Τ.Ε.Ι. ΠΑΤΡΩΝ

ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΟΙΚΟΝΟΜΙΑΣ

ΤΜΗΜΑ ΕΠΙΧΕΙΡΗΜΑΤΙΚΟΥ ΣΧΕΔΙΑΣΜΟΥ ΚΑΙ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΑΝΑΠΤΥΞΗ ΕΚΠΑΙΔΕΥΤΙΚΟΥ ΥΛΙΚΟΥ ΣΤΗΝ JAVASCRIPT

ΜΑΝΔΕΛΛΟΥ ΟΥΡΑΝΙΑ

ΠΟΥΛΑΚΗ ΚΩΝΣΤΑΝΤΙΝΑ

ΕΠΟΠΤΕΥΩΝ ΚΑΘΗΓΗΤΗΣ : Α. ΜΠΑΚΑΛΗΣ

ΠΑΤΡΑ 2009

ΠΕΡΙΕΧΟΜΕΝΑ

ΟΡΓΑΝΩΣΗ ΤΗΣ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Η πτυχιακή εργασία θα ακολουθήσει την παρακάτω δομή. Πρώτα θα γίνει μια αναφορά σε ιστορικά στοιχεία σχετικά με την JavaScript, τι είναι και ποιες οι βασικές γνώσεις που πρέπει να έχουμε για να αναπτύξουμε μια εφαρμογή χρησιμοποιώντας την JavaScript. Στο κεφάλαιο 2 γίνεται ανάλυση του πως δημιουργούμε ένα αρχείο JavaScript και πως αυτό μπορεί να χρησιμοποιηθεί σε συνδυασμό με τον κώδικα της HTML. Το δεύτερο μέρος της πτυχιακής περιέχει τα βασικά δομικά στοιχεία της JavaScript, τα οποία αναλύονται, το κάθε ένα, σε ξεχωριστό κεφάλαιο. Έτσι, τα δομικά στοιχεία που αναλύονται είναι τα ακόλουθα: οι μεταβλητές, οι συναρτήσεις, οι τελεστές, οι προτάσεις υπό όρους, τα αντικείμενα, οι πίνακες, οι συμβολοσειρές, οι χειριστές συμβάντων, οι φόρμες και τα πλαίσια. Το κεφάλαιο 13 περιέχει μια σειρά από παραδείγματα – εφαρμογές της JavaScript και τελικά στο κεφάλαιο 14 αναλύεται η τεχνική AJAX, με την βοήθεια της οποίας η JavaScript μπορεί να επικοινωνεί κατευθείαν με τον server κάνοντας έτσι της εφαρμογές καλύτερες, ταχύτερες και περισσότερο αλληλεπιδραστικές.

ΠΡΟΛΟΓΟΣ

Σε μια εποχή όπου η χρήση του Διαδικτύου έχει εξαπλωθεί σε μεγάλο βαθμό, όλο και μεγαλύτερη γίνεται η ανάγκη στους προγραμματιστές να δημιουργούν ιστοσελίδες που αλληλεπιδρούν με τον χρήστη και τον προσελκύουν να επισκεφτεί τον δικτυακό τους τόπο. Στα πλαίσια της πτυχιακής μας εργασίας θα γίνει μια προσπάθεια γνωριμίας με την γλώσσα προγραμματισμού JavaScript, μια γλώσσα scripting που χρησιμοποιείται για δημιουργία ιστοσελίδων. Είναι ένα εργαλείο που χρησιμοποιείται από τους προγραμματιστές, ώστε να γίνονται οι ιστοσελίδες πιο διαλογικές, να τροποποιούνται τα δεδομένα τους και να αλληλεπιδρούν με τον χρήστη κάνοντας τις πιο εύκολες στην πλοήγηση.

Πριν αρχίσουμε να αναλύουμε περισσότερο τα δομικά στοιχεία της JavaScript και να δίνουμε παραδείγματα εφαρμογών της, καλό θα ήταν να κατανοήσουμε το τι είναι μια γλώσσα προγραμματισμού σαν την JavaScript και που μπορεί να μας φανεί χρήσιμη.

Όπως ήδη αναφέραμε η JavaScript είναι μια γλώσσα scripting, δηλαδή μια γλώσσα προγραμματισμού που επιτρέπει τον έλεγχο ενός ή περισσοτέρων εφαρμογών λογισμικού. Τα “scripts”, δηλαδή τα “σενάρια”, είναι διαφορετικά από τον κύριο κώδικα της εφαρμογής, η οποία είναι γραμμένη συνήθως σε διαφορετική γλώσσα, και συχνά δημιουργούνται ή τουλάχιστον τροποποιούνται από τον τελικό χρήστη.

Η JavaScript χρησιμοποιείται για να επιτρέψει την πρόσβαση μέσω προγραμματισμού σε αντικείμενα τόσο από την μεριά του χρήστη της εφαρμογής όσο και από άλλες εφαρμογές. Χρησιμοποιείται κυρίως από την μεριά του χρήστη και εφαρμόζεται επιτρέποντας την ανάπτυξη ενισχυμένων διεπαφών χρήστη και δυναμικών ιστοσελίδων. Επιπλέον, η JavaScript επηρεάστηκε από πολλές γλώσσες και έχει σχεδιαστεί να μοιάζει με την Java, αλλά να είναι ευκολότερη στη χρήση και από μη προγραμματιστές.

ΜΕΡΟΣ 1: ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΕΝΝΟΙΑ ΤΗΣ ΣΥΝΤΑΞΗΣ ΚΩΔΙΚΑ WEB SCRIPT ΚΑΙ Η ΓΛΩΣΣΑ JAVASCRIPT

ΚΕΦΑΛΑΙΟ 1: ΕΙΣΑΓΩΓΗ ΣΤΗΝ JAVASCRIPT

1.1 Εισαγωγή στην JavaScript

Η JavaScript αναπτύχθηκε αρχικά από τον Brendan Eich της Netscape με το όνομα Mocha, το οποίο αργότερα μετονομάστηκε σε LiveScript και τελικά σε JavaScript. Η αλλαγή του ονόματος από LiveScript σε JavaScript συνέπεσε περίπου χρονικά με την προσθήκη από την Netscape υποστήριξης τεχνολογίας Java σε web browsers (πρόγραμμα περιήγησης) της Netscape.

Η JavaScript εισήχθη για πρώτη φορά και αναπτύχθηκε στην έκδοση του προγράμματος περιήγησης της Netscape 2.0B3 τον Δεκέμβριο του 1995. Η ονομασία προκάλεσε σύγχυση, αφού συνδέθηκε από πολλούς με την Java, και έχει χαρακτηριστεί από πολλούς ως ένα τέχνασμα μάρκετινγκ από την Netscape, για να δώσει στην JavaScript την σφραγίδα της νέας τότε γλώσσας προγραμματισμού Java.

1.2 Τι είναι η JavaScript

Όπως έχουμε ήδη αναφέρει, η JavaScript είναι μια γλώσσα script βασισμένη σε αντικείμενα από την πλευρά του πελάτη (client-side), που μπορούμε να την χρησιμοποιήσουμε για να κάνουμε τις ιστοσελίδες μας πιο δυναμικές. Για να έχει έννοια ένας τέτοιος ορισμός θα εξετάσουμε τα πιο σημαντικά μέρη της ένα προς ένα.

Το "βασισμένη σε αντικείμενα" σημαίνει ότι η JavaScript μπορεί να χρησιμοποιήσει στοιχεία που ονομάζονται αντικείμενα (objects). Ωστόσο, τα αντικείμενα βασίζονται σε κλάσεις (class) (που σημαίνει ότι δεν γίνεται καμία διάκριση μεταξύ μιας κλάσης και ενός στιγμιότυπου). Αντίθετα, είναι απλώς γενικά αντικείμενα.

Η "πλευρά του πελάτη" (client side) σημαίνει ότι η JavaScript τρέχει στον πελάτη (στο λογισμικό) που χρησιμοποιεί ο επισκέπτης και όχι στον Web διακομιστή της τοποθεσίας που εξυπηρετεί τη σελίδα. Σ' αυτήν την περίπτωση, ο πελάτης της θα ήταν ένας Web browser.

Με τη JavaScript μπορούμε να φτιάξουμε σενάρια που να εκτελούν αυτόματες εργασίες, π.χ. όταν μια σελίδα του Web ανοίγει ή κλείνει. Επίσης μπορούμε να κάνουμε την JavaScript να εκτελεί ενέργειες ανταποκρινόμενη σε ένα συγκεκριμένο γεγονός. Για παράδειγμα όταν ο χρήστης επιλέγει ένα κουμπί ή ένα σύνδεσμο, όταν εστιάζει από ένα στοιχείο μιας φόρμας σε ένα άλλο στοιχείο της κοκ.

Οι ενέργειες αυτές μπορεί να είναι απλές. Τα σενάρια μπορεί να ανοίγουν νέα παράθυρα στον browser και να εμφανίζουν συγκεκριμένα HTML έγγραφα ή να παρουσιάζουν μια σελίδα επιλεγμένη από τον κατάλογο ιστορικού του browser. Μπορεί επίσης να είναι και περίπλοκες δηλαδή ένα σενάριο μπορεί να ελέγχει τα περιεχόμενα μιας φόρμας που θέλει να υποβάλει ένας χρήστης και στη συνέχεια να προειδοποιεί τον χρήστη αν τα δεδομένα είναι λάθος. Το σενάριο μπορεί να ψάξει για πληροφορίες σε μια μικρή βάση δεδομένων ή να κάνει πολύπλοκους υπολογισμούς οικονομικών στοιχείων.

Εδώ υπάρχουν μερικά πράγματα που μπορούμε να κάνουμε με την JavaScript:

- Να εμφανίζουμε μηνύματα στο χρήστη ως μέρος μιας ιστοσελίδας, στη γραμμή κατάστασης του browser ή σε πλαίσια προειδοποίησης
- Να επικυρώνουμε το περιεχόμενο μιας φόρμας και να στέλνουμε τα δεδομένα μιας φόρμας με το ηλεκτρονικό ταχυδρομείο.
- Να δημιουργήσουμε κινούμενες εικόνες ή να δημιουργήσουμε εικόνες που αλλάζουν όταν μετακινείτε το ποντίκι πάνω τους

- Να εντοπίζουμε τον browser του χρήστη ή τις λειτουργίες του και να εκτελούμε προχωρημένες λειτουργίες μόνο στους browser που τις υποστηρίζουν

- Να τροποποιούμε ολόκληρο ή μέρος μιας ιστοσελίδας χωρίς να πρέπει ο χρήστης να την ξαναφορτώσει

- Να εμφανίζουμε ή να αλληλεπιδρούμε με τα δεδομένα που ανακτώνται από έναν απομακρυσμένο διακομιστή.[3]

1.3 Τι πρέπει να γνωρίζετε για την JavaScript

Η JavaScript τρέχει στον browser αφού προστεθεί κατευθείαν σε ένα υπάρχον HTML έγγραφο. Μπορούμε να προσθέσουμε ειδικές ετικέτες και εντολές κώδικα HTML οι οποίες "λένε" στον browser ότι πρέπει να τρέξει ένα script. Μόλις δει ο browser αυτές τις ειδικές ετικέτες, διερμηνεύει τις JavaScript εντολές και θα κάνει αυτό που του "λέμε" να κάνει με τον κώδικα μας. Έτσι, τροποποιώντας απλώς ένα HTML έγγραφο, μπορούμε να αρχίσουμε να χρησιμοποιούμε την JavaScript στις ιστοσελίδες μας και να βλέπουμε τα αποτελέσματα.

Πριν αρχίσουμε να μαθαίνουμε την JavaScript, πρέπει να έχουμε μια βασική γνώση των εξής:

• HTML: απαιτείται μια βασική γνώση των ετικετών head και body και η ικανότητα προσθήκης μιας ιδιότητας στις ετικέτες

• Επεξεργαστές κειμένου: θα χρησιμοποιηθούν για να συνταχθεί ο κώδικας

• Web browser: θα χρειαστεί για να ελεγχθούν τα script. Απαιτείται ο web browser να υποστηρίζει JavaScript.

• Τις διαφορετικές εκδόσεις της JavaScript: Η γλώσσα JavaScript έχει εξελιχθεί από τότε που εμφανίστηκε αρχικά στο Netscape 2.0. Από τότε έχουν

υπάρξει διάφορες εκδόσεις της JavaScript. Κάθε μια από αυτές τις εκδόσεις είναι μια βελτίωση ως προς την προηγούμενη έκδοση και περιλαμβάνει διάφορες νέες λειτουργίες. Με σπάνιες εξαιρέσεις, οι browser που υποστηρίζουν τη νέα έκδοση θα υποστηρίζουν επίσης τα script που έχουν γραφεί για τις προηγούμενες εκδόσεις. Για το 2009, η τελευταία έκδοση της γλώσσας είναι η JavaScript 1.8.1.

1.4 Υποστήριξη browser που δεν υποστηρίζουν JavaScript

Μερικοί επισκέπτες της τοποθεσίας μας θα χρησιμοποιούν browser που δεν υποστηρίζουν καθόλου JavaScript. Τόσο ο Internet Explorer όσο και το Firefox περιλαμβάνουν μια επιλογή με την οποία απενεργοποιούν την JavaScript και μερικοί χρήστες το κάνουν αυτό.

Επομένως, δεν μπορούμε να υποθέσουμε ότι όλοι οι επισκέπτες θα υποστηρίζουν JavaScript. Μπορούμε να χρησιμοποιήσουμε τεχνικές για να διασφαλίσουμε ότι αυτοί οι χρήστες μπορούν να χρησιμοποιήσουν την τοποθεσία μας.

1.4.1 Χρησιμοποιώντας την ετικέτα <noscript>

Ένας τρόπος να είμαστε φιλικοί σε browser που δεν υποστηρίζουν την JavaScript είναι να χρησιμοποιήσουμε την ετικέτα <noscript>. Αυτή η ετικέτα, που υποστηρίζεται στους περισσότερους σύγχρονους browser, εμφανίζει ένα μήνυμα σε browser που δεν υποστηρίζουν την JavaScript. Οι browser που υποστηρίζουν JavaScript αγνοούν το κείμενο μεταξύ των ετικετών <noscript>, ενώ άλλοι το εμφανίζουν. Εδώ βλέπουμε ένα απλό παράδειγμα:

```
<noscript>

This page requires JavaScript. You can either switch to a browser that supports JavaScript, turn your browser's script support on, or switch to the <a href="nojs.html">Non-JavaScript</a> version of this page.

</noscript>
```

Αν και αυτό δουλεύει, το πρόβλημα είναι ότι η ετικέτα <noscript> δεν υποστηρίζεται με συνέπεια από όλους τους browser που υποστηρίζουν JavaScript. Μια εναλλακτική λύση που αποφεύγει την ετικέτα <noscript> είναι να στέλνονται οι χρήστες που υποστηρίζουν την JavaScript σε μια άλλη σελίδα. Αυτό μπορεί να επιτευχθεί με μια μόνο πρόταση JavaScript:

```
<script language="JavaScript" type="text/javascript">
window.location="JavaScript.html";
</script>
```

Αυτό το script ανακατευθύνει το χρήστη σε μια διαφορετική σελίδα. Εάν ο browser δεν υποστηρίζει JavaScript, φυσικά, το script δεν θα εκτελεσθεί και η υπόλοιπη σελίδα μπορεί να εμφανίσει ένα μήνυμα προειδοποίησης για να εξηγήσει την κατάσταση.

1.4.2 Κρύβοντας τον JavaScript Κώδικα Από Τους Παλαιότερους Browser

Εάν κάποιος επισκέπτεται τη σελίδα μας με έναν παλαιότερο browser, που δεν υποστηρίζει JavaScript, μπορεί ολόκληρος ο JavaScript κώδικας να εμφανισθεί στη σελίδα ως απλό κείμενο. Για να μην γίνει αυτό, υπάρχει δυνατότητα από τον browser να αγνοήσει αυτά που ακολουθούν τοποθετώντας HTML σχόλια μεταξύ των ετικετών αρχής και τέλους script.

Μετά από την ετικέτα αρχής script, εισάγουμε τον κωδικό αρχής για τα HTML σχόλια (<!--). Ακριβώς πριν από την ετικέτα τέλους script στο έγγραφο, τοποθετούμε τον κωδικό τέλους για τα HTML σχόλια, με δύο αρχικές καθέτους (//-->).

```
<script language="JavaScript">
<! --
O JavaScript μπαίνει εδώ
/ /-- >
```

</script>

1.4.3 Διατηρώντας προαιρετική την JavaScript

Αν και μπορούμε να εντοπίσουμε τους JavaScript browser και να εμφανίσουμε ένα μήνυμα στους υπόλοιπους, η καλύτερη επιλογή είναι να κάνουμε απλώς τα script μας διακριτικά. Πρέπει να χρησιμοποιήσουμε την JavaScript για να βελτιώσουμε την τοποθεσία αντί να παρέχουμε ουσιαστικές λειτουργίες. Μπορούμε να βάλουμε την JavaScript σε ξεχωριστά αρχεία, να αντιστοιχίσουμε χειριστές συμβάντων στο JavaScript αρχείο αντί στο HTML και οι browser, που δεν υποστηρίζουν JavaScript απλώς θα αγνοήσουν το script.

Σε αυτές τις σπάνιες περιπτώσεις όπου χρειαζόμαστε απολύτως την JavaScript, για παράδειγμα, μια εφαρμογή AJAX ή ένα JavaScript παιχνίδι, μπορούμε να προειδοποιήσουμε τους χρήστες ότι η JavaScript είναι απαραίτητη. Ωστόσο, είναι καλύτερα να προσφέρουμε ένα εναλλακτικό τρόπο χωρίς χρήση JavaScript ώστε να χρησιμοποιηθεί η τοποθεσία, ειδικά εάν είναι μια τοποθεσία ηλεκτρονικού εμπορίου ή επαγγελματική τοποθεσία, πάνω στην οποία βασίζεται η επιχείρηση μας.

ΚΕΦΑΛΑΙΟ 2: ΕΝΣΩΜΑΤΩΣΗ JAVASCRIPT ΚΩΔΙΚΑ ΣΕ ΕΝΑ HTML ΑΡΧΕΙΟ

2.1 Χρήση Ετικετών HTML SCRIPT

Οι ετικέτες script "λένε" στον browser που αρχίζει και που τελειώνει κάποιος τύπος γλώσσας script σε ένα HTML έγγραφο. Στην πιο απλή μορφή τους, οι ετικέτες script εμφανίζονται ακριβώς όπως οποιοδήποτε άλλο σύνολο ετικετών HTML:

< script >

Ο JavaScript κώδικας μπαίνει εδώ

</ script >

Όπως φαίνεται, υπάρχει μια ετικέτα αρχής <script>, ο κώδικας JavaScript και έπειτα η ετικέτα τέλους </script>. Όταν χρησιμοποιούνται οι απλές ετικέτες αρχής και τέλους όπως εδώ, πολλοί browser θα υποθέσουν ότι η γλώσσα script που ακολουθεί θα είναι η JavaScript. Ωστόσο, σε μερικούς browser ίσως να πρέπει να αναφερθεί ποια γλώσσα script χρησιμοποιούμε.

Εκτός από το να δίνουν αναφορά στον browser πού αρχίζει και που τελειώνει ένα script, οι ετικέτες script μπορούν επίσης να "πουν" στον browser ποια γλώσσα script θα χρησιμοποιηθεί και να ορίσουμε τη διεύθυνση ενός εξωτερικού αρχείου JavaScript. Αυτές οι πρόσθετες λειτουργίες επιτυγχάνονται μέσω της χρήσης των ιδιοτήτων src (source - προέλευση).

2.2 Κλήση Εξωτερικών Script

Οι ετικέτες script είναι επίσης χρήσιμες εάν θέλουμε να καλέσουμε ένα εξωτερικό αρχείο JavaScript στο έγγραφο μας. Ένα εξωτερικό αρχείο JavaScript είναι ένα αρχείο κειμένου, που περιέχει μόνο κώδικα JavaScript και έχει αποθηκευτεί με την επέκταση .js. Με την κλήση ενός εξωτερικού αρχείου, εξοικονομείται χρόνος κωδικοποίησης μεγάλων γραμμών script σε κάθε σελίδα στην οποία απαιτείται το script.

Μπορούμε να καλέσουμε εξωτερικά script προσθέτοντας μια ιδιότητα src (source - προέλευση) στην ετικέτα script αρχής:

< script language="JavaScript" src="yourfile.js">

</ script >

Εάν το script είναι πολύ μεγάλο, μπορεί να είναι πολύ γρηγορότερο να χρησιμοποιηθεί η ιδιότητα src για να προσθέσει το script σε πολλές σελίδες από το να εισάγουμε ολόκληρο τον κώδικα σε κάθε σελίδα.

2.3 Ποιο Script Τρέχει Πρώτα

Μπορούμε στην πραγματικότητα να έχουμε διάφορα script μέσα σε ένα Web έγγραφο: ένα ή περισσότερα σύνολα ετικετών <script>, εξωτερικά αρχεία JavaScript και οποιονδήποτε αριθμό χειριστών συμβάντων. Η σειρά με την οποία εκτελούνται τα script από τον browser γίνεται με ένα λογικό τρόπο:

- Τα σύνολα των ετικετών <script> μέσα στην ενότητα <head> ενός HTML εγγράφου τρέχουν πρώτα, είτε περιλαμβάνουν ενσωματωμένο κώδικα είτε αναφέρονται σε ένα JavaScript αρχείο. Επειδή αυτά τα script δεν μπορούν να δημιουργήσουν εξόδους στην ιστοσελίδα, είναι ένα καλό μέρος να ορίζουμε συναρτήσεις για μετέπειτα χρήση.
- Τα σύνολα των ετικετών <script> μέσα στην ενότητα <body> του HTML εγγράφου εκτελούνται μετά από αυτά στην ενότητα <head>, ενώ φορτώνεται και εμφανίζεται η ιστοσελίδα. Εάν υπάρχουν περισσότερα από ένα script, εκτελούνται με την σειρά.
- Οι χειριστές συμβάντος εκτελούνται όταν λαμβάνουν χώρα τα συμβάντα τους. Για παράδειγμα, ο χειριστής συμβάντος onLoad εκτελείται όταν φορτώνεται το σώμα μιας ιστοσελίδας. Επειδή η ενότητα <head> φορτώνεται πριν από οποιαδήποτε συμβάντα, μπορούμε να ορίσουμε τις συναρτήσεις μας εκεί και να τις χρησιμοποιήσουμε στους χειριστές συμβάντων.

2.4 Δημιουργία ενός Αρχείου JavaScript

Για να εισάγουμε ένα JavaScript αρχείο σε μια σελίδα HTML χρησιμοποιούμε την ετικέτα <script>. Μέσα στην ετικέτα <script> χρησιμοποιούμε το χαρακτηριστικό του τύπου (type) για τον καθορισμό της γλώσσας scripting. Έτσι, οι ετικέτες <script type="text/javascript"> και </ script> δηλώνουν πότε αρχίζει και τελειώνει το JavaScript: Η λέξη document.write είναι μια τυπική εντολή της JavaScript για τη σύνταξη κειμένου σε μια σελίδα. Εισάγοντας την εντολή document.write μεταξύ των ετικετών <script> και </ script>, ο φυλλομετρητής θα την αναγνωρίσει ως εντολή JavaScript και θα εκτελέσει την γραμμή κώδικα. Σε αυτή την περίπτωση ο φυλλομετρητής θα γράψει Hello World! στη σελίδα. Παρακάτω δίνεται ο κώδικας για το συγκεκριμένο παράδειγμα:

```
<html>
<body>
<script type="text/javascript">
document.write("Hello World!");
</script>
</body>
</html>
```

2.5 Χρήση Σχολίων στην JavaScript

Χρησιμοποιούμε σχόλια στην JavaScript όταν πρέπει να γίνουν κάποιες σημειώσεις στον κώδικα JavaScript, όπως μια περιγραφή στο τι κάνει μια γραμμή κώδικα. Είναι, επίσης, πιθανόν να θέλουμε να απενεργοποιήσουμε μια γραμμή του script για κάποιο λόγο.

Εάν θέλουμε να προσθέσουμε σχόλια σε μια μόνο γραμμή στον κώδικα μας, τοποθετούμε ένα ζευγάρι καθέτων πριν από το κείμενο του σχολίου:

```
// Τα σχόλια σας μπαίνουν εδώ
```

Σε αυτήν τη μορφή, οτιδήποτε υπάρχει πριν από τις δύο καθέτους σε αυτή την γραμμή είναι "ενεργός" κώδικας, δηλαδή, κώδικας που δεν θα εκτελέσει οτιδήποτε υπάρχει μετά τις δύο καθέτους.

Για να προστεθούν σχόλια που εκτείνονται σε πολλές γραμμές, χρησιμοποιούμε μια διαφορετική μορφή σχολίου: τοποθετούμε μια κάθετο που ακολουθείται από έναν αστερίσκο στην αρχή του σχολίου, έπειτα βάζουμε το κείμενο του σχολίου και έπειτα έναν αστερίσκο, που ακολουθείται από μια κάθετο στο τέλος του σχολίου. Παρακάτω δίνεται ένα παράδειγμα:

```
/* το script μου θα γράψει κάποιο κείμενο στο HTMLέγγραφό μου!

Όλο το κείμενο αγνοείται από τον browser.

*/

document.write ("You can see me!");
```

Χρησιμοποιώντας αυτήν τη μορφή, μπορούμε να ξεκινήσουμε το σχόλιο σε μια γραμμή και να το τελειώσουμε σε μια άλλη γραμμή.

ΜΕΡΟΣ 2: ΤΑ ΒΑΣΙΚΑ ΣΤΟΙΧΕΙΑ ΤΗΣ JAVASCRIPT

ΚΕΦΑΛΑΙΟ 3: Η ΧΡΗΣΗ ΤΩΝ ΜΕΤΑΒΛΗΤΩΝ

3.1 Τι είναι οι μεταβλητές

Μια μεταβλητή (variable) αντιπροσωπεύει ή περιέχει μια τιμή. Η πραγματική τιμή μιας μεταβλητής μπορεί να αλλάξει οποιαδήποτε στιγμή.

Οι μεταβλητές της JavaScript είναι παρόμοιες με αυτές που χρησιμοποιούνται στα μαθηματικά. Δίνεται σε μια μεταβλητή ένα όνομα και έπειτα της εκχωρούνται τιμές ανάλογα με τις ανάγκες της εργασίας. Εάν η τιμή της μεταβλητής αλλάξει, θα αλλάξει κάτι που συμβαίνει μέσα στο script.

Η χρήση μεταβλητών προσφέρει δύο πλεονεκτήματα. Πρώτων μπορούν να εξοικονομήσουν χρόνο σύνταξης και ενημέρωσης των script επειδή οι τιμές τους μπορούν να αλλάζουν. Όταν εκχωρείται μια τιμή σε μια μεταβλητή στην αρχή ενός script το υπόλοιπο μπορεί απλώς να χρησιμοποιήσει τη μεταβλητή. Εάν αλλάξει αργότερα η τιμή, θα πρέπει να αλλάξει ο κώδικας μόνο σε ένα μέρος - εκεί όπου εκχωρήθηκε η τιμή στη μεταβλητή - αντί να τον αλλάζει σε πολλά μέρη. Επιπλέον μπορούν να κάνουν πιο σαφή το σκοπό του κώδικα. Αφού οι μεταβλητές αντιπροσωπεύουν κάτι και μπορεί να τους δοθεί ένα όνομα που να έχει κάποια έννοια, είναι συνήθως ευκολότερο να καταλάβουμε τα script όταν τα διαβάζουμε και τα αποσφαλματώνουμε.

3.2 Δήλωση μεταβλητών

Οι μεταβλητές δημιουργούνται όταν τις δηλώνουμε. Κατόπιν τους εκχωρούνται τιμές χρησιμοποιώντας τον τελεστή εκχώρησης (assignment operator) της JavaScript και ονομάζονται ακολουθώντας τους κανόνες ονομασίας των μεταβλητών της JavaScript δίνοντάς τους όσο γίνεται πιο λογικά ονόματα.

Για να δηλωθεί κάποιο κείμενο ως μεταβλητή, χρησιμοποιείται η λέξη κλειδί var, η οποία "λέει" στον browser ότι το κείμενο που ακολουθεί είναι το όνομα μιας νέας μεταβλητής:

var variablename

Ο κώδικας της απόδοσης ενός ονόματος σε μια μεταβλητή είναι απλός αλλά υπάρχουν μερικοί περιορισμοί στις "λέξεις" που μπορούν να χρησιμοποιηθούν για μεταβλητές, όπως επίσης υπάρχει μια ευαισθησία στην εναλλαγή μεταξύ κεφαλαίων και πεζών γραμμάτων.

Για να εκχωρηθεί μια τιμή σε μια μεταβλητή, χρησιμοποιείται ο τελεστής εκχώρησης της JavaScript, που είναι το σύμβολο ίσον (=). Για να δηλωθεί μια μεταβλητή και να της εκχωρηθεί μια τιμή στην ίδια γραμμή, χρησιμοποιούμε αυτήν τη μορφή:

var variablename = variablevalue;

3.3 Ονομασία μεταβλητών

Πριν αρχίσουμε να ονομάζουμε τις μεταβλητές, πρέπει να γνωρίζουμε τους κανόνες ονομασίας της JavaScript. Οι παράγοντες που πρέπει να εξετάσουμε όταν επιλέγουμε ονόματα είναι η ευαισθησία σε κεφαλαία και πεζά, οι άκυροι χαρακτήρες και τα ονόματα που είναι δεσμευμένα από την JavaScript. Επιπλέον πρέπει να προσπαθούμε να δίνουμε στις μεταβλητές ονόματα, που είναι εύκολα να τα θυμόμαστε και λογικά.

3.3.1 Χρησιμοποιώντας κεφαλαία και πεζά στις μεταβλητές

Οι μεταβλητές της JavaScript είναι ευαίσθητες σε κεφαλαία και πεζά. Οι paycheck, PAYCHECK, Paycheck και PaYcHeCk είναι τέσσερις διαφορετικές μεταβλητές. Όταν δημιουργούμε μια μεταβλητή, πρέπει να χρησιμοποιούμε τον ίδιο συνδυασμό κεφαλαίων-πεζών όταν γράφουμε το όνομα αυτής της μεταβλητής και αργότερα μέσα στο script. Εάν αλλάξουμε τα κεφαλαία-πεζά, η

JavaScript θεωρεί ότι αυτή είναι μια νέα μεταβλητή ή επιστρέφει ένα λάθος. Όπως και να είναι, αυτό μπορεί να προκαλέσει προβλήματα στο script.

3.3.2 Χρησιμοποιώντας επιτρεπόμενους χαρακτήρες

Ένας σημαντικός κανόνας που πρέπει να θυμόμαστε είναι ότι ένα όνομα μεταβλητής πρέπει να ξεκινά με ένα γράμμα ή με τον χαρακτήρα υπογράμμισης (_). Το όνομα της μεταβλητής δεν μπορεί να ξεκινά με έναν αριθμό ή οποιονδήποτε άλλο χαρακτήρα, ο οποίος δεν είναι γράμμα (εκτός από τον χαρακτήρα υπογράμμισης). Οι άλλοι χαρακτήρες στο όνομα της μεταβλητής μπορεί να είναι γράμματα, αριθμοί, ή χαρακτήρες υπογράμμισης. Δεν επιτρέπονται κενά στα ονόματα μεταβλητών.

3.3.3 Αποφεύγοντας δεσμευμένες λέξεις

Ένας άλλος κανόνας, που πρέπει να θυμόμαστε όταν ονομάζουμε τις μεταβλητές είναι να αποφεύγουμε τις δεσμευμένες λέξεις της JavaScript. Αυτές είναι ειδικές λέξεις που χρησιμοποιούνται για έναν συγκεκριμένο σκοπό στην JavaScript. Για παράδειγμα, η δεσμευμένη λέξη var χρησιμοποιείται για να δηλώνει μια μεταβλητή στην JavaScript. Η χρησιμοποίηση της ως όνομα μεταβλητής μπορεί να προκαλέσει πολλά προβλήματα στο script, αφού αυτή η λέξη είναι προορισμένη να χρησιμοποιείται με διαφορετικό τρόπο.

Ο παρακάτω πίνακας 1 δείχνει τις δεσμευμένες λέξεις της JavaScript. Όλες αυτές οι λέξεις είναι σε πεζά γράμματα.

abstract	delete	function	null	throw
boolean	do	goto	package	throws
break	double	if	private	transient
byte	else	implements	protected	true

case	enum	import	public	try
catch	export	in	return	typeof
char	extends	instanceof	short	var
class	false	int	static	void
continue	finally	long	switch	while
debugger	float	native	synchronized	with
const	final	interface	super	volatile
default	for	new	this	-

ΠΙΝΑΚΑΣ 1 Οι δεσμευμένες λέξεις της JavaScript

3.3.4 Δίνοντας στις μεταβλητές λογικά ονόματα

Αν και το x είναι ένα αποδεκτό όνομα για μια μεταβλητή, είναι πιθανό να μην μπορούμε να θυμηθούμε τι αντιπροσωπεύει εάν πρέπει να διορθώσουμε το πρόγραμμα αργότερα. Επίσης, εάν κάποιος άλλος προσπαθήσει να μας βοηθήσει να διορθώσουμε τον κώδικα, η δουλειά του θα είναι ακόμα πιο δύσκολη.

Πρέπει να προσπαθούμε να δίνουμε ονόματα στις μεταβλητές που περιγράφουν αυτό που αντιπροσωπεύουν όσο το δυνατόν με μεγαλύτερη σαφήνεια. Υποθέτουμε ότι θέλουμε να χρησιμοποιήσουμε μια μεταβλητή για έναν αριθμό ενός παραδείγματος σε μια σελίδα. Αντί να χρησιμοποιήσουμε το x, ex ή μια άλλη σύντομη μεταβλητή, χρησιμοποιούμε κάτι πιο περιγραφικό:

```
var example_number=2;
```

Η μεταβλητή example_number θα είναι πιο εύκολο να αναγνωριστεί αργότερα και οι άλλοι προγραμματιστές θα είναι πιο πιθανό να καταλάβουν γρήγορα τη χρήση της.

Όσο περισσότερες μεταβλητές χρησιμοποιούμε σε ένα script, τόσο πιο σημαντικό είναι να χρησιμοποιούμε περιγραφικά και ευκολομνημόνευτα ονόματα.

3.4 Τύποι μεταβλητών

Στην JavaScript, οι τιμές των μεταβλητών ή τύποι των μεταβλητών, μπορεί να συμπεριλαμβάνουν αριθμούς, συμβολοσειρές, λογικές (Booleans) και null.

Αντίθετα από τις πιο αυστηρές γλώσσες προγραμματισμού, η JavaScript δεν μας αναγκάζει να δηλώνουμε τον τύπο της μεταβλητής όταν την ορίζουμε. Αντίθετα, η JavaScript επιτρέπει να εκχωρηθεί οποιαδήποτε τιμή σε οποιαδήποτε μεταβλητή. Αν και αυτό μας δίνει ευελιξία κατά τον προγραμματισμό, πρέπει να είμαστε προσεκτικοί επειδή μπορεί να καταλήξουμε με μερικά αναπάντεχα αποτελέσματα - ειδικά κατά την πρόσθεση αριθμών.

3.4.1 Αριθμοί

Οι μεταβλητές με αριθμητικό τύπο είναι απλώς αυτό, δηλαδή είναι αριθμοί. Η JavaScript δεν απαιτεί να δηλωθούν οι αριθμοί ως ακέραιοι, κινητής υποδιαστολής (floating-point - δεκαδικοί) ή οτιδήποτε άλλο. Αντίθετα, όλοι οι αριθμοί θεωρούνται ίδιοι, είτε είναι το 7, το -2, το 3.453 ή κάτι άλλο. Ο αριθμός θα παραμείνει στον ίδιο τύπο, εκτός και αν εκτελέσουμε έναν υπολογισμό για να αλλάξουμε τον τύπο. Για παράδειγμα, εάν χρησιμοποιήσουμε έναν ακέραιο αριθμό σε μια μεταβλητή, δεν θα έχει δεκαδικά ψηφία εκτός και αν εκτελέσουμε κάποιον υπολογισμό για να το αλλάξουμε αυτό (για παράδειγμα, μια μη ακέραιη διαίρεση).

Ορίζουμε μια μεταβλητή με αριθμητικό τύπο χρησιμοποιώντας τη λέξη κλειδί var:

var variablename= number;

Εδώ βλέπουμε μερικά παραδείγματα:

- *var paycheck=1200;*
- *var phonebill=29.99;*
- *var savings=0;*
- *var sparetime=-24.5;*

3.4.2 Συμβολοσειρές

Οι μεταβλητές τύπου συμβολοσειράς (string) είναι μεταβλητές που αντιπροσωπεύουν μια συμβολοσειρά κειμένου. Η συμβολοσειρά μπορεί να περιέχει γράμματα, λέξεις, κενά, αριθμούς, σύμβολα ή σχεδόν οτιδήποτε θέλουμε. Οι συμβολοσειρές ορίζονται με έναν λίγο διαφορετικό τρόπο από τους αριθμούς, χρησιμοποιώντας αυτήν τη μορφή:

var variablename=" stringtext";

Παρακάτω δίνονται μερικά παραδείγματα μεταβλητών τύπου συμβολοσειράς:

- *var mycar="Corvette";*
- *var oldcar="Big Brown Station Wagon";*

Μπορούμε να τοποθετήσουμε όλα τα είδη του κειμένου και άλλων χαρακτήρων μέσα σε μεταβλητές συμβολοσειρών. Ωστόσο, πρέπει να λάβουμε υπόψη τα εισαγωγικά, μερικούς ειδικούς χαρακτήρες και την ευαισθησία των συμβολοσειρών σε κεφαλαία-πεζά.

3.4.2.1 Χρησιμοποιώντας ειδικούς χαρακτήρες

Οι ειδικοί χαρακτήρες επιτρέπουν να προσθέσουμε πράγματα στις συμβολοσειρές που δεν θα μπορούσαν να προστεθούν διαφορετικά. Για παράδειγμα, ας υποθέσουμε ότι θέλουμε να γράψουμε την παρακάτω πρόταση σε μια ιστοσελίδα: "Go to the directory c:\javascript on your computer". Εάν

χρησιμοποιήσουμε τη συμβολοσειρά, έτσι όπως είναι γραμμένη, ο κώδικας θα δείχνει ως εξής:

<script language=”JavaScript”>

document.write("Go to the directory c:\javascript on your computer.");

</ script >

Το πρόβλημα είναι ότι η μόνη ανάποδη κάθετος δεν θα τυπωθεί στην ιστοσελίδα. Θα εμφανιστεί ως εξής:

Go to the directory c:javascript on your computer

Αν η ανάποδη κάθετος δεν ακολουθείται από έναν κωδικό ενός ειδικού χαρακτήρα, η JavaScript τυπώνει τον χαρακτήρα μετά την κάθετο όπως είναι. Για να διορθωθεί αυτό, χρησιμοποιείται ο ειδικός κωδικός \\ για να τυπώσουμε μια μόνο ανάποδη κάθετο στη σελίδα:

< script language=”JavaScript”>

document.write ("go to the directory c:\\javascript on your computer.'

</ script >

Οι ειδικοί χαρακτήρες που χρησιμοποιούνται στην JavaScript παρουσιάζονται στον παρακάτω πίνακα 2.

