JavaScript

Working with Events

Peter Kassenaar – info@kassenaar.com

Event Driven JavaScript

Making your website or app react to events

Event-driven JavaScript

- Events happen – mostly – in the browser

  - either system-dependent (`scroll`, `load`, etc)

  - or initiated by the user (`click`, `mouseover`, `hover`)

  - Also: events thrown by external systems (http-connection, websockets)

- Events are fired by (elements on the) page

- Events are always written in lowercase.

- Events *can* be captured and trigger the execution of a script
  - Capturing is not required
  - If not captured, an event simply dies, without something happening
  - Events 'bubble up' the DOM tree (most of them)
- Some events can be fired by all elements
  - like `mouseover`
- Some by only some elements
  - like `change`

Events example

- Window
  - `load`
  - `resize`
  - `unload`
  - `scroll`
- input, select, textarea
  - `change`
  - `focus`
  - `blur`
- button
  - `focus, blur`

- visible elements
  - `mouseover`
  - `mouseup`
  - `mouseout`
  - `mousedown`
  - `mousemove`
- all
  - `click`
- …dozens of others (countless!)

Relatively new: mobile events

- `touch`
- `tap`
- `swipe`
- `pinch`
- `zoom`
- `rotate`
- `orientationchange`
- `And more…`

Events ARE NOT HTML-attributes

- Don't do this inline: (*deprecated!*)

    - onload

    - onmouseover

    - etc

    - `<button onclick="doFunction()">`

- Instead, use `addEventListener()`

    - `load`

    - `mouseover`

    - etc.

- Write a function, the Event handler, to do something

    - This can be an anonymous function or named function, like

    - onLoad()

    - onMouseover() etc.

eduvision
opleiding & training

In short, NEVER do this:

```
HTML

<button onclick="alert('Hello, this is my old-fashioned event handler!');">
   Press me
</button>
```

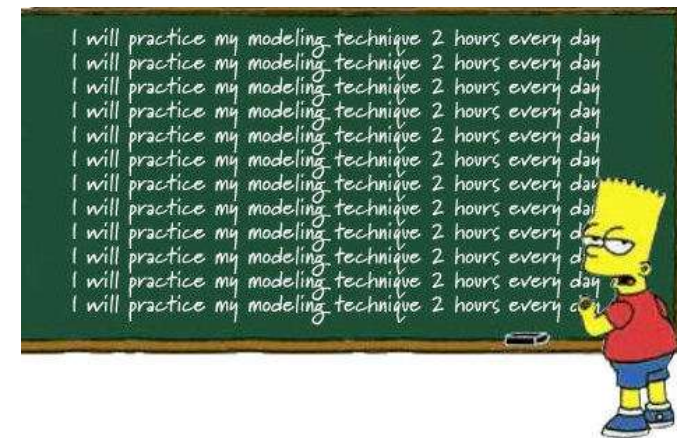# Instead, do something like:

```html
HTML                                          </> Play

<input id="textBox" type="text" />
<div id="output"></div>
```

Or, use document.getElementById (...)

```js
JS                                            </> Play

const textBox = document.querySelector("#textBox");
const output = document.querySelector("#output");
textBox.addEventListener("keydown", (event) => {
  output.textContent = `You pressed "${event.key}".`;
});
```

