Χειρισμός Συμβάντων (Events) στην JavaScript

Τα HTML συμβάντα (events) είναι ενέργειες που συμβαίνουν σε συστατικά της HTML

Με την javaScript μπορούμε να γράψουμε κώδικα που να ανταποκρίνεται σε αυτά τα συμβάντα.

Τα ακόλουθα είναι ορισμένα παραδείγματα συμβάντων

Συμβάν	Περιγραφή
onchange	Συμβαίνει όταν τροποποιείται κάποιο στοιχείο της HTML
onclick	Συμβαίνει όταν ο χρήστης πατά σε ένα στοιχείο της HTML
onmouseover	Συμβαίνει όταν το ποντίκι μετακινείται πάνω σε ένα HTML στοιχείο
onmouseout	Συμβαίνει όταν το ποντίκι μετακινείται έξω από ένα HTML στοιχείο
onkeydown	Συμβαίνει όταν ο χρήστης πιέζει ένα πλήκτρο.
onload	Συμβαίνει όταν ο browser έχει ολοκληρώσει το φόρτωμα μιας σελίδας.

Τρόποι Χειρισμού Συμβάντων

Υπάρχουν τρεις βασικές μέθοδοι για τον χειρισμό συμβάντων δηλαδή για την σύνδεση τους με τον κώδικα JavaScript που θα εκτελεσθεί όταν συμβούν.

Ανάθεση χειρισμού συμβάντων μέσω ιδιοτήτων HTML

Αυτός ο τρόπος φαίνεται να είναι ο πιο απλός αλλά δεν συνίσταται σε σύγχρονες εφαρμογές γιατί δημιουργεί εξαρτήσεις μεταξύ του κώδικα HTML και του κώδικα JavaScript με αποτέλεσμα να δημιουργούνται δυσκολίες στην συντήρηση μιας εφαρμογής.

Τα στοιχεία HTML, που μπορούν να δημιουργήσουν συμβάντα περιέχουν ιδιότητες με ονόματα παρόμοια των συμβάντων. Στις ιδιότητες αυτές καταχωρείται ο κώδικας Javascript που θα κληθεί όταν δημιουργηθεί το αντίστοιχο συμβάν. Συνήθως μπορούν να περιέχουν την κλήση μια συνάρτησης Javascript η οποία ορίζεται σε κάποιο άλλο σημείο ή αρχείο.

```
<!DOCTYPE html>
<html>
<body>
```

```
<p>Click the button to display the date.</p>

<input type=”submit” onclick="displayDate()" value=”The time is?” >

<script>
function displayDate() {
   document.getElementById("demo").innerHTML = Date();
}
</script>

<p id="demo"></p>

</body>
</html>
```

Ανάθεση Χειρισμού Συμβάντων μέσω ιδιοτήτων DOM

Με την μέθοδο αυτή χρησιμοποιούνται ιδιότητες των αντικειμένων του μοντέλου DOM για τον καθορισμό του κώδικα που θα κληθεί όταν συμβεί το συμβάν. Συνήθως είναι το όνομα μιας συνάρτησης.

```
<!DOCTYPE html>
<html>
<body>

<p>Click "Try it" to execute the displayDate() function.</p>

<intput type=”submit”  id="myBtn" value=”Try it” >

<p id="demo"></p>

<script>
document.getElementById("myBtn").onclick = displayDate;

function displayDate() {
   document.getElementById("demo").innerHTML = Date();
}
</script>

</body>
</html>
```

Ένα μειονέκτημα της μεθόδου αυτής καθώς και της προηγούμενης είναι ότι δεν δίνει την δυνατότητα να καθοριστούν περισσότεροι από ένα χειριστές συμβάντων για κάποιο συστατικό HTML και για κάποιο συγκεκριμένο συμβάν. Κάτι τέτοιο μπορεί να περιορίσει την ευελιξία κατά την ανάπτυξη δυναμικών σελίδων. Αν υπάρξει ανάγκη επέκτασης της διαδικασίας χειρισμού συμβάντων θα πρέπει να τροποποιηθεί η μοναδική συνάρτηση που θα χειριστεί το συμβάν. Πολλές φορές είναι επιθυμητό να υπάρχει η δυνατότητα καθορισμού επιπρόσθετων χειριστών συμβάντων. Η επόμενη μέθοδος υποστηρίζει αυτή την δυνατότητα.

Ανάθεση χειρισμού συμβάντων σε ένα χειριστή συμβάντων (Event Listener)

Τα αντικείμενα που ανήκουν στο μοντέλο DOM διαθέτουν μια μέθοδο **addEventListener** την οποία μπορούμε να χρησιμοποιούμε για να προσθέτουμε συναρτήσεις που θα κληθούν όταν κάποιο γεγονός συμβεί. Σημείωση: Η δυνατότητα αυτή παρέχεται από όλους τους συγχρόνους φυλομετρητες αλλά δεν παρέχεται από την έκδοση του Internet Explorer. Για τις εκδόσεις IE 5-8 υπάρχει άλλη μέθοδος που επιτυγχάνει παρόμοια συμπεριφορά την οποία δεν θα καλύψουμε.

