**[Instituto Federal de Educação, Ciência e Tecnologia de São Paulo](https://spo.ifsp.edu.br/" \o "Câmpus São Paulo)**

**CAMPUS SÃO PAULO**

**KEVIN NUNES – SP3028739**

**Murilo Benicio Ramos de Araújo - SP3028615**

JQUERY

São Paulo

novembro/2021

**KEVIN NUNES – SP3028739**

MURILO BENICIO RAMOS DE ARAÚJO - SP3028615

JQUERY

Trabalho bimestral da disciplina de Aplicações Web 1 (AW1) utilizado para compor a nota do quarto bimestre do ano de 2021, para a turma 313.

Área de concentração: Aplicações Web 1

Orientador: Johnata Souza Santicioli

São Paulo

Novembro/2021

**Sumário**

[**1. INTRODUÇÃO** 4](#_Toc88554906)

[**2. JQUERY: CONCEPÇÃO** 5](#_Toc88554907)

[**2.1. CONSTRUTOR** 6](#_Toc88554908)

[**2.2. FUNÇÕES** 6](#_Toc88554909)

[**3. ANIMAÇÕES** 8](#_Toc88554910)

[**4. MANIPULAÇÃO DE EVENTOS** 11](#_Toc88554911)

[**5. AJAX** 13](#_Toc88554912)

[**6. JSON** 15](#_Toc88554913)

[**7. Conclusão** 17](#_Toc88554914)

[**8. Referências** 18](#_Toc88554915)

# **1. INTRODUÇÃO**

A pedagogia neste ano letivo, para o ensino de informática, foi um caminho estreito. Pouco tiveram os alunos de uma aprendizagem mais ampla devido ao aperto de prazos importantes e introduções de novos métodos de ensino. Os alunos deviam dobrar os seus esforços e alcançar novos conhecimentos fora da sala de aula, buscando a ajuda de seus colegas quando o professor não estiver presente.

O seguinte trabalho foi uma idealização de tentativas e falhas para chegar a um consenso do funcionamento de uma importante aplicação em JavaScript que resumia linhas de códigos complexos em poucas outras, que deixavam a página criada mais chamativa ou agradável ao usuário.

Os conhecimentos aprendidos aqui deveriam ser mostrados juntos a outras funções de framework, vendo suas funcionalidades vantagens e desvantagens de cada uma possui em algum nicho que o usuário se sujeitar.

A qualidade que se encontra pelo menos é descente nas mínimas funções devido a problemas técnicos que corromperam o arquivo anterior, mas nada que comprometa a apresentação de suas mínimas funções.

# **2. JQUERY: CONCEPÇÃO**

Uma biblioteca criada pelo programador John Resig (hoje Engenheiro de Software do Khan Academy), e apresentada numa palestra do mesmo em 16 de janeiro de 2006, em sua BarCamp. Seu motivo para a criação da biblioteca era a dificuldade que encontrava em criar um programa JavaScript que funcionasse em diversos navegadores.

É de código aberto, possuindo as licenças do MIT e GPL para a utilização de fins comerciais e pessoais sem qualquer meios monetários de compra.

Tal simplicidade, no uso e no acesso, que o framework apresentava para sua época acabou aumentando a sua popularidade entre os usuários, se tornando a maior desse mérito, hoje mantida por equipe de voluntários de uma entidade que carrega o seu nome: a Fundação jQuery.

Framework do JavaScript que facilita na programação de **animações**, **manipulação de eventos**, no uso de requisições do AJAX e do objeto JSON. Pode rodar em qualquer navegador, sem nenhum impedimento, desde que, obviamente, se possua um arquivo *jQuery.js* no diretório.

Muitas vezes ele apresenta esse formato:

query-3.5.1.min.js

Para incluir o arquivo jQuery dentro de um HTML, dentro deste último utilizamos do parâmetro SRC e da tag SCRIPT, inseridas dentro de HEAD. Como mostra o segundo exemplo:

<html>

<head>

<title>...</title>

<!-- Versão development jQuery -->

<script type="text/javascript" src="jquery.js"></script>

</head>

## **2.1. CONSTRUTOR**

A característica central do jQuery é a utilização da função “construtor” (um cifrão com dois parênteses - $()), que se usa para encontrar algum dado desejado dentro de um HTML e na utilização das funções da própria biblioteca. Quando nos encontramos em uma situação de uso de duas bibliotecas cuja as funções sempre são iniciada pelo cifrão, as chamadas são feitas pela escrita jQuery().

Enfim, o que se espera para esta função é que entre os parênteses seja informado o “alvo”, isto é, um dado, uma Tag ou o ID de uma classe, todas sendo postas entre aspas simples:

$(‘para encontrar um elemento’);

$(‘#para\_encontrar\_um\_conteudo’);

$(‘.para\_referenciar\_uma\_classe’) (José, 2013);

## **2.2. FUNÇÕES**

**$(document).ready()**

A linha já apresentada, em resumo possui o método “ready()” como foco. Graças ao $(document), é de se esperar que ele carregue seu conteúdo no mesmo tempo em que a página é carregada. Geralmente acompanhada por uma função anônimos, com os comandos que se deseja executar (José, 2013).