The event is passed as a parameter to the event handling function

Mini workshop

Create the example from the previous slide

- Catch a keypress

- Show in the console, or show in the page which key was pressed

- Q: How can you get an overview of all event-properties available in the `event` object?
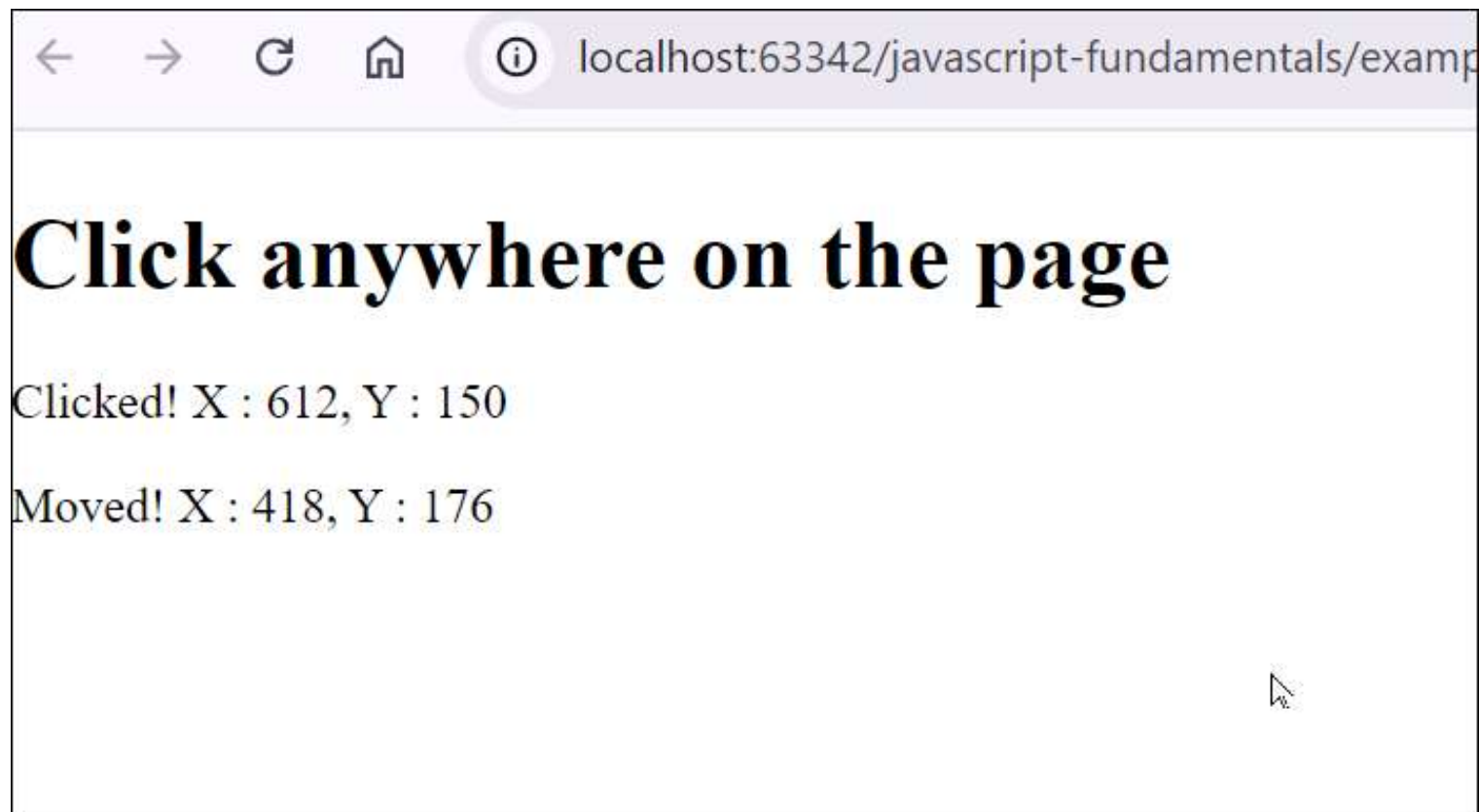
Example – getting the cursor location

```html
<script>
    // example: catching mouse clicks and mouse moves.
    window.addEventlistener('load', function(){
        document.addEventListener('click', handleClick);
        document.addEventListener('mousemove', handleMove);
    };

    // Event handlers
    function handleClick(evt){
        const xCoord = evt.clientX;
        const yCoord = evt.clientY;
        const msg = 'Clicked! X : ' + xCoord + ', Y : ' + yCoord;
        document.getElementById('coords').innerHTML = msg;
    }

    function handleMove(evt){
        const xCoord = evt.clientX;
        const yCoord = evt.clientY;
        const msg = 'Moved! X : ' + xCoord + ', Y : ' + yCoord;
        document.getElementById('mouseMove').innerHTML = msg;
    }
</script>
```

```html
<h1>Click anywhere on the page</h1>
<p id="coords"></p>
<p id="mouseMove"></p>
```
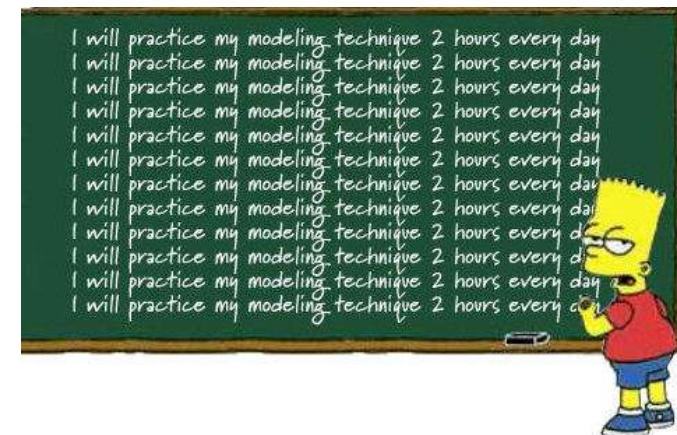
Sample output

