DISCIPLINA: **Desenvolvimento Web app**	PRONATEC
Funec - Riacho	
Prof. Rubens Palhares	

O **Flexbox** é um conceito do **CSS3** que visa organizar os elementos de uma página HTML dentro de seus containers de forma dinâmica. Ou seja, independente das suas dimensões eles sempre manterão um layout flexível dentro do seu elemento pai, reorganizando-se e acordo com a necessidade.

Neste documento conheceremos as **propriedades do CSS** que fazem parte desse recurso, analisando seu funcionamento, bem como exemplos práticos de uso.

Tópicos

Conceitos iniciais

O **Flexbox** é um conjunto de propriedades que tem por objetivo organizar itens dentro de um elemento pai, normalmente chamado de container, conforme ilustra a **Figura 1**.

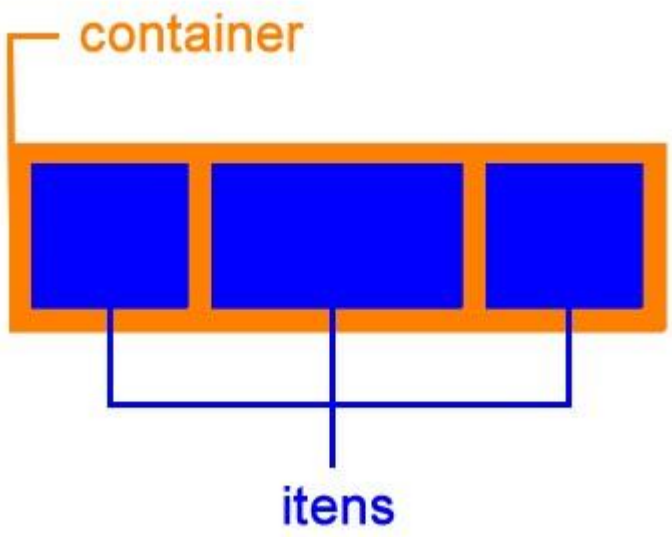

Figura 1 Estrutura dos elementos para aplicação do Flexbox

Portanto, para utilizar esse recurso é necessário ter no HTML ao menos um elemento (container) contendo outros (itens), como no código abaixo:

```
<div class="container">
<div class="item item1"></div>
<div class="item item2"></div> <div
class="item item3"></div>
</div>
```

Conforme veremos a seguir, algumas propriedades serão aplicadas ao container, enquanto outras serão aplicadas aos itens.

Alinhamento dos eixos

Para entender o **funcionamento do Flexbox** é necessário ter em mente alguns conceitos fundamentais que dizem respeito a como os itens são distribuídos no container. O principal deles é o conceito de eixo principal e eixo transversal, que depende do valor atribuído à propriedade flex-direction. Se essa propriedade receber o valor row ou row-reverse (organização em linhas), o eixo principal do container será o horizontal. Já se essa propriedade receber o valor column ou column-reverse (organização em coluna), o eixo principal será o vertical. Consequentemente isso definirá qual é o eixo transversal. Se o principal for o vertical, o transversal será o horizontal e vice-versa.

Na **Figura 2** podemos ver esse e outros conceitos ilustrados para o caso de o eixo principal ser o horizontal.

