

**Ανοικτά Ακαδημαϊκά Μαθήματα**

**Τεχνολογικό Εκπαιδευτικό Ίδρυμα Αθήνας**

**Βάσεις Δεδομένων ΙΙ (Ε)**

**Ενότητα 9:** Στατικές και δυναμικές σελίδες (JSP)

Χ. Σκουρλάς, Α. Τσολακίδης, Δ. Βάσσης

Τμήμα Μηχανικών Πληροφορικής ΤΕ

|  |  |
| --- | --- |
| Το περιεχόμενο του μαθήματος διατίθεται με άδεια Creative Commons εκτός και αν αναφέρεται διαφορετικά | Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους. |

Περιεχόμενα

[1. Στατικές και δυναμικές σελίδες (JSP) 3](#_Toc401507995)

[Άσκηση 1 4](#_Toc401507996)

[Άσκηση 2 5](#_Toc401507997)

[Άσκηση 3 6](#_Toc401507998)

[Άσκηση 3.1 6](#_Toc401507999)

[Άσκηση 4 7](#_Toc401508000)

[Άσκηση 5 8](#_Toc401508001)

[Άσκηση 6 11](#_Toc401508002)

[Άσκηση 7 13](#_Toc401508003)

[Άσκηση 7.1 14](#_Toc401508004)

Στόχος του εργαστηρίου είναι η εξοικείωση με τη δημιουργία και τη διαχείριση δυναμικών ιστοσελίδων με χρήση της γλώσσας JSP. Μετά την επεξεργασία του εργαστηρίου ο ενδιαφερόμενος θα έχει κατανοήσει τα θέματα δημιουργίας δυναμικών φορμών και θέματα διαχείρισης δεδομένων για ενδεχόμενη παραπέρα επεξεργασία τους.

**Λέξεις κλειδιά:** Επαναληπτικοί βρόχοι, JSP, Διαχείριση ηλεκτρονικών φορμών

# Στατικές και δυναμικές σελίδες (JSP)

Μέχρι στιγμής είδαμε σελίδες HTML με τις οποίες μπορούμε να γράφουμε κείμενο και να το μορφοποιούμε όπως εμείς θέλουμε. Το κείμενο όμως που θα εμφανίζει η ιστοσελίδα μας δεν αλλάζει. Παραμένει πάντα το ίδιο. Τέτοιο κείμενο λέγεται στατικό και μια ιστοσελίδα που έχει στατικό κείμενο ονομάζεται στατική ιστοσελίδα (static web page).

**Τι γίνεται όμως όταν θέλουμε να έχουμε κείμενο που αλλάζει;**

Σκεφτείτε το εξής παράδειγμα στο Word. Έχουμε μια παράγραφο με στατικό κείμενο και στο τέλος εισάγουμε την ημερομηνία. Το κείμενο που δείχνει την ημερομηνία ανανεώνεται κάθε φορά που ανοίγουμε το έγγραφο με το word. Ένα τέτοιο κείμενο ονομάζεται δυναμικό κείμενο. Το Word έχει τον τρόπο να ανανεώνει την ημερομηνία κάθε φορά που ανοίγει το έγγραφο.

Πώς όμως θα φτιάξουμε μια ιστοσελίδα που κάνει την παραπάνω δουλειά;

Ιστοσελίδες που εμφανίζουν στατικό αλλά και δυναμικό κείμενο ονομάζονται δυναμικές ιστοσελίδες. Η εργασία αυτή δε μπορεί να γίνει μόνο με HTML. Χρειάζεται να ενσωματώσουμε και κάποια άλλη τεχνολογία για να γράφουμε δυναμικό κείμενο. Ο πιο διαδεδομένος τρόπος σήμερα για να γράφουμε δυναμικό κείμενο σε ιστοσελίδες είναι οι script γλώσσες προγραμματισμού ASP (Active Server Pages), JSP (Java Server Pages) και PHP (Hypertext PreProcessor). Οι παραπάνω script γλώσσες μας παρέχουν τον τρόπο να γράφουμε δυναμικό κείμενο και, γενικότερα, να επιτελούμε κάποιες διεργασίες που θα μπορούσαμε να εκτελέσουμε με γλώσσες προγραμματισμού. Έτσι, μπορούμε να γράψουμε και να διαβάσουμε από αρχεία, να ορίσουμε και να μεταβάλλουμε μεταβλητές, να εμφανίσουμε την τιμή μεταβλητών σε ιστοσελίδες (αντίστοιχα, στις γλώσσες προγραμματισμού θα εμφανίζαμε την τιμή τους στην οθόνη ή σε ένα παράθυρο, αν επρόκειτο για visual γλώσσες), να συνδεθούμε σε βάσεις δεδομένων κ.α.