The **events** are `click` and `mousemove`.

The **properties** used are `clientX` and `clientY`.

Workshop #30

- Create a script where after a button is clicked, all links in the page change background colour.

- Usage:

  - A simple HTML-page, containing some hyperlinks

  - An array of colours

  - `getElementsByTagName()` or `document.querySelectorAll()`

  - Loop over a NodeList object

Sample output
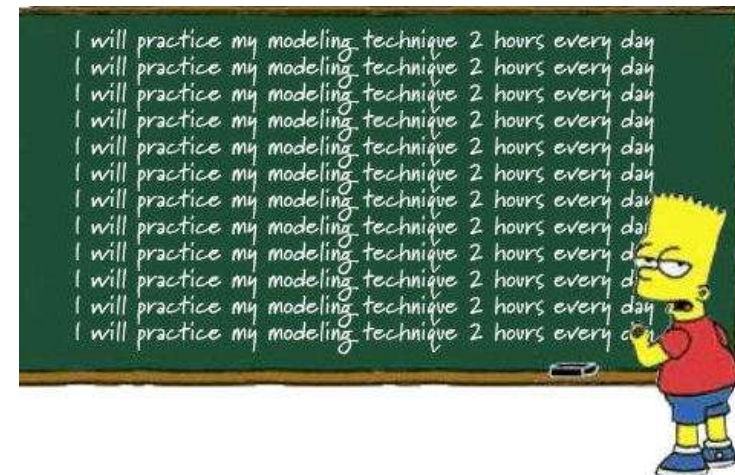
Workshop #31

- Write a script that checks a value in a form:

- a `<textarea>` element must contain no more than xx characters. Set this number yourself.

  - (Yes, like Twitter, Mastodon and other Social Media Platforms)

- Show how many characters the visitor can still type

- Show if the number of characters is exceeded.

- the check takes place after each character is typed in the textarea

  - Tip: use the event `input`

Sample output

Character Counter Example

Type in the textarea below. The maximum allowed characters are 100.

```
Lorem ipsum dolor sit,
bladybladybla
```

You can type 64 more characters.

Color (CSS-class) if OK

Character Counter Example

Type in the textarea below. The maximum allowed characters are 100.

```
Lorem ipsum dolor sit,
bladybladybla<script>
// Maximum number of characters
allowe const maxChars = 100;
```

You have exceeded the limit by 5 characters!

Color (CSS-class) if Not OK

Workshop #32

- An input field must contain a number smaller than 10

- This check takes place after this input field has lost focus

  - This comes in handy for age checks, not exceeding a certain value, etc.

- This is an easy one!

Number Validation Example

Enter a number smaller than 10:

`9`

Number Validation Example

Enter a number smaller than 10:

`99`

Please enter a number smaller than 10.