

Trabalho Prático 1

Foi utilizado HTML, CSS, Javascript (biblioteca Jquery e JSON para estruturar os dados para utilizar a API do YouTube). Todo trecho de código está comentado. O sistema roda corretamente nos navegadores Chrome e Firefox. O sistema funciona da seguinte forma: ao preencher o formulário (não necessariamente todos os campos) e clicar no botão ‘Indicação’ é feita uma requisição Ajax à página <https://leandrojsa.github.io/movies.html> a qual está hospedada toda a base de dados.

A imagem mostra um formulário web com o título "Indique-me um filme". Abaixo do título, há uma pergunta: "Quer assistir um filme e não sabe qual?". O formulário contém quatro campos de entrada: "Escolher categoria:" (um menu suspenso), "Ano" (um campo de texto), "Atores" (um campo de texto) e "Outras informações" (um campo de texto). Na base do formulário, há dois botões: "INDICAÇÃO" em verde e "LIMPAR CAMPOS" em vermelho.

Figura 1 – Formulário de parâmetros de busca ao filme

O usuário não obrigatoriamente terá que preencher todos os campos. Caso todos os campos sejam deixados em branco, o filme indicado será totalmente aleatório. Em caso da combinação dos valores dos elementos de entrada ser algo que não exista na base de dados, será exibida a mensagem: “Não há filme que atenda aos parâmetros especificados”. O botão de ‘Limpar campos’ foi criado melhorar a usabilidade do usuário em termos de praticidade, “zera” valores de todos os campos que tenham sido colocados em buscas anteriores.

Categoria é buscado com um `<select>` `<option>` onde possui um *dropdown* com todas as categorias existentes na base de dados. Para o ano só é permitido caracteres de números pelo atributo ‘number’ em seu respectivo input e a busca é feita apenas no intervalo de 1909 até 2012, o qual os filmes da base de dados estão contidos. O campo de ‘Outras informações’ pode ser buscado título ou parte da sinopse que encontrará como sugestão. No campo ‘Atores’ pode ser digitado com espaços antes ou depois da vírgula e com espaços antes ou depois da palavra que não haverá problemas, pois foi tratado na comparação com o método `trim()` como podemos ver no procedimento abaixo:

```
117 // busca de sugestão por atores
118 if (atores[0] != "") {
119     console.log(atores);
120     for (var i = 0; i < filmes.length; ++i) {
121         var certificação=[],cont=0;
122         $(atores).each(function (a,ator){
123             certificação[a]=false;
124             for (var j = 0; j < todos_atores[i].children.length; ++j) {
125
126                 if (todos_atores[i].children[j].textContent.toLowerCase().indexOf(ator.toLowerCase().trim()) > -1) {
127                     certificação[a]=true;
128                 }
129             }
130         });if(certificação.length===atores.length && !certificação.includes(false)){
131             sugestoAtores.push(i);
132         }
133     }
134 }
```

Figura 2 – método em que são tratados os atores buscados – code.js

Neste mesmo procedimento podemos observar o método `toLowerCase()`, que foi utilizado em todos os outros parâmetros também, que permite que a busca encontre resultados sendo *Case Insensitive*, ou seja, não há distinção entre “A Batalha de Argel” ou “a batalha de argel” na busca. O mesmo ocorre para nomes dos atores, títulos dos filmes e sinopses.

Um outro ponto importante dessa busca é que são encontradas todas as substrings, ou seja, não é necessário digitar o nome completo do filme para encontrá-lo, para o mesmo exemplo, se fosse buscado a string “batalha de Ar”, poderia ser encontrado o filme “A batalha de Argel”. Idem para os atores, títulos e sinopses.

Há uma div com id ‘situacao’ que exibe a situação da requisição – através do *onreadystatechange*, variando em “Página não encontrada”, “Estabelecendo conexão com o servidor...”, “Requisição recebida”, “Processando requisição...”, “Requisição finalizada e resposta pronta.” para que o usuário possa acompanhar o processo da requisição condizente com o que está ocorrendo.