Γενικότερα, μέσω των παραπάνω τεχνολογιών μπορούμε να γράψουμε ιστοσελίδες που περιέχουν στατικό κείμενο και προγραμματιστική λογική. Στην περίπτωσή μας θα ασχοληθούμε με τη γλώσσα JSP. Μια ιστοσελίδα JSP αποτελείται από κείμενο HTML, για να γράψουμε το κείμενο που θέλουμε, και από τμήματα κώδικα JSP όπου ενσωματώνουμε κώδικα Java. Κάθε αρχείο JSP έχει κατάληξη .jsp.

**Ερώτηση: Μπορεί ένα αρχείο που έχει μόνο κώδικα HTML να έχει κατάληξη .jsp;**

**Απάντηση:** ΝΑΙ. Αφού είπαμε ότι ένα αρχείο JSP έχει κώδικα HTML και JSP. Αν δεν έχει κώδικα JSP παραμένει ακόμη ένα αρχείο JSP, αφού η HTML είναι αναφαίρετο κομμάτι της γλώσσας JSP. Ένα παράδειγμα JSP είναι το ακόλουθο:

Άσκηση 1

<HTML>

<BODY>

<% int x = 3; %>

Hello! The value of the variable x is: <%= x %>.<p>

<% x=x+3; %>

The value of the variable x is changed: <%= x %>.

</BODY>

</HTML>

Αποθηκεύστε το παραπάνω αρχείο ως ex1.jsp, στo project baseis\_x που είχατε φτιάξει για τα παραδείγματα του εργαστηρίου. Δείτε το αποτέλεσμα στο browser

πληκτρολογώντας <http://localhost:8080/baseis_x/ex1.jsp>.

Στο παραπάνω αρχείο, ορίσαμε μια μεταβλητή x. Ο ορισμός έγινε με κώδικα Java, που περικλείεται ανάμεσα σε ειδικά σύμβολα που ονομάζονται tags. (<% και %>).

Γράφουμε κώδικα HTML και, στο σημείο που θέλουμε να εμφανίσουμε την τιμή της μεταβλητής, γράφουμε τη μεταβλητή και την περικλείουμε στα tags <%= και %>. Το υπόλοιπο μέρος του κώδικα καταλαβαίνετε προφανώς τι κάνει. Όπως βλέπουμε, τον κώδικα Java τον διαχωρίζουμε από τον κώδικα HTML μέσω ειδικών tags. Ο ρόλος των tags είναι για να καταλαβαίνει ο compiler ότι δεν πρόκειται για κείμενο HTML, αλλά για κώδικα Java. Ο κώδικας JSP δεν περιορίζεται σε tags που εσωκλείουν κώδικα Java. Υπάρχουν πάρα πολλά διαφορετικά tags που τα χρησιμοποιούμε για να κάνουμε συγκεκριμένες διεργασίες. Στην παρούσα ενότητα θα δούμε μόνο δυο απλά tags.

**Ο ρόλος των Web Containers**

Βρείτε ένα αρχείο .html που έχετε φτιάξει. Κάντε διπλό κλικ πάνω του για να ανοίξει με έναν browser. Έχετε κάποιο πρόβλημα να το δείτε; Μάλλον όχι.

Προσπαθήστε να κάνετε το ίδιο και με το αρχείο ex1.jsp που φτιάξατε παραπάνω, δηλαδή να το ανοίξετε με έναν web browser. Έχετε κάποιο πρόβλημα να το δείτε; Μάλλον ναι. Γιατί συμβαίνει αυτό;

Η γλώσσα HTML είναι μια ψευδογλώσσα που μπορεί να τη μεταφράσει ο browser και να εμφανίσει το κατάλληλο αποτέλεσμα. Δεν ισχύει όμως το ίδιο και με τον κώδικα των δυναμικών σελίδων όπως οι JSP, οι οποίες περιέχουν και κώδικα Java. Απαιτείται λοιπόν κάποιος άλλος μηχανισμός από την πλευρά του server ο οποίος θα φορτώνει τα αρχεία JSP, θα τα μεταφράζει σε καθαρό κώδικα HTML αντικαθιστώντας τις τιμές των δυναμικών παραμέτρων με στατικές, και θα τα δίνει στο browser του client, ο οποίος θα μεταφράζει τo κείμενο HTML που του έδωσε ο server. Ένας τέτοιος μηχανισμός-πρόγραμμα ονομάζεται *web container*. Ο ρόλος του web container είναι να καθιστά δυνατή τη χρήση αρχείων JSP σε μια εφαρμογή. Αν λοιπόν θέλουμε να φτιάξουμε μια εφαρμογή με JSP, δε μας αρκεί ένας απλός web server, αλλά πρέπει να εγκαταστήσουμε και έναν web container. Οι περισσότεροι μεγάλοι application servers έρχονται σήμερα με ενσωματωμένο web container (π.χ. Inspire, iPlanet, WebSphere). Υπάρχουν ωστόσο plugins που εγκαθιστούν web containers σε υπάρχοντες web servers ώστε να τρέχουν τα JSP (π.χ. JRun, ServletExec). O πιο διαδεδομένος web container είναι ο Apache Tomcat, o οποίος είναι open source και έχει ενσωματωμένο και web server. Τον Tomcat, ως γνωστό, τον έχετε ήδη εγκαταστήσει στον υπολογιστή σας.