Χαρακτήρας Εξόδου	**Ειδικός Κωδικός**
Ανάποδη κάθετος (\)	\\
Διπλά εισαγωγικά (")	\’’
Μονά εισαγωγικά (')	\’
Backspace	\b

Τροφοδοσία φόρμας	\f
Αλλαγή γραμμής (newline)	\n
Χαρακτήρας επιστροφής (carriage return)	\r
Στηλοθέτης (tab)	\t

ΠΙΝΑΚΑΣ 2 Ειδικοί χαρακτήρες της JavaScript

3.4.2.2 Οι χαρακτήρες διαφυγής

Η JavaScript επιτρέπει να χρησιμοποιούμε κάποιους χαρακτήρες διαφυγής (escape), έτσι ώστε να εμφανίζονται σωστά και να αποφεύγονται τα λάθη. Όπως οι ειδικοί χαρακτήρες, έτσι και οι χαρακτήρες διαφυγής χρησιμοποιούν τον χαρακτήρα της ανάποδης καθέτου (\), που μπαίνει πριν από τον χαρακτήρα που θέλουμε να χρησιμοποιηθεί με ειδικό τρόπο. Η JavaScript ελέγχει κάθε συμβολοσειρά για την παρουσία ειδικών χαρακτήρων πριν την εμφανίσει. Αυτό είναι χρήσιμο εάν θέλουμε να εμφανίσουμε ένα εισαγωγικό μέσα σε μια συμβολοσειρά. Για παράδειγμα, ας υποθέσουμε ότι θέλουμε να τυπώσουμε την παρακάτω πρόταση σε μια ιστοσελίδα:

John said, "JavaScript is easy."

Εάν ο κώδικας έδειχνε κάπως έτσι τα εισαγωγικά δεν θα εμφανίζονταν.

```
< script language="JavaScript">
document.write("John said, "JavaScript is easy."");
</ script >
```

Για να αποφύγουμε προβλήματα με τα εισαγωγικά, χρησιμοποιούμε τον χαρακτήρα της ανάποδης καθέτου. Τοποθετώντας μια ανάποδη κάθετο μπροστά από κάθε ένα διπλό εισαγωγικό αναγκάζουμε τα εσωτερικά διπλά

εισαγωγικά να φαίνονται ως τμήμα συμβολοσειράς κειμένου και όχι ως τμήμα της πρότασης της JavaScript:

```
< script language="JavaScript">

document.write("John said,  \'JavaScript is easy.\'");

</ script >
```

3.4.3 Λογικές μεταβλητές

Μια μεταβλητή τύπου Boolean (λογική) είναι μια τιμή true (αληθές) ή false (ψευδές), όπως για παράδειγμα:

```
var JohnCodes=true;

 var JohnlsCool=false;
```

Οι λέξεις true και false δεν πρέπει να περικλείονται σε εισαγωγικά. Αυτό συμβαίνει επειδή είναι δεσμευμένες λέξεις, τις οποίες αναγνωρίζει η JavaScript ως τιμές Boolean.

Αντί να χρησιμοποιήσουμε τις λέξεις true και false, η JavaScript επιτρέπει επίσης να χρησιμοποιήσουμε τον αριθμό 1 για το true και τον αριθμό 0 για το false:

```
var JohnCodes=1;

var JohnisCool=0;
```

Οι λογικές μεταβλητές είναι χρήσιμες όταν χρειαζόμαστε μεταβλητές που μπορούν να έχουν μόνο τις τιμές αληθής και ψευδής.

3.4.4 Μεταβλητές null

Το null σημαίνει ότι η μεταβλητή δεν έχει καμία τιμή. Δεν είναι ένα κενό ή το μηδέν, απλώς δεν είναι τίποτα. Εάν πρέπει να ορίσουμε μια μεταβλητή τιμή null, χρησιμοποιούμε μια δήλωση όπως αυτό:

var variablename=null;

Όπως και με τις μεταβλητές Boolean, δεν πρέπει αυτή η τιμή να περικλείεται σε εισαγωγικά όπως γίνεται με τις συμβολοσειράς, επειδή η JavaScript αναγνωρίζει το null ως λέξη-κλειδί με μια προκαθορισμένη τιμή (το τίποτα). Οι μεταβλητές null είναι χρήσιμες για τον έλεγχο της εισόδου στα script.

ΚΕΦΑΛΑΙΟ 4: Η ΧΡΗΣΗ ΤΩΝ ΣΥΝΑΡΤΗΣΕΩΝ

4.1 Τι Είναι Μια Συνάρτηση

Μια συνάρτηση (function) είναι βασικά ένα μικρό script μέσα σε ένα μεγαλύτερο script. Ο σκοπός της είναι να εκτελεί μια μόνο εργασία ή μια σειρά εργασιών. Αυτό που κάνει μια συνάρτηση εξαρτάται από τον κώδικα που τοποθετείται μέσα στην συνάρτηση. Για παράδειγμα, μια συνάρτηση μπορεί να γράφει μια γραμμή κειμένου στον browser ή να υπολογίζει μια αριθμητική τιμή και να επιστρέφει αυτήν την τιμή στο κύριο script. Κατά κύριο λόγο, οι συναρτήσεις βοηθούν ώστε να οργανώνονται τα διάφορα μέρη ενός script για διαφορετικές εργασίες, που πρέπει να επιτευχθούν.

Ένας άλλος λόγος που είναι χρήσιμες οι συναρτήσεις είναι η μεταφερτότητα τους. Μπορούν να χρησιμοποιηθούν περισσότερες από μία φορά μέσα στο script για να εκτελέσουν την εργασία τους. Αντί να γραφτεί ξανά ένα ολόκληρο μπλοκ κώδικα, μπορεί απλώς να καλεστεί πάλι η συνάρτηση.

4.2 Η δόμηση των συναρτήσεων

Μια συνάρτηση πρέπει να δηλωθεί με το όνομα της και τον κώδικα της. Υπάρχουν μερικές προαιρετικές προσθήκες, που μπορεί να χρησιμοποιηθούν για να γίνουν οι συναρτήσεις ακόμα πιο χρήσιμες. Μπορεί να εισαχθούν μια ή περισσότερες μεταβλητές στη συνάρτηση, οι οποίες ονομάζονται παράμετροι. Μπορεί επίσης να επιστρέψει μια τιμή στο κύριο script από τη συνάρτηση χρησιμοποιώντας την πρόταση return.

Η πρώτη γραμμή μιας συνάρτησης, δηλώνει την συνάρτηση, την ονομάζει και υποδεικνύει εάν θα δέχεται παραμέτρους. Για να δηλωθεί μια συνάρτηση χρησιμοποιείται η δεσμευμένη λέξη function, ακολουθούμενη από το όνομα της και έπειτα από ένα σύνολο παρενθέσεων:

function functionname()

Η δεσμευμένη λέξη function "λέει" στον browser ότι δηλώνουμε μια συνάρτηση και ακολουθούν και άλλες πληροφορίες. Το επόμενο τμήμα των πληροφοριών είναι το όνομα της συνάρτησης. Μετά από αυτό, το σύνολο των παρενθέσεων υποδεικνύει εάν η συνάρτηση δέχεται παραμέτρους.

Για παράδειγμα, για να ονομάσουμε την συνάρτηση μας reallycool και να δείξουμε ότι δεν χρησιμοποιεί παραμέτρους, η πρώτη γραμμή θα δείχνει ως εξής:

function reallycool()

Ο ορισμός του κώδικα των συναρτήσεων γίνεται με την χρήση των αγκίστρων ({ }) που περιβάλλουν τον κώδικα μέσα στη συνάρτηση. Το αριστερό χαρακτηρίζει την αρχή του κώδικα της συνάρτησης, μετά έρχεται ο κώδικας και, τελικά, το δεξί άγκιστρο χαρακτηρίζει το τέλος της συνάρτησης σε αυτή την μορφή:

function reallycool ()

{

JavaScript code here

}

Ο browser θα εκτελέσει όλο τον κώδικα μέσα στα άγκιστρα όταν κληθεί η συνάρτηση. Όταν ο browser φτάσει στο δεξί άγκιστρο, γνωρίζει ότι η συνάρτηση έχει τελειώσει. Ο browser θα πάει στην επόμενη γραμμή κώδικα ή θα συνεχίσει αυτό που έκανε πριν κληθεί η συνάρτηση.

Η ονομασία των συναρτήσεων γίνεται όπως και στις μεταβλητές, όπου πρέπει η ονομασία να ορίζεται προσεκτικά για να αποφεύγονται προβλήματα με τα script. Οι ίδιοι βασικοί κανόνες που ισχύουν για τις μεταβλητές ισχύουν και για την ονομασία των συναρτήσεων: ευαισθησία σε κεφαλαία-πεζά, χρήση

επιτρεπόμενων χαρακτήρων, αποφυγή των δεσμευμένων λέξεων και απόδοση ευκολομνημόνευτων και λογικών ονομάτων στις συναρτήσεις.

Επιπλέον, οι χαρακτήρες που επιτρέπονται στα ονόματα των συναρτήσεων είναι οι ίδιοι με αυτούς που μπορούμε να χρησιμοποιήσουμε στα ονόματα μεταβλητών.

Υπάρχει δυνατότητα προσθήκης παραμέτρων στις συναρτήσεις όπου οι παράμετροι χρησιμοποιούνται για να επιτραπεί σε μια συνάρτηση να εισάγει μια ή περισσότερες τιμές από κάπου έξω από τη συνάρτηση. Οι παράμετροι ορίζονται στην πρώτη γραμμή της συνάρτησης μέσα σε ένα σύνολο παρενθέσεων, μ' αυτήν τη μορφή:

function functionname (variable1, variable2)

Οποιαδήποτε τιμή εισάγεται από μια παράμετρο γίνεται μια μεταβλητή μέσα στη συνάρτηση, χρησιμοποιώντας το όνομα που δίνουμε στην παράμετρο μέσα στις παρενθέσεις.

Για παράδειγμα, εδώ βλέπουμε πώς ορίζουμε μια συνάρτηση reallycool με παραμέτρους (μεταβλητές) coolcar και coolplace:

```
function reallycool (coolcar, coolplace)
{
 Ο JavaScript κώδικας μπαίνει, εδώ
}
```

Παρατηρούμε ότι στην JavaScript, δεν χρησιμοποιείται η λέξη-κλειδί var όταν ορίζονται οι παράμετροι μιας συνάρτησης. Η JavaScript δηλώνει τις μεταβλητές αυτόματα όταν ορίζονται ως παράμετροι μιας συνάρτησης, έτσι δεν χρησιμοποιείται η λέξη κλειδί var. Για παράδειγμα, μια γραμμή όπως αυτή θα μπορούσε να προκαλέσει προβλήματα όταν θα εκτελεστεί η συνάρτηση:

function reallycool(var coolcar, var coolplace)

Οι παράμετροι λαμβάνονται έξω από τη συνάρτηση όταν γίνεται κλήση στην συνάρτηση. Όταν εκχωρούνται οι παράμετροι σε μια συνάρτηση, μπορεί να τις χρησιμοποιήσει όπως οποιεσδήποτε άλλες μεταβλητές. Για παράδειγμα, θα μπορούσαμε να δώσουμε την τιμή της μεταβλητής coolcar σε μια άλλη μεταβλητή χρησιμοποιώντας τον τελεστή εκχώρησης, όπως σε αυτό το παράδειγμα:

```
function reallycool (coolcar, coolplace)
{
var mycar=coolcar;
}
```

Αυτό εκχωρεί την τιμή της παραμέτρου coolcar σε μια μεταβλητή με όνομα mycar. Αντί να κάνουμε εκχώρηση της τιμής της σε μια άλλη μεταβλητή, θα μπορούσαμε απλώς να χρησιμοποιήσουμε την παράμετρο coolcar στη συνάρτηση, όπως σε αυτό το παράδειγμα:

```
function reallycool (coolcar, coolplace)
{
document.write ("My car is a "+coolcar);
}
```

Εάν η τιμή της coolcar είναι Corvette, τότε η συνάρτηση θα τυπώνει αυτήν την γραμμή στον browser όταν θα καλείται:

My car is a Corvette

Παρατηρούμε ότι το προηγούμενο παράδειγμα είχε δύο παραμέτρους, αλλά χρησιμοποιήθηκε μόνο μια παράμετρος. Μία συνάρτηση μπορεί να έχει λίγες ή πολλές παραμέτρους. Όταν ορίζονται πολλές παράμετροι σε μια συνάρτηση, η

συνάρτηση δεν χρειάζεται να τις χρησιμοποιήσει όλες. Μπορεί να χρησιμοποιήσει μια παράμετρο, μερικές παραμέτρους ή καμία. Το πόσες παράμετροι θα χρησιμοποιηθούν εξαρτάται από αυτό που κάνει η συνάρτηση και πώς καλείται. Ο μόνος κανόνας είναι ότι, εάν έχουμε περισσότερες από μια παραμέτρους πρέπει να χωρίσουμε κάθε παράμετρο με ένα κόμμα, ώστε ο browser να ξέρει τι να κάνει.

Για παράδειγμα εδώ φαίνεται πώς μπορεί να αλλάξει η συνάρτηση για να χρησιμοποιήσει και τις δύο παραμέτρους:

```
function reallycool (coolcar, coolplace)
{
document.write ("My car is a "+coolcar+" and I drive it to "+coolplace);
}
```

4.3 Η Κλήση Των Συναρτήσεων στα Script

Μια κλήση σε μια συνάρτηση JavaScript χρειάζεται απλώς το όνομα της συνάρτησης μαζί με ένα σύνολο παρενθέσεων με ή χωρίς παραμέτρους μεταξύ της αριστερής και δεξιάς παρένθεσης τελειώνει με ένα ελληνικό ερωτηματικό, όπως μια κανονική πρόταση της JavaScript.

```
functionname();
```

Μπορούμε να καλέσουμε συναρτήσεις στην ενότητα head μιας σελίδας ή στην ενότητα body της σελίδας. Μπορεί ακόμη να γίνει κλήση μιας συνάρτησης από μια άλλη συνάρτηση. Ένας καλός κανόνας που ακολουθείται είναι να τοποθετείται ο ορισμός μιας συνάρτησης πριν από την κλήση της συνάρτησης στο script.

Ο ορισμός μιας συνάρτησης πριν την κλήση της είναι απλώς μια σύσταση ορθής πρακτικής προγραμματισμού και όχι ένας αυστηρός κανόνας. Μια συνάρτηση μπορεί να κληθεί οπουδήποτε στην JavaScript, αλλά ο κώδικας της

συνάρτησης πρέπει να φορτωθεί από τον browser πριν τρέξει η συνάρτηση. Γι' αυτό προτείνεται να ορίζουμε τις συναρτήσεις μας πριν τις καλέσουμε.

Η κλήση μιας συνάρτησης μπορεί να γίνει στην ενότητα head. Όταν δημιουργούμε μια συνάρτηση, που μπορούμε να καλέσουμε μέσα στην ενότητα head του εγγράφου, χρειαζόμαστε μόνο ένα σύνολο ετικετών script.

Για παράδειγμα, ας υποθέσουμε ότι θέλουμε να εμφανίσουμε το μήνυμα "This is an alert!" στο αναδυόμενο πλαίσιο. Θα μπορούσαμε να γράψουμε την εξής εντολή:

```
window.alert("This is an alert!");
```

Τοποθετώντας το script μέσα στις ετικέτες <head> και </head> του εγγράφου, εξασφαλίζουμε ότι θα εκτελεστεί πριν φορτωθεί η υπόλοιπη σελίδα:

```
<html>
<head>
<title>Functions</title>
<script language="JavaScript">
window.alert ("This is an alert!");
</ script >
</ head >
<body>
Ο HTML κώδικας μπαίνει εδώ
</body>
</html>
```

Τέλος, η κλήση μιας συνάρτησης μπορεί να γίνει μέσα από μια άλλη συνάρτηση που αποτελεί ένα χρήσιμο τρόπο για να οργανωθεί η σειρά των

συμβάντων. Συνήθως η συνάρτηση τοποθετείται μέσα σε μία άλλη συνάρτηση που έχει να επιτελέσει μια μεγαλύτερη εργασία.

Όταν τοποθετούμε μια κλήση συνάρτησης μέσα σε μια συνάρτηση, πρέπει να οριστεί η συνάρτηση που θα κληθεί πριν οριστεί η συνάρτηση που την καλεί.

Παρακάτω βλέπουμε ένα παράδειγμα δύο συναρτήσεων μέσα στην ενότητα head όπου η δεύτερη συνάρτηση καλεί την πρώτη συνάρτηση:

```
<head>
<title>More Functions</title>
 <script language="JavaScript">
function update_alert()
{
window.alert("Welcome! This site is updated daily!");
}
function call_alert( )
{
update_alert();
}
call_alert() ;
</script>
 </head>
```

4.4 Εφαρμογή Βασισμένη στη Χρήση των Συναρτήσεων

4.4.1 Τροποποίηση κειμένου μέσα σε μια σελίδα

Στην συνέχεια, θα δημιουργήσουμε ένα απλό script για να τροποποιήσουμε τα περιεχόμενα μιας επικεφαλίδας μέσα σε μια ιστοσελίδα. Η ιδιότητα nodeValue ενός κόμβου κειμένου περιέχει το κείμενο του κόμβου και ο κόμβος κειμένου για μια επικεφαλίδα είναι παιδί αυτής της επικεφαλίδας. Έτσι, η σύνταξη για να αλλάξει το κείμενο μιας επικεφαλίδας με αναγνωριστικό head1 θα ήταν:

```
Var head1 = document.getElementById("head1");

Head1.firstChild.nodeValue = "New Text Here";
```

Αυτός ο κώδικας αντιστοιχεί τη μεταβλητή head1 στο αντικείμενο της επικεφαλίδας. Η ιδιότητα firstChild επιστρέφει τον κόμβο κειμένου, που είναι το μόνο παιδί της επικεφαλίδας και η ιδιότητα του nodeValue περιέχει το κείμενο της επικεφαλίδας.

Χρησιμοποιώντας αυτήν την τεχνική, είναι εύκολο να δημιουργηθεί μια σελίδα που επιτρέπει να αλλάζει η επικεφαλίδα δυναμικά. Η Λίστα 4.1 παρουσιάζει το πλήρες HTML έγγραφο αυτού του script.

Λίστα 4.1 Το Πλήρες Παράδειγμα Τροποποίησης Κειμένου

```
<html>

<head>

<title>Dynamic Text in JavaScript</title>

<script language="Javascript" type="text/javascript">

function ChangeTitle() {

if (!document.getElementById) return;

var newtitle = document.form1.newtitle.value;
```

```
var head1 = document.getElementById("head1");

head1.firstChild.nodeValue=newtitle;

}

</script>

</head>

<body>

<h1 ID="head1">Dynamic Text in JavaScript</h1>

<p>Using the W3C DOM, you can dynamically change the heading at the top of this page. Enter a new title and click the Change button.</p>

<form name="form1">

<input type="text" name="newtitle" size="25">

<input type="button" value="Change!"

onClick="ChangeTitle();">

</form>

</body>

</html>
```

Αυτό το παράδειγμα ορίζει μια φόρμα που επιτρέπει στον χρήστη να εισάγει μια νέα επικεφαλίδα για τη σελίδα. Το πάτημα του κουμπιού καλεί την συνάρτηση ChangeTitle(), που ορίζεται στην κεφαλίδα. Η συνάρτηση παίρνει την τιμή που εισάγει ο χρήστης στη φόρμα και αλλάζει την τιμή της επικεφαλίδας με το νέο κείμενο.

Η Εικόνα 1 παρουσιάζει αυτήν την σελίδα στον Internet Explorer, αφού έχει εισαχθεί ένας νέος τίτλος και έχει γίνει κλικ στο κουμπί Change.

ΕΙΚΟΝΑ 1 Το παράδειγμα αλλαγής της επικεφαλίδας.

ΚΕΦΑΛΑΙΟ 5: Η ΧΡΗΣΗ ΤΩΝ ΤΕΛΕΣΤΩΝ

5.1 Οι τύποι των τελεστών

Ένας τελεστής (operator) είναι ένα σύντομο σύμβολο της JavaScript που εκτελεί κάποιο είδος υπολογισμού, σύγκριση ή εκχώρηση με μια ή περισσότερες τιμές. Σε μερικές περιπτώσεις, ένας τελεστής παρέχει μια συντομότερη μέθοδο για την εκτέλεση κάποιας εργασίας, έτσι ώστε να χρειάζεται να πληκτρολογήσουμε λιγότερο κώδικα.

Η JavaScript χρησιμοποιεί διάφορους τύπους τελεστών:

- **Μαθηματικοί:** Αυτοί οι τελεστές χρησιμοποιούνται πιο συχνά για να εκτελούν μαθηματικούς υπολογισμούς με δύο τιμές. Οι μαθηματικοί ι είναι πιθανώς οι πιο γνωστοί. Χρησιμοποιούν σύμβολα όπως και *.

- **Εκχώρησης:** Αυτοί οι τελεστές χρησιμοποιούνται για να εκχωρούν τιμές σε μεταβλητές.

- **Σύγκρισης:** Αυτοί οι τελεστές χρησιμοποιούνται για να συγκρίνουν δύο μεταβλητές ή ίσως δύο πιο μεγάλες προτάσεις. Χρησιμοποιούν σύμβολα όπως το > (για το "είναι μεγαλύτερο από") και το < (για το "είναι μικρότερο από").

- **Λογικοί τελεστές:** Αυτοί οι τελεστές χρησιμοποιούνται για να συγκρίνουν δυο προτάσεις υπό όρους και μας "λένε" εάν η μια ή και οι δύο προτάσεις είναι αληθείς για να προχωρήσουμε αναλόγως. Χρησιμοποιούν σύμβολα && (επιστρέφει αληθές εάν οι προτάσεις και στις δύο πλευρές του είναι αληθείς) και το || (επιστρέφει αληθές εάν μια πρόταση σε όποια πλευρά του τελεστή είναι αληθής).

- **Bitwise:** Αυτοί είναι οι λογικοί τελεστές που δουλεύουν σε επίπεδο bit (ένα και μηδέν). Χρησιμοποιούν σύμβολα όπως το << (για αριστερή μετατόπιση των bit) και το >> (για δεξιά μετατόπιση των bit).

5.2 Οι μαθηματικοί τελεστές

Για έναν μαθηματικό υπολογισμό χρησιμοποιείται ένας μαθηματικός τελεστής. Οι τιμές που χρησιμοποιούνται μπορεί να είναι οποιοδήποτε είδος τιμών. Μερικοί από αυτούς τους τελεστές μπορούν να εκτελέσουν μια εργασία με την τιμή μιας μόνο μεταβλητής. Παρακάτω βλέπουμε ένα παράδειγμα δύο συμβολοσειρών που συνδυάζονται με τον τελεστή πρόσθεσης:

```
window.alert("I begin and "+"this is the end.");
```

Μπορούμε επίσης να χρησιμοποιήσουμε τον τελεστή πρόσθεσης όταν μια από τις τιμές είναι μια μεταβλητή, όπως σε αυτό το παράδειγμα:

```
var part2="this is the end."

window.alert("I begin and "+part2);
```

Ο τελεστής πρόσθεσης δουλεύει, επίσης, όταν και οι δύο τιμές είναι μεταβλητές όπως στο επόμενο παράδειγμα:

```
var part1="I begin and ";

var part2="this is the end."

window.alert(part1+part2);
```

Αυτά τα παραδείγματα μας δείχνουν πώς μπορούμε να χρησιμοποιήσουμε τους μαθηματικούς τελεστές με διάφορες τιμές ή/και μεταβλητές. Αυτό μας επιτρέπει κάποια ευελιξία στον τρόπο που γράφουμε τα script.

Οι τρεις τελεστές που δουλεύουν με μια μόνη τιμή είναι οι τελεστές αύξησης, μείωσης και οι μοναδιαίοι τελεστές άρνησης. Οι τελεστές αύξησης και μείωσης είναι, στην πραγματικότητα, συντομεύσεις της πρόσθεσης ή αφαίρεσης.

Παρακάτω δίνεται ο πίνακας 3 που περιέχει διάφορους μαθηματικούς τελεστές και την λειτουργία τους:

Τελεστής	**Σύμβολο**	**Λειτουργία**
Πρόσθεση	+	Προσθέτει δυο τιμές
Αφαίρεση	-	Αφαιρεί μια τιμή από μια άλλη
Πολλαπλασιασμός	*	Πολλαπλασιάζει 2 τιμές
Διαίρεση	/	Διαιρεί μια τιμή με μια άλλη τιμή
Υπόλοιπο	%	Διαιρεί μια τιμή με μια άλλη τιμή και επιστρέφει το υπόλοιπο
Αύξηση	++	Συντόμευση για πρόσθεση του 1 σε έναν αριθμό
Μείωση	--	Συντόμευση για αφαίρεση του 1 σε έναν αριθμό
Μοναδιαία άρνηση	-	Κάνει έναν θετικό αριθμό αρνητικό ή έναν αρνητικό αριθμό θετικό

ΠΙΝΑΚΑΣ 3 Μαθηματικοί τελεστές της JavaScript

5.3 Οι τελεστές εκχώρησης

Οι τελεστές εκχώρησης εκχωρούν μια τιμή σε μια μεταβλητή. Δεν συγκρίνουν δυο στοιχεία, ούτε εκτελούν λογικούς ελέγχους.

Ο παρακάτω πίνακας 4 συνοψίζει τους τελεστές εκχώρησης.

ΤΕΛΕΣΤΗΣ	**ΣΥΜΒΟΛΟ**	**ΛΕΙΤΟΥΡΓΙΑ**
Εκχώρησης	=	Εκχωρεί την τιμή στη δεξιά πλευρά

ΤΕΛΕΣΤΗΣ	**ΣΥΜΒΟΛΟ**	**ΛΕΙΤΟΥΡΓΙΑ**
		του τελεστή σε μια μεταβλητή
Πρόσθεση και Εκχώρηση	+=	Προσθέτει την τιμή στη δεξιά πλευρά του τελεστή στη μεταβλητή που είναι στην αριστερή πλευρά του και εκχωρεί έπειτα τη νέα τιμή στη μεταβλητή
Αφαίρεση και Εκχώρηση	-=	Αφαιρεί την τιμή στη δεξιά πλευρά του τελεστή από τη μεταβλητή που είναι στην αριστερή πλευρά του και εκχωρεί έπειτα τη νέα τιμή στη μεταβλητή
Πολλαπλασιασμός και εκχώρηση	*=	Πολλαπλασιάζει την τιμή στη δεξιά πλευρά του τελεστή με τη μεταβλητή που είναι στην αριστερή πλευρά του και εκχωρεί έπειτα τη νέα τιμή στη μεταβλητή
Διαίρεση και εκχώρηση	/=	Διαιρεί την μεταβλητή στη αριστερή πλευρά του τελεστή με τη τιμή που είναι στην δεξιά πλευρά του και εκχωρεί έπειτα τη νέα τιμή στη μεταβλητή
Υπόλοιπο και εκχώρηση	%=	Παίρνει το ακέραιο υπόλοιπο της διαίρεσης της μεταβλητής που είναι στην αριστερή πλευρά με την τιμή που

ΤΕΛΕΣΤΗΣ	ΣΥΜΒΟΛΟ	ΛΕΙΤΟΥΡΓΙΑ
		είναι στη δεξιά πλευρά και εκχωρεί τη νέα τιμή στη μεταβλητή
Ίσο	==	Επιστρέφει αληθές εάν οι τιμές και στις δύο πλευρές του τελεστή δεν είναι ίσες μεταξύ τους

ΠΙΝΑΚΑΣ 4 Οι τελεστές εκχώρησης της JavaScript

5.4 Οι τελεστές σύγκρισης

Οι τελεστές σύγκρισης χρησιμοποιούνται συνήθως με προτάσεις υπό όρους και προκειμένου να εκτελεστούν κάποιες ενέργειες μόνο όταν ικανοποιείται συγκεκριμένη συνθήκη. Αφού αυτοί οι τελεστές συγκρίνουν δύο τιμές, τιμή αληθές ή ψευδές, ανάλογα με τις τιμές σε κάθε πλευρά του τελεστή.

Ο παρακάτω πίνακας 5 συνοψίζει τους τελεστές σύγκρισης.

ΤΕΛΕΣΤΗΣ	ΣΥΜΒΟΛΟ	ΛΕΙΤΟΥΡΓΙΑ
Άνισο, διάφορο	!=	Επιστρέφει αληθές εάν οι τιμές και στις δυο πλευρές του τελεστή δεν είναι ίσες μεταξύ τους
Μεγαλύτερο από	>	Επιστρέφει αληθές εάν η τιμή στην αριστερή πλευρά του τελεστή είναι μεγαλύτερη από την τιμή στη δεξιά πλευρά
Μικρότερο από	<	Επιστρέφει αληθές εάν η τιμή στην αριστερή πλευρά του τελεστή είναι μικρότερη από την

ΤΕΛΕΣΤΗΣ	ΣΥΜΒΟΛΟ	ΛΕΙΤΟΥΡΓΙΑ
		τιμή στη δεξιά πλευρά
Μεγαλύτερο από ή ίσο με	>=	Επιστρέφει αληθές εάν η τιμή στην αριστερή πλευρά του τελεστή είναι μεγαλύτερη ή ίσο με την τιμή στη δεξιά πλευρά
Μικρότερο από ή ίσο με	<=	Επιστρέφει αληθές εάν η τιμή στην αριστερή πλευρά του τελεστή είναι μικρότερη ή ίση με την τιμή στη δεξιά πλευρά
Αυστηρά ίσο	===	Επιστρέφει αληθές εάν οι τιμές και στις δύο πλευρές είναι ίσες και είναι του ίδιου τύπου
Αυστηρά άνισο	!==	Επιστρέφει αληθές εάν οι τιμές και στις δύο πλευρές δεν είναι ίσες μεταξύ τους ή δεν είναι του ίδιου τύπου

ΠΙΝΑΚΑΣ 5 Οι τελεστές σύγκρισης της JavaScript

5.5 Οι λογικοί τελεστές

Οι τρεις λογικοί τελεστές μας επιτρέπουν να συγκρίνουμε δυο προτάσεις υπό όρους για να μάθουμε εάν η μία ή και οι δύο προτάσεις είναι αληθείς και να προχωρήσουμε αναλόγως. Οι λογικοί τελεστές είναι χρήσιμοι για να ελέγξουμε περισσότερες από μια συνθήκες τη φορά και να χρησιμοποιήσουμε τα αποτελέσματα. Όπως και οι τελεστές σύγκρισης, έτσι και οι λογικοί τελεστές επιστρέφουν είτε αληθές είτε ψευδές, ανάλογα με τις τιμές σε κάθε πλευρά του τελεστή.

5.5.1 Ο λογικός τελεστής AND (&&)

Ο λογικός τελεστής AND επιστρέφει αληθές εάν οι συγκρίσεις και στις δύο πλευρές του τελεστή && δώσουν αληθές. Εάν η μια ή και οι δύο συγκρίσεις σε κάθε του τελεστή είναι ψευδείς, επιστρέφει ψευδές. Στον παρακάτω πίνακα 6 παρουσιάζονται μερικές προτάσεις, που χρησιμοποιούν τον τελεστή AND.

Σύγκριση	**Τιμή Επιστροφής**	**Αιτιολόγηση**
(1==1)&&(2==2)	True	Οι συγκρίσεις και στις δύο πλευρές δίνουν αληθές: το 1 είναι ίσο με 1 και το 2 είναι ίσο με 2
(2>1)&& (3<=4)	True	Οι συγκρίσεις και στις δύο πλευρές δίνουν αληθές: το 2 είναι μεγαλύτερο από 1 και το 3 είναι μικρότερο από 4
(“A”<= “A”)&&(“c”!= “d”)	True	Οι συγκρίσεις και στις δύο πλευρές δίνουν αληθές: Το "A" είναι ίσο με το "A" και το "c" δεν είναι ίσο με το "d"
(1==1)&&(2==3)	False	Η σύγκριση στα δεξιά δίνει ψευδές
(“a”!= “a”)&&(“b”!= “q”)	False	Η σύγκριση στα αριστερά δίνει ψευδές
(2>7)&&(5>=20)	False	Οι συγκρίσεις και στις δύο

Σύγκριση	**Τιμή Επιστροφής**	**Αιτιολόγηση**
		πλευρές δίνουν ψευδές

ΠΙΝΑΚΑΣ 6 Παραδείγματα του τελεστή AND (&&)

5.5.2 Ο Τελεστής OR (| |)

Ο λογικός τελεστής OR επιστρέφει αληθές εάν η σύγκριση σε κάθε πλευρά του τελεστή επιστρέφει αληθές. Έτσι, για να επιστρέψει αληθές ο τελεστής, μόνο μια από τις προτάσεις σε μια πλευρά του πρέπει να δώσει αληθές.

Για να επιστρέψει ψευδές, οι συγκρίσεις και στις δύο πλευρές πρέπει να επιστρέψουν ψευδές. Ο παρακάτω πίνακας 7 παρουσιάζει μερικά παραδείγματα συγκρίσεων χρησιμοποιώντας τον τελεστή OR.

Σύγκριση	**Τιμή Επιστροφής**	**Αιτιολόγηση**
(2==2)||(3>5)	True	Η σύγκριση στα αριστερά δίνει αληθές
(5> 17) || (41=9)	True	Η σύγκριση στα δεξιά δίνει αληθές
(3==3)||(7<9)	True	Και οι δύο συγκρίσεις δίνουν αληθές
(4<3)||(2= = 1)	False	Και οι δύο συγκρίσεις δίνουν ψευδές
(31=3)| |(4>=8)	False	Και οι δύο συγκρίσεις δίνουν ψευδές

ΠΙΝΑΚΑΣ 7 Παραδείγματα του τελεστή OR (||)

5.5.3 Ο Τελεστής NOT (!)

Ο λογικός τελεστής NOT μπορεί να χρησιμοποιηθεί σε μια μόνο σύγκριση για να πει: "Αν αυτό δεν ισχύει, τότε επιστρέφω αληθές". Βασικά, μπορεί να κάνει μια έκφραση, που κανονικά θα επέστρεφε ψευδές, να επιστρέψει αληθές ή να κάνει μια έκφραση που κανονικά θα επέστρεφε αληθές να επιστρέψει ψευδές. Ο παρακάτω πίνακας 8 παρουσιάζει μερικά παραδείγματα αυτού του τελεστή.

Σύγκριση	**Τιμή Επιστροφής**	**Αιτιολόγηση**
!(3==3)	False	'Οτι το 3 είναι ίσο με 3 είναι αληθές, αλλά ο τελεστής NOT κάνει αυτήν την πρόταση να επιστρέψει ψευδής
!(2>5)	True	Ότι το 2 είναι μεγαλύτερο από 5 είναι ψευδές και ο τελεστής NOT κάνει τη πρόταση να επιστρέψει αληθές

ΠΙΝΑΚΑΣ 8 Παραδείγματα του τελεστή NOT (!)

5.6 Η σειρά των πράξεων

Ο παρακάτω πίνακας 9 δίνει την προτεραιότητα των τελεστών από την υψηλότερη προς την χαμηλότερη. Οι παρενθέσεις έχουν την υψηλότερη προτεραιότητα και επικαλύπτουν τους άλλους τελεστές. Συνήθως είναι βολικές όταν δεν είμαστε βέβαιοι για την προτεραιότητα των τελεστών που χρησιμοποιούμε ή εάν θέλουμε να κάνουμε κάτι πιο σαφές.

Τύπος Τελεστή	**Παράδειγμα Πράξεων**
Παρενθέσεις (επικαλύπτει τους	()

Τύπος Τελεστή	**Παράδειγμα Πράξεων**
άλλους τελεστές)	
Μοναδιαίος (μαθηματικός, λογικός ή επιπέδου bit)	- + + - - ! ~ type of void delete
Πολλαπλασιασμός, διαίρεση, υπόλοιπο	* / %
Πρόσθεση, αφαίρεση	+ -
Μετατοπίσεις (επιπέδου bit)	> > > > > > < <
Σχεσιακή σύγκριση	> > = < < = in instance of
Σύγκριση ισότητας	= = != = = = != =
AND (επιπέδου bit)	&
XOR (επιπέδου bit)	^
OR (επιπέδου bit)	I
AND (λογικός)	&&
OR (λογικός)	II
Υπό Όρους	? :
Εκχώρηση	= += -= *= /= %= < <= > > = > > > = &= ^= I =
κόμμα	,

ΠΙΝΑΚΑΣ 9 Η προτεραιότητα των τελεστών από την υψηλότερη προς τη χαμηλότερη

ΚΕΦΑΛΑΙΟ 6: ΕΛΕΓΧΟΣ ΤΗΣ ΡΟΗΣ ΜΕ ΣΥΝΘΗΚΕΣ ΚΑΙ ΒΡΟΧΟΥΣ

6.1 Χρησιμοποιώντας συντομευμένες προτάσεις υπό όρους

Μια πρόταση υπό όρους είναι μια πρόταση που μπορεί να χρησιμοποιηθεί για να εκτελέσει κάποιον κώδικα με βάση μια συνθήκη ή για να κάνει κάτι άλλο εάν αυτή η συνθήκη δεν ικανοποιείται. Μπορούμε να θεωρήσουμε μια πρόταση υπό όρους σαν να είναι το αίτιο και το αιτιατό.

Μια πρόταση υπό όρους μπορεί να είναι αρκετά χρήσιμη. Αντί να εκτελεστεί κάθε γραμμή του κώδικα του script όπως είναι, θα μπορούσε να εκτελέσει ορισμένες ενότητες του script, μόνο όταν ικανοποιείται μια συγκεκριμένη συνθήκη. Θα μπορούσαμε ακόμη και να επεκτείνουμε αυτήν την συνθήκη σε έναν συνδυασμό συνθηκών, που πρέπει να ικανοποιηθούν για να εκτελεστούν μέρη του κώδικα.

6.2 Χρήση Προτάσεων Υπό Όρους

Παρακάτω γίνεται αναφορά στους δυο τύπους προτάσεων υπό όρους που είναι χρήσιμοι στην JavaScript: τις προτάσεις if/else και τις προτάσεις switch.

6.2.1 Χρησιμοποιώντας Προτάσεις if/else

Αρχίζουμε μια πρόταση if/else με τη λέξη-κλειδί if της JavaScript, ακολουθούμενη από μια σύγκριση σε παρενθέσεις. Η παρακάτω γραμμή δείχνει ένα δείγμα της μορφής της πρώτης γραμμής:

if (η σύγκριση μπαίνει εδώ)

Για παράδειγμα, ας υποθέσουμε ότι θέλουμε να στείλουμε ένα μήνυμα που να "λέει": "You have the right number-of boats" εάν η μεταβλητή boats είναι ίση με 3. Εάν δεν ισχύει αυτό θέλουμε να στέλνει ένα μήνυμα που να "λέει": "You do not have the right number of boats". Ο κώδικας για αυτό παρουσιάζεται στο παρακάτω παράδειγμα:

if (boats==3)

```
{
window.alert("You have the right number of boats");
}
else
{
window.alert("You do not have the right number of boats");
}
```

Επιπλέον, υπάρχει η δυνατότητα να βάλουμε μια δομή μέσα σε μια άλλη δομή που έχει ίδια ή παρόμοια μορφή. Με τα μπλοκ των προτάσεων if/else, μπορούμε να τοποθετήσουμε ένθετες προτάσεις if/else μέσα στο πρώτο μπλοκ μετά τη σύγκριση (το "μπλοκ if") ή μέσα στο δεύτερο μπλοκ μετά από την λέξη-κλειδί else (το μπλοκ else").

