Formulários

Formulários

Um dos principais motivos pelos quais a linguagem de programação JavaScript foi criada foi a necessidade de validar os dados dos formulários diretamente no navegador do usuário.

Usando JavaScript, é possível gerenciar os valores e eventos do formulário e dos elementos dele.

É possível selecionar um formulário utilizando os métodos getElementById(), getElementsByTagName() ou querySelector().

```
var form = document.getElementById(idFormulario);
```

Formulários - document.forms

A propriedade forms do objeto document retorna uma coleção de objetos tipo array contendo todos os formulários existentes no documento HTML, com índice 0 para o primeiro formulário por ordem de exibição, índice 1 para o seguinte e assim sucessivamente.

```
var formulario = document.forms[0];
```

Formulários - document.forms.elements

A propriedade elements de cada objeto form obtido através de document.forms retorna uma coleção de objetos tipo array contendo todos os campos existentes no formulário concreto ao qual fazemos referência, com índice 0 para o primeiro elemento por ordem de exibição, índice 1 para o seguinte e assim sucessivamente.

Formulários - document.forms.elements

```
// retorna todos os elementos do form

var elementos = form.elements;
// retorna o elemento do formulário que está no
índice indicado

var elemento = form.elements[indice];
```

Formulários - Atributos

O objeto formulário tem atributos **como action, target, encoding e method**. Como são objetos de JavaScript, eles podem ser acessados da mesma forma que qualquer outro objeto.

Action: Estabelece a URL do documento que vai processar as informações enviadas pelo formulário.
Exemplo: form.**action**;

Target: Estabelece o nome ou a palavra reservada a partir da qual é necessário mostrar a resposta depois do envio do formulário.
Exemplo: form.**target**;

Formulários - Atributos

Encoding: Estabelece o tipo MIME usado para criptografar os dados.
Exemplo: form.**encoding**;

Method: Estabelece o método de HTTP que será usado para enviar os dados. Pode ser get ou post.
Exemplo: form.**method**;

Name: Estabelece o nome do formulário.
Exemplo: form.**name**;

Formulários - Elementos - Value

Os elementos do formulário têm um atributo chamado value que permite estabelecer ou obter o valor de um elemento.

```
//obtemos o valor do elemento
elemento.value;
```

```
// estabelecemos o novo valor do elemento.
elemento.value = novoValor;
```

Formulários - Eventos

Por meio do evento **submit** do formulário, podemos **enviar os dados carregados** no formulário para o documento estabelecido no atributo **action** do formulário.

```
form.onsubmit = function() {
        // o formulário foi enviado
}
```

Formulários - Eventos

Com JavaScript, é possível gerenciar o evento **onsubmit() ou submit.** Assim como o restante dos eventos, é possível controlar que **o evento não seja executado por padrão** com o método do evento **preventDefault()**

```
form.onsubmit = function(evento) {
evento.preventDefault();
}
```

Formulários - Elementos - Input

Os elementos input **type=text** têm uma propriedade chamada value que permite obter ou estabelecer o valor de um campo de texto.

```
// retorna o valor do elemento
Elemento.value;
```

```
// estabelece um novo valor
elemento.value = valor;
```

DigitalHouse > Coding School

Formulários - Elementos - Select

Através dos elementos select, podemos obter o valor escolhido. O objeto select tem o atributo **selectedIndex** que retorna o índice numérico da opção selecionada.

Outro atributo do objeto **select** é **options**, que retorna a coleção de elementos **options**.

Formulários - Elementos - Select / options

Combinando esses dois atributos, podemos obter o valor do option selecionado no elemento select.

Assim como os outros elementos HTML do formulário, o objeto option tem um atributo value que retorna o valor dele.

Formulários - Elementos - Select / options

select.**selectedIndex**; // retorna o índice do valor selecionado

select.**options**; // retorna a coleção de elementos options

select.**options[indice]**; // retorna o option selecionado

select.**options[indice].value**; // retorna o valor do elemento selecionado.

