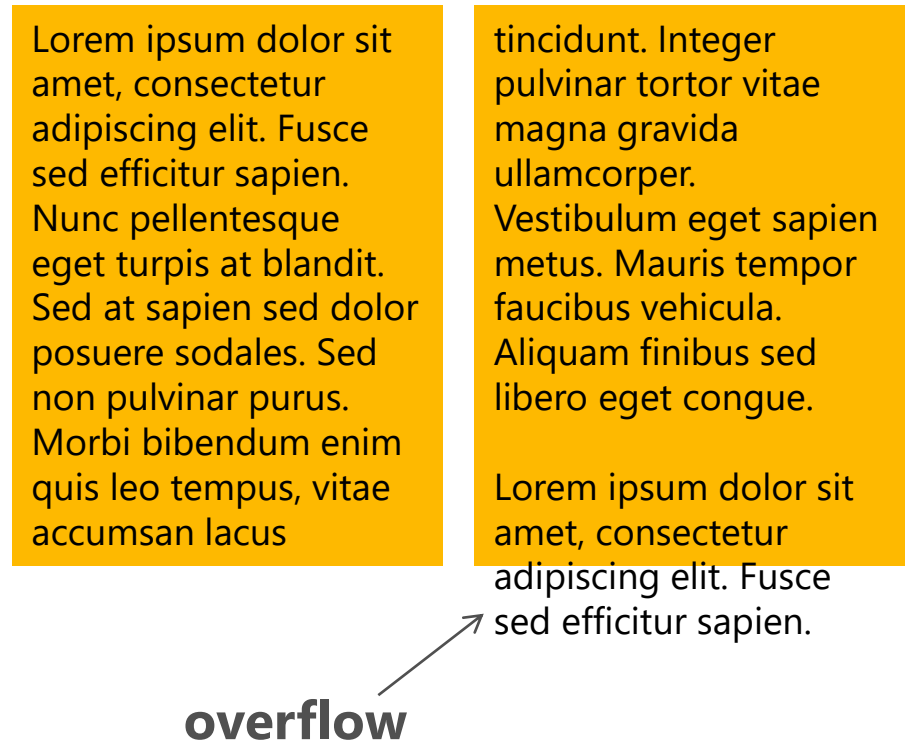
- A propriedade **float** permite mover um elemento totalmente para a esquerda ou direita de uma página
 - Demais conteúdos irão se posicionar no entorno
- O valor de float é comumente configurado em `left` ou `right`
- Use a propriedade **clear** para prevenir que outros elementos flutuantes toquem o lado esquerdo ou direito de um elemento
- O valor de clear pode ser `left`, `right`, `both`, ou `none`
 - `left` deixa limpo o lado esquerdo
 - `right` deixa limpo o lado direito
 - `both` deixa limpo ambos lados
 - `none` permite que elementos se toquem

Transbordo de Conteúdo



Bounding Box

- Cada elemento HTML em uma página ocupa uma área retangular chamada de **bounding box**
- Com CSS, é possível modificar a altura ou largura do bounding box
- Se um elemento não cabe dentro dos limites do bounding box, então diz-se que o conteúdo transbordou - **overflow**
- É possível modificar como o conteúdo que transbordou é estilizado utilizando a propriedade `overflow` no CSS



Scrolling Overflow

- Se o valor da propriedade **overflow** é definido como **scroll**, então todo o conteúdo do elemento permanece dentro dos limites do bounding box
- Isto permite que usuário utilizem uma barra de rolagem



Overflow Visível e Overflow Escondido

- Com valor **visible**, conteúdo transbordado irá aparecer fora dos limites do bounding box e potencialmente sobrepor outros elementos
- Com valor **hidden**, conteúdo transbordado não irá aparecer fora dos limites do bounding box

Visible Overflow

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sed efficitur sapien. Nunc pellentesque eget turpis at blandit. Sed at sapien sed dolor posuere sodales. Sed non pulvinar purus. Morbi bibendum enim quis leo tempus, vitae accumsan lacus tincidunt. Integer pulvinar tortor vitae magna gravida ullamcorper. Vestibulum eget sapien metus. Mauris tempor faucibus vehicula. Aliquam finibus sed libero eget congue.

Phasellus fringilla a lacus quis tempor. Nulla quis commodo purus. Integer vitae orci quis quam congue scelerisque. In sodales augue tellus, id ullamcorper felis aliquet molestie. Fusce sodales semper augue id varius. Suspendisse lobortis cursus dolor eu tincidunt. Praesent et tortor a quam auctor tincidunt non ac odio. In varius, felis et molestie eleifend, ante ipsum ligula vel dui. Sed at nunc cursus ex vel ullamcorper.

Hidden Overflow

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sed efficitur sapien. Nunc pellentesque eget turpis at blandit. Sed at sapien sed dolor posuere sodales. Sed non pulvinar purus. Morbi bibendum enim quis leo tempus, vitae accumsan lacus tincidunt. Integer pulvinar tortor vitae magna gravida ullamcorper. Vestibulum eget sapien metus. Mauris tempor faucibus vehicula. Aliquam finibus sed libero eget congue.

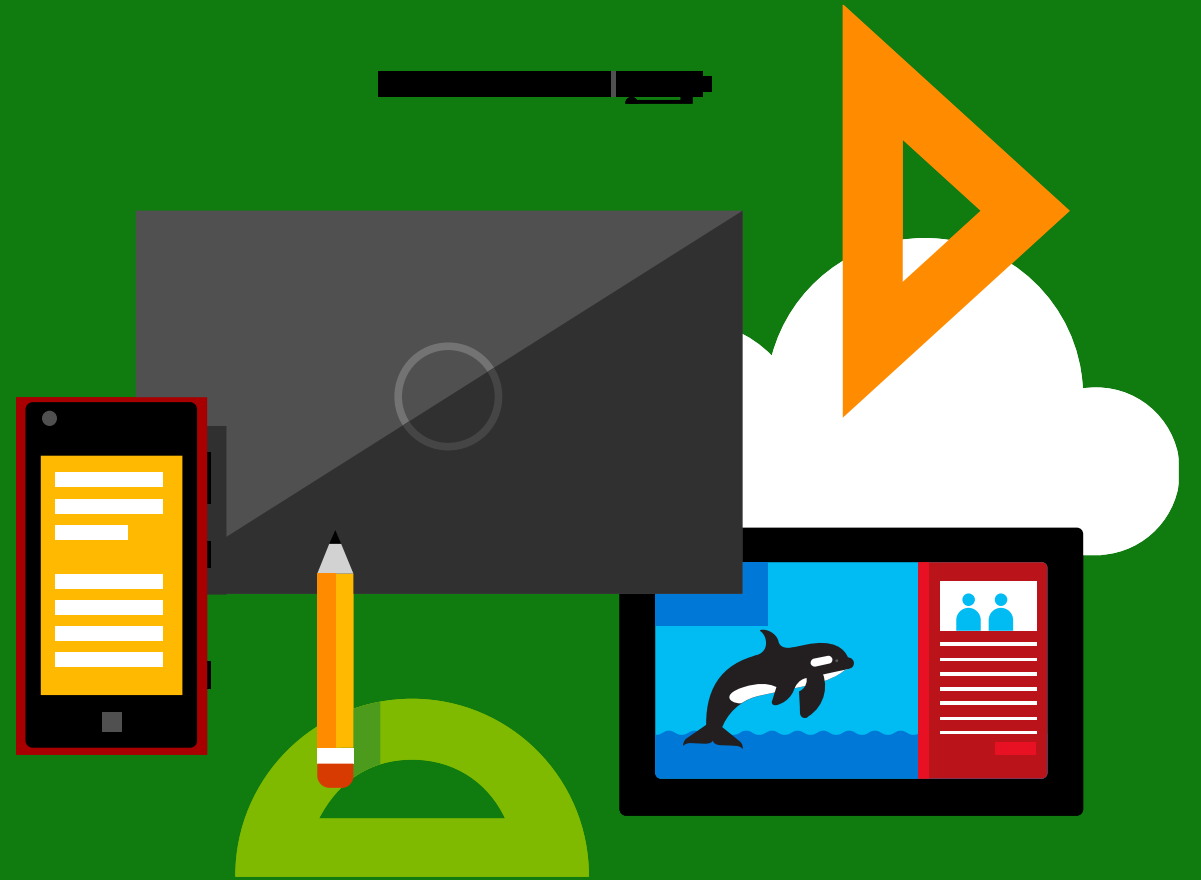
Phasellus fringilla a lacus quis tempor. Nulla quis commodo purus. Integer vitae orci quis quam congue scelerisque. In

