

Formulários

10

- ✓ Atributos para controle de formulários;
- ✓ Atributos para validar formulários;
- ✓ Atributo type (elemento input).

10.1.Introdução

Em HTML, um formulário é uma seção de documento que abrange texto, marcação, controles e rótulos. A função de um formulário é recolher dados recolhidos pelo usuário e enviá-los para serem processados no servidor. Os dados são inseridos pelo usuário nos formulários por meio de modificações nos controles (digitando um texto, selecionando um item em um menu, etc.) e depois processados por scripts no servidor.

Como, ao preencher um formulário, um usuário pode inserir não só textos simples, mas também scripts inteiros, isso pode gerar um problema de segurança, com a entrada de scripts maliciosos que prejudicam o site e seus usuários. Por isso, ao desenvolver um formulário e o script para o seu processamento, o mecanismo de validação é importante. A validação é o que permite verificar os dados digitados por um usuário nos controles de um formulário.

Até a HTML5, a validação era feita com JavaScript, mas podia ser facilmente anulada. Na HTML5, a validação é simples e eficiente, reduzindo a necessidade de desenvolvimento de scripts. Há novos atributos nos controles que permitem definir o tipo de dado a ser entrado e disparam mecanismos nativos de validação e mensagens de erro.

Nesta leitura, você verá as funcionalidades para formulários na HTML5, implementadas por meio de atributos usados nos controles dos formulários.

10.2.Atributos para controle de formulários

As funcionalidades para formulários na HTML5 estão relacionadas a novos atributos usados nos controles. Como veremos, eles servem para validação e outras funcionalidades que antes eram implementadas com o uso de scripts.

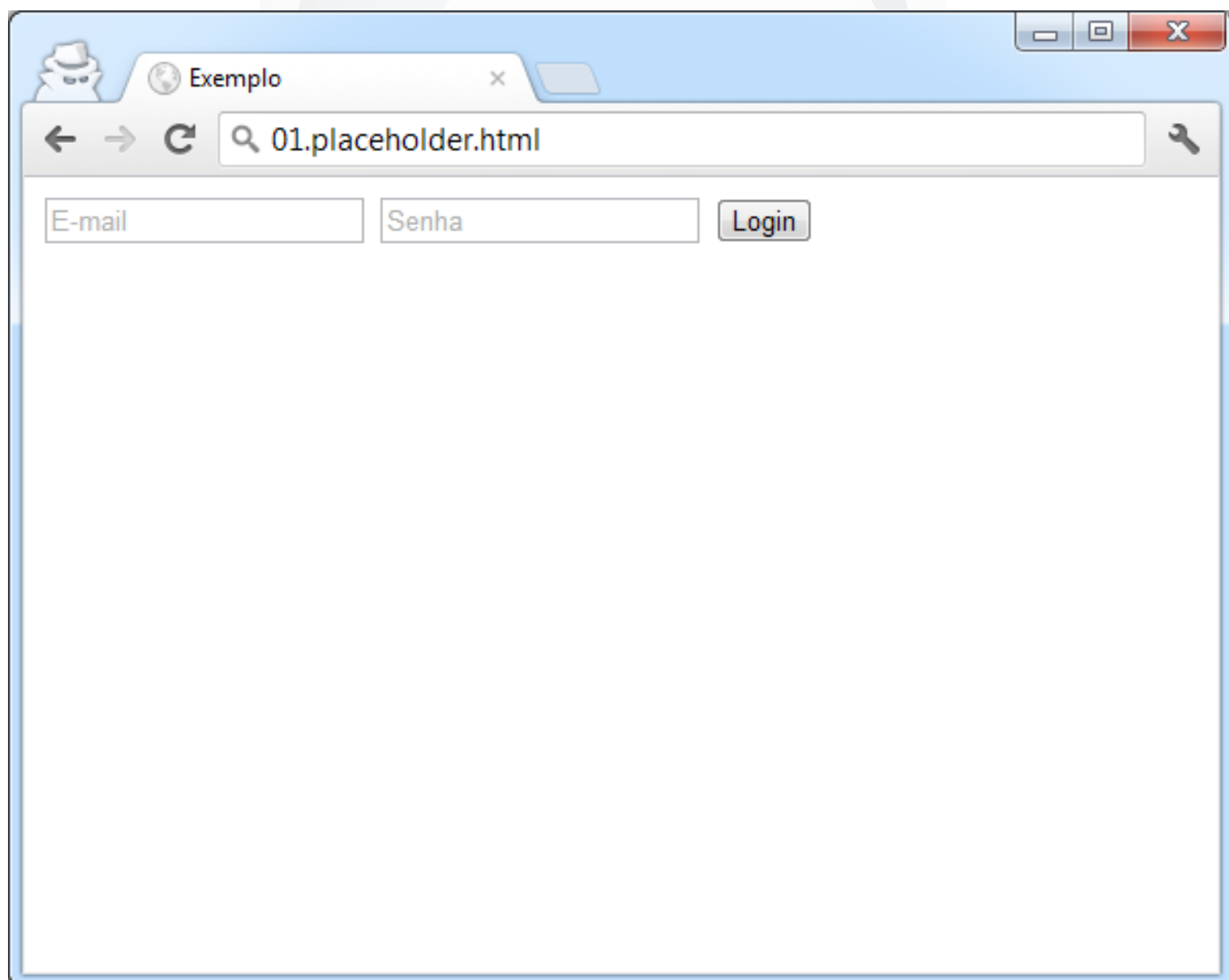
10.2.1.placeholder

Sua função é definir uma dica de preenchimento de um campo, composta por uma palavra ou uma frase curta. Deve ser usado exclusivamente com os elementos **input** e **textarea**.

A seguir, um exemplo ilustrando o uso de **placeholder**:

```
1 <!doctype html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>Exemplo</title>
6 </head>
7 <body>
8
9     <form>
10         <input type="text" placeholder="E-mail">
11         <input type="password" placeholder="Senha">
12         <button type="submit" class="btn">Login</button>
13     </form>
14
15 </body>
16 </html>
```

Esse código produzirá o seguinte resultado:



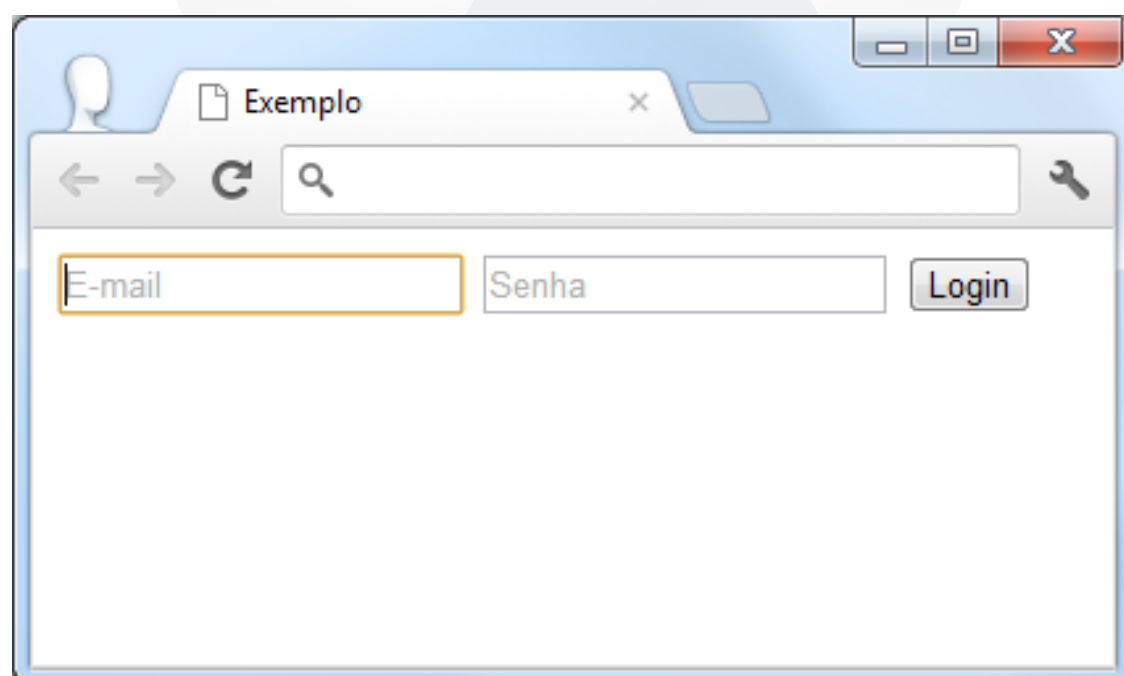
10.2.2 autofocus

A função deste atributo é definir qual o elemento deve ter o foco quando a página for carregada. Pode ser usado com os elementos **button**, **input**, **keygen**, **select** e **textarea**.

