ΚΑΤΑΣΚΕΥΑΖΟΝΤΑΣ ΚΛΑΣΕΙΣ ΙΙ

Δημήτρης Κ. Ιακωβίδης

Στόχος αυτής της ενότητας είναι η επέκταση των γνώσεων που παρείχε η προηγούμενη ενότητα. Εξηγούνται οι έννοιες της υπερφόρτωσης μεθόδων, της ασφάλειας – ενθυλάκωσης (encapsulation), των μεθόδων μεταλλαγής (mutator) και προσπέλασης (accessor) και της τυπικής μεθόδου equals για τη σύγκριση αντικειμένων της ίδιας κλάσης.

5.1 Μια νέα κλάση

Έστω ότι επιθυμούμε να κατασκευάσουμε μια κλάση που να αναπαριστά μια ημερομηνία (Date). Σκεπτόμενοι όπως και στο παράδειγμα της κλάσης Complex, που περιγράφηκε στην προηγούμενη ενότητα, θα πρέπει να εντοπίσουμε τι χαρακτηρίζει μια ημερομηνία, ή με άλλα λόγια, σε τι διαφέρει μια ημερομηνία από μια άλλη. Τις χαρακτηριστικές ιδιότητες μιας ημερομηνίας τις δηλώνουμε ως μεταβλητές υπόστασης της κλάσης μας (της ημερομηνίας). Βάσει του παραδείγματος της Complex μπορούμε να κατασκευάσουμε την κλάση Date όπως περιγράφει το Σχ.1.

```
1    class Date
2    {
3          // Instance variables
4          int day, month, year;
5
6          // Constructor
7          Date(int day, int month, int year)
8          {
9                this.day = day;
10               this.month = month;
11               this.year = year;
12         }
13
14         // Standard toString method
15         public String toString()
16         {
17               String s;
18               s = day + "/" + month + "/" + year;
19               return s;
20         }
21
22         public static void main(String args[])
23         {
24               Date y = new Date(12, 3, 2014);
```

```
25                  System.out.println(y);
26          }
27  }
```

Σχήμα 1. Μια κλάση που αναπαριστά ημερομηνίες.

5.2 Κλήση μεθόδων δημιουργών από μεθόδους δημιουργούς

Η Java μας δίνει τη δυνατότητα να έχουμε πολλές μεθόδους που έχουν το ίδιο όνομα και διαφορετικές παραμέτρους (λαμβάνουν διαφορετικά ορίσματα). Όμως όλες αυτές οι μέθοδοι πρέπει να επιστρέφουν τιμές του ίδιου τύπου. Αυτό χαρακτηρίζεται ως **υπερφόρτωση μεθόδων (overloading)**. Αναλόγως, και οι μέθοδοι δημιουργοί μπορούν να υπερφορτωθούν.

Όταν έχουμε πολλές (υπερφορτωμένες) μεθόδους δημιουργούς, συχνά δημιουργείται η ανάγκη να καλούμε μέσα από εκείνες που έχουν λιγότερες παραμέτρους εκείνες που έχουν περισσότερες παραμέτρους. Σε αυτή την περίπτωση η κλήση της μεθόδου δημιουργού δε γίνεται με το όνομά της αλλά με τη χρήση της λέξης this. Αυτή τη φορά όμως χρησιμοποιώντας παρενθέσεις αντί για τελεία μετά τη λέξη this, δηλαδή καλείται ως this(ορίσματα). Παράδειγμα εικονίζεται στο Σχ. 2.

```
1   class Date
2   {
3       // Instance variables
4       int day, month, year;
5
6       // Constructor with three arguments
7       Date(int day, int month, int year)
8       {
9           this.day = day;
10          this.month = month;
11          this.year = year;
12      }
13
14      // Overloaded constructor with one argument
15      // The argument represents the year
16      Date(int y)
17      {
18          this(1,1,y);
19      }
20
21      // Standard toString method
22      public String toString()
23      {
```

```
24                String s;
25                s = day + "/" + month + "/" + year;
26                return s;
27          }
28
29          public static void main(String args[])
30          {
31                // Calling of three-argument constructor
32                Date y = new Date(12, 3, 2014);
33                System.out.println(y);
34
35                // Calling of one-argument constructor
36                Date z = new Date(2014);
37                System.out.println(z);
38
39                /*
40                The problem of this class is that it can
41                accept even logically incorrect values
42                */
43                y.day = 32;
44          }
    }
```

Σχήμα 2. Η κλάση Date με υπερφορτωμένες μεθόδους δημιουργούς.