Στην γενική περίπτωση για την προσθήκη ενός χειριστή συμβάντων καλούμε την μέθοδο addEventListener παρέχοντας 2 ή 3 παραμέτρους.

Η πρώτη παράμετρος είναι μια συμβολοσειρά (string) και καθορίζει το συμβάν που θα χειριστούμε. Στην περίπτωση αυτή τα ονόματα που χρησιμοποιούνται για τα συμβάντα δεν περιέχουν το πρόθεμα “on” που περιέχεται στα ονόματα που είδαμε στις προηγούμενες περιπτώσεις. Για παράδειγμα για να προσθέσουμε ένα χειριστή συμβάντων για το γεγονός click θα χρησιμοποιήσουμε το όνομα click και όχι onclick.

Η δεύτερη παράμετρος μπορεί να περιέχει το όνομα της συνάρτησης που θα χειριστεί το συμβάν ή τον καθορισμό μιας ανώνυμης συνάρτησης. Η χρήση ανώνυμων συναρτήσεων θα περιγραφεί στην συνέχεια.

Η τρίτη προαιρετική παράμετρος είναι Boolean (true, false) και καθορίζει πότε θα κληθεί ο χειριστής του συμβάντος. Η προεπιλεγμένη τιμή είναι false και σημαίνει ότι θα κληθεί στην φάση ανάδυσης (bubbling) του συμβάντος. Τιμή true σημαίνει ότι θα κληθεί στην φάση σύλληψης (capturing). Οι δύο φάσεις θα περιγραφούν στην συνέχεια.

```
<!DOCTYPE html>
<html>
```

```
<body>

<p>This example uses the addEventListener() method to attach a click event to a button.</p>

<input type="text" id="myBtn" value='Try it'>

<p id="demo"></p>

<script>

document.getElementById("myBtn").addEventListener("click", displayDate);

function displayDate() {
   document.getElementById("demo").innerHTML = Date();
}
</script>

</body>
</html>
```

Πέρασμα Παραμέτρων σε ακροατή συμβάντων

Όταν προσθέτουμε έναν ακροατή συμβάντων (even listener) σε κάποιο συστατικό μπορούμε να τον προσδιορίσουμε με το όνομα μιας συνάρτησης που θέλουμε να κληθεί όταν συμβεί το γεγονός. Στην περίπτωση αυτή η συνάρτηση που θα δημιουργήσουμε μπορεί να μην περιέχει καμία παράμετρο. Αν θέλουμε να περάσουμε κάποιες τιμές σαν παραμέτρους στον χειριστή του συμβάντος θα πρέπει να χρησιμοποιήσουμε την δημιουργία μια ανώνυμης συνάρτησης η οποία θα καλεί τον χειριστή του συμβάντος.

Παράδειγμα

```
<!DOCTYPE html>
<html>
<body>

<p>This example demonstrates how to pass parameter values when using the
addEventListener() method.</p>

<p>Click the button to perform a calculation.</p>

<button id="myBtn">Try it</button>
```

```
<p id="demo"></p>

<script>
var p1 = 5;
var p2 = 7;

document.getElementById("myBtn").addEventListener("click", function() {
  myFunction(p1, p2);
});

function myFunction(a, b) {
  var result = a * b;
  document.getElementById("demo").innerHTML = result;
}
</script>

</body>
</html>
```

Σύλληψη (capturing) και Ανάδυση (bubbling) Συμβάντων

Στο μοντέλο DOM υπάρχει μια ιεραρχική σχέση μεταξύ των διάφορων συστατικών του λόγω του ότι κάποιο συστατικό περιέχεται και σε κάποιο άλλο. Με τον τρόπο αυτό κάποιο γεγονός που συμβαίνει σε ένα εσωτερικό αντικείμενο μπορούμε να θεωρήσουμε ότι συμβαίνει και στο περιέχον αντικείμενο. Στην περίπτωση αυτή θα μπορούσαν να κληθούν χειριστές συμβάντων και για τα δύο αντικείμενα αν υπάρχουν. Το θέμα είναι με ποια σειρά θα κληθούν.

Για τον καθορισμό της σειράς έχουν προσδιοριστεί δύο φάσεις μετάδοσης του συμβάντων. Η πρώτη φάση ξενικά την μετάδοση του συμβάντος από τη ανώτερη θέση της ιεραρχίας (πιο γενική) προς την κατώτερη (πιο συγκεκριμένη). Η φάση αυτή ονομάζεται σύλληψη του συμβάντος (event capturing).