Exemplo

```
<body>
  <h1>Overflow Demo</h1>
  <p>Lorem ipsum dolor sit amet,
consectetur adipiscing elit.
Fusce sed efficitur sapien.
Nunc pellentesque eget turpis
at blandit. Sed at sapien sed
dolor posuere sodales. Sed non
pulvinar purus. Morbi bibendum
enim quis leo tempus, vitae
accumsan lacus tincidunt.</p>
</body>
```

```
<style>
  p {
    height: 200px;
    width: 200px;
    background-color: #e1e1e1;
    overflow: scroll | visible | hidden;
  }
</style>
```

Flexbox



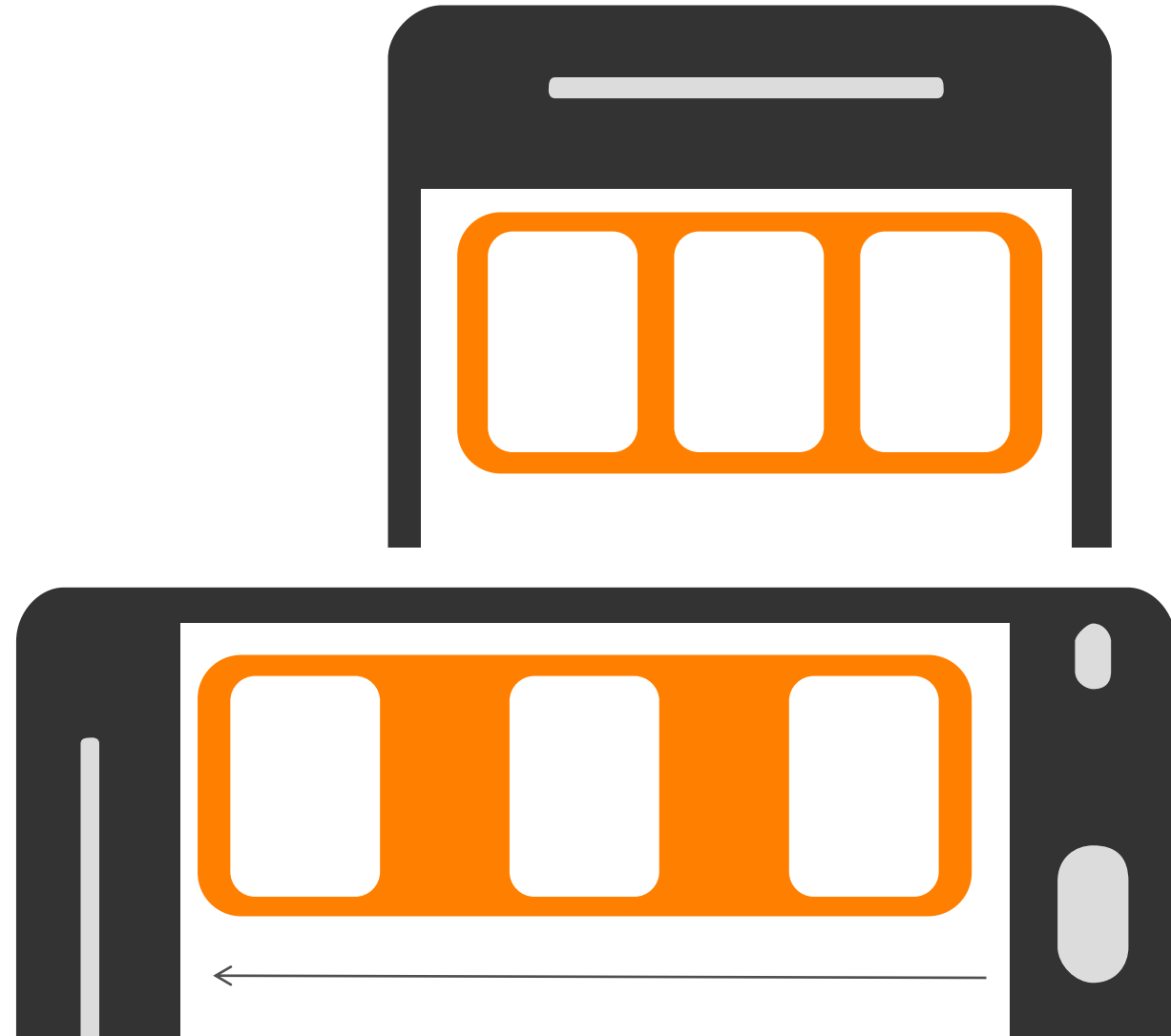
CSS Flexbox Box Model

- CSS3 inclui o **Flexbox Box Model**, que é um modo de layout que provê uma maior flexibilidade quando o usuário altera o tamanho da janela do navegador
- Elementos, barras de navegação, formulários e imagens se redimensionarão e se reposicionarão automaticamente para ocupar o espaço disponível
- Utiliza-se o suporte de **media queries**
 - CSS utiliza a informação para ajustar o documento HTML



