INSTITUTO FEDERAL DE MATO GROSSO DO SUL
CAMPUS NAVIRAÍ

NOTAS DE AULA DE

LINGUAGEM DE APRESENTAÇÃO E ESTRUTURAÇÃO DE CONTEÚDOS

I

Prof. MSc. Luiz F. Picolo

NAVIRAÍ - MS

Atualizado em 20 de agosto de 2019

1 Introdução

Aprender é a única coisa que a mente nunca se cansa,
nunca tem medo e nunca se arrepende
Autor desconhecido

1.1 O que é FlexBox

O Layout do Flexbox oficialmente chamado de **Módulo de Layout de Caixa Flexível CSS** é um novo módulo de layout em CSS3 feito para melhorar os itens de alinhamento, direções e ordem no container, mesmo quando eles estão com tamanho dinâmico ou mesmo desconhecido. A principal característica do FlexBox é a capacidade de modificar a largura ou altura de seus filhos para preencher o espaço disponível da melhor maneira possível em diferentes tamanhos de tela [Stojanov 2015].

Muitos designers e desenvolvedores acreditam que a disposição layout por meio do FlexBox se torna mais fácil, já que o posicionamento dos elementos é mais simples. Assim, layouts mais complexos podem ser obtidos com menos código, levando a um processo de desenvolvimento mais simples e rápido. O algoritmo de layout do Flexbox é baseado na direção, diferente do layout em bloco ou inline, que são baseados vertical e horizontalmente [Stojanov 2015].

Esta nova tecnologia, hoje suportada por todos os navegadores modernos, fornece ferramentas que permitem a criação rápida de layouts complexos e flexíveis e recursos que eram difíceis com o CSS2.

1.2 Modelo do flexbox

Antes de começarmos a descrever as propriedades do flexbox, vamos dar uma pequena introdução ao modelo do Flexbox. O layout flex é constituído pelo container pai referido como **flex container** e seus filhos imediatos, que são chamados **flex items**.

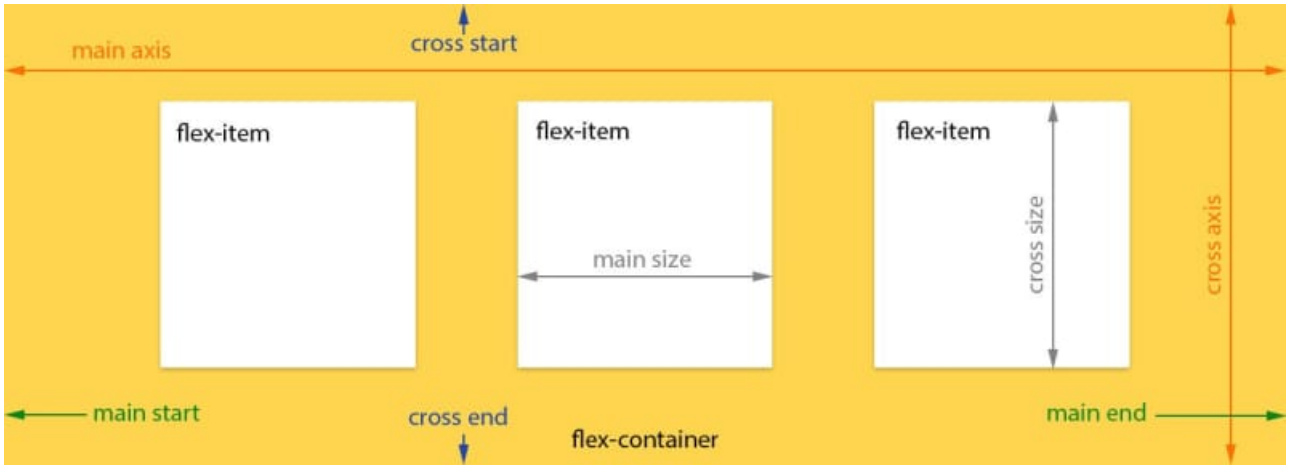

Figura 1 – Modedo do flexbox

Fonte: [Stojanov 2015]

Na caixa acima, você pode ver as propriedades e a terminologia usada para descrever o conteiner Flex e seus filhos. Para mais informações sobre o seu significado, leia o modelo oficial do flexbox da W3C (https://www.w3.org/TR/css-flexbox). No próximo capítulo veremos na prática como utilizar algumas propriedade do Flexbox.

2 Usando Flexbox

Aquele que não luta pelo futuro que quer,
deve aceitar o futuro que vier
Autor desconhecido

2.1 Terminologias básicas

Algumas terminologias básicas apresentadas neste capítulos devem ser entendidas para que haja um bom aproveitamento do conteúdo que será repassado. As duas são: **Containers** e **Itens**

A ideia de um container é muito semelhante a dos containers de navíos. Ou seja, um objeto que tem como única função conter outros objetos. Já os itens são, como você já deve ter imaginado, os objetos contidos dentro do container.

