CSS3 INTRODUÇÃO

Uma imagem contendo Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto

Descrição gerada automaticamente com confiança média

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Texto, Linha do tempo

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

**Definição e seletores**

Após a criação do HTML a necessidade de formatar as páginas ficou evidente, assim, em 1996, foi criada a linguagem de estilo que conhecemos por CSS.

A sintaxe é bem simples e pode ser explicada com a frase "você cria regras de estilo para elementos ou grupos de elementos".

Vamos usar um elemento HTML que vimos anteriormente, a âncora <a>, para exemplificar.

Uma regra CSS é representada por um seletor ou um grupo de seletores, no nosso caso é o <a>, então dentro de um par de chaves adicionamos as declarações, no exemplo acima estamos alterando cor e tamanho da fonte dessa âncora, as declarações são formadas por uma propriedade e um valor.

Percebam que podemos colocar vários seletores em uma regra separando-os por vírgula.

E há um último detalhe nesse exemplo: a pseudo-classe. Elementos HTML sofrem alterações causadas pela interação do usuário, como mover o mouse por cima ou clicar nesse elemento.

O *a:hover* do exemplo significa que a âncora também terá essa aparência quando o usuário passar o mouse por cima de um *hyperlink*.

**ID x Classe**

No exemplo anterior criamos uma regra que altera um elemento HTML diretamente, mas isso significa que todos os elementos <a> ficarão com aquela aparência, e normalmente temos sites mais complexos que precisam de várias regras diferentes para elementos iguais.

Para ficar mais tangível vamos relembrar um pouco o site que começamos a fazer no módulo passado, ele tinha vários elementos header, mas não vamos querer que o header principal tenha a mesma formatação que o header de uma postagem, é aí que entram os IDs e Classes.

O seletor que vimos no primeiro exemplo é um seletor de tipo, pois ele representa um elemento HTML, e com IDs e Classes podemos representar qualquer tipo de elemento mas há algumas diferenças entre eles:

ID: é representado pelo símbolo # (hash) seguido de um nome para esse ID.

Classe: a classe é representada de forma parecida do ID, mas é precedida por um ponto em vez do hash.

E a diferença mais importante entre eles é a forma como devem ser usados: o ID só pode ser usado uma vez em uma página HTML enquanto a classe não tem restrições.

**Exercício**

Vamos adicionar algumas classes no nosso site e alterar alguns elementos, mas antes precisamos adicionar um arquivo CSS a nossa página.

No módulo de HTML descobrimos que podemos adicionar CSS de duas formas, com o elemento *style*, e assim suas regras ficarão no arquivo HTML, ou podemos criar um arquivo CSS e adicioná-lo na página através do elemento *link*, e é essa forma que usaremos.

Crie um elemento *link* dentro do head do seu arquivo e adicione os atributos *rel="stylesheet"* e *href="style.css"*, o *rel* denota o tipo de arquivo que estamos incluindo na página e o *href* é o caminho para o arquivo. E na mesma pasta do arquivo *HTML* crie um arquivo chamado *style.css*.

Agora sim vamos ao CSS, adicione um ID #title ao h1 da página, pois queremos que ele seja único, e depois adicione as classes .subtitle e .post\_title ao h2 e h3, respectivamente.

No arquivo CSS vamos mudar a cor desses três títulos, e depois alterar o tamanho da fonte do título da postagem.

**Box-model**

Quando estamos criando o layout de um site o navegador representa cada elemento HTML  como uma caixa retangular, isso é o box-model. E com CSS nós alteramos a aparência dessa caixa (largura, altura, cor de fundo, etc.). Essa caixa é composta por 4 áreas: o conteúdo, o padding, a borda e a margem.

* As margens (margin) são espaçamentos entre elementos;
* As bordas (border) ;
* O padding é um espaçamento entre as bordas e o conteúdo, a diferença para as margens é que declarações de imagem de fundo funcionam nele;
* O conteúdo (content) é o que o seu bloco representa, um texto, uma imagem, um vídeo;

Diagrama

Descrição gerada automaticamente

**Exercício**

Para enxergamos o box-model vamos adicionar cores e bordas a alguns elementos.

Primeiro adicionaremos uma cor de fundo para a visualização ficar mais fácil, usaremos a propriedade *background* com o valor *#fcfcfc* no elemento *body*.

Depois vamos adicionar uma classe ao <article>, pode ser .post, e então vamos colocar a cor branca de fundo com a propriedade *background* e o valor *#FFF*. Agora conseguimos enxergar o *content* do *box-model*.

Vamos adicionar um *padding* de 10 pixels neste mesmo *article*. Perceberam o espaçamento que surgiu em volta do nosso conteúdo?

Agora adicionamos um borda mais escura a ele com a propriedade border. Vou falar mais detalhadamente sobre border mais a frente, mas por enquanto vamos deixar essa borda com 3 pixels de largura, o contorno sólido e a cor azul.

E por último vamos adicionar uma margem do lado de fora do post com a propriedade *margin* e o valor 10 pixels.

E agora inspecionando o nosso elemento conseguimos todas aquelas camadas citadas antes: o conteúdo em azul, o *padding* em verde, as bordas em marrom e as margens em laranja.

E já que começamos a falar sobre bordas e cor de fundo, no próximo vídeo vamos nos aprofundar nessas propriedades.

