

## Preview de imagem com Javascript e JQuery

### Transcrição

A ideia agora é deixarmos nossa aplicação um pouco mais interativa. Por exemplo, no momento em que o usuário clicar em "Mudar capa" e selecionar uma imagem, queremos que ela seja mostrada na tela como *preview*. Não é possível fazer isso com CSS ou de forma simples usando somente o Flask, portanto precisaremos do Javascript.

Felizmente, é bem simples adicionar um código ou uma biblioteca Javascript no nosso projeto Flask.

Em `novo.html`, temos um `<img>` que exibe a foto da capa. Quando o usuário clicar no link "Mudar capa", queremos remover a capa padrão e colocar a imagem nova selecionada pelo usuário para que ela seja exibida como *preview*.

Para isso, vamos inserir um código Javascript que irá trocar essa imagem e usar a biblioteca JQuery para facilitar um pouco a seleção dos elementos que temos no HTML.

Nesse curso, recomendamos a utilização [dessa \(https://caelum-online-public.s3.amazonaws.com/739-python-flask2/04/jquery.zip\)](https://caelum-online-public.s3.amazonaws.com/739-python-flask2/04/jquery.zip) versão do JQuery, que é estável e não trará nenhuma surpresa. O arquivo `jquery.js` nesse `.zip` desse ser colocado na nossa pasta de estáticos ("static").

Em seguida, criaremos um novo arquivo `app.js`, no qual colocaremos o seguinte código:

```
$( 'form input[type="file"]' ).change( event => {  
    let arquivos = event.target.files;  
    if (arquivos.length === 0) {  
        console.log('sem imagem pra mostrar')  
    } else {  
        if(arquivos[0].type == 'image/jpeg') {  
            $('img').remove();  
            let imagem = $('<img class="img-responsive">');  
            imagem.attr('src', window.URL.createObjectURL(arquivos[0]));  
            $('figure').prepend(imagem);  
        } else {  
            alert('Formato não suportado')  
        }  
    }  
});
```

Esse código seleciona e verifica, no *input* de arquivo do nosso formulário, o evento de modificação (*change*). Quando isso acontece, ele verifica se o tamanho do arquivo selecionado é igual a 0. Nesse caso, será exibida a mensagem "sem imagem pra mostrar" no *log* do console do Javascript.

Caso contrário (se o tamanho do arquivo selecionado for maior que 0), ele verificará qual o tipo de arquivo. Se for uma imagem, ele irá remover nossa capa padrão e, em seguida, criará uma variável `imagem` recebendo um objeto criado pelo Javascript, que é justamente uma nova imagem.

A essa nova `imagem`, é adicionado um atributo *source* (`src`) que criará um objeto baseado na URL do arquivo que está sendo passado. Portanto, o arquivo recebido no `input file` fica em memória.

Em seguida, com `$('figure').prepend(imagem)`, passamos essa imagem criada dinamicamente para o template.

Se o usuário seleciona um arquivo que não é uma imagem, ele recebe a mensagem "Formato não suportado".

Agora só falta adicionarmos a biblioteca JQuery no nosso projeto. Diferente do CSS, o Javascript não é adicionado por meio de links, mas sim de *scripts*. Como você já deve saber, a melhor prática é colocar os *scripts* no final da página - já que, dessa forma, o Javascript é carregado por último e não impede o carregamento do resto da página.

Nosso `<script>` deve ter um `src` (*source*), cujo caminho acessaremos por meio do `url_for()`, passando o caminho dos nossos estáticos (`static`) e o nome dos nossos arquivos (`jquery.js` e `app.js`, em *scripts* separados).

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Jogoteca</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='bootstrap.css') }}">
    <link rel="stylesheet" href="{{ url_for('static', filename='app.css') }}">
  </head>
  <body>
    <div class="container">
      {% with messages = get_flashed_messages() %}
      {% if messages %}
        <ul id="messages" class="list-unstyled">
          {% for message in messages %}
            <li class="alert alert-success">{{ message }}</li>
          {% endfor %}
        </ul>
      {% endif %}
      {% endwith %}
      <div class="page-header">
        <h1>{{ titulo }}</h1>
      </div>

      {% block conteudo %}{% endblock %}
    </div>
    <script type="text/javascript" src="{{ url_for('static', filename='jquery.js') }}"></script>
    <script type="text/javascript" src="{{ url_for('static', filename='app.js') }}"></script>
  </body>
</html>
```

Feitas essas modificações, o comportamento dinâmico da nossa página passará a funcionar:

## Novo jogo



Mudar capa

Nome

Categoria

Console

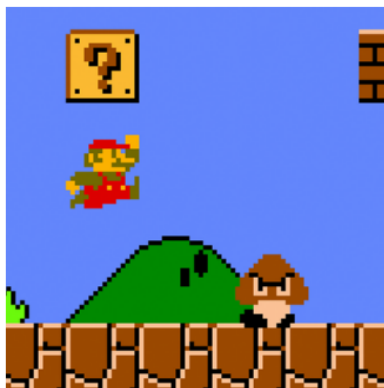
Salvar

Voltar

Também precisamos implementar esse comportamento na nossa tela de edição, o que é bastante simples: basta adicionarmos as classes de `figure`, `img` e `label` que colocamos em `novo.html`:

```
{% extends "template.html" %}
{% block conteudo %}
    <form action="{{ url_for('atualizar') }}" method="post" enctype="multipart/form-data">
        <figure class="thumb col-md-4">
            
            <figcaption>
                <label class="fileContainer">
                    Mudar capa
                    <input type="file" name="arquivo" accept=".jpg">
                </label>
            </figcaption>
        </figure>
    </form>
```

## Editando jogo



Mudar capa

Nome

Mario

Categoria

Plataforma

Console

NES

Salvar

Voltar

No próximo vídeo implementaremos uma forma de remover a imagem do banco. Até lá!