Veja o uso de **autofocus** no seguinte exemplo:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form>
10         <input autofocus type="text" placeholder="E-mail">
11         <input type="password" placeholder="Senha">
12         <button type="submit" class="btn">Login</button>
13     </form>
14
15 </body>
16 </html>
```

O resultado será o seguinte:



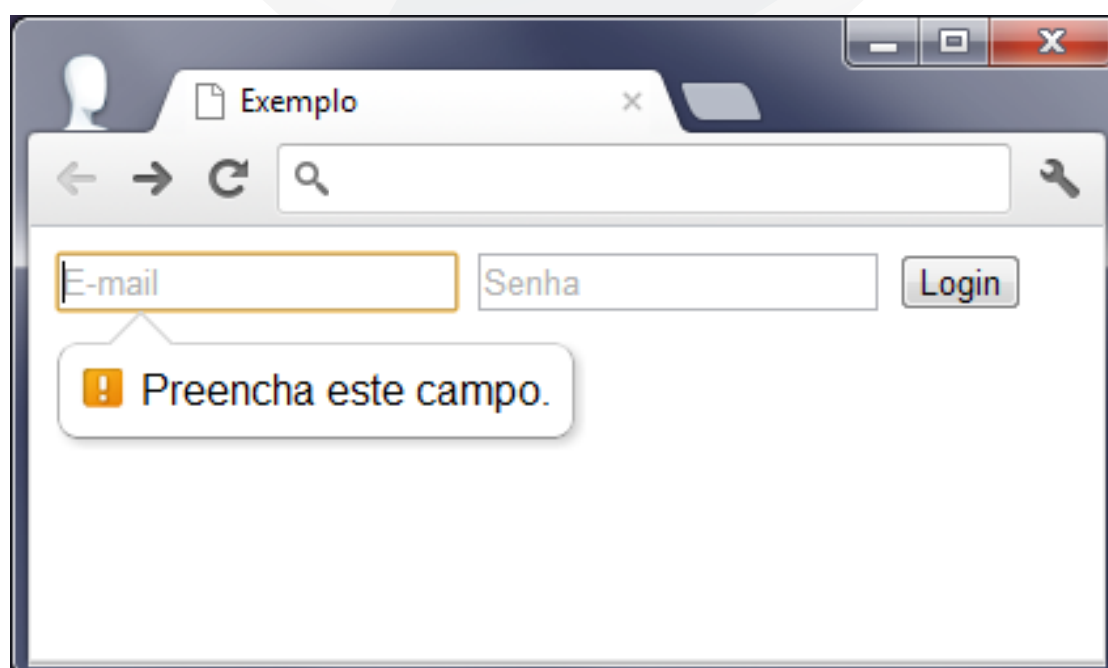
10.2.3.required

Sua função é marcar um controle de preenchimento obrigatório. Pode ser usado com **input**, **select** e **textarea** e é um atributo booleano, ou seja, basta sua presença na sintaxe, sem declaração de valor.

Veja, a seguir, um exemplo de **required**:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form>
10         <input required type="text" placeholder="E-mail">
11         <input required type="password" placeholder="Senha">
12         <button type="submit" class="btn">Login</button>
13     </form>
14
15 </body>
16 </html>
```

Dê uma olhada no resultado:



10.2.4.autocomplete

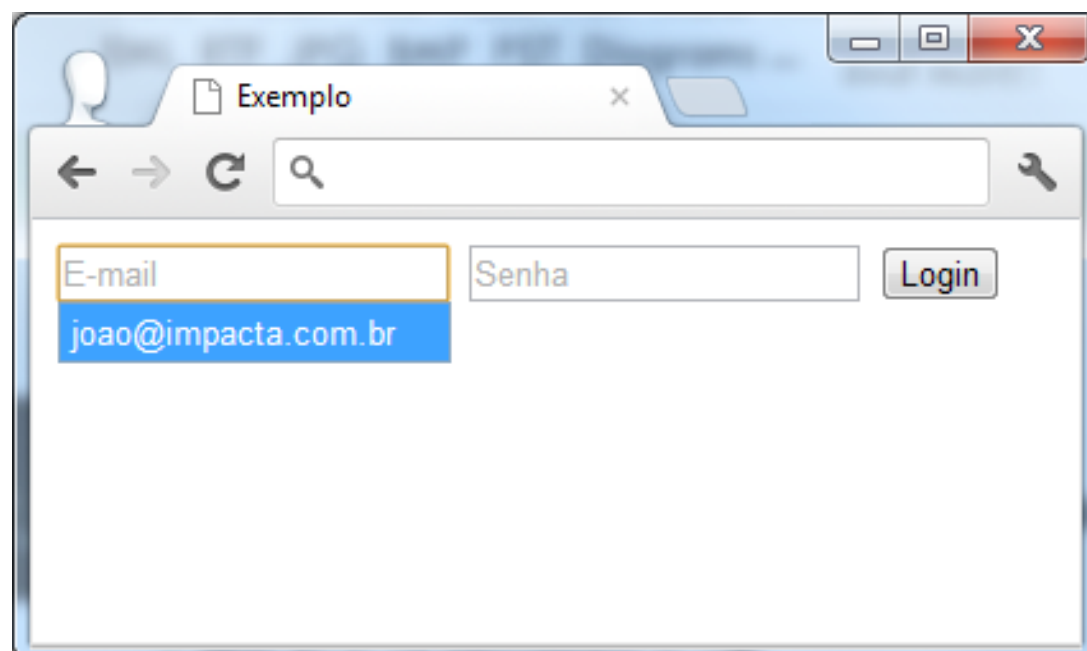
Este atributo oferece uma lista de opções para o preenchimento de um controle, de acordo com o que o usuário digitou nele anteriormente. É implementado de diferentes maneiras nos navegadores. Pode ser usado com os elementos **form** e **input** e admite os seguintes valores: **on** e **off**. Eles definem, respectivamente, se o navegador fará ou não sugestões de autopreenchimento.

Quando não é especificado para um controle específico, ele assume o comportamento de autopreenchimento do elemento **form** ao qual o controle pertence. Quando o **autocomplete** não é especificado para o **form**, o padrão de autopreenchimento é **on**.

Veja um exemplo de **autocomplete**:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form autocomplete="on">
10         <input type="text" name="email" placeholder="E-mail">
11         <input type="password" placeholder="Senha">
12         <button type="submit" class="btn">Login</button>
13     </form>
14
15 </body>
16 </html>
```

O código produzirá o seguinte resultado:

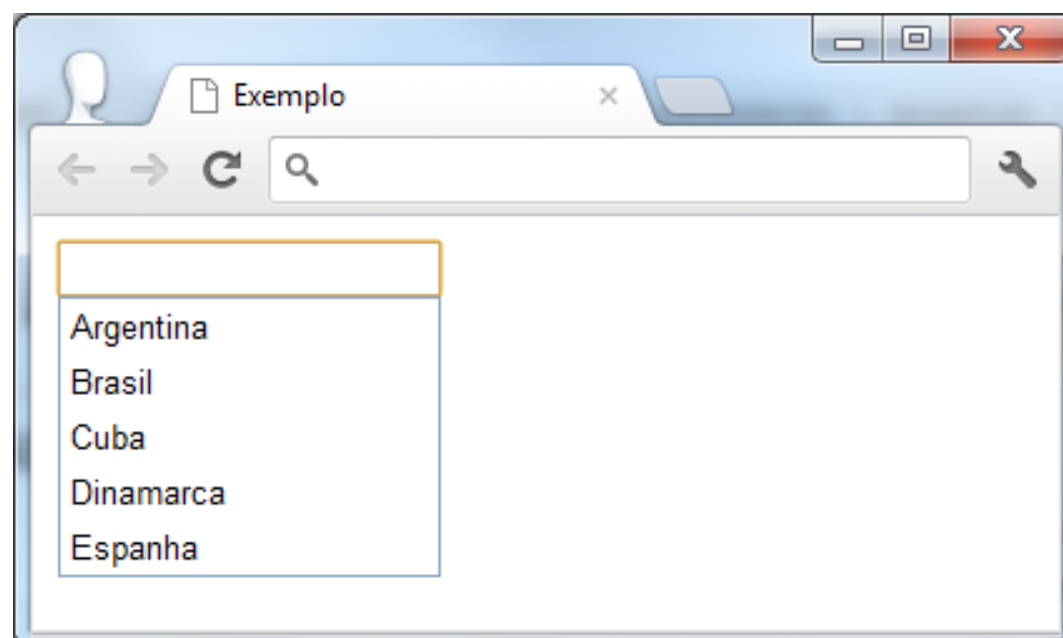


10.2.5.list

Define uma lista com opções de preenchimento para um campo de texto. Deve ter o mesmo valor que o id de um elemento **datalist** e só pode ser usado com o elemento **input**. Veja um exemplo a seguir:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form>
10
11          <input list="países" />
12
13          <datalist id="países">
14              <option value="Argentina">
15              <option value="Brasil">
16              <option value="Cuba">
17              <option value="Dinamarca">
18              <option value="Espanha">
19          </datalist>
20
21      </form>
22
23  </body>
24  </html>
```

O resultado será este:



10.2.6.max

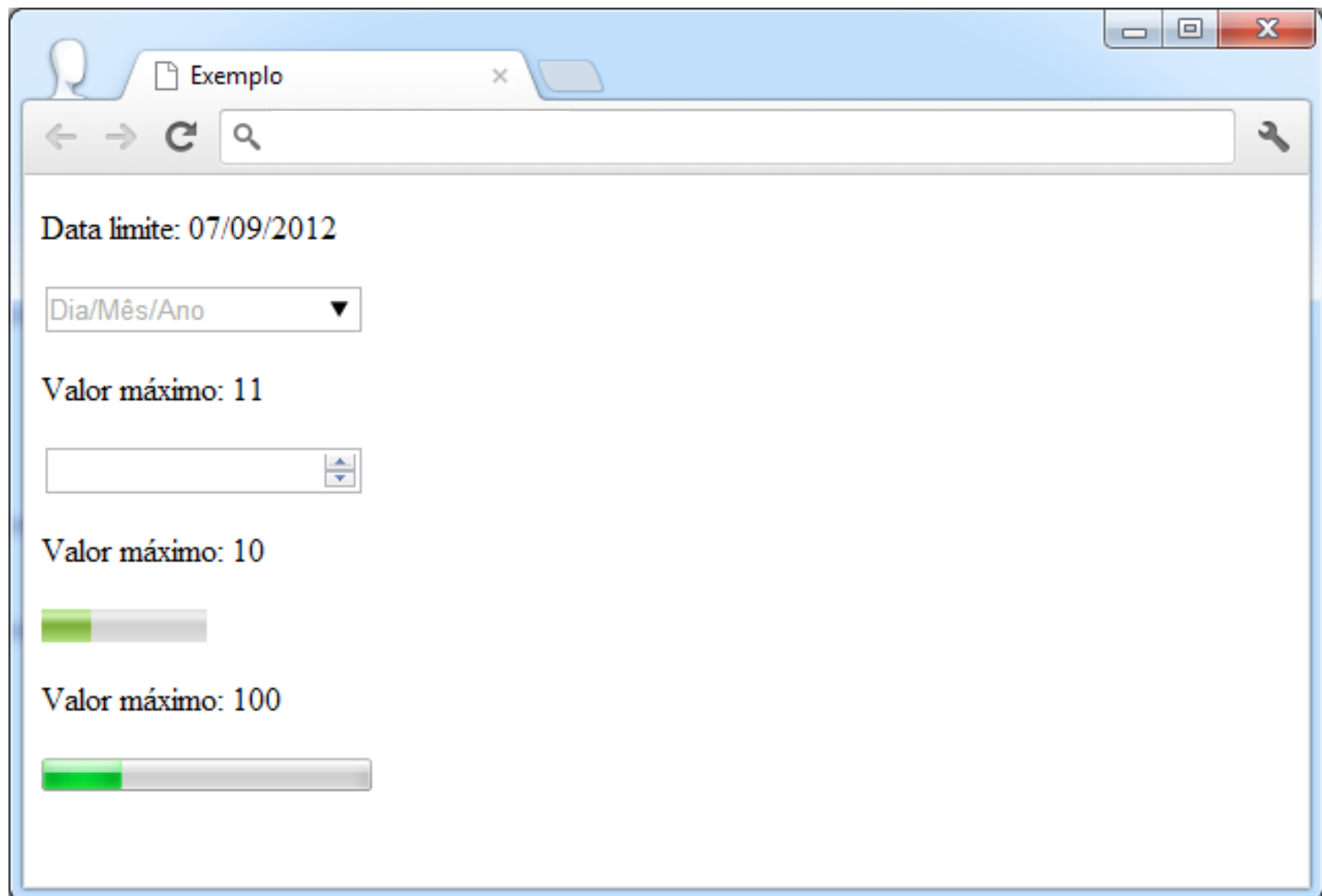
Este atributo tem como função definir um valor máximo para os diferentes elementos com os quais pode ser usado:

- Para o elemento **meter**, define o valor máximo da escala da medida marcada. Caso seja omitido, seu valor é 1;
- Para o elemento **progress**, define o valor máximo de uma barra de progresso, que mostra a progressão de uma tarefa;
- Para o elemento **input**, define o valor máximo que será permitido como entrada em um campo. Se for usado junto com o atributo **min**, definirá uma faixa de entrada de dados no campo.

Veja, a seguir, um exemplo de como utilizar o atributo **max**:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form>
10
11          <p>Data limite: 07/09/2012</p>
12          <input type="date" max="2012-09-07"><br/>
13
14          <p>Valor máximo: 11</p>
15          <input type="number" max="11" /><br/>
16
17          <p>Valor máximo: 10</p>
18          <meter value="3" max="10">3 de 10</meter><br/>
19
20          <p>Valor máximo: 100</p>
21          <progress value="25" max="100"></progress>
22
23      </form>
24
25  </body>
26  </html>
```


E agora observe o resultado:



The screenshot shows a web browser window with a single tab titled 'Exemplo'. The address bar is empty. The page content includes three form elements:

- A text label 'Data limite: 07/09/2012' followed by a date input field with a dropdown arrow and the placeholder text 'Dia/Mês/Ano'.
- A text label 'Valor máximo: 11' followed by a numeric input field with a small up/down arrow on the right.
- A text label 'Valor máximo: 10' followed by a horizontal progress bar (meter) that is approximately 20% filled with green.
- A text label 'Valor máximo: 100' followed by a horizontal progress bar (meter) that is approximately 10% filled with green.

10.2.7.min

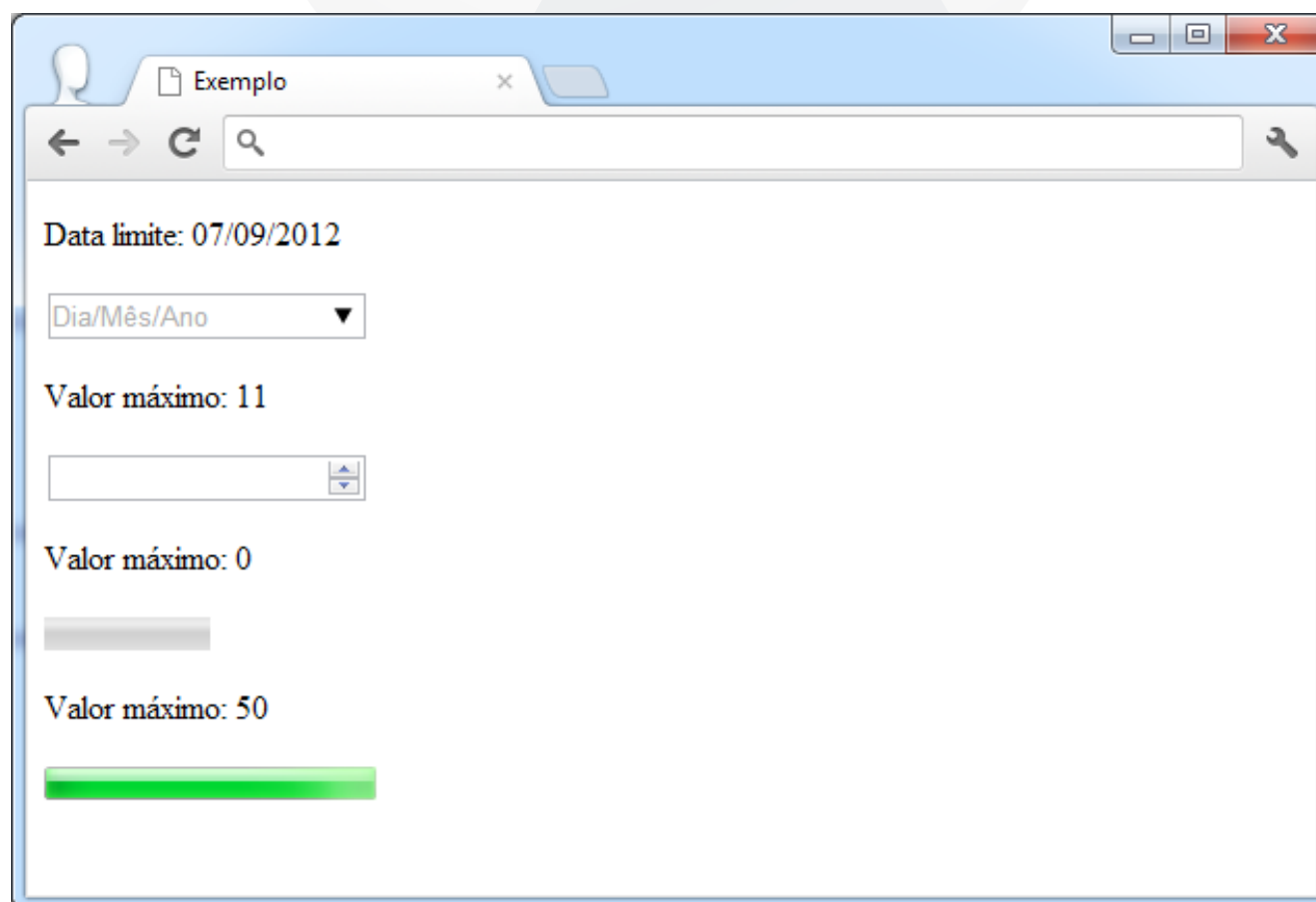
Ao contrário de **max**, serve para definir um valor mínimo. Além disso, só pode ser usado com **meter** e **input**:

- Para **meter**, define o valor mínimo da escala da medida marcada. Caso seja omitido, seu valor é 0;
- Para **input**, define o valor mínimo que será permitido como entrada em um campo.

Veja, no exemplo, como utilizar **min**:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form>
10
11          <p>Data limite: 07/09/2012</p>
12          <input type="date" min="2012-09-07"><br/>
13
14          <p>Valor minimo: 11</p>
15          <input type="number" min="11" /><br/>
16
17          <p>Valor minimo: 0</p>
18          <meter value="1" min="0">3 de 10</meter><br/>
19
20          <p>Valor minimo: 50</p>
21          <progress value="50" min="50"></progress>
22
23      </form>
24
25  </body>
26  </html>
```