Texto

Descrição gerada automaticamente Texto, Linha do tempo

Descrição gerada automaticamente

**Estilizando elementos**

Agora que entendemos o box-model podemos focar em deixar nosso site mais bonito, então vamos repassar pelas propriedades já citadas:

**Padding e Margin**

Anteriormente usamos o *padding* e o *margin* da forma mais básica, com apenas um valor, mas eles são mais poderosos que isso. Se quisermos atribuir tamanhos diferentes para cada lado do *box* nós podemos, e vamos ver três formas de fazer isso.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

A primeira é colocando um valor para as partes superior e inferior e depois para os lados esquerdo e direito.

O valor de 10 *pixels* se refere ao eixo Y, ou partes superior e inferior, e os 5 *pixels* se referem aos lados esquerdo e direito.

 A segunda forma é dando valores para cada lado do *box*.

Texto

Descrição gerada automaticamente

Então começamos pelo topo com 15 pixels, passamos o lado direito com 10 pixels, depois para a parte inferior com 5 pixels e por último o lado esquerdo com 0, e sempre nessa ordem.

Uma boa dica também é que quando o valor for 0 não precisamos não precisamos colocar a unidade.

A terceira forma é com as propriedades específicas para cada lado, até agora tínhamos visto atalhos para essas propriedades.

Essa opção é mais usada quando temos o mesmo valor para 3 lados, e o quarto precisa ter um valor diferente, então usamos o padding com apenas um valor e uma dessas opções para representar o lado diferente.

**Background**

Texto

Descrição gerada automaticamente

A propriedade *background* também é um atalho para várias propriedades, mas isso vocês podem absorver aos poucos, e uma boa opção de leitura é a documentação do MDN – Mozilla Developer Network

Por enquanto veremos apenas como mudar a cor de fundo.

E aqui temos 3 formas de colocar uma cor de fundo, e ainda existem outras.

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

A primeira é pelo nome da cor em inglês, a segunda é pelo código hexadecimal e a terceira é usando apenas o atalho *background*.

**Border**

Vimos que a propriedade *border* pode ter 3 valores: a largura, a cor e o estilo, mas existem algumas particularidades nisso.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

A largura pode ser usada com várias unidades, como px, em e mm. A cor pode ser atribuída pelo nome ou por um código hexadecimal, assim como fizemos com o *background*, e o estilo é representada por palavras-chave, vamos ver algumas delas:

**solid**: mostra uma borda simples e reta;

**dotted**: são bolinhas com um pequeno espaçamento entre elas;

**dashed**: forma uma linha tracejada.

E aproveitando que mostrei esse código temos que falar sobre como separar a estilização dos lados de uma borda.

Texto

Descrição gerada automaticamente

E se você não quiser usar a propriedade *border* existem as propriedade específicas para cada aspecto de uma borda, são elas *border-width* para a largura, *border-color* para a cor e *border-style* para o estilo.

Aqui temos o mesmo código anterior de duas formas diferentes, a primeira com o atalho *border* e a segunda com cada propriedade específica.

E depois disso podemos juntar os lados com os aspectos de uma borda e criar uma regra mais específica ainda.

Texto

Descrição gerada automaticamente

Border-radius

E a última propriedade é o *border-radius*, ele permite arredondar os cantos de um elemento. Podemos usar várias unidades, mas as mais comuns são os pixels e a porcentagem.

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

Colocando apenas um valor mudamos todos os cantos do elemento, mas seguindo aquela mesma ordem que vimos no *padding* e *margin* - topo, direita, inferior e esquerda -  conseguimos alterar cada canto separadamente.

Se quiser transformar o quadrado em círculo, basta colocar border-radius de 50% e objetivo será atingido.

**Exercício**

Neste exercício vamos deixar o nosso site um pouco mais bonito usando as propriedades que acabamos de ver.

Vamos aumentar o padding para 15 pixels e colocar uma margem de também de 15 pixels só na parte de baixo do post.

Quando olhamos para os textos percebemos que os espaçamentos estão diferentes do restante do post, então vamos padronizar isso.

Texto

Descrição gerada automaticamente

No título do post vamos retirar todas as margens para depois colocar apenas uma margem inferior de 15 pixels. E no corpo do post precisamos adicionar uma classe e remover todas as margens para depois adicionar uma margem superior de 15 pixels.

Texto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamente

Podemos manter o background branco, mas vamos diminuir a largura das bordas para 2 pixels e mudar a cor para a mesma do texto - #505050 - e por último adicionaremos um border-radius, 5 pixels são suficientes. Podemos adicionar esse mesmo de valor de border-radius na imagem, para isso vamos acrescentar uma class a imagem antes.

 É uma boa prática manter as margens sempre na mesma direção. Então, se quiser dar espaçamentos entre elementos, escolha colocar a margem apenas acima, ou apenas abaixo.

**Estilizando textos**

Já sabemos que podemos mudar cor e tamanho de algumas fontes, e agora vamos nos aprofundar nisso.

**font-family**

Com o font-family podemos alterar a fonte dos nossos textos, como uma fonte da internet ou uma que esteja instalada no nosso computador, mas vamos nos ater às fontes seguras, chamadas de web safe fonts.

Texto

Descrição gerada automaticamente

Essas fontes são chamadas assim pois são encontradas em quases todos os sistemas e podem ser usadas sem preocupação.

No exemplo acima, colocou duas fontes para, caso a primeira não funcione, a outra servirá como backup.

**font-size**

O font-size nos ajuda a mudar o tamanho do texto, existem algumas unidades de medida para ele mas por enquanto os pixels são suficientes para nós.