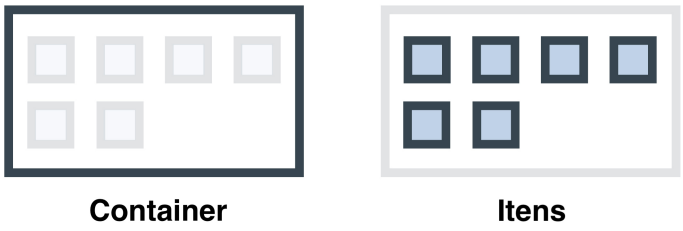

Figura 2 – Modedo do flexbox

Fonte: [Buytaert 2018] adaptado pelo autor

2.2 Propriedades do Flexbox

Nesta sessão vamos aprender algumas propriedades do FlexBox. Fiquei ciente que nem todos os códigos, principalmente o de estilo, são descritos no texto abaixo mas apenas os códigos básicos para o entendimento dos conceitos apresentados.

2.2.1 Flex Container

A primeira propriedade que veremos é a **flex-container**. Esta propriedade define qual será o nosso container. Para tanto apenas utilizamos a propriedade **display** com

o valor **flex**. Desta forma, passamos a nosso componente que ele deve se comportar de acordo com a diretiva, ou seja, organizar os itens de forma horizontal.

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

```
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>
```

Como resultado apresentado na Figura 3, obtemos os itens organizados de forma horizontal. Lembrando novamente que o resultado contém estilos e formatação, sendo que o código acima demostra apenas as propriedades básicas sem estilo.

Figura 3 – Flex Container

Fonte: O autor

2.2.2 Flex Direction

A propriedade **flex-direction** define em qual direção o conteiner deseja empilhar os itens. O valor **column** empilha os itens verticalmente (de cima para baixo):

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

Como resultado apresentado na Figura 4 obtemos os itens em forma de pilha, organizados de cima para baixo abedecendo o formato de colunas, como passado na propriedade citada.

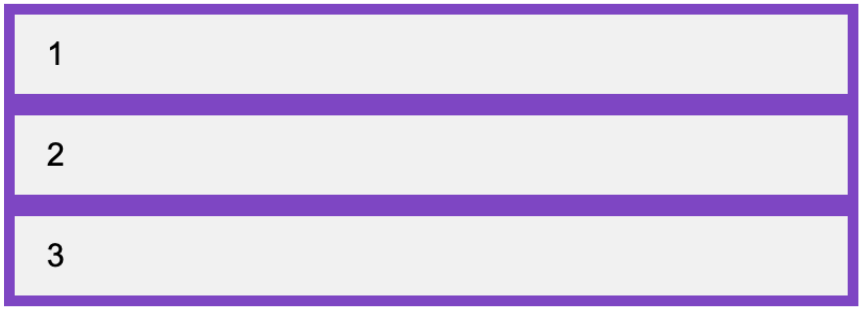

Figura 4 – Flex direction column

Fonte: O autor

Já o valor **column-reverse** empilha os itens verticalmente também. Contudo de baixo para cima:

```
1  .flex-container {
2    display: flex;
3    flex-direction: column-reverse;
4  }
```

Resultado:

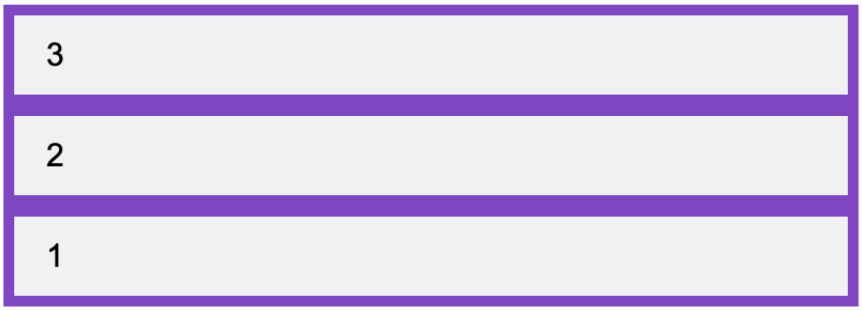

Figura 5 – Flex direction column reverse

Fonte: O autor

Muito semelhante ao **column** ou o **column-reverse** o valor **row** e o **row-reverse** executam função muito parecidas mas os itens são enfileirados:

```
1  .flex-container {
2    display: flex;
3    flex-direction: row;
4  }
```

Resultado:

Figura 6 – Flex direction row

Fonte: O autor

```
1  .flex-container {
2    display: flex;
3    flex-direction: row-reverse;
4  }
```