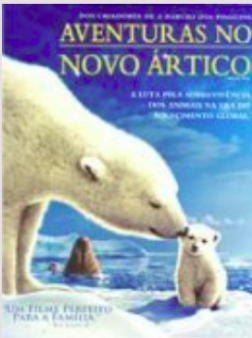
```

41 <a name="anchor"></a>
42 <div id="situacao"></div>
43
44 <div id="resultadoBusca"></div>
45
46 <div id="conteudovideo" align="center" >
47   <iframe id="video-container" hidden></iframe>
48 </div>
49
50 <div id="novostrailers"></div>

```

Figura 3 – espaço onde o conteúdo é gerado ao buscar por um filme – index.html

No momento que a resposta da sugestão está pronta a tela “desce” para melhor visualização, onde foi utilizada a tag `<a>` com o atributo nome de valor ‘anchor’, que pode ser visto na figura acima. O `<iframe>` é onde será exibido o filme encontrado com a API do YouTube e a `<div>` de id “novostrailers” é onde vai ser exibido um link para a página do YouTube com outros resultados para o trailer, caso queira ver outros trailers além do sugerido.

Aventuras no Paraíso 2


Ano: 1986

Categoria: Comédia

Sinopse:
Comédia erótica sobre uma equipe e suas confusões ao tentar produzir um filme na Grécia.

Atores:
James Karen, Alba Francesca, Louise Baker, Alexandros Mylonas, Robert Rhine, Brad Zutaut, Sam Temeles, Curt Wilmot, Nana, Brenda Bakke, Ulrika Hellstrom, Lifteris Andrikos, Thanassis Christopoulos, Sorrells Pickard, Roberta Collins, Julie Rohde-Brown, Alexandra Pavlidou, Elizabeth Pericolo, Vicky Kougianos, Fabiana Udenio, Giorgos Tzifos, Panos Logothetis, Vassili Karis, Charles P. Bernuth, Yorgos Kotanidis, Yula Gavala, Nadine Ames, Mark Kubr

Figura 4 – conteúdo que é na `<div>` resultadoBusca

Ao ser efetuada a busca (clicando no botão ‘Indicação’) será exibido um filme de acordo com a combinação dos parâmetros, ou seja, se buscar ator Robert De Niro com o ano de 1993, será exibido um filme que contenha ambos os conteúdos juntos, se não houver o usuário receberá a

mensagem de que não há resposta para estes parâmetros. Para filmes que não possuem imagem definimos uma como padrão. Todas as funcionalidades são realizadas sem recarregar a página totalmente, utilizando para isso o Ajax, o que além de dar interatividade permite que seja feita de modo mais rápido, já que não é necessário carregar toda a página novamente, mas apenas o novo conteúdo.

Exibimos um vídeos referente ao primeiro resultado da busca (às vezes varia dependendo do resultado da API) utilizando a string formada da seguinte forma: *título_do_filme* + *trailer* + *ano*. Vale ressaltar que alguns vídeos carregado do trailer aparece como ‘vídeo indisponível’, que é uma falha da API, mas isso ocorre apenas em algumas exceções. Na maioria das vezes o vídeo do trailer é carregado corretamente sem problemas.

