Σημειώσεις για Βοηθητικές Κλάσεις Java

& Χειρισμό Παραμέτρων Φορμών από Servlet API

Ορισμένες χρήσιμες κλάσεις και διασυνδέσεις (interfaces) στην Java

Emumeration (Interface)

Επιτρέπει την διαδοχική παραλαβή προς επεξεργασία μιας ομάδας αντικειμένων.

Μια κλάση που υλοποιεί την διασύνδεση Enumeration πρέπει να παρέχει δύο βασικές μεθόδους

hasMoreElements() Επιστρέφει true αν υπάρχουν διαθέσιμα αντικείμενα να επιστραφούν

nextElement() Επιστρέφει το επόμενο αντικείμενο.

Μεταβλητές μπορεί να δηλώνονται σαν interfaces και να τους εκχωρούνται αντικείμενα των οποίων η κλάση υλοποιεί το interface. Δεν χρειάζεται ο προγραμματιστής να γνωρίζει λεπτομέρειες για την ακριβή κλάση που χρησιμοποιήθηκε για να χρησιμοποιήσει το αντικείμενο.

Η διασύνδεση Enumeration μπορεί να χρησιμοποιηθεί και με Java Generics δηλαδή το χαρακτηριστικό της Java να δημιουργούνται σύνθετοι τύποι. Έτσι μπορούμε να δηλώνουμε μεταβλητές τύπου Enumeration<String>. Με το τρόπο αυτό αναμένουμε να λαμβάνουμε αντικείμενα που είναι κλάσης String.

StringTokenizer

Ανήκει στο πακέτο java.util

Είναι μια βοηθητική κλάση που χρησιμοποιείται για να διασπά κάποιο String που περιέχει διαχωριστικούς χαρακτήρες (delimiters) στα επιμέρους τμήματα. Για παράδειγμα μπορούμε να έχουμε το string “oranges,apples,bananas” και να θέλουμε να εξάγουμε τις λέξεις oranges, apples, bananas.

Δημιουργούμε ένα αντικείμενο

st = StringTokenizer(myString, “,”)

Χρησιμοποιούμε την μέθοδο hasMoreTokens() για να ελέγχουμε αν υπάρχουν περισσότερα τμήματα προς επιστροφή.

Χρησιμοποιούμε την μέθοδο nextToken() για να μας επιστρέφεται το επόμενο τμήμα.

Η μέθοδος split()

Σε πρόσφατες εκδόσεις της Java η κλάση String παρέχει μια μέθοδο split που μπορεί να χρησιμοποιηθεί για τον διαχωρισμό ενός String. Χρησιμοποιείται ως εξής.

```
String string = "004-034556";
String[] parts = string.split("-");
String part1 = parts[0]; // 004
String part2 = parts[1]; // 034556
```

StringBuilder, StringBuffer

Οι δύο κλάσεις χρησιμοποιούνται να συνθέτουμε κάποια String με διαδοχικές τροποποιήσεις/επεκτάσεις σε κάποιο αρχικό String. Η βασική τους διαφορά βρίσκεται στο γεγονός ότι η StringBuffer παρέχει ασφάλεια νημάτων σε σχέση με την StringBuilder. Η StringBuilder είναι πιο γρήγορη και συνίσταται όταν τα αντικείμενα δεν προσπελαύνονται από διαφορετικά νήματα.

Ο απλός τρόπος για να συνενώσουμε δύο String σε ένα είναι να χρησιμοποιήσουμε το τελεστή συνένωσης (+)

Πχ

String a = “s1” + “,s2”;

String a = a + “,s3”;

Κλπ

Στην περίπτωση αυτή όλα τα ενδιάμεσα String που θα δημιουργηθούν θα υπάρχουν σαν αντικείμενα στην μνήμη σωρού (heap) της java.

Ένας εναλλακτικός τρόπος είναι να χρησιμοποιήσουμε την κλάση StringBuilder (η StringBuffer)

Sb = StringBuilder(“s1”);

Sb.append (“s2”);

Sb.append (“s3”);

Μπορούμε να χρησιμοποιήσουμε την μέθοδο toString() για να εξάγουμε το αποθηκευμένο String.

String s = sb.toString() // s = s1s2s3

Τρόποι αποστολής επιπρόσθετων πληροφοριών μέσω αιτήσεων HTTP

Υπάρχουν δύο βασικοί τρόποι αποστολής επιπρόσθετων πληροφοριών μέσω αιτήσεων HTTP.

1. Μέσω του τμήματος ερώτησης (query string) ενός URL. Αυτό αποτελείται από μια ακολουθία ζευγών name=value χωρισμένα με τον χαρακτήρα &
2. Μέσω των τιμών των πεδίων μιας φόρμας. Όταν η μέθοδος (method) της φόρμας είναι GET τότε οι τιμές των πεδίων της φόρμας τοποθετούνται από τον φυλλομετρητή στο URL. Όταν η μέθοδος της φόρμας είναι POST τότε οι τιμές των παραμέτρων τοποθετούνται στο σώμα της αίτησης HTTP σαν μια σειρά από ζεύξη name=value.