O resultado você vê a seguir:



10.2.8.form

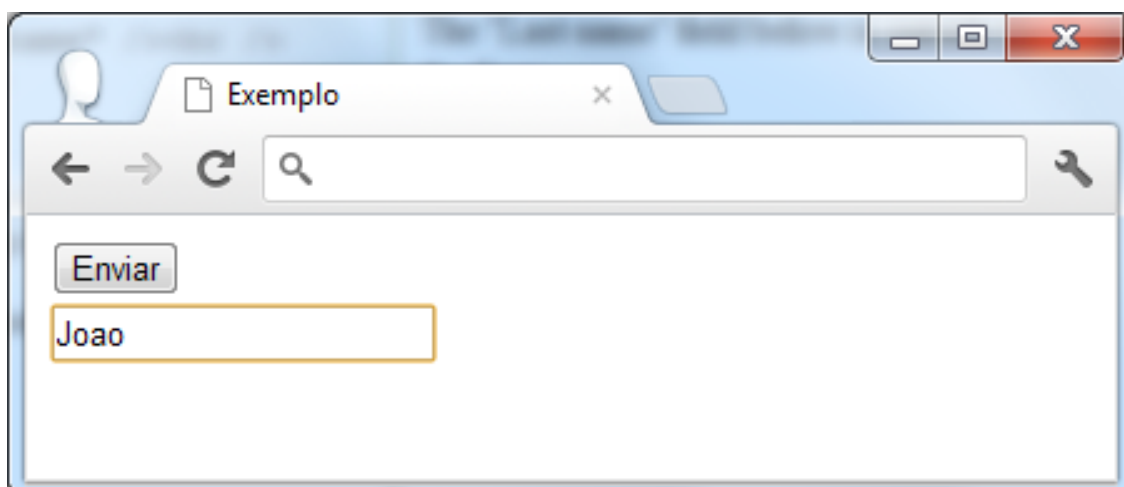
Associa um controle de formulário a um elemento formulário. Permite, inclusive, associar a um formulário controles que estão fora dele ou dentro de outros formulários, o que não era possível nas versões anteriores da HTML. O atributo **form** tem o mesmo valor do atributo **id** do formulário ao qual estiver associado.

Podemos usar **form** com os seguintes elementos: **input**, **output**, **select**, **object**, **button**, **textarea**, **meter**, **label**, **progress**, **keygen** e **fieldset**.

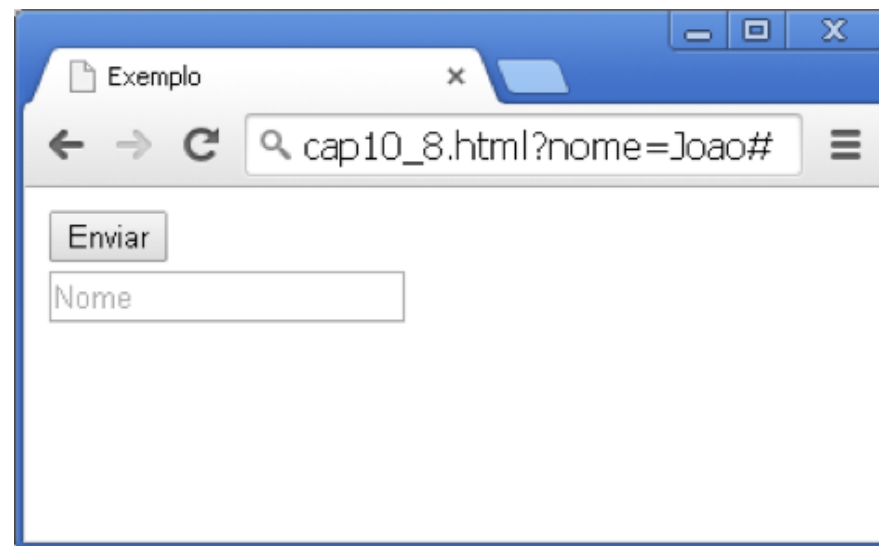
Veja como usar o atributo no exemplo a seguir:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form id="formulario" action="#">
10         <input type="submit" value="Enviar">
11     </form>
12
13     <input type="text" name="nome" placeholder="Nome" form="formulario">
14
15 </body>
16 </html>
```

E este será o resultado:



Observe, na imagem a seguir, que, mesmo o elemento estando fora da tag do formulário, o seu valor é enviado:



10.2.9.formaction

Este atributo define uma URL que será o endereço para o qual o formulário será enviado e onde será processado. Pode ser usado com os elementos **input** e **button**. Veja como usá-lo no exemplo a seguir:

```
1 <!doctype html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>Exemplo</title>
6 </head>
7 <body>
8
9     <form action="login_aluno.html">
10         <input type="text" placeholder="E-mail">
11         <input type="password" placeholder="Senha">
12         <input type="submit" value="Login Aluno">
13         <input type="submit" value="Login Instrutor" formaction="login_instrutor.html">
14     </form>
15
16 </body>
17 </html>
```

10.2.10.formenctype

Este atributo define qual é o tipo de conteúdo enviado pelo formulário e pode ser usado com **input** e **button**. Veja como usá-lo no exemplo a seguir:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#" method="post">
10         <input type="text" placeholder="Nome">
11         <input type="submit" value="Envio Padrão">
12         <input type="submit" value="Envio Texto Plano" formenctype="text/plain">
13     </form>
14
15 </body>
16 </html>
```

10.2.11.formmethod

A função deste atributo é definir qual método será usado para enviar os dados do formulário. Pode ser usado com **input** e **button**. Veja como no seguinte exemplo:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#" method="post">
10         <input type="text" name="nome" placeholder="Nome">
11         <input type="submit" value="POST">
12         <input type="submit" value="GET" formmethod="get">
13     </form>
14
15 </body>
16 </html>
```

10.2.12.formtarget

Permite determinar qual será o destino do documento aberto após o formulário ser enviado. Pode ser usado com **input** e **button** e admite os seguintes valores: **_blank**, **_top**, **_self**, **_parent** ou outro nome na janela atual.

O exemplo a seguir mostra como usar este atributo:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input type="text" name="nome" placeholder="Nome">
11         <input type="submit" value="Enviar Padrão">
12         <input type="submit" value="Enviar Para Nova Janela" formtarget="_blank">
13     </form>
14
15 </body>
16 </html>
```

10.3.Atributos para validar formulários

Como dissemos, uma característica importante da HTML5 é introduzir atributos que facilitam a validação de formulários, automatizando o processo todo ou pelo menos parte dele. Vejamos os atributos que são utilizados nessa tarefa.

10.3.1.formnovalidate

Com este atributo, você desabilita a validação de dados do formulário. Pode ser usado com os elementos **button** e **input** e é um atributo booleano. Assim, basta inserir o atributo em um elemento para que o formulário não tenha seus dados validados. Veja no exemplo a seguir:

```
1 <!doctype html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>Exemplo</title>
6 </head>
7 <body>
8
9     <form action="#">
10         <input required type="text" name="nome" placeholder="Nome">
11         <input type="submit" value="Enviar com Validação">
12         <input type="submit" value="Enviar sem Validação" formnovalidate="formnovalidate">
13     </form>
14
15 </body>
16 </html>
```

10.3.2.novalidate

Possui o mesmo efeito que o **formnovalidate**. A diferença é que deve ser usado exclusivamente com um elemento **form**. Veja um exemplo com esse atributo:

```
1 <!doctype html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>Exemplo</title>
6 </head>
7 <body>
8
9     <form action="#" novalidate="novalidate">
10         <input required type="email" name="nome" placeholder="Nome">
11         <input type="submit" value="Enviar">
12     </form>
13
14 </body>
15 </html>
```

10.3.3.pattern

Com **pattern**, você pode definir expressões regulares para validação de formulários, sem precisar de JavaScript.