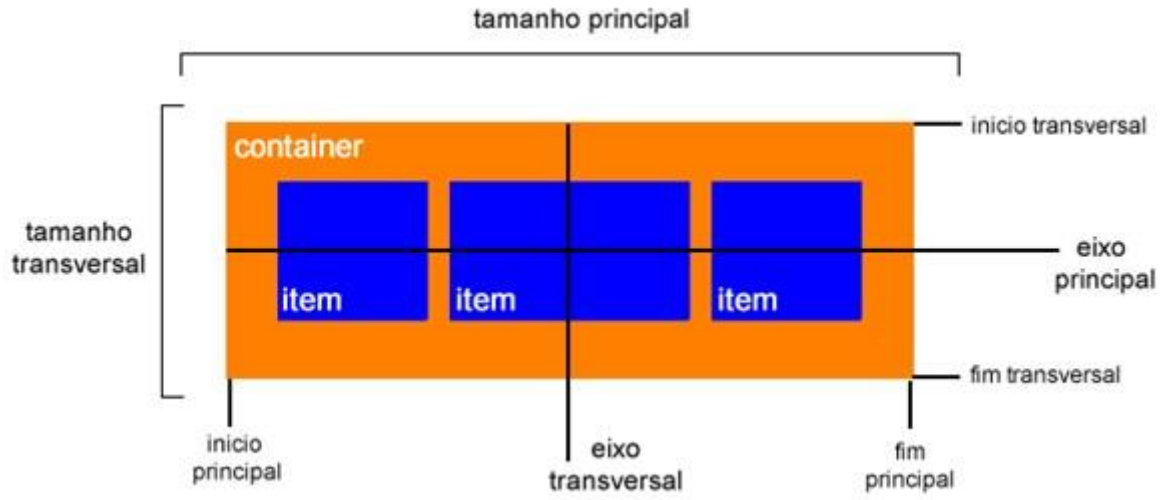

Figura 2. Alinhamento dos eixos do Flexbox

- **Tamanho principal:** É a dimensão do elemento na direção do eixo principal (largura, caso horizontal e altura caso vertical);
- **Tamanho transversal:** É a dimensão do elemento na direção do eixo transversal (largura, caso horizontal e altura caso vertical);
- **Início principal e final principal:** Representam o início e o fim do eixo principal;

- **Início transversal e final transversal:** Representam o início e o fim do eixo transversal.

Essas informações serão importantes para compreendermos o funcionamento das diversas **propriedades do Flexbox** que veremos a seguir. **display**

O primeiro passo para **utilizar o Flexbox** é definir a propriedade display do container com o valor flex. Isso é necessário para que as demais propriedades apresentem o resultado esperado.

A sintaxe de uso dessa propriedade é a seguinte:

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

flex-direction

A propriedade flex-direction deve ser aplicada ao container e define o eixo/fluxo de exibição em que os elementos serão organizados. A sintaxe e os valores possíveis para essa propriedade são apresentados a seguir:

```
.container { display: flex; flex-direction: [row /
row-reverse / column / column-reverse];
}
```

- **row** (padrão): Os itens são organizados em forma de linha da esquerda para a direita;
- **row-reverse**: Os itens são organizados em forma exibição em linha da direita para a esquerda;

- **column**: Os itens são organizados em forma de colunas iniciando de cima para baixo;
- **column-reverse**: Os itens são organizados em forma de colunas iniciando de baixo para cima.

A **Figura 3** ilustra o funcionamento de cada valor.

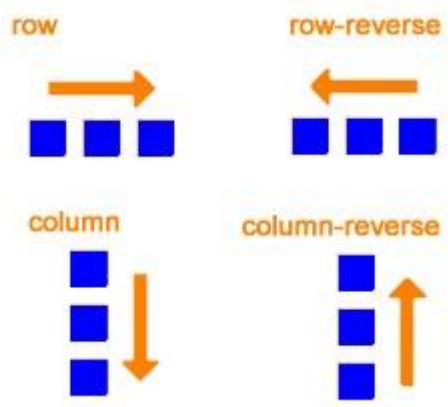

Figura 3. Funcionamento da propriedade flex-direction **flex-wrap**

Por padrão os itens do container tentarão se ajustar em uma única linha dentro do container, mas para isso a sua largura original pode ser ajustada para que todos caibam na largura do elemento pai. Com a propriedade flex-wrap aplicada ao container podemos alterar esse comportamento, fazendo com que ocorra a “quebra de linha” nos itens.

A sintaxe e os valores possíveis para essa propriedade são apresentados a seguir:

```
.container { display: flex; flex-wrap:
[nowrap / wrap / wrap-reverse];
}
```

- **nowrap** (padrão): Todos os itens serão dispostos em uma linha;

- **wrap**: Ocorrerá a quebra de linha e os itens mais à direita serão deslocados para a linha de baixo;
- **wrap-reverse**: Ocorrerá a quebra de linha e os itens mais à direita serão deslocados para a linha de cima;

flex-flow

Esta propriedade é uma forma abreviada para a escrita das propriedades flexdirection e flex-wrap, nesta ordem. Portanto, ela se aplica ao container.

