Events:

In JavaScript, events are actions or occurrences that happen in the browser that the browser can detect and respond to. Events can be triggered by user interactions like clicking a button, typing in a text field, moving the mouse, or by the browser itself such as when the page finishes loading.

Some Examples:

1. click: Triggered when an element is clicked.
2. dblclick: Triggered when an element is double-clicked.
3. mouseover: Triggered when the mouse pointer hovers over an element.
4. mouseout: Triggered when the mouse pointer leaves an element.
5. mousemove: Triggered when the mouse pointer moves within an element.
6. mousedown: Triggered when a mouse button is pressed down on an element.
7. mouseup: Triggered when a mouse button is released over an element.

Example:

```
<button id="container">Click Me</button>

    <script>
        document.querySelector("#container").addEventListener("click", ()=>{
            alert("Button Clicked");
        });
    </script>
```

On clicking the button, we get to see an alert "Button Clicked".

Event Bubbling:

Event bubbling is a concept in JavaScript where an event triggered on a child element propagates (or "bubbles up") to its parent and ancestor elements in the DOM (Document Object Model).

Example:

```
 <div class="container">
    <div class="childContainer">
        <div class="child">
            I am a Child
        </div>
    </div>
</div>

<script>
    document.querySelector(".child").addEventListener("click", ()=>{
        alert("Child was clicked");
        });

    document.querySelector(".childContainer").addEventListener("click", ()=>{
        alert("Child Container was clicked");
    });

    document.querySelector(".container").addEventListener("click", ()=>{
         alert("Container was clicked");
    });
</script>
```

1. **Child Element: When you click on the element with the class .child, the click event is first captured by this element, and the alert "Child was clicked" is shown.**
2. **Child Container: After the event is handled by the .child element, it bubbles up to the parent element with the class**

.childContainer. The alert "Child Container was clicked" is shown.

3. **Container: The event then bubbles up to the outermost parent element with the class .container, showing the alert "Container was clicked".**

To stop this bubbling, we use stopPropagation().

1. If we want to stop this bubbling after viewing the alert box where we are able to see "Child was clicked", we write the below code:

```
document.querySelector(".child").addEventListener("click", (e)=>{
    e.stopPropagation();
    alert("Child was clicked");
});

document.querySelector(".childContainer").addEventListener("click", ()=>{
    alert("Child Container was clicked");
});

document.querySelector(".container").addEventListener("click", ()=>{
    alert("Container was clicked");
});
```

2. If we want to stop this bubbling after viewing the alert box where we are able to see "Child Container was clicked", we write the below code:

```
document.querySelector(".child").addEventListener("click", ()=>{
    alert("Child was clicked");
});

document.querySelector(".childContainer").addEventListener("click", (e)=>{
    e.stopPropagation();
    alert("Child Container was clicked");
});

document.querySelector(".container").addEventListener("click", ()=>{
    alert("Container was clicked");
});
```

3. If we want to stop this bubbling after viewing the alert box where we are able to see "Container was clicked", we write the below code:

```
document.querySelector(".child").addEventListener("click", ()=>{
    alert("Child was clicked");
});

document.querySelector(".childContainer").addEventListener("click", ()=>{
    alert("Child Container was clicked");
});

document.querySelector(".container").addEventListener("click", (e)=>{
    e.stopPropagation();
    alert("Container was clicked");
```

```
});
```