Eventos en Elementos HTML

En HTML, existen una amplia variedad de eventos que se pueden utilizar para capturar las acciones y el comportamiento del usuario en los elementos de la página web. Estos eventos permiten la interacción y la ejecución de código JavaScript en respuesta a las acciones del usuario.

A continuación, se detallan algunos de los eventos más comunes en los elementos HTML, junto con ejemplos de código que muestran cómo se utilizan:

Evento onclick

El evento `onclick` se desencadena cuando un elemento es clicado por el usuario. Este evento es ampliamente utilizado para realizar acciones en respuesta a los clics del usuario.

```
<button onclick="saludar()">Haz clic aquí</button>

<script>
  function saludar() {
    alert("¡Hola! Has hecho clic en el botón.");
  }
</script>
```

En este ejemplo, cuando el usuario hace clic en el botón, se ejecuta la función `saludar()`, que muestra un cuadro de diálogo emergente con el mensaje “¡Hola! Has hecho clic en el botón.”.

Evento onmouseover y onmouseout

Los eventos `onmouseover` y `onmouseout` se desencadenan cuando el cursor del mouse pasa por encima de un elemento o se retira de él, respectivamente. Estos eventos son útiles para realizar acciones cuando el usuario interactúa con un elemento mediante el movimiento del mouse.

```
<div onmouseover="cambiarColor('blue')" onmouseout="cambiarColor('red')">Pasa

<script>
  function cambiarColor(color) {
    var elemento = document.querySelector("div");
    elemento.style.backgroundColor = color;
  }
</script>
```

En este ejemplo, cuando el cursor del mouse pasa por encima del elemento `<div>`, se ejecuta la función `cambiarColor('blue')`, que cambia el color de fondo del elemento a azul. Cuando el cursor se retira del elemento, se ejecuta la función `cambiarColor('red')`, que restaura el color de fondo a rojo.

Evento onkeydown y onkeyup

Los eventos `onkeydown` y `onkeyup` se desencadenan cuando se presiona una tecla y se suelta, respectivamente. Estos eventos son útiles para capturar la entrada del teclado del usuario y realizar acciones en consecuencia.

```
<input type="text" onkeydown="mostrarTecla(event)" />

<script>
  function mostrarTecla(event) {
    var tecla = event.key;
    alert("Has presionado la tecla: " + tecla);
  }
</script>
```

En este ejemplo, cuando el usuario presiona una tecla en el campo de texto, se ejecuta la función `mostrarTecla(event)`, que captura la tecla presionada y muestra un cuadro de diálogo emergente con el mensaje “Has presionado la tecla: [tecla]”.

Evento onload

El evento `onload` se desencadena cuando la página web se carga por completo, incluidos todos los recursos externos, como imágenes y hojas de estilo. Este evento es útil para realizar tareas una vez que la página está lista y todos los elementos están disponibles.

```
<body onload="cargarPagina()">

<script>
  function cargarPagina() {
    alert("La página se ha cargado completamente.");
  }
</script>
```

En este ejemplo, cuando la página se carga por completo, se ejecuta la función `cargarPagina()`, que muestra un cuadro de diálogo emergente con el mensaje “La página se ha cargado completamente.”

Otros eventos comunes

Además de los eventos mencionados anteriormente, existen muchos otros eventos que se pueden utilizar en los elementos HTML. Algunos ejemplos adicionales son:

- **onsubmit**: Se desencadena cuando se envía un formulario.
- **onchange**: Se desencadena cuando el valor de un elemento cambia (por ejemplo, un campo de entrada).
- **onfocus**: Se desencadena cuando un elemento obtiene el foco.
- **onblur**: Se desencadena cuando un elemento pierde el foco.

```
<input type="text" onchange="mostrarValor(this.value)" />

<script>
  function mostrarValor(valor) {
    alert("El valor ha cambiado a: " + valor);
```

```
    }
</script>
```

En este ejemplo, cuando el valor del campo de texto cambia, se ejecuta la función `mostrarValor(valor)`, que muestra un cuadro de diálogo emergente con el mensaje “El valor ha cambiado a: [valor]”.

Conclusión

Los eventos en los elementos HTML permiten la interacción del usuario y la ejecución de código JavaScript en respuesta a acciones específicas. A través de eventos como `onclick`, `onmouseover`, `onkeydown`, `onload` y muchos otros, puedes capturar y responder a las acciones del usuario de manera dinámica.

Utilizando eventos en conjunto con funciones JavaScript, puedes crear interacciones personalizadas y agregar funcionalidad a tus páginas web. Experimenta con diferentes eventos y acciones para lograr la experiencia interactiva deseada en tu proyecto web.