Normalmente essas propriedades são definidas uma a uma, da seguinte forma:

```
.container { display: flex; flex-
direction: column;
04  flex-wrap: wrap;
05  }
```

Já com o flex-flow podemos escrever as duas de forma simplificada:

```
flex-flow: column wrap;
```

justify-content

A propriedade justify-content define o alinhamento dos itens ao longo do eixo principal do container. A sintaxe e os valores possíveis para essa propriedade são apresentados a seguir:

```
.container { display: flex; justify-content: [flex-
start/flex-end/center/space-between/space-around];
}
```

- **flex-start (padrão)**: Os itens são alinhados a partir do início do eixo principal;
- **flex-end**: Os itens são alinhados a partir do fim do eixo principal;
- **center**: Os itens são alinhados ao centro do eixo principal;
- **space-between**: O primeiro item é deslocado para o início do eixo principal, o último é deslocado para o final do eixo principal e os demais são distribuídos uniformemente entre eles;

- **space-around**: Os itens são uniformemente distribuídos ao longo do eixo principal. Aqui, porém, são atribuídas margens iguais à esquerda e à direita (ou acima e abaixo, dependendo da direção do eixo principal). Por isso o primeiro e o último item não ficam “colados” nas bordas do container.

A **Figura 4** ilustra o funcionamento de cada valor:

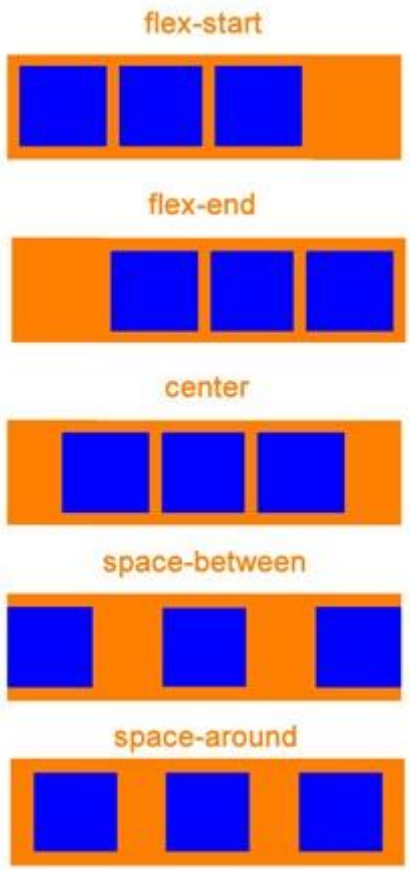

Figura 4 Representação da propriedade justify-

content **align-content**

Essa propriedade define como as linhas são distribuídas ao longo do eixo transversal do container. Ela só terá efeito se o elemento tiver mais de uma linha, ou seja, se ele tiver elementos suficientes para quebrar a linha e a propriedade flexwrap:wrap tiver sido definida. A sintaxe e os valores possíveis para essa propriedade são apresentados a seguir:

```
.container { display: flex; align-content: [stretch/flex-start/flex-end/center/space-between/space-around];
}
```

- **stretch** (padrão): As linhas são distribuídas uniformemente ao longo do eixo transversal, ocupando todo o espaço disponível;
- **flex-start**: As linhas são distribuídas a partir do início do eixo transversal;
- **flex-end**: As linhas são distribuídas a partir do fim do eixo transversal;
- **center**: As linhas são mantidas no centro do eixo transversal;
- **space-between**: A primeira linha é deslocada para o início do eixo transversal, a última é deslocada para o final do eixo transversal e as demais são distribuídas uniformemente entre eles;
- **space-around**: As linhas são uniformemente distribuídas ao longo do eixo transversal. Aqui, porém, são atribuídas margens iguais à esquerda e à direita

(ou acima e abaixo, dependendo da direção do eixo transversal). Por isso a primeira e a última linha não ficam “coladas” nas bordas do container.