Formulários - Elementos - Radio

O objeto formulário também tem objetos radio button (**type=radio**). No geral, esse elemento é utilizado para selecionar entre vários valores sendo que apenas um pode estar selecionado. Para conseguir isso, utilizamos o atributo **name**.

Por meio do seletor querySelector(), podemos obter o elemento **radiobutton** que está ativo. Para isso, devemos utilizar um seletor de CSS :**checked**.

var **radioButton** = document.querySelector(seletor :checked);
radioButton.value; // obtemos o valor do elemento

Formulários - Elementos - Checkbox

Outro dos elementos do formulário é checkbox. Esse objeto tem um atributo checked que retorna um valor booleano (true/false) através do qual é possível saber se ele foi selecionado.

O atributo checked pode ser utilizado tanto para obter o valor do objeto quanto para estabelecer um novo valor.

checkbox.checked; // retorna o valor do objeto checked

checkbox.checked = true/false; // estabelece um novo valor para o elemento.

Formulários - Elementos - Hidden

Outro dos elementos do formulário é input type=hidden. Esse elemento permite incluir um valor oculto no formulário. O usuário não verá esse elemento, mas como faz parte do formulário, ele será enviado junto com o restante dos dados.

Assim como o restante dos elementos do formulário, o input type=hidden tem um atributo value que retorna ou permite estabelecer o valor do campo oculto.

Formulários - Elementos - Hidden

```
elementoOculto.value;

elementoOculto.value = valorOcultoComoString;
```

Formulários - Elementos - Textarea

Outro dos elementos do formulário é textarea. Esse elemento permite escrever várias linhas.

Com JavaScript, é possível obter e estabelecer o valor de um objeto textarea por meio do atributo value.

```
textarea.value; // obter o valor de textarea

textarea.value = novoValor; // estabelecer o novo valor de textarea
```

Formulários - Elementos - Eventos

Os elementos do formulário podem gerenciar eventos por meio dos métodos: **onfocus, onblur, onchange, oninput**. Eles também podem ser escritos usando o método addEventListener(callback).

Focus: Este evento é disparado quando um elemento é enfocado

Blur: Este evento é disparado o foco é removido de um elemento

Change: Este evento é disparado quando o valor de um elemento muda

Input: Este evento é disparado quando inserimos dados em um elemento

Formulários - Elementos - Eventos

```
elemento.onfocus = function() {
// código que gerencia o focus do elemento
}

elemento.onblur = function() {
// código que gerencia o blur do elemento
}
```

Formulários - Elementos - Eventos

```
elemento.onchange = function() {
// código que gerencia a mudança de valor
}

elemento.oninput = function() {
// código que gerencia a entrada de dados em um elemento
}
```

Formulários - Elementos - Eventos

Por meio do evento **change()**, é possível gerenciar a mudança de seleção de um elemento select.

```
elementoSelect.onchange = function() {

var index = elementoSelect.selectedIndex;

var option = elementoSelect.options[index].value;

}
```

DigitalHouse > Coding School

Formulários - Validação

Por meio do atributo **value**, podemos obter o valor de um elemento.

Além disso, é possível utilizar o atributo **length** do objeto string para saber qual é o **comprimento** dele. Dessa forma, é possível validar se o elemento está vazio.

```
if (elemento.length === 0) {
        // elemento vazio
}
```

Formulários - Regex

JavaScript tem expressões regulares como tipo de dado e permite encontrar padrões de texto em uma string.

No geral, é possível utilizá-los para validar os campos de texto.

As expressões regulares ou regex têm um método test() que permite validar uma string para saber se ela cumpre a expressão regular.

