



NAZARBAYEV  
UNIVERSITY



# Web Programming and Problem Solving

## JavaScript Events

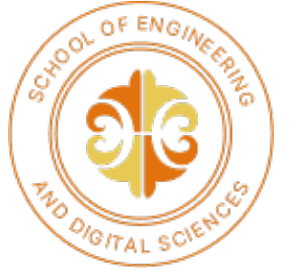
Date: 16.11.2022

Instructor: Zhandos Yessenbayev



NAZARBAYEV  
UNIVERSITY

# Content



- Introduction
- Inline Events
- Event Listeners
- Event Object

When browsing a website, the user can **interact** with the website by:

- selecting, clicking, hovering over the elements on the page;
- filling the forms and pressing the keys;
- resizing or closing the browser window;
- playing or pausing the video or audio track, and so on . . .

All of these actions are called **events** and can be **handled** by the webpage.

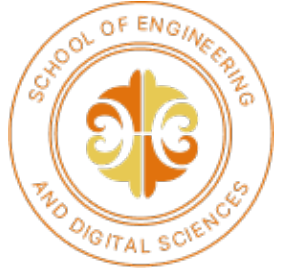
Each time a user interacts with the web page, the browser triggers one of the predefined events such as:

- Mouse Events
- Keyboard Events
- Form Events
- Window Events

To handle the event, we can **directly** write event handler in the HTML element's opening tag.



# Inline Events



Each type of events has a **predefined name** to handle that event.

For example for the mouse events:

- mouseover
- mouseenter
- mouseout
- etc.

```
<p id="text1"  
  onmouseenter="style='background-color: yellow'"  
  onmouseout="style='background-color: white'"  
>
```

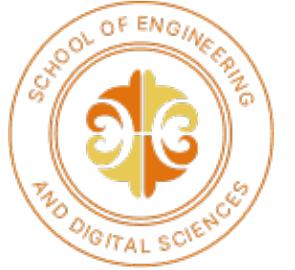
# this keyword

**this** keyword can be used to access the element where the event has been fired.

```
<h2 id="title" onclick="this.style.color = 'red'">  
  Inline Events  
</h2>  
  
<p id="text2" onclick="changeFont(this)">
```



# Event Listeners



Inline event handling has some drawbacks:

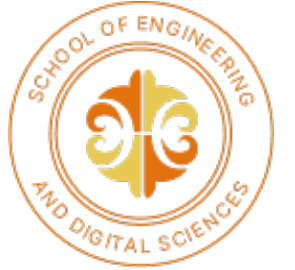
- Difficult to manage the code (debugging, flexibility)
- Cannot have different handler of the same event
- Mixes HTML and JavaScript in one document

To overcome these problems, we can use **event listeners**.

Using the event listeners, we can achieve the same results but have more flexibility and features.



# Event Listeners



- Syntax of the event listeners

*element.addEventListener(event, function, useCapture);*

```
let h = document.getElementById("title");  
h.addEventListener(  
    "click",  
    function (event) {  
        this.style.color = "red";  
    }  
);
```

Event **type**

Event **handler**

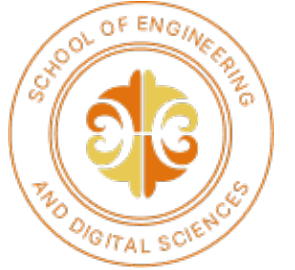
Note the **event** argument  
of the function – **event object**

The **removeEventListener()** method removes event handlers that have been attached with the **addEventListener()**





# Event Object



In some cases it is important not only know what type of event happened, but also know the context of the event such as:

- what combination of the keys was pressed?
- what are the the coordinates of the mouse when clicked?
- when the event happened?

These context information is called **event object** and the can be accessed in the event listeners.



# Event Object

The event object can be used to provide better user experience, add some features to the page, or something else.

---

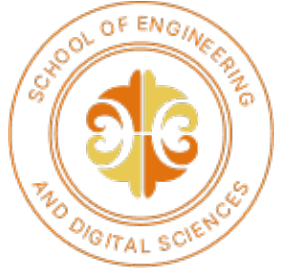
▼ `MouseEvent {isTrusted: true, screenX: 223, screenY: 322, clientX: 223, clientY: 220, ...}` ⓘ

```
isTrusted: true
altKey: false
bubbles: true
button: 0
buttons: 0
cancelBubble: false
cancelable: true
clientX: 223
clientY: 220
composed: true
ctrlKey: false
```

```
h.addEventListener(
  "click",
  function (event) {
    console.log(event.clientX, event.clientY)
  }
);
```



# Summary



- **Key takeaways:**

- **JavaScript** can be used to handle **events** caused by user interaction with the web page: mouse, keyboard, browser, form, window events
- There are **two ways** to handle the events on the webpage:
  - **Inline** events
  - Event **listeners**
- **this** keyword can be used to access the element where event happened
- To get more information about the event, we can use **event object** provided by the event listeners

Thanks for Attention!