Formularios

Formulario basico

Todo formulario precisa estar entre as tags <form> e </form>:

<form> </form>

Para fazer uma caixa que possibilita escrever dentro basta declarar a tag <input>, tem vários tipos de input, um deles é o text, junto do text é preciso declarar o name=” “ e o id=” “, ou seja, toda caixa do tipo text precisa ter o name e o id:

<input type=”text” name=” “ id=” “>

Para fazer um botão de enviar basta declarar um input do tipo submit, junto do submit é preciso declarar o value=” “, o value é o que vai estar escrito dentro do botão / no botão:

<input type=”submit” value=” “>

Para desligar o autocomplete do formulario basta declarar a propriedade autocomplete como off na tag <form>:

<form autocomplete=”off”>

</form>

```
<form autocomplete="off">
    <input type="text" name="nome" id="nome">
    <input type="text" name="sobrenome" id="sobrenome">
    <input type="submit" value="Enviar">
</form>
```

Resultado:

Para onde os dados são mandados

Normalmente os dados são mandados para alguma linguagem de programação: js, php etc.

Para enviar os dados pra alguma linguagem de programação, é preciso declarar a propriedade action na tag <form>, action significa que alguma ação vai ser disparada quando clicarmos no botão de submit:

<form action=” “>

</form>

Labels (etiquetas)

Tem a função de criar uma ligação entre dois elementos, sendo que um deles vai ser um elemento de entrada. O label vem com um unico parametro chamado for, que significa “pra quem”, ou seja, “um label pra quem”, o for recebe o mesmo valor do id ou do name que esta no input, é necessario escolher um dos dois, mas geralmente é o id que no final vai funcionar como uma etiqueta.

Uma das vantagens do label é que a area de toque fica maior, aumentando assim a sensibilidade, que pra celulares é muito importante

<label for=” “> </label>

Diferença entre name e id

Para a linguagem de programação PHP funcionar direito, é necessario que todo objeto de entrada tenha o atributo name=” “ e para linguagens de programação como JavaScript o id=” “ é mais utilizado / importante, pois funciona melhor com ele, ou seja, name=” “ mais util para PHP e Html, id=” “ mais util para JavaScript

e Label. Então é preciso declarar o id e o name para todos os campos de entrada, para melhor identificação

---

Metodos GET e POST

Existem dois tipos de metodos que são metodos simples pra envio de formulario, o primeiro é o metodo padrão GET, ou seja, quando não colocamos nada ele fica com o metodo GET

<form method=”GET“>

</form>

Se não quisermos o metodo padrão, existe tambem o metodo POST, que tem a função de não mostrar suas informações na URL, fazendo com que elas fiquem ocultas na URL

<form method=”POST”>

</form>

Quando usar GET ou POST:

GET: O metodo GET sera utilizado quando os seus dados não forem sensiveis, ou seja, quando não for senha, cep, endereço, numero de cartão de credito, imagem etc. Outra utilidade do GET é que ele pode ser compartilhado, porem o GET só pode enviar até 3 mil bytes de dados (letras), caso queira enviar mais que isso, o GET não vai ser o metodo correto

POST: O metodo POST sera utilizado quando os seus dados forem sensiveis, ou seja, quando for senha, cep, endereço, numero de cartão de credito, imagem etc. Quando você quiser que os dados não apareçam na URL, quando os seus dados tiverem mais de 3 mil bytes ou precisar fazer envio de arquivos. Ao contrario do GET o POST não pode ser compratilhado