Texto

Descrição gerada automaticamente

**font-style**

Usamos o font-style para tornar um texto itálico, na maioria das vezes você usará apenas o valor *italic* para ele, mas se precisar tirar o itálico de um texto você pode usar o valor *norma*l que é o padrão como a fonte foi desenhada

Texto

Descrição gerada automaticamente

Temos que avaliar se nosso navegador tem suporte ao italic, pois do contrário ele vai forçar a inclinação e a aparência não será agradável.

**font-weight**

Texto

Descrição gerada automaticamente

Altera o peso do texto. Existem várias palavras chaves para o valor e algumas fontes têm valores numéricos também, mas são usados quando temos fontes mais complexas, que possuem vários pesos.

Para as fontes mais simples, o valor normal (comum) e o bold (negrito) já resolvem.

Text-transform: alterna o texto entre maiúsculo minúsculo

Texto

Descrição gerada automaticamente

Uppercase: coloca todo o texto em caixa alta

Lowercase: coloca todo o texto em minúsculo

Capitalize: coloca a primeira letra em maiúsculo

Text-decoration: utilizado para dar destaque em algum texto, pois ele coloca linhas

Texto

Descrição gerada automaticamente

Underline: linha abaixo da palavra

Overline: linha acima da palavra

Line-through: linha no centro, cortando a palavras

ESTILIZANDO LINHAS

List-style-type: utilizada para alterar o marcador das listas. Existem várias opções.

Texto

Descrição gerada automaticamente

Square: alterando o símbolo de uma lista não ordenada para quadrado

Upper-roman: alterando o marcador da lista ordenada para algorismos romanos

No terceiro itetm: altera o marcador de uma lista não ordenada por um símbolo (IF44D = emoji)

Podemos também utilizar imagens como marcador, bastando utilizar;

List-style-image: url(“xxxxx. Jpg”);

None: remove o marcadores

CSS funciona como cascata: <a> é um elemento dentro do <li> que é elemento do <ul>:

Diagrama

Descrição gerada automaticamente com confiança baixa

É possível estilizar o elemento seguindo este fluxo:

Texto

Descrição gerada automaticamente

PROPRIEDADES DE DIMENSÕES E ALINHAMENTO

São muito importantes no layout. Existem algumas propriedades:

Width: largura. Valor em px ou %. Lembrar de ajustar o conteúdo para que caiba dentro do espaço limitado, como uma imagem, devendo atribuir à ela o width: 100%.

Height: altura

Max-width: largura máxima que os elementos podem ter.

Max-eight: altura máxima do elemento

Margin: para colocar espaçamento entre elementos. Tem o valor “auto” criando um espaçamento de cada lado, mantendo uma margem sempre adequada ao tamanho da tela.

Text align: para alinhar textos, left, right, center ou justifie

POSICIONANDO ELEMENTOS COM FLEXBOX EM CSS

Apresentar os fundamentos e aplicações da propriedade flexbox na criação de layouts responsivos, sem a necessidade a definição de valores fixos.

INTRODUÇÃO AO FLEXBOX

Objetivos da aula

1. Conhecer a estrutura básica

2. Entender a diferença entre Flex Container e Flex Item

3. Conhecer inicialmente alguma propriedades

Suporte:

Foi projetado como um modelo de layout unidimensional e como um método que pode oferecer distribuição de espaço entre itens em uma interface e recursos de alinhamento.

Interface gráfica do usuário

Descrição gerada automaticamente

FLEX CONTAINER (COMO SE FOSSE O PAI) E FLEX ITEM É O FILHO DESTE CONTAINER

É a tag que envolve os itens, será nela que iremos aplicar a propriedade “display: flex”. Transforma todos os seus itens filhos em flex itens.

Gráfico

Descrição gerada automaticamente

Pode ser feita em qualquer tipo de tag,seja div, h1, h2... até mesmo um link.

Propriedades relacionadas:

* Display: inicializador do container
* Flex-direction: vai fazer o direcionamento dos itens, seja em linha ou coluna
* Flex-wrap: quebra de linha
* Flex-flow: abreviação para o Direction ou wrap
* Justify-content: alinhamento dos itens do container de acordo com sua direção
* Align-items: alinha os itens de acordo com o eixo do container
* Align-content: alinhar as linhas do container

FLEX-ITEM

São os elementos filhos diretos do Flex Container. E também podem ser aplicados em propriedade display e se tornar Flex Container.

Uma imagem contendo Gráfico

Descrição gerada automaticamente

Propriedades relacionadas:

● flex-grow: define o fator de crescimento

● flax-basis: define o tamanho inicial do item antes da distribuição do espaço restante dentro do container

● flex-shrink: define capacidade de redução

● flex: abreviação para os anteriores

● order: relacionado à ordem de distribuição

● align-self: define o alinhamento de um item específico

FUNDAMENTOS

DISPLAY FLEX

Sugestões:

✔ HTML Snippets

✔ Live HTML Previewer

Linha do tempo

Descrição gerada automaticamente Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Objetivo da aula

1. Conhecer e aplicar a propriedade de inicialização do flex container. Vai ser a base para toda a construção do nosso curso.

Display: flex;

Torna a tag um elemento do tipo flex container, e assim automaticamente todos os seus filhos diretos desta tag, tornam-se em flex items. Serve para qualquer tag

Gráfico de barras

Descrição gerada automaticamente com confiança média

FLEX DIRECTION:

Objetivos:

1. Entender o comportamento padrão de orientação horizontal de um flex container

2. Aprender a modificar a orientação horizontal

Flex-direction: É a propriedade que estabelece o eixo principal do container, definindo assim a direção que os flex items são colocados no flex container.