A **Figura 5** ilustra o funcionamento de cada valor:

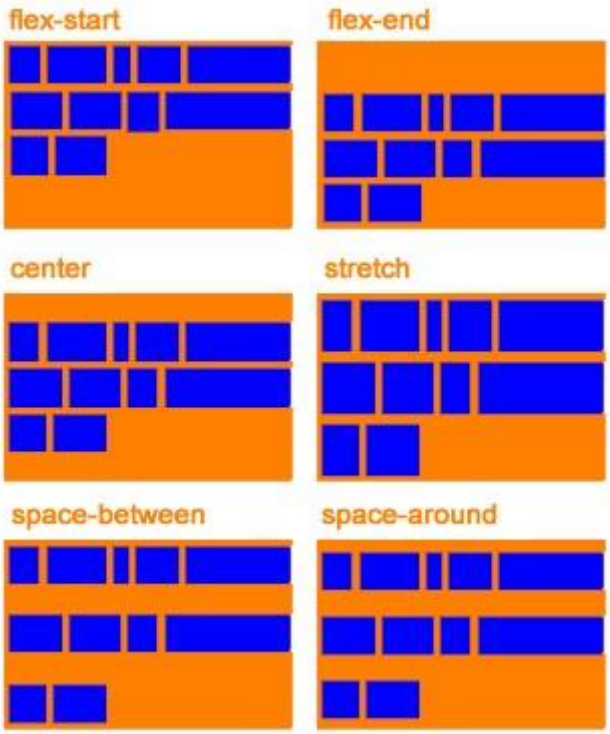

Figura 5. Funcionamento da propriedade align-content **align-items**

Essa propriedade define como os itens são distribuídos ao longo do eixo transversal do container. A sintaxe e os valores possíveis para essa propriedade são apresentados a seguir:

```
.container { display: flex; align-items:
[stretch/flex-start/flex-end/center/baseline];
}
```

- **stretch** (padrão): Os itens serão esticados para preencher toda a dimensão do eixo transversal (altura ou largura);

- **flex-start**: Os itens são deslocadas para o início do eixo transversal;
- **flex-end**: Os itens são deslocadas para o final do eixo transversal;
- **center**: Os itens são centralizados no eixo transversal;
- **baseline**: Os itens são alinhados a partir da base da primeira linha de texto de cada um.

A **Figura 6** ilustra o funcionamento de cada valor:

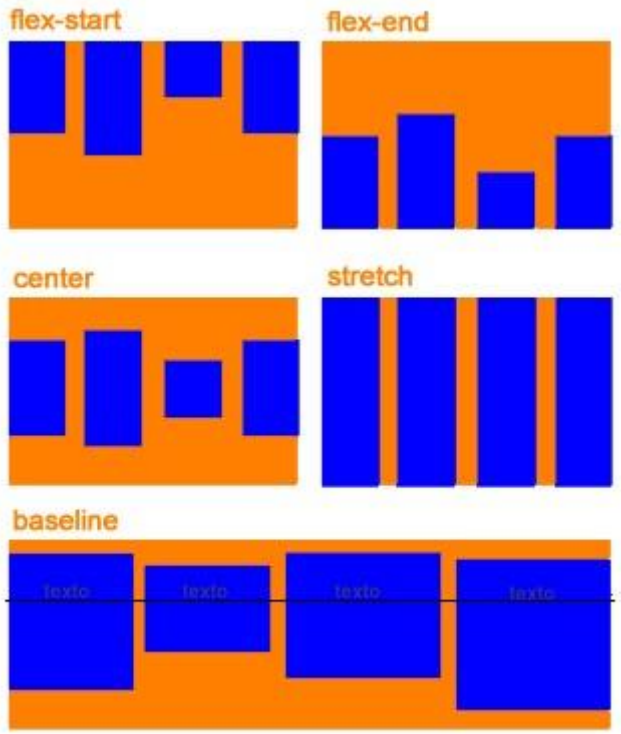

Figura 6. Representação da propriedade align-items

Order

Por padrão, os itens são distribuídos no container na ordem em que são inseridos no HTML. No entanto, essa ordem pode ser alterada por meio da propriedade order, cuja sintaxe vemos abaixo:

```
.item2 { order: [número];
```

```
}
```

O valor numérico atribuído a essa propriedade define a ordem do item. Por exemplo, o valor 2 faz com que o item seja o segundo item ao longo do eixo principal, enquanto o valor -1 faz com que ele apareça antes do primeiro. **flex-grow**

Esta propriedade define a proporção com que um item deve crescer caso seja necessário. Por padrão seu valor é 0, o que indica que o item não deve crescer, e são aceitos apenas valores numéricos positivos.