Veja como utilizá-lo no exemplo a seguir:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <label for="CPF_1">Digite o CPF com máscara:</label>
11         <input
12             id="CPF_1"
13             type="text"
14             pattern="\d{3}\.\d{3}\.\d{3}-\d{2}"
15             placeholder="____.____.____-__">
16         <br/>
17         <label for="CPF_2">Digite o CPF sem máscara:</label>
18         <input
19             id="CPF_2"
20             type="text"
21             pattern="[0-9]{11}"
22             placeholder="Apenas números">
23         <br/>
24         <input type="submit" value="Enviar">
25     </form>
26
27 </body>
28 </html>
```


10.4.Atributo type (elemento input)

Na HTML5, o atributo **type** do elemento **input** possui diversos novos valores que definem o tipo de dado esperado pelo controle. Na HTML4, o **type** tinha dez valores diferentes (**text**, **password**, **checkbox**, **radio**, **submit**, **reset**, **file**, **hidden**, **image** e **button**). Com a HTML5, esses valores foram mantidos e foram acrescentados outros, que você verá adiante.

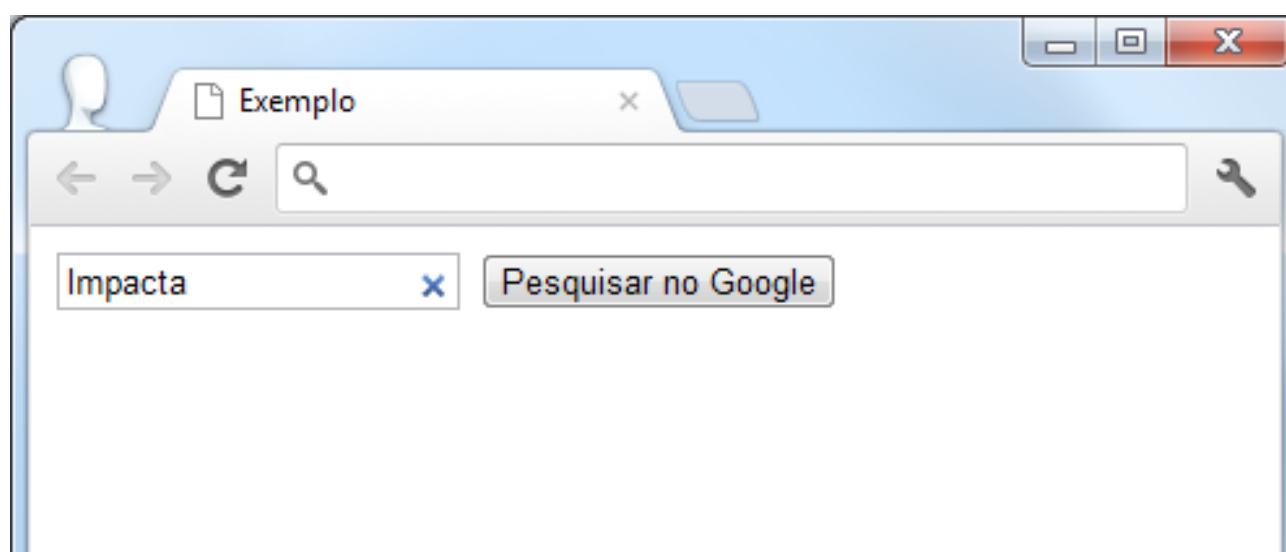
10.4.1.search

Define um controle **input** para busca, diferenciando-o dos outros controles alterando sua estilização e comportamento. Cada navegador possui um modo de estilização e comportamento próprios.

Veja como funciona o atributo **type** com valor **search**:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="https://www.google.com.br/search">
10         <input type="search" name="q" value="Impacta">
11         <input type="submit" value="Pesquisar no Google">
12     </form>
13
14 </body>
15 </html>
```

O resultado é mostrado a seguir:

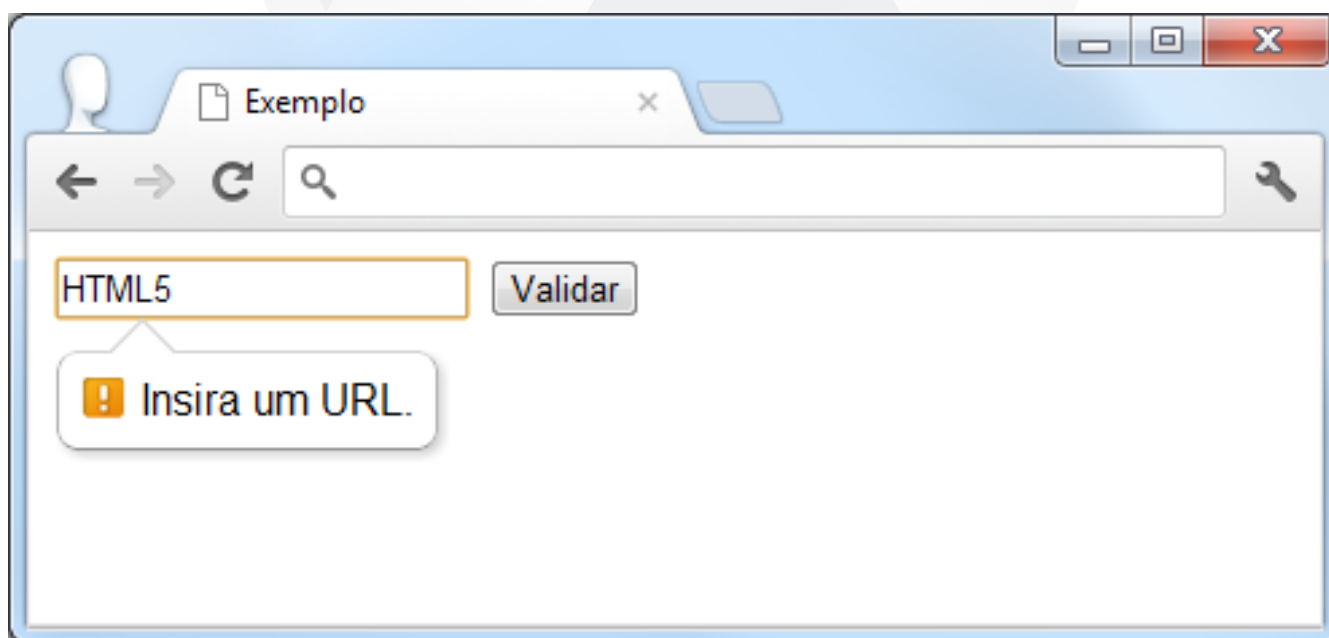


10.4.2.url

Garante que um controle **input** receba uma URL como valor. Ele fornece ao usuário, entre outras coisas, uma indicação visual de que o valor digitado é uma URL válida. Veja como funciona no exemplo a seguir:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input type="url" name="URL" placeholder="Digite uma URL">
11         <input type="submit" value="Validar">
12     </form>
13
14 </body>
15 </html>
```

Agora veja o resultado:



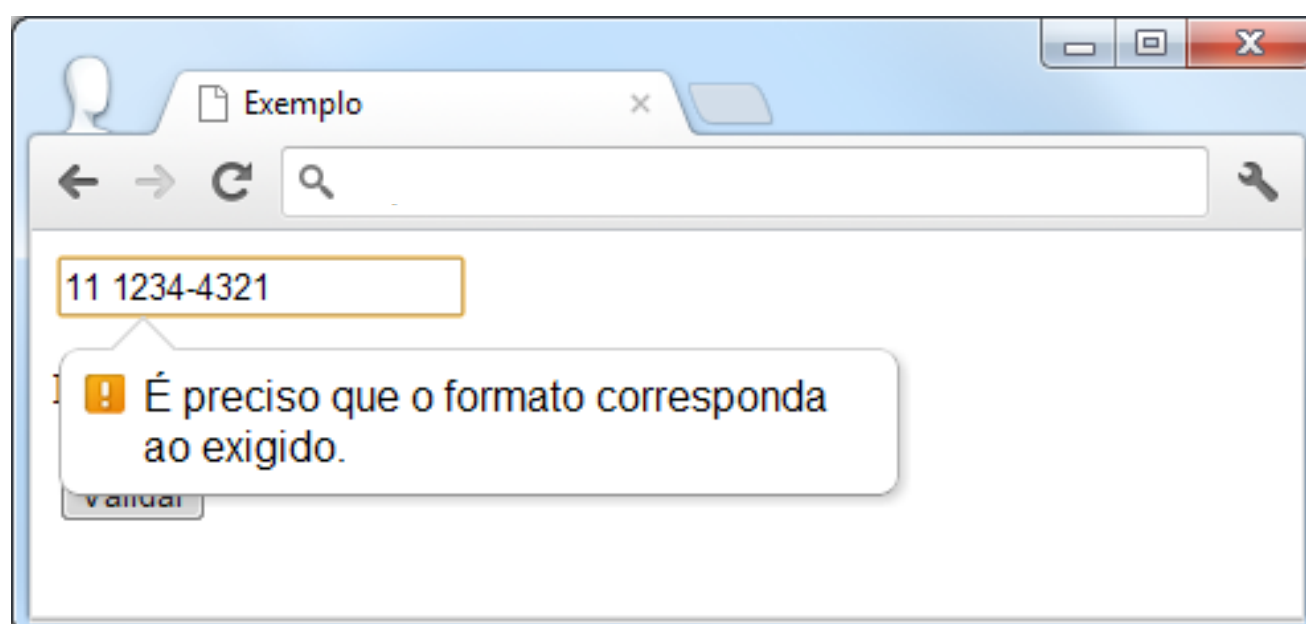
10.4.3.tel

Define um controle **input** destinado a receber um número de telefone. Ele não restringe o controle a uma sintaxe particular porque os formatos de números de telefone podem variar muito. Portanto, ele apenas define o destino do controle.

Para validar um controle desse tipo, você tem duas opções: Usar o atributo **pattern** ou o método **setCustomValidity()**, que é disponibilizado como um mecanismo JavaScript nativo do navegador. O exemplo a seguir mostra como validar esse controle utilizando o atributo **pattern**:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input
11             type="tel"
12             pattern="\([0-9]{2}\) [\s][0-9]{4}-[0-9]{4}"
13             placeholder="Telefone">
14         <p>Ex: (11) 1234-4321</p>
15         <input type="submit" value="Validar">
16     </form>
17
18 </body>
19 </html>
```