EIXOS:

Row (padrão): direção do texto esquerda para direita

Uma imagem contendo Ícone

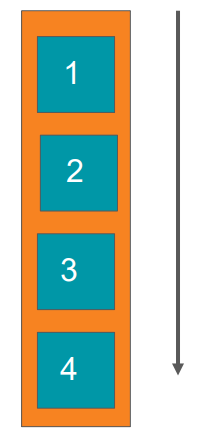
Descrição gerada automaticamente

Row-reverse: sentido oposto à direção do texto:

Uma imagem contendo Ícone

Descrição gerada automaticamente

Column: ordenação de cima para baixo em coluna única



Column-reverse: ordenação inversa, de baixo para cima

Interface gráfica do usuário

Descrição gerada automaticamente

Exercício prático:

<!doctype html>

<html>

<head>

<meta charset='utf-8'>

<title>Flex Direction</title>

<style>

.flex-container{

margin: 0;

padding: 0;

display: flex;

list-style: none;

}

.flex-item{

background: blue;

height: 50px;

width: 50px;

line-height: 50px;

font-size: 20px;

color: white;

text-align: center;

margin: 5px;

}

.row{

flex-direction: row;

}

.row-reverse{

flex-direction: row-reverse;

}

.row-reverse li{

background-color: red;

}

.column{

float: left;

flex-direction: column;

}

.column li{

background: orange;

}

.column-reverse{

float: right;

flex-direction: column-reverse;

}

.column-reverse li{

background: purple;

}

</style>

</head>

<body>

<ul class="flex-container row">

<li class="flex-item">1</li>

<li class="flex-item">2</li>

<li class="flex-item">3</li>

<li class="flex-item">4</li>

<li class="flex-item">5</li>

</ul>

<ul class="flex-container row-reverse">

<li class="flex-item">1</li>

<li class="flex-item">2</li>

<li class="flex-item">3</li>

<li class="flex-item">4</li>

<li class="flex-item">5</li>

</ul>

<ul class="flex-container column">

<li class="flex-item">1</li>

<li class="flex-item">2</li>

<li class="flex-item">3</li>

<li class="flex-item">4</li>

<li class="flex-item">5</li>

</ul>

<ul class="flex-container column-reverse">

<li class="flex-item">1</li>

<li class="flex-item">2</li>

<li class="flex-item">3</li>

<li class="flex-item">4</li>

<li class="flex-item">5</li>

</ul>

</body>

<html>

Resultado:

Linha do tempo

Descrição gerada automaticamente

FLEX WRAP: É a propriedade que define se os itens devem ou não quebrar a linha. Por padrão eles não quebram linhas, isso faz com que os flex itens sejam compactados além do limite do conteúdo e do container

Nowrap: é o padrão, não permite quebra de linha

Interface gráfica do usuário, Aplicativo, Ícone

Descrição gerada automaticamente

Wrap: permite a quebra de linha assim que um dos flex itens não puder mais ser compactado

Ícone

Descrição gerada automaticamente

Wrap-reverse

Wrap: permite a quebra de linha assim que um dos flex itens não puder mais ser compactado, porém na direção contrária da linha, acima

Interface gráfica do usuário

Descrição gerada automaticamente com confiança média

Exercício prático:

<!doctype html>

<html>

<head>

<meta charset='utf-8'>

<title>Flex Wrap</title>

<style>

.flex-container{

display: flex;

height: 150px;

border: 2px solid black;

max-width: 310px

}

.item{

font-size: 24px;

height: 50%;

text-align: center;

min-width: 100px;

line-height: 50px;

}

.nowrap{

flex-wrap: nowrap;

}

.wrap{

flex-wrap: wrap;

}

.wrap-reverse{

flex-wrap: wrap-reverse;

}

.blue{

background-color: blue;

}

.green{

background-color: lightgreen;

}

.orange{

background-color: orangered;

}

</style>

</head>

<body>

<h1>no wrap</h1>

<div class="flex-container nowrap">

<div class="item orange">1</div>

<div class="item green">2</div>

<div class="item blue">3</div>

<div class="item orange">4</div>

</div>

<h1>wrap</h1>

<div class="flex-container wrap">

<div class="item orange">1</div>

<div class="item green">2</div>

<div class="item blue">3</div>

<div class="item orange">4</div>

</div>

<h1>wrap reverse</h1>

<div class="flex-container wrap-reverse">

<div class="item orange">1</div>

<div class="item green">2</div>

<div class="item blue">3</div>

<div class="item orange">4</div>

</div>

</body>

<html>

Resultado:

Gráfico, Gráfico de cascata

Descrição gerada automaticamente

FLEX FLOW

É um atalho para as propriedades flex-direction e flex-wrap. Porém seu uso não é tão comum, visto que, quando mudamos o flex-direction para column, mantemos o padrão do flex-wrap que é nowrap.

