# INFINITY SCHOOL

VISUAL ART CREATIVE CENTER

AULA 05 – CSS AVANÇADO

# O QUE IREMOS APRENDER

01 DISPLAY

02 BOX MODEL

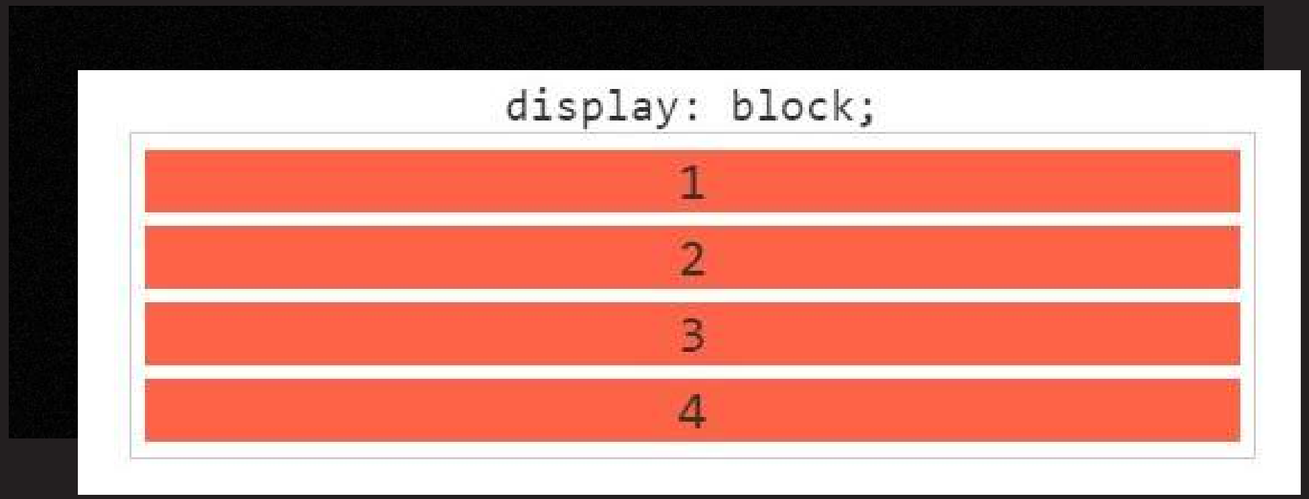
03 POSITION

# DISPLAY

A propriedade Display do CSS desempenha um papel fundamental na renderização dos elementos HTML em uma página. Ela é responsável por determinar o tipo de caixa que um elemento cria e como ele se relaciona e interage com outros elementos na página. Ao utilizar a propriedade Display, é possível controlar o comportamento e a aparência dos elementos, proporcionando uma experiência visual mais adequada e personalizada aos usuários.
Existem três principais valores que podem ser atribuídos à propriedade Display, cada um com características e efeitos específicos na exibição dos elementos na página:

# DISPLAY BLOCK

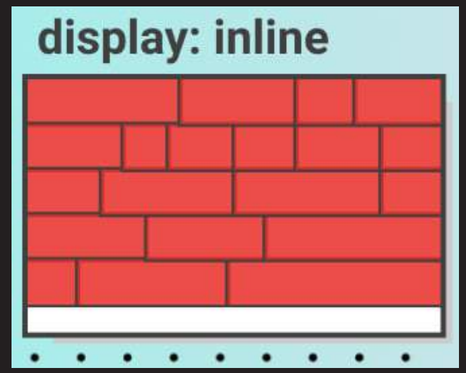
Display Block: Um elemento com display: block é exibido em um bloco separado na página, ocupando toda a largura disponível. Ele inicia uma nova linha e é utilizado para criar seções de conteúdo distintas, como parágrafos, títulos (headings), divs, entre outros.



# DISPLAY INLINE

Um elemento com display: inline é exibido em linha, permitindo que outros elementos fiquem ao lado dele na mesma linha. Elementos inline são usados para agrupar pequenos pedaços de conteúdo, como links, spans e imagens. Esse tipo de display não permite definir altura e largura.

```
<span class="meu-elemento"> Este é um elemento inline </span>

.meu-elemento {
display: inline;
}
```

