**jQuery**

**Introdução**

Biblioteca JavaScript com a proposta de: escreva menos e faça mais;

Recursos:

Seleção e manipulação de elementos HTML,

Navegação pelo DOM,

Manipulação de CSS,

Eventos: click sobre botões, entrada de texto no input,

Efeitos e animações: efeitos já prontos; ex.: ocultar ou exibir elemento HTML,

Ajax: realização de requisições assíncronas,

Plugins: podem ser reaproveitados dentro das aplicações; por exemplo: ao invés de criar uma galeria de imagens, pode ser utilizado um plugin que já tem essa finalidade.

**Configurando e testando o jQuery**

No terminal: **npm i jquery;**

**jquery.com > https://code.jquery.com/jquery-3.7.1.min.js(salvar link no projeto em que o jquery será utilizado).**

**CDN** – conectar com o link do j Query**:**

[https://releases.jquery.com](https://releases.jquery.com/) > minified > CTRL + C > Copia na tag script

**Seletores – Selecionando elementos HTML parte 1**

Dependendo do objetivo para modificar elemento, eles precisam ser selecionados.

Selecionando elementos por tag - **$(tag)**

// // // classe - **$(.classe)**

// // // id - **$(#id)**

**$** - permite selecionar elementos, acesso aos recursos da biblioteca jQuery;

- permite executar ações da biblioteca jQuery;

- permite o acesso aos recursos da biblioteca jQuery.

**Seletores – Selecionando elementos HTML parte 2**

Praticando a seleção de elementos de um modo um pouco mais avançado.

**-** Combinação de seletores do jQuery;

**-** Como atuar sobre o array de retorno da seleção de elemento.

  //selecionar elementos por classe de tag específica

      console.log($("li.exemplo1"));

retorna a classe **exemplo1** quando ela estiver dentro de um elemento **li**, caso exista **mais** **de** **uma li** com a mesma classe, **retorna** também;

      //selecionar elementos por classe de classe específica

      console.log($(".lista1 .exemplo2"));

retorna os elementos de **qualquer tag** **com** a classe **lista1** que **possui elementos** **com** a **classe** **exemplo2**

      console.log($(".lista1 div.exemplo2"));

retorna os elementos de **qualquer tag com** a classe **lista1** que **possui** elementos **div com** a **classe exemplo2**

//atuação sobre o array de retorno dos elementos selecionados

**:first 🡪** retorna primeiro elemento da seleção

**:last 🡪** retorna último elemento da seleção

**:even 🡪** retorna elementos contidos nos índices pares

**:odd 🡪** retorna elementos contidos nos índices ímpares

Alguns elementos que permite **reaplicar filtro** **no retorno de elementos já filtrados**.

**Sequência de execução de código e $(document).ready()**

*Formas tradicionais com JS puro*

<body onload =”funcao()” />

windows.onload = function() { ... }

*Utilizando jQuery*

$(document).ready(function() { ... })

$(function() { ... })

- Essas, são 2 formas de implementar o controle do momento em que o código JS entra em ação.

$(document).ready(function () {

        //'construi' código javascript no momento em que a página está carregada

        function teste() {

          console.log($("#exemplo"));

        }});

      $(teste); //nao foi colocado o parênteses após a função, pois não será passado nenhum parâmetro, a função está apenas sendo referenciada

$(function () {

        //passando direto a função como parâmetro

        console.log($("#exemplo"));

      });

**Recuperando e manipulando os atributos dos elementos HTML**