<!doctype html>

<html>

<head>

<meta charset='utf-8'>

<title>Fundamentos Flex Flow</title>

<style>

.flex-container{

display: flex;

border: 1px solid black;

max-width: 500 px;

margin-bottom: 10px;

}

.default{

flex-flow: row nowrap;

}

.wrap{

flex-flow: row wrap;

}

.wrap-reverse{

flex-flow: row wrap-reverse;

}

.r-nowrap{

flex-flow: row-reverse-nowrap;

}

.item{

background-color: blueviolet;

margin: 5px;

color: white;

text-align: center;

width: 100px;

height: 50px;

line-height: 50px;

}

</style>

</head>

<body>

<h1>Default row nowrap</h1>

<div class="flex-container default">

<div class="item">1</div>

<div class="item">2</div>

<div class="item">3</div>

<div class="item">4</div>

<div class="item">5</div>

</div>

<h1>wrap</h1>

<div class="flex-container wrap">

<div class="item">1</div>

<div class="item">2</div>

<div class="item">3</div>

<div class="item">4</div>

<div class="item">5</div>

<div class="item">6</div>

</div>

<h1>wrap reverse</h1>

<div class="flex-container wrap-reverse">

<div class="item">1</div>

<div class="item">2</div>

<div class="item">3</div>

<div class="item">4</div>

<div class="item">5</div>

<div class="item">6</div>

</div>

<h1>row-reverse-nowrap</h1>

<div class="flex-container r-nowrap">

<div class="item">1</div>

<div class="item">2</div>

<div class="item">3</div>

<div class="item">4</div>

<div class="item">5</div>

<div class="item">6</div>

</div>

</body>

<html>

Interface gráfica do usuário, Aplicativo, Teams

Descrição gerada automaticamente

JUSTIFY CONTENT:

Essa propriedade vai se encarregar de alinhar os itens dentro do container de acordo com a direção pretendida e tratar da distribuição de espaçamento entre eles. OBS: caso seus itens esteja ocupando 100% de todo o container, ela não se aplica

As variações:

● flex-start: início do container.

● flex-end: final do container.

● center: ao centro do container.

● space-between: cria um espaçamento igual entre os elementos. Mas tem um destaque: ele vai pegar o primeiro elemento, vai colocar muito próximo ao início deste container, ou seja, próximo a borda esquerda, e o outro elemento será levado para a margem à direita, ou seja, no final.

● space-around: os espaçamentos do meio são duas vezes maiores que o inicial e final.

<!doctype html>

<html>

    <head>

        <meta charset='utf-8'>

        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <title>Fundamentos Justify-content</title>

        <style>

            .container{

                max-width: 400px;

                border: 1px solid black;

                margin: 0;

                display: flex;

                margin-bottom: 10px;

            }

            .item{

            background-color: blueviolet;

            margin: 5px;

            color: white;

            text-align: center;

            }

            .flex-start{

                justify-content: flex-start;

                }

            .flex-end{

                justify-content: flex-end;

            }

            .flex-center{

                justify-content: center;

            }

            .space-between{

                justify-content: space-between;

            }

            .space-around{

                justify-content: space-around;

            }

        </style>

    </head>

    <body>

        <p>flex-start</p>

        <section class="container flex-start">

            <div class="item">1</div>

            <div class="item">item 2</div>

            <div class="item">outro item</div>

        </section>

        <p>flex-end</p>

        <section class="container flex-end">

            <div class="item">1</div>

            <div class="item">item 2</div>

            <div class="item">outro item</div>

        </section>

        <p>flex-center</p>

        <section class="container flex-center">

            <div class="item">1</div>

            <div class="item">item 2</div>

            <div class="item">outro item</div>

            <div class="item">4</div>

            <div class="item">5</div>

        </section>

        <p>space-between</p>

        <section class="container space-between">

            <div class="item">1</div>

            <div class="item">item 2</div>

            <div class="item">outro item</div>

            <div class="item">4</div>

            <div class="item">5</div>

        </section>

        <p>space-around</p>

        <section class="container space-around">

            <div class="item">1</div>

            <div class="item">item 2</div>

            <div class="item">outro item</div>

            <div class="item">4</div>

            <div class="item">5</div>

        </section>

    </body>

    <html>

Resultado:

Texto

Descrição gerada automaticamente

ALIGN ITEMS

Trata do alinhamento dos flex itens de acordo com o eixo do container. O alinhamento é diferente para quando os itens estão em colunas ou linhas. Permite o alinhamento central no eixo vertical.

Tipos de alinhamento

● center: alinhamento dos itens ao centro

● stretch: padrão, e os flex itens cresçam igualmente

● flex-start: alinhamento dos itens no início

● flex-end: alinhamento dos itens no final

● baseline: alinhamento de acordo com a linha base da tipografia dos itens

Interface gráfica do usuário, Diagrama, Aplicativo, Teams

Descrição gerada automaticamente Interface gráfica do usuário, Aplicativo, Teams

Descrição gerada automaticamente

Stretch é importante quando queremos manter o mesmo padrão das colunas, acompanhando a que tem o maior conteúdo.

Zerar flex, remove algumas propriedades relacionadas ao flex items

     }

        .central{

            height: 400px;

            justify-content: center;

        }

        .central .item{

            flex: 0;

            padding: 20px;

        }

    </style>

</head>

<body>

    <p>Alinhamento central ao container</p>

    <section class="container central center">

        <div class="item">adf afasd sdfash çlhasçh fçad çsdfhaç hfçahf çash fçash fçashdf çashfçasdf</div>

    </section>

</body>

</html>

Diagrama

Descrição gerada automaticamente

ALIGN CONTENT

É a propriedade responsável por tratar o alinhamento das linhas do container em relação ao eixo vertical do container. Vai de fato trabalhar as linhas do container.