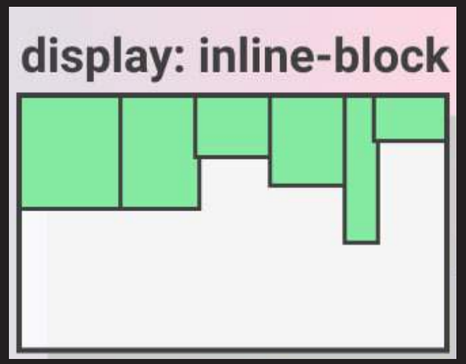



IN

# DISPLAY INLINE-BLOCK

O valor inline-block combina as características de um elemento inline e de um elemento block. Ele permite que outros elementos fiquem ao lado dele, como um elemento inline, mas também aceita a definição de largura e altura, como um elemento block. Essa propriedade é útil quando desejamos criar um layout em que os elementos sejam exibidos em linha, mas ainda tenham controle sobre suas dimensões.

```
<span class="meu-elemento"> Este é um elemento inline-block </span>

.meu-elemento {
display: inline-block;
 }
```



# BOX MODEL

Box Model: O Box Model é um conceito fundamental para entender como os elementos HTML são dimensionados e espaçados na página. Destaca-se que consiste em quatro elementos principais que desempenham um papel crucial no layout e no design da página. Esses elementos são responsáveis por definir a largura, altura, margens, preenchimento e bordas dos elementos HTML.
Ao compreender e dominar o Box Model, os desenvolvedores web podem criar designs mais sofisticados, controlando com precisão o espaço e a aparência dos elementos em suas páginas. Portanto, é fundamental ter um bom entendimento do Box Model ao trabalhar com HTML e CSS, pois isso permitirá que você crie layouts visualmente atraentes e bem estruturados.

# CONTEÚDO DE UM ELEMENTO

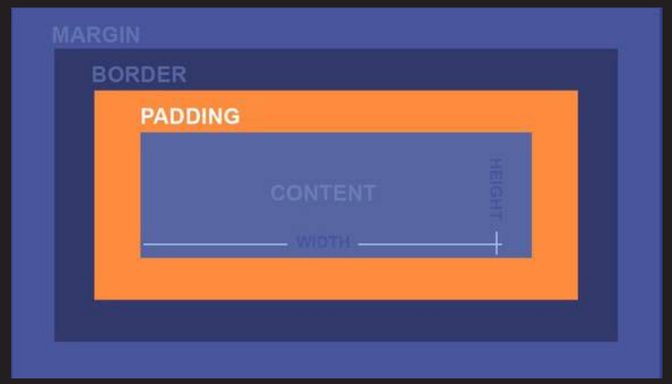
O conteúdo de um elemento é a área que contém o texto, imagens ou outros elementos dentro do próprio elemento. É o que percebemos como o conteúdo principal desse elemento. O tamanho do conteúdo é determinado tanto pelo seu próprio conteúdo quanto pelas propriedades de largura e altura aplicadas a ele.

```
<div class="meu-elemento">Este é um elemento de conteúdo</div>

.meu-elemento {
 width: 200px;
 height: 100px;
}
```

# PADDING

O padding é a área de preenchimento entre o conteúdo do elemento e sua borda, utilizado para adicionar espaço interno ao elemento. Pode ser configurado individualmente para cada lado do elemento (superior, direito, inferior e esquerdo) ou de forma simplificada para todos os lados ao mesmo tempo. A propriedade CSS utilizada para definir o tamanho do padding é a padding.

```
<div class="meu-elemento">Este é um elemento com padding</div>

.meu-elemento {
 padding: 20px;
}
```



# BORDER

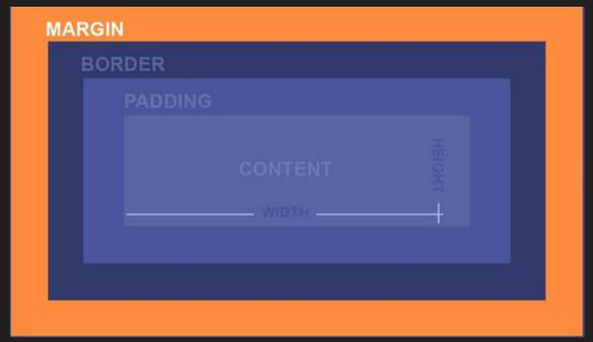
A borda é a linha que envolve o conteúdo e o padding do elemento. Ela tem a função de delimitar visualmente o elemento e pode ser estilizada com diferentes cores, espessuras e estilos utilizando a propriedade CSS border. Pode ser configurada individualmente para cada lado do elemento ou de forma simplificada para todos os lados simultaneamente.

```
<div class="meu-elemento">Este é um elemento com borda</div>

.meu-elemento {
 border: 3px solid red;
}
```

