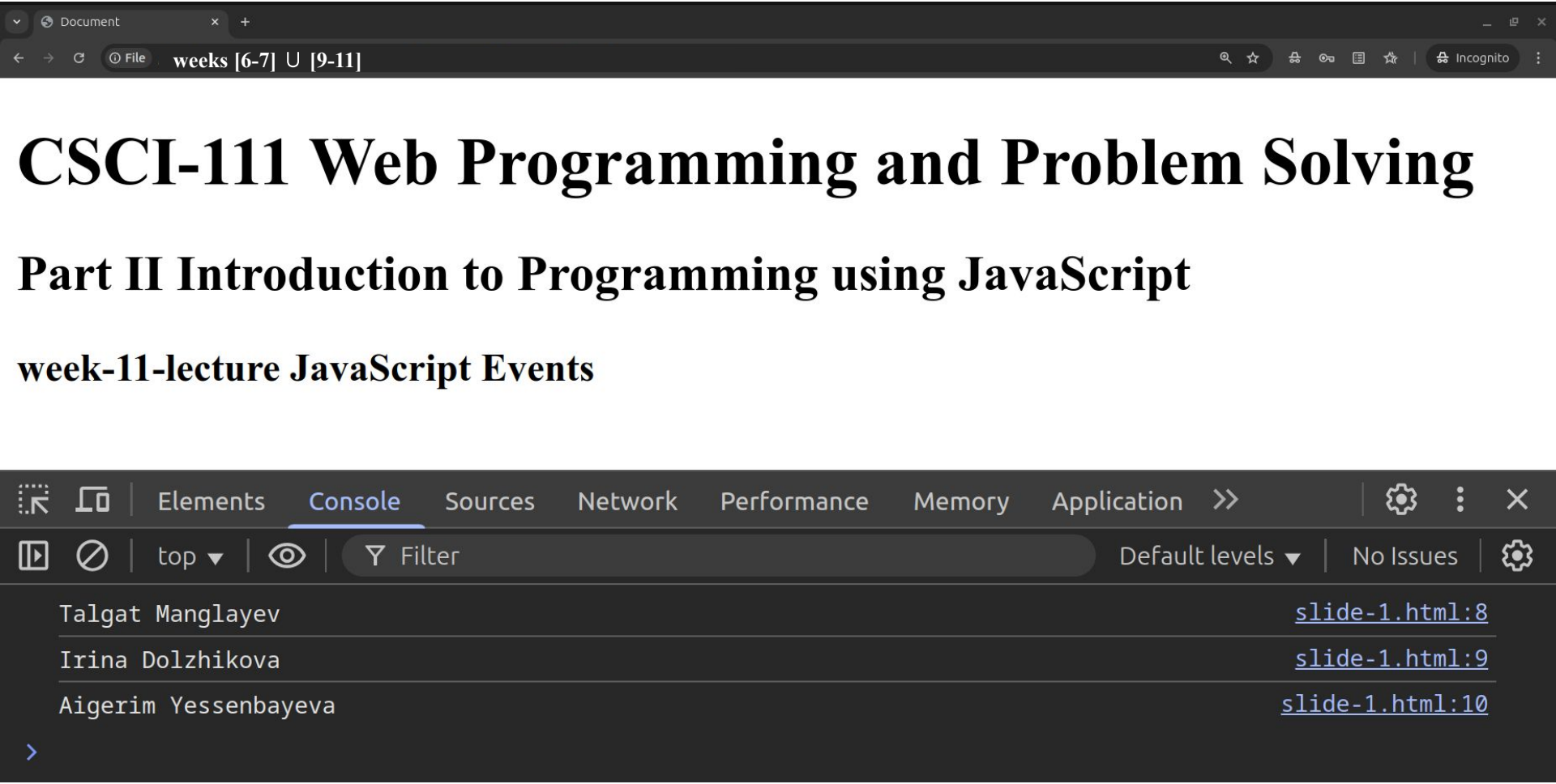

Document
File
weeks [6-7] ∪ [9-11]
Incognito
CSCI-111 Web Programming and Problem Solving
Part II Introduction to Programming using JavaScript
week-11-lecture JavaScript Events
Elements
Console
Sources
Network
Performance
Memory
Application
top
Filter
Default levels
No Issues
Talgat Manglayev
slide-1.html:8
Irina Dolzhikova
slide-1.html:9
Aigerim Yessenbayeva
slide-1.html:10

