**1 - Organizando o container com css grid**

**Objetivos da Aula**

* Dar os primeiros passos com css grid
* Entender como colocar grid no container

**Snippets**

Nós conseguir fazer com o grid, uma organização por colunas e linhas, iremos ver aqui nessa aula essa organização base

1. Faremos uma configuração básica com html e css, assim como fizemos com flexbox
2. <!DOCTYPE html>
3. <html lang="pt-br">
4. <head>
5. <meta charset="UTF-8" />
6. <meta http-equiv="X-UA-Compatible" content="IE=edge" />
7. <meta name="viewport" content="width=device-width, initial-scale=1.0" />
8. <link rel="stylesheet" href="./style.css" />
9. <title>Css Grid</title>
10. </head>
11. <body>
12. <div class="container">
13. <div class="item">1</div>
14. <div class="item">2</div>
15. <div class="item">3</div>
16. <div class="item">4</div>
17. <div class="item">5</div>
18. <div class="item">6</div>
19. </div>
20. </body>
21. </html>
22. \* {
23. padding: 0;
24. margin: 0;
25. }
26. .container {
27. min-height: 100vh;
28. background-color: black;
29. }
30. .item {
31. height: 200px;
32. background-color: #bbb;
33. border: 1px solid black;
34. }
35. Agora podemos fazer as modificações que queremos com o grid. Iremos fazer o container virar grid e criar agora as colunas

Cada frame (fr) que você da, cria uma coluna. Você pode até mesmo criar colunas vazias, colocando mais fr do que existe de itens dentro do container, e o espaço vai se adaptando

E caso queira deixar apenas 1 coluna com um width maior, basta você mudar de 1fr, para o número que quiser.

.container {

min-height: 100vh;

background-color: black;

display: grid; /\* LINHA NOVA \*/

grid-template-columns: 1fr 1fr 1fr; /\* LINHA NOVA \*/

}

1. O melhor meio de criar colunas é usando essa funcionalidade só que com um repeat, que aí você não vai precisar ficar repetindo “1fr” cada vez que quiser uma nova coluna
2. .container {
3. min-height: 100vh;
4. background-color: black;
5. display: grid;
6. grid-template-columns: repeat(3, 1fr); /\* LINHA NOVA \*/
7. }
8. Agora iremos fazer a adaptação de linhas, fazendo com que elas fique de uma forma mais bonita visualmente.

É a mesma coisa basicamente que o template columns, só que mudando de columns para rows.

grid-template-rows: repeat(3, 1fr);

1. E caso você queira, o gap aqui também funciona, você tem tanto o gap gera, que funciona para linhas e colunas quanto também o gap individual para cada uma delas
2. gap: 10px;
3. row-gap: 10px;
4. column-gap: 10px;

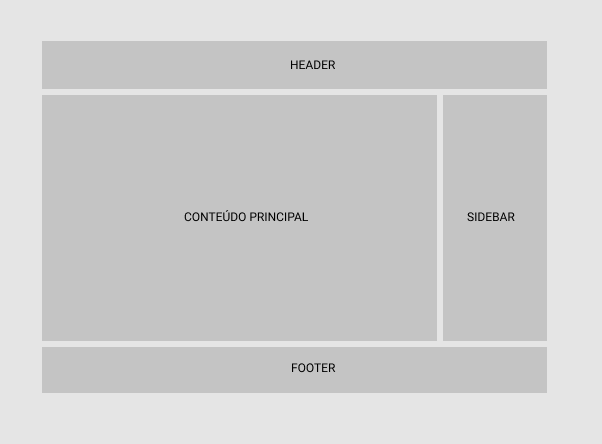
Com isso, vimos o básico de organização de grid do container, como você bem sabe, caso esteja confuso e fora de visão agora, quando formos fazer o layout na prática, ficará simples para você entender, pode ficar tranquilo!

**Objetivos da Aula**

* Mostrar funcionalidades que farão a divisão de linhas e colunas
* Avançar no uso do css grid

**Snippets**

Vamos ver agora alguns ajustes a serem feitos para conseguirmos ter algo semelhante ao que queremos depois, com o nosso layout:



1. Primeiro nós precisamos adicionar uma diferenciação com as colunas, colocando uma classe especial para elas
2. <div class="item item1">1</div>
3. <div class="item item2">2</div>
4. <div class="item item3">3</div>
5. <div class="item item4">4</div>
6. <div class="item item5">5</div>
7. <div class="item item6">6</div>
8. Com isso, vamos selecionar os itens 1 e 2 para fazer alguma brincadeira com eles. vamos fazer com que eles ocupem o lugar que queremos

Com isso, podemos facilmente falar que o item 1 é o nosso header, o item 2 é o nosso main-content e o 3 é o nosso sidebar. Só que iremos fazer essas manipulações melhores na próxima aula

.item1 {

grid-column: 1/4;

}

.item2 {

grid-column: 1/3;

}

1. Vamos agora ver a grid-row, que será parecida com a grid-column, só que, focada na linha, e não na coluna.
2. .item1 {
3. grid-column: 1/4;
4. grid-row: 1/3;
5. }

Com isso, você agora sabe como fazer manipulações “mais profundas” com as colunas e linhas, fazendo com que elas ocupem espaços que você definir

**3 - Criando um template básico com o css grid**

**Objetivos da Aula**

* Colocar em prática em um layout simples o que vimos nesse módulo até agora!

