DOM Events

Types of events

The different types of events that can occur on a browser are:

- **Load:** when the window, image or script loads.
- **Click:** when a button, paragraph etc is clicked
- **Keys events:** keyup, keydown, keypress etc.
- **Mouse events:** mousenter, mouseexit, hover
- **Submit:** when form is submitted

Load Event

The window.onload() event occurs when the window object is loaded or the page is loaded.

```
window.onload = function(){
  console.log("This is my website.");
};
```

Click Event

The onclick() event occurs when the user clicks on an element.

```
button1.onclick = function(){
    console.log("This is clicked");
};
```

Add event listener

The addEventListener() method is used to attach an event handler to an element. A function is called when the event happens.

```
element1.addEventListener("click", function() {
  document.getElementById("para").innerHTML = "This is my
website.";
```

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution is prohibited.

```
    });
```

Event Handler Context

When we use ‘this’ with an event handler, it gives us the DOM node on which the event has occurred.

```
    document.querySelector('button').addEventListener('click',
    function {
       console.log(this);
    }; // Output: <button> Click Me </button>
```

Event Object

All event objects in the HTML DOM are based on the Event Object. There are many properties of the Event Object which are useful. Some of the information that can be useful are:

- type: the type of event
- shiftKey: Is the shift key pressed when clicked?
- screenX, screenY: where on the button, the click event has happened.

```
document.querySelector('button').addEventListener('click', function
{
   console.log(event);
}; // Output: MouseEvent object
```

Event Propagation

The event propagation tells us the order in which the events are received by the HTML elements.
Bubbling: By default, the events are handled in this manner. In bubbling, the events are propagated from the child element to its parents and then ancestors.

```
    let grandParentDiv = document.querySelector("#div1");
    let parentDiv = document.querySelector("#div2");
    let childDiv = document.querySelector("#div3");

     grandParentDiv.addEventListener("click", function (event) {
           alert("Div 1 clicked");
```

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution is prohibited.

```
  });

parentDiv.addEventListener("click", function (event) {
      alert("Div 2 clicked");
  });

childDiv.addEventListener("click", function (event) {
      alert("Div 3 clicked");
  });
```

Here the click() event is propagated from childDiv to parentDiv to grandparentDiv.

Stop Propagation

The stopPropagation() method is used to prevent the propagation of events from bubbling up. It's used with the event object.

```
let grandParentDiv = document.querySelector("#div1");
let parentDiv = document.querySelector("#div2");
let childDiv = document.querySelector("#div3");

 grandParentDiv.addEventListener("click", function (event) {
       alert("Div 1 clicked");
   });

 parentDiv.addEventListener("click", function (event) {
       alert("Div 2 clicked");
       event.stopPropagation();
  });

 childDiv.addEventListener("click", function (event) {
       alert("Div 3 clicked");
  });
```

Here, the stopPropagation() method is used to stop the propagation of the event from the parent to the grandparent.

Proprietary content. © Great Learning. All Rights Reserved. Unauthorized use or distribution is prohibited.