**Programa DIO**

**Formação HTML Web Developer**

**Trabalhando com Formulários**

Tag **< form >**

Formulário no HTML é um trecho onde se tem campos para que o usuário insira informações.

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente Assim, pode-se colocar um campo onde se insira texto, número, múltipla seleção, área de texto um pouco maior, campo de definição para ativo ou inativo, etc.

Formulário caminha muito próximo dos servidores, pois, além de gerar o formulário para o lado do cliente, na hora que o usuário inserir a informação e clicar no botão de enviar, por exemplo, a informação vai para o servidor, que por sua vez irá processar a informação, que poderá ou não gravar as informações em um banco de dados

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Atributos

* **Name** = serve para se dar um nome a tag.
* **Action** = é a ação, no caso da tag form, significa que será determinado para onde os dados inseridos no formulário irão. Colocando o endereço do servidor.
  + Obs. não se pode enviar formulário para um site que não seja dono. Esse controle de segurança é implementado pelo browser.
* **Method** = possui 2 valores “**gest**” e “**post**”:
  + **Get:**  utilizando o method get, ao clicar no botão de enviar as informações do formulário, ele enviará as informações pela URL
  + **Post:**  envia informações do formulário, sendo primeiro passo mandar uma requisição para o servidor, e esse corpo da requisição (body do servidor) vai estar embutido as informações. Assim, as informações não vão aparecer no endereço (url).

**Obs. Assim o método post se torna mais seguro para enviar as informações mais sensíveis.**

* **Autocomplete** = deixa completado as informações de determinados campos, pois, ficam gravados no navegador. Autocomplete possui 2 valores “**off**” e o “**on**”.
  + On – deixa ativado o autocomplete.
  + Off – deixa desativado o autocomplete.
* **On + alguma coisa**: no HTML tudo que possui “on”, significa evento e no formulário usa-se mais o “**onsubmit**”, nele será embutido um código JavaScript. O onsubmit é quando se clica no botão de enviar e o que ele fará depois? Assim, no JavaScript é onde estará determinado sua reação.

Tag **< input >**

É um campo na tela onde o usuário poderá colocar informações, ou seja, inserir valores.

Obs. por conta de compatibilidade entre navegadores, tem alguns tipos de inputs que não irão funcionar.

O input possui vários tipos e valores:

* < input **type =”text”** /> - é um campo texto
* < input **type=”number”** /> - que por sua vez possui atributo **min =”0”** e **max=”99”,** que delimita o range.
  + Como também o atributo **step=”5”** que é o quanto será contado por vez que clicar par aumentar ou diminuir.

Exemplo:

Idade: <input type="number" name="age" min="0" max ="140" step="10" />

* < input **type=”button”** /> button possui 2 formas de ser escrito:
  + Possui value=”Enviar” – mas semanticamente coloca-se a tag **<button>Enviar</button>**

Ou: **<input type="button" value="Enviar"/>**

* + Há também 3 tipos de botões no HTML, e esses botões vão agir de uma forma semelhante, mas com funcionalidades diferentes:
    - Type=”**button**” – sintaxe : <button **type=”button”**> Clicável </button>
      * Simplesmente será criado um botão. E seu funcionamento somente ocorrerá após colocar uma funcionalidade pelo JavaScript.
        + Ex. <button type=”button” **onclick=”alert(‘cliquei aqui’)”**> Clicável</button>
        + Pode-se colocar eventos, por exemplo: <button type=”button” **onmouseover=”alert(‘passei aqui’)”**> Clicável</button>
    - Type=”**reset**” – basicamente é o limpar. Ele irá limpar os campos do <form> que ele está dentro.
      * Pode-se através de JavaScript alterar sua funcionalidade.
    - Type=”**submit**” – ele basicamente irá enviar o formulário
      * Obs. nos formulários, quando se dá um submit é fundamental colocar um evento chamado “**onsubmit**” que se coloca na **tag <form>**. E nesse submite poderá colocar evento para validar envio do formulário. Assim, pode-se validar se os campos estão preenchidos corretamente e avisar o usuário.

Obs. o enter no formulário define o botão submite, por isso, precisa colocar validação.

Obs. convencionalmente os campos obrigatórios coloca-se asteriscos “\*”.

Obs. no formulário por exemplo ao se editar um cadastro, e não deixar habilitado o password por exemplo, um pessoa mal-intencionada poderá abrir o console do html e retirar o atributo “disabled” e assim redefinir a senha do usuário sem ele saber, burlando o sistema.