Η διασύνδεση των Servlet παρέχει την δυνατότητα εξαγωγής των παραμέτρων από τις αιτήσεις HTTP μέσω μεθόδων της διασύνδεσης HttpServletRequest.

Παρέχονται οι εξής μέθοδοι:

String getParameter(String)

Παίρνει σαν παράμετρο το όνομα μιας παραμέτρου και επιστρέφει την τιμή της με την μορφή String.

String[] getParameteValues(String)

Παίρνει σαν παράμετρο το όνομα μιας παραμέτρου και επιστρέφει ένα πίνακα String με τις τιμές της παραμέτρου. Χρησιμοποιείται όταν αναμένουμε να έχουν σταλεί πολλές τιμές με το ίδιο όνομα (όπως συμβαίνει όταν χρησιμοποιούμε checkboxes)

Enumeration<String> getParameterNames()

Επιστρέφει μια απαρίθμηση από όλα τα ονόματα παραμέτρων που έχουν σταλεί με την αίτηση HTTP

ServletConfig

Περιγράφει ένα αντικείμενο που δημιουργείται από τον περιέκτη servlet (servlet container) και δίνεται στο servlet κατά την αρχικοποίηση.

Η μέθοδος getServletConfig() ενός servlet χρησιμοποιείται για να επιστρέφει στο servlet αυτό το αντικείμενο.

Ένα αντικείμενο ServletConfig παρέχει τις εξής μεθόδους:

String getInitParameter(String) : Επιστρέφει την τιμή της μιας παραμέτρου αρχικοποίησης το όνομα της οποίας δίνεται σαν παράμετρος στην μέθοδο.

Enumeration getInitParameterNames() : Επιστρέφει ένα Enumeration με όλα τα ονόματα των παραμέτρων αρχικοποίησης

String getServletName() : Επιστρέφει το όνομα που έχει δοθεί στο servlet

ServletContext getServletContext() : Επιστρέφει ένα αντικείμενο τύπου ServletContext που περιέχει πληροφορίες για την εφαρμογή στην οποία ανήκει το Servlet.

Παράδειγμα ορισμού παραμέτρων servlet στο αρχείο web.xml

```
<servlet>
    <servlet-name>HelloWorld</servlet-name>
    <servlet-class>TestServlet</servlet-class>
    <init-param>
        <param-name>myprop</param-name>
        <param-value>value</param-value>
    </init-param>
</servlet>
```

ServletContext

Η κλάση **ServletContext** παρέχει πληροφορίες για το περιβάλλον (context) μιας εφαρμογής.

Η μέθοδος **getServletContext()** που παρέχεται από την διασύνδεση HttpServlet επιστρέφει ένα αντικείμενο ServletContext.

Η κλάση ServletContext παρέχει τις ακόλουθες μεθόδους

String getInitParameter(String) : Επιστρέφει την τιμή μιας παραμέτρου αρχικοποίησης της εφαρμογής με το όνομα που δίνεται.

Enumeration getInitParameterNames() : Επιστρέφει όλα τα ονόματα των παραμέτρων.

Object getAttribute(String) : Επιστρέφει ένα αντικείμενο που έχει τοποθετηθεί στο ServletContext με κάποιο όνομα. Αυτός μπορεί να είναι και ένα τρόπος επικοινωνίας μεταξύ διαφορετικών servlet μιας εφαρμογής

setAttribute(String, Object) : Θέτει κάποιο αντικείμενο στο ServletContext με κάποιο όνομα

Enumeration getAttributeNames() : Επιστρέφει όλα τα ονόματα των αντικειμένων που έχουν τοποθετηθεί στο ServletContext.

Παράδειγμα ορισμού παραμέτρων εφαρμογής στο αρχείο web.xml

```
<web-app>
    <context-param>
        <param-name>Country</param-name>
        <param-value>India</param-value>
    </context-param>
    <context-param>
        <param-name>Age</param-name>
        <param-value>24</param-value>
    </context-param>
</web-app>
```

Μεθοδος setCharacterEncoding()

"Ενα αντικείμενο HttpServletResponse παρέχει τις μεθόδους

setContentType() και setCharacterEncoding για προσδιορισμό του είδους του περιεχομένου που θα επιστραφεί (MIME type) και τον καθορισμό της κωδικοποίησης των χαρακτήρων.

Για παράδειγμα:

response.setContentType("text/html");

response.setCharacterEncoding("utf-8");

Επίσης μπορεί να χρησιμοποιηθεί και το

response.setContentType("text/html;charset=utf-8");

Οι προηγούμενες κλήσεις τοποθετούν τις κατάλληλες πληροφορίες στις επικεφαλίδες HTTP

Επίσης ένα αντικείμενο HttpServletRequest παρέχει και αυτό μια μέθοδο setCharacterEncoding().

Σε αυτή την περίπτωση καθορίζουμε την κωδικοποίηση που χρησιμοποιήθηκε για να σταλούν οι παράμετροι μιας φόρμας.