Aula 05 – jQuery

Por conta das dificuldades enfrentadas pelos programadores Javascript para páginas Web, foi criada uma biblioteca que traz diversas funcionalidades voltadas à solução contornando o uso do Javascript puro.

A principal vantagem na adoção de uma biblioteca em Javascript é permitir uma maior compatibilidade de um mesmo código com diversos navegadores. Uma maneira de se atingir esse objetivo é criando funções que verificam quaisquer características necessárias e permitam que o programador escreve um código único para todos os navegadores.

Além dessa vantagem o jQuery, uma das bibliotecas mais difundidas na programação *front-end* para Web, traz uma sintaxe mais fluida nas tarefas mais comuns ao programador Web que são: selecionar um elemento no HTML e alterar suas características.

A função $

Uma das mais importantes funções do jQuery é a função **$**. Com ela é possível selecionar com maior facilidade, maior compatibilidade, e com menos código um determinado elemento e seu valor no HTML. Vejam o exemplo:

- Vamos abrir a pasta Aula05 no VSCode.
- Crie um arquivo na pasta JS chamado funcoesJquery.js
- Na raiz do projeto, crie um arquivo HTML chamado jquery.html
    - Inicie o HTML
    - Crie as tags de scripts [jquery-3.4.1.min.js; funcoesJquery.js; bootstrap.js] e link do css do Bootstrap.
    - Vamos criar o formulário igual ao exemplo:
- No arquivo funcoesJquery.js vamos implementar algumas rotinas do Javascript puro para buscar o texto do campo **Nome** e uma rotina utilizando jQuery para a mesma função.
- No jQuery, vamos adicionar um controle de mudança de classe no nosso campo de Idade:
    - Adicionar um botão no HTML e gerar o jQuery:
    - **HTML**

```
<div class="btn btn-warning" id="btnMudarClass">Mudar classe com jQuery</div>
```

    - **jQuery:**

```
$( "#btnMudarClass" ).click(function(){
      $("#idade").addClass('bg-danger text-white');
   });
```

- Agora vamos programar para que se o campo estiver com a classe bg-danger, ele volta a classe original do HTML:

```
 $( "#btnMudarClass" ).click(function(){
      var classe = $("#idade").attr("class");
      if(classe.indexOf("bg-danger") > 0){
         $("#idade").removeClass('bg-danger');
         $("#idade").removeClass('text-white');
```

```
      }else{
         $("#idade").addClass('bg-danger text-white');
      }
   });
```

- Agora vamos trabalhar com uma componente do Bootstrap chamado *Modal*. São *popups* de trabalho. Nosso campo de endereço está desabilitado, vamos digitar os dados de tipo de logradouro, nome e número no *Modal* e o jQuery vai inserir no campo endereço estes dados formatados.
- Adicionar no HTML o código do *Modal* abaixo do fechamento da tag de **<form>**:

```
   <!-- Modal -->
   <div class="modal fade" id="modalEndereco" tabindex="-1" role="dialog" aria-
labelledby="modalEndereco"
      aria-hidden="true">
      <div class="modal-dialog" role="document">
         <div class="modal-content">
            <div class="modal-header">
               <h5 class="modal-title" id="modalEndereco">Endereço:</h5>
               <button type="button" class="close" data-dismiss="modal" aria-label="Close">
                  <span aria-hidden="true">&times;</span>
               </button>
            </div>
            <div class="modal-body">
               <form class="highlight">
                  <div class="form-row">
                     <label for="cmbTipo" class="col-sm-3 col-form-label">Tipo:</label>
                     <div class="col-sm-5">
                        <select class="form-control" id="cmbTipo">
                           <option>Rua</option>
                           <option>Avenida</option>
                           <option>Praça</option>
                           <option>Travessa</option>
                        </select>
                     </div>
                  </div>
                  <div class="form-row">
                     <label for="txtLogradouro" class="col-sm-3 col-form-
label">Logradouro:</label>
                     <div class="col-sm-7">
                        <input type="text" class="form-
control" id="txtLogradouro" placeholder="Nome do logradouro">
                     </div>
                  </div>
                  <div class="form-row">
                     <label for="txtNumero" class="col-sm-3 col-form-label">Número:</label>
                     <div class="col-sm-3">
                        <input type="text" class="form-
control" id="txtNumero" placeholder="Número">
                     </div>
                  </div>
               </form>
            </div>
            <div class="modal-footer">
               <button type="button" class="btn btn-success" id="btnGravarModal">Gravar</button>
               <button type="button" class="btn btn-danger" data-dismiss="modal">Fechar</button>
            </div>
         </div>
      </div>
   </div>
```

- Vamos configurar nosso arquivo de Javascript
  - Chamar o modal
  - Fazer o *get* dos dados do Modal e lançar em tela.
  - Fechar o Modal;

Concluído, vamos trabalhar com nosso arquivo fazendo as validações de campos.

Como atividade para agregar conhecimento, pesquise como se faz as validações de campos de e-mails, datas e senhas.

No Javascript utilizamos o **alert('texto');** para enviar uma mensagem para o operador. Este *alert* resolve nossa demanda, porém visualmente não é muito apresentável. Podemos melhorar... acessem o *site* <https://sweetalert2.github.io/>. Este é uma biblioteca de *alerts* com um visual melhorado. Tentem importar esta biblioteca e trabalhar com ela.