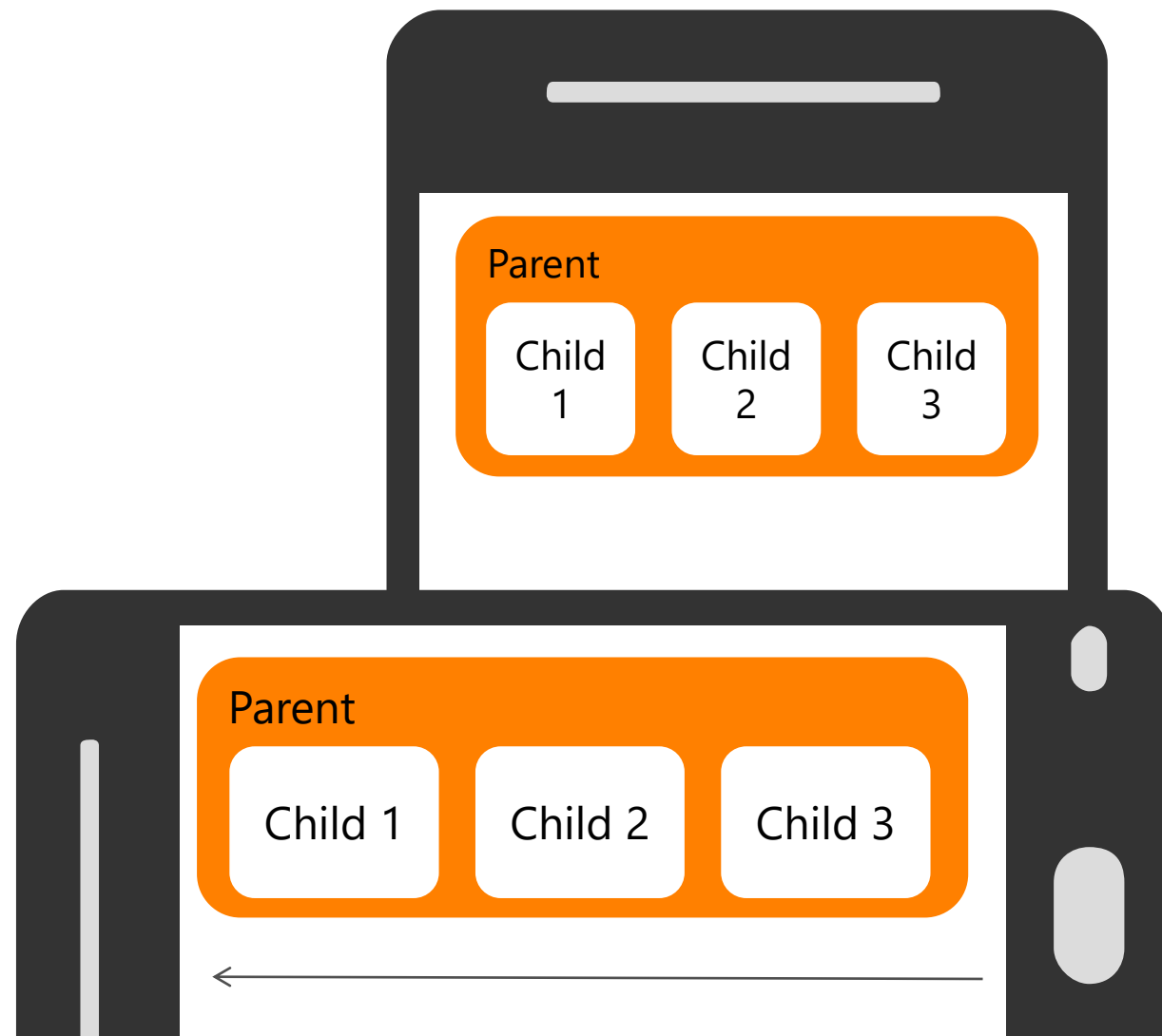
Itens Flexbox

- Define-se que um elemento utiliza o modelo Flexbox através da propriedade **display**
- Dois valores: **flexbox** e **inline-flexbox**
 - O valor `flexbox` define a caixa como block-level
 - O valor `inline-flexbox` define a caixa como inline-level
- Exemplos:
 - https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout



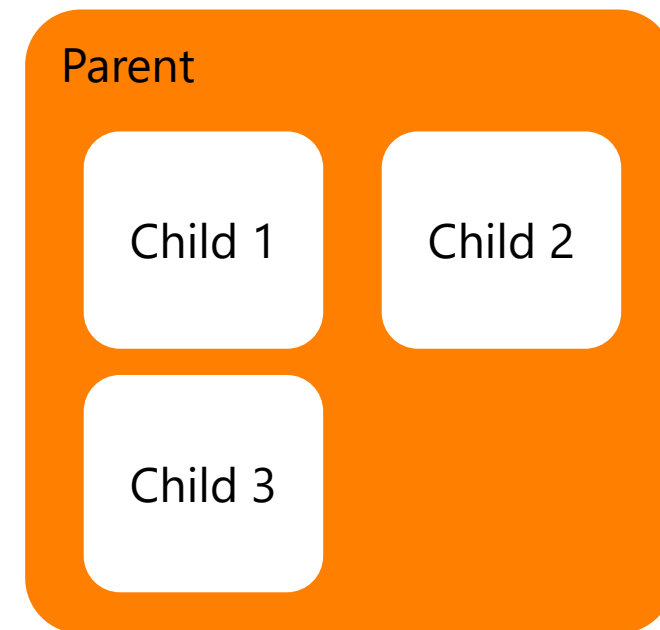
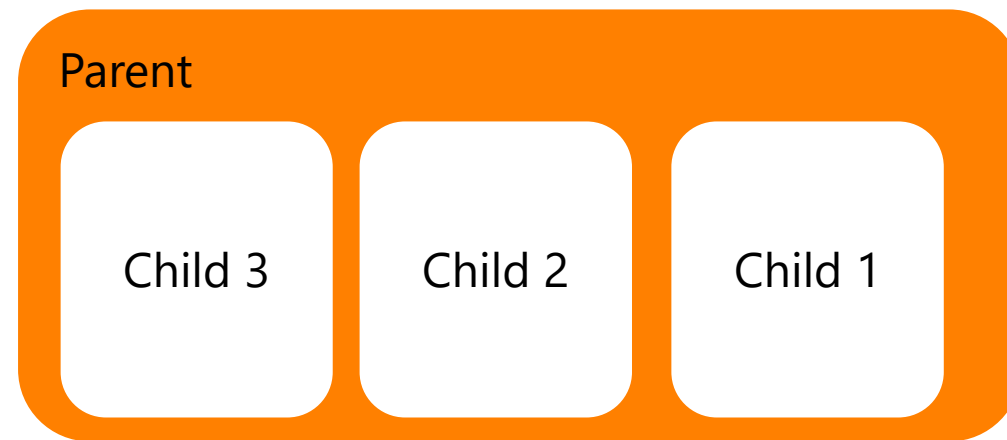
Trabalhando com Flexboxes

- Flexboxes podem conter outras caixas filhas, que são referenciadas como **flexbox items**
- Com a propriedade **flex**, é possível tornar os itens flexíveis também
- Lembre-se que a propriedade **display** é utilizada para tornar as caixas pais flexíveis
- A propriedade **flex** também pode ser usada para escalonar proporcionalmente itens flexbox quando o flexbox aumenta ou diminui de tamanho