Το τελευταίο πράγμα που θα πρέπει να ξέρουμε για την ένθεση είναι ότι μπορούμε να βάλουμε ένθετα όσα μπλοκ θέλουμε μέσα σε άλλα μπλοκ.

6.2.2 Χρήση της Πρότασης switch

Η εντολή switch μας επιτρέπει να πάρουμε μια μόνο τιμή μεταβλητής και να εκτελέσουμε μια διαφορετική γραμμή κώδικα με βάση την τιμή της μεταβλητής. Εάν θέλουμε να ελέγξουμε πολλές διαφορετικές τιμές, αυτή μπορεί να αποδειχθεί ευκολότερη μέθοδος ως προς τη χρήση ενός συνόλου ένθετων προτάσεων if/else.

Η πρώτη γραμμή μιας εντολής switch έχει την παρακάτω σύνταξη:

```
switch (varname)
```

Αντικαθιστούμε το varname με το όνομα της μεταβλητής που εξετάζουμε. Θα μπορούσαμε επίσης να το αντικαταστήσουμε με κάποιο άλλο είδος έκφρασης, την πρόσθεση δύο μεταβλητών ή κάποιους παρόμοιους

υπολογισμούς. Ο παρακάτω κώδικας είναι ένα παράδειγμα του πώς χρησιμοποιείται μια εντολή switch:

```
var thename="Fred";
  switch (thename)
  {
  case "George" :
  window.alert("George is an OK name");
  break;
  case "Fred" :
  window.alert("Fred is the coolest name!");
  window.alert ("Hi there, Fred!");
  break;
default :
window.alert("Interesting name you have there");
}
```

Κατ' αρχάς, αυτό το παράδειγμα δηλώνει μια μεταβλητή που ονομάζεται thename και της εκχωρεί την τιμή Fred. Έπειτα, αρχίζει η εντολή switch, χρησιμοποιώντας τη μεταβλητή thename ως βάση για τη σύγκριση. Κατόπιν, το μπλοκ ανοίγει με ένα άγκιστρο, ακολουθούμενο από την πρώτη πρόταση όπως είναι γραμμένη "λέει": "εάν το thename είναι ίσο με George τότε εκτέλεσε τις εντολές μετά την άνω και κάτω τελεία στο τέλος αυτής της γραμμής". Εάν το thename ήταν ίσο με George, θα βλέπαμε ένα μήνυμα.

Έπειτα η πρόταση break λέει στον browser να βγει από το μπλοκ του κώδικα και να μετακινηθεί στην επόμενη γραμμή του κώδικα μετά το μπλοκ.

Χρησιμοποιούμε την πρόταση break στο μπλοκ switch για να διασφαλίσουμε ότι εκτελείται μόνο μία από τις περιπτώσεις. Διαφορετικά, διατρέχουμε τον κίνδυνο να εκτελέσουμε όλες τις περιπτώσεις μετά από αυτήν που επέστρεψε αληθές, επειδή, εξ ορισμού, ο browser θα συνεχίσει στην επόμενη αντί να βγει τελείως από το μπλοκ όταν βρίσκει ότι μία από τις περιπτώσεις είναι αληθής. Για να βεβαιωθούμε ότι ο browser θα βγει από το μπλοκ, προσθέτουμε τη πρόταση break.

Εάν επιστρέψουμε στο script, βλέπουμε ότι το thename δεν είναι ίσο με George έτσι αγνοείται αυτή η περίπτωση. Ωστόσο, η επόμενη σύγκριση επιστρέφει αληθές, επειδή το thename είναι ίσο με το Fred στο script. Κατά συνέπεια, θα εκτελεσθεί το σύνολο των προτάσεων σε αυτό το μπλοκ case.

Τέλος, βλέπουμε την λέξη-κλειδί default. Αυτή χρησιμοποιείται στην περίπτωση που καμία από τις προτάσεις case δεν επιστρέφει αληθές. Εάν συμβεί αυτό θα εκτελεσθεί η ενότητα default του κώδικα. Παρατηρούμε ότι δεν χρειαζόμαστε τη πρόταση break μετά την ενότητα default του κώδικα, επειδή είναι του μπλοκ switch, έτσι ο browser θα βγει από το μπλοκ, χωρίς να έχει ανάγκη για μια πρόταση break.

6.3 Τι είναι βρόχος και γιατί χρησιμοποιείται

Ένα βρόχος είναι ένα μπλοκ κώδικα, που επιτρέπει να επαναλαμβάνουμε μια ενότητα κώδικα. Οι βρόχοι είναι χρήσιμοι επειδή μας επιτρέπουν να επαναλαμβάνουμε γραμμές κώδικα χωρίς να ξανά πληκτρολογούμε τον ίδιο κώδικα. Μπορεί επίσης να αλλάζει μια ή περισσότερες τιμές μεταβλητών κάθε φορά που περνά ο browser από το βρόχο, οπότε εξοικονομείται ο χρόνος και ο κόπος της πληκτρολόγησης μιας γραμμής που είναι λίγο διαφορετική από την προηγούμενη γραμμή.

Για ένα απλό παράδειγμα, υποθέτουμε ότι θέλουμε να γράψουμε μια πρόταση σε μια ιστοσελίδα πάνω από δέκα φορές στην σειρά χρησιμοποιώντας

JavaScript. Για να το κάνουμε αυτό κανονικά, θα έπρεπε να γράψουμε δέκα φορές την παρακάτω πρόταση:

*document.write("All this typing gets tiring after a while!
");*

Με έναν βρόχο, θα μπορούσε η πρόταση document.write να γραφτεί μόνο μια φορά και να οριστεί έπειτα ο αριθμός των φορών που θέλουμε να γραφτεί.

6.3.1 Χρήση του Βρόχου For

Η δομή του βρόχου for είναι πολύ παρόμοια με αυτήν των μπλοκ υπό όρους. Η πρώτη γραμμή βρόχου for θα δείχνει παρόμοια με την παρακάτω γραμμή:

for (varname=1; varname<11; varname+=1)

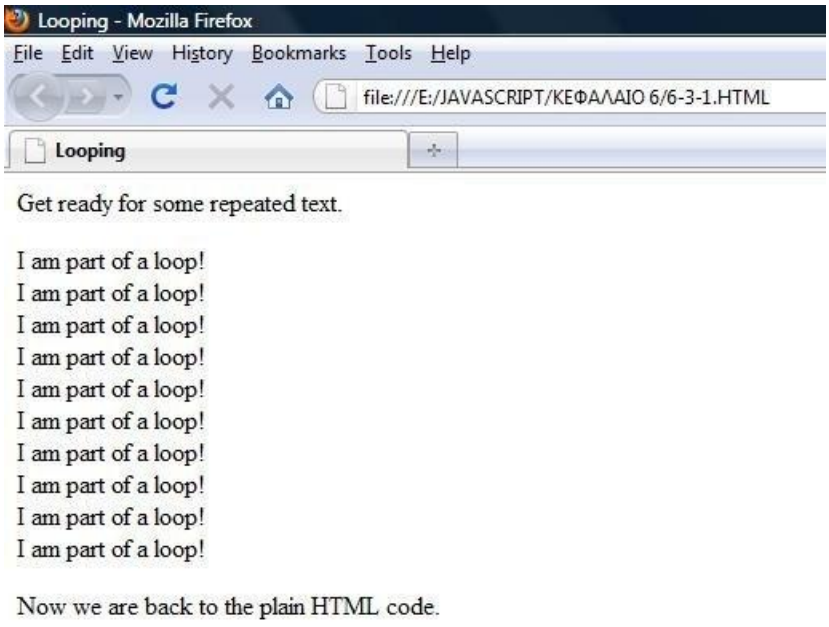
Το πρώτο πράγμα που πρέπει να εισαχθεί είναι η λέξη-κλειδί for. Ακολουθεί ένα σύνολο παρενθέσεων με τρεις προτάσεις. Αυτές οι τρεις προτάσεις "λένε" στο βρόχο πόσες φορές πρέπει να επαναληφθεί δίνοντας του ειδικές πληροφορίες.

Η πρώτη πρόταση (varname = 1) δημιουργεί μια μεταβλητή η οποία ονομάζεται varname και της εκχωρεί μια αρχική τιμή 1 η οποία χρησιμοποιείται ως αφετηρία για τον αριθμό των φορών, που θα επαναληφθεί ο βρόχος.

Η επόμενη πρόταση (varname <11) "λέει" στο βρόχο πότε να σταματήσει. Ο βρόχος θα σταματήσει να τρέχει με βάση αυτήν την πρόταση υπό όρους. Η τελευταία πρόταση (το varname + = 1) προσδιορίζει το ποσοστό κατά το οποίο αλλάζει η μεταβλητή και εάν θα γίνεται μεγαλύτερη ή μικρότερη κάθε φορά. Στον προηγούμενο κώδικα, προσθέτουμε 1 στη μεταβλητή κάθε φορά που επαναλαμβάνουμε το βρόχο.

Για να τελειώσουμε τη δομή, εισάγουμε τα άγκιστρα για να περικλείσουμε τον κώδικα που θέλουμε να χρησιμοποιήσουμε μέσα στο βρόχο. Ένα παράδειγμα πλήρους δομής ενός βρόχου for βλέπουμε στον παρακάτω κώδικα:

```
<html>
<head>
<title>Looping</title>
</head>
<body>
Get ready for some repeated text.
<p>
<script language="JavaScript">
  for (count=1;count<11;count+=1)
    {
    document.write("I am part of a loop!<BR>");
    }
</script>
<p>
Now we are back to the plain HTML code.
</body>
</html>
```



ΕΙΚΟΝΑ 2 Ο βρόχος εμφανίζει την γραμμή του κειμένου δέκα φορές

Επιπλέον δίνεται η δυνατότητα για ένθετο βρόχο μέσα σε ένα άλλο βρόχο. Το παρακάτω σας δείχνει ένα βρόχο for μέσα σε ένα άλλο βρόχο for:

```
for (count=1;count<11;count+=1) {
document.write(count+". I am part of a loop!<BR>");
  for (nestcount=1;nestcount<3;nestcount+=1){
    document.write("I keep interrupting in pairs!<BR>")
  }
}
```

Υπάρχει η δυνατότητα επίσης να βάλουμε ένθετους διαφορετικούς τύπους μπλοκ, τον ένα μέσα στον άλλο. Για παράδειγμα, μπορούμε να δημιουργήσουμε ένα μπλοκ if/else μέσα σε ένα βρόχο for:

```
for (count=1;count<11;count+=1) {
    if (count==5) {
      document.write("The loop is halfway done!<BR>");
```

```
    }
    else {
      document.write("I am part of a loop!<BR>");
    }
}
```

Σε αυτήν την περίπτωση, ο browser θα ελέγξει εάν η μεταβλητή count έχει τιμή 5. Εάν έχει τιμή 5, ο browser θα τυπώσει ένα διαφορετικό μήνυμα οθόνη από ότι θα τύπωνε διαφορετικά.

6.3.2 Χρήση του Βρόχου WHILE

Ένας βρόχος while κάνει απλώς μια σύντομη σύγκριση και επαναλαμβάνεται έως ότου η σύγκριση δεν δίνει πλέον αληθές. Η γενική σύνταξη της πρώτης γραμμής ενός βρόχου while είναι:

```
while (varname<11)
```

Ωστόσο η πρόταση while δεν δημιουργεί μια μεταβλητή με τον τρόπο που το κάνει η πρόταση for. Όταν χρησιμοποιούμε ένα βρόχο while, πρέπει να δηλώνουμε τη μεταβλητή που θέλουμε να χρησιμοποιήσουμε και να της εκχωρούμε τιμή πριν να την χρησιμοποιήσουμε στον βρόχο while.

Αν και η σύγκριση "μικρότερο από" είναι πιθανώς η πιο συνηθισμένη, μπορούμε να χρησιμοποιήσουμε όποιους άλλους τύπους σύγκρισης θέλουμε. Αυτό περιλαμβάνει επίσης, περίπλοκες συγκρίσεις με τους λογικούς τελεστές. Έτσι, θα μπορούσαμε να έχουμε μια πρώτη γραμμή όπως η παρακάτω:

```
while (( varname>4 )&&( varname<11) )
```

Αυτήν τη φορά, ο βρόχος τρέχει μόνο ενώ η μεταβλητή είναι μεγαλύτερη από 4 και μικρότερη από 11. Για να τρέξει ο βρόχος, η αρχική τιμή της

μεταβλητής θα πρέπει να είναι μέσα σε αυτό το εύρος, διαφορετικά, ο βρόχος θα αγνοούταν τελείως.

Ο παρακάτω κώδικας μας δείχνει την γενική δομή ενός πλήρους βρόχου while, ώστε να μπορούμε να δούμε πώς φαίνεται:

```
var count=1;

while (count<6)

{

O JavaScript Κώδικας Μπαίνει Εδώ

count+=1;

}
```

Παρατηρούμε ότι η τιμή 1 εκχωρείται στη μεταβλητή count πριν αρχίσει ο βρόχος. Αυτό είναι σημαντικό γιατί έτσι ο βρόχος θα τρέξει με τον τρόπο που περιμένουμε να τρέξει. Αυτός ο βρόχος έχει ορισθεί να επαναλαμβάνεται πέντε φορές, με βάση την αρχική τιμή της μεταβλητής και την αύξηση της τιμής της μεταβλητής κατά 1 κάθε φορά, που επαναλαμβάνεται ο βρόχος (count+ =1).

Σε έναν βρόχο while, πρέπει επίσης να θυμόμαστε να αλλάξουμε την τιμή της μεταβλητής που χρησιμοποιούμε, ώστε να μην κολλήσουμε σε έναν ατέρμονα βρόχο. Εάν ο προηγούμενος βρόχος δεν συμπεριελάμβανε τον κώδικα count+ =1, ο βρόχος θα επαναλαμβανόταν επ' αόριστον .

Για ένα παράδειγμα του βρόχου while σε δράση, μπορούμε να γράψουμε ξανά τον κώδικα του script που επαναλαμβάνει μία πρόταση χρησιμοποιώντας ένα βρόχο while:

```
<html>

<head>

<title>Looping</title>
```

```
</head>
 <body>
Get ready for some repeated text.
<p >
<script language="JavaScript">
var count=1;
 while  (count<11)
  {
    document.write(count+". I am part of a loop!<BR>");
    count+=1;
  }
</script>
<p>
Now we are back to the plain HTML code.
</body>
</html>
```

Σε σχέση με την ένθεση μέσα σε βρόχους while, αυτή δουλεύει όπως και στους βρόχους for. Μπορούμε να εισάγουμε ένα άλλο βρόχο while, ένα βρόχο for ή ένα μπλοκ if/else μέσα σε έναν βρόχο while. Μπορούμε επίσης εάν θέλουμε να εισάγουμε έναν βρόχο while μέσα σε άλλα μπλοκ προτάσεων. [2]

6.3.3 Χρήση του Βρόχου Do While

Ο βρόχος do while είναι ειδικός, επειδή ο κώδικας μέσα στο βρόχο εκτελείται τουλάχιστον μια φορά, ακόμα κι αν η σύγκριση επιστρέψει ψευδές. Μια σύγκριση που επιστρέφει ψευδές σε άλλους βρόχους την φορά θα είχε ως

αποτέλεσμα να μην εκτελεσθούν ποτέ. Στην περίπτωση ενός βρόχου do while, ο βρόχος εκτελείται μια φορά και έπειτα χρησιμοποιείται η σύγκριση κάθε φορά στο τέλος για να προσδιορισθεί εάν πρέπει να επαναληφθεί ο βρόχος ή όχι. Το παρακάτω αποτελεί ένα παράδειγμα ενός βρόχου do while, που θα τρέξει πέντε φορές:

```
var count=1;

do

{

document. write ("Hi!");

count+=1;

} while (count<6);
```

Παρατηρούμε ότι η λέξη-κλειδί do είναι το μόνο πράγμα στην πρώτη γραμμή του μπλοκ στον προηγούμενο κώδικα. Κατόπιν, όταν ολοκληρωθεί το μπλοκ βλέπουμε την πρόταση while και τη σύγκριση. Η πρόταση while είναι ο τρόπος που εξασφαλίζουμε ότι το μπλοκ του κώδικα εκτελείται τουλάχιστον μια φορά.

Μετά από αυτό, ο browser ελέγχει αν η σύγκριση επιστρέφει αληθές πριν επαναληφθεί. Σε αυτήν την περίπτωση, ο βρόχος επαναλαμβάνεται πέντε φορές, αφού η μεταβλητή count αρχίσει από το 1 και αυξάνεται κάθε φορά κατά 1. Όταν η τιμή της count φθάσει στο 6, ο βρόχος αγνοείται και δεν εκτελείται πλέον.

Ένας βρόχος do while είναι πιο χρήσιμος όταν υπάρχει κάποιος κώδικας, που πρέπει να εκτελεστεί τουλάχιστον μια φορά αλλά πρέπει να επαναλαμβάνεται μόνο εάν ικανοποιούνται ορισμένες συνθήκες, διαφορετικά, θα χρησιμοποιηθεί ένας από τους άλλους δύο βρόχους.

ΚΕΦΑΛΑΙΟ 7: Η ΧΡΗΣΗ ΤΩΝ ΑΝΤΙΚΕΙΜΕΝΩΝ

7.1 Τι είναι τα αντικείμενα

Η JavaScript υποστηρίζει επίσης τα αντικείμενα, που είναι ένα πιο σύνθετο είδος μεταβλητής τα οποία μπορούν να αποθηκεύουν πολλαπλά στοιχεία δεδομένων και συναρτήσεις.

Αν και μια μεταβλητή μπορεί να έχει μόνο μια τιμή κάθε φορά, ένα αντικείμενο μπορεί να περιέχει πολλαπλές τιμές, καθώς επίσης και συναρτήσεις για να μπορεί να δουλέψει με τις τιμές. Αυτό μας επιτρέπει να ομαδοποιούμε σχετικά αρχεία δεδομένων και τις συναρτήσεις που τα χειρίζονται σε ένα μόνο αντικείμενο.

Κάθε αντικείμενο έχει μια ή περισσότερες ιδιότητες, δηλαδή μεταβλητές που θα αποθηκευτούν μέσα στο αντικείμενο. Όπως και οι μεταβλητές, κάθε ιδιότητα αντικειμένου έχει μια τιμή. Για να διαβάσουμε την τιμή μιας ιδιότητας, απλώς πρέπει να συμπεριλάβουμε το όνομα του αντικειμένου και το όνομα της ιδιότητας, χωρισμένα με μια τελεία, σε οποιαδήποτε έκφραση. Μπορούμε να αλλάξουμε τη τιμή μιας ιδιότητας χρησιμοποιώντας τον τελεστή = όπως κάνουμε και με μία μεταβλητή.

Μαζί με τις ιδιότητες, κάθε αντικείμενο μπορεί να έχει μια ή περισσότερες μεθόδους. Αυτές είναι συναρτήσεις που δουλεύουν με τα δεδομένα του αντικειμένου. Όπως και στις κανονικές συναρτήσεις, οι μέθοδοι μπορούν να δεχτούν ορίσματα σε παρενθέσεις και μπορούν να επιστρέψουν τιμές.

Αν και οι μεταβλητές και οι πίνακες της JavaScript είναι ευέλικτοι τρόποι να αποθηκεύουμε δεδομένα, μερικές φορές χρειαζόμαστε μια πιο περίπλοκη δομή. Για παράδειγμα, ας υποθέσουμε ότι δημιουργούμε ένα script για να δουλέψουμε με μια βάση δεδομένων με επαγγελματικές κάρτες που περιέχει τα ονόματα, τις διευθύνσεις και τους τηλεφωνικούς αριθμούς διαφόρων ατόμων.

Εάν επρόκειτο να χρησιμοποιήσουμε κανονικές μεταβλητές, θα έπρεπε να χρησιμοποιήσουμε διαφορετικές μεταβλητές για κάθε άτομο της βάσης δεδομένων: μια μεταβλητή για το όνομα, μια μεταβλητή για την διεύθυνση κλπ. Αυτό θα ήταν πολύ μπερδεμένο.

Με τους πίνακες θα μπορούσαμε να έχουμε έναν πίνακα ονομάτων, έναν πίνακα διευθύνσεων και έναν πίνακα τηλεφωνικών αριθμών. Κάθε άτομο στη βάση δεδομένων θα είχε ένα στοιχείο σε κάθε πίνακα. Αυτό θα ήταν πιο βολικό, αλλά όχι τέλειο.

Με τα αντικείμενα, μπορούμε να κάνουμε τις μεταβλητές που αποθηκεύουν τη βάση δεδομένων τόσο λογικές όσο και τις επαγγελματικές κάρτες. Κάθε άτομο αντιπροσωπεύεται από ένα αντικείμενο Card, το οποίο έχει ιδιότητες για το όνομα, την διεύθυνση και τον αριθμό τηλεφώνου. Μπορούμε ακόμη να προσθέσουμε μεθόδους στο αντικείμενο ώστε να εμφανίσουμε ή να δουλέψουμε με τις πληροφορίες του.

Στην JavaScript, υπάρχουν πολλά προκαθορισμένα αντικείμενα που μπορούμε να χρησιμοποιήσουμε για να έχουμε πρόσβαση σε ορισμένες ιδιότητες ή μεθόδους που μπορεί να χρειαζόμαστε για να κάνουμε τα script ακόμα πιο διαλογικά.

7.2 Ορισμός του αντικειμένου document

Το αντικείμενο document είναι ένα αντικείμενο που δημιουργείται από το browser για κάθε νέα HTML σελίδα (document - έγγραφο), που προβάλλεται. Έτσι, η JavaScript παρέχει την πρόσβαση σε διάφορες ιδιότητες και μεθόδους που μπορούν να επηρεάσουν το έγγραφο με διάφορους τρόπους.

7.2.1 Οι ιδιότητες του αντικειμένου document

Ο πίνακας 10 αναφέρει τις ιδιότητες του αντικειμένου document, με μια σύντομη περιγραφή τους. Μετά από τον πίνακα, αναλύεται κάθε ιδιότητα με περισσότερη λεπτομέρεια.

Ιδιότητα	**Περιγραφή**
activeElement	Επιστρέφει μια συμβολοσειρά που περιέχει την τιμή του ενεργού στοιχείου του εγγράφου
alinkColor	Επιστρέφει τη δεκαεξαδική τιμή του ενεργού χρώματος συνδέσεων του εγγράφου
anchors	Ένας πίνακας όλων των αγκυρώσεων με όνομα του εγγράφου
applets	Ένας πίνακας όλων των Java applets (βοηθητικές εφαρμογές) ενός εγγράφου
bgcolor	Επιστρέφει τη δεκαεξαδική τιμή του χρώματος φόντου του εγγράφου
charset	Επιστρέφει μια συμβολοσειρά που αντιπροσωπεύει το σύνολο χαρακτήρων, που χρησιμοποιείται για να κωδικοποιηθεί το έγγραφο
classes	Χρησιμοποιείται για να ορίζει τις κλάσεις των φύλλων στυλ του εγγράφου
cookie	Χρησιμοποιείται για να ορίζει τα JavaScript cookies ενός εγγράφου
defaultCharset	Επιστρέφει μια συμβολοσειρά, που αντιπροσωπεύει το προεπιλεγμένο σύνολο χαρακτήρων το οποίο χρησιμοποιείται για να κωδικοποιηθεί το έγγραφο

Ιδιότητα	**Περιγραφή**
designMode	Επιστρέφει μια συμβολοσειρά που παρέχει πληροφορίες για το εάν μπορεί να τροποποιηθεί το έγγραφο
dir	Επιστρέφει μια συμβολοσειρά που αντιπροσωπεύει την κατεύθυνση ανάγνωσης του εγγράφου
doctype	Επιστρέφει τη δήλωση doctype που σχετίζεται με το έγγραφο
documentElement	Επιστρέφει μια σειρά συμβολοσειρά, που αντιπροσωπεύει τον κόμβο ρίζας του εγγράφου
domain	Επιστρέφει το όνομα τομέα του διακομιστή (server) του εγγράφου
embeds	Ένας πίνακας όλων των ετικετών <embed> του εγγράφου
expando	Επιστρέφει μια λογική τιμή ανάλογα με το αν μπορούν να δημιουργηθούν αυθαίρετες μεταβλητές μέσα στο έγγραφο
fgColor	Επιστρέφει τη δεκαεξαδική τιμή του προεπιλεγμένου χρώματος κειμένου του εγγράφου
fileCreatedDate	Επιστρέφει την ημερομηνία που δημιουργήθηκε το έγγραφο

Ιδιότητα	**Περιγραφή**
fileModifiedDate	Επιστρέφει την ημερομηνία που τροποποιήθηκε τελευταία φορά το έγγραφο
fileSize	Επιστρέφει το μέγεθος των αρχείων του εγγράφου, σε byte
formName	Δεν είναι μια ιδιότητα, αλλά δημιουργεί μια νέα ιδιότητα με κάθε φόρμα με όνομα που τοποθετείται στο έγγραφο
forms	Ένας πίνακας όλων των ετικετών <form> του εγγράφου
ids	Χρησιμοποιείται για να ορίσει στυλ για τα id των φύλλων στυλ του εγγράφου
images	Ένας πίνακας όλων των ετικετών <image> του εγγράφου
implementation	Επιστρέφει μια συμβολοσειρά, που αντιπροσωπεύει το αντικείμενο implementation του εγγράφου
lastModified	Επιστρέφει την ημερομηνία της τελευταίας τροποποίησης του εγγράφου
layers	Ένας πίνακας όλων των ετικετών <layer> της σελίδας (μόνο στο Netscape Navigator)
all	Επιτρέπει την πρόσβαση σε όλα τα αντικείμενα μιας σελίδας (μόνο στον Internet Explorer)

Ιδιότητα	**Περιγραφή**
linkColor	Επιστρέφει την δεκαεξαδική τιμή του προεπιλεγμένου χρώματος σύνδεσης του εγγράφου
links	Ένας πίνακας όλων των ετικετών link (a) του εγγράφου
parentWindow	Επιστρέφει μια αναφορά στο παράθυρο γονέα
plugins	Ένας πίνακας όλων των πρόσθετων που χρησιμοποιούνται στο έγγραφο
protocol	Επιστρέφει το τμήμα του πρωτοκόλλου της Web διεύθυνσης του εγγράφου
readyState	Επιστρέφει μια συμβολοσειρά που αντιπροσωπεύει την τρέχουσα κατάσταση του εγγράφου
referrer	Επιστρέφει το url του εγγράφου που παρέπεμψε τον επισκέπτη στο τρέχον έγγραφο
tags	Ορίζει το στυλ μιας HTML ετικέτας του εγγράφου
title	Επιστρέφει το κείμενο που χρησιμοποιείται μέσα στις ετικέτες <title> του εγγράφου
uniquelD	Επιστρέφει μια συμβολοσειρά που αντιπροσωπεύει
URL	Επιστρέφει το url του τρέχοντος εγγράφου

Ιδιότητα	**Περιγραφή**
URLUnencoded	Επιστρέφει το url του εγγράφου χωρίς κάποια κωδικοποίηση
vlinkColor	Επιστρέφει την δεκαεξαδική τιμή του χρώματος σύνδεσης του εγγράφου μετά την επίσκεψη του χρήστη
XMLDocument	Χρησιμοποιείται με XML έγγραφα
XSLDocument	Χρησιμοποιείται με XSL έγγραφα

ΠΙΝΑΚΑΣ 10 Οι ιδιότητες του αντικειμένου document

Η Ιδιότητα bgColor

Η ιδιότητα bgColor περιέχει την τιμή του χρώματος φόντου μιας ιστοσελίδας. Το χρώμα φόντου ορίζεται στην ιδιότητα bgColor της ετικέτας αρχής body ή ορίζεται χρησιμοποιώντας τις προεπιλογές του browser. Μπορεί επίσης να ορισθεί με ένα script στην ενότητα head μιας σελίδας χρησιμοποιώντας την ιδιότητα bgColor του αντικειμένου document.

Ο παρακάτω κώδικας δείχνει τη χρήση αυτής της ιδιότητας χρησιμοποιώντας ένα όνομα χρώματος:

```
<head>
<script language="JavaScript">
document.bgColor="lightblue";
</script>
</head>
```

Η Ιδιότητα cookie

Η ιδιότητα cookie χρησιμοποιείται για να ορίσει ένα JavaScript cookie που θα αποθηκεύσει πληροφορίες για τον επισκέπτη. Για να ορίσουμε ένα cookie ορίζουμε την τιμή της ιδιότητας document.cookie σε μια συμβολοσειρά που περιέχει τις πληροφορίες, τις οποίες θέλουμε να αποθηκεύσουμε για τον επισκέπτη. Παρακάτω δίνεται η σύνταξη:

document.cookie = string;

Θα πρέπει να αντικαταστήσουμε το string με μια συμβολοσειρά κειμένου που περιέχει τις πληροφορίες, τις οποίες θέλουμε να χρησιμοποιήσουμε. Συνήθως, αυτή είναι μια μορφή όπως στο παρακάτω παράδειγμα ενός cookie:

document.cookie = "site = homepage:food = cheeseburgers";

Υπάρχουν δύο πράγματα που ορίζονται από το cookie. Η τοποθεσία είναι η homepage και το food (τρόφιμα) είναι cheeseburgers. Μεταξύ των δύο υπάρχει μια άνω και κάτω τελεία (στην πραγματικότητα μπορεί να είναι οποιοσδήποτε χαρακτήρας της επιλογής σας), που χρησιμοποιείται για να διαχωρίζει πράγματα όταν διαβάζεται το cookie.

Η Ιδιότητα forms

Η ιδιότητα forms, που είναι και αυτή πίνακας, έχει ένα στοιχείο για κάθε φόρμα μιας ιστοσελίδας. Όπως και με άλλες παρόμοιες ιδιότητες, μπορούμε να χρησιμοποιήσουμε την ιδιότητα length (μήκος) στο document.forms.length για να ανακαλύψουμε πόσες φόρμες υπάρχουν στη σελίδα.

Η Ιδιότητα images

Η ιδιότητα images είναι ακόμα ένας πίνακας. Αυτός ο πίνακας έχει ένα στοιχείο για κάθε εικόνα μιας ιστοσελίδας. Όπως και με τις άλλες παρόμοιες ιδιότητες μπορούμε να χρησιμοποιήσουμε την ιδιότητα length (μήκος) με τη μορφή document.images.length για να ανακαλύψουμε τον αριθμό των εικόνων που υπάρχουν σε ένα HTML έγγραφο.

Κάτι άλλο στο οποίο μπορεί να χρησιμοποιηθεί αυτή η ιδιότητα είναι να εντοπίζει τους browser που υποστηρίζουν το αντικείμενο Image με JavaScript. Αυτό είναι χρήσιμο εάν θέλουμε να προφορτώσουμε μια εικόνα συνήθως όταν χρησιμοποιούμε rollover script εικόνων.

Για να ελέγξουμε εάν ένας browser υποστηρίζει το αντικείμενο Image, μπορούμε να τοποθετήσουμε τον κώδικα που χρειάζεται το αντικείμενο Image μέσα σε ένα μπλοκ if. Το μπλοκ if θα ελέγξει αν υπάρχει η ιδιότητα document.images. όπως φαίνεται στον παρακάτω κώδικα:

```
if (document.images) {
    JavaScript statements
}
```

Η Ιδιότητα lastModified

Η ιδιότητα lastModified κρατά την τιμή της ημερομηνίας και ώρας που τροποποιήθηκε τελευταία φορά το τρέχον έγγραφο. Χρησιμοποιείται συνήθως για ενημερωτικούς λόγους, όπως για εμφάνιση της ημερομηνίας που τροποποιήθηκε τελευταία φορά το έγγραφο, ώστε ο επισκέπτης να ξέρει πότε ενημερώθηκε τελευταία φορά η σελίδα. Η τιμή αυτής της ιδιότητας εξαρτάται από τον browser που χρησιμοποιούμε, αφού ο κάθε browser δίνει διαφορετικά αποτελέσματα εάν γράψουμε την ημερομηνία τελευταίας τροποποίησης στη σελίδα. Ο παρακάτω κώδικας γράφει την τιμή της ιδιότητας document.lastModified σε μια ιστοσελίδα για να εμφανίσει την ημερομηνία και ώρα της τελευταίας τροποποίησης:

```
<body>
<h1>My Always Updated Web Page!</h1>
<script language="JavaScript">
    document.write("Last Updated: "+document.lastModified);
```

```
</script>

</body>
```

Η Ιδιότητα title

Η ιδιότητα title κρατά την τιμή της συμβολοσειράς του τίτλου του HTML. Ο τίτλος ορίζεται μέσα στις ετικέτες <title> και </title> μιας σελίδας. Μπορούμε να χρησιμοποιήσουμε την ιδιότητα title για να εμφανίσουμε τον τίτλο της σελίδας στον επισκέπτη σε κάποιο μέρος, εκτός από την γραμμή στην κορυφή του παραθύρου. Δίνεται ο παρακάτω κώδικας: [2]

```
<html>

<head>

<title>Lions, Tigers and Bears !</title>

</head>

<body>

<script language="JavaScript">

   document.write("<h1>"+document.title+"</h1>");

</script>

Lions and tigers and bears were what I saw when I went to..

</body>

</html>
```

7.2.2 Οι μέθοδοι του αντικειμένου document

Οι μέθοδοι του αντικειμένου document μας επιτρέπουν να κάνουμε μερικά πράγματα που δεν μπορούσαμε να κάνουμε πριν. Ο Πίνακας 11 αναφέρει τις μεθόδους του αντικειμένου document με μια σύντομη περιγραφή σε κάθε μια.

(διάφορες μέθοδοι αφορούν συγκεκριμένους browser, όπως συνέβη και με τις ιδιότητες)

Μέθοδοι	**Περιγραφή**
attachEventf()	Επισυνάπτει μια συνάρτηση σε ένα συμβάν, έτσι ώστε να τρέχει όταν λαμβάνει χώρα το συμβάν - μόνο στον Internet Explorer
createAttribute()	Δημιουργεί ένα αντικείμενο attribute με όνομα στέλνεται σε αυτό ως παράμετρος - μόνο στον Internet Explorer
createComment()	Δημιουργεί ένα σχόλιο με την τιμή που στέλνεται σε αυτό ως παράμετρος - μόνο στον Internet Explorer
createDocumentFragment()	Δημιουργεί ένα νέο έγγραφο
createElement()	Δημιουργεί ένα στοιχείο, το οποίο έχει τον τύπο που ως παράμετρος
createEventObject()	Δημιουργεί ένα αντικείμενο event, με σκοπό το πέρασμα πληροφοριών για το συμβάν - μόνο στον Internet Explorer
createStyleSheet()	Δημιουργεί ένα φύλλο στυλ για το έγγραφο που χρησιμοποιείται - μόνο στον Internet Explorer
createTextNode()	Δημιουργεί μια συμβολοσειρά κειμένου από την τιμή που στέλνουμε σε αυτό ως παράμετρος
detach Event()	Αποσυνδέει μια συνάρτηση από ένα συμβάν –

Μέθοδοι	**Περιγραφή**
	μόνο στο Internet Explorer
elementFromPoint()	Επιστρέφει το αντικείμενο element που εμφανίζουμε: η οποία ορίζεται με δύο τιμές παραμέτρων (πίξελ από αριστερά και πίξελ την κορυφή) - μόνο στον Internet Explorer
execCommand()	Εκτελεί μια εντολή στο έγγραφο - μόνο στον Internet Explorer
getElementByld()	Επιστρέφει μια αναφορά για το αντικείμενο με ID η οποία στέλνεται ως παράμετρος
getElementsByName()	Επιστρέφει αναφορές σε αντικείμενα, όπου η ιδιότητα name περνά ως παράμετρος
getElementsByTagName()	Επιστρέφει αναφορές σε στοιχεία, όπου το όνομα ετικέτας στέλνεται ως παράμετρος
getSelection()	Επιστρέφει την τιμή συμβολοσειράς του επιλεγμένου κειμένου του εγγράφου - μόνο στον Netscape Navigator
hasFocus()	Επιστρέφει μια λογική τιμή ανάλογα με το αν το έγγραφο έχει την εστίαση - μόνο στον Internet Explorer
mergeAttributes()	Αντιγράφει ιδιότητες από ένα αντικείμενο
open()	Ανοίγει ένα νέο έγγραφο που μας επιτρέπει να γράψουμε τα περιεχόμενα του χρησιμοποιώντας

Μέθοδοι	**Περιγραφή**
	τις προτάσεις write() ή writeln()
Close()	Κλείνει ένα νέο έγγραφο που έχει ανοίξει με την μέθοδο open()
queryCommandEnabled()	Επιστρέφει μια λογική τιμή ανάλογα με το αν μια εντολή, που έχει σταλεί σε αυτό ως παράμετρος μπορεί να εκτελεστεί - μόνο στον Internet Explorer
queryCommandIndeterm()	Επιστρέφει μια λογική τιμή ανάλογα με το αν μια εντολή, που έχει σταλεί σε αυτό ως παράμετρος είναι σε κατάσταση indeterminate (απροσδιόριστη) - μόνο στον Internet Explorer
queryCommandState()	Επιστρέφει μια λογική τιμή ανάλογα με το αν μια εντολή, που έχει σταλεί σε αυτό ως παράμετρος έχει εκτελεστεί - μόνο στον Internet Explorer
queryCommandSupported()	Επιστρέφει μια λογική τιμή ανάλογα με το αν υποστηρίζεται μια εντολή, που έχει σταλεί σε αυτό ως παράμετρος - μόνο στον Internet Explorer
queryCommandValue()	Επιστρέφει την τρέχουσα τιμή του εγγράφου για την εντολή που έχει σταλεί σε αυτό ως παράμετρος - μόνο στον Internet Explorer
recalc()	Υπολογίζει ξανά τις δυναμικές ιδιότητες του

Μέθοδοι	**Περιγραφή**
	εγγράφου- μόνο στον Internet Explore
releaseCapture()	Ελευθερώνει την σύλληψη του ποντικιού από το έγγραφο - μόνο στον Internet Explorer
setActive()	Ορίζει ένα αντικείμενο ως ενεργό, αλλά δεν του δίνει την εστίαση -μόνο στον Internet Explorer
write()	Μας επιτρέπει να γράψουμε μια συμβολοσειρά κειμένου σε ένα HTML έγγραφο
writeIn()	Μας επιτρέπει να γράψουμε μια συμβολοσειρά κειμένου σε ένα HTML έγγραφο, αλλά τελειώνει τη γραμμή με έναν χαρακτήρα αλλαγής γραμμής της JavaScript

ΠΙΝΑΚΑΣ 11 Οι μέθοδοι του αντικειμένου document

Οι Μέθοδοι open() και close()

Η μέθοδος open() επιτρέπει να ανοίξουμε ένα νέο έγγραφο και να δημιουργήσουμε τα περιεχόμενα του εξ ολοκλήρου με τις προτάσεις document και document.writlen(). Όταν καλείται η μέθοδος open(), ο browser ψάχνει τις προτάσεις, ώστε να μπορεί να γράψει στη νέα σελίδα. Αφού ολοκληρωθούν οι προτάσεις write() ή/και writeln(), πρέπει να χρησιμοποιήσουμε την document.close() για να ολοκληρώσουμε τη νέα σελίδα.

Για να αναλύσουμε ένα παράδειγμα της χρήσης της μεθόδου open() ας υποθέσουμε ότι θέλουμε να γράψουμε μια νέα σελίδα βασισμένη στο όνομα του επισκέπτη. Για να το κάνουμε αυτό, όχι μόνο πρέπει να χρησιμοποιήσουμε τις μεθόδους open() και close(), αλλά πρέπει επίσης να δημιουργήσουμε μια

ιδιότητα formName για να συλλάβουμε το όνομα, που δίνει ο επισκέπτης σε ένα πλαίσιο κειμένου.