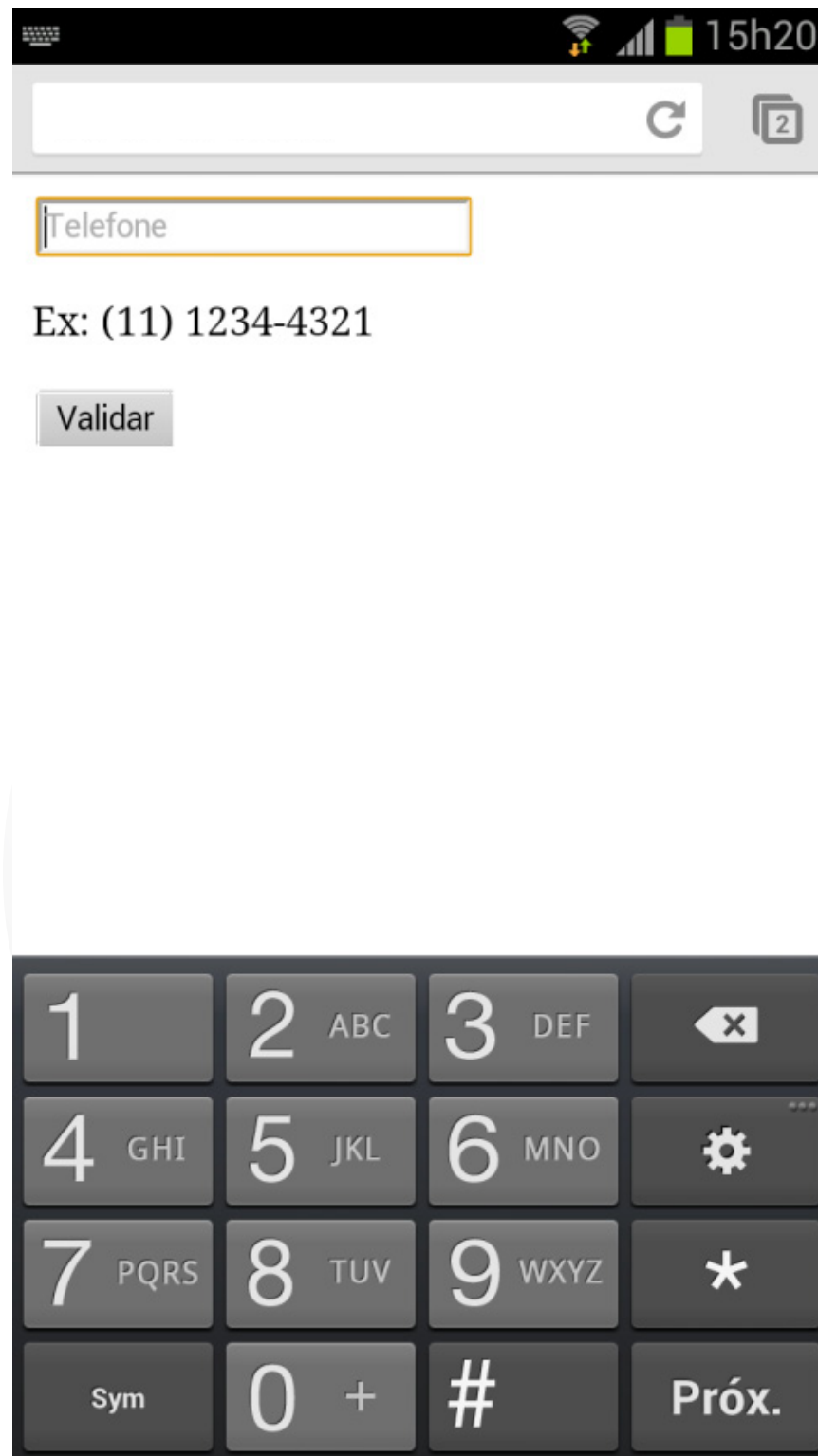
E o resultado você vê logo adiante:



HTML5 Fundamentos (online)

190

Ao acessar uma página com um input com o **type="tel"** de um dispositivo móvel, o teclado se adapta para que o usuário possa digitar mais facilmente o valor esperado. Veja:

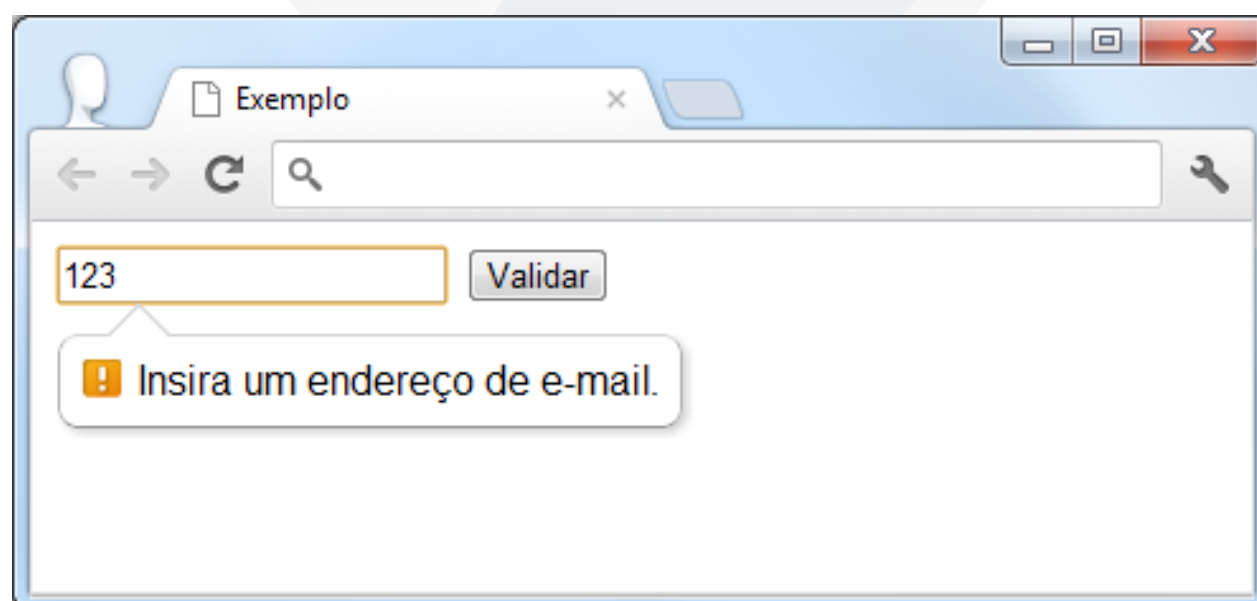


10.4.4.email


Com este valor, o controle **input** pode receber apenas um endereço de e-mail como valor. Ele, na verdade, não verifica se o e-mail digitado é válido, apenas se o formato digitado é válido. Veja como usá-lo no seguinte exemplo:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input
11             type="email"
12             placeholder="E-mail">
13         <input type="submit" value="Validar">
14     </form>
15
16 </body>
17 </html>
```

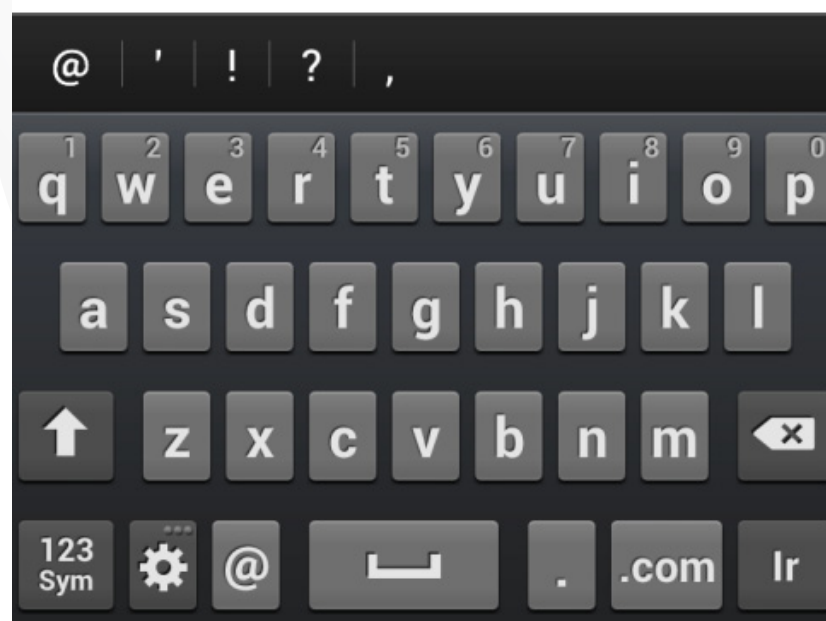
Agora veja o resultado:



Assim como o **type="tel"**, o **type="email"** também faz o teclado dos dispositivos móveis se adaptarem para facilitar a entrada de um endereço de e-mail. Veja:



A screenshot of a mobile browser interface. At the top, there's a status bar with icons for Wi-Fi, cellular signal, and battery, along with the time 15h31. Below the status bar is a browser address bar with a refresh icon and a tab indicator showing '2'. Underneath the address bar is a form with a text input field containing the placeholder text 'E-mail' and a button labeled 'Validar'.



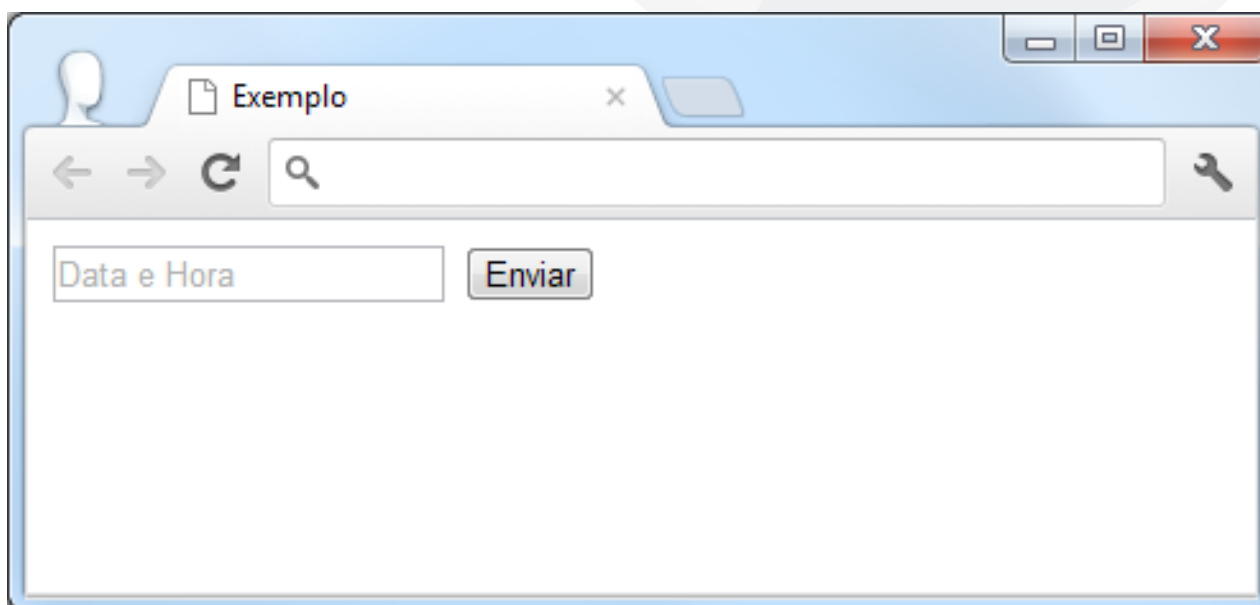
10.4.5.datetime

Um controle **input** com esse valor deverá receber uma string de data e hora UTC. O agente de usuário é quem define como será a representação do grupo data/hora, que dependerá do local e do usuário. É comum o agente de usuário usar um widget do tipo date picker para entradas no controle de data e um campo com slider de seta para entradas de hora.

Veja o uso de **datetime** a seguir:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input type="datetime" placeholder="Data e Hora">
11         <input type="submit" value="Enviar">
12     </form>
13
14 </body>
15 </html>
```

E o resultado é este:

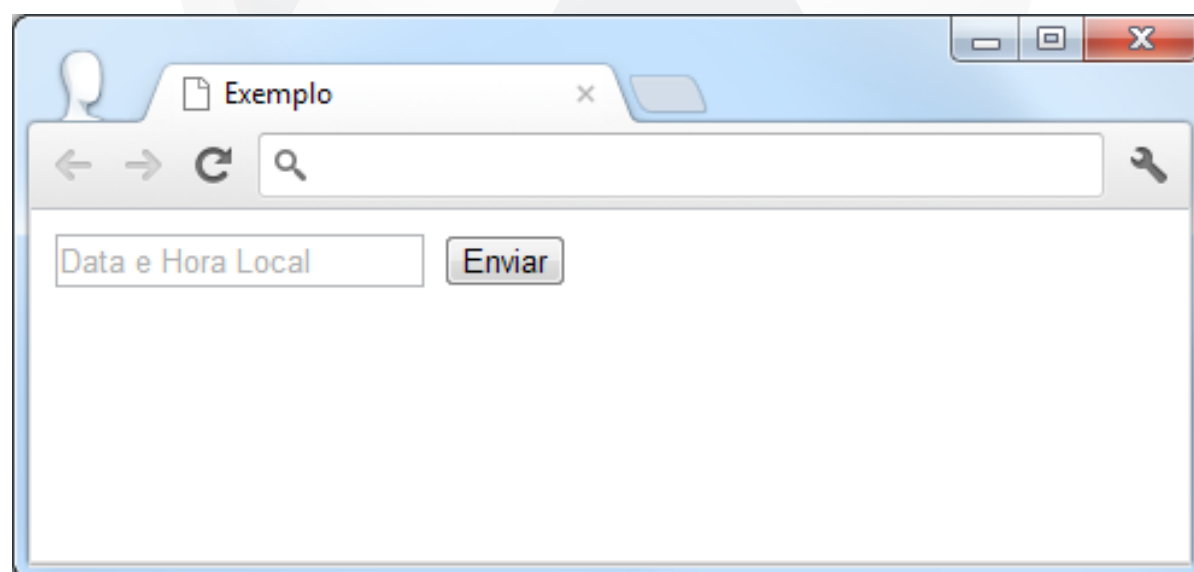


10.4.6.datetime-local

Diferencia-se do **datetime** por aceitar como valor uma string representando data e hora locais, e não UTC. Veja o exemplo a seguir:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input type="datetime-local" placeholder="Data e Hora Local">
11         <input type="submit" value="Enviar">
12     </form>
13
14 </body>
15 </html>
```

E, em seguida, o resultado:



10.4.7.date

Um controle **input** com esse valor deverá receber uma string de data. O agente de usuário é quem define como será a representação da data, que dependerá do local e do usuário. É comum o agente de usuário usar um widget date picker para entradas no controle de data. Veja o exemplo a seguir:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input type="date" placeholder="Nascimento">
11         <input type="submit" value="Enviar">
12     </form>
13
14 </body>
15 </html>
```

Com o seguinte resultado:

dom	seg	ter	qua	qui	sex	sáb
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	1	2	3	4	5	6

10.4.8.time

Um controle **input** com esse valor deverá receber uma string de hora. Geralmente, o agente de usuário usa um slider de seta para entradas no controle de data. Veja o exemplo a seguir:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input type="time" placeholder="Hora">
11         <input type="submit" value="Enviar">
12     </form>
13
14 </body>
15 </html>
```

10.4.9.month

Define para um controle **input** uma string de mês, cuja representação fica por conta do agente de usuário, de acordo com o local e o usuário. Geralmente, utiliza-se um widget date picker para a entrada no controle. Veja o exemplo a seguir:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input type="month" placeholder="Mês">
11         <input type="submit" value="Enviar">
12     </form>
13
14 </body>
15 </html>
```

10.4.10.week

Define um controle que deverá receber uma string representando uma semana do ano. Normalmente, o agente de usuário usa um widget date picker para a entrada. Veja o exemplo a seguir:

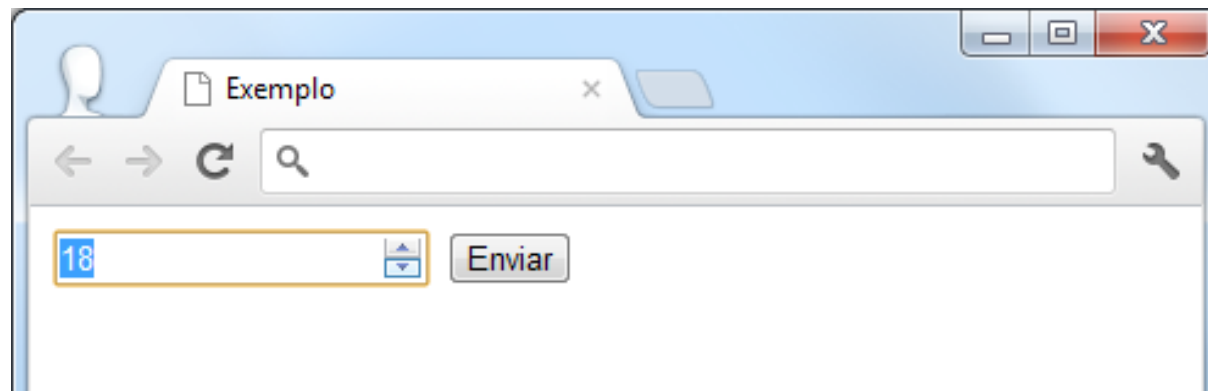
```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input type="week" placeholder="Semana do Ano">
11         <input type="submit" value="Enviar">
12     </form>
13
14 </body>
15 </html>
```