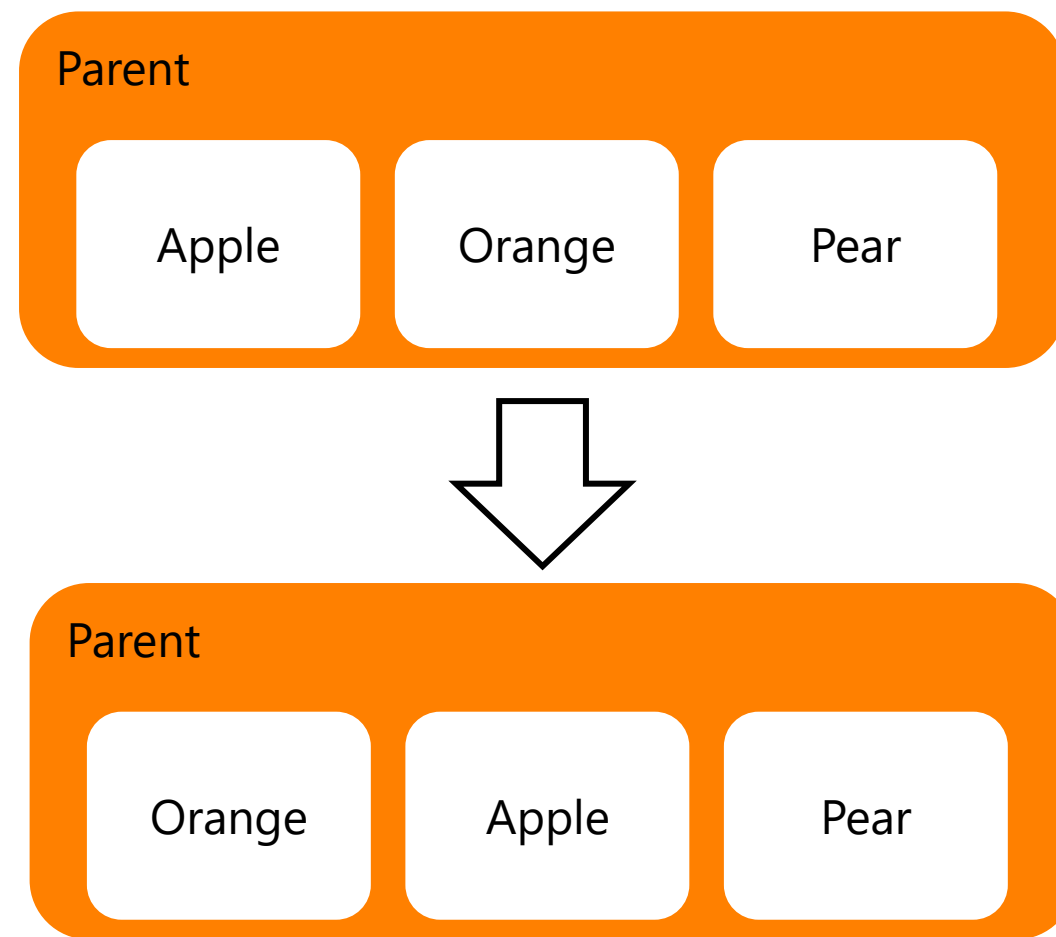
Trocando a Direção dos Itens Filhos

- A propriedade **flex-direction** permite que se troque a direção das caixas filhas dentro de um flexbox
 - Valores `row`, `row-reverse`, `column`, e `column-reverse`
- A propriedade **flex-wrap** determina se as caixas filhas irão se deslocar para a próxima linha se a janela diminui de tamanho
 - Valores `nowrap`, `wrap`, e `wrap-reverse`
- A propriedade **flex-flow** define **flex-direction** e **flex-wrap** ao mesmo tempo

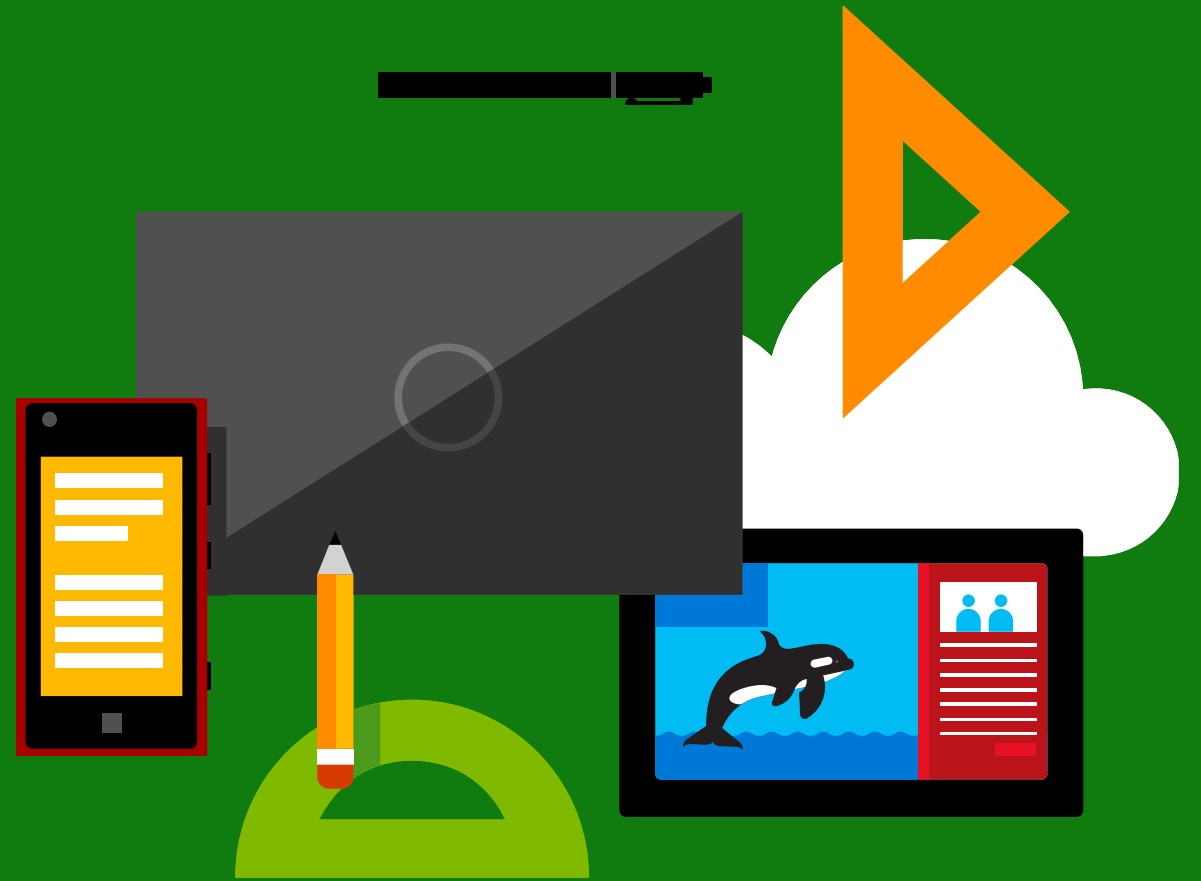


Ordenando e Arranjando Conteúdo

- A propriedade **flex-order** permite ajustar a ordem do conteúdo dentro de um flexbox
- A propriedade agrupa caixas filhas a fim de controlar a ordem que aparecem em um layout
- O valor padrão é 0