# MARGIN

**A margem é a área externa ao elemento, criando espaço entre o próprio elemento e outros elementos vizinhos. Ela é utilizada para adicionar espaçamento entre os elementos na página. A margem pode ser configurada individualmente para cada lado do elemento ou de forma simplificada para todos os lados simultaneamente. Para definir o tamanho da margem, usamos a propriedade CSS margin.**

```
<div class="meu-elemento">Este é um elemento com margem</div>

.meu-elemento {
 margin: 10px;
}
```



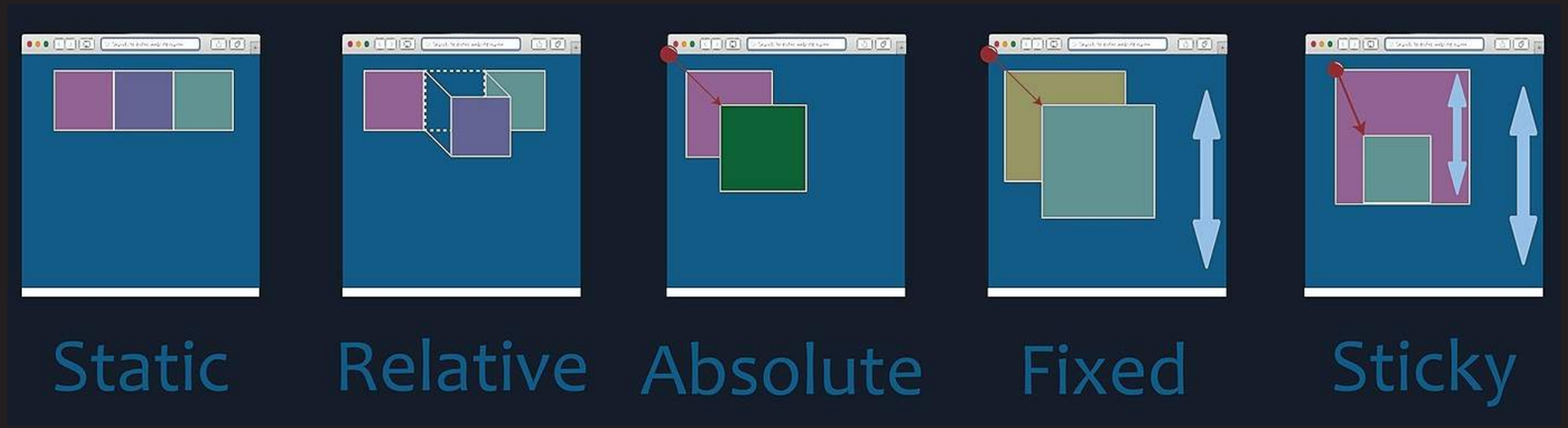
# BOX SIZING

Box-sizing é uma propriedade do CSS que define como a largura e altura de um elemento são calculadas. O valor padrão é content-box, que leva em consideração apenas o conteúdo do elemento ao calcular suas dimensões. No entanto, também podemos utilizar o valor border-box, que inclui o padding e a borda no cálculo da largura e altura do elemento. Essa é uma opção útil quando queremos controlar o espaço total ocupado pelo elemento, levando em conta o conteúdo, o padding e a borda.

```
<div class="meu-elemento">Elemento com box-sizing: border-box</div>

.meu-elemento {
 box-sizing: border-box;
 width: 200px;
 padding: 20px;
 border: 1px solid black;
}
```

# POSITION

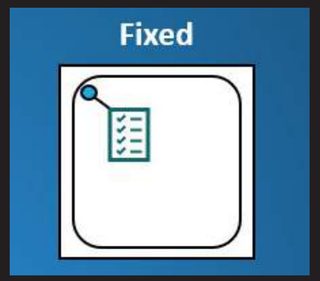
A propriedade position permite controlar a posição de um elemento na página e oferece diferentes valores que determinam como o elemento é posicionado em relação aos seus elementos pais e à janela de visualização. Esses valores são essenciais para garantir um layout adequado e personalizado para o seu site. Existem cinco principais valores para a propriedade position que você pode utilizar de acordo com suas necessidades específicas.



# FIXED

O valor fixed para a propriedade position posiciona o elemento de forma fixa em relação à janela de visualização. Isso significa que, mesmo quando a página é rolada, o elemento permanece no mesmo lugar. Essa configuração é útil quando desejamos fixar um elemento, como um menu, no topo da página para que ele fique sempre visível.