Resultado:

Figura 7 – Flex direction roe reverse

Fonte: O autor

2.2.3 Flex wrap

A propriedade **flex-wrap** especifica se os itens devem ser agrupados ou não. Os exemplos abaixo contém 12 itens, os quais podem ou não ser agrupados mediante ao tamanho do container pai.

```
1  .flex-container {
2    display: flex;
3    flex-wrap: wrap;
4  }
```

Resultado:

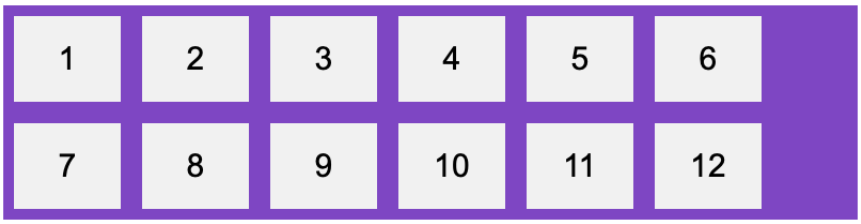

Figura 8 – Flex wrap

Fonte: O autor

```
1  .flex-container {
2    display: flex;
3    flex-wrap: nowrap;
4  }
```

Resultado:

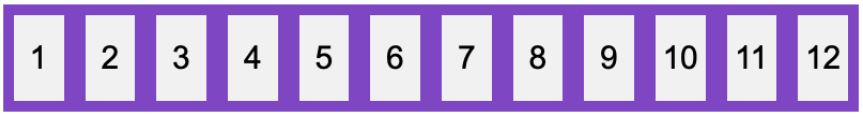

Figura 9 – Flex nowrap

Fonte: O autor

O **flex-wrap** também possui o seu reverso. Logo, usando a propriedade **wrap-reverse** os itens são empilhados na ordem inversa.

```
1  .flex-container {
2    display: flex;
3    flex-wrap: wrap-reverse;
4  }
```

Resultado:

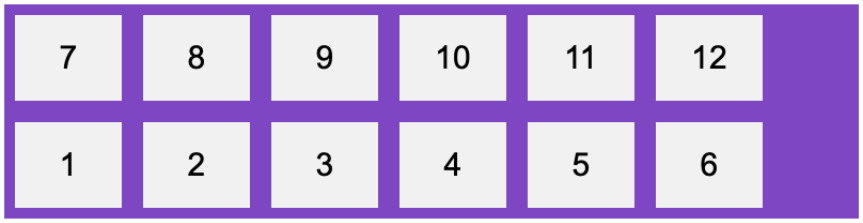

Figura 10 – Flex wrap reverse

Fonte: O autor

2.2.4 Alinhamento de itens na horizontal

A propriedade **justify-content** é usada para alinhar os itens em um container, seus valores podem ser **center**, **flex-start** e **flex-end**:

```
1  .flex-container {
2    display: flex;
3    justify-content: center;
4  }
```

Resultado:

Figura 11 – Justify content center

Fonte: O autor

O valor **flex-start** alinha os itens no início do container (este é o padrão):

```
1  .flex-container {
2    display: flex;
3    justify-content: flex-start;
4  }
```

Resultado:

Figura 12 – Justify content Flex Start

Fonte: O autor

O valor **flex-end** alinha os itens no final do conteiner:

```
1  .flex-container {
2    display: flex;
3    justify-content: ;
4  }
```

Resultado:

Figura 13 – Justify content Flex End

Fonte: O autor

2.2.5 Alinhamento de itens na vertical

Ao trocarmos a propriedade **justify-content** por **align-items** obtemos a possibilidade do alinhamento de itens na verticial sendo que os valores continuam os mesmos do **justify-content**

```
1  .flex-container {
2    display: flex;
3    align-items: center;  /* flex-start  ou flex-end */
4  }
```

2.3 Conclusão

Podemos aprender neste capítulo algumas das propriedades que vamos utilizar durante o curso. Contudo, Flexbox contém outras propriedades que podem ajudar no desenvolvimento de projetos de forma simples. Assim, caso haja alguma dúvida ou necessite de alguma propriedade que não foi listada neste texto, recomendo este link (https://origamid.com/projetos/flexbox-guia-completo/) o qual contém todos as propriedade juntamente com exemplos de seu uso.

Referências

BUYTAERT, D. *Redesigning a Website Using CSS Grid and Flexbox*. 2018. Disponível em: <https://dzone.com/articles/redesigning-a-website-using-css-grid-and-flexbox>. Citado na página 4.

STOJANOV, D. *A Visual Guide to CSS3 Flexbox Properties*. 2015. Disponível em: <https://scotch.io/tutorials/a-visual-guide-to-css3-flexbox-properties>. Citado 2 vezes nas páginas 2 e 3.