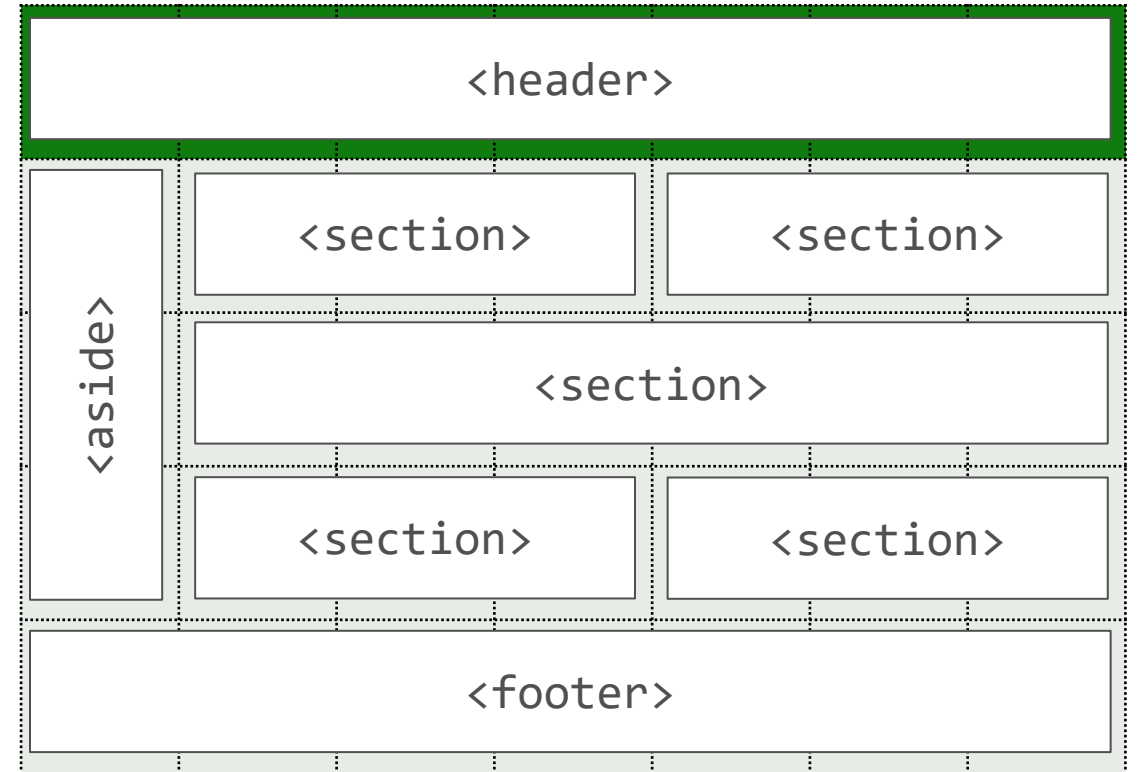


Grid Layout



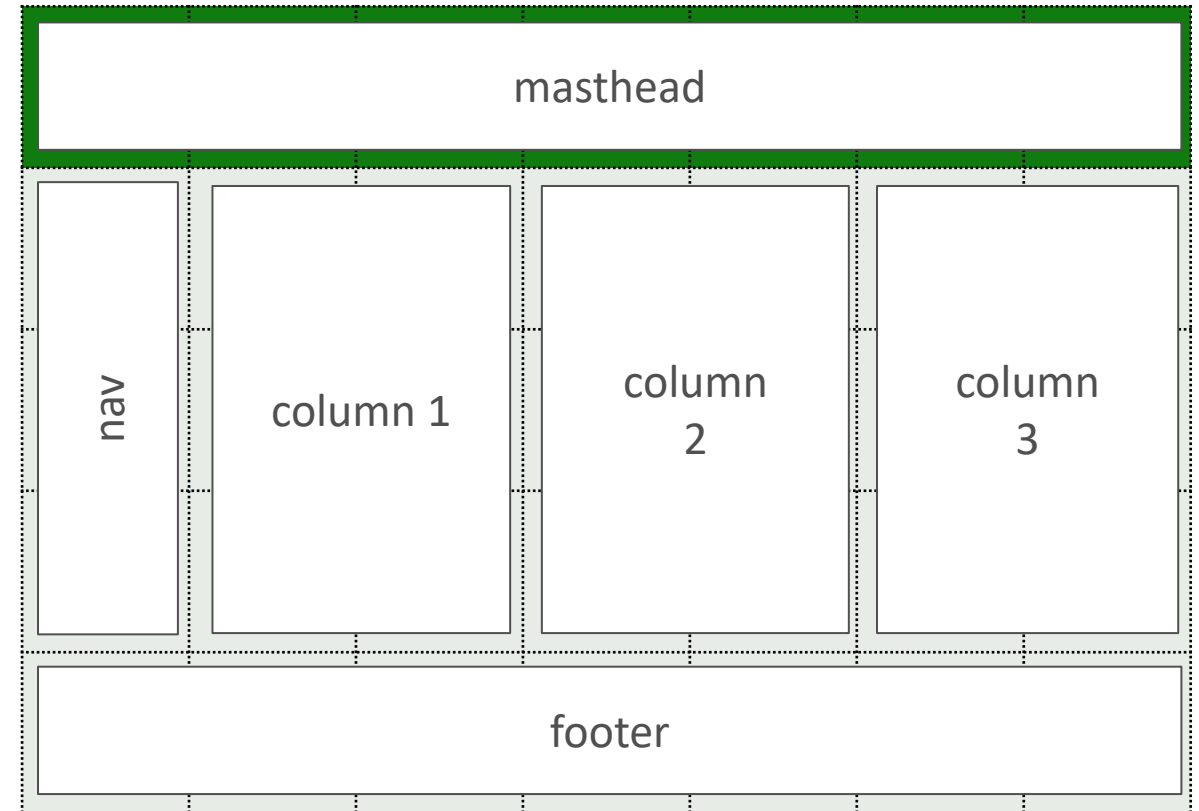
Grid Layout Model

- Quando o Flexbox Box Model não for apropriado, pode-se utilizar o Grid Layout Model
- O Grid Layout Model utiliza CSS para estruturar conteúdo utilizando linhas e colunas
- Grids são extremamente flexíveis e uma alternativa fácil de organizar conteúdo sem recorrer a tabelas do HTML
- Exemplos:
 - https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Grids