<script type="text/javascript">

$(document).ready(function(){

//o nosso código jquery vai aqui.

});

</script>

Juntando as funcionalidades das funções e do construtor para o código a seguir, conseguimos um resultado semelhante ao da página apresentada.

<head>

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

<title>Introdução jQuery </title>

<script type="text/javascript" src="jquery.js"></script>

<script type="text/javascript">

$(document).ready(function(){

alert("página carregada")

});

</script>

</head>

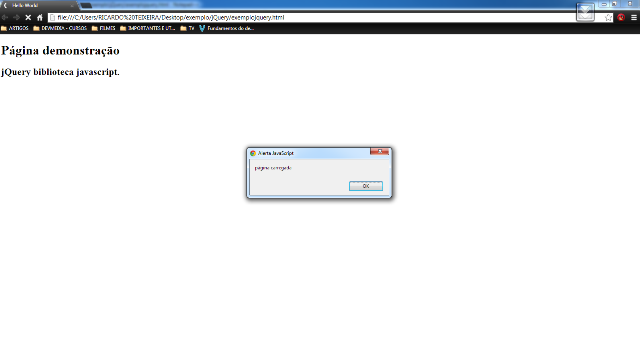
<body>

<h1>Página demonstração</h1>

<h2>jQuery biblioteca javascript.</h2>

</body>

</html>



**3. ANIMAÇÕES**

O jQuery geralmente é usado para a criação de animações que deixam o site mais chamativo. Usando-o no inicio de uma página HTML localizando-o no arquivo que lhe fizemos o download, como é ilustrado (DevMedia, 2013):

<script type="text/javascript"

src="/jquery/jquery-3.5.1.min.js"></script>

<script type="text/javascript">

<html>

<head>

<title>Teste jQuery</title>

<script type="text/javascript"

src="/jquery/jquery-3.5.1.min.js"></script>

<script type="text/javascript">

</script>

</head>

<body>

</body>

</html>

<html>

<head>

<title>Iniciando com jQuery</title>

<script src="https://code.jquery.com/jquery-3.5.1.min.js">

</script>

<script>

$(document).ready(function(){

$("button").click(function(){

$("div").animate({left:'700px'});

});

});

</script>

</head>

<body>

Clique no botão para animar a div.

<div style="background:red;height:10px;width:10px;position:absolute;">

</div>

<button>Clique aqui</button>

</body>

</html>

Na criação de scripts customizados, se cria um arquivo com nome semelhante à meuscript.js no mesmo diretório (/JQuery, no caso). O esqueleto desse script irá se parecer com isso (DevMedia, 2013):

$(document).ready(function() {

// seu código sera adicionado aqui

});

Código para o meuscript.js:

$(document).ready(function() {

$("div").click(function() {

alert ("Parabéns, o código está funcionando.");

});

});

Agora se pode escrever o meuscript.js como segue:

<html>

<head>

<title>Animações com jQuery</title>

<script type="text/javascript"

src="/jquery/jquery-3.5.1.min.js"></script>

<script type="text/javascript"

src="/JQuery/meuscript.js "></script>

</head>

<body>

<div>

Clique aqui par aver a mensagem.

</div>

</body>

</html>

**4. MANIPULAÇÃO DE EVENTOS**

Denominação para as funções que jQuery possui para manipular o clicar de um botão, o sobrepousar o mouse, o selecionamento de um menu, etc., ações feitas na interface com o usuário, sem atualizar toda a página do HTML.

O propósito de cada evento é iniciar alguma ação, como no código abaixo, com uma lista de opções dentro (DevMedia, 2013):

<html>

<head>

<script src="http://code.jquery.com/jquery-3.5.1.min.js"></script>

</head>

<body>

Selecione seu personagem favorite:

<select name="Comics" multiple="multiple">

<option>Thor</option>

<option>Hulk</option>

<option>Spider man </option>

<option>Batman</option>

</select>

<div></div>

<script>

$("select").change(function () {

var escolhido = "";

$("select option:selected").each(function () {

Escolhido += $(this).text() + " ";

});

alert("Você prefere o "+escolhido);

})

.change();

</script>

</body>

</html>

No código acima, a tag do script chama a função que produz um box de alerta baseado no valor selecionado da lista de opções. Este valor é guardado na variável favorita e é mostrado com o valor da *string* que é anexada (DevMedia, 2013).

**5. AJAX**

Implementando as funcionalidades do programa Ajax (*Assynchronus JavaScript and XML* - JavaScript e XML assíncronos) no jQuery nos é permitido carregar informações nos servidores sem a necessidade de atualizar a página, já que agora ela ocorre de maneira assíncrona, utilizando o método *Load()* (DevMedia, 2013).

