Alt + shift + seta para baixo => clona um elemento nas próximas linhas abaixo (só falta descobrir como usar esse atalho, não funcionou comigo)

Uma boa pratica de formatação é sempre iniciar definindo margin e padding padrão para sua página. Assim, independente da tela ou do navegador a formatação ficará a mesma que a gente criar (os devs chamam isso de *reset*):

```css
*{
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}
```

o **box-sizing: border-box** define que o limite de do meu item será a borda da caixa, ou seja, ele não vai extrapolar o tamanho de algo que eu criar, se os itens dentro dele não couberem, eles serão espremidos (reduzidos) dentro da caixa para não saírem dela!

Vamos criar variáveis dentro do CSS, isso é uma boa prática para quando repetimos algo no nosso estilo. Por ex, sabemos as cores padrões que serão usadas no site que são vermelho e preto, e caso eu queira trocar essas cores, trocarei somente na minha variável, e não em cada elemento que eu apliquei a cor!

Então para isso vamos declarar as variáveis antes do *reset* (antes do *):

```css
:root{
  --vermelho: #e50914;
  --preto: #141414;
}
```

depois chamo ela assim:

```css
body{
  background: var(--preto);
}
```

Felipe deu uma dica sobre deixar por último os detalhes de formatação mais específicos, tipo escolha da fonte por exemplo. O mais comum é montar a estrutura toda do site primeiro e depois dedicar aos detalhes que queremos alterar, podendo visualizar as alterações com ele praticamente pronto, dando assim uma visualização melhor do resultado.

Vamos criar a parte principal do site, apresentação a série House of Cards em destaque.
Para isso Felipe usou a tag **<main>**, para devs isso é semântico mas não muda nada na nossa vida, pois ela é a mesma coisa que uma . Porém, para ferramentas de SEO a identificação do conteúdo fica mais fácil e elas já entendem que o que está dentro de <main> é o conteúdo principal da página.

Felipe usou como fonte das imagens e sinopses o site themoviedb.org.

Para deixar as imagens meio escuras (imitando netflix) vamos usar gradient com cor escura, assim ele cria uma camada fina e escura sobre a imagem criando contraste melhor com o texto sobre ela:

```css
/* filme principal */
.filme-principal{
  font-size: 16px;
  background: linear-gradient(rgba(0,0,0,.50),rgba(0,0,0,.50)100%), url('/img/capa-house.jpg');

  height: 400px;
  background-size: cover;
}
```

É preciso definir duas cores para o gradient funcionar!

Outra dica é usar **role=”button”** por ex, pra todo lugar onde a página for renderizada não ocorrer confusão em relação à função do elemento, que neste caso é um botão. Ou seja, abrindo esta página em celular, tablet ou PC, o meu botão criado sempre será renderizado de forma correta.

Estilização dos botoes:

```css
.botao{
  background-color: rgba(0,0,0,.50);
  border: none;
  color: #fff;

  padding: 15px 30px;
  margin-right: 15px;
  font-size: 12px;

  cursor: pointer;
  transition: .3s ease all;
}

.botao:hover{
  background-color: #fff;
  color: black;
}
```

rgba é uma sigla para *red green black alfa*, quando defino **0,0,0,** significa que dei valor zero para as cores, ou seja, defini uma cor preta. O *alfa* é em porcentagem, na verdade colocamos o valor decimal, ex: 50%, colocaremos **.50**

vamos tirar a borda padrão que foi criada pro botão usando **border: none;**

vamos dar um **padding**, para o botão ter um tamanho: **padding: 15px 30px;**

margin-right: 15px para afastar um do outro

O **transition** é para fazer um efeito mais leve ao hover fazendo o efeito ser aplicado com um “delay” de 0.3s

Resultado:

Agora vamos adicionar os ícones fa (font awesome) aos nossos botões. Para isso é só ir no site fontawesome.com clicar em “start for free”, colocar seu e-mail para receber o link e colocá-lo no seu código (ele envia o “kit code” pro seu e-mail):

```html
        <div class="botoes">
          <button role="button" class="botao">
            <i class="fas fa-play"></i>
            ASSISTIR AGORA
          </button>
          <button role="button" class="botao">
            <i class="fas fa-info-circle"></i>
            MAIS INFORMAÇÕES
          </button>
        </div>

      </div>
    </div>
  </main>

  <script src="https://kit.fontawesome.com/2c36e9b7b1.js"></script>

</body>
</html>
```

Copiei o link do Felipe:

https://kit.fontawesome.com/2c36e9b7b1.js

Dessa forma estamos referenciando um link externo do arquivo javascript, não está na nossa pasta.

Agora vou no site fontawesome e escolho meu icone, copio o html que ele já fornece e colo no meu cód no lugar onde vou usar:

```html
<div class="botoes">
  <button role="button" class="botao">
    <i class="fas fa-play"></i>
    ASSISTIR AGORA
  </button>
```

Wrapper é a ideia de agrupar tudo que faz parte de um conjunto, ou uma seção, de elementos que compõem algo. Dessa forma, quando eu quiser dar uma margem por exemplo, não preciso fazer em cada elemento, mas sim no meu wrapper, uma vez só.

No nosso projeto Felipe usou a classe **container** para aplicar essa ideia de wrapper ao *titulo*, *descricao* e aos *botoes* da parte de dentro da seção *filme-principal* do nosso site.

JQUERY

Usaremos jquery para fazer nosso carousel com as imagens dos filmes e series.

Mesmo sabendo que jquery está entrando em desuso, é bom conhecer e saber como trabalhar com essa ferramenta.

Felipe entrou no seguinte link: https://owlcarousel2.github.io/OwlCarousel2/

Baixou o arquivo zip de todo o conteúdo do carousel clicando em download.