Ξεκινώντας με τον κώδικα της ενότητας body της αρχικής σελίδας χρειαζόμαστε μια φόρμα με ένα πλαίσιο κειμένου και έναν τρόπο να καλέσουμε μια συνάρτηση που θα δημιουργήσει το νέο έγγραφο. Δίνεται ο παρακάτω κώδικας:

```
<body>
<b>Enter your name in the box below, then click the button to see a personalized page!</b> <p>
<form name="myform">
Name: <input type="text" name="yourname" size="25">
<p>
<input type="button" value="Click" onClick="newpage()">
</form>
</body>
```

Στην ετικέτα input του κουμπιού, καλείται μια συνάρτηση που ονομάζεται "newpage()" με το χειριστή συμβάντος onClick. Πρέπει τώρα να δημιουργήσουμε τη συνάρτηση newpage() στην ενότητα head του εγγράφου, ώστε να δουλέψει αυτή η φόρμα.

Η συνάρτηση newpage() πρέπει να συλλάβει τα περιεχόμενα του πλαισίου κειμένου και να τα εκχωρήσει σε μια μεταβλητή. Πρέπει έπειτα να ανοίξει τη νέα προσαρμοσμένη σελίδα στο παράθυρο του browser. Ο παρακάτω κώδικας δείχνει πώς μπορεί να γίνει αυτό:

```
<head>
<script language="JavaScript">
```

```
function newpage() {
    var thename= document.myform.yourname.value;
    document.open();
    document.write("<h1>Welcome!</h1>");
    document.write("Hello, “+thename+”, and welcome to my page!”);
    document.close();
}
</script>
</head>
```

Το πρώτο πράγμα που κάνει η συνάρτηση είναι να συλλάβει τα περιεχόμενα του πλαισίου κειμένου. Αφού το όνομα της φόρμας μας είναι myform, μπορούμε να χρησιμοποιήσουμε το document.myform για να έχουμε πρόσβαση στη φόρμα.

Για να αποκτήσουμε πρόσβαση στο πλαίσιο κειμένου, πρέπει να χρησιμοποιήσουμε το όνομα του, το οποίο είναι yourname. Κατά συνέπεια, μπορούμε να φτάσουμε στο πλαίσιο κειμένου χρησιμοποιώντας το document.myform.yourname. Για να πάρουμε τα περιεχόμενα του πλαισίου κειμένου, πρέπει να χρησιμοποιήσουμε την ιδιότητα value, την οποία προσθέτουμε στο τέλος, ως εξής: document.myform.yourname.value. Αυτή η τιμή εκχωρείται έπειτα στη μεταβλητή thename για εύκολη χρήση μέσα στις εντολές document.write().

Μόλις αποκτήσουμε αυτήν την τιμή μπορούμε να ανοίξουμε τη νέα σελίδα χρησιμοποιώντας την εντολή document.open(), η οποία επιτρέπει να χρησιμοποιήσουμε μια σειρά από προτάσεις document.write() μέχρι να βρει την εντολή document.close(). Χρησιμοποιούμε τις προτάσεις document.write() για

να γράψουμε έναν χαιρετισμό στον επισκέπτη πάνω στη σελίδα. Ο παρακάτω κώδικας συνδυάζει τις ενότητες head και body του κώδικα σε μια σελίδα HTML:

```
<html>
<head>
<title>New Page Maker</title>
<script language="JavaScript">
   function newpage() {
      var thename= document.myform.yourname.value; document.open();
      document.write("<h1>Welcome!</h1>") ;
      document.write("Hello, "+thename+", and welcome to my page!");
      document.close();
   }
</script>
</head>
<body>
<b>Enter your name in the box below, then click the button to see a personalized page!</b> <p>
<form name="myform">
   Name: <input type="text" name="yourname" size="25">
   <p>
   <input type="button" value="Click" onClick="newpage()">
</form>
```

```
</body>

</html>
```

Η Μέθοδος getElementById ()

Η μέθοδος getElementById () επιστρέφει μια αναφορά στο πρώτο αντικείμενο με συγκεκριμένη ταυτότητα (id).

Η σύνταξη είναι:

```
document.getElementById(id)
```

Παρακάτω δίνεται ένα απλό παράδειγμα της χρήσης αυτής της μεθόδου, όπου όταν πατάμε ένα κλικ πάνω στην επικεφαλίδα τότε εμφανίζει ένα πλαίσιο διαλόγου αναφέροντας ότι αυτή είναι μια επικεφαλίδα.

```
<html>
<head>
<script type="text/javascript">
function getValue()
 {
 var x=document.getElementById("myHeader")
 alert(x.innerHTML)
 }
</script>
</head>
<body>
<h1 id="myHeader" onclick="getValue()">This is a header</h1>
<p>Click on the header to alert its value</p>
</body>
</html>
```

Η Μέθοδος write()

Η μέθοδος document.write() χρησιμοποιείται για να γράψουμε μια τιμή συμβολοσειράς στη σελίδα. Ο παρακάτω κώδικας θα γράψει "hi there" μέσα σε ένα HTML έγγραφο:

```
<body>
<script language="JavaScript">
document.write("hi there");
</script>
</body>
```

7.3 Ορισμός του αντικειμένου window

Το αντικείμενο window δημιουργείται για κάθε παράθυρο που εμφανίζεται στην οθόνη. Ένα παράθυρο μπορεί να είναι το κύριο παράθυρο, ένα σύνολο πλαισίων ή ένα μεμονωμένο πλαίσιο ή ακόμα και ένα νέο παράθυρο που δημιουργείται με JavaScript. Αυτό το αντικείμενο καθιστά διαθέσιμες στα script διάφορες νέες ιδιότητες και μεθόδους, μερικές από τις οποίες επιτρέπουν να έχουμε περισσότερη αλληλεπίδραση με τον επισκέπτη της σελίδας μας.

7.3.1 Οι ιδιότητες του αντικειμένου window

Σε αυτή την ενότητα δίνεται ένας πίνακας με τις ιδιότητες του αντικειμένου window και μετά μια σύντομη περιγραφή τους.

Ιδιότητα	**Περιγραφή**
closed	Περιέχει την τιμή που "λέει" εάν ένα παράθυρο έχει κλείσει ή όχι
defaultStatus	Ορίζει το προεπιλεγμένο μήνυμα που εμφανίζεται στην γραμμή κατάστασης

Ιδιότητα	**Περιγραφή**
frames	Ένας πίνακας που αντιπροσωπεύει όλα τα πλαίσια σε ένα συγκεκριμένο παράθυρο
length	Περιέχει μια τιμή ίση με τον αριθμό των πλαισίων ενός παραθύρου
location	Περιέχει την τιμή του τρέχοντος URL του παραθύρου
name	Επιτρέπει σε ένα παράθυρο να ονομαστεί
opener	Αναφέρεται στο παράθυρο που άνοιξε ένα άλλο παράθυρο
parent	Αναφέρεται στο σύνολο πλαισίων, που περιέχει το τρέχον πλαίσιο
self	Παρέχει έναν άλλο τρόπο να αναφέρετε το τρέχον παράθυρο
status	Επιτρέπει σε ένα μήνυμα να τοποθετηθεί στην γραμμή κατάστασης - επικαλύπτει το defaultStatus
top	Μια αναφορά στο κορυφαίο παράθυρο που περιέχει ένα πλαίσιο, σύνολο πλαισίων ή ένα ένθετο σύνολο πλαισίων

ΠΙΝΑΚΑΣ 12 Οι ιδιότητες του αντικειμένου window

Η Ιδιότητα closed

Η ιδιότητα closed χρησιμοποιείται για να εξετάσει εάν ένα παράθυρο έχει κλείσει από τον επισκέπτη. Ο τρόπος που χρησιμοποιείται κανονικά είναι με το όνομα ενός παραθύρου, ακολουθούμενο από την ιδιότητα closed, όπως στο παρακάτω παράδειγμα:

if (windowname.closed)

Θα πρέπει να αντικαταστήσουμε το μέρος windowname με το όνομα του παραθύρου που θέλουμε να εξετάσουμε. Αυτό είναι συνήθως ένα νέο παράθυρο που ανοίξαμε με κώδικα JavaScript.

Η Ιδιότητα location

Η ιδιότητα location (θέση) περιέχει το τρέχον URL του παραθύρου και αυτό το URL μπορεί να αλλάξει δυναμικά για να κατευθύνει έναν επισκέπτη σε μια νέα σελίδα, όπως θα έκανε οποιαδήποτε σύνδεση. Αυτό είναι βολικό εάν θέλουμε να κάνουμε ένα κουμπί να δουλεύει ως σύνδεση ή για να στείλουμε τον επισκέπτη σε μια άλλη σελίδα, ανάλογα με κάποια ορισμένη ενέργεια.

Για να δημιουργήσουμε ένα κουμπί το οποίο να δουλεύει ως σύνδεση, θα αλλάξουμε την τιμή αυτής της ιδιότητας χρησιμοποιώντας το χειριστή συμβάντος onClick:

```
<form>
  <input type="button" value="Click to Search the Web"
  onClick="window.location='http://www.yahoo.com';">
</form>
```

Η Ιδιότητα parent

Η ιδιότητα parent χρησιμοποιείται μόνο όταν υπάρχουν πλαίσια σε μια σελίδα. Μας επιτρέπει να έχουμε πρόσβαση στο γονικό σύνολο πλαισίων του τρέχοντος πλαισίου. Αυτό είναι χρήσιμο όταν θέλουμε να αλλάξουμε μια ιδιότητα σε ένα πλαίσιο από ένα άλλο πλαίσιο.

Η Ιδιότητα top

Η ιδιότητα top χρησιμοποιείται για πρόσβαση στο κορυφαίο (top) παράθυρο από όλα τα σύνολα πλαισίων (που θα μπορούσαν να είναι ένθετα το ένα στο άλλο). Αυτό είναι λίγο διαφορετικό από την ιδιότητα parent, η οποία πηγαίνει

μόνο στο κορυφαίο πλαίσιο που περιέχει το τρέχον πλαίσιο. Η ιδιότητα πηγαίνει αντίθετα μέχρι το κορυφαίο παράθυρο, ακόμα κι αν το παράθυρο περιέχει ένθετα σύνολα πλαισίων

7.3.2 Οι μέθοδοι του αντικειμένου window

Ο πίνακας 13 αναφέρει διάφορες μεθόδους του αντικειμένου window , με μια σύντομη περιγραφή κάθε μιας. Μετά τον πίνακα, περιγράφεται κάθε μέθοδος με περισσότερη λεπτομέρεια.

Μέθοδος	**Περιγραφή**
alert()	Εμφανίζεται ένα μήνυμα στον επισκέπτη, ο οποίος πρέπει έπειτα να κάνει OK για να προχωρήσει
confirm ()	Εμφανίζει ένα παράθυρο διαλόγου επιβεβαίωσης στον επισκέπτη, ο οποίος έπειτα πρέπει να κάνει κλικ στο OK ή στο Cancel για να προχωρήσει
Find()	Επιτρέπει στον επισκέπτη να ξεκινήσει το βοήθημα Find στον browser για κάποιο κείμενο σε μια σελίδα
Home()	Στέλνει τον επισκέπτη στην αρχική σελίδα, που ο ίδιος έχει ορίσει στις ρυθμίσεις Web browser
Print()	Τυπώνει τα περιεχόμενα ενός παραθύρου
Prompt()	Εμφανίζεται ένα παράθυρο διαλόγου προτροπής που ζητά από τον επισκέπτη να εισάγει κάποιες πληροφορίες
Open()	Ανοίγει ένα νέο παράθυρο στον browser
close()	Κλείνει ένα παράθυρο του browser
Blur()	Αφαιρεί την εστίαση από ένα παράθυρο

Μέθοδος	**Περιγραφή**
Focus()	Δίνει την εστίαση σε ένα παράθυρο
moveBy()	Μετακινεί ένα παράθυρο κατά ορισμένα πίξελ, που στέλνονται ως παράμετροι
moveTo()	Μετακινεί την πάνω αριστερή γωνία του παραθύρου στις συντεταγμένες που στέλνονται ως παράμετροι
resizeBy()	Αλλάζει μέγεθος σε ένα παράθυρο μετακινώντας την κάτω δεξιά γωνία κατά ορισμένα πίξελ, που στέλνονται ως παράμετροι
resizeTo()	Αλλάζει μέγεθος σε ένα ολόκληρο παράθυρο, ώστε να αποκτήσει το ύψος και το πλάτος που στέλνονται ως παράμετροι
scrollBy()	Κυλά την περιοχή προβολής ενός παραθύρου κατά ορισμένα πίξελ, που στέλνονται ως παράμετροι
scrollTo()	Κυλά την περιοχή προβολής του παραθύρου στις συγκεκριμένες συντεταγμένες που στέλνονται ως παράμετροι
setlnterval()	Καλεί μια συνάρτηση κάθε φορά που περνά ένα ορισμένο χρονικό διάστημα
Clearlnterval()	Ακυρώνει τη ενέργεια της κλήσης μιας μεθόδου setlnterval()
setTimeout()	Καλεί μια συνάρτηση μόλις περάσει ένα ορισμένο χρονικό διάστημα
clearTimeout()	Ακυρώνει τη ενέργεια της κλήσης μιας μεθόδου setTimeout()

ΠΙΝΑΚΑΣ 13 Οι μέθοδοι του αντικειμένου window

Η Μέθοδος alert()

Αυτή η μέθοδος εμφανίζει ένα μήνυμα στον επισκέπτη και αυτός πρέπει να κάνει κλικ σε ένα κουμπί OK για να συνεχίσει. Η σύνταξη της μεθόδου είναι όπως φαίνεται παρακάτω:

window.alert("Hi there!");

Αυτό απλώς εμφανίζει στον επισκέπτη ένα μήνυμα "Hi there!" στην οθόνη.

Η JavaScript είναι αρκετά επιεικής και επιτρέπει να χρησιμοποιούμε συντομεύσεις όπως αυτή, σε μερικές περιπτώσεις. Σ' αυτήν την περίπτωση, η συντόμευση επιτρέπεται επειδή το αντικείμενο window είναι το προεπιλεγμένο αντικείμενο της JavaScript. Αφού υποτίθεται ότι υπάρχει, δεν χρειάζεται να το καλέσουμε.

Αυτός ο τύπος συντόμευσης δουλεύει με όλες τις ιδιότητες και τις μεθόδους του αντικειμένου window, στις περισσότερες περιπτώσεις. Στην πραγματικότητα, το αντικείμενο document είναι κάτω από το αντικείμενο window σε ιεραρχία αντικειμένων. Μπορούμε να παραλείψουμε το window στις κλήσεις του αντικειμένου document, επειδή το window υποτίθεται ότι υπάρχει.

Μέθοδος confirm()

Η μέθοδος confirm() μπορεί να χρησιμοποιηθεί για να δώσει στον επισκέπτη την δυνατότητα να επιβεβαιώσει ή να απορρίψει μια ενέργεια. Αυτή η μέθοδος επιστρέφει μια λογική τιμή αληθές ή ψευδές, έτσι το αποτέλεσμα της συνήθως εκχωρείται σε μια μεταβλητή όταν χρησιμοποιείται.

Το παρακάτω αποτελεί την σύνταξη της εκχώρησης της τιμής σε μια μεταβλητή:

var.varname = window.confirm (" Your Message");

Υπάρχουν δύο κουμπιά που μπορεί να επιλέξει ο επισκέπτης για να κάνει κλικ: OK και Cancel. Ανάλογα τον browser, αυτό το μήνυμα μπορεί να εμφανιστεί λίγο διαφορετικά. Εάν ο επισκέπτης κάνει κλικ στο OK, η μέθοδος επιστρέφει αληθές. Εάν ο χρήστης κάνει κλικ στο Cancel, η μέθοδος επιστρέφει ψευδές. Το μειονέκτημα είναι ότι δεν μπορούμε να αλλάξουμε την τιμή του κειμένου στα κουμπιά. Πρέπει να χρησιμοποιήσουμε το OK και το Cancel, τουλάχιστον όταν χρησιμοποιούμε αυτήν τη μέθοδο επιβεβαίωσης.

Μέθοδος prompt()

Η μέθοδος prompt() προτρέπει τον επισκέπτη να εισάγει πληροφορίες. Χρησιμοποιώντας αυτήν την μέθοδο, μπορούμε να κάνουμε διάφορες ενέργειες, ανάλογα με αυτό που εισάγει ο επισκέπτης στο πλαίσιο κειμένου της προτροπής. Αυτό επιτρέπει να προσθέσουμε περισσότερη αλληλεπίδραση σε μια σελίδα ακόμα και για να προσαρμόσουμε μια σελίδα με το όνομα του επισκέπτη. Το παρακάτω είναι ένα παράδειγμα της σύνταξης της μεθόδου:

```
var varname = window.prompt("Your Text","Default Entry");
```

Θα μπορούσαμε να χρησιμοποιήσουμε τη μέθοδο prompt() για να ρωτήσουμε το όνομα του επισκέπτη και να στείλουμε ένα μήνυμα ενώ φορτώνεται η σελίδα. Ο παρακάτω κώδικας προτρέπει τον επισκέπτη να εισάγει το όνομα του και εμφανίζει ένα μήνυμα στον επισκέπτη:

```
<head>
<script language="JavaScript">
var thename=window.prompt("What's Your name?","");
if ((thename=="") || (thename==null) ) {
   thename="Anonymous Visitor";
   window.alert("Hello "+thename+"!");
```

}

</script>

</head>

Η Μέθοδος blur()

Η Μέθοδος blur() επιτρέπει να βάζουμε το παράθυρο στο παρασκήνιο, πίσω από το κύριο παράθυρο. Μια καλή χρήση της μεθόδου είναι η δημιουργία ενός κουμπιού που να επιτρέπει στον επισκέπτη να μετακινεί το παράθυρο, αλλά να μην το κλείνει, έτσι ώστε να μπορεί να χρησιμοποιηθεί πάλι αργότερα. Για να το κάνουμε αυτό, θα πρέπει πάλι να τροποποιήσουμε τον κώδικα newpage.html σας, ως εξής:

<html>

<body>

I am a new window! I am newer than that old window that opened me, so I am special. Ha, ha!

<form>

<input type="button" value="Hide Window" onClick="window.blur();">

<p>

<input type="button" value="Close Window" onClick="window.close();">

</body>

</html>

Οπότε όταν ανοίγει αυτό το παράθυρο, υπάρχουν επιλογές είτε να κρύβεται είτε να κλείνει. Το κλικ του κουμπιού Hide Window κάνει το νέο παράθυρο να χάνει την εστίαση και να μετακινείται πίσω από το κύριο παράθυρο. Το παράθυρο μπορεί να πάρει πάλι την εστίαση από τον επισκέπτη

χρησιμοποιώντας Λειτουργικό σύστημα ή μπορούμε να το επαναφέρουμε σε εστίαση από το παράθυρο καλώντας το όνομα του με τη μέθοδο focus.

Η Μέθοδος focus()

Η μέθοδος focus() μας επιτρέπει να φέρουμε ένα παράθυρο σε εστίαση (πάνω από τα άλλα παράθυρα). Ας υποθέσουμε ότι θέλουμε να επιτρέψουμε στον επισκέπτη να επαναφέρει το παράθυρο newpage.html αφού έκανε κλικ στο κουμπί Hide Window (απόκρυψη-παραθύρου). Θα πρέπει να το φέρουμε πίσω σε εστίαση με κάποιον τρόπο. Για να το κάνουμε αυτό, θα πρέπει να τροποποιήσουμε τον κώδικα newpage.html:

<html>

<body>

I am a new window! I am newer than that old window that opened me, so I am special. Ha, ha!

<form>

<input type="button" value="Hide Window" onClick="window.blur();">

<p>

<input type="button" value="Close Window" onClick="window.close();”>

</body>

</html>

Με την προσθήκη του χειριστή συμβάντος onBlur στην ετικέτα αρχής όταν μετακινείται η εστίαση από το παράθυρο, η εστίαση πηγαίνει πάλι πίσω στο παράθυρο! Αυτό κάνει άχρηστο το κουμπί Hide Window. Ωστόσο, το κουμπί Close Window δουλεύει ακόμα για να κλείσει το παράθυρο.

7.4 Εφαρμογές Βασισμένες Στα Αντικείμενα Window και Document

7.4.1 Ανοίγοντας και κλείνοντας παράθυρα

Επίσης, μπορούμε να ανοίξουμε και να κλείσουμε παράθυρα χρησιμοποιώντας την JavaScript. Η μέθοδος window.close() κλείνει ένα παράθυρο. Οι browser δεν μας επιτρέπουν κανονικά να κλείσουμε το κύριο παράθυρο του browser χωρίς την άδεια του χρήστη. Ο κύριος σκοπός αυτής της μεθόδου είναι το κλείσιμο των παραθύρων που έχουμε δημιουργήσει. Για παράδειγμα, αυτή η πρόταση κλείνει ένα παράθυρο που ονομάζεται updatewindow:

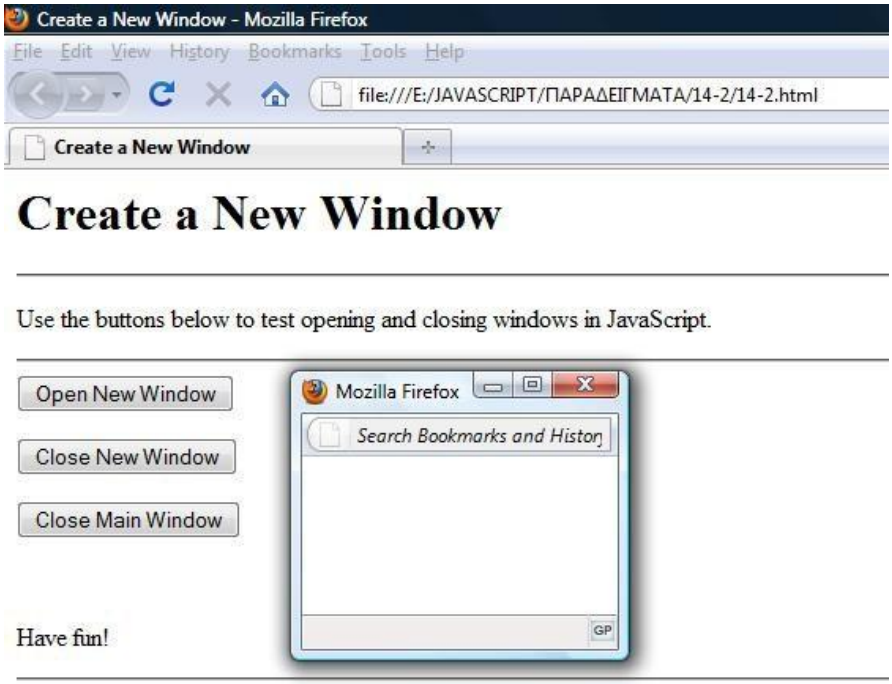
```
updatewindow.close();
```

Ως ένα άλλο παράδειγμα, η Λίστα 7.1 δείχνει ένα HTML έγγραφο όπου μας επιτρέπει να ανοίξουμε ένα μικρό νέο παράθυρο πατώντας ένα κουμπί. Μπορούμε έπειτα να πατήσουμε ένα άλλο κουμπί για να κλείσουμε το νέο παράθυρο. Το τρίτο κουμπί προσπαθεί να κλείσει το τρέχον παράθυρο. Ανάλογα με τον browser που έχουμε και τις ρυθμίσεις του, αυτό μπορεί να δουλέψει, αλλά μπορεί και όχι. Εάν κλείσουμε το παράθυρο, οι περισσότεροι browser θα ζητήσουν πρώτα επιβεβαίωση.

Λίστα 7.1 Ένα HTML έγγραφο που χρησιμοποιεί JavaScript που μας επιτρέπει να δημιουργούμε και να κλείνουμε παράθυρα

```
<html>
<head><title>Create a New Window</title>
</head>
<body>
<h1>Create a New Window</h1>
<hr>
<p>Use the buttons below to test opening and closing windows in JavaScript.</p>
```

```
<hr>

<form name="winform">

<input type="button" value="Open New Window"
onClick="NewWin=window.open('','NewWin', 'toolbar=no, status=no, width=200,
height=100'); ">

<p><input type="button" value="Close New Window"onClick="NewWin.close();"
></p>

<p><input type="button" value="Close Main Window"
onClick="window.close();"></p>

</form>

<br><p>Have fun!</p>

<hr>

</body>

</html>
```



ΕΙΚΟΝΑ 3 Ένα νέο παράθυρο browser *που άνοιξε με* JavaScript.

7.4.2 Δημιουργία ενός script για εμφάνιση παραθύρων διαλόγου

Το αντικείμενο window περιλαμβάνει τρεις μεθόδους που είναι χρήσιμες για εμφάνιση μηνυμάτων και αλληλεπίδραση με το χρήστη. Παρακάτω δίνεται μια περίληψη τους:

- Η μέθοδος *window.alert (message)* εμφανίζει ένα παράθυρο διαλόγου προειδοποίησης. Αυτό το παράθυρο διαλόγου δίνει απλώς ένα μήνυμα στο χρήστη.

- Η μέθοδος *window.confirm (message)* εμφανίζει ένα παράθυρο διαλόγου επιβεβαίωσης. Αυτό εμφανίζει ένα μήνυμα και περιλαμβάνει κουμπιά OK και Cancel. Αυτή η μέθοδος επιστρέφει true εάν πατηθεί το OK και false εάν πατηθεί το Cancel.

- Η μέθοδος *window.prompt (message, default)* εμφανίζει ένα μήνυμα και προτρέπει τον χρήστη να εισάγει κάτι. Επιστρέφει το κείμενο που

εισάγεται από τον χρήστη. Εάν ο χρήστης δεν εισάγει τίποτα, χρησιμοποιείται η προκαθορισμένη τιμή.

Για να χρησιμοποιήσουμε τις μεθόδους confirm και prompt, χρησιμοποιούμε μια μεταβλητή για να λάβουμε την απάντηση του χρήστη. Για παράδειγμα, αυτή η πρόταση εμφανίζει μια προτροπή και αποθηκεύει το κείμενο που εισάγει ο χρήστης στη μεταβλητή text:

```
text = window.prompt("Enter some text","Default value");
```

Για να δείξουμε αυτόν τον τύπο παραθύρων διαλόγου, η Λίστα 7.2 δείχνει ένα HTML έγγραφο που χρησιμοποιεί κουμπιά και χειριστές συμβάντων προκειμένου να μας επιτρέψει να ελέγξουμε τα παράθυρα διαλόγου.

Λίστα 7.2 Ένα HTML έγγραφο που χρησιμοποιεί JavaScript για να εμφανίζει ειδοποιήσεις, επιβεβαιώσεις και προτροπές

```
<html>
<head><title>Alerts, Confirmations, and Prompts</title>
</head>
<body>
<h1>Alerts, Confirmations, and Prompts</h1> <hr>
Use the buttons below to test dialogs in JavaScript. <hr>
<form name="winform">
<p><input type="button" value="Display an Alert" onClick="window.alert('This is a test alert.'); "></p>
<p><input type="button" value="Display a Confirmation"
onClick="window.confirm('Would you like to confirm?');"></p>
<p><input type="button" value="Display a Prompt"
```

```
onClick="window.prompt('Enter some Text:','This is the default value');">
</p>
</form>
<br>Have fun!
<hr>
</body>
</html>
```

Αυτό το έγγραφο εμφανίζει τρία κουμπιά και κάθε ένα χρησιμοποιεί έναν χειριστή συμβάντος για να εμφανίσει ένα από τα παράθυρα διαλόγου.

Η Εικόνα 4 δείχνει το script της Λίστας 7.2 σε δράση. Αυτήν την στιγμή εμφανίζεται το παράθυρο διαλόγου προτροπής και δείχνει την προκαθορισμένη τιμή.

ΕΙΚΟΝΑ 4 Εμφάνιση παραθύρων διαλόγου με την JavaScript

7.4.3 Δημιουργία ενός κινητού επιπέδου

Με την βοήθεια της JavaScript μπορούμε να δημιουργήσουμε ένα HTML έγγραφο που ορίζει ένα επίπεδο και να το συνδυάσουμε με ένα script για να

επιτρέψουμε στο επίπεδο να μετακινείται, να κρύβεται ή να εμφανίζεται χρησιμοποιώντας κουμπιά. Η Λίστα 7.3 παρουσιάζει το HTML έγγραφο που ορίζει τα κουμπιά και το επίπεδο.

Λίστα 7.3 Το HTML έγγραφο για το παράδειγμα του κινητού επιπέδου

```
<html>
<head>
<title>Positioning Elements with JavaScript</title>
<script language="javascript" type="text/javascript"
src="position.js">
</script>
<style>
#square {
    position:absolute;
    top:   150;
    left:  100;
    width:  200;
    height: 200;
    border: 2px solid black;
    padding:  lOpx;
    background-color:   #E0E0E0;
}
</style>
</head>
```

```
<body>
<h1>Positioning Elements</h1> <hr>
<form name="forml">
    <input type="button" name="left" value="<- Left"
    onClick="pos(-1,0);">
    <input type="button" name="right" value="Right ->"
    onClick="pos(1,0);">
    <input type="button" name="up" value="Up"
    onClick="pos(0,-l);">
    <input type="button" name="down" value="Down"
    onClick="pos(0,1);">
    <input type="button" name="hide" value="Hide"
    onClick="hideSquare();">
    <input type="button" name="show" value="Show"
    onClick="showSquare();">
</form> <hr>
<div id="square">
<p>This square is an absolutely positioned
layer that you can move using the buttons above.</p>
</div>
</body>
</html>
```

Εκτός από κάποιον απλό κώδικα HTML, αυτό το έγγραφο αποτελείται από τα εξής:

- Η ετικέτα <script> στην επικεφαλίδα διαβάζει ένα script που ονομάζεται position.js.
- Η ενότητα <style> είναι ένα σύντομο φύλλο στυλ που ορίζει τις ιδιότητες για το κινητό επίπεδο. Ορίζει την ιδιότητα location σε absolute για να υποδείξει ότι μπορεί να τοποθετηθεί σε μια ακριβή θέση, ορίζει την αρχική θέση στις ιδιότητες top και left και ορίζει τις ιδιότητες border και background-color για να γίνει το επίπεδο ορατό.
- Οι ετικέτες <input> μέσα στην ενότητα <form> ορίζουν έξι κουμπιά: τέσσερα για να μετακινείται το επίπεδο αριστερά, δεξιά, επάνω ή κάτω και δύο που θα ελέγχουν εάν θα είναι ορατό ή κρυμμένο.
- Η ενότητα ορίζει το ίδιο το επίπεδο. Η ιδιότητα id ορίζεται στην τιμή "square" (τετράγωνο). Αυτό το id χρησιμοποιείται στο φύλλο στυλ για να αναφέρεται στο επίπεδο και επίσης χρησιμοποιείται στο script.

Το script της Λίστας 7.4 προσθέτει την δράση στον HTML κώδικα.

Λίστα 7.4 To Script για το παράδειγμα του κινητού επιπέδου

```
var x=100,y=150;
function pos(dx,dy) {
    if (!document.getElementById) return;
    χ += 10*dx;
    y += 10*dy;
    obj = document.getElementById("square");
    obj.style.top=y;
```

```
    obj.style.left=x;
}
function hideSquare() {
    if (!document.getElementById) return;
     obj = document.getElementById("square");
    obj.style.display="none";
}
function showSquare() {
    if (!document.getElementByld) return;
    obj = document.getElementById("square");
    obj.style.display="block";
}
```

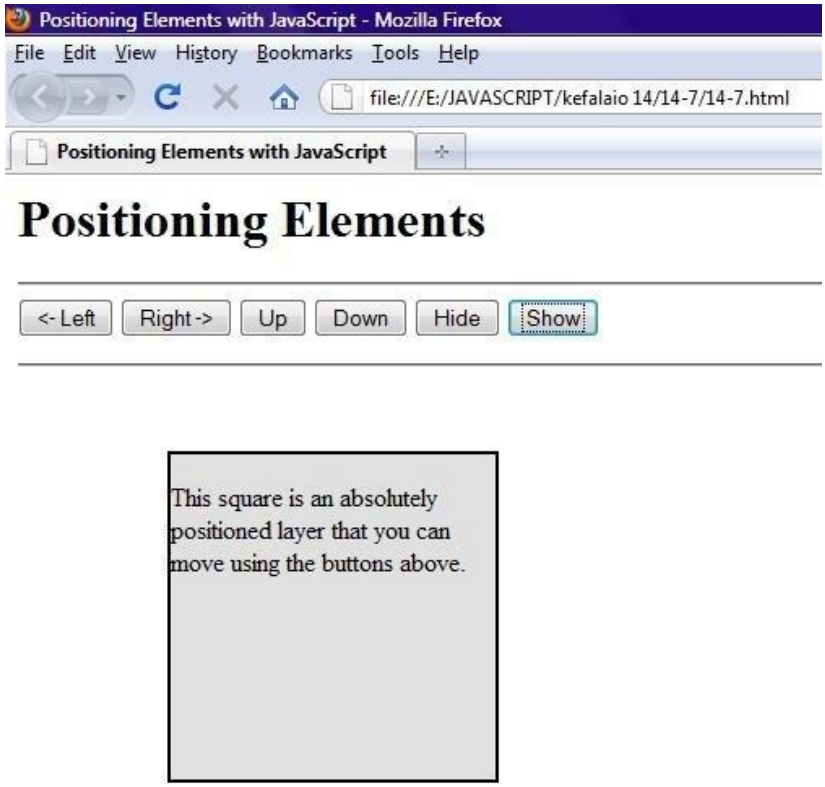
Η πρόταση var στην αρχή του script ορίζει δύο μεταβλητές, τις χ και y, που θα κρατούν την τρέχουσα θέση του επιπέδου. Η συνάρτηση pos καλείται από τους χειριστές συμβάντων και για τα τέσσερα κουμπιά μετακίνησης.

Οι παράμετροι της συνάρτησης pos (), οι dx και dy, λένε στο script πώς πρέπει να μετακινηθεί το επίπεδο: Εάν το dx είναι αρνητικό, θα αφαιρεθεί ένας αριθμός από το χ, μετακινώντας το επίπεδο στα αριστερά. Εάν το dx είναι θετικό, θα προστεθεί ένας αριθμός στο χ, μετακινώντας το επίπεδο προς τα δεξιά. Παρόμοια, το dy υποδεικνύει εάν θα μετακινηθεί πάνω ή κάτω.

Η συνάρτηση pos () αρχίζει διασφαλίζοντας ότι υποστηρίζεται η συνάρτηση getElementById(), ώστε να μην προσπαθήσει να τρέξει σε παλαιότερους browser. Πολλαπλασιάζει έπειτα το dx και το dy με 10 (για να κάνει τη μετακίνηση πιο προφανή) και εφαρμόζει το αποτέλεσμα στο χ και το y. Τέλος, ορίζει τις ιδιότητες top και left στη νέα θέση, μετακινώντας έτσι το επίπεδο.

Δύο επιπλέον συναρτήσεις, οι hideSquare() και showSquare(), κρύβουν ή εμφανίζουν το επίπεδο ορίζοντας την ιδιότητα τους display σε "none" (κρυφό) ή σε "block" (ορατό).

Για να χρησιμοποιήσουμε αυτό το script, το αποθηκεύουμε ως position.js και έπειτα φορτώνουμε το HTML έγγραφο της Λίστας 7.3 στον browser μας. Η Εικόνα 5 δείχνει αυτό το script σε δράση.



ΕΙΚΟΝΑ 5 Δημιουργία κινητού επιπέδου

7.4.4 Απόκρυψη και εμφάνιση αντικειμένων

Ένα απλό παράδειγμα, που μπορεί να δημιουργηθεί από ένα script είναι να κρύβει ή να εμφανίζει αντικείμενα μέσα σε μια σελίδα. Τα αντικείμενα έχουν μια ιδιότητα στυλ visibility που καθορίζει εάν αυτήν την στιγμή είναι ορατά μέσα στη σελίδα:

Object.style.visibility="hidden";

// κρύβει ένα αντικείμενο

```
Object.style.visibility="visible";
// εμφανίζει ένα αντικείμενο
```

Χρησιμοποιώντας αυτήν την ιδιότητα, μπορούμε να δημιουργήσουμε ένα script που να κρύβει ή να εμφανίζει αντικείμενα και στους δύο browser. Η Λίστα 7.5 δείχνει το HTML έγγραφο για ένα script που επιτρέπει σε δύο επικεφαλίδες να εμφανίζονται ή να κρύβονται.

Λίστα 7.5 Απόκρυψη και Εμφάνιση Αντικειμένων

```
<html>
<head>
<title>Hiding and Showing Objects</title>
<script language="Javascript" type="text/javascript">
function ShowHide() {
if (!document.getElementById) return;
var head1 = document.getElementById("head1");
var head2 = document.getElementById("head2");
var showhead1 = document.form1.head1.checked;
var showhead2 = document.form1.head2.checked;
head1.style.visibility=(showhead1) ? "visible" : "hidden";
head2.style.visibility=(showhead2) ? "visible" : "hidden";
}
</script>
</head>
<body>
```

<h1 ID="head1">This is the first heading</h1>

<h1 ID="head2">This is the second heading</h1>

* Using the W3C DOM, you can choose whether to show or hide the headings on this page using the checkboxes below. <form name="form1"> <input type="checkbox" name="head1"*

checked onClick="ShowHide();">

Show first heading

*
*

<input type="checkbox" name="head2"

checked onClick="ShowHide();">

Show second heading

*
*

</form>

</body>

</html>

Οι ετικέτες <h1> σε αυτό το έγγραφο ορίζουν τις επικεφαλίδες με τα αναγνωριστικά head1 και head2. Η ενότητα <form> ορίζει μια φόρμα με δύο πλαίσια ελέγχου, ένα για κάθε ένα από επικεφαλίδα. Όταν τροποποιηθεί ένα πλαίσιο ελέγχου, χρησιμοποιείται η μέθοδος onClick για να καλέσει τη συνάρτηση ShowHide().

Αυτή η συνάρτηση ορίζεται μέσα στις δηλώσεις <script> της κεφαλίδας. Η συνάρτηση αντιστοιχεί τις μεταβλητές head1 και head2 στα αντικείμενα των επικεφαλίδων, χρησιμοποιώντας τη μέθοδο getElementById(). Μετά, αντιστοιχεί τις μεταβλητές showhead1 και showhead2 στα περιεχόμενα των

ΕΙΚΟΝΑ 6 Το παράδειγμα εμφάνισης/ απόκρυψης κειμένου

παραθύρων ελέγχου. Τέλος, η συνάρτηση χρησιμοποιεί τις ιδιότητες style.visibility για να ορίσει την ορατότητα των επικεφαλίδων.

Στην εικόνα 6, το πλαίσιο ελέγχου της δεύτερης επικεφαλίδας δεν είναι επιλεγμένο, έτσι μόνο η πρώτη επικεφαλίδα είναι ορατή.

7.4.5 Δημιουργία ενός Δέντρου Πλοήγησης

Μια συνηθισμένη χρήση της JavaScript και του DOM είναι η δημιουργία ενός δυναμικού συστήματος πλοήγησης σε μορφή δέντρου για μια τοποθεσία, με ενότητες που μπορούν να επεκτείνονται και να συμπτύσσονται. Αν και αυτό δεν είναι απαραίτητο σε μικρές τοποθεσίες, αποτελεί έναν καλό τρόπο να οργανώνουμε τις, πιθανόν, εκατοντάδες συνδέσεις μιας μεγαλύτερης τοποθεσίας. Έτσι μπορούμε να δημιουργήσουμε ένα απλό δέντρο πλοήγησης χρησιμοποιώντας το DOM.