**Scriptlet και expression tags**

Ένα τμήμα κώδικα Java που περικλείεται μέσα σε ένα αρχείο JSP ονομάζεται *scriptlet.* To scriptlet το περικλείουμε ανάμεσα σε scriptlet tags (<% και %>) ως εξής:

html code

<% Java code %>

html code

Αν θέλουμε να εμφανίσουμε την τιμή μιας μεταβλητής στον browser τότε γράφουμε τη μεταβλητή στο σημείο που θέλουμε να εμφανιστεί και την περικλείουμε σε expression tags (<%= και %>) ως εξής:

Html

text <%= paramName %>

text

html

Άσκηση 2

Ακολουθεί ένα παράδειγμα που εμφανίζει την ημερομηνία.

<%@ page import="java.util.\*" %>

<html>

<body>

<%

Date theDate = new Date();

%>

Hello! The time is now <%= theDate %>.

</body>

</html>

Σώστε το παρακάτω κομμάτι κώδικα ως ex2.jsp και αποθηκεύστε το στον κατάλογο baseis\_x. Δείτε το αποτέλεσμα στο web browser πληκτρολογώντας το κατάλληλο URL.

Η πρώτη γραμμή του αρχείου είναι ένα ειδικό tag για να εισάγουμε τις βιβλιοθήκες της Java για ημερομηνία. Στη συνέχεια αρχίζει ο κώδικας HTML. Παρεμβάλλεται ένα scriptlet όπου ορίζουμε μια μεταβλητή που κρατά την ημερομηνία. Συνεχίζουμε τον κώδικα HTML και, στο σημείο που θέλουμε να μας εμφανίσει την ημερομηνία παρεμβάλλουμε expression tags όπου εσωκλείουμε τη μεταβλητή που ορίσαμε πιο πριν.

Άσκηση 3

Ας δούμε ένα πιο σύνθετο παράδειγμα. Αντιγράψτε τον παρακάτω κώδικα σε ένα αρχείο και ονομάστε το game.jsp. Δημιουργείστε το project baseis\_x. Δείτε το αποτέλεσμα στο web browser πληκτρολογώντας το κατάλληλο URL. Κάντε reload τη σελίδα για επανάληψη.

<%

int dice1 = (int)(6\*Math.random()+1);

int dice2 = (int)(6\*Math.random()+1);

String answer;

if( dice1==dice2 )

answer ="Congradulations. You won!!!";

else

answer = "You are GADEMIS. Please try again...";

%>

<html>

<body>

Let's play <p>

You scored <%= dice1 %> and <%= dice2 %>.<p>

The System says: <%= answer %>

</body>

</html>

Το παραπάνω αρχείο JSP ενσωματώνει ένα κομμάτι κώδικα Java (scriptlet) που παράγει δυο τυχαίους αριθμούς (dice1 και dice2, καθώς και ένα string (answer), με βάση το αποτέλεσμα. Στη συνέχεια, με κώδικα HTML και με το expression tag <%=, φτιάχνουμε το περιεχόμενο της ιστοσελίδας, εμφανίζοντας τα αποτελέσματα στο χρήστη.

Έχετε υπόψη ότι μπορούμε να γράψουμε τον κώδικα Java οπουδήποτε μέσα στο αρχείο. Αρκεί να τον ενσωματώνουμε στα scriptlet tags. Για παράδειγμα, στο παραπάνω, αρχείο θα μπορούσαμε να σπάσουμε τον κώδικα Java σε δυο scriptlets όπως παρακάτω, και να τα αναμείξουμε με τον κώδικα HTML:

Άσκηση 3.1

<%

int dice1 = (int)(6\*Math.random()+1);

int dice2 = (int)(6\*Math.random()+1);

String answer;

%>

<html>

<body>

<%

if( dice1==dice2 )

answer ="Congradulations. You won!!!";

else

answer = "You are GADEMIS. Please try again...";

%>

Let's play <p>

You scored <%= dice1 %> and <%= dice2 %>.<p>

The System says: <%= answer %>

</body>

</html>

**Επαναληπτικοί βρόχοι σε JSP**

Μέσω των επαναληπτικών βρόχων που προσφέρει η Java (while, for, do-while) μπορούμε να εμφανίσουμε επαναληπτικά κείμενο HTML στο browser. Η φιλοσοφία είναι απλή, αρκεί να καταλάβουμε το παρακάτω παράδειγμα. Δημιουργείστε το αρχείο loop.jsp και δείτε το αποτέλεσμα στον browser.