_____

AVISO: Se for usar dados sensiveis, o ideal mesmo é aprender sobre HTTPS

Atributos

Required - Significa requerido ou obrigatorio, faz com que seja obrigatorio preencher os campos de formulario, caso os campos não estejam preenchidos ele vai barrar qualquer ação, essa propriedade é colocada dentro da tag <input>

minlength=" " - Define o minimo de dados que o formulario é obrigado a ter, ou seja, o tamanho minimo que o conteudo deve ter, essa propriedade é colocada dentro da tag <input>

maxlength=" " - Define o maximo de dados que o formulario deve ter, ou seja, o tamanho maximo que o conteudo vai ter, essa propriedade é colocada dentro da tag <input>

min=" " - tem a mesma funcionalidade e comportamento do minlength

max=" " - tem a mesma funcionalidade e comportamento do maxlength

size=" " - Define o tamanho da caixa, ou seja, faz com que a caixa tenha o tamanho para caber uma quantidade de dados especifica, quantidade de dados que ele vai mostrar por vez, essa propriedade é colocada dentro da tag <input>

placeholder=" " - Define o que vai aparecer dentro da caixa antes de algo ser digitado, é basicamente um valor / dica que aparece, quando começar a digitar essa dica desaparece, essa propriedade é colocada dentro da tag <input>

autocomplete=" " - Define se vai aparecer sujestões ou não, essa propriedade é colocada dentro da tag <form>

autocomplete=" " - quando colocada dentro da tag input, ela vai poder receber muito mais opções, o autocomplete dentro do input é pra dizer qual o papel que ela exerce, qual a função dela, se é pra uma nova senha (new-password), usuario (username), endereço etc. Pra campos que são padronizados em formulario (só vai funcionar se o autocomplete do <form> estiver ligado (on))

step=” “ - de quanto um numero vai mudando, ou seja, configura quais tipos de numero serão aceitos, se vai aceitar numeros quebrados, numeros inteiros etc. De quanto ele vai pulando, andando. Serve pro input com o tipo number

value=” “ - Define um valor que já vai iniciar dentro da caixa, ou seja, a caixa já vai aparecer preenchida com algum valor. Lembrando que o placeholder desaparece quando esse atributo é declarado

checked - Faz com que já apareceça marcado (checado), ou seja, já começa marcado

label=” “ - Recebe algum nome

selected - Faz um item já aparecer selecionado, ou seja, já começa selecionado

list=” “ - esse atributo é colocado no input

oninput=” “ - quando entrar com dados algo vai acontecer, ou quando mexer com algum input algo vai acontecer, esse atributo é colocado dentro do input e geralmente é usado com js

---

Input de senha

Para fazer um input que receba senhas basta declarar um input do tipo password:
<input type=”password”>

As vantagens do input tipo password é que ele esconde o conteudo que esta dentro da caixa

---

Botão de limpar

Para fazer um botão que limpa o que esta dentro da caixa basta declarar um input do tipo reset:

<input type=”reset” value=” “>

Quando for clicado ele vai limpar todos os dados que foram colocados no input e que estiverem entre o <form> e </form> dele

---

Input que aceita apenas numeros

Para fazer um input que aceita apenas numeros basta declarar o input do tipo number:

<input type=”number”>

Essa caixa vai aceitar apenas numeros, negando outros tipos de valores, seja simbolo, emoji ou letra

---

Quando o input for do tipol numerico, não são aceitos os atributos minlength nem maxlength, apenas o min e o max que tem o mesmo objetivo e funcionalidade

---

Inputs de data

Para fazer um input que vai conter o mês e o ano, basta declara um input do tipo month:

<input type=”month”>

Vai dar a possibilidade de escolher o mês (por extenso) e o ano: -mês- de -ano-

---

Para fazer um input que vai conter o dia, mês e ano basta declarar um input do tipo date:

<input type=”date”>

Vai dar a possibilidade de escolher o dia, mês (numerico) e ano: dia/mês/ano

---

Para fazer um input que vai conter a hora, basta declarar um input do tipo time:

<input type=”time”>

Vai dar a possibilidade de escolher a hora

Inputs de telefone e e-mail

Para fazer um input que vai conter e-mails basta declarar um input do tipo email:

<input type=”email”>

Vai fazer uma validação que confere se o que esta escrito dentro da caixa realmente é um e-mail

---

Para fazer um input que vai conter numeros de telefone basta declarar um input do tipo tel:

<input type=”tel”>

Vai ter uma compatibilidade extra com celulares já que ele vai conseguir puxar sua lista de contatos

---

Input de marcação

Para fazer um input que marca e desmarca, ou seja, um botão de marcação basta declarar um input do tipo checkbox:

<input type=”checkbox”>

O tipo checkbox te deixa marcar quantas caixas quiser, ou seja, deixa marcar todas as caixas ao mesmo tempo e não apenas uma

---

Para fazer um input que marca apenas uma opção, ou seja, um botão de marcação único basta declarar um input do tipo radio:

<input type=”radio”>

O tipo radio te deixa marcar apenas uma opção, ou seja, caso for marcar uma opção e depois a outra, a primeira opção vai ser desmarcada e a atual sera marcada, vice e versa

Para fazer uma ligação entre os inputs do tipo radio que estão no mesmo grupo é necessario os inputs terem o mesmo nome (name=” “), assim sendo, quando um for marcado o outro sera desmarcado, permitindo apenas uma escolha

---

A diferença entre o checkbox e o radio é que no checkbox é permitido marcar quantas opções quiser, enquanto que no radio só pode marcar uma opção por vez, alem disso o checkbox é um botão quadrado e o radio é um botão redondo e os do tipo radio precisam ter o mesmo name=” “

<input type=”checkbox”>

<input type=”radio”>

Caso queira que alguma das opções do checkbox ou do radio já apareça marcada, basta declarar o atributo checked no input:

<input type=”checkbox” checked>

<input type=”radio” checked>

Lembrando que nos inputs do tipo radio não é permitido botar checked em mais de um, já que o radio serve apenas para uma única opção ser marcada

---

Controles Adicionais

Para fazer um input que coloca cor basta fazer um input do tipo color:

<input type=”color”>

Ele vai te dar a opção de escolher qualquer cor, para fazer o input já começar com uma cor basta botar o atributo value no input com algum codigo de cor hexadecimal:

<input type=”color” value=”#00ff00”>

Para fazer um input de medida / nivel que vai de 0 a 100 basta declarar um input do tipo range:

<input type=”range”>

Da pra personalizar o nivel dele (0 a 100), basta botar os atributos min=” “ e max=” “, ou caso queira começar com um valor basta botar o atributo value=” “

Para fazer um input que realize upload de qualquer arquivo basta declarar o input do tipo file:

<input type=”file”>

Ele da a possibilidade de upar qualquer arquivo, foto, pdf, pastas etc.

Quando for usar esse tipo de controle (input – nesse caso), controle de arquivos, não é permitido usar o metodo GET no <form> é necessario o metodo POST já que um arquivo usa muito mais do que 3k bytes.

Seleção

Para fazer uma lista de seleção única, ou seja, uma lista em que apenas um item é selecionado / escolhido basta declarar a tag <select> e dentro de select colocar tags do tipo <option>, o option vai ser o item da lista:
<select>

<option value=” “> </option>

</select>

É permitido botar quantos itens quiser, basta declarar o option e dentro escolher o nome do item ou o próprio item dependendo

Para fazer com que um item já começe selecionado basta declara o atributo selected dentro do <option>

Um maçete é fazer um option escrito ‘escolha’ e dexar ele como selected

---

Para fazer o agrupamento dos itens que estão na seleção, e dividir os itens em grupos especificos basta envelopar os options usando a tag <optgroup> e dar o titulo do grupo usando o atributo label=” “:

<optgroup label=” “>

</optgroup>

Ex: caso queira selecionar estados pela região, basta dividir os itens com o optgroup