Precisamos dos arquivos javascript do carousel. Na pasta docs/assets/vendors, copie o arquivo *jquery.min*, coloque na pasta owl (criada dentro de js no seu projeto). Pegue também o arquivo *owl.carousel.min* na pasta docs/assets/owlcarousel.

Agora precisamos dos arquivos css, vamos salvá-los na pasta style dentro do nosso projeto. Dentro da pasta dist/assets, copie os arquivos *owl.carousel.min* e *owl.theme.default.min*.

Proximo passo é referenciar esses arquivos no **head**:

```
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="style/main.css">

  <!--owl css-->
  <link rel="stylesheet" href="style/owl/owl.carousel.min.css">
  <link rel="stylesheet" href="style/owl/owl.theme.default.min.css">

  <title>Netflix Clone</title>
</head>
```

Vamos fazer a mesma coisa para os arquivos **javascript**. Poderia ser também no head, porém é **muito recomendado** colocar no final (inclusive o script para font awesome já está lá embaixo) pois a página sempre carrega em ordem sequencial e se tentar carregar um javascript antes da pagina ter carregado totalmente, pode acontecer um erro.

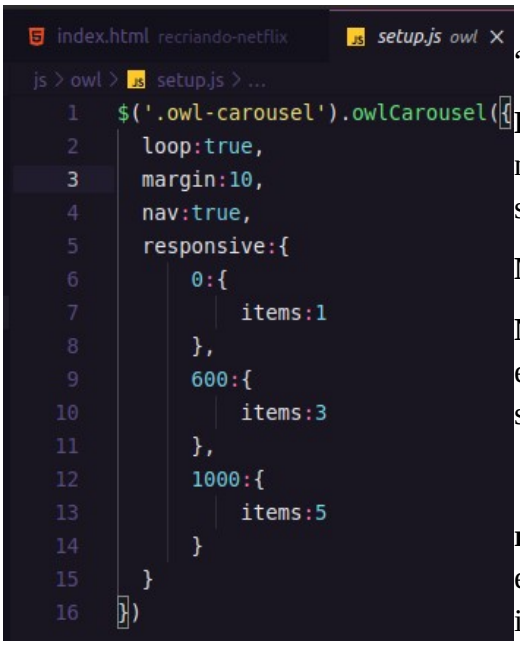
```
  <script src="https://kit.fontawesome.com/2c36e9b7b1.js"></script>
  <script src="js/owl/jquery.min.js"></script>
  <script src="js/owl/owl.carousel.min.js"></script>

</body>
</html>
```

No site do **carousel** no menu, vá em **Demos** e explore as opções oferecidas.

Felipe foi em **basic** mesmo. Já vem todo o cód pronto pra te ajudar a construir o carousel.

Referenciar pelo jquery

Toda vez que ver o $ pode saber que é o JQUERY sendo usado. Vc o chama com $ e em seguida dentro de parênteses coloca a sua configuração de elementos em tela.
Vamos copiar da parte **setup** do site *owlcarousel2* para um documento **setup.js** que vamos criar dentro da pasta js/owl :



Este arquivo é responsável por fazer o carousel funcionar. Se não "chamarmos" ele no nosso código nada acontecerá!

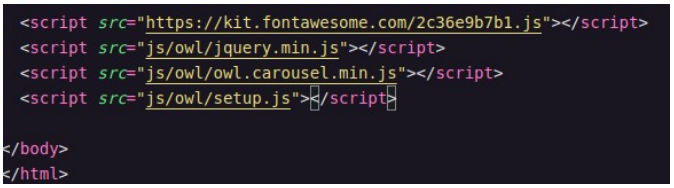
Loop: define se o scroll das imagens é infinito, ou seja, igual da netflix, vou passando pelos filmes e acabo retornando no primeiro sem ter que voltar pra trás.

Margin: altera o espaçamento entre uma imagem e outra.

Nav: são as setinhas de navegação das imagens que ficam embaixo delas. Se tirar essas setinhas (colocando **false**) apareceria só as bolinhas:

responsive: aqui eu digo quantos itens no meu carousel eu quero exibir de acordo com a tela. Ex: acima de 1000px posso exibir 5 itens/ entre 600-1000px exibo 3/ entre 0-600 exibo 1 item apenas.

Então vamos chamar ele junto com os outros scripts que estamos usando no nosso codigo (js/owl/setup.js):

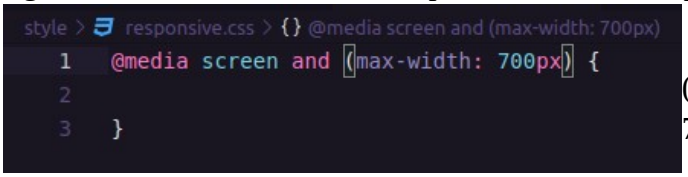


Tratando a responsividade do site – media queries

Felipe costuma criar um arquivo separado para fazer a formatação css responsiva. Ele ficará dentro da pasta **style** junto com nosso arquivo **main.css** (nosso css principal).

Vamos referenciar ele dentro de **head** logo acima do css do carousel.

Agora vamos alterar nosso menu para telas menores que 700px:



aqui estou dizendo para aplicar na tela (screen) e também quando for no maximo 700px

Vamos mudar a **flex-direction** do container do header para **column**, quando for menor que 700px (agora ele fica embaixo da logo).

E também colocar um **margin-top: 5px** para os botões não ficarem muito colados um no outro.

Vamos também configurar a descrição do filme para não se espalhar pela tela quando ela for muito grande. Acima de 1000px quero que ela tenha um tamanho de 50%.

```
@media screen and (min-width:1000px){
  .descricao{
    width: 50%;
  }
}
```

Desafios:

Criar as paginas pros links, ex: clicar no filme e ir pro trailer do filme no youtube.