Por isso precisa validar também no servidor essa alteração, e não só no front end que valida as coisas. E também antes de tratar os dados pode-se validar esses dados novamente. Ou seja, validar também no front end quanto no back end justamente por conta da segurança

Obs. Em termos de função eles fazem a mesma coisa. Mas em termos de semântica ao se olhar a tag <button> já se sabe que é um botão. E que por sua vez a tag button terá seus próprios atributos, ou seja, tipos de botões.

* < input **type=”range”** /> - ele cria um campo scroll (tipo rolagem que serve para aumentar ou diminuir, mudar de cor por exemplo). Assim, pode-se colocar valor min e max e rola entre eles.
* < input **type=”color”** /> - não estabelece mínimo e máximo, mas ele abrirá uma caixa onde disponibilizará as cores.

Obs. cada tipo de navegar implementa de forma diferente o atributo, mas todos podem ser formatados via CSS.

Dúvida, se formatar via CSS padroniza?

* < input **type=”email”** /> - há problema com relação a compatibilidade, pois, por padrão ele estabelece filtro, onde por exemplo só aceita inserção de endereço de e-mail válido. Assim, se estiver faltando @ já irá aparecer mensagem padrão informando o usuário que está faltando o @. Mas também pode-se criar o próprio validador.
* < input **type=”url”** /> - mesa ideia vale para o atributo url, onde precisa colocar http na frente.
* < input **type=”date”** /> - pega o idioma que está definido no navegador para colocar calendário para selecionar a data.
* < input **type=”week”** /> - escolhe a semana do ano

Obs. no FireFox o week não funciona, só funciona no Chrome e Edge.

* < input **type=”checkbox”** /> - utiliza-se valores booleanos
  + A ideia do checkbox é selecionar múltiplos valores, porém, na hora que vai transferir para o servidor, precisa informar quais valores foram selecionados. Assim, o checkbox sendo uma variável contendo múltiplos valores
    - Ex. <input type="checkbox" name="opcional" value="queijo"/>
  + Só que tem um porém se colocar mesmo name somente, ao for enviado para o servidor (linguagem php, node js) no método get como vai para Url, o servidor não entendera os checkbox como uma lista, caso mais de uma opção seja marcada, e pegará somente o último valor. Pois, entenderá que a variável (name) foi troca e pegará a última troca.
  + Assim, uma forma de se fazer com que todos sejam enviados, coloca-se os colchetes depois do valor do name. Pois, os colchetes indica para o servidor que esses checkbox são uma lista.
    - <input type="checkbox" name="opcional[]" value="queijo"/> + queijo

Obs. enviando vários checkbox com mesmo nome e diferentes valores precisa-se enviar via post e não com método get na tag <form>

* <input type="**radio**"/> - marcadores onde se aceita apenas a seleção de uma caixa. Para funcionar elas precisam ter o mesmo nome “**name**”.

Atributo “**disabled**” faz com que deixa determinada caixa desabilitada.

Ex.  <input type="checkbox" **disabled** name="opcional[]" value="queijo"/> + queijo

Referencias

<https://www.w3schools.com/html/html_form_input_types.asp>

* <input type="**hidden**"/> - é um campo que fica invisível para o navegador (usuário) mais por trás ele está sendo enviado ao se clicado em enviar.
* <input type="**file**" /> - campo de enviar arquivos. E quando se coloca o atributo “**multiple**” consegue-se selecionar vários arquivos para serem enviados. Se não colocar esse atributo por padrão só pode selecionar 1 arquivo. Então mesmo apertando CTRL não deixa selecionar mais de um arquivo.
* <input type="**search**" /> - campo de busca. não é muito utilizado, pois, está disponível em poucos navegadores, como ex. Chrome.

Tag **< select >**

Se estiver em uma estrutura de formulário onde o preenchimento do campo, tipo de área pretendida, por exemplo, precisa ser preenchida, e se deixar sem nenhum filtro de validação, o usuário poderá escrever “banana” se ele quiser e submeter o formulário. Então, faça o campo com valores já pré definidos.

Ex. <select name="">

* A caixa pré definida conterá as opções para escolher, e utiliza-se **tag filha** chamada “**option**”.
  + Obs. a primeira opção do option pode ser valor vazio, com texto para informar o usuário.

<option **value=""**> Selecione o cargo </option>

Obs. quando se usa o atributo “selected” a opção já aparece primeiro

* Atributo “**multiple**” do select

Tag **< textarea >**

É um campo texto onde é possível colocar várias informações.

Obs. o usuário pode modificar o tamanho da caixa.

Pode-se estabelecer limite da área do texto via CSS, definindo as dimensões.

* Atributo “**rows**” – defini quantidade de linhas que aparecerá de início para o usuário.
* Atributo “**cols**” – são quantidade de colunas que inicialmente aparecerá para o usuário.