Content

- Introduction
- Inline Events
- Event Listeners
- Event Object

Introduction

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 website

Inline events

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

`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,** which are more flexible and interactive.

Event listeners. Syntax.

```
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

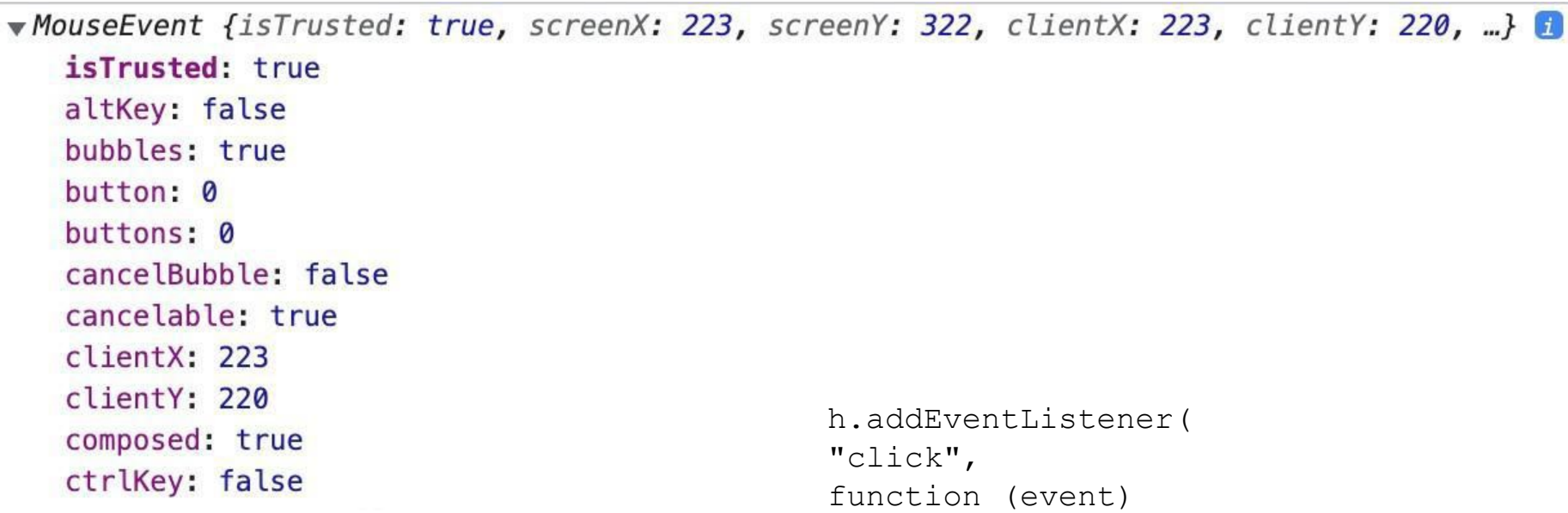
In some cases it is important not only know what type of event happened, but also know the context of the event such as:

- what combination of the keys was pressed?
- what are the 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.

Style manipulation

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



Useful links

- https://www.w3schools.com/js/js_events.asp
- https://www.w3schools.com/js/js_htmldom_events.asp
- https://www.w3schools.com/js/js_htmldom_eventlistener.asp
- https://www.w3schools.com/js/js_events_examples.asp
- https://www.w3schools.com/js/js_this.asp

Summary

- **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

bonus task: create GitHub accounts