Grid Items

- Grid layouts são bastante similares a tabelas, pois são compostos de linhas e colunas
- O conteúdo dentro de grid layouts são também modulares, permitindo mover conteúdo de uma parte para outra
- Elementos filhos em um grid são chamados de grid items, e podem ser posicionados de acordo com grid tracks, grid lines, ou grid cells

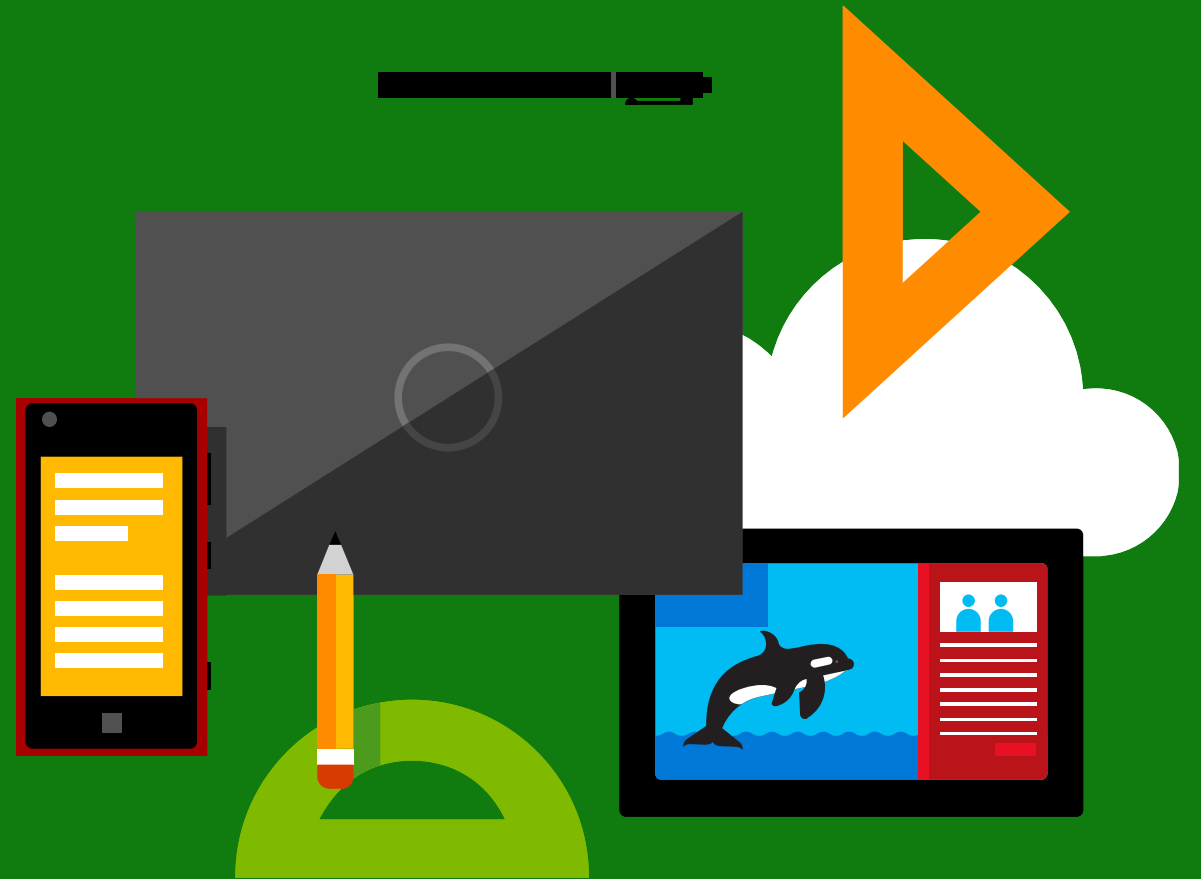


Definindo o Grid Layout

- Defina um layout através da propriedade **display**, junto com os valores `grid` ou `inline-grid`

CONCEITO DE LAYOUT	DESCRIÇÃO
Grid tracks	As colunas e linhas de um grid
Grid lines	As linhas horizontais ou verticais que separam colunas e linhas
Grid cells	O espaço lógico onde o conteúdo é posicionado