Formulários - Regex

```
// valida que a string tenha números e letras

var regexCampoVazio = /^[a-z0-9]+$/i;

// valida uma estrutura de e-mail. Existem muitas opções para fazer isso

var regexMail = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$/;
```

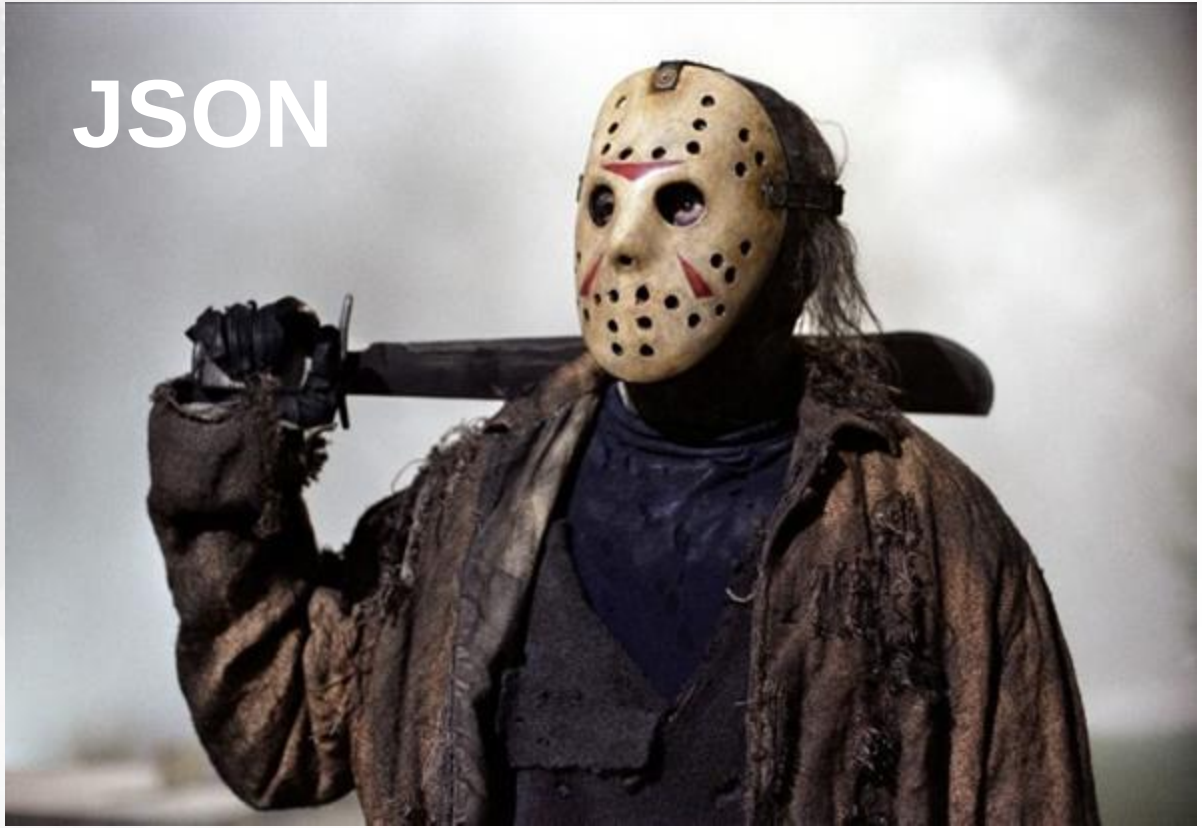
Formulários - Regex

// retorna um valor boolean, dependendo se ele cumpre ou não com a expressão regular.

regexCampoVazio.test(textoParaBuscar);

regexMail.test(textoParaBuscar);

Ajax


JSON

Formulários - Json

O formato de dados de JavaScript Object Notation, ou JSON para abreviar, é derivado dos literais da linguagem de programação JavaScript. Esse formato de troca de dados é um subconjunto da linguagem JavaScript, e não uma linguagem de programação.

Sua estrutura é muito parecida a um objeto literal de JavaScript, com algumas diferenças. Os nomes das propriedades são escritos entre aspas duplas, assim como os valores de strings.

```
var objetoEmFormatoJSON = '{ "atributo": "valor", "atributo1": 1, "atributo2": [],
"atributo3": null, "atributo4": false }';
```

Formulários - Json

JavaScript tem um objeto JSON com os métodos stringify() e parse().