Στην συνέχει το συμβάν αναδύεται από το κατώτερο συστατικό προς το ανώτερο συστατικό και στην φάση αυτή θα κληθούν οι ακροατές συμβάντων με αυτή την σειρά (για όσα συστατικά έχουν καθοριστεί ακροατές συμβάντων

Τις περισσότερες φορές είναι επιθυμητό ένας χειριστής συμβάντων να κληθεί μόνο είτε κατά την σύλληψη είτε κατά την ανάδυση του συμβάντος. Για το λόγο αυτό παρέχεται η τρίτη παράμετρος στην μέθοδο addEventListener. Όταν η τιμή της παραμέτρου είναι true ο χειριστής του συμβάντος καλείται στην φάση της σύλληψης (capturing) και όταν είναι false,

που είναι και η προεπιλεγμένη τιμή όταν δεν προσδιορίζεται η τρίτη παράμετρος, καλείται στην φάση της ανάδυσης (bubbling).

Παράδειγμα

```
 <!DOCTYPE html>
<html>
<head>
<style>
div {
   background-color: coral;
   border: 1px solid;
   padding: 50px;
}
</style>
</head>
<body>

<p>This example demonstrates the difference between bubbling and capturing when adding event listeners.</p>

<div id="myDiv">
  <p id="myP">Click this paragraph, I am Bubbling.</p>
</div><br>

<div id="myDiv2">
  <p id="myP2">Click this paragraph, I am Capturing.</p>
</div>

<script>
document.getElementById("myP").addEventListener("click", function() {
   alert("You clicked the P element!");
}, false);

document.getElementById("myDiv").addEventListener("click", function() {
   alert("You clicked the DIV element!");
}, false);

document.getElementById("myP2").addEventListener("click", function() {
   alert("You clicked the P element!");
}, true);
```

```
document.getElementById("myDiv2").addEventListener("click", function() {
   alert("You clicked the DIV element!");
}, true);
</script>

</body>
</html>
```

Ακύρωση ενός Συμβάντος

Πολλές φορές όπως στον έλεγχο φορμών είναι επιθυμητό να ακυρώσουμε την προκαθορισμένη δράση ενός συμβάντος (δηλ. το τι είναι σχεδιασμένος να κάνει ο φυλομετρητής σε ανταπόκριση του συμβάντος).

Για παράδειγμα όταν πατούμε ένα κουμπί τύπου submit σε μια φόρμα η προκαθορισμένη συμπεριφορά του φυλομετρητή είναι να στείλει κάπου (όπως καθορίζεται από την ιδιότητα action της φόρμας) μια αίτηση HTTP. Αυτή είναι η προεπιλεγμένη συμπεριφορά του συμβάντος submit.

Σε ορισμένες περιπτώσεις θέλουμε να ελέγξουμε τις τιμές μιας φόρμας και αν κάποια πεδία δεν έχουν αναμενόμενες τιμές να αποφύγουμε την υποβολή της. Αυτό μπορεί να συμβεί ακυρώνοντας την δράση του συμβάντος submit καλώντας την μέθοδο preventDefault στο αντικείμενο που αναπαριστά το συμβάν. Κάτι τέτοιο ισχύει για τα συμβάντα που έχουν την ιδιότητα «ακυρώσιμο» (cancelable). Υπάρχει και μια επιπλέον μέθοδος η stopPropagation που εμποδίζει και την μετάδοση του συμβάντος σε άλλα συστατικά.

Το παρακάτω είναι ένα παράδειγμα χειρισμού φόρμας όπου ένα αντικείμενο συμβάντος μπορεί να περαστεί κατά το χρόνο εκτέλεσης σαν η πρώτη παράμετρος στον ακροατή συμβάντων.

```
<!DOCTYPE html>
<html>
  <head><meta charset="UTF-8"></head>
<body>
<h1>JavaScript Can Validate Input</h1>
<p>Please input a number between 1 and 10:</p>

<form id="myform" action="http://example.com">
```

```
  <input id="numb">
  <input type="submit" id="mybtn" value="Submit">
  <p id="demo"></p>
</form>

<script>

document.getElementById("myform").addEventListener('submit',myFunction);

function myFunction(e) {
   var x, text;

   // Get the value of the input field with id="numb"
   x = document.getElementById("numb").value;

   // If x is Not a Number or less than one or greater than 10
   if (isNaN(x) || x < 1 || x > 10) {
      text = "Input not valid";
      document.getElementById("demo").innerHTML = text;
                  e.preventDefault();
   }
}
</script>
</body>
</html>
```

Διαγραφή ακροατή συμβάντων

Για να αφαιρέσουμε ένα χειριστή συμβάντων μπορούμε να χρησιμοποιήσουμε την ακόλουθη κλήση.

element.removeEventListener(*event*, *function*, *useCapture*)