Tais chamadas assíncronas, ao iniciadas, fazem o ciclo de código não ser interrompido até que a resposta seja obtida. A reposta é conquistada em um momento posterior, após dar início a requisição, independentemente, e desenvolvido por meio da função (Rodrigo, 2016).

Uma das funções mais utilizadas é a *$.ajax()*, que permite enviar e tratar o resultado de requisições assíncronas (Rodrigo, 2016). No código abaixo, ele é usado para carregar dados estatísticos num arquivo HTML de nome JQueryajax.html (DevMedia, 2013).

<html>

<head>

<title>jQuery e Ajax</title>

<script type="text/javascript"

src="/jquery/jquery-3.5.1.min.js"></script>

<script type="text/javascript" language="javascript">

$(document).ready(function() {

$("#driver").click(function(event){

$('#stage').load('/JQuery/conteudo.html');

});

});

</script>

</head>

<body>

<p>Clique no botão abaixo para carregar o conteúdo:</p>

<div id="stage" style="background-color:red;">

Conteúdo

</div>

<input type="button" id="driver" value="Carregar" />

</body>

</html>

Para executar este código é necessário criar um arquivo HHTML no diretório jQuery que foi criado anteriormente, de nome conteúdo.html, que deverá possuir o código abaixo:

<html>

<head>

<title>Página para teste</title>

</head>

<body>

Texto da página de conteúdo.

</body>

</html>

No clique do botão “*Load Some Text*”, em JQueryajax.html, será carregada a informação estática, do corpo de “output.html”.

Vamos dar uma olhada acima, no “JQueryajax.html”, vamos dar o método Load().

***$(‘#stage’).load(‘/JQuery/conteúdo.html’);***

O *Load()* aceita o caminho do output.html como seu parâmetro e pede que o seu conteúdo seja carregado na tela. Isso irá fazer com que o arquivo HTML carregue os conteúdos de outro arquivo de mesmo tipo (DevMedia, 2013).

**6. JSON**

Para implementarmos dados de um arquivo *json.*

Para isso, será preciso criar um arquivo HTML como o abaixo, esse de nome JQueryson.html (DevMedia, 2013)

<html>

<head>

<title>Trabalhando com JSON</title>

<script type="text/javascript"

src="/jquery/jquery-3.5.1.min.js"></script>

<script type="text/javascript" language="javascript">

$(document).ready(function() {

$("#driver").click(function(event){

$.getJSON('/JQuery/dados.json', function(v) {

$('#stage').html('<p> Filme 1: ' + v.filme1 + '</p>'); $('#stage').append('<p>Filme 2 : ' + v.filme2+ '</p>');

$('#stage').append('<p> Filme 3: ' + v.filme3+ '</p>');

});

});

});

</script>

</head>

<body>

<p>Clique no botão para carregar os dados</p>

<div id="stage" style="background-color:red;">

Filmes:

</div>

<input type="button" id="driver" value="Carregar " />

</body>

</html>

O arquivo criado será “dados.json” no diretório / JQuery. O arquivo json deve parecer com este:

{

"filme1": "O Homem de Aço",

"filme2" : "Os Mercenários",

"filme3": "O Hobbit"

}

No código anterior, a função *$.getJSON()* pega o caminho /JQuery/dados.json, que é o caminho do arquivo json, que é o arquivo que o usuário criou no diretório JQuery.

Então a função *$.getJSON()* aceita outra função com parâmetro v como seu próprio parâmetro.

A variável v pega os valores que são pegos no arquivo. Os valores guardados nessa variável são filme1, filme2 e filme3. Então , quando o código chama o valor pela chamada de suas respectivas chaves, digamos v.filme1, etc, os nomes respectivos são mostrados (DevMedia, 2013).

# **7. Conclusão**

Foi um projeto meio conturbado em seu desenvolvimento. Seus principais integrantes, mantiveram boa comunicação na durante as semanas, mas os interesses de cada em focar em outras matérias e outras atividades complexas comprometeram a execução de um trabalho de qualidade maior.

Mas mesmos assim ambos os usuários conseguiram aumentar um pouco de seus conhecimentos de uma nova biblioteca e a sua funcionalidade principal, que será usada para seus futuros e prováveis projetos.

# **8. Referências**

DevMedia. (12 de julho de 2013). *jQuery básico: Visão geral da biblioteca.* Fonte: DevMedia: https://www.devmedia.com.br/jquery-basico-visao-geral-da-biblioteca/28565

José. (23 de fevereiro de 2013). *jQuery Tutorial.* Fonte: DevMedia: https://www.devmedia.com.br/jquery-tutorial/27299

Rodrigo. (11 de outubro de 2016). *Ajax com jQuery: Trabalhando com requisições assíncronas.* Fonte: DevMedia: https://www.devmedia.com.br/ajax-com-jquery-trabalhando-com-requisicoes-assincronas/37141