Το HTML έγγραφο που ορίζει τα περιεχόμενα του δέντρου πλοήγησης φαίνεται στην Λίστα 7.6.

Λίστα 7.6 Ο Κώδικας HTML για το Παράδειγμα του Δέντρου Πλοήγησης

```
<html>

<head><title>Creating a Navigation Tree</title>
```

```
<style>
A {text-decoration: none;}
#productsmenu,tsupportmenu,#contactmenu {
display: none;
margin-left: 2em;
}
</style>
</head>
<body>
<h1>Navigation Tree Example</h1>
<p>The navigation tree below allows you to expand and
collapse items.</p>
<ul>
<li><a id="products" href="#">[+] Products</a>
<ul ID="productsmenu">
<li><a href="prodlist.html">Product List</a></li>
<li><a href="order.html">Order Form</ax/li>
<li><a href="pricelist.html">Price List</a></li>
</ul> </li>
<li><a id="support" href="#">[+] Support</a>
<ul id="supportmenu">
<li><a href="sforum.html">Support Forum</a></li>
<li><a href="scontact.html">Contact Support</a></li>
```

```
</ul>
</li>
<li><a ID="contact" href="#">[+] Contact Us</a>
<ul id="contactmenu">
<li><a href="contact1.html">Service Department</a></li>
<li><a href="contact2.html">Sales Department</a></li>
 </ul>
</li>
 </ul>
<script language="javascript" type="text/javascript"
src="tree.js">
</script>
</body>
</html>
```

Σ' αυτό το έγγραφο, οι συνδέσεις έχουν διαταχθεί σε μια ένθετη λίστα χρησιμοποιώντας ετικέτες <ul> και <li>. Χρησιμοποιώντας μια τυπική λίστα όπως αυτή αντί για ετικέτες έχουμε δυο οφέλη: κατ' αρχάς, ο browser μορφοποιεί αυτόματα το δέντρο σαν μια λίστα με κουκκίδες. Δεύτερον, υποστηρίζει παλαιότερους browser - ακόμη και ένας browser που δεν υποστηρίζει CSS ή JavaScript θα φορτώσει και θα εμφανίσει τη λίστα σωστά. Δεν θα έχει τις δυναμικές λειτουργίες, αλλά οι συνδέσεις θα δουλεύουν.

Το δέντρο έχει τρεις κύριους κόμβους: τους Products (προϊόντα), Support (υποστήριξη) και Contact Us (επικοινωνία). Κάθε ένας έχει μια σύνδεση στην οποία μπορούμε να κάνουμε κλικ για να εμφανίσουμε ή να κρύψουμε τις συνδέσεις αυτής της ενότητας. Έχει χρησιμοποιηθεί η ιδιότητα id σε κάθε

ετικέτα <a> ώστε το script να μπορεί να επισυνάψει έναν χειριστή συμβάντος σ' αυτήν. Κάθε κόμβος έχει επίσης ένα υπό-μενού με ετικέτες <ul> και <li>. Χρησιμοποιείται επίσης μια ιδιότητα id στην ετικέτα <ul> ώστε το script να μπορεί να κρύψει ή να εμφανίσει τη λίστα.

Η ετικέτα <script> στο τέλος του εγγράφου περιλαμβάνει το script που θα δημιουργήσουμε παρακάτω. Η ετικέτα τοποθετείται μετά το σώμα της σελίδας, ώστε το script να μπορεί να προσθέσει χειριστές συμβάντων σε στοιχεία της σελίδας.

Το μπλοκ <style> στην αρχή του εγγράφου προσθέτει κάποια μορφοποίηση στις συνδέσεις και χρησιμοποιεί την ιδιότητα display: none για να κρύψει αρχικά τα υπό-μενού. Αυτά θα αποκαλύπτονται από το script όταν γίνεται κλικ στην σύνδεση.

Το script γι' αυτό το παράδειγμα παρουσιάζεται στη Λίστα 7.7.

Λίστα 7.7 Το JavaScript Αρχείο για το Παράδειγμα Πλοήγησης του Δένδρου

```
function Toggle(e) {
// Μην το δοκιμάσετε αυτό στους παλιούς browser
if (!document.getElementByld) return;
// Λήψη του αντικειμένου event
if (!e) var e = window.event;
/ / Σε ποια σύνδεση έγινε κλικ;
whichlink = (e.target) ? e.target.id : e.srcElement.id;
// Λήψη του αντικειμένου menu
obj=document.getElementById(whichlink+"menu");
// Είναι ορατό το μενού;
```

```
visible=(obj.style.display=="block")

// Λήψη του αντικειμένου key (η ίδια η σύνδεση)

key=document.getElementById(whichlink);

// Λήψη του ονόματος (Products, Contact, κλπ.)

keyname = key.firstChild.nodeValue.substring(3);

        if (visible) {

        // απόκρυψη του μενού

        obj.style.display="none";

        key.firstChild.nodeValue = "[+]" + keyname; }

         else {

        // εμφάνιση του μενού

        obj.style.display="block";

        key.firstChild.nodeValue = "[-]" + keyname;

        }

}

document. getElementById( "products").onclick=Toggle;

 document.getElementById("support").onclick=Toggle;
document.getElementById("contact").onclick=Toggle;
```

Η συνάρτηση Toggle () εμφανίζει ή κρύβει αρχικά ένα μενού. Προσδιορίζει πρώτα ποια σύνδεση προκάλεσε το συμβάν και χρησιμοποιεί έπειτα την ιδιότητα id της σύνδεσης για να βρει τα αντικείμενα για το μενού και για την ίδια τη σύνδεση. Εάν το μενού είναι ορατό αυτήν την στιγμή, κρύβεται και εάν είναι κρυμμένο αυτήν την στιγμή, εμφανίζεται. Προστίθεται το κατάλληλο

σύμβολο [+] ή [-] στο όνομα της σύνδεσης και εμφανίζεται αλλάζοντας την ιδιότητα nodevalue του κόμβου κειμένου.

Οι τελευταίες τρεις γραμμές του script αντιστοιχούν τη συνάρτηση Toggle() ως τον χειριστή συμβάντος onclick για τις τρεις πρώτες συνδέσεις του δέντρου.

Για να χρησιμοποιήσουμε αυτό το script, το αποθηκεύουμε ως tree.js στον ίδιο φάκελο με το HTML έγγραφο που δημιουργήσαμε προηγουμένως και φορτώνουμε το HTML αρχείο σε έναν browser. Η Εικόνα 7 δείχνει το παράδειγμα σε δράση, αφού έχουν επεκταθεί και οι τρεις κόμβοι του δέντρου.

ΕΙΚΟΝΑ 7 Το παράδειγμα του δέντρου πλοήγησης

ΚΕΦΑΛΑΙΟ 8: ΟΙ ΠΙΝΑΚΕΣ ΤΗΣ JAVASCRIPT

8.1 Τι είναι πίνακας

Ένας πίνακας (array) είναι ένας τρόπος αποθήκευσης παρόμοιων τύπων δεδομένων για μετέπειτα εύκολη πρόσβαση από ένα script. Στην JavaScript, ένας πίνακας είναι βασικά ένα αντικείμενο ορισμένο από το χρήστη, που προσπελάζεται, γενικά, με έναν διαφορετικό τρόπο από τα άλλα αντικείμενα. Σε έναν κανονικό πίνακα, η πρόσβαση σε ένα στοιχείο γίνεται συνήθως μέσο ενός αριθμητικού δείκτη. Ένας συσχετιζόμενος πίνακας (associative array) επιτρέπει την πρόσβαση χρησιμοποιώντας μια συμβολοσειρά αντί για αριθμό.

Η χρήση αριθμών διευκολύνει την πρόσβαση στις αποθηκευμένες πληροφορίες επειδή, χρησιμοποιώντας αριθμούς, υπάρχει η δυνατότητα της χρήσης ενός βρόχου αντί να πληκτρολογηθεί κάθε στοιχείο με το χέρι.

Για παράδειγμα, εάν εκχωρήσουμε το όνομα κάθε σπουδαστή μιας λίστας σε μία μεταβλητή και γράψουμε έπειτα τα ονόματα στην οθόνη, θα πρέπει να ξαναγράψουμε κάθε όνομα μεταβλητής στις προτάσεις document.write(). Ο παρακάτω κώδικας δείχνει ένα τέτοιο παράδειγμα:

```
<html>

<head>

<script language= "JavaScript">

var student0="Thomas";

var student1="Roger";

var student2="Amber";

 var student3="Jennifer";

</script>

</head>
```

```
<body>
document.write(student0+"<br>");
document.write(student1+"<br>") ;
document.write(student2+"<br>");
 document.write(student3+"<br>");
 </body>
</html>
```

Εάν υπάρχει η δυνατότητα χρήσης ενός βρόχου για την επανάληψη μιας μόνο πρόταση document.write() για κάθε σπουδαστή, δεν θα χρειαστεί να γραφτούν τέσσερεις διαφορετικές προτάσεις document.write(). Ένας πίνακας είναι ένας βολικός τρόπος να αποθηκεύονται τιμές (ονόματα σπουδαστών), επειδή μπορεί να περάσει όλες τις τιμές με έναν βρόχο αντί να γραφτεί κάθε τιμή σε μια ξεχωριστή πρόταση document.write().

Ένας πίνακας είναι βασικά ένας γρήγορος τρόπος να δημιουργούμε ένα αντικείμενο (αφού ήδη έχει μια συνάρτηση δημιουργίας) με μια λίστα από ιδιότητες. Ένα κανονικό αντικείμενο μπορεί να προσπελαστεί όπως ένας πίνακας, αλλά είναι συνήθως ευκολότερο να δημιουργηθεί ένας πίνακας, αντί ένα αντικείμενο, αφού δεν χρειάζεται να ασχοληθεί με τη συνάρτηση δημιουργίας.

Η ονομασία ενός πίνακα γίνεται με την χρήση των ίδιων κανόνων που ισχύουν για την ονομασία μεταβλητών, συναρτήσεων και αντικειμένων. Επιπλέον ο ορισμός ενός πίνακα είναι παρόμοιος με τη δημιουργία ενός στιγμιότυπου ενός αντικειμένου. Στην πραγματικότητα, μια μέθοδος ορισμού ενός πίνακα δείχνει και ενεργεί ακριβώς όπως η μέθοδος δημιουργίας ενός στιγμιότυπου αντικειμένου. Γιατί αυτό που γίνεται πραγματικά, είναι η δημιουργία ενός στιγμιότυπου του αντικειμένου Array της JavaScript.

Το παρακάτω παράδειγμα δείχνει την γενική σύνταξη του ορισμού ενός πίνακα στην JavaScript:

var arrayname= new Array(element0, element 1);

Για παράδειγμα, θα πάρουμε τα ονόματα των σπουδαστών από το προηγούμενο παράδειγμα και θα χρησιμοποιηθούν ως στοιχεία ενός πίνακα. Αυτός ο πίνακας θα έχει τέσσερα στοιχεία και καθένα θα έχει ως τιμή το όνομα ενός σπουδαστή. Το παρακάτω παράδειγμα δείχνει τον ορισμό του πίνακα, με τα ονόματα ως συμβολοσειρές:

var s_list= new Array("Thomas","Roger","Amber","Jennifer");

Τώρα έχουμε αποθηκεύσει τις τιμές μέσα σε έναν πίνακα τεσσάρων στοιχείων που ονομάζεται s_list. Επειδή οι τιμές είναι συμβολοσειρές, χρησιμοποιούνται εισαγωγικά (όπως σε ένα κανονικό αντικείμενο).

Η πρόσβαση στα στοιχεία ενός πίνακα, γίνεται με την χρήση αυτό ενός "δείκτη πίνακα" (index number), με τον οποίο μπορεί να υπάρχει πρόσβαση σε κάθε στοιχείο του πίνακα δίνοντας τη θέση του μέσα στον πίνακα. Για παράδειγμα, στο παρακάτω παράδειγμα βλέπουμε την σύνταξη για το πώς να εκχωρήσουμε το πρώτο στοιχείο ενός πίνακα σε μια μεταβλητή:

var varname= arrayname[0];

Θα πρέπει να αντικαταστήσουμε το varname με ένα όνομα μεταβλητής και το arrayname με το όνομα του πίνακα στον οποίο θέλουμε να έχουμε πρόσβαση. Το 0 στις, τετράγωνες αγκύλες μετά το arrayname είναι ο αριθμός δείκτη για το πρώτο στοιχείο του πίνακα. Ο αριθμός δείκτη είναι 0, επειδή οι πίνακες αρχίζουν από το 0 και όχι από το 1. Κατά συνέπεια, πρέπει να υπάρχει μεγάλη προσοχή ώστε να μην γίνει λάθος με τον αριθμό δείκτη ενός στοιχείου πίνακα. Το πρώτο στοιχειό έχει αριθμό δείκτη 0, το δεύτερο έχει αριθμό δείκτη 1, το τρίτο έχει αριθμό δείκτη 2 κ.λπ.

Μέσα στις τετράγωνες αγκύλες, αμέσως μετά το όνομα του πίνακα, τοποθετείται ο αριθμός δείκτη του στοιχείου του πίνακα που θέλουμε να προσπελάσουμε. Για να το δούμε αυτό, θα χρησιμοποιήσουμε πάλι το παράδειγμα των τεσσάρων σπουδαστών. Ο πίνακας για το όνομα κάθε σπουδαστή ορίζεται με τον παρακάτω κώδικα:

```
var s_list= new Array("Thomas","Roger","Amber","Jennifer");
```

Τώρα, ας υποθέσουμε ότι θέλουμε να εκχωρήσουμε την τιμή του πρώτου στοιχείου του πίνακα (το Thomas, σ' αυτή η περίπτωση, αφού αυτό είναι το πρώτο στοιχείο της λίστας) στη μεταβλητή που ονομάζεται tall_student. Το πρώτο στοιχείο έχει αριθμό δείκτη 0, έτσι, για να εκχωρήσουμε την τιμή του πρώτου στοιχείου στη μεταβλητή, θα χρησιμοποιήσουμε τον παρακάτω κώδικα:

```
var tall_student= s_list[0];
```

Για να δούμε ότι η τιμή της μεταβλητής tall_student είναι αυτή που θέλουμε, θα μπορούσαμε να γράψουμε ένα σύντομο script για να την γράψουμε στην σελίδα. Το παρακάτω παράδειγμα γράφει την τιμή στη σελίδα ως μέρος μιας πρότασης:

```
<html>
 <head>
<script language="JavaScript">
var s_list= new Array("Thomas","Roger","Amber""Jennifer");
var tall_student= s_list[0];
</script>
 </head>
<body>
<script language="JavaScript">
```

```
document.write("The tallest student in class is "+tall_student);
</script>
</body>
</html>
```

8.2 Οι ιδιότητες και οι μέθοδοι των πινάκων

Όπως και με άλλα αντικείμενα, ένα στιγμιότυπο του αντικειμένου Array της JavaScript μπορεί να χρησιμοποιήσει διάφορες ιδιότητες και μεθόδους του αντικειμένου Array.

8.2.1 Ιδιότητες των Πινάκων

Ο πίνακας 14 αναφέρει τις ιδιότητες του αντικειμένου Array και παρέχει μια σύντομη περιγραφή κάθε μιας.

Ιδιότητα	**Περιγραφή**
constructor	Αναφέρεται στη συνάρτηση δημιουργίας που χρησιμοποιείται για να δημιουργηθεί ένα στιγμιότυπο ενός αντικειμένου
index	Χρησιμοποιείται όταν ένας πίνακας δημιουργείται από ένα ταίριασμα με μια κανονική παράσταση (regular expression)
input	Χρησιμοποιείται όταν ένας πίνακας δημιουργείται από ένα ταίριασμα με μια κανονική παράσταση (regular expression)
length	Περιέχει μια αριθμητική τιμή ίση με τον αριθμό των στοιχείων ενός πίνακα
prototype	Επιτρέπει την προσθήκη ιδιοτήτων και μεθόδων σε αντικείμενα, όπως στο αντικείμενο Array της JavaScript

ΠΙΝΑΚΑΣ 14 Ιδιότητες του Αντικειμένου Array

Η Ιδιότητα constructor

Η ιδιότητα constructor περιέχει την τιμή του κώδικα της συνάρτησης που κατασκευάζει έναν πίνακα:

function Array() { [εγγενής κώδικας] }

Η ιδιότητα constructor είναι βασικά χρήσιμη για ενημερωτικούς λόγους ή για συγκρίσεις. Για να έχουμε πρόσβαση στην ιδιότητα, χρησιμοποιούμε το όνομα του πίνακα ακολουθούμενο από τον τελεστή τελείας και τη λέξη-κλειδί constructor. Το παρακάτω είναι η γενική σύνταξη της χρήσης μιας ιδιότητας ενός αντικειμένου Array:

arrayname.property

Η Ιδιότητα length

Η ιδιότητα length (μήκος) επιστρέφει έναν αριθμό που μας "λέει" πόσα στοιχεία έχει ένας πίνακας.

Για να χρησιμοποιήσουμε την ιδιότητα length, θα χρησιμοποιήσουμε το όνομα του πίνακα και έπειτα θα προσθέσουμε την ιδιότητα length. Ο παρακάτω κώδικας παρουσιάζει ένα παράδειγμα της ιδιότητας length που χρησιμοποιείται για να δηλώσει σε ένα επισκέπτη πόσα στοιχεία υπάρχουν σε έναν πίνακα:

```
var s_list= new Array(4)

s_list[0]=”Thomas”;

s_list[l]= “Roger”;

s_list[2]=”Amber”;

 s_list[3]="Jennifer";

 window.alert("The array has "+s_list.length+" elements");
```

Αυτός ο κώδικας θα στείλει ένα μήνυμα στον επισκέπτη που "λέει": "The array has 4 elements" (ο πίνακας έχει 4 στοιχεία).

Ιδιότητα prototype

Η ιδιότητα prototype μας επιτρέπει να προσθέσουμε ιδιότητες και μεθόδους σε ένα αντικείμενο που ήδη υπάρχει, όπως το αντικείμενο Array. Χρησιμοποιώντας αυτήν την ιδιότητα, μπορούμε να προσθέσουμε ιδιότητες στο αντικείμενο έξω από τη συνάρτηση δημιουργίας (constructor). Ωστόσο πρέπει να λάβουμε υπόψη, ότι μια αλλαγή που γίνεται με την ιδιότητα prototype επηρεάζει σε κάθε στιγμιότυπο ενός αντικειμένου που χρησιμοποιεί την ίδια συνάρτηση δημιουργίας.

Για παράδειγμα, εάν αποφασίζαμε ότι το αντικείμενο Array χρειάζεται μια άλλη ιδιότητα σε μία από τις σελίδες μας, θα μπορούσαμε να χρησιμοποιήσουμε την ιδιότητα prototype για να εκχωρήσουμε μια νέα ιδιότητα χρησιμοποιώντας την παρακάτω σύνταξη:

Array.prototype.new_property=default_ value;

θα αντικαταστήσουμε το new_property με το όνομα που θέλουμε να χρησιμοποιήσουμε για τη νέα ιδιότητα μας και να αντικαταστήσουμε το default_value με μια προκαθορισμένη τιμή γι' αυτήν την ιδιότητα.

8.2.2 Μέθοδοι των Πινάκων

Αυτή η ενότητα δείχνει τις μεθόδους που μπορούμε να χρησιμοποιήσουμε για να κάνουμε διαφορετικά πράγματα με τους πίνακες. Ο πίνακας 15 αναφέρει τις διάφορες μεθόδους και παρέχει μια σύντομη περιγραφή αυτών που κάνει κάθε μέθοδος.

Μέθοδος	**Περιγραφή**
contact()	Συνδυάζει τα στοιχεία δύο ή περισσότερων πινάκων σε έναν

Μέθοδος	**Περιγραφή**
	νέο πίνακα
join()	Συνδυάζει τα στοιχεία ενός πίνακα σε μια μόνο συμβολοσειρά, με έναν διαχωριστικό χαρακτήρα
pop()	Αφαιρεί το τελευταίο στοιχείο από έναν πίνακα και έπειτα επιστρέφει το στοιχείο που αφαιρέθηκε, αν χρειάζεται
push()	Προσθέτει στοιχεία στο τέλος ενός πίνακα και έπειτα επιστρέφει την αριθμητική τιμή του νέου μήκους του πίνακα, αν χρειάζεται
reverse()	Αντιστρέφει την κατεύθυνση των στοιχείων ενός πίνακα: το πρώτο στοιχείο μετακινείται στην τελευταία υποδοχή και το τελευταίο στοιχείο μετακινείται στην πρώτη υποδοχή κ.λπ.
shift()	Αφαιρεί το πρώτο στοιχείο από έναν πίνακα και επιστρέφει έπειτα αυτό το στοιχείο, αν χρειάζεται
unshift()	Προσθέτει στοιχεία στην αρχή ενός πίνακα και επιστρέφει την αριθμητική τιμή του μήκους του νέου πίνακα, αν χρειάζεται
slice()	Βγάζει μια συγκεκριμένη ενότητα ενός πίνακα και επιστρέφει την ενότητα ως έναν νέο πίνακα
splice()	Αφαιρεί στοιχεία από έναν πίνακα ή αντικαθιστά στοιχεία σε έναν πίνακα
sort()	Ταξινομεί τα στοιχεία ενός πίνακα σε αλφαβητική σειρά, ανάλογα με τις τιμές των συμβολοσειρών των στοιχείων του

Μέθοδος	Περιγραφή
	πίνακα

ΠΙΝΑΚΑΣ 15 Μέθοδοι του Αντικειμένου Array

Η Μέθοδος concat()

Η μέθοδος concat() χρησιμοποιείται για να συνδυάζει (ή να συνενώνει) τα στοιχεία δύο ή περισσοτέρων πινάκων και να επιστρέφει έναν νέο πίνακα, που περιλαμβάνει όλα τα στοιχεία. Χρησιμοποιούμε τα ονόματα των πινάκων ως παραμέτρους και ορίζουμε τη σειρά με την οποία θέλουμε να συνδυαστούν τα στοιχεία των πινάκων.

Αυτό το παράδειγμα συνδυάζει τα στοιχεία δύο πινάκων. Για να γίνει αυτό, πρέπει να χρησιμοποιηθεί μόνο μια παράμετρος - το όνομα το οποίο θα προσθέσουμε στο τέλος του πίνακα που χρησιμοποιείται για να καλέσουμε τη μέθοδο:

```
var fruits= new Array("oranges","apples");

var veggies= new Array("corn","peas");

var fruits_n_veggies=fruits.concat(veggies);
```

Η Μέθοδος join()

Η μέθοδος join() χρησιμοποιείται για να συνδυάζει τα στοιχεία ενός πίνακα σε μία μόνο συμβολοσειρά, όπου κάθε στοιχείο χωρίζεται με έναν χαρακτήρα που στέλνεται στη μέθοδο ως παράμετρος. Εάν δεν σταλεί καμία παράμετρος χρησιμοποιείται ένα κόμμα ως ο προκαθορισμένος διαχωριστικός χαρακτήρας όταν καλείται η μέθοδος.

Το παρακάτω παράδειγμα ορίζει έναν πίνακα και καλεί τη μέθοδο join() χρησιμοποιώντας το όνομα του πίνακα:

```
<html>
```

```
<head>
<script language="JavaScript">
var fruits= new Array (“oranges”,”apples”,”pears”);
var fruits_sting=fruits.join();
</script>
</head>
<body>
< script language="JavaScript">
document.write(“The new string is ” +fruit_string);
</script>
</body>
</html>
```

Η Μέθοδος pop()

Η μέθοδος pop() χρησιμοποιείται για να αφαιρεί το τελευταίο πίνακα. Εάν εκχωρήσουμε το αποτέλεσμα της μεθόδου σε μια μεταβλητή θα επιστραφεί το στοιχείο που αφαιρείται και θα εκχωρηθεί στη μεταβλητή.

Στο παρακάτω παράδειγμα χρησιμοποιήσουμε τη μέθοδο pop():

```
var fruits= new Array("oranges","apples","pears");
fruits.pop();
```

Εάν θέλουμε να αφαιρέσουμε ένα στοιχείο, αλλά να το χρησιμοποιήσουμε με κάποιο τρόπο αργότερα, μπορούμε να εκχωρήσουμε το αποτέλεσμα της μεθόδου σε μια μεταβλητή. Η μεταβλητή θα πάρει την τιμή του στοιχείου που αφαιρέθηκε.

Η Μέθοδος push ()

Η μέθοδος push() χρησιμοποιείται για να προσθέσουμε στοιχεία στο τέλος ενός πίνακα. Οι παράμετροι που στέλνονται στη μέθοδο είναι το νέο στοιχείο ή στοιχεία που θέλουμε να προσθέσουμε στον πίνακα.

Παρακάτω δίνεται ένα παράδειγμα κώδικα ο οποίος προσθέτει ένα στοιχείο στο τέλος ενός πίνακα:

```
var fruits= new Array("oranges","apples");

fruits.push("pears");
```

Αυτός ο κώδικας δημιουργεί έναν πίνακα που ονομάζεται fruits, με δύο στοιχεία (oranges, apples) και χρησιμοποιεί έπειτα τη μέθοδο push() για να προσθέσει ένα τρίτο στοιχείο, το pears. Ο πίνακας περιέχει τώρα τρία στοιχεία (oranges, apples, pears), όπου το pears είναι το τελευταίο στοιχείο του πίνακα. Μπορούμε να προσθέσουμε περισσότερα από ένα στοιχεία στέλνοντας περισσότερες από μία παραμέτρους. Τα στοιχεία θα προστεθούν με τη σειρά, με την οποία στέλνονται στη λίστα των παραμέτρων. Ο παρακάτω κώδικας προσθέτει δύο στοιχεία στον πίνακα fruits:

```
var fruits= new Array("oranges","apples");

fruits.push("pears","grapes");
```

Η Μέθοδος reverse()

Η μέθοδος reverse() χρησιμοποιείται για να αντιστρέφει τη σειρά των στοιχείων ενός πίνακα. Αφού αυτό είναι το μόνο που κάνει, δεν υπάρχει ανάγκη να σταλούν παράμετροι ή να επιστραφεί μια τιμή.

Για να δείξουμε πώς μπορούμε να χρησιμοποιήσουμε αυτή τη μέθοδο ο παρακάτω κώδικας δημιουργεί έναν πίνακα και αντιστρέφει έπειτα τη σειρά των στοιχείων του πίνακα:

```
var fruits= new Array("oranges","apples","pears");

fruits.reverse();
```

Η αρχική σειρά του πίνακα είναι oranges, apples, pears. Όταν καλείται η μέθοδος reverse() σε αυτόν τον πίνακα, ο πίνακας των στοιχείων αντιστρέφεται και ο πίνακας γίνεται τώρα pears, apples, oranges.

Η Μέθοδος shift()

Η μέθοδος shift() χρησιμοποιείται για να αφαιρεί το πρώτο στοιχείο του πίνακα και επιστρέφει την τιμή του στοιχείου που αφαιρέθηκε στην περίπτωση που πρέπει να την χρησιμοποιήσουμε αργότερα στο script.

Έτσι, εάν θέλουμε να αφαιρέσουμε το πρώτο στοιχείο ενός πίνακα θα μπορούσαμε να χρησιμοποιήσουμε κάτι παρόμοιο με τον παρακάτω κώδικα:

```
var fruits= new Array("oranges","apples","pears");

 fruits.shift();
```

Αυτός ο κώδικας δημιουργεί έναν πίνακα με τρία στοιχεία (oranges, apples, pears) και αφαιρεί το πρώτο στοιχείο με τη μέθοδο shift(). Αυτό αναγκάζει τον πίνακα να έχει μόνο δυο στοιχεία (apples, pears).

Για να χρησιμοποιήσουμε την τιμή του στοιχείου που αφαιρέθηκε μπορούμε να εκχωρήσουμε το αποτέλεσμα της μεθόδου σε μια μεταβλητή. Ο παρακάτω κώδικας εκχωρεί το στοιχείο που αφαιρέθηκε σε μια μεταβλητή και στέλνει ένα μήνυμα στον επισκέπτη με αυτήν την τιμή:

```
var fruits= new Array("oranges","apples","pears");

var picked_fruit=fruits.shift();

window.alert("You picked my "+picked_fruit);
```

Αυτός ο κώδικας εμφανίζει το μήνυμα "You picked my oranges” στο παράθυρο του browser.

Η Μέθοδος unshift()

Η μέθοδος unshift() χρησιμοποιείται για να προσθέτει στοιχεία στην αρχή του πίνακα. Τα στοιχεία που θέλουμε να προσθέσουμε στον πίνακα στέλνονται ως παράμετροι στη μέθοδο. Η τιμή που επιστρέφεται από τη μέθοδο είναι η αριθμητική τιμή του νέου μήκους του πίνακα.

Το παρακάτω παράδειγμα προσθέτει ένα νέο στοιχείο στην αρχή ενός πίνακα:

```
var fruits= new Array("apples","pears");

fruits.unshift("oranges");
```

Αυτό δημιουργεί έναν πίνακα, που ονομάζεται fruits με δυο στοιχεία (apples ,pears) και προσθέτει έπειτα ένα στοιχείο στην αρχή του πίνακα χρησιμοποιώντας τη μέθοδο unshift(). Ο πίνακας θα περιέχει έπειτα τρία στοιχεία (oranges, apples, pears).

Η Μέθοδος slice()

Η μέθοδος slice() χρησιμοποιείται για να αφαιρεί μια συγκεκριμένη ενότητα ενός πίνακα και έπειτα να δημιουργήσει έναν νέο πίνακα χρησιμοποιώντας τα στοιχεία από την τεμαχισμένη ενότητα του παλιού πίνακα.

Το παρακάτω είναι η γενική σύνταξη της κλήσης αυτής της μεθόδου:

```
arrayname.slice(start,  stop)
```

Θα πρέπει να αντικαταστήσουμε το arrayname με το όνομα του πίνακα, από τον οποίο θέλουμε να αφαιρέσουμε ένα ορισμένο σύνολο στοιχείων ώστε να μπουν σε έναν νέο πίνακα. Θα αντικαταστήσουμε το start με τον αριθμό δείκτη από τον οποίο θέλουμε να αρχίζει ο τεμαχισμός. Θα αντικαταστήσουμε το stop

με τον αριθμό δείκτη που έρχεται μετά το τελευταίο στοιχείο που θέλουμε να τεμαχίσουμε.

Για παράδειγμα, ο παρακάτω κώδικας τεμαχίζει δύο στοιχεία από έναν πίνακα και δημιουργεί έναν νέο πίνακα με αυτά τα στοιχεία:

```
var fruits= new Array("oranges","apples","pears","grapes");

var somefrults=fruits.slice(1,3);
```

Η Μέθοδος splice()

Η μέθοδος splice() μας επιτρέπει να αφαιρούμε ή να αντικαθιστούμε στοιχεία ενός πίνακα. Οι παράμετροι που μπορούν να σταλούν περιλαμβάνουν τον αριθμό δείκτη, από τον οποίο θα αρχίσει η διαδικασία, ο αριθμός των στοιχείων που θα αφαιρεθούν και την επιλογή να προστεθούν νέα στοιχεία στον πίνακα.

Εάν θέλουμε να αφαιρέσουμε ένα μόνο στοιχείο από έναν πίνακα, θα μπορούσαμε να χρησιμοποιήσουμε τον παρακάτω κώδικα:

```
var fruits= new Array("oranges","apples","pears","grapes");

 var somefruits=fruits.splice(2,1);
```

Εάν θέλουμε να χρησιμοποιήσουμε τη μέθοδο splice() προκειμένου να προσθέσουμε ένα ή περισσότερα στοιχεία σε έναν πίνακα, αλλά να μην αφαιρέσουμε τίποτα, μπορούμε να ορίσουμε τη δεύτερη παράμετρο σε 0 (αφαιρώντας έτσι μηδέν στοιχεία). Ορίζουμε την πρώτη παράμετρο στον αριθμό δείκτη, στον οποίο θέλουμε να αρχίσουμε να προσθέσουμε στοιχεία. Για παράδειγμα, δίνεται ο παρακάτω κώδικας:

```
var fruits= new Array("oranges","apples","pears","grapes");

 var   somefruits=fruits.splice(2,0,"watermelons","plums");
```

Αυτήν τη φορά η προσθήκη των στοιχείων αρχίζει στον αριθμό δείκτη καθορίζεται από την πρώτη παράμετρο και δεν αφαιρείται τίποτα, όπως καθορίζεται από τη δεύτερη παράμετρο. Προστίθενται δύο στοιχεία, οπότε ο πίνακας θα έχει μετά έξι στοιχεία (oranges, apples, watermelons, plums, grapes).

Η Μέθοδος sort()

Η μέθοδος sort() ταξινομεί έναν πίνακα κατά αλφαβητική σειρά (όπως ένα κατάλογο). Ωστόσο, αυτή δεν είναι αριθμητική σειρά,. Για παράδειγμα εάν θέλουμε να αλλάξουμε τη σειρά των στοιχείων του πίνακα fruits, ώστε να είναι σε αλφαβητική σειρά, θα μπορούσαμε να χρησιμοποιήσουμε τον παρακάτω κώδικα:

```
var fruits= new Array("oranges”, “apples”, “pears”, “grapes”);

fruits.sort();
```

Αυτός ο κώδικας θα αναδιατάξει τον πίνακα έτσι, ώστε τα στοιχεία του να είναι σε αλφαβητική σειρά, οπότε η σειρά θα αλλάξει από oranges, apples, pears, grapes σε apples, grapes, oranges, pears.

ΚΕΦΑΛΑΙΟ 9: ΟΙ ΣΥΜΒΟΛΟΣΕΙΡΕΣ

9.1 Τι είναι συμβολοσειρά

Το αντικείμενο String παρέχει ιδιότητες και μεθόδους για να πάρουμε πληροφορίες για συμβολοσειρές ή για να τροποποιήσουμε συμβολοσειρές. Ένα αντικείμενο string δημιουργείται από έναν προγραμματιστή χρησιμοποιώντας τη λέξη-κλειδί new με τη συνάρτηση δημιουργίας. Η σύνταξη είναι η εξής:

var instance_name = new String("η τιμή της συμβολοσειράς μπαίνει εδώ");

Για να το χρησιμοποιήσουμε αντικαθιστούμε το instance_name με το όνομα που θέλουμε να χρησιμοποιήσουμε για το στιγμιότυπο του αντικειμένου String.

9.2 Οι ιδιότητες του αντικειμένου string

Το αντικείμενο String έχει μόνο τρεις ιδιότητες. Ο Πίνακας 16 παρέχει μια συνοπτική περιγραφή των ιδιοτήτων, ενώ ακολουθεί εκτενέστερη περιγραφή τους.

Ιδιότητα	**Σκοπός**
constructor	Περιέχει την τιμή της συνάρτησης δημιουργίας για ένα στιγμιότυπο του αντικειμένου
length	Περιέχει την αριθμητική τιμή του μήκους της συμβολοσειράς (τον αριθμό των χαρακτήρων του)
prototype	Μας επιτρέπει να προσθέσουμε ιδιότητες στο αντικείμενο

ΠΙΝΑΚΑΣ 16 Οι ιδιότητες του αντικειμένου string

Η Ιδιότητα constructor

Αυτή η ιδιότητα επιστρέφει την τιμή της συνάρτησης δημιουργίας. Δίνεται ο παρακάτω κώδικας:

<html>

```
<head>
<script language="JavaScript">
var guitar_string= new String( "G" );
</script>
<body>
<script language="JavaScript">
document.write(guitar_string.constructor);
<script>
</body>
</html>
```

Η Ιδιότητα length

Αυτή η ιδιότητα επιστρέφει το μήκος της συμβολοσειράς, που είναι ο αριθμός των χαρακτήρων οι οποίοι περιλαμβάνονται στη συμβολοσειρά. Δίνεται ο παρακάτω κώδικας:

```
<html>
<head>
<script language="JavaScript">
var myname="John";
</script>
 <body>
<script language="JavaScript">
document.write("The name has "+myname.length+" characters.”);
</script>
```

</body>

</html>

Η Ιδιότητα prototype

Μπορούμε να χρησιμοποιήσουμε την ιδιότητα prototype για να προσθέσουμε ιδιότητες ή μεθόδους στα αντικείμενα String της σελίδας. Ο παρακάτω κώδικας δείχνει ένα παράδειγμα:

string .prototype. attitude="cool";

var rightnow= new String("Joe") ;

window.alert("This string is "+rightnow.attitude);

Τώρα το αντικείμενο String "Joe" έχει μια ιδιότητα attitude ίση με “cool”!.

9.3 Οι μέθοδοι του αντικειμένου string

Το αντικείμενο String έχει πολλές μεθόδους. Ο πίνακας 17 περιέχει τις μεθόδους και ακολουθεί μια σύντομη περιγραφή.

Μέθοδος	**Σκοπός**
anchor()	Δημιουργεί μια HTML ετικέτα σύνδεσης που έχει προορισμό μια σελίδα
big()	Προσθέτει ετικέτες <big> και </big> γύρω από μια συμβολοσειρά
blink()	Προσθέτει ετικέτες <blink> και </blink> γύρω από μια συμβολοσειρά (μπορεί να δουλεύει μόνο στον Netscape Navigator)
bold()	Προσθέτει ετικέτες και γύρω από μια συμβολοσειρά

Μέθοδος	**Σκοπός**
charAt()	Ανακαλύπτει ποιος χαρακτήρας είναι σε μια συγκεκριμένη θέση σε μια συμβολοσειρά
charCodeAt()	Βρίσκει τον κωδικό ενός χαρακτήρα σε μια συγκεκριμένη θέση σε μια συμβολοσειρά
concat()	Προσθέτει δύο ή περισσότερες συμβολοσειρές μαζί και επιστρέφει τη νέα συνενωμένη τιμή της συμβολοσειράς
fixed()	Προσθέτει <tt> και </tt> ετικέτες γύρω από μια συμβολοσειρά
fontcolor()	Προσθέτει ετικέτες και γύρω από μια συμβολοσειρά, που αλλάζουν το χρώμα της συμβολοσειράς σε ένα συγκεκριμένο χρώμα
fontsize()	Προσθέτει ετικέτες και γύρω από μια συμβολοσειρά, που αλλάζουν το μέγεθος της συμβολοσειράς σε ένα συγκεκριμένο μέγεθος που δίνεται από το number
fromCharCode()	Χρησιμοποιεί κωδικούς χαρακτήρων που στέλνονται ως παράμετροι για να δημιουργήσουν μια νέα συμβολοσειρά
indexOf()	Ψάχνει για ένα χαρακτήρα που στέλνεται ως παράμετρος σε μια συμβολοσειρά: εάν τον βρει, επιστρέφεται η θέση του πρώτου στιγμιότυπου του χαρακτήρα, διαφορετικά, επιστρέφει -1
italics()	Προσθέτει ετικέτες και γύρω από μια συμβολοσειρά

Μέθοδος	**Σκοπός**
lastIndexOf()	Ψάχνει για ένα χαρακτήρα που στέλνεται ως παράμετρος σε μια συμβολοσειρά: εάν τον βρει, επιστρέφεται η θέση του τελευταίου στιγμιότυπου του χαρακτήρα, διαφορετικά, επιστρέφει -1
link()	Δημιουργεί HTML συνδέσεις χρησιμοποιώντας τη συμβολοσειρά ως το κείμενο σύνδεσης και συνδέεσαι με το url που στέλνεται ως παράμετρος
match()	Συγκρίνει μια κανονική παράσταση και μια συμβολοσειρά για να δει εάν ταιριάζουν
replace()	Βρίσκει εάν μια κανονική παράσταση ταιριάζει με μια συμβολοσειρά και αντικαθιστά έπειτα την συμβολοσειρά που βρίσκει με μια νέα συμβολοσειρά
search()	Εκτελεί την αναζήτηση για ένα ταίριασμα μεταξύ μιας κανονικής παράστασης και μιας συγκεκριμένης συμβολοσειράς
slice()	Εξάγει ένα συγκεκριμένο τμήμα μιας συμβολοσειράς και το επιστρέφει σαν μια νέα συμβολοσειρά
small()	Προσθέτει ετικέτες και γύρω από μια συμβολοσειρά
split()	Χωρίζει μια συμβολοσειρά σε έναν πίνακα συμβολοσειρών με βάση έναν χαρακτήρα που στέλνεται ως παράμετρος στη μέθοδο
strike()	Προσθέτει ετικέτες <strike> και </strike> γύρω από μια

Μέθοδος	**Σκοπός**
	συμβολοσειρά
sub()	Προσθέτει ετικέτες _{και} γύρω από μια συμβολοσειρά
substr()	Επιτρέπει να επιστραφεί ένα τμήμα της συμβολοσειράς που καθορίζεται από μια αρχική θέση και από μια τελική θέση μετά από ένα συγκεκριμένο αριθμό χαρακτήρων
substring()	Επιτρέπει να επιστραφεί ένα τμήμα μιας συμβολοσειράς που κάθε αρχική θέση και από μια τελική θέση
sup()	Προσθέτει ετικέτες ^{και} γύρω από μια συμβολοσειρά
toString()	Επιστρέφει την κυριολεκτική συμβολοσειρά ενός αντικειμένου String
toLowerCase()	Μετατρέπει μια συμβολοσειρά σε πεζά γράμματα και επιστρέφει το αποτέλεσμα
toUpperCase()	Μετατρέπει μια συμβολοσειρά σε κεφαλαία γράμματα και επιστρέφει το αποτέλεσμα

ΠΙΝΑΚΑΣ 17 Οι ιδιότητες του αντικειμένου string

ΚΕΦΑΛΑΙΟ 10: ΟΙ ΧΕΙΡΙΣΤΕΣ ΣΥΜΒΑΝΤΩΝ

10.1 Τι είναι οι χειριστές συμβάντων

Ένας χειριστής συμβάντος (event handler) είναι μια προκαθορισμένη λέξη-κλειδί της JavaScript που χρησιμοποιείται για τον χειρισμό ενός συμβάντος σε μια ιστοσελίδα. Συνήθως, ένα συμβάν είναι κάτι που συμβαίνει όταν ο επισκέπτη της σελίδας εκτελεί κάποια ενέργεια, όπως να κάνει κλικ με ένα κουμπί του ποντικιού, να κάνει κλικ σε ένα κουμπί της σελίδας, να αλλάξει τα περιεχόμενα ενός στοιχείου μιας φόρμας ή να μετακινήσει το ποντίκι πάνω από μια υπερσύνδεση της σελίδας.

Όταν λαμβάνουν χώρα συμβάντα όπως αυτά, μπορούμε να χρησιμοποιήσουμε τους χειριστές συμβάντων της JavaScript για να τα προσδιορίσουμε και να τα εκτελέσουμε έπειτα με μια συγκεκριμένη εργασία ή ένα σύνολο εργασιών.

Οι χειριστές συμβάντων είναι χρήσιμοι επειδή μας επιτρέπουν να αποκτήσουμε πρόσβαση σε συμβάντα που μπορεί να εμφανιστούν στη σελίδα. Για παράδειγμα εάν θέλουμε να εμφανίσουμε ένα μήνυμα στον επισκέπτη όταν μετακινεί το ποντίκι του πάνω από μια σύνδεση, θα μπορούσαμε να καλέσουμε τα μηνύματα της JavaScript μέσω του χειριστή συμβάντος, στον οποίο να έχουμε γράψει τον κώδικα για να αντιδρά στο συμβάν. Τώρα μπορούν να συμβαίνουν πράγματα ανάλογα με τις ενέργειες του επισκέπτη, το οποίο μας επιτρέπει να κάνουμε τις ιστοσελίδες μας πιο διαλογικές.

Με τους χειριστές συμβάντων, μπορούμε να δημιουργήσουμε εκπλήξεις στον επισκέπτη ή να δημιουργήσουμε μερικά script, που θα προσθέσουν περισσότερη λειτουργικότητα στη σελίδα.

Οι χειριστές συμβάντων μπορούν να χρησιμοποιηθούν σε διάφορες θέσεις. Συνήθως χρησιμοποιούνται σε στοιχεία μιας φόρμας, σε ετικέτες συνδέσεων (<a>) και στην ετικέτα αρχής body.

Για να χρησιμοποιήσουμε έναν χειριστή συμβάντος, πρέπει να ξέρουμε τη λέξη-κλειδί του χειριστή συμβάντος και πού να τοποθετήσουμε το χειριστή συμβάντος στον HTML κώδικα. Για παράδειγμα, θα παρουσιαστεί ο χειριστής συμβάντος onClick, που χρησιμοποιείται για να κάνει κάτι να συμβεί όταν ο επισκέπτης κάνει κλικ σε μια συγκεκριμένη περιοχή του εγγράφου.

Μια από τις έγκυρες θέσεις που μπορεί να γίνει κλικ είναι ένα κουμπί μιας φόρμας. Έτσι, ας υποθέσουμε ότι θέλουμε να στείλουμε ένα μήνυμα στον επισκέπτη όταν κάνει κλικ σε ένα κουμπί μιας φόρμας. Σ' αυτή την περίπτωση, θα γράφαμε κάτι παρόμοιο με τον παρακάτω κώδικα:

<input type="button" onClick=" Ο κώδικας JavaScript μπαίνει εδώ ">

Για να χρησιμοποιήσουμε έναν χειριστή συμβάντος, τον προσθέτουμε ως μια πρόσθετη ιδιότητα σε μια HTML ετικέτα. Η μόνη διαφορά μεταξύ ενός χειριστή συμβάντος "attribute" (ιδιότητα) και μιας HTML ιδιότητας είναι ότι μπορούμε να προσθέσουμε κώδικα JavaScript μέσα σε μια ιδιότητα χειριστή συμβάντος αντί να προσθέσουμε απλώς μια τιμή ιδιότητας.

Η χρήση του χειριστή συμβάντος onClick ως μια ιδιότητα απαιτεί να χρησιμοποιήσουμε διπλά εισαγωγικά γύρω από όλο τον κώδικα JavaScript όταν χρειαστούμε εισαγωγικά για το μήνυμα, χρησιμοποιούμε μονά εισαγωγικά προκειμένου να αποφευχθούν πιθανά λάθη.

Επίσης παρατηρούμε ότι η εντολή alert τελειώνει με ένα ελληνικό ερωτηματικό. Αυτό μας επιτρέπει να προσθέσουμε συμπληρωματικό κώδικα JavaScript μετά το μήνυμα, πράγμα που μας επιτρέπει να εκτελέσουμε πολλές ενέργειες στο συμβάν κλικ αντί να βάλουμε μόνο μια πρόταση της JavaScript.

10.2 Παρουσίαση των Χειριστών Συμβάντων

Στην παρακάτω ενότητα γίνεται αναφορά στα είδη των χειριστών συμβάντων που χρησιμοποιούνται κάθε φορά στα συμβάντα μιας σελίδας. Ο

Πίνακας 18, αναφέρει τα πιο συνηθισμένα συμβάντα, τους χειριστές τους και δείγματα των ενεργειών, που μπορεί να προκαλέσει κάθε συμβάν.

Συμβάν	**Χειριστής Συμβάντος**	**Ενέργεια που το Προκαλεί**
Click	onClick	Ο επισκέπτης κάνει κλικ σε μια περιοχή (όπως σε ένα κουμπί ή σε μια περιοχή εισόδου μιας φόρμας)
Mouseover	onMouseOver	Ο επισκέπτης μετακινεί το ποντίκι πάνω από μια σύνδεση
Mouseout	onMouseOut	Ο επισκέπτης μετακινεί το ποντίκι μακριά από μια σύνδεση
Load	onLoad	Τελειώνει η φόρτωση της ιστοσελίδας
Unload	onUnload	Ο επισκέπτης αφήνει την τρέχουσα σελίδα
Focus	onFocus	Ο επισκέπτης εστιάζει σε κάτι
Blur	onBlur	Ο επισκέπτης απομακρύνει την εστίαση από κάτι
Change	onChange	Ο επισκέπτης αλλάζει τα περιεχόμενα ενός στοιχείου μιας φόρμας
Submit	onSubmit	Ο επισκέπτης υποβάλλει μια φόρμα της σελίδας

Συμβάν	**Χειριστής Συμβάντος**	**Ενέργεια που το Προκαλεί**
Abort	on Abort	Ο επισκέπτης σταματά τη φόρτωση μιας εικόνας και μπορεί να τοποθετηθεί σε μια ετικέτα εικόνας (img) ενός HTML εγγράφου
Error	onError	Εμφανίζεται ένα λάθος στη φόρτωση μιας σελίδας ή μιας εικόνας και μπορεί να χρησιμοποιηθεί σε μια ετικέτα img, object ή style μιας σελίδας
Keydown	onKeyDown	Ο επισκέπτης πατά ένα πλήκτρο στο πληκτρολόγιο. Μπορούμε να τοποθετήσουμε αυτόν τον χειριστή συμβάντος σχεδόν σε οποιαδήποτε HTML ετικέτα μιας ιστοσελίδας
Keypress	onKeyPress	Ο επισκέπτης πατά ένα πλήκτρο στο πληκτρολόγιο και το απελευθερώνει ή το κρατά πατημένο και μπορεί να τοποθετηθεί σχεδόν σε οποιαδήποτε HTML ετικέτα μιας ιστοσελίδας.
Keyup	onKeyUp	Ο επισκέπτης ελευθερώνει ένα πλήκτρο στο πληκτρολόγιο και μπορεί να τοποθετηθεί σχεδόν σε οποιαδήποτε HTML ετικέτα μιας

Συμβάν	**Χειριστής Συμβάντος**	**Ενέργεια που το Προκαλεί**
		ιστοσελίδας.
Mousedown	onMouseDown	Ο επισκέπτης πατά το κουμπί του ποντικιού και μπορεί να τοποθετηθεί μέσα σε μια ετικέτα σύνδεσης μιας ιστοσελίδας ή σε άλλες περιοχές.
Mouseup	onMouseUp	Ο επισκέπτης ελευθερώνει το κουμπί του ποντικιού και μπορεί να τοποθετηθεί σε μια ετικέτα σύνδεσης (αγκύρωση) μιας σελίδας ή σε άλλες περιοχές (όπως σε ετικέτες DIV) σε μερικούς νεότερους browser.
Mousemove	onMouseMove	Ο επισκέπτης μετακινεί το ποντίκι (μετακινεί τον δρομέα) και χρησιμοποιείται συνήθως στην ετικέτα body αρχής της σελίδας. Μπορεί επίσης να χρησιμοποιηθεί και σε διάφορες άλλες ετικέτες μιας σελίδας (όπως σε ετικέτες div).
Move	onMove	Μετακινείται ένα παράθυρο και μπορεί τοποθετηθεί σχεδόν σε οποιαδήποτε HTML ετικέτα μιας

Συμβάν	**Χειριστής Συμβάντος**	**Ενέργεια που το Προκαλεί**
		ιστοσελίδας.
Reset	on Reset	Ο επισκέπτης επαναφέρει μια φόρμα της σελίδας στην αρχική της μορφή και ο χειριστής συμβάντος προστίθεται στην ετικέτα form αρχής μιας φόρμας. Ο χειριστής συμβάντος onReset καλεί επίσης μια σχετική μέθοδο, που ονομάζεται reset().
Resize	onResize	Αλλάζει μέγεθος ένα παράθυρο και μπορεί να χρησιμοποιηθεί σχεδόν σε οποιαδήποτε HTML ετικέτα μιας ιστοσελίδας.
Select	onSelect	Ο χρήστης κάνει μια επιλογή σε ένα πεδίο μιας φόρμας και μπορεί να τοποθετηθεί μέσα σε μια ετικέτα κειμένου input σε μια ετικέτα κειμένου area ή στην ετικέτα αρχής body ενός HTML εγγράφου.

ΠΙΝΑΚΑΣ 18 Τα Συμβάντα και οι Χειριστές Συμβάντων

Το Συμβάν Click (onClick)

Το συμβάν Click λαμβάνει χώρα όταν ένας επισκέπτης κάνει κλικ σε ορισμένες περιοχές μιας ιστοσελίδας. Αυτές οι περιοχές βρίσκονται σε στοιχεία μιας φόρμας και σε συνδέσεις.

Ο ευκολότερος τρόπος για να δούμε το συμβάν κλικ σε δράση είναι να χρησιμοποιήσουμε ένα κουμπί φόρμας. Όταν γίνεται κλικ στο κουμπί, θέλουμε να εμφανιστεί ένα συμβάν. Για να συμβεί αυτό, πρέπει να τοποθετήσουμε το χειριστή συμβάντος onClick μέσα στην ετικέτα εισόδου του κουμπιού, όπως στο παρακάτω παράδειγμα:

```
<html>
<head>
<title>onClick</title>
</head>
 <body>
  <form>
<input type="button" value="Do not Click Here"
onClick="window.alert('I told you not to click me!');">
</form>
</body>
</html>
```

Αυτό θα στείλει στον επισκέπτη ένα μήνυμα μόλις κάνει κλικ στο κουμπί. Η εικόνα 8 μας δείχνει το αποτέλεσμα αυτού του κώδικα όταν ο επισκέπτης κάνει κλικ στο κουμπί.

ΕΙΚΟΝΑ 8 Αυτό το μήνυμα εμφανίζεται όταν ο επισκέπτης κάνει κλικ στο κουμπί

Συμβάν Mouseover (onMouseOver)

Το συμβάν mouseover λαμβάνει χώρα όταν ένας επισκέπτης μετακινεί το δρομέα του ποντικιού πάνω από μια περιοχή της σελίδας, όπως πάνω από μια σύνδεση κειμένου, μια συνδεμένη εικόνα, ή ένα συνδεμένο τμήμα ενός χάρτη εικόνας. Το συμβάν mouseover θα δουλέψει επίσης και σε πολλές άλλες περιοχές όπως σε κελιά ενός πίνακα και σε ετικέτες και . Το συμβάν mouseover το χειρίζεται ο χειριστής συμβάντος onMouseOver.

Ο ταχύτερος τρόπος να χρησιμοποιήσουμε ένα χειριστή συμβάντος onMouseOver είναι να δημιουργήσουμε μια σύνδεση κειμένου. Όταν προσθέσουμε το χειριστή συμβάντος onMouseOver στη σύνδεση, έχουμε την επιλογή να εκτελέσουμε εντολές της JavaScript όταν ο επισκέπτης περνά το δρομέα πάνω από τη σύνδεση. Κατά συνέπεια, εάν θέλουμε να εμφανισθεί ένα μήνυμα όταν ο επισκέπτης μετακινεί το ποντίκι πάνω από μια σύνδεση, θα μπορούσαμε να γράψουμε κώδικα παρόμοιο με τον εξής:

**

onMouseOver="window.alert('I told you not to try to click me!');">

Don't Try Clicking Me!

Το Συμβάν Mouseout (onMouseOut)

Το συμβάν mouseout εμφανίζεται όταν ο επισκέπτης μετακινεί το δρομέα του ποντικιού μακριά από μια περιοχή της σελίδας, όπως από μια σύνδεση, μια συνδεμένη εικόνα ή μια συνδεμένη περιοχή ενός χάρτη εικόνας. Όπως και το συμβάν mouseover, έτσι και εδώ οι περισσότεροι browser υποστηρίζουν το συμβάν mouseout σε πολλές περιοχές. Χειριζόμαστε ένα συμβάν mouseout χρησιμοποιώντας τον χειριστή συμβάντος onMouseOut.

Και πάλι, μπορούμε να δούμε το συμβάν mouseout πιο εύκολα δημιουργώντας μια σύνδεση και προσθέτοντας στον χειριστή συμβάντος onMouseOut σύνδεση. Αυτήν τη φορά, μπορούμε να εμφανίσουμε ένα μήνυμα μόλις μετακινήσει ο χρήστης το ποντίκι μακριά από τη σύνδεση, αφού περάσει πάνω της (υποθέτοντας ότι δεν έχει κάνει κλικ). Για να το κάνουμε αυτό, θα μπορούσαμε να χρησιμοποιήσουμε τον παρακάτω κώδικα:

**

onMouseOut="window.alert('What, you didn\'t like my link?')

Click Me!

Το Συμβάν Load (onLoad)

Το συμβάν load εμφανίζεται όταν τελειώσει η φόρτωση μιας ιστοσελίδας. Για να χειριστούμε αυτό το συμβάν, χρησιμοποιούμε τον χειριστή συμβάντος onLoad και τοποθετούμε στην ετικέτα αρχής BODY μιας ιστοσελίδας. Θα λάβουμε υπόψη μας ότι το συμβάν load εμφανίζεται σε έναν λίγο διαφορετικό χρόνο από τα script μηνυμάτων που τοποθετήσαμε στην ενότητα HEAD των σελίδων. Με το συμβάν load, ωστόσο, οι εργασίες εκτελούνται μόλις η σελίδα τελειώσει τη διαδικασία φόρτωσης. Εάν θέλουμε να εμφανιστεί ένα μήνυμα

όταν τελειώσει η φόρτωση της σελίδας, θα μπορούσαμε να χρησιμοποιήσουμε τον παρακάτω κώδικα:

```
<body  onLoad="window.alert('I\'m done loading now!');">
<B>Text for the body of the page</B>
</body>
```

Το Συμβάν Unload (onUnload)

Το συμβάν unload συμβαίνει όταν αφήνει ένας επισκέπτης την τρέχουσα σελίδα. Ο επισκέπτης μπορεί να φύγει κάνοντας κλικ σε μια σύνδεση, πληκτρολογώντας μια άλλη διεύθυνση στον browser ή κλείνοντας το παράθυρο. Ο χειριστής συμβάντος που χρησιμοποιείται για το συμβάν unload είναι ο onUnload και τοποθετείται στην ετικέτα αρχής body μιας ιστοσελίδας.

Αυτό το συμβάν είναι γνωστό ότι συνήθως ενοχλεί τους επισκέπτες, επειδή επιτρέπει στον ιδιοκτήτη της τοποθεσίας να κάνει κάτι ενώ οι επισκέπτες προσπαθούν να μετακινηθούν προς μια άλλη σελίδα ή προς μια άλλη Web τοποθεσία (αναγκάζοντας τους να περιμένουν). Για να εμφανιστεί ένα μήνυμα όταν ο χρήστης αφήνει τη σελίδα, θα μπορούσαμε να γράψουμε τον παρακάτω κώδικα:

```
<body
 onUnload="window.alert('Be sure to come back, OK?');">
<b>Other HTML code goes here</b>
</body>
```

Το Συμβάν Focus (onFocus)

Το συμβάν focus εμφανίζεται όταν ο επισκέπτης εστιάζει σε ένα παράθυρο ή ένα στοιχείο φόρμας μιας ιστοσελίδας. Ένας επισκέπτης εστιάζει σε κάτι

κάνοντας κλικ κάπου μέσα στο στοιχείο, χρησιμοποιώντας το πληκτρολόγιο για να μετακινηθεί προς το στοιχείο (συνήθως μέσω του πλήκτρου TAB) ή μέσω script. Για παράδειγμα, ένας επισκέπτης ο οποίος κάνει κλικ σε ένα πλαίσιο εισαγωγής κειμένου (πριν εισάγει οτιδήποτε) δίνει την εστίαση σε αυτό το πλαίσιο-κειμένου. Επίσης, το κλικ σε ένα ανενεργό παράθυρο και η μετατροπή του σε ενεργό παράθυρο θα δώσουν στο παράθυρο την εστίαση.

Ο χειριστής συμβάντος που χρησιμοποιείται με αυτό το συμβάν είναι ο onFocus και μπορεί να χρησιμοποιηθεί σε ένα στοιχείο φόρμας ή στην ετικέτα body αρχής μιας ιστοσελίδας (για εστίαση σε ένα παράθυρο).

Για να δούμε το συμβάν focus σε δράση, μπορούμε να δημιουργήσουμε ένα πλαίσιο εισαγωγής κειμένου, το οποίο είναι ένα από τα στοιχεία μιας φόρμας θα μας επιτρέψει να δώσουμε την εστίαση στο στοιχείο. Το παρακάτω παράδειγμα μας δείχνει πώς να κάνουμε αυτό, καθώς επίσης και πώς να γράψουμε τον κώδικα για να εμφανίσουμε ένα πλαίσιο υπενθύμισης όταν ο επισκέπτης δίνει την εστίαση στο πλαίσιο κειμένου:

```
<form>

Enter your Name:

<input type= “text” onFocus="window.alert('Don\’t forget to capitalize!' );">

</form>
```

Το Συμβάν Blur (onBlur)

Το συμβάν blur είναι το αντίθετο του συμβάντος focus και εμφανίζεται όταν ο επισκέπτης απομακρύνει την εστίαση από ένα στοιχείο μιας φόρμας ή από ένα παράθυρο. Για να απομακρύνει την εστίαση από κάτι, ο επισκέπτης εστιάζει σε κάτι άλλο. Για παράδειγμα, ο επισκέπτης θα μπορούσε να μετακινηθεί από ένα στοιχείο μιας φόρμας προς ένα άλλο ή από ένα παράθυρο προς ένα άλλο.

Ο χειριστής συμβάντος onBlur χειρίζεται αυτό το συμβάν και μπορεί να χρησιμοποιηθεί στην ετικέτα ενός στοιχείου μιας φόρμας ή στην ετικέτα BODY (σε παράθυρα).

Για να δούμε το συμβάν blur σε δράση, μπορούμε επίσης να χρησιμοποιήσουμε το πλαίσιο κειμένου για το χειριστή συμβάντος onBlur. Το παρακάτω παράδειγμα χρησιμοποιεί δυο πλαίσια κειμένου: το κλικ στο πρώτο πλαίσιο κειμένου σου δίνει την εστίαση και το κλικ στο δεύτερο πλαίσιο κειμένου καλεί το συμβάν blur στο πρώτο πλαίσιο κειμένου.

```
<form>
Give this box focus: <br>
<input  type= “text” onblur= “window.alert ( ‘hey! Come Back!’);”>
<br>
Then give this box focus to blur the first one :<br>
<input type="text">
</form>
```

Όταν οι επισκέπτες κάνουν κλικ στο δεύτερο πλαίσιο κειμένου, βλέπουν το μήνυμα από το πρώτο πλαίσιο κειμένου που τους "λέει" να επισκεφτούν πάλι την τοποθεσία.

Το συμβάν Change (onChange)

Το συμβάν change εμφανίζεται όταν αλλάζει ένας επισκέπτης κάτι μέσα σε ένα στοιχείο μιας φόρμας. Για παράδειγμα, ο επισκέπτης μπορεί να αλλάξει το κείμενο σε ένα πλαίσιο κειμένου ή να κάνει μια επιλογή από ένα πλαίσιο επιλογής.

Μπορούμε να δούμε πώς δουλεύει αυτό δημιουργώντας ένα πλαίσιο επιλογής. Μπορούμε να δώσουμε στον επισκέπτη μερικές επιλογές μέσα στο

πλαίσιο επιλογής. Εάν ο χρήστης αλλάξει την προκαθορισμένη επιλογή επιλέγοντας κάτι άλλο, στέλνουμε ένα μήνυμα για να τον ρωτήσουμε γιατί την άλλαξε, όπως φαίνεται στο παρακάτω παράδειγμα:

```
<html>
<body>
<form>
Are you cool?<br>
<select onChange="window.alert( 'Why did you change that?');”>
 <option selected>Yes</option>
 <option>No</option>
 <option>Undecided</option>
</select>
</form>
</body>
</html>
```

Εάν ο επισκέπτης προσπαθήσει να αλλάξει την προκαθορισμένη απάντηση Yes, εμφανίζεται ένα πλαίσιο που μεταφέρει στον επισκέπτη ένα μήνυμα. Εάν το πλαίσιο επιλογής δεν πειραχτεί ή αν ο επισκέπτης επιλέξει την πρώτη επιλογή, τίποτα δεν θα συμβεί.

Το συμβάν change είναι πολύ χρήσιμο για δημιουργία πλαισίων κειμένου πλοήγησης.

Το Συμβάν Submit (onSubmit)

Το συμβάν submit εμφανίζεται μόνο όταν στέλνει ο επισκέπτης μια φόρμα από μια ιστοσελίδα. Αυτό το συμβάν χρησιμοποιεί το χειριστή συμβάντος ο

οποίος μπορεί να κληθεί από μια ετικέτα form αρχής ενός εγγράφου. Ο χειριστής συμβάντος onSubmit καλεί, επίσης, μια σχετική μέθοδο submit().

Για να δούμε το συμβάν submit σε δράση, πρέπει να δημιουργήσουμε μια φόρμα που να μπορεί να σταλεί με το κουμπί submit. Θα πρέπει έπειτα να προσθέσουμε το χειριστή συμβάντος onSubmit στην ετικέτα αρχής form. Ο κώδικας θα εμφανίσει ένα μήνυμα "Thank You" στον επισκέπτη όταν γίνει κλικ στο κουμπί submit.

```
<html>

<body>

<form onSubmit="window.alert('Thank You');">

 What's your name?<br>

<input type="text" name="thename">

<br>

<input type="submit">

</form>

</body>

</html>
```

10.3 Εφαρμογές Πάνω στους Χειριστές Συμβάντων

10.3.1 Προσθέτοντας περιγραφές συνδέσεων σε μια ιστοσελίδα

Μία από τις πιο συνηθισμένες χρήσεις ενός χειριστή συμβάντος είναι να εμφανίζουν περιγραφές των συνδέσεων όταν ο χρήστης μετακινεί το ποντίκι πάνω τους. Για παράδειγμα, η μετακίνηση του ποντικιού πάνω από τη σύνδεση Order Form μπορεί να εμφανίσει ένα μήνυμα όπως "Order a product or check an order's status" (Δώστε μια παραγγελία για ένα προϊόν ή ελέγξτε την κατάσταση μιας παραγγελίας).

Οι περιγραφές τέτοιων συνδέσεων εμφανίζονται γενικά με το χειριστή συμβάντος onMouseOver. Θα δημιουργήσουμε ένα script που θα εμφανίζει μηνύματα με αυτόν τον τρόπο και θα καθαρίζει το μήνυμα χρησιμοποιώντας το χειριστή συμβάντος onMouseOut. Θα χρησιμοποιήσουμε συναρτήσεις για να απλοποιήσουμε αυτήν την διαδικασία.

Αυτό αποτελεί επίσης ένα παράδειγμα για τον ορισμό χειριστών συμβάντων εξ ολοκλήρου με JavaScript. Το HTML έγγραφο, που φαίνεται στη Λίστα 10.1, δεν περιλαμβάνει ετικέτες <script> ή χειριστές συμβάντων - το μόνο πράγμα που απαιτεί είναι μερικές ιδιότητες id στα αντικείμενα που χρησιμοποιούνται στο script.

Λίστα 10.1 Το Έγγραφο HTML για το Παράδειγμα με τις Περιγραφές των Συνδέσεων

```
<html>
<head>
   <title>Descriptive Links</title>
</head>
<body>
<h1>Descriptive Links</h1>
<p>Move the mouse pointer over one of these links to view a description:</p>
<ul>
   <li><a href="order.html" id="order">Order Form</a>
   <li><a href="email.html" id="email">Email</a>
   <li><a href="complain.html" id="complain">Complaint Department</a>
</ul>
```

```
<h2 id="description"></h2>
<script language="JavaScript" type="text/javascript" src="linkdesc.js">
</script>
</body>
</html>
```

Αυτό το έγγραφο ορίζει τρεις συνδέσεις σε μια λίστα με κουκκίδες. Κάθε ετικέτα <a> ορίζεται με μια id ιδιότητα που θα χρησιμοποιήσει το script για να επισυνάψει έναν χειριστή συμβάντος. Η ετικέτα <h2> με id τιμή description, που είναι αυτήν την στιγμή κενή, θα χρησιμοποιηθεί για να εμφανίσει μια περιγραφή κάθε σύνδεσης.

Το script θα αρχίσει με μια συνάρτηση που χρησιμεύει ως ο χειριστής συμβάντος onMouseOver για τις συνδέσεις:

```
function hover(e) {
    if (!e) var e = window.event;
     // σε ποια σύνδεση περνά το ποντίκι;
    whichlink = (e.target) ? e.target.id : e.srcElement.id;
    // επιλέγει την κατάλληλη περιγραφή
    if (whichlink=="order") desc = "Order a product";
    else if (whichlink=="email") desc = "Send us a message";
    else if (whichlink=="complain") desc = "Insult us, our products, or our families";
    // εμφάνιση της περιγραφής στην h2
    d = document.getElementById("description");
    d.innerHTML = desc;
}
```

Η συνάρτηση hover χρησιμοποιεί τις ιδιότητες target ή srcElement για να βρει το αντικείμενο προορισμού της σύνδεσης και βρίσκει έπειτα την ιδιότητα του id. Τρεις προτάσεις if υπολογίζουν το id και επιλέγουν μια κατάλληλη περιγραφή. Τέλος, το script χρησιμοποιεί τη μέθοδο getElementById() για να βρει την ετικέτα <h2> που θα εμφανίσει τις περιγραφές και εμφανίζει την περιγραφή χρησιμοποιώντας την ιδιότητα innerHTML.

Η πρόταση υπό όρους στην τρίτη γραμμή της συνάρτησης hover ελέγχει εάν υπάρχει η ιδιότητα target και εάν όχι, χρησιμοποιεί την ιδιότητα srcElement. Αυτό ονομάζεται αίσθηση λειτουργίας, που εντοπίζει εάν ο browser υποστηρίζει μια λειτουργία.

Θα απαιτηθεί μια ακόμη συνάρτηση. Η συνάρτηση cleardesc() θα χρησιμεύσει ως χειριστής συμβάντος onMouseOut και θα καθαρίσει την περιγραφή όταν το ποντίκι δεν θα είναι πλέον πάνω από μία από τις συνδέσεις.

```
function cleardesc() {
    d = document.getElementById("description");
    d.innerHTML = "";
}
```

Τώρα που ορίσθηκαν οι συναρτήσεις, πρέπει να τις αντιστοιχίσουμε ως χειριστές συμβάντων για τις συνδέσεις. Κάθε σύνδεση απαιτεί τις παρακάτω τρεις γραμμές κώδικα:

```
orderlink = document.getElementById("order");
orderlink.onmouseover=hover;
orderlink.onmouseout=cleardesc;
```

Αφού χρησιμοποιηθεί η getElementByld() για να βρει το αντικείμενο με την id ιδιότητα "order", ορίζει τις συναρτήσεις hover () και cleardesc() ως χειριστές

συμβάντων onMouseOver και onMouseOut. Αυτό θα πρέπει να επαναληφθεί και για τις άλλες δύο συνδέσεις. Βάζοντας όλα αυτά μαζί, το πλήρες JavaScript αρχείο γι' αυτό το παράδειγμα φαίνεται στη Λίστα 10.2.

Λίστα 10.2 Το JavaScript Αρχείο «linkdesc.js» για το Παράδειγμα με τις Περιγραφές των Συνδέσεων

```
function cleardesc() {
    d = document.getElementByld("description");
    d.innerHTML = "";
}
function hover(e) {
    if (!e) var e = window.event;
    // σε ποια σύνδεση περνά το ποντίκι;
    whichlink = (e.target) ? e.target.id : e.srcElement.id;
    //επιλέγει την κατάλληλη περιγραφή
    if (whichlink=="order") desc = "Order a product";
    else if (whichlink=="email") desc = "Send us a message";
    else if (whichlink=="complain") desc = "Insult us, our products, or our families";
    // εμφάνιση της περιγραφής στην H2
    d = document.getElementById("description");
    d.innerHTML = desc;
}
// Διαμόρφωση των χειριστών συμβάντων
orderlink = document.getElementById("order");
```

```
orderlink.onmouseover=hover;

orderlink.onmouseout=cleardesc;

emaillink = document.getElementById("email");

emaillink.onmouseover=hover;

emaillink.onmouseout=cleardesc;

complainlink = document.getElementById("complain");

complainlink.onmouseover=hover;

complainlink.onmouseout=cleardesc;
```

Για να ελέγξουμε αυτό το script, το αποθηκεύουμε ως linkdesc.js στον ίδιο φάκελο με το HTML έγγραφο και φορτώνουμε το HTML αρχείο σε έναν browser. Αυτό το script πρέπει να δουλεύει σε οποιονδήποτε browser που υποστηρίζει την JavaScript. Η οθόνη του Firefox φαίνεται στην Εικόνα 9.

ΕΙΚΟΝΑ 9 Οι περιγραφές των συνδέσεων σε μια σελίδα

10.3.2 Δημιουργία δυναμικών στυλ

Η HTML δεν έχει σκοπό να χειρίζεται πράγματα όπως η διάταξη, η στοίχιση και τα κενά ενός εγγράφου. Η HTML χειρίζεται τη δομή ενός εγγράφου,

δηλαδή πώς θα διαιρείται το έγγραφο σε παραγράφους, επικεφαλίδες, λίστες και άλλα στοιχεία.

Έτσι, το World Wide Web Consortium (W3C) συνειδητοποίησε ότι οι Web συντάκτες πρέπει να ελέγχουν την διάταξη και την παρουσίαση εγγράφων. Αυτό οδήγησε στο πρότυπο Cascading Style Sheets (CSS - φύλλα στυλ).

Οπότε μας δίνεται η δυνατότητα να δημιουργήσουμε μια σελίδα η οποία θα μας επιτρέπει να ελέγχουμε κατευθείαν τα χρώματα που χρησιμοποιούνται στο κείμενο της σελίδας. Οπότε θα χρειαστούμε μια φόρμα στην οποία να μπορούμε να επιλέγουμε χρώματα. Θα χρησιμοποιήσουμε ετικέτες <select> για να επιτρέψουμε μια επιλογή χρώματος:

```
<select name="heading" onChange="changehead();">
<option value="black">Black</option>
<option value="red">Red</option>
<option value="blue">Blue</option>
<option value="green">Green</option>
<option value="yellow">Yellow</option>
</select>
```

Ο ορισμός <select> χρησιμοποιεί τις ιδιότητες onChange στις ετικέτες για να καλέσει δύο συναρτήσεις, την changehead() και την changebody (), όταν αλλάζει η αντίστοιχη επιλογή τους.

Ο συνδυασμός δύο εξ αυτών των επιλογών με κάποιο απλό HTML κώδικα οδηγεί στο πλήρες HTML έγγραφο που φαίνεται στη Λίστα 10.3.

Λίστα 10.3 Το HTML αρχείο για το παράδειγμα με τα δυναμικά στυλ

```
<html>
```

```
<head>
<title>Controlling Styles with JavaScript</title>
<script language="Javascript" type="text/javascript"
src="styles.js">
 </script>
</head>
<body>
<h1 id="head1">Controlling Styles with JavaScript</h1> <hr>
<p id="p1">
```

Select the color for paragraphs and headings using the form below. The colors you specified will be dynamically changed in this document. The change occurs as soon as you change the value of either of the drop-down lists in the form. </p>

```
<form name="form1"> <b>Heading color: </b>
<select name="heading" onChange="changehead();">
    <option value="black">Black</option>
    <option value="red">Red</option>
    <option value="blue">Blue</option>
    <option value= " green" >Green< /option>
    <option value="yellow">Yellow</option>
    </select> <br>
<b>Body text color: </b>
<select name="body" onChange="changebody();">
    <option value="black">Black</option>
```

```
    <option value="red">Red</option>
    <option value="blue">Blue</option>
    <option value="green">Green</option>
    <option value="yellow">Yellow</option>
</select>
</form>
</body>
</html>
```

Παρατηρούμε ότι η ετικέτα <h1> έχει μια id ιδιότητα "head1" και η ετικέτα <p> έχει μια id ιδιότητα "p1". Αυτές είναι οι τιμές που θα χρησιμοποιήσει το script στη συνάρτηση getElementById(). Η ετικέτα <script> στην ενότητα <head> συνδέει το έγγραφο στο script styles.js, το οποίο θα δημιουργήσουμε στην συνέχεια.

Το HTML αρχείο θα το αποθηκεύσουμε ως styles.html. Μπορούμε να το ελέγξουμε τώρα σε έναν browser, αλλά οι δυναμικές λειτουργίες δεν θα δουλέψουν μέχρι να δημιουργήσουμε το JavaScript αρχείο που περιέχει τις συναρτήσεις με τα script. Η Λίστα 10.4 δείχνει τον κώδικα JavaScript αυτού του παραδείγματος.

Λίστα 10.4 Το JavaScript αρχείο για το παράδειγμα των δυναμικών στυλ

```
function changehead() {
    i = document.form1.heading.selectedIndex;
    headcolor = document.form1.heading.options[i].value;
    document.getElementById("head1").style.color = headcolor;
}
```

```
function changebody() {
    i = document.form1.body.selectedIndex;
    doccolor = document.form1.body.options [ i ].value;
    document.getElementById("p1").style.color = doccolor;
}
```

Αυτό το script ορίζει αρχικά τη συνάρτηση changehead(). Αυτή διαβάζει το δείκτη για το τρέχον επιλεγμένο χρώμα επικεφαλίδας και έπειτα διαβάζει την τιμή χρώματος του δείκτη. Αυτή η συνάρτηση χρησιμοποιεί τη μέθοδο getElementById() για να αλλάξει το χρώμα. Η συνάρτηση changebody χρησιμοποιεί την ίδια σύνταξη για να αλλάξει το χρώμα του σώματος του κειμένου.

ΕΙΚΟΝΑ 10 Δημιουργία δυναμικών στυλ με την JavaScript

Αποθηκεύουμε το JavaScript αρχείο ως styles.js στον ίδιο φάκελο με το HTML έγγραφο που αποθηκεύσαμε τη Λίστα 10.3. Για να ελέγξουμε το script με τα δυναμικά στυλ, φορτώνουμε τη Λίστα 10.3 (styles.html) στον browser. Επιλέγουμε το επιθυμητό χρώμα και παρατηρούμε την άμεση αλλαγή στην επικεφαλίδα ή στο σώμα της σελίδας. Η Εικόνα 10 δείχνει μια τυπική εμφάνιση αυτού του εγγράφου αφού αλλάξουν τα χρώματα του.

10.3.5 Rollover με JavaScript

Μια από τις κλασικές χρήσεις της JavaScript είναι να δημιουργούμε rollover, δηλαδή εικόνες που αλλάζουν όταν μετακινούμε το ποντίκι πάνω τους.

Γενικά χρησιμοποιούνται για να δημιουργούμε συνδέσεις πλοήγησης που δίνουν στο χρήστη κάποια καθοδήγηση φωτίζοντας αυτήν την σύνδεση, στην οποία το ποντίκι είναι πάνω της.

Όλο αυτό απαιτεί να συνδυάσουμε στην JavaScript έναν χειριστή συμβάντος onMouseOver με μια δυναμική εικόνα. Η προσθήκη του onMouseOut επιτρέπει στο script μας να επαναφέρει την αρχική εικόνα όταν απομακρύνεται το ποντίκι. Η Λίστα 10.5 δείχνει έναν απλό τρόπο να γίνει αυτό με ενσωματωμένους (inline) χειριστές συμβάντων.

Λίστα 10.5 Χρησιμοποιώντας Απλά JavaScript Rollover

```
<html>
<head>
<title> Rollovers - JavaScript </title>
</head>
<body>
<h1>JavaScript Rollovers</h1>
<a href="home.html"
onmouseover="document.images[0].src='home2.gif';"
onmouseout="document.images[0].src='home1.gif';">
<image border="0" src="home1.gif">
</a> <br>
<a href="archives.html"
onmouseover="document.images[1].src='archives2.gif'
onmouseout="document.images[1].src='archives1.gif;
<image border="0" src="archives1.gif">
```

```
</a>

</body>

</html>
```

Αυτό είναι απλώς ένα απλό τμήμα inline κώδικα JavaScript, έτσι μπορούμε να ελέγξουμε φορτώνοντας απλώς το HTML αρχείο σε ένα browser.

ΚΕΦΑΛΑΙΟ 11: Η JAVASCRIPT ΚΑΙ ΟΙ ΦΟΡΜΕΣ

11.1 Η πρόσβαση σε φόρμες

Οι φόρμες είναι μεταξύ των πιο χρήσιμων λειτουργιών της γλώσσας HTML. Η προσθήκη κώδικα JavaScript σε φόρμες μπορεί να τις κάνει πιο διαλογικές και να παρέχει διάφορες χρήσιμες λειτουργίες. Μπορούμε να χρησιμοποιήσουμε την JavaScript για να κάνουμε μία φόρμα πιο διαλογική, να επικυρώσουμε τα δεδομένα που εισάγει ο χρήστης και να εισάγουμε δεδομένα βασισμένα σε άλλα δεδομένα.

Μια HTML φόρμα αρχίζει με την ετικέτα <form>. Αυτή η ετικέτα υποδεικνύει ότι αρχίζει μια φόρμα και επιτρέπει να χρησιμοποιηθούν τα στοιχεία της φόρμας. Η ετικέτα <form> περιλαμβάνει διάφορες ιδιότητες:

- Η ιδιότητα *name* είναι απλώς ένα όνομα για την φόρμα.

- Η ιδιότητα *method* είναι είτε *GET* είτε *POST*. Αυτοί είναι οι δύο τρόποι που μπορούν να σταλούν τα δεδομένα στον διακομιστή.

- Η ιδιότητα *action* είναι το CGI script, στο οποίο θα σταλούν τα δεδομένα της φόρμας όταν σταλεί η φόρμα. Μπορούμε επίσης να χρησιμοποιήσουμε την ενέργεια mailto: για να στείλουμε τα αποτελέσματα της φόρμας σε μια διεύθυνση ηλεκτρονικού ταχυδρομείου. Κάνοντας μια παρένθεση, θα πρέπει να αναφέρουμε ότι το CGI (Common Gateway Interface) είναι ένα πρότυπο πρωτόκολλο για την εξωτερική διεπαφή λογισμικού εφαρμογών με έναν διακομιστή πληροφοριών, συνήθως έναν web server.

- Η ιδιότητα *enctype* είναι ο τύπος MIME, στον οποίο θα κωδικοποιηθούν τα δεδομένα της φόρμας. Αυτό συνήθως δεν είναι απαραίτητο.

Για παράδειγμα, η ετικέτα <form> για μια φόρμα που ονομάζεται Order χρησιμοποιεί τη μέθοδο GET και στέλνει τα δεδομένα της σε ένα CGI script

που ονομάζεται order. cgi και είναι στον ίδιο κατάλογο με την ίδια την ιστοσελίδα:

<form name="Order" method="GET" action="order.cgi">

Η ετικέτα <form> ακολουθείται από ένα ή περισσότερα στοιχεία της φόρμας. Αυτά είναι τα πεδία δεδομένων της φόρμας, όπως πεδία κειμένου, κουμπιά πλαίσια ελέγχου.

Πεδία κειμένου

Πιθανώς, τα πιο συχνά χρησιμοποιούμενα στοιχεία μιας φόρμας είναι τα πεδία κειμένου. Μπορούμε να τα χρησιμοποιήσουμε για να ζητήσουμε ένα όνομα, μια διεύθυνση ή οποιεσδήποτε άλλες πληροφορίες. Στην JavaScript, μπορούμε να εμφανίσουμε κείμενο στο πεδίο αυτόματα. Το παρακάτω αποτελεί ένα παράδειγμα ενός απλού πεδίου κειμένου:

<input type="text" name="text1" value="hello" size="30">

Κουμπιά

Ένας από τους πιο χρήσιμους τύπους στοιχείων μιας φόρμας είναι τα κουμπιά. Τα κουμπιά χρησιμοποιούν την ετικέτα <input> και μπορούν να χρησιμοποιήσουν έναν από τους τρεις διαφορετικούς τύπους:

- Το type=submit είναι ένα κουμπί Submit. Αυτό το κουμπί στέλνει τα δεδομένα των πεδίων της φόρμας στο CGI script.

- Το type=reset είναι ένα κουμπί Reset. Αυτό το κουμπί επαναφέρει όλα τα πεδία της φόρμας στην προκαθορισμένη τους τιμή, δηλαδή το κενό.

- Το type=button είναι ένα γενικό κουμπί. Αυτό το κουμπί δεν εκτελεί καμία ενέργεια από μόνο του, αλλά μπορούμε να του εφαρμόσουμε κάποια χρησιμοποιώντας έναν JavaScript χειριστή συμβάντος.

Και οι τρεις τύποι κουμπιών περιλαμβάνουν μια ιδιότητα name για να προσδιορίσουν το κουμπί και μια τιμή value που υποδεικνύει το κείμενο που θα εμφανιστεί πάνω στο κουμπί. Το παρακάτω παράδειγμα ορίζει ένα κουμπί Submit με όνομα sub1 και τιμή "Click Here":

```
<input type="submit" name="sub1" value="Click Here">
```

Αναπτυσσόμενες λίστες

Ένα τελευταίο στοιχείο φόρμας, που είναι επίσης χρήσιμο για πολλαπλές επιλογές, είναι οι αναπτυσσόμενες λίστες. Η HTML ετικέτα <select> χρησιμοποιείται για να ορίσουμε μια λίστα επιλογής ή μια αναπτυσσόμενη λίστα από στοιχεία κειμένου. Το παρακάτω είναι ένα παράδειγμα μιας λίστας επιλογής:

```
<select name="select1" size=40>
<option value="choice1" selected>This is the first choice.
<option value="choice2">This is the second choice.
<option value="choice3">This is the third choice.
</select>
```

Κάθε μια από τις ετικέτες <option> ορίζει μια από τις πιθανές επιλογές. Η ιδιότητα value είναι το όνομα που επιστρέφεται στο πρόγραμμα και το κείμενο έξω από την ετικέτα <option> εμφανίζεται ως κείμενο της επιλογής.

Μια προαιρετική ιδιότητα της ετικέτας, η multiple, μπορεί να καθοριστεί ώστε να επιτρέπει να επιλεχθούν πολλαπλά στοιχεία. Οι browsers εμφανίζουν συνήθως μια μόνο επιλογή <select> ως αναπτυσσόμενη λίστα και μια λίστα πολλαπλών επιλογών ως μια κυλιόμενη λίστα.

Το αντικείμενο στις λίστες επιλογής είναι το αντικείμενο select. Το ίδιο το αντικείμενο έχει τις παρακάτω ιδιότητες:

- Το *name* είναι το όνομα της λίστας επιλογής.
- Το *length* είναι ο αριθμός των επιλογών της λίστας.
- Το *options* είναι ο πίνακας των επιλογών. Κάθε επιλέξιμη επιλογή έχει ένα στοιχείο σ' αυτόν τον πίνακα.
- Το *selectedIndex* επιστρέφει την τιμή δείκτη του τρέχοντος επιλεγμένου στοιχείου. Μπορούμε να την χρησιμοποιήσουμε για να ελέγξουμε εύκολα την τιμή. Σε μια λίστα πολλαπλών επιλογών, υποδεικνύει το πρώτο επιλεγμένο στοιχείο.

Ο πίνακας options έχει μια μόνο ιδιότητα, το length, που υποδεικνύει τον αριθμό των επιλογών. Επί πλέον, κάθε στοιχείο στον πίνακα options έχει τις παρακάτω ιδιότητες:

- Το *index* είναι ο δείκτης στον πίνακα.
- Το *defaultSelected* υποδεικνύει την κατάσταση της ιδιότητας selected.
- Το *selected* είναι η τρέχουσα κατάσταση της επιλογής. Ορίζοντας αυτήν την ιδιότητα σε true επιλέγεται η επιλογή. Ο χρήστης μπορεί να επιλέξει πολλαπλές επιλογές εάν η ιδιότητα multiple συμπεριλαμβάνεται στην ετικέτα <select>.
- Το *name* είναι η τιμή της ιδιότητας και χρησιμοποιείται από τον διακομιστή.
- Το *text* είναι το κείμενο που εμφανίζεται στην επιλογή.

Το αντικείμενο select έχει δύο μεθόδους, τις blur() και focus (), που εκτελεί τους ίδιους σκοπούς με τις αντίστοιχες μεθόδους του αντικειμένου text. Οι τελεστές συμβάντος οι onBlur, onFocus και onChange, είναι επίσης παρόμοιοι με άλλα αντικείμενα.

Η ανάγνωση της τιμής ενός επιλεγμένου στοιχείου είναι μια διαδικασία δύο φάσεων. Πρώτα χρησιμοποιούμε την ιδιότητα *selectedIndex* και έπειτα χρησιμοποιούμε την ιδιότητα value για να βρούμε την τιμή της επιλεγμένης επιλογής. Εδώ βλέπουμε ένα παράδειγμα:

ind = document.navform.choice.selectedindex;

val = document.navform.choice.options[ind].value;

Αυτό χρησιμοποιεί τη μεταβλητή ind για να αποθηκεύσει τον επιλεγμένο δείκτη και έπειτα εκχωρεί τη μεταβλητή val στην τιμή της επιλεγμένης επιλογής. Τα πράγματα γίνονται λίγο πιο περίπλοκα με μια πολλαπλή επιλογή: πρέπει να εξετάσουμε χωριστά την ιδιότητα selected κάθε επιλογής.

11.2 Οι ιδιότητες και οι μέθοδοι του αντικειμένου form

Ο παρακάτω πίνακας 19 περιλαμβάνει τις ιδιότητες και τις μεθόδους των αντικειμένων μιας HTML φόρμας:

Τύπος Στοιχείου	**Όνομα Αντικειμένου**	**Ιδιότητες**	**Μέθοδοι**
κουμπί	button	form, name, type, value	blur(), click(), focus()
πλαίσιο ελέγχου	checkbox	checked, defaultchecked form, name, type, value	blur(), click(), focus()
κρυφό πεδίο	hidden	form, name, type, value	καμία
κουμπί επιλογής	radio	checked, defaultchecked form, name, type, value	blur(), click(), focus()
κουμπί επαναφοράς	reset	form, name, type, value	blur(), click(), focus()

Τύπος Στοιχείου	**Όνομα Αντικειμένου**	**Ιδιότητες**	**Μέθοδοι**
πλαίσιο επιλογής	select	form, name, options, selectedIndex, type	blur(), click(), focus()
κουμπί υποβολής φόρμας	submit	form, name, type, value	blur(), click(), focus()
πλαίσιο κειμένου	text	defaultValue, form, name, type, value	blur(), click(), focus()
περιοχή κειμένου	textarea	defaultValue, form, name, type, value	blur(), click(), focus()

ΠΙΝΑΚΑΣ 19 Στοιχεία μιας φόρμας με τα αντικείμενα και τις μεθόδους τ ους

11.3 Παραδείγματα Πάνω στις Ιδιότητες και Μεθόδους των Φορμών

11.3.1 Εμφάνιση, αποστολή και επικύρωση δεδομένων μιας φόρμας

11.3.1.1 Εμφάνιση Δεδομένων από μια Φόρμα

Ως ένα απλό παράδειγμα χρήσης των φορμών, η Λίστα 11.1 παρουσιάζει μια φόρμα με όνομα, διεύθυνση και τηλέφωνα, καθώς επίσης και μια JavaScript συνάρτηση που εμφανίζει τα δεδομένα από τη φόρμα σε ένα αναδυόμενο παράθυρο.

Λίστα 11.1 Μια φόρμα που εμφανίζει δεδομένα σε ένα αναδυόμενο παράθυρο

```
<html>
<head>
<title>Form Example</title>
<script language="JavaScript" type="text/javascript">
```

```
function display() {
  DispWin = window.open('','NewWin', 'toolbar=no,status=no,width=300,height=200')
message="<ul><li><b>name: </b>" + document.form1.yourname.value;
message+="<li><b>address:</b>"+document.forml.address.value; message +=
"<lixb>phone: </b>" + document.forml.phone.value + "</ul>";
DispWin.document.write(message);
}
</script>
</head>
<body>
<h1>Form Example</h1>
Enter the following information. When you press the Display button, the data you entered
will be displayed in a pop-up window.
<form name="forml">
   <p><b>Name:</b> <input type="text" size="20" name="yourname"> </p>
   <p><b>Address:</b> <input type="text" size="30" name="address"> </p>
   <p><b>Phone: </b> <input type="text" size="15" name="phone"> </p>
   <p><input type="button" value="Display" onClick="display();"></p>
</form>
</body>
</html>
```

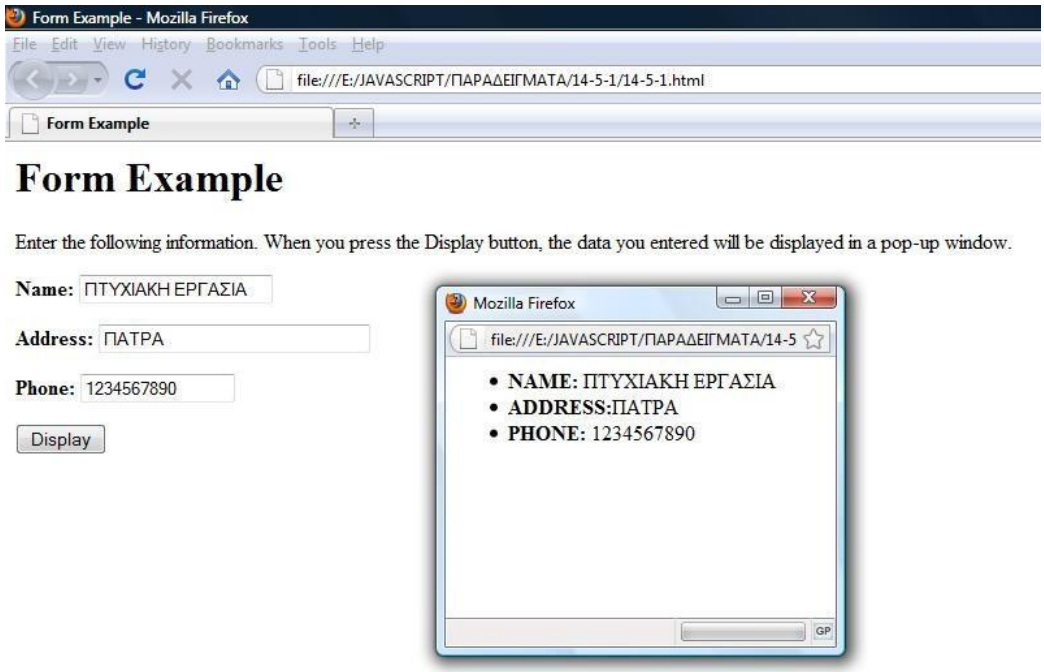
Ακολουθεί μια ανάλυση για το πώς δουλεύει αυτό το HTML έγγραφο και το script:

- Η ενότητα <script> στην επικεφαλίδα του εγγράφου ορίζει μια συνάρτηση που ονομάζεται display () και η οποία ανοίγει ένα νέο παράθυρο και εμφανίζει τις πληροφορίες από τη φόρμα.

- Η ετικέτα <form> αρχίζει τη φόρμα. Επειδή αυτή την φόρμα την χειρίζεται εξ ολοκλήρου η JavaScript, δεν απαιτείται καμία ενέργεια ή μέθοδος.

- Οι ετικέτες <input> ορίζουν τα τρία πεδία της φόρμας: yourname, address και phone. Η τελευταία ετικέτα ορίζει το κουμπί Display, το οποίο ορίζεται να τρέχει την συνάρτηση display ().

Η Εικόνα 11 δείχνει αυτήν την φόρμα σε δράση. Το κουμπί Display έχει πατηθεί και το αναδυόμενο παράθυρο δείχνει τα αποτελέσματα.



ΕΙΚΟΝΑ 11 Εμφάνιση δεδομένων από μια φόρμα

11.3.1.2 Αποστολή των Αποτελεσμάτων μιας Φόρμας με Ηλεκτρονικό Ταχυδρομείο

Ένας εύκολος τρόπος για να χρησιμοποιήσουμε μια φόρμα είναι να σταλούν τα αποτελέσματα με ηλεκτρονικό ταχυδρομείο. Μπορούμε να το κάνουμε αυτό

χωρίς να χρησιμοποιήσουμε κώδικα JavaScript, αν και θα μπορούσαμε να χρησιμοποιήσουμε JavaScript για να επικυρώσουμε τις πληροφορίες που εισάγονται.

Για να στείλουμε τα αποτελέσματα μιας φόρμας με ηλεκτρονικό ταχυδρομείο, χρησιμοποιούμε την ενέργεια mailto: στην ιδιότητα action της φόρμας. Η Λίστα 11.2 είναι μια τροποποιημένη έκδοση της φόρμας με τα ονόματα και τις διευθύνσεις της Λίστας 11.2, η οποία στέλνει τα αποτελέσματα με ηλεκτρονικό ταχυδρομείο.

Λίστα 11.2 Στέλνοντας τα αποτελέσματα μιας φόρμας με ηλεκτρονικό ταχυδρομείο

```
<html>
<head>
<title>Email Form Example</title>
</head>
<body>
<h1>Email Form Example</h1>
Enter the following information. When you press the Submit button, the data you entered will be sent by email.
<form name="form1"  action="mailto:user@host.com"
enctype="text/plain" method="post">
<p><b>Name:</b> <input type="text" size="20" name="yourname"> </p>
<p><b>Address:</b> <input type="text" size="30" name="address"> </p>
<p><b>Phone: </b> <input type="text" size="15" name="phone">
</p>
```

<p><input type="submit" value="Submit"></ρ>

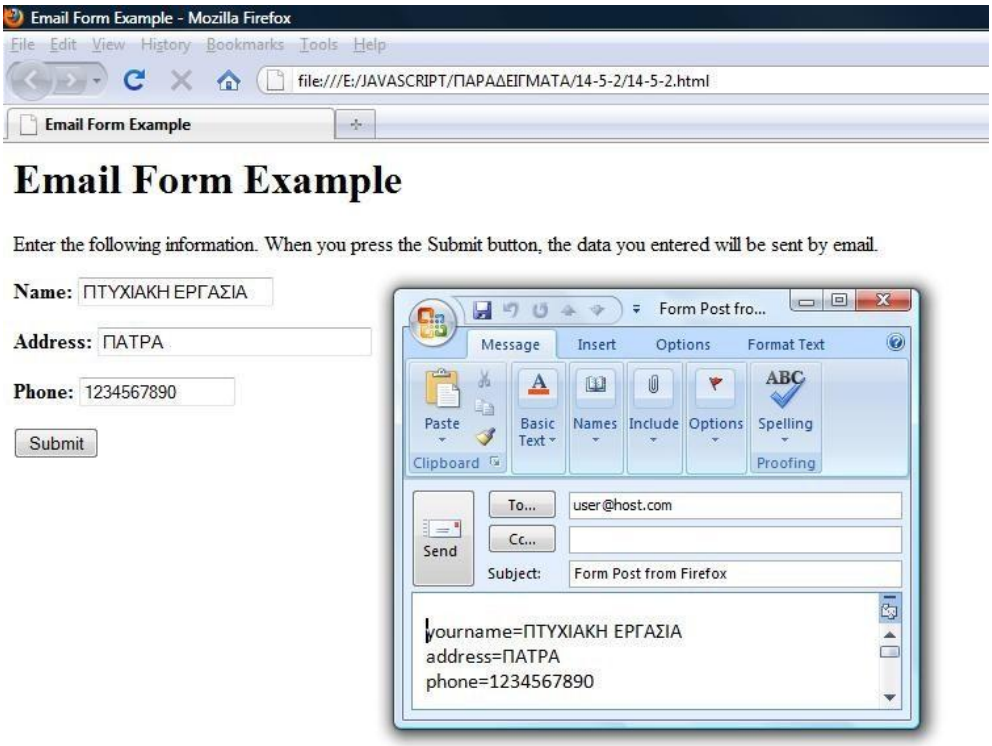
</form>

</body>

</html>

Για να χρησιμοποιήσουμε αυτήν την φόρμα πρέπει να αλλάξουμε το user@host.com στην ιδιότητα action της ετικέτας <form> στην δική μας διεύθυνση ηλεκτρονικού ταχυδρομείου. Η ιδιότητα enctype=text/plain της ετικέτας <form> εξασφαλίζει ότι οι πληροφορίες στο μήνυμα ηλεκτρονικού ταχυδρομείου θα είναι σε μια αναγνώσιμη απλή μορφή κειμένου αντί να είναι κωδικοποιημένες.

Αν και αυτό παρέχει έναν γρήγορο τρόπο να ανακαλέσουμε δεδομένα από μια φόρμα, το μειονέκτημα αυτής της τεχνικής είναι ότι εξαρτάται κατά πολύ από τον browser. Το αν θα δουλέψει σε κάθε χρήστη της σελίδας μας εξαρτάται από τη διαμόρφωση του browser του χρήστη και από τον πελάτη ηλεκτρονικού ταχυδρομείου. Για έναν πιο αξιόπιστο τρόπο να στέλνουμε τα αποτελέσματα μιας φόρμας, μπορούμε να χρησιμοποιήσουμε μια CGI έξοδο φόρμας σε ηλεκτρονικό ταχυδρομείο. Είναι διαθέσιμα διάφορα δωρεάν CGI script και υπηρεσίες στο internet.



ΕΙΚΟΝΑ 12 Αποστολή αποτελεσμάτων μιας φόρμας με το ηλεκτρονικό ταχυδρομείο

11.3.1.3 Επικύρωση μιας Φόρμας

Ένας από τους χρήσιμους σκοπούς της JavaScript είναι η επικύρωση φορμών. Αυτό σημαίνει ότι χρησιμοποιούμε ένα script για να επαληθεύσουμε ότι οι πληροφορίες που εισάγονται είναι έγκυρες, για παράδειγμα, ότι κανένα πεδίο δεν είναι κενό και ότι τα δεδομένα είναι στην σωστή μορφή.

Μπορούμε να χρησιμοποιήσουμε την JavaScript για να επικυρώσουμε μια φόρμα είτε στέλνεται με ηλεκτρονικό ταχυδρομείο είτε σε ένα CGI script, είτε χρησιμοποιείται απλώς από ένα script. Η Λίστα 11.3 είναι μια έκδοση της φόρμας με τα ονόματα και διευθύνσεις που περιλαμβάνει και επικύρωση.

Λίστα 11.3 Μια φόρμα με ένα Script επικύρωσης

```
<html>

<head>

<title>Form Example</title>

<script language="JavaScript" type="text/javascript">
```

```
function validate() {
    if (document.form1.yourname.value.length < 1) {
        alert("Please enter your full name.");
        return false;
    }
    if (document.form1.address.value.length < 3) {
        alert("Please enter your address.");
        return false;
    }
    if (document.form1.phone.value.length < 3) {
        alert("Please enter your phone number.");
        return false;
    }
return true;
}
</script>
</head>
<body>
<h1>Form Example</h1>
<p>Enter the following information. When you press the Submit button, the data you entered will be validated, then sent by email.</p>
<form name="form1" action="mailto:user@host.com" enctype="text/plain"
method="post" onSubmit="return validate();">
```

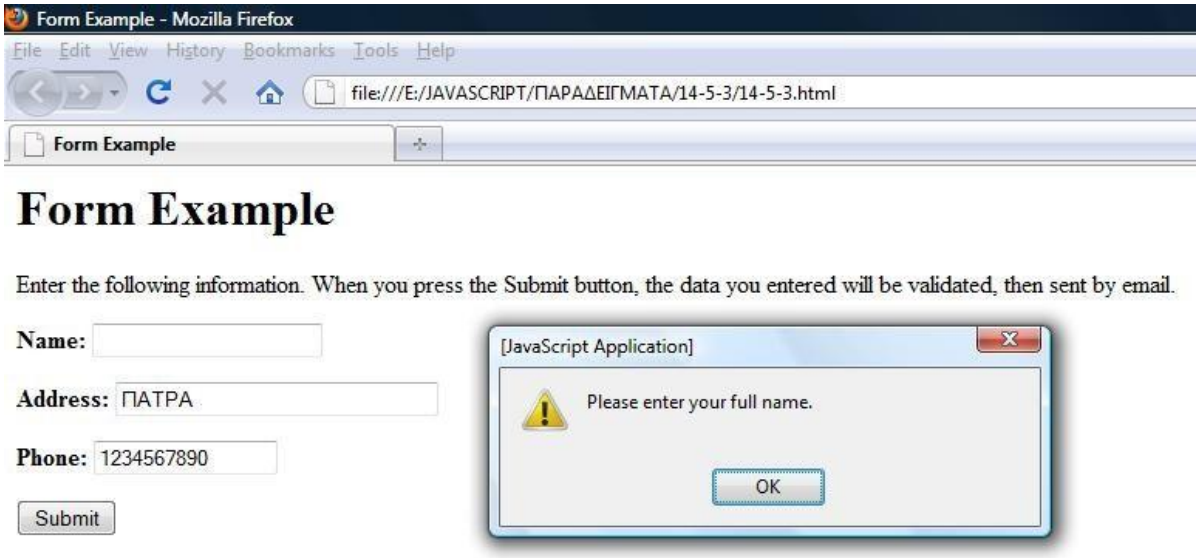
```
<p><b>Name:</b> <input type="text" size="20" name="yourname"> </p>
<p><b>Address:</b> <input type="text" size="30" name="address"> </p>
<p><b>Phone: </b> <input type="text" size="15" name="phone"> </p>
<p><input type="submit" value="Submit"x/p>
</form>
</body>
</html>
```

Αυτή η φόρμα χρησιμοποιεί μια συνάρτηση που ονομάζεται validate () για να ελέγξει τα δεδομένα σε κάθε ένα από τα πεδία της φόρμας. Κάθε πρόταση if αυτής της συνάρτησης ελέγχει το μήκος ενός πεδίου. Εάν το πεδίο είναι αρκετά μεγάλο για να είναι έγκυρο, η φόρμα μπορεί να σταλεί, διαφορετικά, η αποστολή σταματά και εμφανίζεται ένα μήνυμα προειδοποίησης.

Αυτή η φόρμα έχει διαμορφωθεί ώστε να στέλνει τα αποτελέσματα της με ηλεκτρονικό ταχυδρομείο, όπως στη Λίστα 11.3. Η ετικέτα <form> χρησιμοποιεί έναν χειριστή συμβάντος onSubmit για να καλεί την συνάρτηση validate (). Η λέξη-κλειδί return εξασφαλίζει ότι η τιμή που επιστρέφεται από την validate () θα προσδιορίσει το εάν θα υποβληθεί η φόρμα.

Μπορούμε επίσης να χρησιμοποιήσουμε τον χειριστή συμβάντος onChange σε κάθε πεδίο της φόρμας για να καλέσουμε μια ρουτίνα επικύρωσης. Αυτό επιτρέπει στο πεδίο να επικυρώνεται πριν πατηθεί το κουμπί Submit.

Η Εικόνα 13 δείχνει αυτό το script σε δράση, όπως εμφανίζεται από το Firefox. Η φόρμα είναι συμπληρωμένη εκτός από το όνομα και το παράθυρο διαλόγου υποδεικνύει ότι πρέπει να εισαχθεί και το όνομα.



ΕΙΚΟΝΑ 13 Επικύρωση των δεδομένων μιας φόρμας

11.3.2 Δημιουργία μιας δυναμικής φόρμας

Χρησιμοποιώντας το W3C DOM, το οποίο επιτρέπει στα script μας να χειρίζονται ιστοσελίδες, παράθυρα και έγγραφα, μπορούμε να προχωρήσουμε ένα βήμα ακόμα, δημιουργώντας ένα script που να μπορεί να προσθέτει στοιχεία σε μια φόρμα ή να εμφανίζει ή να κρύβει ενότητες μιας φόρμας.

Η Λίστα 11.4 δείχνει το HTML έγγραφο γι' αυτό το παράδειγμα, το οποίο ορίζει μια φόρμα παραγγελίας. Η φόρμα θα έχει δύο δυναμικές λειτουργίες: κατ' αρχάς, τα πεδία διευθύνσεων Ship To (αποστολή σε) δεν φαίνονται εκτός και αν απαιτούνται και δεύτερον, ένα κουμπί μας επιτρέπει να προσθέσουμε και άλλα στοιχεία στη φόρμα.

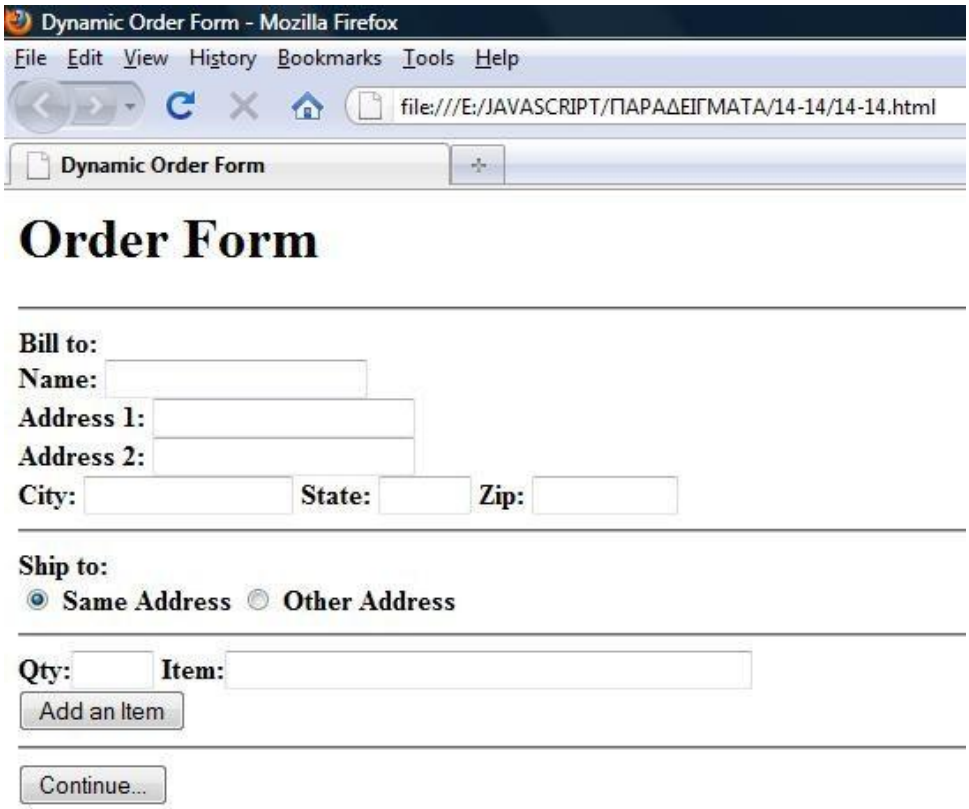
Λίστα 11.4 Το HTML Έγγραφο για το Παράδειγμα της Δυναμικής Φόρμας

```
<html>

<head>

<title>Dynamic Order Form</title>

<script language="JavaScript" type="text/javascript"
```

```
src="dform.js"> </script>

</head>

<body>

<h1>Order Forra</h1> <hr>

<form name="form1">

<b>Bill to:</b><br>

<b>Name:</b>

<input type="text" name="customer" size="20"><br>

<b>Address 1:</b> <input type="text" name="addr1" size="20"><br>

<b>Address 2:</b> <input type="text" name="addr2" size="20"><br>

<b>City:</b> <input type="text" name="city" size="15">

<b>State:</b> <input type="text" name="state" size="4">

<b>Zip:</b> <input type="text" name="zip" size="9"> <hr>

<b>Ship to:</b><br> <input type="radio" name="shipopt" value="same" checked
onClick="Show(0);">

<b>Same Address</b> <input type="radio" name="shipopt" value="other"
onClick="Show(1);">

<b>Other Address</b>

<div ID="shipto" style="display: none;"> <br>

<b>Name:</b> <input type="text" name="shipname" size="20"><br>

<b>Address 1:</b> <input type="text" name="shipaddrl" size="20"><br>

<b>Address 2:</b> <input type="text" name="shipaddr2" size="20"><br>

<b>City:</b> <input type="text" name="shipcity" size="15">
```

```
<b>State:</b><input type="text" name="shipstate" size="4">
<b>Zip:</b> <input type="text" name="shipzip" size="9"> </div> <hr>
<div ID="items">
<b>Qty:</b><input type="text" name="qty1" size="3">
<b>Item:</b><input type="text" name="item1" size="45">
<br>
<input type="button" value="Add an Item" onClick="AddItem();" ID="add">
</div>
<hr>
<input type="submit" value="Continue...">
</form>
</body>
</html>
```

Αποθηκεύουμε αυτό το HTML έγγραφο σε έναν φάκελο. Εάν το φορτώσουμε σε έναν browser, θα δούμε την προκαθορισμένη εμφάνιση της φόρμας, αλλά οι δυναμικές λειτουργίες δεν θα δουλεύουν ακόμα. Η Εικόνα 14 παρουσιάζει την προκαθορισμένη εμφάνιση της φόρμας.



ΕΙΚΟΝΑ 14 Η δυναμική φόρμα πριν προστεθεί το script.

Το script γι' αυτό το παράδειγμα θα περιλαμβάνει δύο συναρτήσεις: την AddItem(), για προσθήκη στοιχείων στη φόρμα και την Show(), για την εμφάνιση ή την απόκρυψη της διεύθυνσης αποστολής. Η Λίστα 11.5 δείχνει το αρχείο script.

Λίστα 11.5 Το JavaScript Αρχείο για το Παράδειγμα της Δυναμικής Φόρμας

```
// καθολική μεταβλητή variable

var items=l;

function AddItem() {

if (!document.getElementById) return;

// Προσθήκη ενός στοιχείου στην φόρμα

div=document.getElementById("items");

button=document.getElementById("add");
```

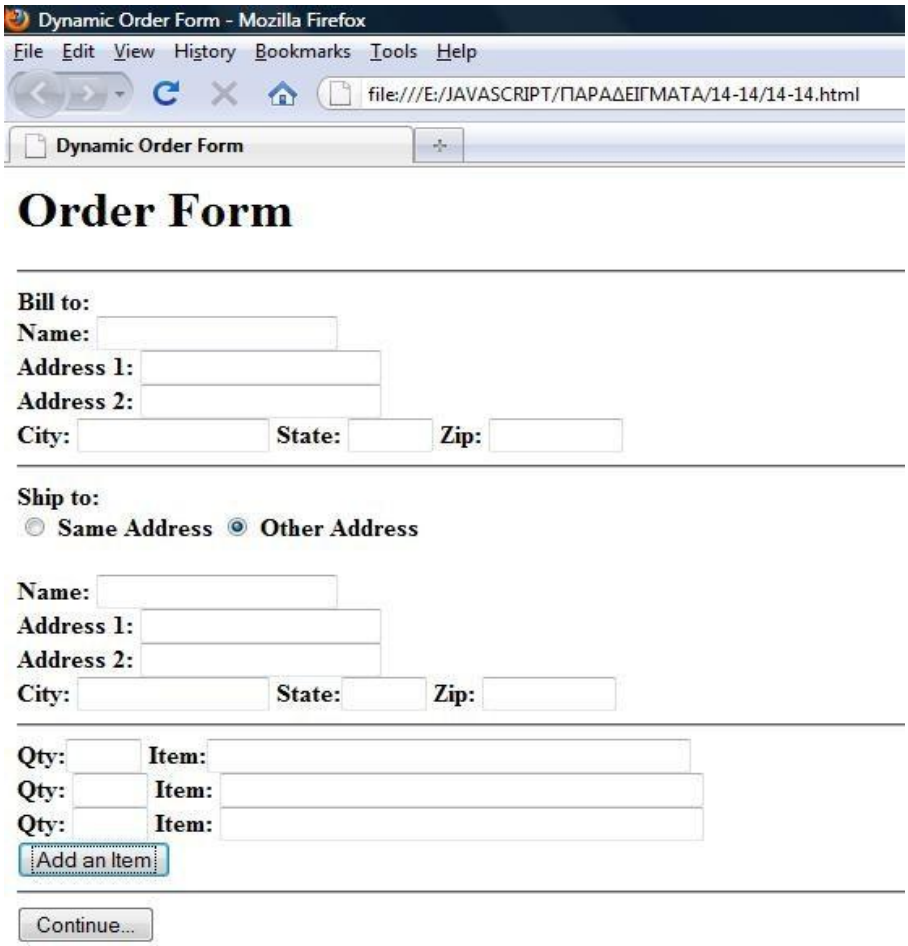
```
items++;
newitem="<b>Qty: </b>";
newitem+="<input type=\"text\" name=\"qty" + items;
newitem+="\" size=\"3\"> ";
newitem+="<b>Item: </b>";
newitem+="<input type=\"text\" name=\"item" + items;
newitem+="\" size=\"45\"><br>";
newnode=document.createElement("span");
newnode.innerHTML=newitem;
div.insertBefore(newnode,button);
}
function Show(a) {
if (!document.getElementById) return;
// Απόκρυψη ή εμφάνιση της διεύθυνσης αποστολής
obj=document.getElementById("shipto");
if (a) obj.style.display="block";
else obj.style.display="none";
}
```

Εδώ υπάρχει μια ανάλυση του τρόπου που δουλεύει αυτό το script:

Η πρώτη γραμμή ορίζει μια καθολική μεταβλητή, την items, η οποία παρακολουθεί τον αριθμό των στοιχείων. Αυτός χρησιμοποιείται για να δοθεί μια μοναδική ιδιότητα name σε κάθε ετικέτα <input>, καθώς αυτές προστίθενται.

Η συνάρτηση AddItem() προσθέτει επιπλέον πεδία Quantity (ποσότητα) και Item (στοιχείο) στη φόρμα χρησιμοποιώντας τη DOM μέθοδο insertBefore ( ). Η συνάρτηση Show() χρησιμοποιεί την ιδιότητα style.display είτε για να εμφανίσει είτε για να κρύψει την ενότητα με την id τιμή shipto.

Για να ελέγξουμε το script, το αποθηκεύουμε ως dform.js και φορτώστε το HTML έγγραφο σε έναν browser. Η Εικόνα 15 δείχνει το έγγραφο με δύο πρόσθετα πεδία και την διεύθυνση αποστολής. [3]



ΕΙΚΟΝΑ 15 Η δυναμική φόρμα σε δράση.