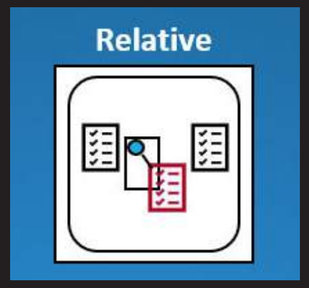
```
<div class="meu-elemento">Este é um elemento com posição fixed</div>

.meu-elemento {
 position: fixed;
top: 0;
left: 0;
}
```



# RELATIVE

Ao usar position: relative, o elemento é posicionado em relação à sua posição normal. Podemos deslocá-lo utilizando as propriedades CSS top, right, bottom e left. Essas propriedades nos permitem mover o elemento para cima, para a direita, para baixo e para a esquerda em relação à sua posição original.

```
<div class="meu-elemento">Este é um elemento com posição relative </div>

.meu-elemento {
 position: relative;
 top: 10px;
 left: 10px;
}
```



# ABSOLUTE

Com position: absolute, o elemento é posicionado de forma absoluta em relação ao elemento pai mais próximo que tenha uma posição definida. Isso significa que o elemento é removido do fluxo normal do documento. Podemos definir a posição absoluta usando as propriedades CSS top, right, bottom e left.
Ao utilizar a posição absolute, é importante observar que o elemento pode se sobrepor a outros elementos, e seu posicionamento pode variar dependendo da altura e largura dos elementos pais.

```
<div class="meu-elemento">Este é um elemento com posição absolute </div>

.meu-elemento {
 position: absolute;
 top: 10px;
 left: 10px;
}
```

# STICKY

**Position**: sticky é um híbrido entre position: relative e position: fixed. Ele posiciona o elemento de forma relativa até que seja rolado para fora da janela de visualização. Quando isso acontece, o elemento se comporta como um elemento fixo, permanecendo no mesmo lugar mesmo ao rolar a página. Essa configuração é útil quando queremos que um elemento fique visível durante a rolagem, mas depois seja fixado em um local específico.

```
<div class="meu-elemento">Este é um elemento com posição sticky</div>

.meu-elemento {
 position: sticky;
 top: 20px;
}
```

# ATIVIDADE PRÁTICA

1. Crie um arquivo HTML básico.
2. Adicione uma seção para cada um dos valores de posição (relative, absolute, fixed, sticky).
3. Dentro de cada seção, crie um elemento como uma  e adicione algum conteúdo descritivo.
4. Aplique estilos CSS para cada seção e elemento.

- Utilize diferentes cores de fundo.
- Defina larguras, alturas, margens, preenchimentos e bordas.
- Experimente diferentes valores para top, right, bottom e left conforme apropriado.

# ATIVIDADE PRÁTICA

1. Expanda o arquivo HTML existente, adicionando seções que representem diferentes breakpoints (pontos de interrupção) de tamanho de tela. Considere tamanhos comuns, como para dispositivos móveis, tablets e desktops.
2. Utilize media queries em seu arquivo CSS para definir estilos diferentes para cada breakpoint. Considere ajustar larguras, alturas, fontes e margens para otimizar a apresentação em cada dispositivo.
3. Adapte o layout para garantir uma experiência de usuário agradável em diferentes dispositivos. Considere a reorganização de elementos, a alteração de tamanhos de fonte e a ocultação de elementos não essenciais em telas menores.
4. Experimente ocultar ou reorganizar elementos para tornar o conteúdo mais acessível em telas menores. Isso pode incluir a utilização de flexbox ou grid para ajustar dinamicamente a disposição dos elementos.
5. Teste o layout em vários dispositivos ou utilize as ferramentas de desenvolvedor do navegador para simular diferentes tamanhos de tela. Certifique-se de verificar a aparência e a usabilidade em dispositivos móveis, tablets e desktops.

# ATIVIDADE PRÁTICA

1. Escolha um dos elementos presentes em seu layout e aplique uma transição suave à propriedade background-color quando o cursor do mouse passar sobre o elemento.
2. Selecione outro elemento e crie uma animação contínua, alterando a opacidade ou posição ao longo do tempo.
3. Adicione botões ou interações para controlar as animações, proporcionando funcionalidades como iniciar, pausar ou reverter.
4. Explore diversas propriedades de animação, como @keyframes, animation-duration, animation-timing-function, entre outras, para garantir uma experiência visualmente agradável.
5. Certifique-se de testar as animações em navegadores modernos para assegurar compatibilidade e uma execução suave em diferentes plataformas.

# SE LIGA NO CONTEÚDO DA PRÓXIMA AULA!

AULA 06 DE CSS.
RESPONSIVIDADE E
CORES

IN

INFINITY SCHOOL
VISUAL ART CREATIVE CENTER