5.3 Ασφάλεια και ενθυλάκωση

Ένα πρόβλημα που έχει η κλάση Date, όπως είναι τώρα, είναι ότι δέχεται ημερομηνίες χωρίς έλεγχο, π.χ. μπορεί να δεχθεί ακόμα και την 32/13/-1920 ως ημερομηνία. Η Java μας δίδει τη δυνατότητα να προστατεύουμε τις κλάσεις μας από τέτοια λογικά σφάλματα.

Μπορούμε να «κρύψουμε» (hide) τις μεταβλητές υπόστασης ώστε να μη μπορεί κάποιος που είναι έξω από την κλάση μας (σε κάποια άλλη κλάση) να τροποποιήσει αυθαίρετα τις τιμές (όπως συμβαίνει στις γραμμές 40-41 του κώδικα του Σχ. 2). Αυτό μπορεί να γίνει χρησιμοποιώντας την λέξη **private (ιδιωτικός)** πριν από τη δήλωση της μεταβλητής που επιθυμούμε να κρύψουμε. Αυτό ονομάζεται **ενθυλάκωση (encapsulation)**. Αντίθετο του private είναι το **public (δημόσιος)** το οποίο επιτρέπει να προσπελαύνουμε την αντίστοιχη μεταβλητή από οποιαδήποτε εξωτερική κλάση, όπου και αν βρίσκεται αυτή (ακόμα και σε διαφορετικό πακέτο). Αν δεν υπάρχει ούτε public, ούτε private, εννοείται η προεπιλεγμένη (**default**) προσπέλαση η οποία δεν επιτρέπει την προσπέλαση των μεταβλητών μας από κλάση που ανήκει σε άλλο πακέτο. **Οι λέξεις public και private χρησιμοποιούνται με τον ίδιο τρόπο και για μεθόδους.**

Για να μπορούμε να τροποποιούμε τις τιμές των μεταβλητών υπόστασης που είναι private υπό έλεγχο κατασκευάζουμε **μεθόδους μεταλλαγής (mutator methods)**. Οι μέθοδοι αυτές είθισται (κατά σύμβαση) να ξεκινούν με το πρόθεμα **set** στο όνομά τους και είναι void. Ως όρισμα λαμβάνουν την τιμή που θα επιθυμούσαμε να εισάγουμε στην αντίστοιχη μεταβλητή υπόστασης. Ως παράδειγμα δίνονται οι μέθοδοι με το πρόθεμα set, στη βελτιωμένη κλάση Date του Σχ. 3.

Αντίστοιχα, επειδή δεν υπάρχει άλλος τρόπος για να προσπελάσουμε (π.χ. να διαβάσουμε) τις τιμές των μεταβλητών που είναι δηλωμένες ως private θα πρέπει να κατασκευάσουμε κατάλληλες μεθόδους που να επιστρέφουν τις τιμές των μεταβλητών αυτών. Οι μέθοδοι αυτές λέγονται **μέθοδοι προσπέλασης (accessor methods)**, και είθισται το όνομά τους να ξεκινά με το πρόθεμα **get**. Ως παράδειγμα δίνονται οι μέθοδοι με το πρόθεμα get, στη βελτιωμένη κλάση Date του Σχ. 3.

Στο παράδειγμα του Σχ. 3 οι μεταβλητές υπόστασης έχουν αρχικοποιηθεί σε κάποιες τιμές που αντιστοιχούν σε λογική προεπιλεγμένη ημερομηνία (1/1/2000). Διαφορετικά θα ελάμβαναν μηδενικές τιμές. Επίσης η μέθοδος δημιουργός καλεί τη μέθοδο μεταλλαγής για να περάσει και αυτή υπό περιορισμούς τις τιμές των μεταβλητών υπόστασης.

Για να μπορέσει να γίνει καλύτερα αντιληπτό το παράδειγμα αυτό είναι καλύτερα να χρησιμοποιηθεί η Date μέσα από μια εξωτερική κλάση όπως είναι η TestDate του Σχ. 4.

```
1    public class Date
2    {
3          private int day = 1, month = 1, year = 2000;
4
5          // Constructor using mutator method setDate
6          public Date(int day, int month, int year)
7          {
8                setDate(day, month, year);
9          }
10
11         // Constructor with one argument
12         public Date(int y)
13         {
14               this(1,1,y);
15         }
16
17         // Mutator methods
18         public void setDay(int d)
```

```
19          {
20                if ( (d > 0) && (d <= 31))
21                {
22                      day = d;
23                }
24          }
25
26          public void setMonth(int m)
27          {
28                if ( (m > 0) && (m <= 12))
29                {
30                      month = m;
31                }
32          }
33
34          public void setYear(int y)
35          {
36                if ( (y > 0) && (y <= 3000))
37                {
38                      year = y;
39                }
40          }
41
42          public void setDate(int d, int m, int y)
43          {
44                setDay(d);
45                setMonth(m);
46                setYear(y);
47          }
48
49
50          // Accessor methods
51          public int getDay()
52          {
53                return day;
54          }
55
56          public int getMonth()
57          {
58                return month;
59          }
60
61          public int getYear()
62          {
63                return year;
64          }
65    }
```

