**Event Model**

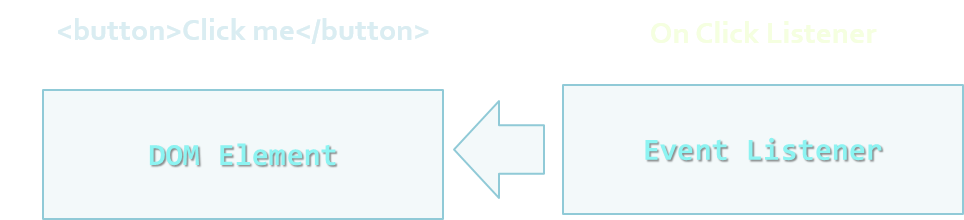
**Touch, Mouse, Keys**

**Table of Contents**

* JavaScript Event Model
* Event Registration
* HTML Attributes, DOM Properties and Methods
* Event Object
* Cross-Browser Event Handler
* Capturing and Bubbling Events
* Custom Events

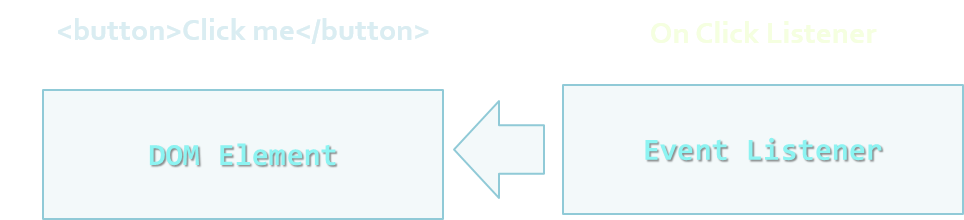
**JavaScript Event Model**

* The DOM event model provides a way for the user to interact with the browser environment
* The DOM event model consists of events and event listeners attached to the DOM objects



**Event Types**

* DOM provides a set of common event types that are used in 99% of the time
  + Mouse events
  + Touch events
  + Form events
  + Keyboard events
  + DOM events
* Full list of all DOM event types:
  + <http://www.w3.org/TR/DOM-Level-3-Events/#event-types-list>
* You could also defined Custom Event Types



**Common Event Types**

| **Mouse Events** | **Keyboard Events** |
| --- | --- |
| click | keypress |
| hover | keydown |
| mouseup | keyup |
| mousedown |  |
| mouseover |  |
| Mouseout |  |

| **UI Events** | **Focus Events** |
| --- | --- |
| load | blur |
| abort | focus |
| select | focusin |
| resize | focusout |
| change |  |
| input |  |

| **Touch Events** |
| --- |
| tap |
| touchstart |
| touchend |
| touchmove |
| touchcancel |
| touchenter |
| touchleave |

**Event Registration**

**Event Handlers**

* The developer could register an event handler/listener for a specific event type and DOM element
* The registration can be performed with:
  + HTML Attributes
  + Using DOM element properties
  + Using DOM event handler

**As HTML Attibute**

* Event handlers can be attached by simply setting a value to the handler attribute
  + This value is pure JavaScript and is not always a function

<button onclick="buttonClickFunction()">Click Me</button>

function buttonClickFunction() {

console.log("You click the Button");

}

***Register Event Handlers using HTML Attributes - Live Demo***

**Using DOM Element Properties**

* Use standard DOM events on certain DOM element and assign a reference to a function
  + Can be anonymous

**HTML**

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

**JavaScript**

var button = document.getElementById("click-button");

button.onclick = function onButtonClick() {

console.log("You clicked the button");

}

***Using DOM Element Properties - Live Demo***

**Using DOM Event Listeners**

* The standard way for attaching event handlers to DOM
  + The Basic Syntax is:

domElement.addEventListener(eventType,

eventHandler,

isCaptureEvent)

* *Example:*

var button = document.getElementById("click-button");

button.addEventListener("click", function () {

console.log("You clicked me");

}, false);

**The Event Object (Get the Event Data)**

**Event Object**

* The event handlers have access to the event object passed as function parameter
* The event object contains information about:
  + The **type** of the event
  + The **target** of the event
  + The **key that was pressed** when a   
    keyboard event was fired
  + The **mouse button that was pressed** when a  
     mouse event was fired
  + The **position** of the mouse on the screen
* The event object is accessible as the only argument of the function handler

function onButtonClick(event) {

console.log(event.target);

console.log(event.type);

console.log(event.clientX, event.clientY);

}

button.addEventListener("click", onButtonClick, false);

* Yet, there is IE - it does not pass event object
  + Keeps the event object in window.event
  + Fortunately there is a simple fix

function onButtonClick(event) {

if(!event) event = window.event;

// Your code…

}

***Event Object - Live Demo***

**Cross-browser Event Handlers (Remember a certain browser)**

**Cross-browser Compatibility**

* addEventListener() is n0t supported everywhere
  + Older versions of IE have their own method for registering event handlers

domElement.attachEvent('onclick', eventHander);

* Yet, you can use **feature detection**

// Up to IE8

if (document.attachEvent){

domElement.attachEvent(…);}

// IE 9, IE 10, Firefox, Chrome, Opera, Safari

else if (document.addEventListener) {

domElement.addEventListener(…); }

// Reeeally old browsers

else { domElement["on" + eventType] = handler; }

**CrossBrowser Event Handler**

* This can be wrapped in a method:
  + Create a function with three parameters
    - Target element
    - Event type
    - Event handler
  + Use the method your browser supports

***Cross-Browser Event Handler - Live Demo***

**Capturing and Bubbling Events**

**Top to Bottom and the other way around**

**The Event Chain**

* When the user clicks on an HTML element, the event is also fired on all of its parents

<html>

<body>

<div>

<button>

Click Me

</button>

</div>

</body>

</html>

* The **button is still the target**, but the click event is fired on all of its parents
  + An event is fired on all elements in the chain

***The Event Chain - Live Demo***

**Two Types of Event Chains**

* There are two types of event chains
  + Capturing and Bubbling
* **Bubbling** handlers **bubble up** the chain
  + The first executed handler is on the target
  + Then its parent's, and its parent's, etc…
* **Capturing** handlers **go down** the chain
  + The first executed handler is on the parent of all
  + The last executed handler is on the target

**Capturing Event Chain**

* Capturing goes down the event chain
  + The first executed handler is the one of the parent of all

**HTML**  
**Body**  
**DIV**  
**BUTTON**

User clicks the ***Button***

**Bubbling Event Chaing**

* Bubbling bubbles up the event chain
  + The first executed handler is the one on the target

**HTML**  
**Body**  
**DIV**  
**BUTTON**

User clicks the ***Button***

# https://cdn.rawgit.com/TelerikAcademy/JavaScript-UI-and-DOM/master/08.%20Event%20Model/slides/imgs/section-demo-capturing-bubbling.png *Capturing and Bubbling Event Chains - Live Demo*

**Custom Events (When built-ins are not enough)**

**Custom Events**

* To create custom events use the CustomEvent() constructor

var event = new CustomEvent(eventType);

* Create custom event tripleclick

var event = new CustomEvent("tripleClick");

* Get body element to attach the custom event to   
  and use addEventListener

var body = document.getElementsByTagName("body")[0];

body.addEventListener("tripleClick", function() {

alert("You click three times");

}, false);