Eventos Javascript

Programação Microinformática

Paulo R. T. Cândido

Eventos

A eventos gerados pelo usuário (clicar em botão, por exemplo), ou decorrentes de outras atividades do sistema, são associados a funções javascript que realizam ações (tratam o evento). Funções contém código javascript que realizam uma (e de preferência apenas uma) tarefa, podendo retornar um resultado.

No código abaixo, foi definida função cujo nome é “função1” (sem parâmetros) e a tarefa que ela realiza é apresentar um texto (“Botão 1 pressionado”) em uma janela de alerta. A função é acionada quando o usuário pressiona o botão (tag button - evento click), conforme definido pelo atributo onclick. Esta formar de tratamento de eventos está em desuso, veremos outras formas nos próximos slides.

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title>Eventos 1</title>
6  </head>
7  <body>
8      <!-- Este forma de tratameno de eventos está caindo em desuso -->
9      <button id="b1" onclick="funcao1()">Botão 1</button>
10 </body>
11 </html>
12 <script type="text/javascript">
13     function funcao1() {
14         alert("Botão 1 pressionado");
15     }
16 </script>
```

Eventos – Propriedade do Objeto

Outra forma para tratar eventos é configurar a propriedade (“onclick”, neste exemplo) do objeto de interface (botão, neste exemplo), atribuindo a ela a função a ser disparada quando o evento ocorrer.

```
7  <body>
8      <button id="b1">Botão 1</button>
9  </body>
10 </html>
11 <script type="text/javascript">
12     b1.onclick = funcao1; // configurar propriedade onclick do objeto botão
13     function funcao1() {
14         alert("Botão 1 pressionado");
15     }
16 </script>
```

```
7  <body>
8      <button id="b1">Botão 1</button>
9  </body>
10 </html>
11 <script type="text/javascript">
12     // configurar propriedade onclick com função anônima
13     b1.onclick = function () {
14                     alert("Botão 1 pressionado");
15                 }
16 </script>
```

Eventos – addEventListener

Outra forma para tratar eventos é adicionar “ouvintes” (Listener) para o evento. Forma mais moderna, mas pode não ser suportada por todos os navegadores. Permite adicionar mais de um “ouvinte” para o mesmo evento e também remover “ouvintes” (removeEventListener).

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Eventos 2</title>
</head>
<body>
    <button id="b1">Botão 1</button>
</body>
</html>
<script type="text/javascript">
    // mais moderno, porém alguns navegadores podem não suportar
    b1.addEventListener("click",funcao1);
    function funcao1() {
        alert("Botão 1 pressionado");
    }
    // permite adicionar mais de uma função para tratar um mesmo evento
    b1.addEventListener("click",function(){alert("Outra função disparada");});
</script>
```

Eventos – Usar mesma função para tratar eventos em elementos diferentes

O código abaixo exemplifica a mesma função sendo acionada em eventos gerados em diferentes elementos da interface (neste exemplo: button, h1, input). Ao parâmetro “e” é atribuído automaticamente um objeto EVENT contendo os dados do evento ocorrido, inclusive dados sobre o elemento em que foi gerado.

```
 7  <body>
 8      <button id="b1">Botão 1</button>
 9      <h1 id="t1">Título</h1>
10      <input type="number" id="i1">
11  </body>
12  </html>
13  <script type="text/javascript">
14      b1.onclick = funcao1;
15      t1.onmouseover = funcao1;
16      i1.onchange = funcao1;
17      function funcao1(e) { // o parâmetro "e" é um objeto contendo os dados do evento
18          // a propriedade e.target corresponde ao elemento em que o evento foi gerado
19          alert( e.target.nodeName );          // nodeName = tipo tag
20          if ( e.target.nodeName=='INPUT' ) {
21              alert( e.target.value );         // value = valor digitado na caixa
22          } else {
23              alert( e.target.textContent );  // textContent = texto na  tag
24          }
25      }
26  </script>
```

Exercício 1

Identificar porque o código abaixo não está funcionando corretamente (ver comentários no código). Correção está no próximo slide, mas execute o código e analise antes de ver a solução.

```
<body>
    <input type="number" id="i1">
    <input type="number" id="i2">
    <button id="b1">Somar</button>
    <h3 id="resultado">Resultado aqui</h3>
</body>
</html>
<!-- Corrigir o código abaixo, ele não está somando corretamente os número
digitador pelo usuário nas caixas de texto. -->
<script type="text/javascript">
    b1.onclick = function () {
        resultado.textContent = i1.value + i2.value;
    }
</script>
```

Exercício 1 - Resolução

Os valores obtidos na caixas de texto (i1.value e i2.value) são strings e precisam ser convertidos para números para serem somados adequadamente. Obs.: soma de strings resultam em concatenação, por exemplo, “3” + “5” resulta em “35”, não sendo isso que se quer.

```
 7  <body>
 8      <input type="number" id="i1">
 9      <input type="number" id="i2">
10      <button id="b1">Somar</button>
11      <h3 id="resultado">Resultado aqui</h3>
12  </body>
13  </html>
14  <script type="text/javascript">
15      b1.onclick = function () {
16          resultado.textContent = Number(i1.value) + Number(i2.value);
17      }
18  </script>
```

Exercício 2

Reescrever os cinco exercícios constantes no documento “Introdução Javascript” usando elementos de interface com usuário para entrada de dados (input, button, etc.) e apresentando os resultados em elementos h ou p da página.