Σχήμα 3. Μια βελτιωμένη κλάση Date. Στην κλάση αυτή δεν έχει προστεθεί main γιατί μέσα από τη main της ίδιας κλάσης επιτρέπεται η προσπέλαση των private μεταβλητών υπόστασης.

```
1     class TestDate
2     {
3           public static void main(String args[])
```

```
4           {
5                   // Calling mutator methods
6                   Date x = new Date(1, 2, 2014);
7                   x.setDay(32); // not accepted
8                   x.setMonth(12);
9                   x.setYear(2015);
10
11                  // Calling accessor methods
12                  System.out.println("The date is " + x.getDay()
13                  + "/" + x.getMonth() + "/" + x.getYear());
14          }
15  }
```

Σχήμα 4. Δοκιμαστική κλάση για τη χρήση της Date.

5.4 Η τυπική μέθοδος equals

Συχνά χρειάζεται να συγκρίνουμε δύο αντικείμενα της ίδιας κλάσης, π.χ. δύο ημερομηνίες, αν είναι ίσα ως προς το περιεχόμενό τους (equal). Η έννοια της ισότητας ορίζεται από τον προγραμματιστή που κατασκευάζει την κλάση, μέσω μιας τυπικής μεθόδου που αναμένεται να έχει η κλάση. Η κλάση αυτή λέγεται equals και πρέπει να δηλώνεται πάντα με τον ίδιο τρόπο, όπως εικονίζεται στο παράδειγμα του Σχ. 5. Η μέθοδος του Σχ. 5 τοποθετείται εντός του κώδικά του Σχ. 4.

Η μέθοδος equals λαμβάνει πάντα ως όρισμα ένα αντικείμενο της κλάσης στην οποία ανήκει, ενώ επιστρέφει πάντα μια τιμή τύπου boolean. Η τιμή που επιστρέφει είναι true αν τα συγκρινόμενα αντικείμενα είναι ίσα, ή false διαφορετικά.

Προσοχή! Αν έχουμε δύο αντικείμενα x, y μιας κλάσης και προσπαθήσουμε να τα συγκρίνουμε με τον τελεστή ==, και όχι με τη χρήση της equals, τότε τα αντικείμενα αυτά θα συγκριθούν ως προς τη θέση τους στη μνήμη. Το όνομα ενός οποιουδήποτε αντικειμένου ισοδυναμεί με μια **αναφορά (reference)** στη μνήμη. Αν θέσουμε x = y, τότε το αντικείμενο x και το αντικείμενο y αντιστοιχούν στην ίδια αναφορά τη μνήμη, δηλαδή όταν αλλάζουμε το ένα, αλλάζει και το άλλο. Σε αυτή την περίπτωση η χρήση του τελεστή == για τη σύγκριση των δύο αντικειμένων οδηγεί σε μια boolean τιμή true. Διαφορετικά θα οδηγούσε σε μια boolean τιμή false.

```
1    class Date
2    {
3          // Standard equals method
4          public boolean equals(Date other)
5          {
6                boolean b = false;
7                if ( (day == other.day) && (month ==
8                other.month) && (year == other.year) )
9                {
10                     b = true;
11               }
12               return b;
13         }
14
15         // Main method to test equals method
16         public static void main(String args[])
17         {
18               Date x = new Date(1, 2, 2014);
19               Date y = new Date(2, 3, 2014);
20
21               boolean b = x.equals(y);
22
23               System.out.println((b)?"equal":"not equal");
24         }
25   }
```

Σχήμα 5. Η μέθοδος equals της κλάσης Date. Ο κώδικας αυτός συνεχίζει τη βελτιωμένη Date του Σχ. 4.

Ασκήσεις

1. Βάσει του παραδείγματος της κλάσης Date, να κατασκευάσετε μια νέα κλάση Time για την αναπαράσταση της ώρας, θεωρώντας ότι η ώρα αποτελείται από ώρες, λεπτά και δευτερόλεπτα. Η κλάση να περιέχει κατάλληλες μεθόδους δημιουργούς, και toString.

2. Να προσθέσετε μια μέθοδο δημιουργό με ένα μόνο όρισμα, την ώρα.

3. Να προφυλάξετε την Time από την εισαγωγή παράλογων τιμών στην ώρα, ασφαλίζοντας τις μεταβλητές και εισάγοντας μεθόδους μεταλλαγής και προσπέλασης.

4. Να προσθέσετε στην Time μια μέθοδο για τη σύγκριση δύο ωρών.