Precisamos que:

● O container utilize quebra de linhas

● A altura do container seja maior que a soma das linhas dos itens

Tipos de alinhamento

● center: alinhamento dos itens ao centro

● stretch: é o padrão e os flex itens crescem igualmente. Parte do maior item como parâmetro

● flex-start: alinhamento dos itens no início

● flex-end: alinhamento dos itens no final

● space-between: cria um espaçamento igual entre os elementos

● space-around: os espaçamentos do meio são duas vezes maiores que o inicial e final

FLEX GROW

Define a proporcionalidade de crescimentos dos itens, respeitando o tamanho de seus conteúdos internos.

OBS: não irá funcionar caso tenhamos adicionado justify-content ao nosso flex container.

Quando falamos de flex grow, temos que ter em mente que ele só aceita números. Ou vai ser zero ou algum número relacionado à proporção de crescimento.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Flex Grow</title>

    <style>

        .flex-container{

            max-width: 300px;

            display: flex;

            border: 2px solid black;

            margin-bottom: 20px;

        }

        .item{

            background-color: aqua;)

            padding: 10px;

            margin: 5px;

            text-align: center;

            color: red;

        }

        .fg-0{

            flex-grow: 0;

        }

        .fg-1{

            flex-grow: 1;

        }

        .fg-2{

            flex-grow: 2;

        }

        .fg-3{

            flex-grow: 3;

        }

    </style>

</head>

<body>

    <p>flex grow:0</p>

    <div class="flex-container">

        <div class="item fg-0">A</div>

        <div class="item fg-0">B</div>

        <div class="item fg-0">C</div>

        <div class="item fg-0">D</div>

    </div>

    <p>flex grow:1</p>

    <div class="flex-container">

        <div class="item fg-1">A</div>

        <div class="item fg-1">B</div>

        <div class="item fg-1">C</div>

        <div class="item fg-1">D</div>

    </div>

    <p>flex grow: variado</p>

    <div class="flex-container">

        <div class="item fg-1">A</div>

        <div class="item fg-2">B</div>

        <div class="item fg-1">C</div>

        <div class="item fg-3">D</div>

    </div>

</body>

</html>

Resultado:

Interface gráfica do usuário, Tabela

Descrição gerada automaticamente com confiança média

FLEX-BASIS

É a propriedade que estabelece o tamanho inicial do item antes da distribuição de espaço restante dentro dele, usando como base o conteúdo interno disposto. É como se eu pegasse o tamanho mínimo que este item vai precisar ter vai fazer a distribuição da sobra de espaço antes e após o conteúdo.

Valores possíveis:

- auto: caso o item não tenha tamanho, este será proporcional ao conteúdo do item

- px, %, em,...: são valores exatos previamente definidos, definindo o valor mínimo que o conteúdo vai ter. Se crescer muito, ele vai vazar e perde o sentido.

- 0 (zero): terá relação com a definição do flex-grow

!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Flex Basis</title>

    <style>

        .flex-container{

            max-width: 500px;

            display: flex;

            border: 1px solid black;

        }

        .items {

            background-color: aqua;)

            padding: 10px;

            margin: 5px;

            text-align: center;

            color: red;

        }

        .bauto{

            flex-basis: auto;

        }

        .fg-1{

            flex-grow: 1;

        }

        .fg-2{

            flex-grow: 2;

        }

        .b-o{

            flex-basis: 0;

        }

        .w100{

            flex-basis: 100px;

        }

    </style>

</head>

<body>

    <p>flex Basis em auto</p>

    <div class="flex-container">

       <div class="items bauto fg-1">A</div>

       <div class="items bauto fg-1">BBB</div>

       <div class="items bauto fg-1">C</div>

       <div class="items bauto fg-2">DEF</div>

    </div>

    <p>flex Basis em zero</p>

    <div class="flex-container">

       <div class="items b-0 fg-1">A</div>

       <div class="items b-0 fg-1">BBB</div>

       <div class="items b-0 fg-1">C</div>

       <div class="items b-0 fg-2">DEF</div>

    </div>

    <p>flex Basis + width</p>

    <div class="flex-container">

       <div class="items w100 fg-2">A</div>

       <div class="items w100 fg-2">BBB</div>

       <div class="items w100 fg-2">C</div>

       <div class="items w100 fg-2">DEF</div>

    </div>

</body>

</html>

Calendário

Descrição gerada automaticamente com confiança baixa

FLEX SHRINK

É a propriedade que estabelece a capacidade de redução ou compressão do tamanho de um item. Nós teremos o tamanho do item, os comportamentos esperados e vai permtir a redução dos itens proporcionalmente, ou impede a redução, caso seja estabelecido valor zero

<!DOCTYPE html>

<html lang="pt-BR">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Flex Shrink</title>

    <style>

        .flex-container{

            display: flex;

            border: 1px solid black;

            max-width: 350px;

            margin-bottom: 10px;

        }

        .item{

            background-color: orangered;

            font-size: 24px;

            text-align: center;

            margin: 5px;

        }

        .s-1{

            flex-shrink: 1;

            flex-basis: 100px;

        }

        .fg-1{

            flex-grow: 1;

        }

        s-2{

            flex-shrink: 2;

            flex-basis: 100px;

        }

        s-3{

            flex-shrink: 3;

            flex-basis: 100px;

        }

    </style>

</head>

Uma imagem contendo Linha do tempo

Descrição gerada automaticamente

FLEX:

Esta propriedade é um atalho ou abreviação de escrita para as propriedades: grow, shrink e basis e exatamente nesta ordem. Vai conseguir definir valores que colocaríamos separadamente simplesmente escrevendo flex. Ao definir o flex como 1, zero, auto, estou dizendo que meu grow será 1, shrink zero e o basis auto.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Flex</title>