---

Lista de dados

Para fazer uma lista de dados, onde é possível escolher ou escrever um item, basta fazer um <input> normal do tipo text com o atributo list=” “ e escolher um nome pra ele (list) e embaixo declarar a tag <datalist> com um id tendo o mesmo valor (nome) de list, fazendo assim a ligação entre os dois (input e datalist), para colocar as opções basta declarar os options dentro dele (datalist):

<input type=”text” list=” “>

<datalist id=” “>

<option value=” “> </option>

</datalist>

Vai fazer com que alem de ter a opção dos itens, ainda tenha como escrever um item

Tem a opção de tirar o value=” “ do option pra tirar o codigo e deixar só o option normal

---

Area de texto

Para fazer uma area de texto basta declarar a tag <textarea>, nesse bloco vai ser permitido digitar quantas letras / palavras quiser, basta declarar o numero de colunas (letras) (cols=” “) e o numero de linhas (rows=” “) que vão aparecer na tela (visivel), não existe numero maximo de linhas, é infinito, pode digitar o quanto quiser:

<textarea cols=” “ rows=” “> </textarea>

Tambem é possível alterar o tamanho da textarea manualmente

Lembrando que pra usar o text area o <form> precisa ter o metodo POST, já que o post não tem limite de bytes igual o GET

Agrupamento de campos

Para agrupar campos basta envelopar esses campos com a tag fieldset (grupo de campos):

<fieldset> </fieldset>

Essa tag faz com que os campos do formulario fiquem com uma “caixa” em volta, como se fosse um container, é bom pra estilizar e fazer um bloco de cadastro, login etc.

Com o fieldset vem a possibilidade de utilizar a tag chamada legend:

<legend> </legend>

Essa tag bota uma legenda no fieldset, faz com que o usuario possa identificar a função daquele bloco, se é um bloco pra dados pessoais, login, cadastro etc

---

Lembrando que o <fieldset> e o <legend> não servem apenas para formularios, e tambem que da pra personalizar os dois com css e deixar mais agradavel

---

Saida (output)

Para fazer uma tag de saida (output) basta declarar a tag <output>:

<output> </output>

ela vai mostrar o que esta no meio da tag, ou seja, ela vai mostrar coisas na tela, pode receber alguns atributos iguais ao input, incluindo o id e name

---

O output tem a capacidade de mostrar na tela, ou seja, ele não envia nada pra nenhum lugar, apenas mostra na tela

---

o output pode ser usado com o atributo oniput que vai dentro do input, e tem a finalidade de mostrar algo automatico na tela, como por exemplo o resultado de uma soma

Fazendo um sistema de soma:

```
<p>
    <label for="n1">Numero 1:</label>
    <input type="number" name="t-n1" id="n1" min="0" max="50"
    required oninput="soma.innerHTML = Number(n1.value) + Number
    (n2.value)">
</p>

<p>
    <label for="n2">Numero 2:</label>
    <input type="number" name="t-n2" id="n2" min="0" max="50"
    required oninput="soma.innerHTML = Number(n1.value) + Number
    (n2.value)">
</p>

<p>
    <label for="soma">Soma: </label>
    <output name="t-soma" id="soma">0</output>
</p>
```

Mostrando os valores do range:

```
<p>
    <label for="num">Numero: </label>
    <input type="range" name="t-num" id="num" min="0" max="10"
    value="5" oninput="val.innerHTML = Number(num.value)">
    <output id="val">5</output>
</p>
```

Calculando a idade:

```
<p>
    <label for="ano">Em que ano você nasceu? </label>
    <input type="number" name="t-ano" id="ano" min="1900"
    max="2022" oninput="calcIdade()">
</p>

<p>
    <label for="idade">Sua idade é: </label>
    <output id="idade">0</output>
</p>

<p>
    <input type="submit" value="Enviar">
    <input type="reset" value="Limpar">
</p>

<script>
    function calcIdade() {
        let atual = new Date().getFullYear()
        idade.innerHTML = Number(atual) - Number(ano.value)
    }
</script>
```
