Introduction to Events

This lesson is a brief introduction to events and how they help adding interactivity to the web page. You will also learn how to add or remove an event from a particular HTML element.

WE'LL COVER THE FOLLOWING

Up until now, your JavaScript code was executed right from the start. The execution order of statements was determined in advance and the only user interactions were data input through `prompt()` calls.

To add more interactivity, the page should react to the user's actions: clicking on a button, filling a form, etc. In that case, the execution order of statements is not determined in advance anymore, but depends on the user behavior. His actions trigger events that can be handled by writing JavaScript code. This way of writing programs is called event-driven programming. It is often used by user interfaces, and more generally anytime a program needs to interact with a user.

First Example

Here's some starter HTML code and the associated JavaScript code.

Output

JavaScript

HTML

```
<html>
```

```
 <head>
 </head>
 <body>

                <button id="myButton">Click me!</button>
 </body>
</html>
```


Adding an Event Listener

Called on a DOM element, the `addEventListener()` method adds a *handler* for a particular event. This method takes as parameter the *event type* and the associated *function*. This function gets called whenever an event of the corresponding type appears for the DOM element.

The above JavaScript code could be rewritten more concisely using an anonymous function, for an identical result.

Output

JavaScript

HTML

```
// Show a message when the user clicks on the button
document.getElementById("myButton").addEventListener("click", () => {
  alert("Hello!");
});
```


Removing an Event Listener

In some particular cases, you might want to stop reacting to an event on a DOM element. To achieve this, call the `removeEventListener()` on the element, passing as a parameter the function which used to handle the event.

> This can only work if the handler function is not anonymous.

Output

JavaScript

HTML

```
// Show a message when the user clicks on the button
document.getElementById("myButton").addEventListener("click", () => {
  alert("Hello!");
});

// Remove the handler for the click event
buttonElement.removeEventListener("click", showMessage);
```