    <style>

        .flex-container{

            max-width: 300px;

            display: flex;

            border: 2px solid black;

            margin-bottom: 20px;

        }

        .item{

            background-color: aqua;)

            padding: 10px;

            margin: 5px;

            text-align: center;

            color: red;

        }

        .flex-1{

            flex: 1;

            /\* grow =1, shrink = 1, basis =0 \*/

        }

        .flex-2{

            flex: 2;

            /\* grow =2, shrink = 1, basis =0 \*/

        }

    </style>

        <p>flex: 1</p>

        <div class="flex-container">

       <div class="item flex-1">A</div>

       <div class="item flex-1">BBB</div>

       <div class="item flex-1">CC</div>

       <div class="item flex-1">D</div>

       <div class="item flex-1">EF</div>

    </div>

    <p>flex: mix</p>

    <div class="flex-container">

   <div class="item flex-2">A</div>

   <div class="item flex-2">BBB</div>

   <div class="item flex-1">CC</div>

   <div class="item flex-1">D</div>

   <div class="item flex-1">EF</div>

</div>

</head>

Calendário

Descrição gerada automaticamente

ORDER: lida diretamente com a ordenação dos itens.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Order</title>

    <style>

        .flex-container{

            max-width: 350px;

            display: flex;

            border:1px solid black;

            margin-bottom: 20px;

        }

        .item{

            flex: 1;

            background-color: brown;)

            color: white;

            text-align: center;

            font-size: 24px;

            margin: 5px;

        }

        .o-1{

            order: 1;

        }

        .o-2{

            order: 2;

        }

        .o-3{

            order: 3;

        }

        .o-4{

            order: 4;

        }

        .direction{

            flex-direction: column;

        }

    </style>

</head>

    <body>

        <p>order: 0</p>

            <div class="flex-container">

                <div class="item">1</div>

                <div class="item">2</div>

                <div class="item">3</div>

                <div class="item">4</div>

                <div class="item">5</div>

            </div>

        <p>order: diferentes</p>

            <div class="flex-container">

                <div class="item o-4">2</div>

                <div class="item o-1">4</div>

                <div class="item o-3">5n</div>

                <div class="item o-2">5m</div>

            </div>

        <h1>Column</h1>

        <p>order: 0</p>

            <div class="flex-container direction">

                <div class="item">1</div>

                <div class="item">2</div>

                <div class="item">3</div>

                <div class="item">4</div>

                <div class="item">5</div>

            </div>

        <p>order: diferentes</p>

            <div class="flex-container direction">

                <div class="item o-4">2</div>

                <div class="item o-1">4</div>

                <div class="item o-3">5n</div>

                <div class="item o-2">5m</div>

            </div>

    </body>

Gráfico, Gráfico de barras

Descrição gerada automaticamente

É possível definir o order com valor negativo.

ALIGN-SELF

É a propriedade que estabelece o alinhamento de modo individual sobre um determinado item. Importante que o container não tenha um aline-items definido.

Valores possíveis:

● auto: valor padrão, irá respeitar a definição de align-items do container

● flex-start: ao início do container

● flex-end: ao final do container

● center: relativo ao centro de acordo com o eixo

● stretch: ocupa todo os espaço relativo

● baseline: utiliza a linha base da tipografia

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Align Selg</title>

    <style>

        .flex-container{

            max-width: 350px;

            display: flex;

            border:1px solid;

            margin-bottom: 10px;

        }

        .align-container{

            align-items: center;

        }

        .item{

            flex: 1;

            background-color: brown;)

            color: white;

            text-align: center;

            font-size: 24px;

            margin: 5px;

        }

        .center{

            align-self: center;

        }

        .start{

            align-self: flex-start;

        }

        .end{

            align-self: flex-end;

        }

        .baseline{

            align-self: baseline;

        }

        .stretch{

            align-self: stretch;

        }

        .auto{

            align-self: auto;

        }

        .direction{

            flex-direction: column;

        }

    </style>

</head>

    <body>

        <h1>Row</h1>

        <p>Align Self Auto</p>

        <section class="flex-container align-container">

            <div class="item auto">AAA</div>

            <div class="item auto">123</div>

            <div class="item auto">asdfasfa asfdasd asdfasf asfasdfasdf</div>

            <div class="item auto">Item 4</div>

            <div class="item auto">Item 5</div>

        </section>

        <p>Align Self Diferentes</p>

        <section class="flex-container">

            <div class="item stretch">AAA</div>

            <div class="item auto">123</div>

            <div class="item end">asdfasfa asfdasd asdfasf asfasdfasdf</div>

            <div class="item center">Item 4</div>

            <div class="item start">Item 5</div>

        </section>

        <h1>Column</h1>

        <p>Align Self Auto</p>

        <section class="flex-container align-container direction">

            <div class="item auto">AAA</div>

            <div class="item auto">123</div>

            <div class="item auto">asdfasfa asfdasd asdfasf asfasdfasdf</div>

            <div class="item auto">Item 4</div>

            <div class="item auto">Item 5</div>

        </section>

        <p>Align Self Diferentes</p>

        <section class="flex-container direction">

            <div class="item stretch">AAA</div>

            <div class="item auto">123</div>

            <div class="item end">af</div>

            <div class="item center">Item 4</div>

            <div class="item start">Item 5</div>

    </body>

</html>

Diagrama

Descrição gerada automaticamente com confiança média

PROJETO:

ORGANIZANDO A INTERFACE

Tratar do alinhamento da página:

Para jogar o menu de links “Quem Somos”, “Serviços” e “Planos” para o canto esquerdo, vamos utilizar o justify-aline.