Código Legado

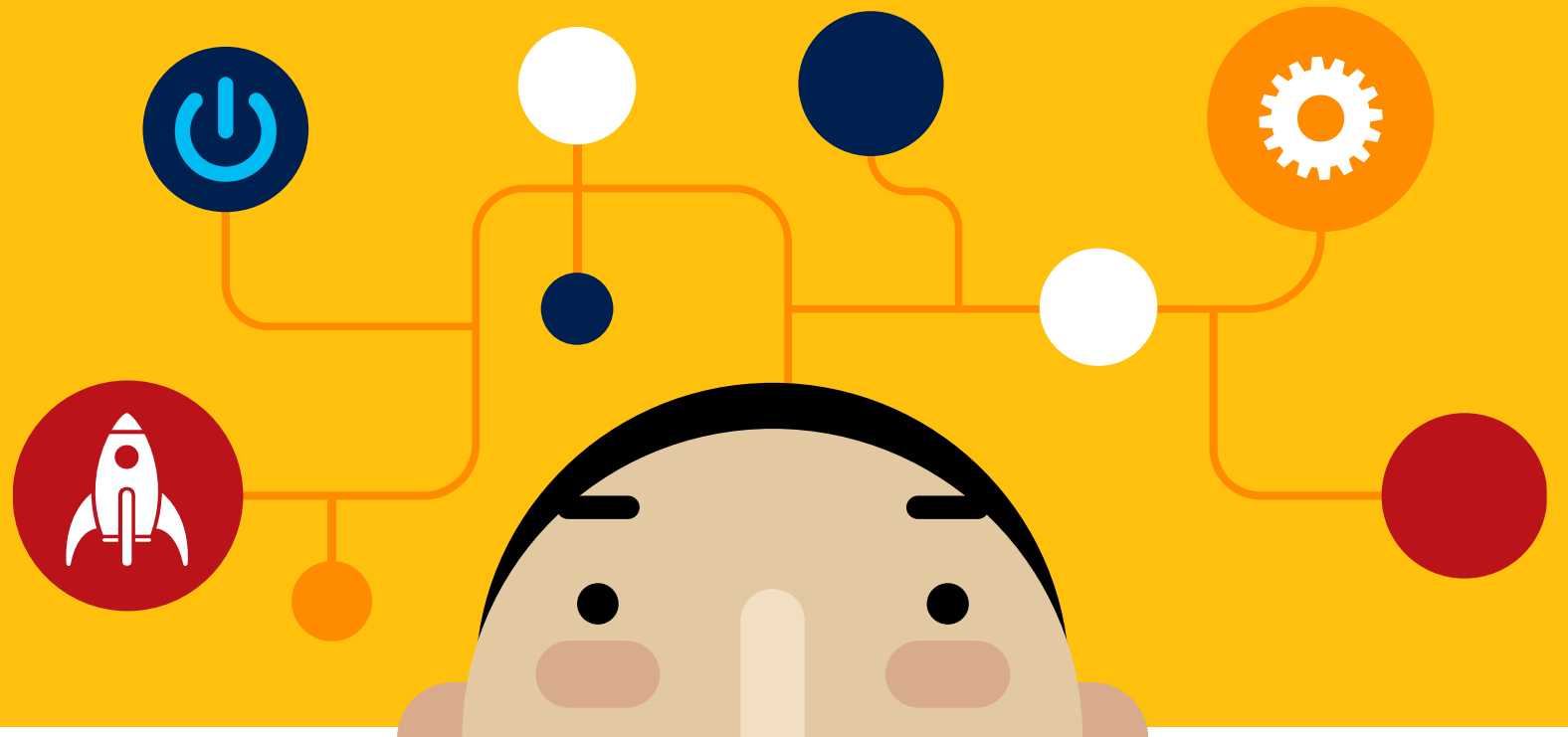


Código Legado

- Cuidado com exemplos e folhas de estilo que NÃO utilizam os métodos mais atuais de posicionamento de elementos em uma página
- Evitar ao máximo usar “métodos legados”
 - Somente utilizar se os navegadores alvo não suportarem os novos padrões CSS
- Exemplo:
 - <https://github.com/mdn/learning-area/blob/master/css/styling-boxes/box-model-recap/css-tables-example.html>
- Saiba mais:
 - https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Legacy_Layout_Methods

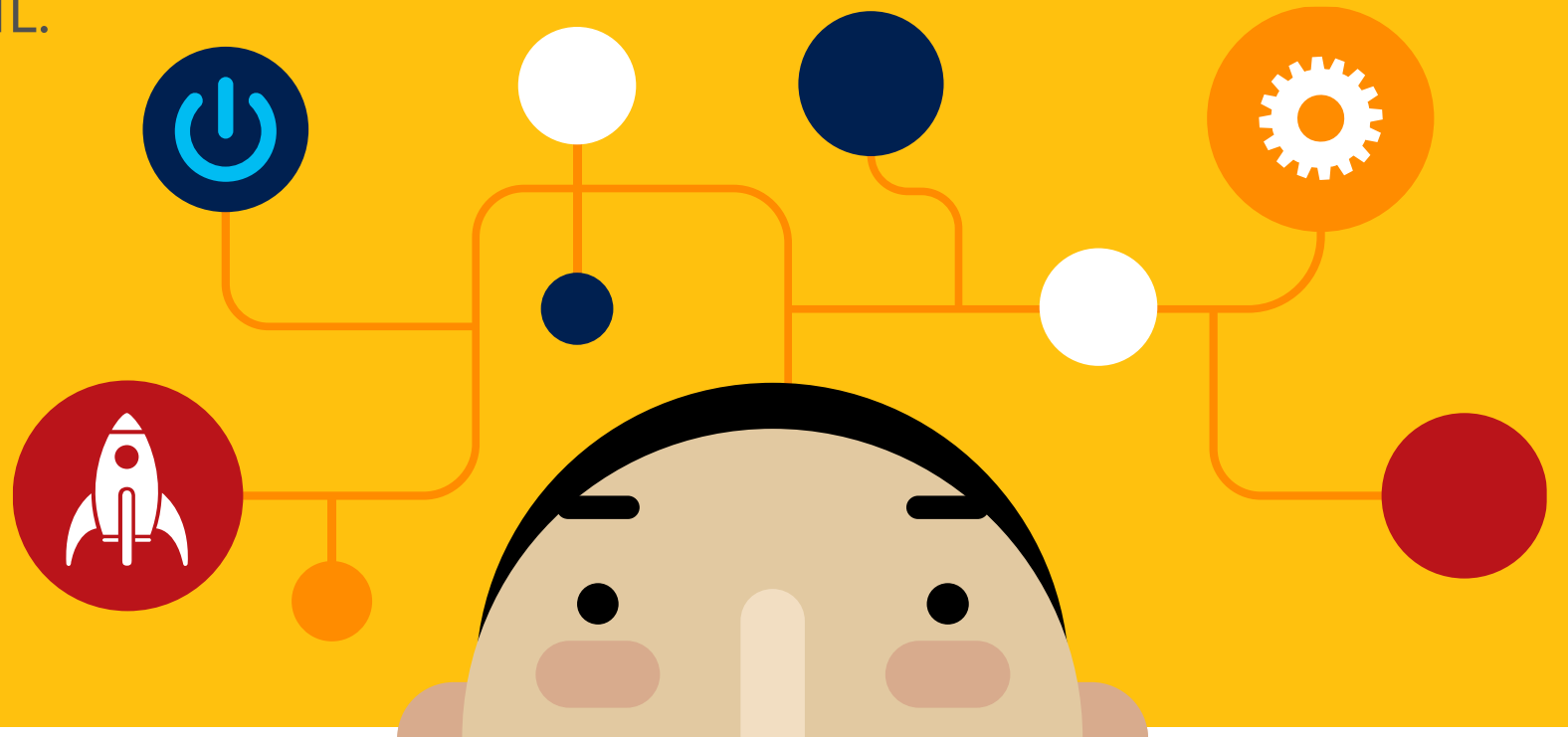
Laboratório

Pesquise na Web por repositórios que fornecem páginas CSS com licença gratuita. Escolha uma e analise sua estrutura. Procure entender exatamente o quê cada propriedade faz.