Άσκηση 4

<body>

LOOP EXAMPLE:<p>

<%

for (int i=1; i<=5; i++)

{

%>

Loop executed <%= i %> times.<br>

<%

}

%>

</body>

</html>

Όπως βλέπετε, γράψαμε ένα scriptlet όπου ορίζουμε την αρχή του βρόχου. Μέσα στο scriptlet γράφουμε κώδικα HTML (και αν θέλουμε κάποια JSP tags, όπως το expression tag που γράψαμε παραπάνω). Κλείνουμε το βρόχο γράφοντας ένα scriptlet που περιέχει μόνο το άγκιστρο ( } ) που κλείνει το βρόχο. Το κείμενο HTML που περικλείεται μέσα στο βρόχο θα εκτελεστεί όσες φορές γίνει η επανάληψη. Προσοχή στη χρήση των scriptlet tags. Η αρχή του βρόχου for είναι ένα κομμάτι κώδικα Java, οπότε τον περικλείουμε στα tags <% και %>. Το κείμενο HTML που γράφουμε στο browser και βρίσκεται μέσα στο βρόχο δεν είναι κώδικας Java, οπότε δεν το περικλείουμε σε scriptlet tags. Ο βρόχος όμως δεν έκλεισε. Μας μένει ένα άγκιστρο. Ακόμη όμως κι αυτό το αγκιστράκι είναι από μόνο του ένα κομμάτι κώδικα Java, οπότε πρέπει να το περικλείσουμε σε tags.

**Τρόπος ανάπτυξης αρχείων JSP**

Γενικότερα, η πιο απλή φιλοσοφία για την ανάπτυξη (υλοποίηση) αρχείων JSP είναι η

εξής:

• Γράφουμε μια στατική σελίδα HTML η οποία θα είναι η δυναμική σελίδα που θέλουμε να φτιάξουμε, απλά το δυναμικό κομμάτι του κειμένου θα έχει κάποιες ενδεικτικές τιμές στις παραμέτρους. Στο παραπάνω παράδειγμα στην αρχική στατική σελίδα που φτιάξαμε, βάλαμε κάποιες ενδεικτικές τιμές για τα χρώματα και το μέγεθος της γραμματοσειράς. Η παραπάνω διαδικασία είναι εξαιρετικά εύκολη,

αν αναλογιστούμε την πληθώρα προγραμμάτων που γράφουν HTML.

• Αποθηκεύουμε την παραπάνω σελίδα με κατάληξη jsp.

• Προσθέτουμε το τμήμα της προγραμματιστικής λογικής (κώδικας java ή κάτι άλλο, όπως θα δούμε παρακάτω).

• Αντικαθιστούμε τα δυναμικά μέρη του κειμένου (που, προς το παρόν, εμφανίζονται στατικά) με τις παραμέτρους των οποίων η τιμή θέλουμε να εμφανίζεται.

Άσκηση 5

Για να δούμε ένα παράδειγμα, ας παίξουμε λίγο με τα χρώματα. Θέλουμε να φτιάξουμε μια ιστοσελίδα η οποία, κάθε φορά που φορτώνεται να έχει διαφορετικό φόντο, και διαφορετικό χρώμα και μέγεθος γραμματοσειράς.

Πρώτα φτιάχνουμε μια απλή σελίδα με τα παραπάνω στοιχεία fixed. Αντιγράψτε την παρακάτω απλή σελίδα HTML σε ένα αρχείο και ονομάστε το colors.jsp.

Αποθηκεύστε το στον κατάλογο baseis\_x και δείτε το αποτέλεσμα στον browser.

<html>

<body bgcolor="#CCCCCC" text="#000000">

<font size="+2">

With dynamic pages you can have different color on each visit !!! <p>

You can also have different font !!!

</font>

</body>

</html>

Ο παραπάνω κώδικας απλά θέτει το background και το text color σε γκρι και μαύρο αντίστοιχα (υπάρχουν συγκεκριμένοι κωδικοί όπως βλέπετε στη γραμμή 2), αυξάνει το φόντο κατά το διπλάσιο του default και γράφει ένα απλό μήνυμα. Πώς μπορούμε να έχουμε διαφορετικό background color, text color και font size κάθε φορά που φορτώνουμε την ιστοσελίδα; Αρκεί να ορίζονται δυναμικά τρεις τιμές: Η #CCCCCC που ορίζει το bgcolor, η #000000 που ορίζει το text color και ο αριθμός 2 που ορίζει το μέγεθος της γραμματοσειράς. Τις τρεις αυτές τιμές πρέπει να τις παράγουμε τυχαία. Ιδού ο τρόπος. Τροποποιήστε το αρχείο που φτιάξατε πριν όπως παρακάτω και δείτε το αποτέλεσμα στο browser, κάνοντας refresh τη σελίδα.

<%

String textColor = Integer.toHexString((int)(16777215\*Math.random()));

String backgrColor = Integer.toHexString((int)(16777215\*Math.random()));

int fontSize = ((int)(4\*Math.random()+1));

%>

<html>

<body bgcolor=<%= backgrColor%> text=<%= textColor%>>

<font size="+<%= fontSize%>">

With dynamic pages you can have different color on each visit !!! <p>

You can also have different font size!!!

</font>

</body>

</html>

Αυτό που κάναμε είναι το εξής:

*Προγραμματιστική λογική*: Προσθέσαμε ένα scriptlet όπου ορίζουμε τρεις παραμέτρους μέσω Java, οι οποίες παράγουν δυο strings που αντιπροσωπεύουν το text και το background color καθώς και έναν integer που αντιπροσωπεύει το μέγεθος της γραμματοσειράς της ιστοσελίδας. Η παραγωγή των strings γίνεται μετατρέποντας έναν τυχαίο αριθμό σε δεκαεξαδικό και μετά σε string.

*Αντικατάσταση στατικού κειμένου με δυναμικό*: Αντί να ορίσουμε ένα standard χρώμα για το text και το background, το ορίζουμε δυναμικά, μέσω των παραμέτρων που δημιουργήσαμε, οι οποίες παίρνουν διαφορετικές τιμές σε κάθε φόρτωμα της ιστοσελίδας. Το ίδιο γίνεται και για τo font size.

**Πέρασμα πληροφορίας σε αρχείο JSP από φόρμα HTML**

Μέχρι στιγμής είδαμε αρχεία JSP που παράγουν δυναμικό κείμενο HTML στο browser, με βάση κάποιες τυχαίες μεταβλητές ή με κάποιο επαναληπτικό βρόχο. Στην πλειονότητα των περιπτώσεων όμως, η χρησιμότητα των JSP και, κατ’ επέκταση, των δυναμικών σελίδων, έγκειται στο ότι μπορούν να παράγουν δυναμικό κείμενο, με βάση πληροφορία που δίνει ο χρήστης. Για παράδειγμα, σε ιστοσελίδες για καταστήματα υπολογιστών, ο χρήστης έχει τη δυνατότητα να δει όλα τα μοντέλα μιας

συγκεκριμένης μάρκας μόνο, ή όλα τα μοντέλα με συγκεκριμένο τύπο επεξεργαστή.

Για να υλοποιηθεί μια εφαρμογή όπως οι παραπάνω, πρέπει να υπάρχουν τα εξής:

Μια ιστοσελίδα HTML που θα κάνει τα εξής:

● Θα έχει μια φόρμα όπου ο χρήστης θα εισάγει τιμές σε κάποιες παραμέτρους.

● Θα καλέσει ένα αρχείο JSP στο οποίο θα δώσει τις τιμές κάποιων παραμέτρων. Το αρχείο αυτό θα κληθεί μόλις πατηθεί το κουμπί της φόρμας. Το αρχείο HTML θα φύγει τότε από το browser και θα φορτωθεί το αρχείο JSP.

Ένα αρχείο JSP που θα κληθεί από την ανωτέρω φόρμα το οποίο θα κάνει τα

ακόλουθα:

● Θα πάρει τις τιμές των παραμέτρων που εισήγαγε ο χρήστης.

● Θα τις επεξεργαστεί με κώδικα Java.

● Θα παράγει ένα δυναμικό κείμενο με HTML, βάσει των τιμών των παραμέτρων του χρήστη, το οποίο θα εμφανιστεί στο browser.

Έχουμε ήδη δει πώς φτιάχνουμε μια φόρμα και πώς καλούμε ένα άλλο αρχείο (σε μορφή link), όταν πατηθεί το κουμπί submit της φόρμας. Το πιο βασικό που πρέπει να θυμόμαστε είναι το εξής:

Στον ορισμό της φόρμας δηλώνουμε και το αρχείο που θα φορτωθεί μόλις πατηθεί το κουμπί της. Το αρχείο αυτό θα πάρει τις παραμέτρους που έδωσε ο χρήστης.

Η δήλωση αυτή γίνεται ως εξής:

<form name="formName" method="post" action="file.jsp">

Με τη δήλωση action="file.jsp", δηλώνουμε ότι, μόλις πατηθεί το κουμπί της φόρμας (το οποίο είναι στην ουσία ένα hyperlink), θα φορτωθεί στον browser το αρχείο file.jsp και θα πάρει τις τιμές απ’ όλες τις παραμέτρους της φόρμας.

**Πώς όμως θα πάρουμε τις τιμές των παραμέτρων της φόρμας ενός αρχείου στο JSP;**

**Η διαδικασία είναι η εξής:**

Μόλις μια φόρμα καλέσει ένα αρχείο JSP, του δίνει κάποιες παραμέτρους. Το JSP μπορεί να πάρει τις τιμές από αυτές τις παραμέτρους που του δώσανε, μέσω μιας συνάρτησης της Java:

**request.getParameter(“param”)**

όπου param είναι το όνομα μιας παραμέτρου της φόρμας. Στο παρακάτω σχήμα απεικονίζεται γραφικά η διαδικασία κατά την οποία ο χρήστης δίνει σε μια φόρμα δυο αριθμούς, γίνεται πρόσθεση και εμφανίζεται το αποτέλεσμα.



Όπως βλέπουμε, ο ρόλος του JSP είναι τριπλός. Παίρνει τις παραμέτρους από το αρχείο HTML, επιτελεί το προγραμματιστικό έργο μέσω Java και παράγει το κείμενο που θα εμφανιστεί στο browser μέσω HTML.

Το παρακάτω παράδειγμα παίρνει τις τιμές που έδωσε ο χρήστης σε μια φόρμα και τις εμφανίζει στο browser.

Άσκηση 6

Αντιγράψτε τον παρακάτω κώδικα σε ένα αρχείο, ονομάστε το yourself.html.

<html>

<body>

Tell me about yourself...<p>

<form name="yourself" method="post" action="yourself.jsp">

What is your name?: <input type="text" name="yourName"><p>

How old are you?: <input type="text" name="yourAge"><p>

Where are you from?: <input type="text" name="yourCity"><p>

What is your favorite pet?: <input type="text" name="yourPet"><p>

May I have your email?: <input type="text" name="yourEmail"><p>

<input type="submit" value="Thank you!">

</form>

</body>

</html>

Αντιγράψτε τώρα τον παρακάτω κώδικα σε ένα αρχείο yourself.jsp και αποθηκεύστε το στον ίδιο φάκελο.

<%

String userName = request.getParameter("yourName");

String userAge = request.getParameter("yourAge");

String userCity = request.getParameter("yourCity");

String userPet = request.getParameter("yourPet");

String userEmail = request.getParameter("yourEmail");

%>

<html>

<body>

This is what I learned from you:<p>

Your name is <%=userName%>.<p>

You are <%=userAge%> years old.<p>

You are from <%=userCity%>.<p>

Your favorite pet is <%=userPet%>.<p>

I can find you in the following address: <%=userEmail%>.<p>

Have a nice day!!!<p>

<a href="yourself.html">Go Back</a>

</body>

</html>

Δείτε το αποτέλεσμα πληκτρολογώντας στο browser το URL του αρχείου

*yourself.html.*

Το αρχείο yourself.html έχει μια φόρμα με 5 text boxes καθένα από τα οποία δέχεται τιμή για μια παράμετρο: *yourName, yourAge, yourCity, yourPet, yourEmail*.

Μόλις πατηθεί το κουμπί “Thank You” θα φορτωθεί το αρχείο *yourself.jsp*.

Στο αρχείο yourself.jsp γίνονται τα εξής: Αρχικά παίρνουμε τις τιμές από τις παραμέτρους της φόρμας και τις καταχωρούμε σε τοπικές μεταβλητές: *userName, userAge, userCit, userPet, userEmail*. Η φόρμα δε μας απασχολεί πλέον καθόλου. Δουλεύουμε με τις μεταβλητές που έχουμε ορίσει, οι οποίες κρατούν την πληροφορία του χρήστη. Στη συνέχεια γράφουμε το κείμενο που θα εμφανιστεί στον browser. Γράφουμε HTML και, όπου χρειάζεται, παρεμβάλλουμε κάποια μεταβλητή κλεισμένη σε expression tags.

Άσκηση 7

Αντιγράψτε τον παρακάτω κώδικα στο αρχείο input.html και αποθηκεύστε το στο φάκελο baseis\_x.

<html>

<body>

What is your sex?<br>

Type 1 for boy or 2 for girl.<p>

<form name="mySelf" method="post" action="respond.jsp">

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

<input type="submit" value="Continue">

</form>

</body>

</html>

Αντιγράψτε τον παρακάτω κώδικα στο αρχείο respond.jsp και αποθηκεύστε το στο φάκελο baseis\_x.

<%

String userSex = request.getParameter("sex");

String resp;

if(userSex.equals("1")) resp = "Let's talk about cars.";

else if(userSex.equals("2")) resp = "Let's talk about fashion.";

else resp = "Invalid choice. Try again.";

%>

<html>

<body>

<%= resp%><p>

<a href="input.html">Go Back</a>

</body>

</html>

Δείτε το αποτέλεσμα πληκτρολογώντας το URL του αρχείου input.html στο browser.

Στο αρχείο HTML έχουμε μια φόρμα όπου ο χρήστης δίνει τιμή σε μια παράμετρο sex. Μόλις πατηθεί το κουμπί της φόρμας καλείται το αρχείο respond.jsp. Το αρχείο JSP παίρνει την τιμή της παραμέτρου που του έδωσε η φόρμα και την καταχωρεί σε μια δική του μεταβλητή με όνομα userSex. Στη συνέχεια παράγει ένα κείμενο στη μεταβλητή resp, ανάλογα με την τιμή που έδωσε ο χρήστης. Το κείμενο αυτό είναι μια πρόταση από το κείμενο που θα εμφανιστεί στο browser. Στη συνέχεια γράφουμε το κείμενο HTML που θα εμφανιστεί στο browser, το οποίο φυσικά εξαρτάται από την τιμή της μεταβλητής resp.

**Πέρασμα πληροφορίας σε αρχείο JSP μέσω URL**

Πολλές φορές είναι επιθυμητό να περνάμε τις παραμέτρους σε ένα αρχείο JSP μέσω του URL του, χωρίς να τις έχει δώσει ο χρήστης σε φόρμα. Τέτοιες περιπτώσεις έχουμε όταν ένα αρχείο JSP ή HTML έχει πάρει ήδη κάποιες παραμέτρους και θέλει να τις δώσει σε κάποιο άλλο αρχείο JSP. Αυτό γίνεται εύκολα, αν ορίσουμε το όνομα και την τιμή των παραμέτρων αυτών στο URL του αρχείου JSP που θα τις πάρει. Η μορφή του URL του αρχείου JSP που θα πάρει τις παραμέτρους θα έχει τότε την εξής μορφή:

http://server\_IP:port/path/file.jsp?param1=value1&param2=value2

Για να γίνει το παραπάνω πιο αντιληπτό, γράψτε στο browser το εξής URL:

<http://localhost:8080/baseis_x/respond.jsp?sex=1>

Θα παρατηρήσετε ότι έχουμε το ίδιο αποτέλεσμα όπως αν θα χρησιμοποιούσαμε τη φόρμα του αρχείου input.html. Φυσικά, δεν υπάρχει ποτέ περίπτωση να βάλουμε το χρήστη να πληκτρολογεί παραμέτρους σε ένα URL που θα γράψει. Τα γράφουμε εμείς στον κώδικά μας.

Για να γίνει το παραπάνω πιο αντιληπτό, θα αλλάξουμε τον κώδικα του αρχείου input.html, ώστε η παράμετρος sex να δίνεται από το URL του JSP που θα κληθεί.

Άσκηση 7.1

Αντιγράψτε τον παρακάτω κώδικα σε ένα αρχείο, ονομάστε το input2.html.

<html>

<body>

What is your sex?<p>

<a href="respond.jsp?sex=1" > Boy </a><br>

<a href="respond.jsp?sex=2" > Girl </a><br>

</body>

</html>

Παρατηρήστε ότι έχουμε ακριβώς το ίδιο αποτέλεσμα όπως και στο προηγούμενο παράδειγμα. Παρατηρήστε επίσης ότι τώρα, μόλις φορτώνεται το αρχείο JSP φαίνονται και οι παράμετροι που δώσαμε.

Το παραπάνω αρχείο HTML απλά δημιουργεί δυο hyperlinks στο αρχείο JSP, περνώντας διαφορετική τιμή για την παράμετρο sex κάθε φορά. Δεν έχουμε γράψει όλο το path του URL (absolute path) αλλά μόνο το όνομα του αρχείου (relative path) αφού τα δυο αρχεία βρίσκονται στον ίδιο server.