:: before = cria a barra antes do item: Uma imagem contendo Logotipo

Descrição gerada automaticamente

Código para criação desta barra:

Texto

Descrição gerada automaticamente

Código CSS completo até esta parte 1:

{

    margin: 0;

    padding: 0;

    font-family: 'Open Sans', sans-serif;

}

ul{

    list-style: none;

    margin: 0;

    padding: 0;

}

ul li a{

    text-decoration: none;

}

div img{

    display: block;

    width: 100%;

}

.flex-container{

    display: flex;

    max-width: 992px;

    margin: auto;

    width: 100%;

}

header{

    background-color: #122A57;

    height: 100px;

    display: flex;

    align-items: center;

    color: #fff;

}

header .list-items{

    display: flex;

    max-width: 260px;

    width: 100%;

    justify-content: space-between;

    align-items: center;

}

.list-items li a{

    color: #fff;

}

header .menu{

    justify-content: space-between;

}

.apresentacao{

    height: 60vh;

    align-items: center;

    justify-content: space-between;

}

.apresentacao .texto-apresentacao{

    min-height: 200px;

}

.texto-apresentacao h1{

    color: #122A57;

    font-size: 48px;

    margin-bottom: 10px;

}

.texto-apresentacao a{

    background-color: #122A57;

    color: #fff;

    text-align: center;

    border-radius: 30px;

    width: 220px;

    display: block;

    text-decoration: none;

    height: 50px;

    line-height: 50px;

    margin-top: 10px;

    margin-bottom: 10px;

}

#quem-somos{

   flex-direction: row-reverse;

   align-items: center;

   justify-content: space-between;

}

#quem-somos h2{

    font-size: 32px;

    color:#122A57;

    display: flex;

    margin-bottom: 20px;

}

#quem-somos h2::before{

    content: "";

    height: 50px;

    width: 5px;

    margin-right: 5px;

    background-color: #122A57;

    position: relative;

}

#quem-somos p{

    margin-bottom: 10px;

    width: 90%;

}

HTML Parte 1:

<!DOCTYPE html>

<html lang="pt-br">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Flex Turismos</title>

    <link rel="stylesheet" href="./style.css">

</head>

<body>

    <header>

        <div class="flex-container menu">

        <div><h1>FlexTurismos</h1></div>

        <ul class="list-items">

            <li><a href="#quem-somos">Quem Somos</a></li>

            <li><a href="#servicos">Serviços</a></li>

            <li><a href="#planos">Planos</a></li>

        </div>

        </ul>

    </header>

    <div class="flex-container apresentacao">

        <div class="texto-apresentacao">

            <h1>Flex <br>Turismos</h1>

            <p>O melhor serviço para você!</p>

            <a href="">Saiba Mais!</a>

        </div>

        <div>

            <div><img src="./images/0-main.png" alt="banner de apresentação"></div>

        </div>

    </div>

    <div class="flex-container apresentacao" id="quem-somos">

        <div>

            <h2>Quem somos</h2>

            <p>It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. </p>

            <p>The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English.</p>

        </div>

        <div>

            <img src="./images/1-quem-somos.png" alt="balcão de atendimento">

        </div>

    </div>

    <div>

        <div>

            <h2>Serviços</h2>

        </div>

        <div>

            <div>

                <div><img src="./images/icon-2.png" alt="hospedagens"></div>

                <p>Hospedagens</p>

                <a href="#">Comprar Agora</a>

            </div>

            <div>

               <div><img src="./images/icon-1.png" alt="pacote de viagens"></div>

               <p>Pacotes de viagens</p>

               <a href="#">Comprar Agora</a>

            </div>

            <div>

                <div><img src="./images/icon-3.png" alt="roteiros personalizados"></div>

                <p>Roteiros personalizados</p>

                <a href="#">Comprar Agora</a>

            </div>

        </div>

    </div>

    <div>

        <div>

            <h3>Plano 2</h3>

            <ul>

                <li>Suporte 24h</li>

                <li>Serviços de quarto</li>

                <li>Guia turístico</li>

            </ul>

            <a href="#">Saiba Mais!</a>

        </div>

        <div>

            <h3>Plano 2</h3>

            <ul>

                <li>Suporte 24h</li>

                <li>Serviços de quarto</li>

                <li>Guia turístico</li>

                <li>Roteiro de trilhas</li>

                <li>Serviço personalizado</li>

            </ul>

            <a href="#">Saiba Mais!</a>

        </div>

        <div>

            <h3>Plano 3</h3>

            <ul>

                <li>Suporte 24h</li>

                <li>Serviços de quarto</li>

                <li>Guia turístico</li>

                <li>Roteiro de trilhas</li>

                <li>Serviço personalizado</li>

                <li>Área Vip</li>

            </ul>

            <a href="#">Saiba Mais!</a>

        </div>

    </div>

    <footer>

        <p>&copy; 2021 CSS Flexbox</p>

        <p>Desenvolvido por: Seu nome aqui dev</p>

    </footer>

</body>

</html>

Resultado:

 Interface gráfica do usuário, Aplicativo, Site

Descrição gerada automaticamente

PARTE 2:

Teve este quebra:

Interface gráfica do usuário

Descrição gerada automaticamente

Ela foi ajustada com o ajuste do tamanho do font-size e definição das imagens:

