Contents

[Event Handling 2](#_Toc110774430)

[Event object 2](#_Toc110774431)

[Mouse & Keyboard Events 3](#_Toc110774432)

[Event Bubbling 4](#_Toc110774433)

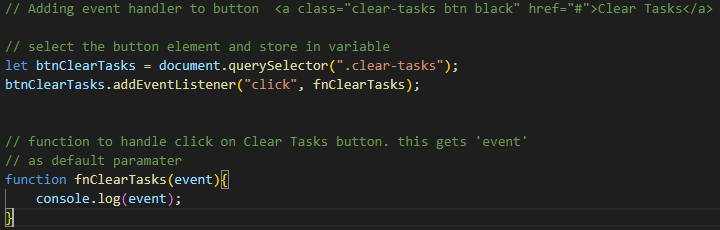
[Event delegation 4](#_Toc110774434)

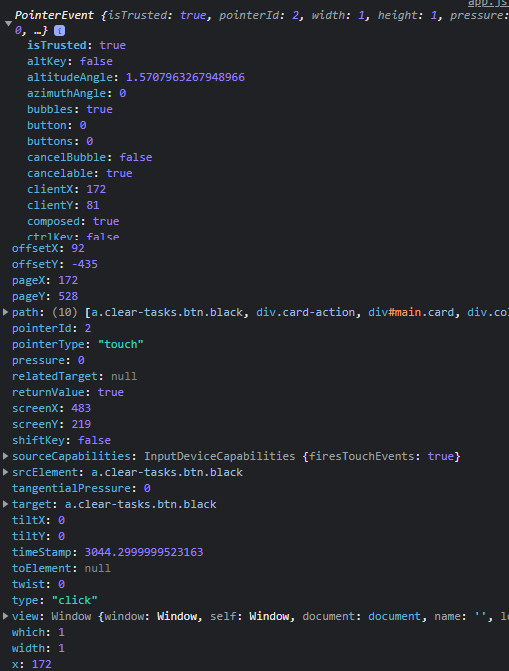
# Event Handling

## Event object

Multiple type of event occurs on a page. One of the most common we handle during development is click event.

If we simply click print event by adding a event listener as shown below





We get ‘event’ object which contains large about of info. We can use this info to perform different logic

## Mouse & Keyboard Events

Event can be added on any target element in DOM. Same way mouse events can be added to buttons and has multiple actions covered. Some of them are as below.

Same way keyboard and other common events are keydown (key press down), keyup (key press up),keypress (key press up-down),focus (selecting element),blur (de selecting element, loosing focus),cut (cut data from text box),paste(paste data in text box)

const clearBtn = document.querySelector('.clear-tasks');

const card = document.querySelector('.card');

const heading = document.getElementById('task');

console.log(heading);

// mouse click event on button

//\*\*\*\*\*\*\*\*\*\*\*\* clearBtn.addEventListener('click', handleEvent);

// mouse double click event on button

//\*\*\*\*\*\*\*\*\*\*\*\* clearBtn.addEventListener('dblclick', handleEvent);

// Mousedown event

//\*\*\*\*\*\*\*\*\*\*\*\* clearBtn.addEventListener('mousedown', handleEvent);

// mouse button up event

//\*\*\*\*\*\*\*\*\*\*\*\* clearBtn.addEventListener('mouseup', handleEvent);

// Mouseenter inside the card

//\*\*\*\*\*\*\*\*\*\*\*\* card.addEventListener('mouseenter', handleEvent);

// Mouseleave the card

//\*\*\*\*\*\*\*\*\*\*\*\* card.addEventListener('mouseleave', handleEvent);

// Mouseover trigger when mouse is over card, when mouse goes over child element inside the

// card the mouse out event is triggered, which means mouse is no more over the card.

//\*\*\*\*\*\*\*\*\*\*\*\* card.addEventListener('mouseover', handleEvent);

// Mouseout

//\*\*\*\*\*\*\*\*\*\*\*\* card.addEventListener('mouseout', handleEvent);

// Mousemove

//\*\*\*\*\*\*\*\*\*\*\*\* card.addEventListener('mousemove', handleEvent);

// Event Handler

function handleEvent(e) {

  console.log(`EVENT TYPE: ${e.type}`);

  heading.value= `MouseX: ${e.offsetX} MouseY: ${e.offsetY}`;

}

Sdasdzxcz zxczxczxczxc

## Event Bubbling

Evets occurring on webpage on DOM element keeps flowing/bubbling up automatically till parent and root element when parents are also registered to handle same type of events.

In below example if we click on ‘p’ elements it gives 3 alerts including for ‘div’ and ‘form’, because its parents ‘div’ and ‘form’ are also subscribed to ‘onClick’ event.

Please note even if ‘p’ element won’t have alert on ‘onClick’ event or doesn’t have click event handler then also we will get two alert boxed from div and form element because it’s that event which propagates/bubble up to top and whoever is handling that event in the way gets invoked.

We can also stop the propagation if we want using event.stopPropagation()

<style>

  body \* {

    margin: 10px;

    border: 1px solid blue;

  }

</style>

<form onclick="alert('form')">FORM

  <div onclick="alert('div')">DIV

    <p onclick="alert('p')">P</p>

  </div>

</form>

## Event delegation

As we have seen in ‘event bubbling’ event gets propagated to parents till root, and we can take advantage of this feature for handling events on all elements in page. Instead of registering a separate event handler on individual child element we can just register handler on root/body element and add conditions to check if event target is child element and perform the required action. This mimic adding event handler on that child element.

// add event listener to body

document.body.addEventListener('click', deleteItem);

function deleteItem(e){

// add check to find details on target and check if is has details like particular class

// if a child element we want and perform respective operation

  if(e.target.id === 'delete-task-button')){

// remove whole item where button is inside by traversing to parent

    e.target.parentElement.parentElement.remove();

  }

}