**Snippets**

Bom, vamos buscar criar um novo layout para fazermos o uso do grid, faremos algo parecido com esse layout, que é um layout comum de se ver por aí!

1. Vamos começar com esse HTML aqui:
2. <!DOCTYPE html>
3. <html lang="pt-br">
4. <head>
5. <meta charset="UTF-8" />
6. <meta http-equiv="X-UA-Compatible" content="IE=edge" />
7. <meta name="viewport" content="width=device-width, initial-scale=1.0" />
8. <link rel="stylesheet" href="./style.css" />
9. <title>Css Grid</title>
10. </head>
11. <body>
12. <div class="container">
13. <div class="item item1">1</div>
14. <div class="item item2">2</div>
15. <div class="item item3">3</div>
16. <div class="item item4">4</div>
17. </div>
18. </body>
19. </html>
20. E também utilizar o seguinte css:
21. \* {
22. padding: 0;
23. margin: 0;
24. }
25. .container {
26. box-sizing: border-box;
27. min-height: 100vh;
28. background-color: black;
29. display: grid;
30. grid-template-columns: repeat(3, 1fr);
31. grid-template-rows: repeat(3, 1fr);
32. gap: 10px;
33. }
34. .item {
35. height: auto;
36. background-color: #bbb;
37. border: 1px solid black;
38. }
39. .item1 {
40. }
41. .item2 {
42. }
43. .item3 {
44. }
45. .item4 {
46. }
47. Com isso, precisamos apenas fazer algumas alterações agora nas linhas em si, com isso, conseguimos fazer através do grid-row
48. .item1 {
49. grid-column: 1/4;
50. }
51. .item2 {
52. grid-column: 1/3;
53. grid-row: 2/12;
54. }
55. .item3 {
56. grid-row: 2/12;
57. }
58. .item4 {
59. grid-column: 1/4;
60. }
61. Vamos colocar uma coisa nova no nosso container, chamada de template areas, onde “desenharemos” o nosso grid.
62. grid-template-areas:
63. "item1 item1 item1"
64. "item2 item2 item3"
65. "item2 item2 item3"
66. "item4 item4 item4";
67. E vamos modificar algumas coisas dentro dos itens que temos
68. .item1 {
69. grid-area: item1;
70. }
71. .item2 {
72. grid-area: item2;
73. }
74. .item3 {
75. grid-area: item3;
76. }
77. .item4 {
78. grid-area: item4;
79. }

E com isso nós temos basicamente o mesmo resultado, com a diferença apenas dos tamanhos de linhas.

Conseguimos criar um layout extremamente semelhante ao que temos como exemplo e que podemos fazer várias outras modificações nele também!

Você vai fazer fazer uma pasta nova e fazer index.html e um style.css. Também teremos uma pasta de imagens, onde ficará a nossa lupa e a nossa logo, que você pode buscar aqui: <https://drive.google.com/drive/folders/1-NBrx3RyMVk1at40-CxbnRzLAxFWwdQT?usp=sharing>

1. Vamos começar com um container, que dentro dele terá o nosso header
2. <div class="container">
3. <header>
4. </header>
5. </div>
6. E iremos fazer a divisão dos conteúdos que estarão aqui dentro, ficará um pouco confuso de começo, mas teremos depois o nosso momento de organização
7. <a href="#">
8. <img src="./logo-home.gif" alt="" class="logo" />
9. </a>
10. <div class="search-div">
11. <input type="search" placeholder="Pesquisar" class="search" />
12. <div class="lupa">
13. <img src="./lupa.png" alt="" />
14. </div>
15. <div class="item mic"></div>
16. </div>
17. <ul class="perfil-itens">
18. <li class="item"></li>
19. <li class="item"></li>
20. <li class="item"></li>
21. </ul>
22. <div class="item burguer"></div>
23. Faremos também a criação de mais dois itens, que vão virar mais duas colunas, onde terão o conteúdo principal e o side bar
24. <div class="main-content">
26. </div>
27. <div class="side-bar">
29. </div>
30. Iremos também montar os itens dentro deles, começando com o main-content
31. <div class="video"></div>
32. <p class="title">Vídeo de teste</p>
33. <div class="interacao"></div>
34. <div class="line"></div>
35. <div class="info-canal"></div>
36. Agora faremos também com o side-bar
37. <div class="side-video">1</div>
38. <div class="side-video">2</div>
39. <div class="side-video">3</div>
40. <div class="side-video">4</div>
41. <div class="side-video">5</div>
42. <div class="side-video">6</div>
43. Com isso, nós temos o nosso html pronto, está todo bagunçado, mas faremos os ajustes necessários para ficar algo organizado, começando com o nosso container
44. \* {
45. padding: 0;
46. margin: 0;
47. }
48. .container {
49. box-sizing: border-box;
50. min-height: 100vh;
51. background-color: #181818;
52. display: grid;
53. grid-template-columns: 1fr 2fr 400px;
54. grid-auto-rows: minmax(65px, auto);
55. }
56. Agora iremos para o nosso header e para os seus elementos
57. header {
58. grid-column: 1/4;
59. background-color: #202020;
60. display: flex;
61. justify-content: space-between;
62. align-items: center;
63. padding: 10px 15px;
64. }
65. header a {
66. text-decoration: none;
67. }
68. .logo {
69. width: 140px;
70. }
71. .search-div {
72. display: flex;
73. align-items: center;
74. }
75. .search {
76. width: 350px;
77. height: 45px;
78. background-color: #121212;
79. border: none;
80. outline: none;
81. color: white;
82. padding: 0px 10px;
83. }
84. .search::placeholder {
85. color: #747474;
86. font-size: 14px;
87. }
88. .lupa {
89. width: 60px;
90. height: 45px;
91. background-color: #313131;
92. cursor: pointer;
93. display: flex;
94. align-items: center;
95. justify-content: center;
96. margin-right: 10px;
97. }
98. .lupa img {
99. width: 25px;
100. }
101. .item {
102. width: 32px;
103. height: 32px;
104. border-radius: 50%;
105. background-color: #bbb;
106. cursor: pointer;
107. }
108. .perfil-itens {
109. list-style: none;
110. display: flex;
111. gap: 10px;
112. }
113. .burguer {
114. display: none;
115. }
116. Agora faremos a manipulação do main-content
117. .main-content {
118. grid-column: 1/3;
119. grid-row: 2/9;
120. padding: 20px 0px;
121. }
122. .video {
123. width: 70%;
124. height: 400px;
125. background-color: #bbb;
126. margin: 0 auto;
127. }
128. .title {
129. color: white;
130. font-family: sans-serif;
131. font-size: 30px;
132. margin-left: 15%;
133. padding: 10px 0px;
134. }
135. .interacao {
136. width: 70%;
137. height: 45px;
138. background-color: #bbb;
139. margin: 0 auto;
140. }
141. .linha {
142. width: 70%;
143. height: 1px;
144. background-color: #747474;
145. margin: 10px auto;
146. }
147. .info-canal {
148. width: 70%;
149. height: 100px;
150. background-color: #bbb;
151. margin: 0 auto;
152. }
153. Agora iremos fazer as nossas manipulações do side-bar
154. .side-bar {
155. grid-row: 2/9;
156. }
157. .side-video {
158. width: 90%;
159. height: 150px;
160. margin: 20px auto;
161. background-color: #bbb;
162. }
163. Agora faremos a nossa responsividade, pois como fizemos diversas manipulações com o grid, precisamos manipular para haver a alteração das colunas
164. @media (max-width: 1130px) {
165. .video {
166. width: 90%;
167. }
168. .title {
169. margin-left: 5%;
170. }
171. .interacao {
172. width: 90%;
173. }
174. .linha {
175. width: 90%;
176. }
177. .info-canal {
178. width: 90%;
179. }
180. }
181. @media (max-width: 950px) {
182. .container {
183. grid-template-columns: 1fr 2fr 32%;
184. }
185. }
186. @media (max-width: 768px) {
187. .main-content {
188. grid-column: 1/4;
189. grid-row: 2 / auto;
190. }
191. .side-bar {
192. grid-column: 1/4;
193. grid-row: 3 / auto;
194. }
195. .perfil-itens {
196. display: none;
197. }
198. .burguer {
199. display: block;
200. }
201. .search-div {
202. width: 60%;
203. }
204. .search {
205. width: 69%;
206. }
207. .logo {
208. width: 100px;
209. }
210. }
211. @media (max-width: 416px) {
212. .mic {
213. display: none;
214. }
215. .video {
216. width: 95%;
217. height: 200px;
218. }
219. }

Com isso, fizemos a nossa página inspirada no youtube, utilizamos o flex dentro de componentes e o nosso layout “geral” foi feito com grid, mas poderia ter sido utilizado apenas nas coluans de main-content e side-bar, mas é questão de escolha e também de necessidade.