



WEB разработка

JavaScript
Events

Съдържание

- JavaScript Event Model
- Регистриране на събития/event
 - HTML Attributes,
 - DOM Properties
 - DOM методи
- Event - обект

JavaScript Event Model

JavaScript Event Model

- **DOM event model** - дава възможност на потребителя да взаимодейства с уеб-страницата /браузъра/
- Състои се от event/event-и и event listener(s), свързани към DOM - обект.

`<button> Click Me</button>`

event listener (функция) - `onButtonClick()`

Event Types

Event Types

- Event-и, които се използват в 99% от случаите
 - **Mouse** events
 - **Touch** events
 - **Form** events
 - **Keyboard** events
 - **DOM** events
- Пълен списък на типовете event-и.
 - <http://www.w3.org/TR/DOM-Level-3-Events/#event-types-list>
- Може да дефинираме и Custom Event Types - свои типове event-и

Event Types - най-често използвани

Mouse Events	Touch Events	Keyboard Events	UI Events	Focus Events
<code>click</code>	<code>tap</code>	<code>keypress</code>	<code>load</code>	<code>blur</code>
<code>hover</code>	<code>touchstart</code>	<code>keydown</code>	<code>abort</code>	<code>focus</code>
<code>mouseup</code>	<code>touchend</code>	<code>keyup</code>	<code>select</code>	<code>focusin</code>
<code>mousedown</code>	<code>touchmove</code>		<code>resize</code>	<code>focusout</code>
<code>mouseover</code>	<code>touchcancel</code>		<code>change</code>	
<code>mouseout</code>	<code>touchenter</code>		<code>input</code>	
	<code>touchleave</code>			

Регистриране на Event

Event Handlers

- Регистрираме event чрез т. нар **callback - функция** за конкретен тип event и **DOM - елемент**.
- Има три начина за регистриране на event:
 - Като HTML - атрибут
 - Използване на свойство на DOM
 - Използване на DOM event handler

As HTML Attribute

- Като HTML атрибут

В html кода →

```
<button onclick="onButtonClick()">Click Me</button>
```

В js кода →

```
function onButtonClick() {  
  console.log("You clicked the Button");  
}
```

Задача→

As HTML Attribute - задача

Напишете JS, с който при преминаване с мишката над картинката - от усмихнатата става сериозна и текста под картинката се сменява от I am serious със Just Kidding!



I am serious



Just kidding

Регистриране на Event Handlers със свойство на DOM - елемента

Прилагаме DOM - event върху конкретен DOM - елемент и присвояваме референция към функция. /Само звучи ужасно :)/

всъщност→

В html кода → имаме шпионин - бутон, който чака да бъде кликнат за да предизвика действие

```
<button id="click-button">Click me</button>
```

В js кода→

- подготвили обекта, съответстващ елемент от HTML кода,
- казали сме му при какви обстоятелства(когато бъде кликнат)
- Какво да направи - да логне някакво съобщение

```
var button = document.getElementById("click-button");  
button.onclick = function onButtonClick() {  
  console.log("You clicked the button");  
}
```

Рег... със свойство на DOM - елемента - задача

Напишете JS код, който сменява настроението на Smiley след натискане на бутон.



Just kidding!

Change Mood

Използване на DOM Event Listeners

- Стандартния начин за прикачване на event handler-и към DOM

- **СИНТАКСИС**

```
domElement.addEventListener(eventType, eventHandler, isCaptureEvent);
```

- *пример*

```
var button = document.getElementById("click-button");  
button.addEventListener("click", function () {  
    console.log("You clicked me");  
}, false);
```

Използване на DOM Event Listeners - задача

Напишете JS код, който сменява настроението на Smiley след натискане на бутон.



I am serious

Change Mood

Bonus****

Event Object



Event Object

- Event handlers имат достъп до т. нар. event object.
- Той съдържа информация за:
 - Типа event
 - Накъде е насочен/**target-a** на event
 - Клавишът, който е натиснат, ако е задействан keyboard event
 - Кой бутон на мишката е натиснат, ако е задействан mouse-event
 - Позиция на мишката върху екрана

For debugging purposes

Event Object

- Как достъпваме event object-а и неговите свойства/информацията, която съдържа/ - подаваме го като аргумент на функцията, управляваща event-а /function handler/

```
function onClick(event) {  
  console.log(event.target);  
  console.log(event.type);  
  console.log(event.clientX, event.clientY);  
}  
  
button.addEventListener("click", onClick, false);
```

demo

Bonus***

Capturing and Bubbling Events

Capturing & Bubbling Events

- Когато, потребителят кликне върху HTML - елемент, event-а се задейства и върху неговите родители.

```
<html>
  <body>
    <div>
      <button>
        Click Me
      </button>
    </div>
  </body>
</html>
```

При кликване на бутона -

- Той си остава целта на event-а, но click event-а се разпространява върху всички негови родители по веригата навън - event chain

demo

Capturing and Bubbling Events

- Има два вида event chain
 - Capturing и Bubbling
- **Bubbling** - event-а върви нагоре по веригата
 - Първо се задейства събитието върху целевия елемент
 - После върху родителя му, върху родителя на родителя и т.н.
- **Capturing** - събитието върви надолу по-веригата
 - Първо върху родителя на всички елементи
 - Последно се изпълнява върху целевия елемент.

Capturing and Bubbling Events

демо