ΚΕΦΑΛΑΙΟ 12: Η JAVASCRIPT ΚΑΙ ΤΑ ΠΛΑΙΣΙΑ

12.1 Τι Είναι Τα Πλαίσια

Τα πλαίσια διαιρούν ένα παράθυρο σε δυο ή περισσότερες περιοχές (ένα σύνολο πλαισίων - frameset), κάθε μία εκ των οποίων περιέχει διαφορετικά περιεχόμενα. Αυτό διαφέρει από τους πίνακες γιατί τα τμήματα ενός συνόλου πλαισίων περιέχουν ένα ξεχωριστό έγγραφο Hypertext Markup Language (HTML) και μπορεί να γίνει αλλαγή σε ένα από τα τμήματα χωρίς να επηρεαστούν τα υπόλοιπα τμήματα.

Το HTML έγγραφο εισάγει τα πλαίσια χρησιμοποιώντας ένα σύνολο ετικετών <frameset> και </frameset> για να δημιουργήσει ένα σύνολο πλαισίων.

Τα πλαίσια έχουν διάφορες εφαρμογές. Για παράδειγμα, μπορεί να χρησιμοποιηθούν ώστε να δημιουργήσουν ένα σύστημα πλοήγησης για μια τοποθεσία, ή να δημιουργήσουν ένα σύστημα αναφοράς, όπου ο πίνακας περιεχομένων είναι σε ένα πλαίσιο και τα αντίστοιχα περιεχόμενα είναι σε ένα άλλο πλαίσιο.

Ο παρακάτω κώδικας περιλαμβάνει το στοιχείο frameset. Βάζει μαζί τον αριθμό και τη δομή των πλαισίων που εμφανίζονται όταν ανοίγει το έγγραφο.

```
<html>

<frameset cols= “20%,80%”>

<frame src= “frame1.html”>

<frame src= “frame2.html”>

</frameset>

<noframes>

Sorry, your browser does not support frames. Use the link
```

*below to go to the frameless version of the site.
*

<A href="noframes . html">Frameless Site</A>

</noframes>

</html>

Αυτός ο κώδικας δημιουργεί ένα απλό σύνολο πλαισίων, το οποίο παράγει ένα μικρότερο πλαίσιο στην αριστερή πλευρά του παραθύρου και ένα μεγαλύτερο πλαίσιο στα δεξιά.

Το έγγραφο που δημιουργήθηκε από τον προηγούμενο κώδικα είναι η σελίδα που θέλουμε να ανοίξουμε στον browser και θα εμφανίζει τα πλαίσια. Αποθηκεύουμε αυτό το έγγραφο ως frameset.html, ώστε να μπορούμε να το χρησιμοποιήσουμε μόλις προσθέσουμε κάποιον κώδικα στα έγγραφα που χρησιμοποιούνται στα πλαίσια.

Μόλις δημιουργηθεί το σύνολο πλαισίων, πρέπει να δημιουργήσουμε τα έγγραφα που θα μπουν στα πλαίσια. Κατ' αρχάς, ας δημιουργήσουμε το έγγραφο που θα χρησιμοποιηθεί στο αριστερά πλαίσιο (frame1.html):

<html>

<body>

I am frame1.html, and I am on the left side!

</body>

</html>

Αυτός ο απλός κώδικας μας "λέει" ποιο έγγραφο θα εμφανίζεται στο κάθε πλαίσιο. Αποθηκεύουμε το αρχείο ως frame1.html στον ίδιο κατάλογο με το frameset.html.

Μετά, πρέπει να δώσουμε τον κώδικα για το άλλο πλαίσιο (frame2.html):

<html>

<body>

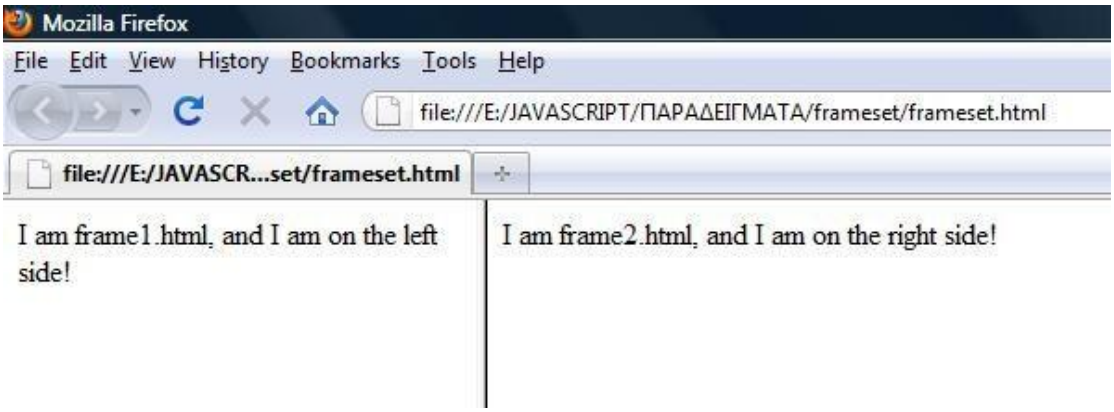
I am frame2.html, and I am on the right side!

</body>

</html>

Το αποθηκεύουμε ως frame2.html στον ίδιο κατάλογο με το frameset.html και το frame1.html.

Τώρα, όταν θα ανοίγουμε το frameset.html στον browser που θα πρέπει να εμφανίζεται αυτό το παράθυρο με δύο πλαίσια, περιέχει το κατάλληλο HTML έγγραφο, όπως φαίνεται στην Εικόνα 16.



ΕΙΚΟΝΑ 16 Ένα σύνολο πλαισίων με δυο πλαίσια, ένα σε κάθε στήλη που δημιουργείται από το σύνολο πλαισίων

Η ετικέτα αρχής FRAMESET μπορεί να πάρει την ιδιότητα cols για να διαιρέσει το παράθυρο σε στήλες. Με τον ίδιο τρόπο, μπορεί να χρησιμοποιήσει την ιδιότητα rows για να διαιρέσει τη σελίδα σε γραμμές από πάνω προς τα κάτω. Εδώ βλέπουμε ένα παράδειγμα κώδικα που δημιουργεί ένα σύνολο πλαισίων με δυο γραμμές:

<html>

<frameset rows= "20%,80%">

<frame src= “frame1.html”>

<frame src= “frame2.html”>

</frameset>

<noframes>

*Use the link below to go to the frameless version of the site.
*

<a href = “noframes.html”> Frameless Site </a

</noframes>

</html>

Δίνεται η δυνατότητα οι αριθμοί που χρησιμοποιούνται στην ιδιότητα rows ή cols να είναι σε πίξελ αντί για ποσοστά. Επίσης, μπορεί να χρησιμοποιηθεί ένας αστερίσκος (*) εάν υπάρχει προτίμηση ένα ορισμένο πλαίσιο να καταλάβει όλο το κενό χώρο που παραμένει αφού ορισθούν όλα τα άλλα πλαίσια.

12.2 Πρόσβαση στα Πλαίσια

Για να προσπελαστεί ένα πλαίσιο με την JavaScript, όπως και στις φόρμες, μπορούμε είτε να χρησιμοποιήσουμε έναν πίνακα frames είτε να ονομαστεί το πλαίσιο και να χρησιμοποιηθεί το όνομα του πλαισίου.

12.2.1 Πρόσβαση στα Πλαίσια με την Χρήση Πίνακα Frames

Χρησιμοποιείται ο πίνακας frames για πρόσβαση σε πλαίσια με βάση τη σειρά τους στον κώδικα προέλευσης. Θα πρέπει να προσπελάσουμε ένα πλαίσιο μέσα από ένα άλλο πλαίσιο, έτσι θα πρέπει να μπορούμε να βρούμε το πλαίσιο που θέλουμε να προσπελάσουμε.

Υπενθυμίζουμε ότι ο πίνακας frames προέρχεται από το αντικείμενο window. Τα πλαίσια έχουν τις περισσότερες από τις ιδιότητες και μεθόδους των κανονικών παραθύρων, αλλά προσπελάζονται διαφορετικά. Αν πάρουμε ως

παράδειγμα τον κώδικα που δημιουργεί ένα σύνολο πλαισίων με δύο πλαίσια που αναφέρεται παραπάνω.

Εάν γράφαμε τον κώδικα για κάποιο script μέσα στο πρώτο πλαίσιο (frame1.html) και θέλαμε να ξέραμε την τιμή της ιδιότητας location του δευτέρου πλαισίου (frame2.html) για να την εμφανίσει στον επισκέπτη, θα έπρεπε να βρούμε πως μπορούμε να προσπελάσουμε το δεύτερο πλαίσιο. Κατά συνέπεια, για να προσπελάσουμε το άλλο πλαίσιο, πρέπει να βρούμε έναν τρόπο να επιστρέψουμε στο κύριο παράθυρο και να αναφέρουμε το πλαίσιο. Η προσπέλαση γίνεται με την βοήθεια του πίνακα frames, επειδή μπορεί να προσπελάσει αυτό το κύριο παράθυρο, το οποίο περιέχει τον κώδικα για το σύνολο πλαισίων. Ο πίνακας frames περιέχει ένα στοιχείο εισόδου για κάθε ετικέτα frame στον κώδικα. Η αρίθμηση αρχίζει από το 0 και συνεχίζεται με τη σειρά που εμφανίζεται κάθε ετικέτα frame στον κώδικα προέλευσης. Κατά συνέπεια, για να προσπελάσουμε το πρώτο πλαίσιο σε ένα σύνολο πλαισίων, θα μπορούσαμε να χρησιμοποιήσουμε την παρακάτω σύνταξη:

top.frames[0]

Χρησιμοποιώντας την ιδιότητα top μπορεί να προσπελαστεί το κύριο παράθυρο και τον κώδικα του συνόλου πλαισίων. Κατόπιν, χρησιμοποιείται το frames[0] για να προσπελαστεί το πρώτο πλαίσιο στον κώδικα προέλευσης. Έτσι, εάν γράφαμε τον κώδικα μέσα στο πρώτο πλαίσιο και θέλαμε να προσπελάσουμε το δεύτερο πλαίσιο, μπορούμε να χρησιμοποιήσουμε αυτήν την σύνταξη:

top.frames[1]

Τώρα, μπορούμε να κάνουμε τον κώδικα στο πρώτο πλαίσιο, να προσπελάσει τις απαραίτητες πληροφορίες στο δεύτερο πλαίσιο προς χάρη του επισκέπτη. Ο παρακάτω κώδικας είναι για το πρώτο πλαίσιο (frame1.html) και

κάνει το πρώτο πλαίσιο να εμφανίσει την τιμή της ιδιότητας location του δεύτερου (frame2.html):

```
<html>

<body>

The second frame is from: <br>

<script language="JavaScript">

document.write(top.frames[1].location);

</script>

</body>

</html>
```

12.2.2 Χρήση Ονομάτων για την Πρόσβαση στα Πλαίσια

Ένας άλλος τρόπος να προσπελαστεί ένα πλαίσιο από ένα άλλο είναι να χρησιμοποιηθεί το όνομα του πλαισίου. Για παράδειγμα αυτός ο κώδικας δίνει ένα όνομα σε κάθε πλαίσιο προσθέτοντας την ιδιότητα name στην ετικέτα frame (και πάλι ονομάζουμε το αρχείο frameset.html):

```
<html>

<frameset cols="50%,50%">

<frame src="framel.html" name="left_side">

<frame src="frame2.html" name="right_side">

</frameset>

<noframes>

Use the link below to go to the frameless version of the of the site.<br>

<a href="noframes.html">frameless site</a>

</noframes>
```

</html>

Μπορούμε τώρα να προσπελάσουμε ένα από τα πλαίσια από το άλλο πλαίσιο χρησιμοποιώντας το όνομα του πλαισίου αντί για τον πίνακα frames. Κατά συνέπεια, εάν θέλουμε να προσπελάσουμε το δεύτερο πλαίσιο (right_side) από το πρώτο, θα μπορούσαμε να χρησιμοποιήσουμε αυτήν την σύνταξη:

top.right_side

Με τον ίδιο τρόπο, θα μπορούσαμε να προσπελάσουμε το πρώτο πλαίσιο μέσα από το δεύτερο πλαίσιο με αυτήν την σύνταξη:

top.left_side

12.3 Αλλαγή Πλαισίων

Για να αλλάξουμε τα περιεχόμενα ενός πλαισίου από ένα άλλο πλαίσιο, πρέπει να χρησιμοποιήσουμε την HTML ιδιότητα target για μία μόνο αλλαγή. Ωστόσο χρησιμοποιώντας την ιδιότητα location στην JavaScript, μπορούν να χρησιμοποιηθούν περισσότερα από ένα πλαίσιο τη φορά.

Στην HTML, εάν θέλουμε να αλλάξουμε ένα πλαίσιο μέσα από ένα άλλο μπορούμε να χρησιμοποιήσουμε την ιδιότητα target μέσα στην ετικέτα και να της δώσουμε την τιμή του ονόματος του πλαισίου. Για παράδειγμα εάν θέλουμε να αλλάξουμε τα περιεχόμενα ενός πλαισίου, που ονομάζεται από ένα άλλο πλαίσιο στο σύνολο πλαισίων, θα μπορούσαμε να δημιουργήσουμε μια σύνδεση ως εξής:

Next Page

Αυτός ο κώδικας θα άνοιγε το nextpage.html στο πλαίσιο right_side, αντί να το ανοίξει στο τρέχον πλαίσιο (που είναι η προεπιλογή για μια σύνδεση).

Για να εκτελέσουμε την ίδια εργασία στην JavaScript, μπορούμε να χρησιμοποιήσουμε το όνομα του πλαισίου μαζί με την ιδιότητα location, όπως στον παρακάτω κώδικα:

top.right_side.location="nextpage.html";

Εδώ, το rightside είναι το όνομα του πλαισίου στο οποίο θα αλλάξουν τα περιεχόμενα και το nextpage.html είναι η σελίδα που εμφανίζεται στην θέση του αρχικού εγγράφου. Για να μετατρέψουμε αυτό σε μια σύνδεση, θα μπορούσαμε να χρησιμοποιήσουμε την τεχνική με την πρόταση return false:

<α href="n" onClick= "top.right_side.location='newpage.html';return false;”>

New Page</α>

Από την άλλη, θα μπορούσαμε επίσης να χρησιμοποιήσουμε μια μέθοδο η οποία να μας επιτρέπει να δείχνουμε στην ιδιότητα href μιας ετικέτας σύνδεσης για να εκτελέσουμε JavaScript προτάσεις, όπως φαίνεται εδώ:

**

New Page

Παρατηρούμε την προσθήκη της λέξης "javascript" σε πεζά, ακολουθούμενη από μια άνω και κάτω τελεία. Μετά από αυτό, μπορούμε να χρησιμοποιήσουμε JavaScript προτάσεις όπως θα τις χρησιμοποιούσαμε σε έναν χειριστή συμβάντος.

Η τεχνική στον προηγούμενο κώδικα δεν απαιτεί να προστεθεί μια πρόταση return false στον κώδικα.

12.4 Πλοήγηση με Πλαίσια

Η JavaScript παρέχει, επίσης, βολικές τεχνικές πλοήγησης με πλαίσια. Για παράδειγμα, θα μπορούσαμε να αλλάξουμε την πλοήγηση με το πλαίσιο επιλογής για να δουλεύει από το ένα πλαίσιο σε ένα άλλο ή θα μπορούσε να

δημιουργηθεί κώδικας ώστε να επιτρέπεται στους επισκέπτες μας να καταργούν τα πλαίσια. Θα μπορούσε, επίσης, να δημιουργηθεί κώδικας, που να στέλνει τους επισκέπτες στα πλαίσια εάν μπουν κατά λάθος σε μια σελίδα που ανήκει σε ένα σύνολο πλαισίων.

Εάν θέλουμε το πλαίσιο επιλογής σε ένα πλαίσιο να ανοίγει ένα έγγραφο σε ένα άλλο πλαίσιο, πρέπει να αλλάξουμε λίγο την πλοήγηση στο πλαίσιο επιλογής. Για να το κάνουμε αυτό, πρέπει να αλλάξουμε μόνο ένα μέρος του κώδικα στο πλαίσιο επιλογής. Χρησιμοποιούμε το όνομα αρχείου frameset2.html για να δούμε πώς θα εμφανίζονται τα πλαίσια. Ο κώδικας φαίνεται εδώ:

```
<html>
<frameset rows="120,*">
<frame src="frame1.html" name="t_frame">
<frame src="frame2.html" name="b_frame">
</frameset>
<noframes>
Use the link below to go to the frameless version of the site.<br>
<a href="noframes.html">Frameless Site</a>
</nofames>
</html>
```

Σε αυτήν την περίπτωση, τα πλαίσια πηγαίνουν από πάνω προς τα κάτω σε γραμμές. Το κορυφαίο πλαίσιο (t_frame) εκτείνεται κατά 120 πίξελ από την κορυφή και το κάτω πλαίσιο (b_frame) καταλαμβάνει τον υπόλοιπο χώρο της σελίδας. Στο προηγούμενο παράδειγμα, αλλάζει ένα πλαίσιο επιλογής όταν ο επισκέπτης κάνει μια επιλογή (χωρίς κουμπί).

Επιπλέον μπορεί να τοποθετηθεί μια σύνδεση στην σελίδα, στην οποία να κάνουν κλικ οι επισκέπτες για να καταργήσουν τα πλαίσια. Πρέπει να προστεθεί ένας ειδικός προορισμός στην ετικέτα <a> όπως φαίνεται εδώ:

* Break Out of Frames *

Ο προορισμός _top «λέει» στον browser να χρησιμοποιήσει το πλήρες παράθυρο όταν θα ανοίξει το url της σύνδεσης, αντί να ανοίξει την σύνδεση μέσα σε ένα πλαίσιο. Πρέπει απλώς να αντικαταστήσουμε το url στην ετικέτα.

Τέλος δίνεται η δυνατότητα της κατάργησης αυτόματα στα πλαίσια, αλλά πρέπει να ανακαλυφθεί εάν τα πλαίσια είναι μέσα σε ένα σύνολο πλαισίων όταν φορτώνεται η σελίδα. Εάν η σελίδα δεν χρησιμοποιεί πλαίσια, τότε μπορεί να προσδιοριστεί αρκετά εύκολα εάν υπάρχουν πλαίσια από μία άλλη τοποθεσία χρησιμοποιώντας την ιδιότητα length.

12.5 Χρησιμοποιώντας πλαίσια με την JavaScript

Οι browser υποστηρίζουν επίσης τα πλαίσια (frames), τα οποία μας επιτρέπουν να διαιρέσουμε το παράθυρο του browser σε πολλαπλά τμήματα. Κάθε τμήμα μπορεί να περιέχει ένα ξεχωριστό url ή την έξοδο ενός script.

Όταν ένα παράθυρο περιέχει πολλαπλά πλαίσια, κάθε πλαίσιο αντιπροσωπεύεται στην JavaScript από ένα αντικείμενο frame. Αυτό το αντικείμενο είναι ισοδύναμο με ένα αντικείμενο window, αλλά χρησιμοποιείται συγκεκριμένα για να χειριζόμαστε αυτό το πλαίσιο. Το όνομα του αντικειμένου frame είναι το ίδιο με την ιδιότητα name που δίνουμε στην ετικέτα <frame>.

Οι λέξεις-κλειδιά window και self αναφέρονται στον τρέχον πλαίσιο. Μια άλλη λέξη-κλειδί, η parent, μας επιτρέπει να αναφερόμαστε στο κύριο παράθυρο.

Κάθε αντικείμενο πλαισίου σε ένα παράθυρο είναι παιδί του αντικειμένου παραθύρου parent. Ας υποθέσουμε ότι ορίζουμε ένα σύνολο πλαισίων χρησιμοποιώντας την παρακάτω HTML:

```
<frameset rows="*,*" cols="*,*">
<frame name="topleft" src="topleft.htm">
<frame name="topright" src="topright.htm">
<frame name="bottomleft" src="botleft.htm">
<frame name="bottomright" src="botright.htm">
</frameset>
```

Αυτός ο κώδικας απλώς διαιρεί το παράθυρο σε τέταρτα. Εάν έχουμε ένα πρόγραμμα JavaScript στο αρχείο topleft.htm, θα αναφερόμασταν στα άλλα παράθυρα ως parent.topright, parent.bottomleft κ.λπ. Η λέξη-κλειδί window και self θα αναφέρονταν στο πλαίσιο topleft.

Ως ένα απλό παράδειγμα της χρήσης των πλαισίων στην JavaScript, θα δημιουργήσουμε ένα HTML έγγραφο που διαιρεί το παράθυρο σε τέσσερα πλαίσια και ένα έγγραφο με ένα script για το πάνω αριστερό πλαίσιο. Τα κουμπιά στο πάνω αριστερό πλαίσιο ξεκινούν τους JavaScript χειριστές συμβάντων που εμφανίζουν κείμενο στα άλλα πλαίσια.

Για να αρχίσουμε, θα χρειαστούμε ένα έγγραφο με ένα σύνολο πλαισίων. Η Λίστα 12.1 δείχνει ένα απλό HTML έγγραφο, που διαιρεί το παράθυρο σε τέσσερα πλαίσια.

Λίστα 12.1 Ένα HTML έγγραφο που διαιρεί το παράθυρο σε τέσσερα πλαίσια

```
<frameset rows="*,*" cols="*,*">
<frame name="top_left" src="topleft.html">
<frame name="top_right" src="">
```

```
<frame name="bottom_left" src="">
<frame name="bottom_right" src="">
</frameset>
```

Το πρώτο πλαίσιο που ορίζεται εδώ, το top_left, θα περιέχει ένα HTML έγγραφο και ένα απλό script. Η Λίστα 12.2 δείχνει τον HTML κώδικα και την JavaScript για το πάνω αριστερό πλαίσιο.

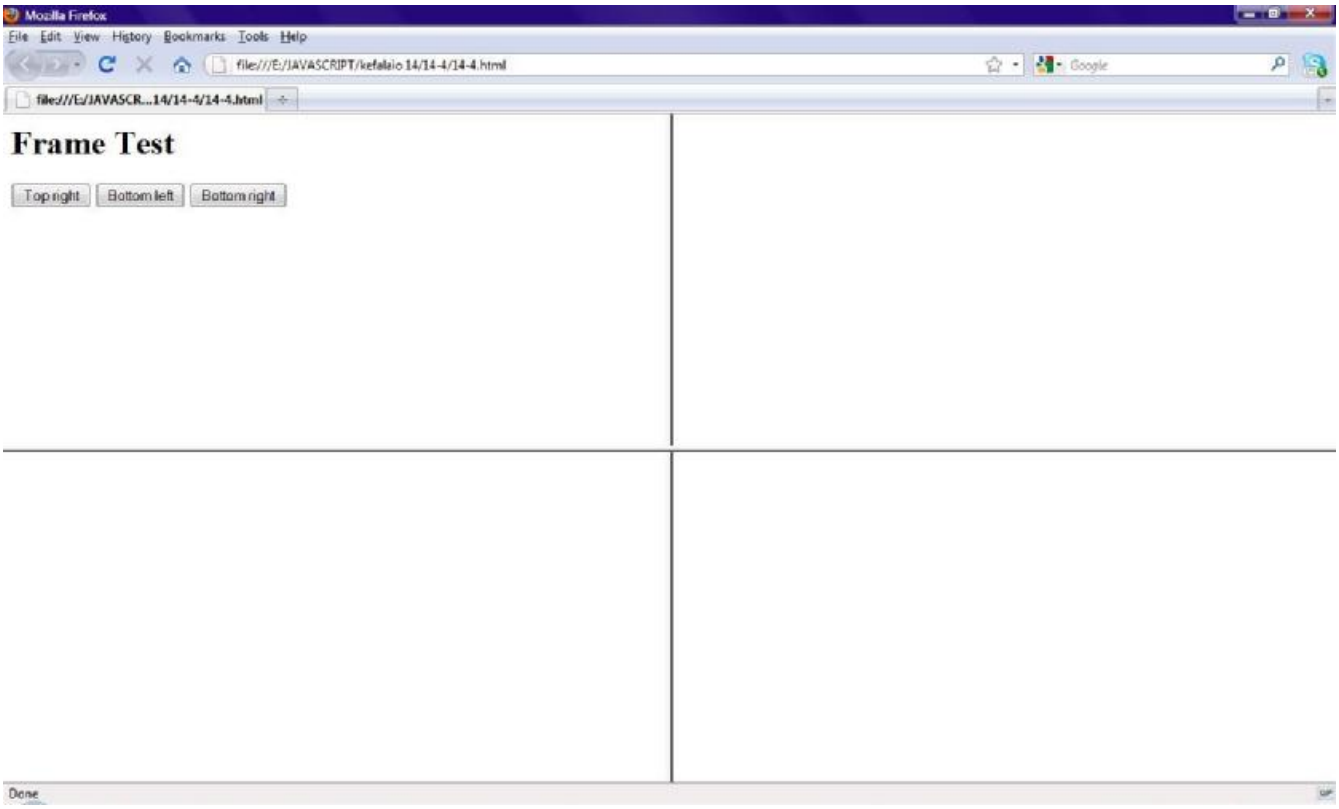
Λίστα 12.2 Η HTML και η JavaScript για το παράδειγμα των πλαισίων

```
<html>
<head>
<title>Frame Test</title>
<script language="javascript" type="text/javascript">
function FillFrame(framename)  {
/ / Εύρεση του αντικειμένου για το πλαίσιο frame
theframe=parent[framename];
/ / Άνοιγμα και καθάρισμα του εγγράφου του πλαισίου
theframe.document.open();
// Δημιουργία κάποιας εξόδου
theframe.document.write("<hl>JavaScript  Output</hl>");
theframe.document.write("<p>This text isinthe");
theframe.document.write(framename + " frame.</p>");
}
</script>
</head>
```

```
<body>
<h1>Frame Test</h1> <form name="form1">
<input type="button" value="Top right"
onClick="FillFrame('top_right');">
<input type="button" value="Bottom left"
onClick="FillFrame('bottom_left');">
<input type="button" id="js" value="Bottom right"
onClick="FillFrame('bottom_right');">
</form>
</body>
</html>
```

Το παρόν έγγραφο ορίζει τρία κουμπιά με χειριστές συμβάντων που καλούν τη συνάρτηση FillFrame με μια παράμετρο για το όνομα του πλαισίου. Η συνάρτηση βρίσκει το σωστό παιδί του αντικειμένου παραθύρου parent για το συγκεκριμένο πλαίσιο, χρησιμοποιεί το document.open για να δημιουργήσει ένα νέο έγγραφο στο πλαίσιο και χρησιμοποιεί το document.write για να εμφανίσει κείμενο στο πλαίσιο.

Για να δοκιμάσουμε αυτό το παράδειγμα, αποθηκεύουμε τη Λίστα 12.2 ως top-left.html στον ίδιο φάκελο με το έγγραφο του συνόλου πλαισίων από τη Λίστα 12.1 και φορτώνουμε τη Λίστα 12.1 σε έναν browser. Η Εικόνα 17 παρουσιάζει το αποτέλεσμα αυτού του παραδείγματος.



ΜΕΡΟΣ 5: ΜΙΑ ΑΝΑΦΟΡΑ ΣΤΟ AJAX

ΚΕΦΑΛΑΙΟ 13: AJAX - ΣΥΝΤΑΞΗ ΑΠΟΜΑΚΡΥΣΜΕΝΩΝ SCRIPT

13.1 Εισαγωγή στο AJAX

Η AJAX δεν είναι μια νέα γλώσσα προγραμματισμού, αλλά μια νέα τεχνική για δημιουργία καλύτερων, ταχύτερων και περισσότερο αλληλεπιδραστικών εφαρμογών για το web. Με την AJAX η JavaScript μπορεί να επικοινωνεί κατευθείαν με τον server με το αντικείμενο XMLHttpRequest. Με αυτό το αντικείμενο η JavaScript μπορεί να ανταλλάσσει δεδομένα με τον web server χωρίς να ξαναφορτώνει την σελίδα.

Η AJAX χρησιμοποιεί ασύγχρονη μεταφορά δεδομένων (HTTP requests) μεταξύ του browser και του web server, επιτρέποντας στις σελίδες web να απαιτούν μικρά bits πληροφορίας από τον server αντί για ολόκληρες σελίδες. Η τεχνική AJAX κάνει τις εφαρμογές του Internet μικρότερες, γρηγορότερες και πιο φιλικές προς το χρήστη.

Η AJAX είναι βασισμένη στα ακόλουθα web πρότυπα:

- JavaScript
- XML
- HTML
- CSS

Οι εφαρμογές AJAX είναι ανεξάρτητες της πλατφόρμας λογισμικού και του browser.

Οι εφαρμογές του Internet έχουν περισσότερα πλεονεκτήματα έναντι των εφαρμογών για desktop, μπορούν να απευθυνθούν σε μεγαλύτερο κοινό, είναι ευκολότερες να εγκατασταθούν και να υποστηριχθούν και ευκολότερο να αναπτυχθούν. Εντούτοις, οι εφαρμογές του Internet δεν είναι πάντα τόσο “πλούσιες” και φιλικές προς το χρήστη όπως παραδοσιακά είναι οι εφαρμογές

των desktop. Με την AJAX οι εφαρμογές του Internet μπορούν να εμπλουτιστούν και να γίνουν φιλικότερες προς το χρήστη.

13.2 Η AJAX χρησιμοποιεί το αντικείμενο XMLHttpRequest

Για να πάρουμε ή να στείλουμε πληροφορίες από και προς μια βάση δεδομένων ή ένα φάκελο σε έναν server με παραδοσιακή JavaScript, θα πρέπει να φτιάξουμε μια HTML φόρμα και ένας χρήστης θα πρέπει να πατήσει το κουμπί “Submit” για να στείλει ή να πάρει την πληροφορία, να περιμένει για τον server να απαντήσει, μετά μια νέα σελίδα να φορτωθεί με τα αποτελέσματα. Επειδή ο server επιστρέφει μια νέα σελίδα κάθε φορά που ο χρήστης υποβάλει την είσοδο, παραδοσιακές εφαρμογές web μπορεί να τρέχουν αργά και να τείνουν να είναι λιγότερο φιλικές προς το χρήστη.

Με την AJAX η JavaScript επικοινωνεί κατευθείαν με τον server μέσω του αντικειμένου XMLHttpRequest.

Με το αντικείμενο XMLHttpRequest μια web page μπορεί να κάνει ένα αίτημα και να πάρει μια απάντηση από έναν web server χωρίς να ξαναφορτώσει μια σελίδα. Ο χρήστης θα παραμείνει στην ίδια σελίδα και δεν θα προσέξει ότι τα script στέλνουν ένα αίτημα για μια σελίδα ή στέλνουν δεδομένα σε έναν server στο παρασκήνιο.

13.2.1 Το αντικείμενο XMLHttpRequest

Χρησιμοποιώντας το αντικείμενο XMLHttpRequest ένας προγραμματιστής μπορεί να ανανεώσει μια σελίδα με δεδομένα από έναν server αφού η σελίδα έχει φορτωθεί. Η AJAX έγινε δημοφιλής το 2005 από τη Google (με το Google Suggest).

Το Google Suggest χρησιμοποιεί το αντικείμενο XMLHttpRequest για να δημιουργήσει ένα πολύ δυναμικό web interface: όταν αρχίζεις να πληκτρολογείς στο πλαίσιο αναζήτησης της Google , ένα JavaScript στέλνει τα γράμματα σε έναν server και ο server επιστρέφει μια λίστα προτάσεων.

Το αντικείμενο XMLHttpRequest υποστηρίζεται από όλους τους κύρους browsers (Internet Explorer, Firefox, Chrome, Opera, Safari). Όλοι οι νέοι browsers χρησιμοποιούν το δομημένο με JavaScript αντικείμενο XMLHttpRequest για να δημιουργήσουν ένα αντικείμενο XMLHttpRequest .

Παρακάτω ενημερώνουμε το αρχείο “testAjax.html” με JavaScript που δημιουργεί ένα αντικείμενο XMLHttpRequest:

```
<html>
<head><title>First Ajax Application</title>
</head>
<body>
<script type=”text/javascript”>
function ajaxFunction() {
var xmlhttp;
if (window.XMLHttpRequest)
{
// code for IE7+, Firefox, Chrome, Opera, Safari
xmlhttp = new XMLHttpRequest();
}
else if (window.ActiveXObject)
{
//code for IE6, IE5
xmlhttp = new ActiveXObject (“Microsoft.XMLHTTP”);
}
```

```
else
{
alert ("Your browser does not support XMLHTTP!");
}
}
</script>
<form name = "myForm">
Name: <input type="text" name="username"/>
Time: <input type="text" name="time"/>
</form>
</body>
</html>
```

Στο παραπάνω παράδειγμα δημιουργήθηκε μια μεταβλητή που την ονομάσαμε xmlhttp για να κρατήσει το αντικείμενο XMLHttpRequest. Προσπαθήσαμε να δημιουργήσουμε το αντικείμενο XMLHttpRequest με την έκφραση xmlhttp = new XMLHttpRequest(). Αν αυτό αποτύχει δοκίμασε xmlhttp = new ActiveXObject ("Microsoft.XMLHTTP") (αυτό είναι για IE6 και IE5). Αν αποτύχει και αυτό τότε ο χρήστης έχει έναν πολύ παλιό browser και θα λάβει ένα μήνυμα που θα δηλώνει ότι ο browser δεν υποστηρίζει XMLHTTP.

Πριν να στείλουμε δεδομένα σε έναν server, θα πρέπει να κοιτάξουμε τρεις σημαντικές ιδιότητες του αντικειμένου XMLHttpRequest.

13.2.2 Οι ιδιότητες του XMLHttpRequest

Η ιδιότητα onreadystatechange

Μετά από ένα αίτημα στον server χρειαζόμαστε μια συνάρτηση για να λάβει τα δεδομένα που επιστρέφουν από τον server. Η ιδιότητα onreadystatechange αποθηκεύει μια συνάρτηση που θα επεξεργαστεί την απάντηση από τον server. Η συνάρτηση αποθηκεύεται στην ιδιότητα ώστε να κληθεί αυτόματα.

Ο παρακάτω κώδικας θέτει την ιδιότητα onreadystatechange και αποθηκεύει μια κενή συνάρτηση σε αυτή:

```
xmlhttp.onreadystatechange = function () {

//we are going to write some code here

}
```

Η ιδιότητα readyState

Η ιδιότητα readyState κρατά την κατάσταση της απάντησης του server. Κάθε φορά που η ιδιότητα readyState αλλάζει θα εκτελείται η συνάρτηση onreadystatechange.

Παρακάτω δίνονται πιθανές τιμές της ιδιότητας readyState:

Κατάσταση	Περιγραφή
0	Το αίτημα δεν αρχικοποιήθηκε
1	Το αίτημα έχει συσταθεί
2	Το αίτημα έχει σταλεί
3	Το αίτημα είναι υπό επεξεργασία
4	Το αίτημα ολοκληρώθηκε

ΠΙΝΑΚΑΣ 20

Προσθέτουμε μια δήλωση if στην συνάρτηση onreadystatechange για να δοκιμάσουμε αν η απάντηση ολοκληρώθηκε (σημαίνει ότι τώρα μπορούμε να πάρουμε τα δεδομένα μας):

```
xmlhttp.onreadystatechange = function () {

if (xmlhttp.readyState==4)

{

//Get data from the server's response

}

}
```

Η ιδιότητα responseText

Τα δεδομένα που στέλνονται πίσω από τον server μπορούν να έχουν ανακτηθεί με την ιδιότητα responseText. Τώρα, μπορούμε να θέσουμε την τιμή του πεδίου εισόδου "Time" ίσο με το responseText:

```
xmlhttp.onreadystatechange = function () {

If (xmlhttp.readyState == 4)

{

document.myForm.time.value = xmlhttp.responseText;
```

ΕΠΙΛΟΓΟΣ

Συμπερασματικά, ένα πρόγραμμα γραμμένο σε JavaScript είναι μια ακολουθία δηλώσεων, οι οποίες δουλεύουν μαζί με σκοπό να πετύχουν ένα συγκεκριμένο αποτέλεσμα, που αποτελούνται από συνιστώσες γραμματικών στοιχείων, όπως εκφράσεις και τελεστές. Επειδή η JavaScript είναι μια γλώσσα προγραμματισμού που εφευρέθηκε για χρήση στο Web, είναι προσανατολισμένη στις ειδικές ανάγκες των προγραμματιστών του Web. Το σύνολο των προκαθορισμένων αντικειμένων αντανακλά ευρέως τα χαρακτηριστικά των ιστοσελίδων, επιτρέποντας στο πρόγραμμα JavaScript να χειρίζεται, να τροποποιεί και να επιδρά στις σελίδες.

Διαλογικότητα είναι η κινητήρια δύναμη πίσω από την JavaScript, και έτσι τα περισσότερα προγράμματα σε JavaScript ξεκινούν με έναυσμα τις ενέργειες, οι οποίες συμβαίνουν στην ιστοσελίδα, συχνά από τον χρήστη. Στο πλαίσιο αυτό, ο σκοπός της JavaScript είναι να αποφεύγεται η στατική δομή των ιστοσελίδων και να εμφανίζονται δεδομένα εφαρμογών τα οποία μπορούν να επεξεργαστούν και να αλληλεπιδράσουν με το χρήστη.

Δεν υπάρχει έλλειψη έντυπου υλικού στο θέμα της JavaScript. Αφθονούν οι σελίδες στο διαδίκτυο, όπως επίσης τα tutorials και οι οδηγοί σχετικά με την JavaScript. Κανένα από αυτά δεν είναι δύσκολο να εντοπιστεί.

Στο πλαίσιο αυτής της πτυχιακής εργασίας έγινε μια προσπάθεια να αναφερθούν τα βασικά στοιχεία που απαρτίζουν την JavaScript, καθώς και παραδείγματα – εφαρμογές της. Οι δυνατότητες της JavaScript είναι πάρα πολλές και δεν μπορούν να συμπεριληφθούν στα πλαίσια μιας πτυχιακής. Γι’ αυτό τον λόγο δίνονται παρακάτω κάποιοι σύνδεσμοι στο internet που περιλαμβάνουν μεγάλο εύρος από script που μπορούν να ενσωματωθούν σε μια ιστοσελίδα, ανάλογα με τις ανάγκες του εκάστοτε προγραμματιστή.

http://www.javascriptkit.com/

http://www.wdvl.com/Authoring/JavaScript/Tutorial/

http://www.scripts.com/javascript-scripts/

http://www.tizag.com/javascriptT/

http://en.wikipedia.org/wiki/JavaScript

http://www.it.uom.gr/project/Dhtml_Jscripts/erg1.htm

http://www.w3schools.com/JS/default.asp

ΠΗΓΕΣ

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. ΟΔΗΓΟΣ ΤΗΣ JAVASCRIPT, ΣΥΓΓΡΑΦΕΑΣ ΠΟΛΟΚ ΤΖΟΝ, ΕΚΔΟΣΕΙΣ Μ. ΓΚΙΟΥΡΔΑΣ, 2006
2. ΜΑΘΕΤΕ ΤΗΝ JAVASCRIPT ΣΕ 24 ΩΡΕΣ, 4Η ΕΚΔΟΣΗ, ΣΥΓΓΡΑΦΕΑΣ ΜΟΝΚΟΥΡ ΜΙΚΑΕΛ, ΕΚΔΟΣΕΙΣ Μ. ΓΚΙΟΥΡΔΑΣ, 2007

ΙΣΤΟΣΕΛΙΔΕΣ

1. http://www.w3schools.com/js/default.asp
2. http://en.wikipedia.org/wiki/JavaScript
3. http://www.it.uom.gr/project/Dhtml_Jscripts/jvscr.htm