**ΠΡΟΣΟΧΗ:** Έχετε υπόψη κάτι σημαντικό. Δεν περικλείουμε τις τιμές των παραμέτρων σε quotes, τις αφήνουμε όπως είναι. Ακόμη και για τα strings. Αν ένα string έχει και κενά, τότε τα κενά τα αντικαθιστούμε με το σύμβολο ‘ % ’.

Τώρα μπορούμε να καταλάβουμε γιατί όταν σε κάποια sites κάνουμε κλικ σε ένα link, προκύπτει στο browser ένα URL με μήκος κάποιων χιλιομέτρων και με διάφορα σύμβολα όπως ‘ ? ‘, ‘ = ‘, ‘ % ‘ κ.α.

|  |
| --- |
| **Ανοικτά Ακαδημαϊκά Μαθήματα**  **Τεχνολογικό Εκπαιδευτικό Ίδρυμα Αθήνας** |
| **Τέλος Ενότητας** |
| **Χρηματοδότηση**   * Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα. * Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο ΤΕΙ Αθήνας**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού. * Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους. |

**Σημειώματα**

**Σημείωμα Αναφοράς**

Copyright ΤΕΙ Αθήνας, Χ. Σκουρλάς, Α. Τσολακίδης, Δ. Βάσσης 2014. Χ. Σκουρλάς, Α. Τσολακίδης, Δ. Βάσσης. «Βάσεις Δεδομένων ΙΙ (Ε). Ενότητα 9: Στατικές και δυναμικές σελίδες (JSP)». Έκδοση: 1.0. Αθήνα 2014. Διαθέσιμο από τη δικτυακή διεύθυνση: [ocp.teiath.gr](https://ocp.teiath.gr/).

**Σημείωμα Αδειοδότησης**

Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά, Μη Εμπορική Χρήση Παρόμοια Διανομή 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό. Οι όροι χρήσης των έργων τρίτων επεξηγούνται στη διαφάνεια «Επεξήγηση όρων χρήσης έργων τρίτων».

Τα έργα για τα οποία έχει ζητηθεί άδεια αναφέρονται στο «Σημείωμα Χρήσης Έργων Τρίτων».

[](file:///C:\Users\pantelis\Downloads\%5b1%5d%20http:\creativecommons.org\licenses\by-nc-sa\4.0\)

[1] http://creativecommons.org/licenses/by-nc-sa/4.0/

Ως **Μη Εμπορική** ορίζεται η χρήση:

* που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο
* που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο
* που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

**Επεξήγηση όρων χρήσης έργων τρίτων**

|  |  |
| --- | --- |
| © | Δεν επιτρέπεται η επαναχρησιμοποίηση του έργου, παρά μόνο εάν ζητηθεί εκ νέου άδεια από το δημιουργό. |
| διαθέσιμο με άδεια CC-BY | Επιτρέπεται η επαναχρησιμοποίηση του έργου και η δημιουργία παραγώγων αυτού με απλή αναφορά του δημιουργού. |
| διαθέσιμο με άδεια CC-BY-SA | Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού, και διάθεση του έργου ή του παράγωγου αυτού με την ίδια άδεια. |
| διαθέσιμο με άδεια CC-BY-ND | Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού. Δεν επιτρέπεται η δημιουργία παραγώγων του έργου. |
| διαθέσιμο με άδεια CC-BY-NC | Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού. Δεν επιτρέπεται η εμπορική χρήση του έργου. |
| διαθέσιμο με άδεια CC-BY-NC-SA | Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού και διάθεση του έργου ή του παράγωγου αυτού με την ίδια άδεια. Δεν επιτρέπεται η εμπορική χρήση του έργου. |
| διαθέσιμο με άδεια CC-BY-NC-ND | Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού. Δεν επιτρέπεται η εμπορική χρήση του έργου και η δημιουργία παραγώγων του. |
| διαθέσιμο με άδεια CC0 Public Domain | Επιτρέπεται η επαναχρησιμοποίηση του έργου, η δημιουργία παραγώγων αυτού και η εμπορική του χρήση, χωρίς αναφορά του δημιουργού. |
| διαθέσιμο ως κοινό κτήμα | Επιτρέπεται η επαναχρησιμοποίηση του έργου, η δημιουργία παραγώγων αυτού και η εμπορική του χρήση, χωρίς αναφορά του δημιουργού. |
| χωρίς σήμανση | Συνήθως δεν επιτρέπεται η επαναχρησιμοποίηση του έργου. |

**Διατήρηση Σημειωμάτων**

* Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:
* το Σημείωμα Αναφοράς
* το Σημείωμα Αδειοδότησης
* τη δήλωση Διατήρησης Σημειωμάτων
* το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει) μαζί με τους συνοδευόμενους υπερσυνδέσμους.