A sintaxe dessa propriedade é a seguinte:

```
.item2 { flex-grow:
[número];
}
```

flex-shrink

Esta propriedade define a proporção com que um item deve encolher caso seja necessário. Por padrão seu valor é 0, o que indica que o item não deve encolher, e são aceitos apenas valores numéricos positivos.

A sintaxe dessa propriedade é a seguinte:

```
.item2 {
flex-shrink: [número];
}
```

flex-basis

O flex-basis define o tamanho inicial que um item deve ter antes que o espaço ao seu redor seja distribuído. Ou seja, dependendo da direção do eixo principal (horizontal ou vertical), essa propriedade define a largura ou altura mínima do elemento antes que ele seja redimensionado por outras propriedades.

A sintaxe dessa propriedade é a seguinte:

```
.item2 { flex-basis:
[número];
}
```

O valor atribuído a essa propriedade pode ser em percentual, em pixels, ou a palavra auto, que é o valor padrão (considera as dimensões do item - width e height). **flex**

Esta propriedade é uma forma abreviada para a escrita das propriedades flexgrow, flex-shrink e flex-basis, nesta ordem.

A sintaxe dessa propriedade é a seguinte:

```
.item2 { flex: [flex-grow] [flex-shrink]
[flex-basis];
}
```

align-self

Esta propriedade permite sobrescrever no item o comportamento que foi definido pela propriedade align-items.

A sintaxe e os valores possíveis para essa propriedade são apresentados a seguir:

```
.item2 { align-self: [auto/stretch/flex-start/flex-
end/center/baseline];
}
```

- **auto** (padrão): Respeita o comportamento definido no container por meio do align-items;
- **stretch**: O item será esticado para preencher toda a dimensão do eixo transversal (altura ou largura);
- **flex-start**: O item é deslocado para o início do eixo transversal;
- **flex-end**: O item é deslocado para o final do eixo transversal;
- **center**: O item é centralizado no eixo transversal;
- **baseline**: O item é alinhado a partir da base da primeira linha de texto dos demais.

Exemplo prático

A seguir podemos ver um exemplo prático de uso do Flexbox. Primeiramente temos a estrutura do HTML na qual foram aplicadas as propriedades que vimos anteriormente:

```
<div class="container">
<div class="item item1">1</div>
<div class="item item2">2</div>
<div class="item item3">3</div>
</div>
```

E abaixo temos o código CSS responsável pela formatação desse layout:

```
.container {
```

```
background-color:
#FF5722; height: 100vh;
```

```
width: 100%;
display: flex; flex-direction: column; justify-content: center; align-items: center;
}

.item {
background-color: #0000ff;
color: #fff;
padding: 10px; margin: 10px 0;
}

.item1{ height: 100px; width: 100px;
}

.item2{ height: 50px; width: 150px;
}

.item3{ height: 100px;
```

```
width: 200px;

}
```

Run

Linha 2: Definimos a cor do plano de fundo do container para facilitar a visualização;

Linhas 3 e 4: Definimos a altura e largura do container para ocupar 100% da página;

Linha 5: Definimos o display flex no container para que as demais propriedades surtam o efeito esperado;

Linha 6: Com a propriedade flex-direction definida como column definimos que os itens devem ser organizados em coluna (um abaixo do outro);

Linha 7: Nesta linha definimos que os itens serão centralizados no eixo principal (vertical);

Linha 8: A propriedade align-items define aqui que os itens serão centralizados também ao longo do eixo transversal (horizontal);

Linhas 11 a 30: Neste trecho estamos definindo algumas características dos itens. Eles possuem diferentes alturas e larguras, o que facilita a visualização do resultado final (mesmo com diferentes dimensões, o layout é mantido organizado).

Fonte de pesquisa:
https://www.devmedia.com.br/css3-flexbox-funcionamento-e-propriedades/29532