10.4.11.number

Controles com esse valor devem receber números. É comum que o agente de usuário use sliders de seta para esse tipo de entrada. Veja um exemplo:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input type="number" min="18" placeholder="Idade">
11         <input type="submit" value="Enviar">
12     </form>
13
14 </body>
15 </html>
```

O resultado é o seguinte:

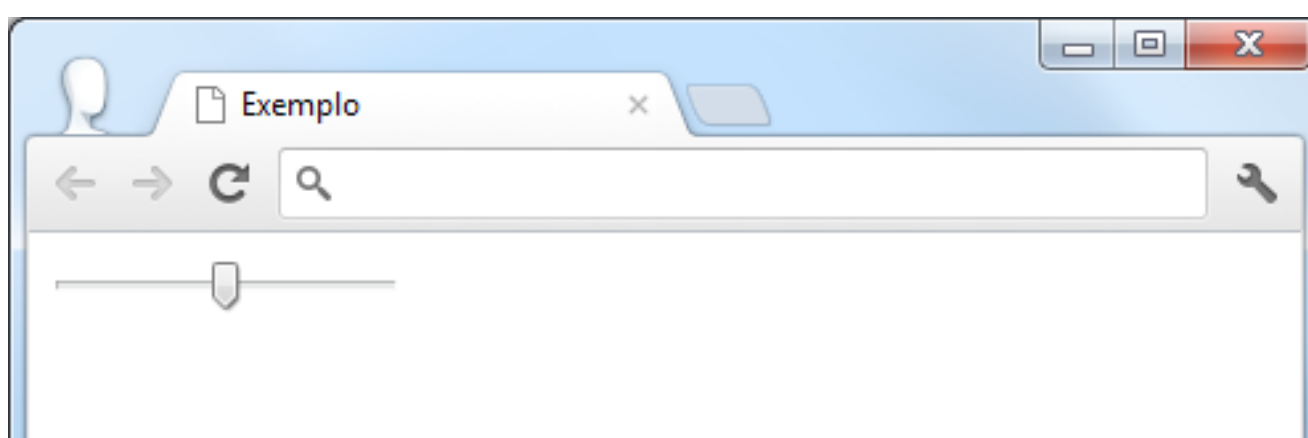


10.4.12.range

Em controles com **range**, a entrada deve ser um número situado dentro de um intervalo definido. É comum que essas entradas tenham um slider deslizante. Veja um exemplo com **range**:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <input type="range" min="0" max="100" value="50">
11     </form>
12
13 </body>
14 </html>
```

E o resultado é o seguinte:

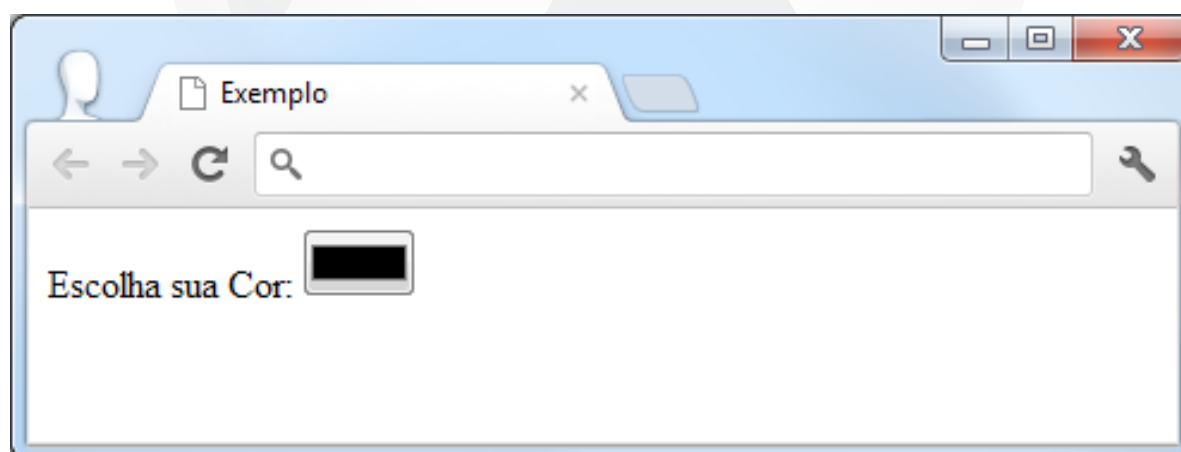


10.4.13.color

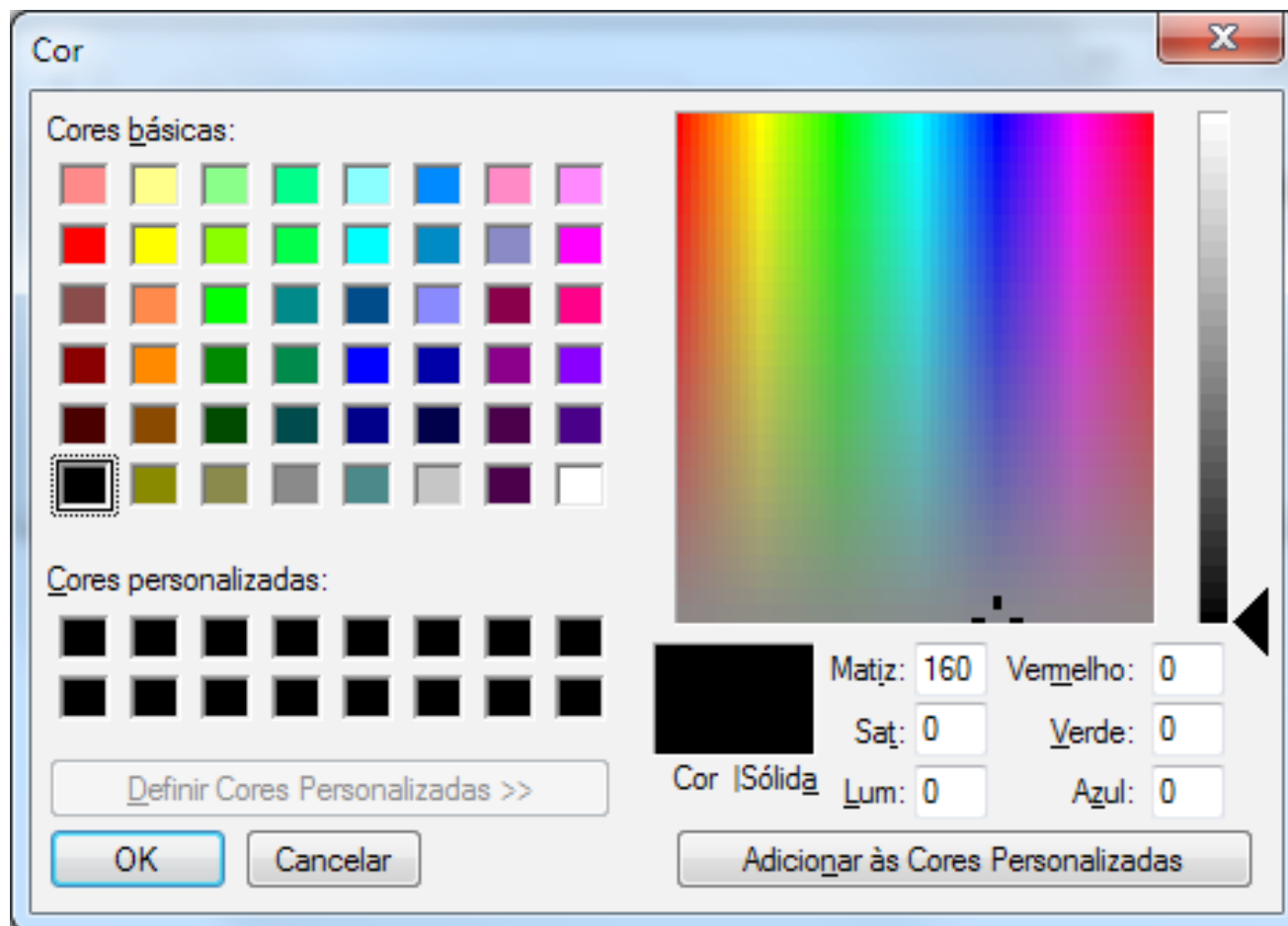
Define que um controle **input** receba um código sRGB de cor. Geralmente, utiliza-se um color pick para esse tipo de entrada. Veja um exemplo:

```
1  <!doctype html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9      <form action="#">
10         <label for="cor">Escolha sua Cor:</label>
11         <input type="color" id="cor" name="cor_escolhida">
12     </form>
13
14 </body>
15 </html>
```

Agora observe o resultado:



Ao clicar no input **type="color"**, a seguinte janela será aberta:



Após escolher uma cor, o input assume a cor escolhida:

