02

# Terminando mobile

### Transcrição

Na última aula discutimos sobre `flex-container`, `flex-itens` e também sobre os eixos. Vimos, ainda, a importância do `flex-direction` para layout Mobile uma vez que os itens ficam dispostos um embaixo dos outros, portanto, este recurso será comumente utilizado!

Por enquanto a distribuição dos elementos está da seguinte maneira:

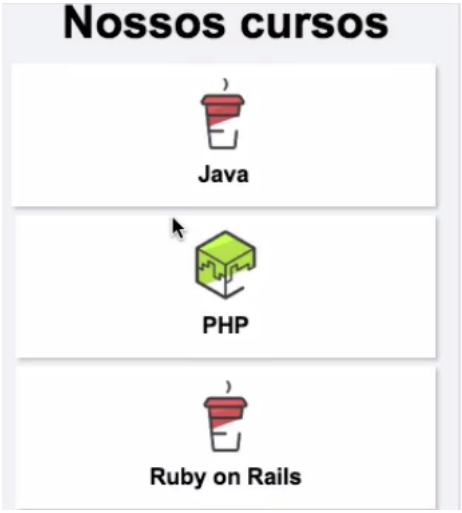

No layout ideal os cursos ocupam uma largura inteira:


Para arrumar a distribuição dos objetos é preciso alterar a largura e a direção dos itens. Para saber qual classe devemos manusear, abrimos o `Inspect` e verificamos que é a `.conteudoPrincipal-cursos-link`.

Para inserir a largura máxima colocamos o `width:100%`:

```
.conteudoPrincipal-cursos-link {
    width:100%;
}
```

Ao fazer isso podemos verificar que os cursos quebram para a próxima linha automaticamente:

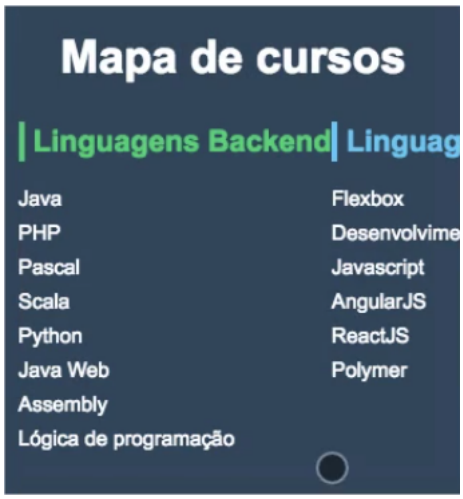



Apenas para deixar o código o mais correto possível ainda acrescentamos o `.conteudoPrincipal-cursos` junto do `flex-direction: column`. E com o objetivo de organizar a página inserimos primeiro o `.conteudoPrinciapl-cursos-link` e depois o `.conteudoPrincipal-cursos`. Assim o arquivo `flexbox.css` fica da seguinte maneira:

```
.conteudoPrincipal-cursos {
    flex-direction: column;
}

.conteudoPrincipal-cursos-link {
    width:100%;
}
```

Vamos pular a parte do vídeo e ir direto para o **Mapa de Cursos**. Nós temos o seguinte:




Dando um `Inspect` podemos verificar que existe uma `div` com uma altura de `350 px` fixa que uma vez desativada faz com que os elementos sejam distribuídos um embaixo dos outros livremente. Portanto, no Mobile é preciso arrancar a altura!

Acessamos o arquivo `flexbox.css` e adicionamos o `.rodapePrincipal-navMap-list` e colocamos o `height: auto`:

```
.rodapePrincipal-navMap-list {
    height: auto;
}
```

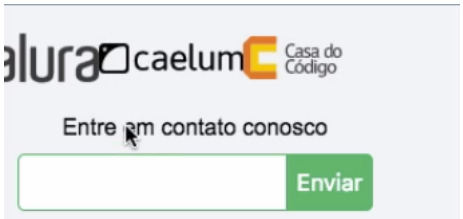
Assim, temos o seguinte:


Agora, vamos lidar com a próxima seção, o menu inferior:


Observando no `Inspect` o código referente a este trecho vemos que temos uma `div container` e nela temos o `display-flex` e o `direction` padrão desse elemento é `row`, ou seja, um do lado do outro. Portanto, no geral teremos que mudar tudo o que é `row` para `column`. Não conseguiremos alterar tudo utilizando apenas o `container`, portanto, pegaremos junto o `rodapePrincipal-patrocinadores`. Assim, acrescentamos junto disso o `flex-direction: column`. Teremos:

```
.rodapePrincipal-patrocinadores .container {
    flex-direction: column;
}
```

Fazendo isso temos o seguinte:




Falta alinhar os elementos no centro, para fazer isso vamos pensar um pouco. Estamos lidando com o `flex-container` e nele existe o eixo principal e o eixo cruzado. Quando introduzimos o `flex direction` os eixos trocaram de lugar o

principal passou a ser o cruzado e vice e versa. Então para alinhar estes elementos vamos utilizar o `align-items: center` :

```
.rodapePrincipal-patrocinadores .container {
    flex-direction: column;
    align-items: center;
}
```

Assim, fazendo isso temos os itens de cima alinhados enquanto os de baixo estão descentralizados. O que acontece é que o formulário está minúsculo, com uma largura definida de 25%, portanto, é preciso retirar esse valor. Portanto, acrescentamos no `flexbox.css` o `.rodapePrincipal-contatoForm` e adicionamos `width: 100%` para que ocupe a largura inteira:

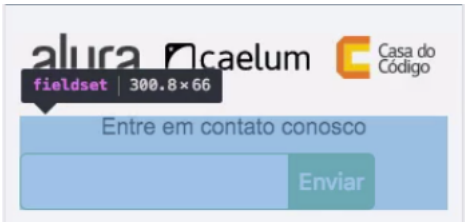
```
.rodapePrincipal-contatoForm {
    width: 100%;
}
```

O mesmo vale para a `ul` que está com o `width` definido para ocupar o espaço de `70` e possui também uma margem. Portanto, vamos redefinir as medidas e retirar a margem. Assim, no `.rodapePrincipal-patrocinadores-list` vamos adicionar `margin: 0` e `width: 100%` :

```
.rodapePrincipal-patrocinadores-list {
    margin: 0;
    width: 100%;

}
```

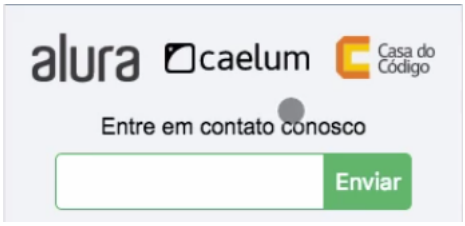
O reload mostra que tivemos melhorias significativas!




Ainda que o formulário esteja deslocado para a esquerda. Para arrumar isso vamos observar antes o código por meio do `Inspect` e descobrimos que ele a `div` possui um `display-flex` e para justificar esse conteúdo no centro é preciso colocar o `justify-content` que possui a propriedade `center` . Assim, escrevemos `.rodapePrincipal-contatoForm-fieldset` e o `justify-content:center` :

```
.rodapePrincipal-contatoForm-fieldset {
    justify-content:center
}
```

Assim, temos o seguinte:




Muito fiel ao que buscamos!

Não existem muitos segredos para modificar para **Mobile**. O grosso do trabalho é mudar de `flex-direction: row` para `flex-direction:column` para que os itens tenham uma distribuição vertical e fazendo tudo isso tendo em mente a mudança dos eixos.

Falta um detalhe! Ao aumentar a tela, até quando ela permanece com a cara do layout Mobile? Pois, mesmo colocando o tamanho equivalente a tela do desktop, a aparência de um celular segue:


Depois de `768 px` é interessante que a aparência deixe de ser a do Tablet e volte a ser a de *Desktop*. Portanto, é preciso inserir um `media query` logo que o código de Mobile inicia. bem abaixo do comentário `/*parte mobile*/` . Assim, no `flexbox.css` temos:

```
/*parte mobile*/
@media(max-width: 768px)
```

Com isso, até `768 px` ele renderiza **Mobile** e depois volta ao layout para **Desktop**!