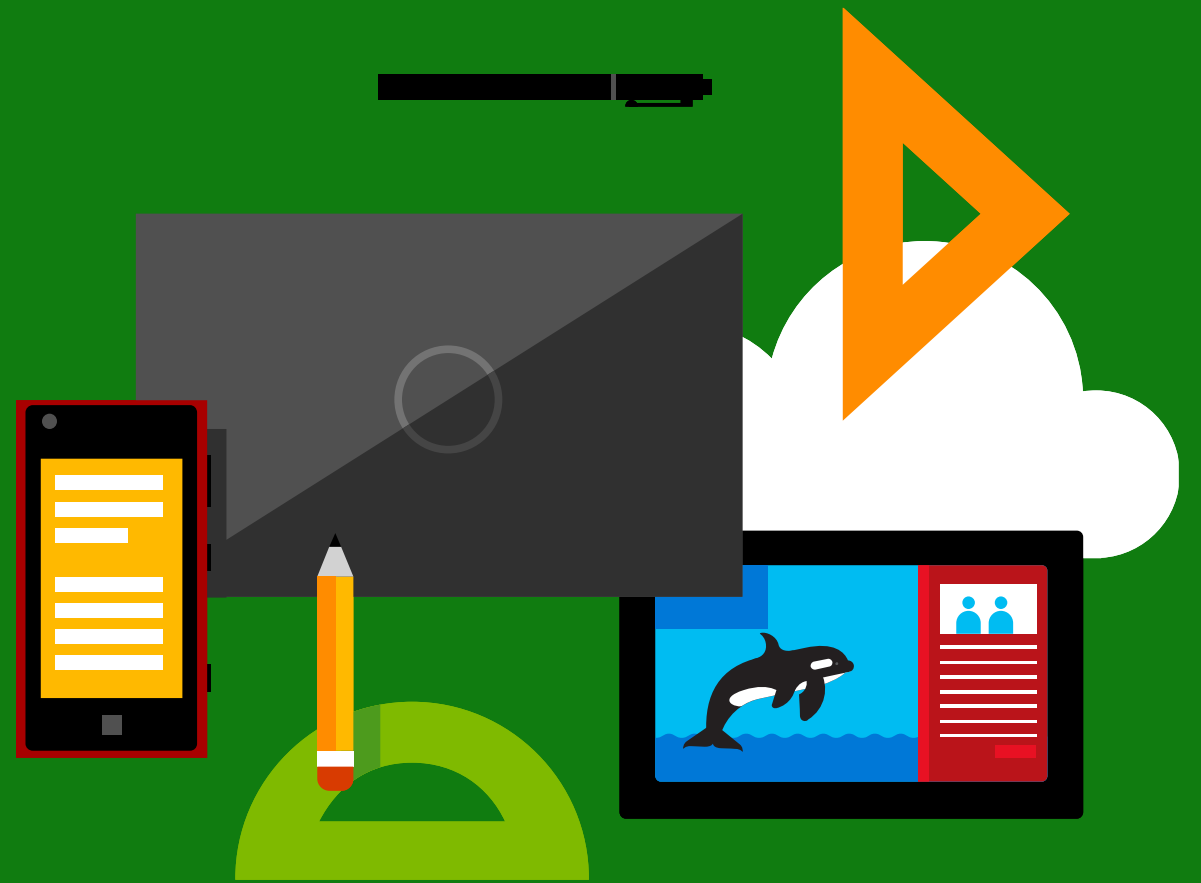


Laboratório

Pense na estrutura e informações de um currículo. Crie um documento HTML com as informações essenciais do seu currículo. Depois, crie uma página de estilos CSS e aplique ao documento HTML.



Processadores de CSS



Extensões ao CSS

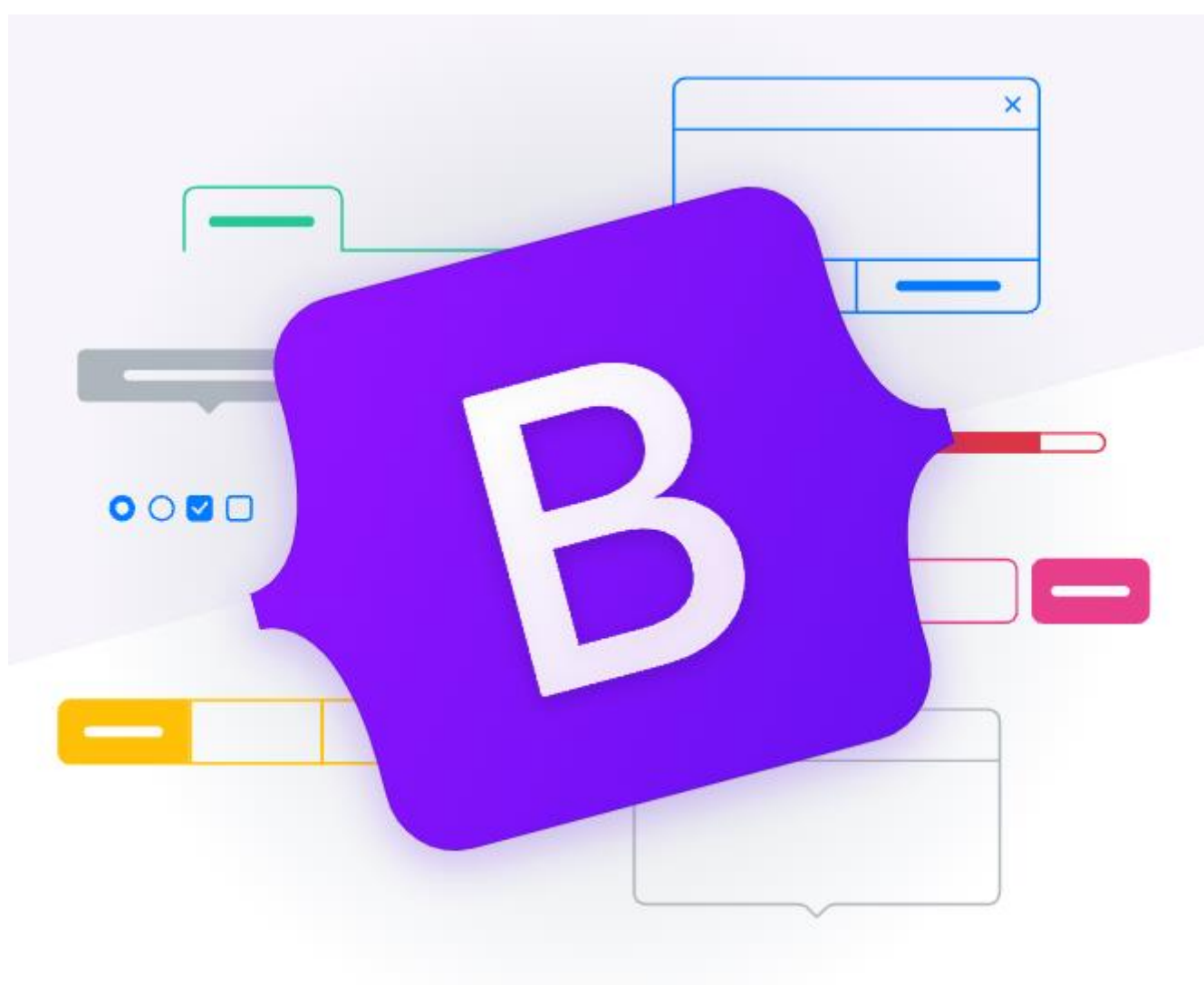
- Alguns frameworks fazem uso de processadores de CSS que permitem adicionar elementos novos à linguagem
 - LESS <https://lesscss.org/>
 - SASS <https://sass-lang.com/>
 - Tailwind <https://tailwindcss.com/>

Bootstrap



Bootstrap

- Uso da linguagem CSS pode ser bastante complexo
- Suporte a diferentes navegadores é um desafio
- Bootstrap é um framework de temas e layout criado pelo Twitter
 - Resolve rapidamente problemas básicos de layout
 - Torna o uso dos recursos de CSS muito mais simples
- Utiliza CSS3 para criação de páginas responsivas, com isto a página se adapta dinamicamente para diferentes navegadores e tamanhos de tela



<http://getbootstrap.com/>

Bootstrap

- Cuidado!
- O Bootstrap possui uma coletânea de componentes de UI opcionais que dependem de bibliotecas de JavaScript adicionais (tais como jQuery)
- Evite, sempre que possível, utilizar essas bibliotecas adicionais de forma conjunta com outros frameworks a fim de não causar conflitos

Laboratório

Retome o documento do exercício sobre uma página de currículo. Aplique o Bootstrap sobre o documento. Procure também aplicar um tema do Bootstrap sobre a página.