Stringify: O método stringfy permite passar um objeto ou valor de JavaScript para o formato JSON.

Parse: O método parse pega uma cadeia de caracteres em formato JSON e transforma em um objeto ou valor de JavaScript.

Graças a esses dois métodos, é possível utilizar o formato JSON para a troca de dados em formato de texto.

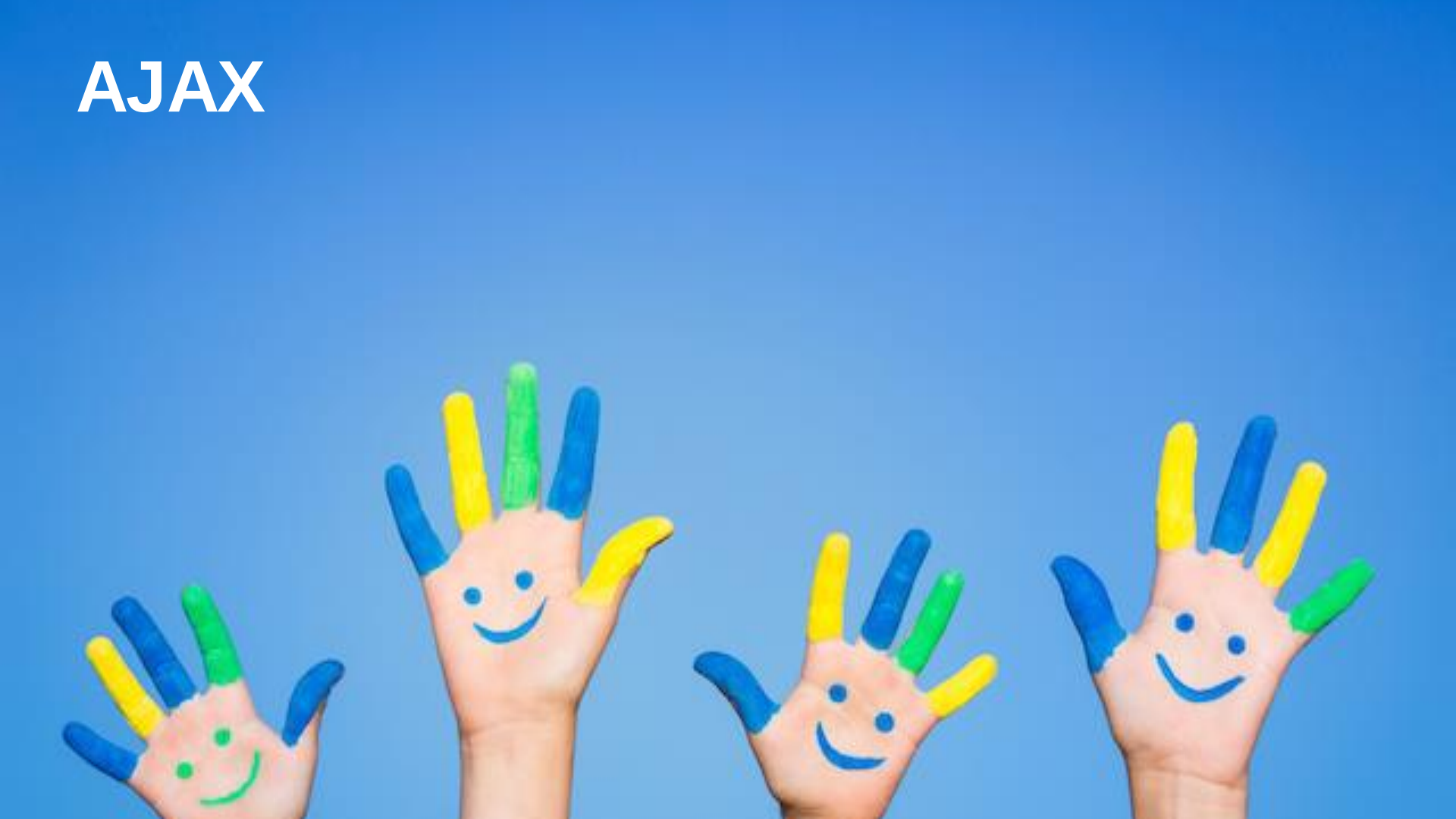
Formulários - Json

É possível acessar os atributos de uma variável em formato JSON da seguinte maneira:

var objetoEmFormatoJSON = JSON.parse("atributo2": [{“numero”:1},{“numero”:2}]’);

objetoEmFormatoJSON.atributo2[1].numero

AJAX



AJAX

Ajax não é uma tecnologia, mas sim um termo implementado por Jesse James Garrett em 2005.

Ajax significa Asynchronous JavaScript and XML e se transformou em sinônimo do desenvolvimento moderno de front-end por um bom motivo. Essa técnica oferece a possibilidade de iniciar solicitações HTTP, como GET e POST, sem precisar sair da página web atual.

Por isso, podemos dizer que Ajax é uma técnica para criar páginas dinâmicas.

AJAX

AJAX permite:

- Atualizar o conteúdo de uma página sem recarregar
- Pedir informações a um servidor
- Receber informações de um servidor
- Enviar informações a um servidor

AJAX - Passos

1. var xmlhttp = new XMLHttpRequest();

AJAX - Passos

O único método global da interface XMLHttpRequest é o do construtor que, quando invocado, retorna ao aplicativo uma nova instância do objeto XMLHttpRequest. Através da interface herdada por esse objeto, vamos iniciar e gerenciar solicitações.

Como a solicitação HTTP é produzida de forma assíncrona, é necessário que o aplicativo seja notificado sobre qualquer mudança de estado enquanto a solicitação estiver vigente, por exemplo se a resposta tiver sido fornecida ou se a conexão tiver expirado.

Essa é uma forma fácil de obter informações de uma URL sem precisar recarregar toda a página. XMLHttpRequest é muito usado na programação AJAX.

AJAX - Passos

2.

```
xmlhttp.onreadystatechange = function() {
if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                        console.log(xmlhttp.responseText);
                        //Meu código
                        }
};
```

AJAX - Passos

```
xmlhttp.onreadystatechange = function() {
if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                        console.log(xmlhttp.responseText);
                        //Meu código
                        }
};
```

AJAX - Passos

xmlhttp.onreadystatechange

Uma função do objeto JavaScript invocado quando o atributo readyState muda. O callback é invocado a partir da interface do usuário.

xmlhttp.readyState

0 se não se inicializou, 1 se está carregando, 2 se a solicitação já foi enviada, 3 se a resposta está sendo baixada e 4 se terminou

AJAX - Métodos

xmlhttp.status
O status da resposta ao pedido. Este é o código HTTPresult (por exemplo, o status é 200 no caso de uma solicitação bem-sucedida). Somente leitura.

xmlhttp.responseText
A resposta ao pedido como texto, ou null se o pedido não teve sucesso ou se ainda não foi enviado. Somente leitura.

AJAX - xmlhttp.open("GET", "url", true);

Inicializa o pedido. Este método deve ser usado a partir do código JavaScript.

method
O método HTTP que será usado: "POST" ou "GET". É ignorado para URLs que não são de HTTP.

url
A URL para a qual o pedido é enviado.

async
Um parâmetro opcional, booleano, que por padrão é true. Indica se a operação é realizada de forma assíncrona

AJAX - Passos

```
var xmlhttp = new XMLHttpRequest();

xmlhttp.onreadystatechange = function() {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                console.log(xmlhttp.responseText);
                //Meu código
        }
};

xmlhttp.open("GET", "url", true);
xmlhttp.send();
```

Envia o pedido

AJAX - Parâmetros

```
xhttp.open("POST", "ajax_info.txt", true);

xhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");

var params = “nomes=João&sobrenomes=Silva”

xhttp.send(params);
```

Até a próxima aula!