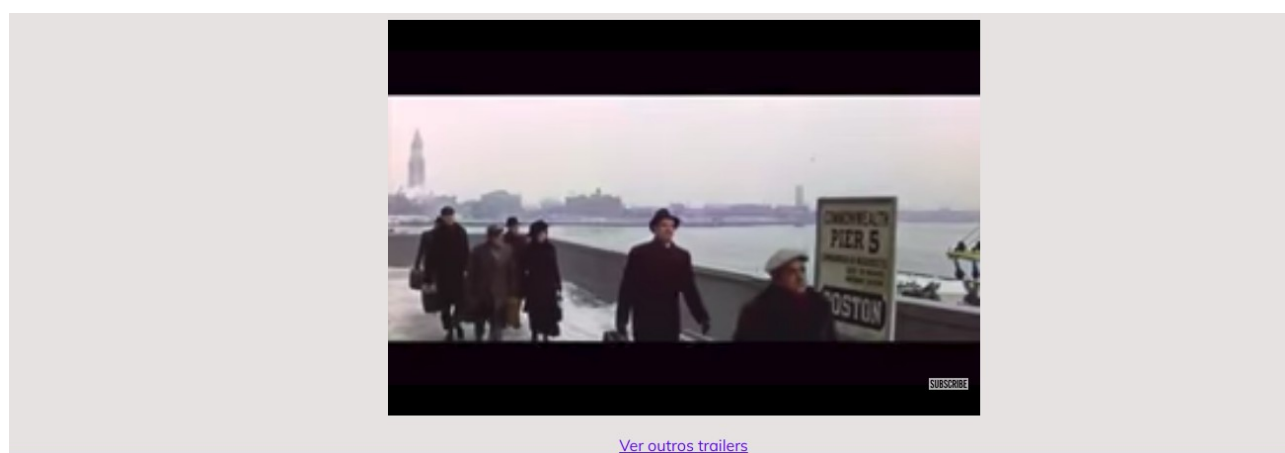


Figura 5 – Vídeo do trailer que é carregado logo abaixo do conteúdo do filme

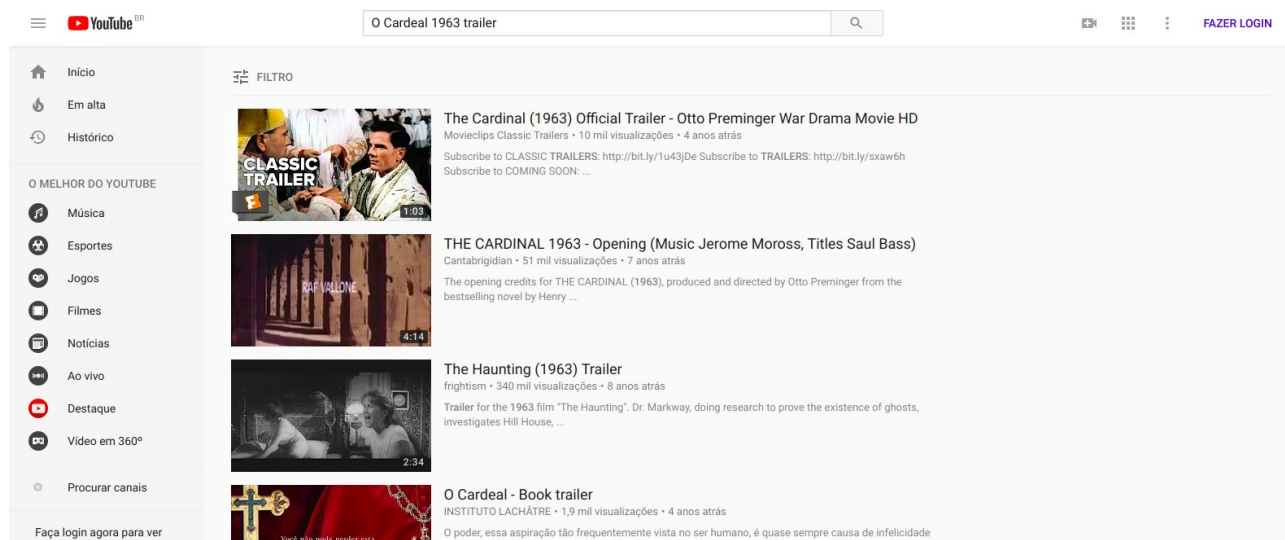


Figura 6 – redirecionamento de “Ver outros trailers” localizado embaixo do trailer como mostrado na figura anterior