CSS Grid: Simplificando layouts

1. **Aula 1 – Definindo o Layout Base:**
   1. Para fazer um grid, nós primeiro precisamos determinar que aquela seção, ou até mesmo o próprio body, se você quiser que o grid se aplique a todo o seu site, é/se comporte como um grid.
      1. Para isso nós colocamos uma classe específica naquela seção e no CSS colocar um *display: grid;*, e fazer algumas outras configurações, determinando quais serão as áreas do cabeçalho, conteúdo e rodapé.
      2. Essas áreas são as classes colocadas em cada sessão/tag que você queira que sejam grids, como por exemplo as classe: .cabecalho, .conteudo, .rodape.
      3. Tendo isso em mente, como definimos o <body> como *display: grid;* é nele que iremos determinar essas áreas usando o *grid-tamplate-areas: “cabecalho” “conteudo” “rodape”;*.
      4. Em seguida, podemos colocar o *grid-tamplate-columns: auto;*, para setarem as colunas automaticamente. O mesmo vale para as linhas *grid-tamplate-rows: auto;*. Porém com um adendo: Cada área tem sua própria linha, dessa forma, precisamos colocar valores para cada um ficando: *grid-tamplate-rows: 50px auto auto;*, no caso do nosso exemplo.
   2. Para determinar que cada área que você colocou seja referente a que está na configuração da tag <body>, precisamos colocar um *grid-area: nomeDoGrid;*, em todas as tags que determinamos que serão áreas, colocando o nome referente ao que foi colocado na config inicial. Geralmente coloca-se o mesmo nome da classe da tag.
2. **Aula 2 – Criando o Layout da Seleção de destaques:**
   1. Para que uma imagem de background ocupe a quantidade de linhas e colunas desejadas, elas precisam ser especificadas no css, caso contrário, irá ser carregada apenas no primeiro bloco do grid.
      1. Para definir qual será essa quantidade, precisamos ir nas configurações de css referente a imagem que desejamos carregar e colocar o *grid-column-start: 1;* e o *grid-column-end: 4;* por exemplo. Dessa forma, a imagem irá carregar da coluna um à coluna 3.
      2. O mesmo é valido para as linhas: *grid-row-start: 1;* e *grid-row-end: 3;*. Desse modo, carregando a imagem nas 2 primeiras linhas do grid referente à sua sessão.
      3. Porém, precisamos notar que quando colocamos esses valores estamos dizendo para o grid que ele tem que ir até o começo da linha ou coluna determinada, ou seja, no caso das colunas acima, ele irá carregar a imagem na coluna 1 e 2, e não 1, 2 e 3, pois determinamos que o final é no começo da terceira coluna. Se tivéssemos colocado 4 no *end* estaríamos dizendo que é pra imagem ser carregada do 1 ao 3. O mesmo vale para as linhas.
   2. Para que a gente não fique repetindo códigos todas as vezes que vamos configurar um grid, nós podemos utilizar os shorthands e colocar apenas uma linha de código para várias configurações:
      1. No background, ao invés de colocar o todas as configurações de repetição, posição e tals, nós podemos colocar uma linha com *background: url(localDaImagem) posição / cover* (para cobrir toda grid) *repeteOuNão*.
      2. O mesmo vale para a definição de colunas e linhas que começam e acabam: *grid-column: início / fim;* e o *grid-row: início / fim;*. Tem que separar pela barra, caso contrário não funciona.
   3. Podemos selecionar um item específico de vários que possuem a mesma classe usando a extensão :*nth-child(número){}* logo após o nome da classe que desejamos usar. Porém com uma observação: O número colocado é referente ao elemento/divisão mãe desse filho, e não a quantidade de elementos com as mesmas classes, ou seja, se você possui vários elementos com a classe *.destaques\_\_secundario* que vem logo após uma classe *.destaques\_\_primario*, todas estão dentro da divisão com classe *.destaques* e você quer fazer a configuração específica do primeiro elemento da classe *.destaques\_\_secundario* o seu *:nth-child(número){}* terá o número 2, pois antes dele vem apenas 1 elemento da classe *.destaques\_\_primario*. Se tivessem 2 elementos primários antes do primeiro secundário, o número do child do primeiro secundário seria ***3*** ao invés de 2.
   4. Existe outra propriedade chamada *grid-gap: px/rm/rem;* onde nós podemos definir o gap, o espaçamento vazio que terá em volta de um grid. Quase como um border só que próprio dos grids. Quando colocamos isso nós basicamente definimos que haverá um espaço em branco/backgorund-color entre as imagens ou seja lá qual o conteúdo que tenha dentro do grid.
   5. **O que aprendemos nessa aula:** 
      1. Definir um template sem áreas.
      2. Definir a quantidade das colunas.
      3. Definir a quantidade das linhas.
      4. Dizer para os elementos onde devem começar e terminar.
      5. Definir espaçamento entre linhas e colunas.
3. **Aula 3 – Estilizando a sessão de populares:**
   1. O que aprendemos na aula:
      1. Que é possível haver um grid dentro de outro.
      2. Calcular largura.