3 Αλληλεπίδραση Αντικειμένων

Πώς συνεργάζονται
τα αντικείμενα που δημιουργούμε

Αφαίρεση και Αρθρωσιμότητα

- **Αφαίρεση** (abstraction) είναι η δυνατότητα να αγνοούμε τις λεπτομέρειες και να εστιάζουμε την προσοχή μας σε ένα ανώτερο επίπεδο κατανόησης του προβλήματος.
- **Αρθρωσιμότητα** (modularization) είναι η διαδικασία κατάτμησης του όλου σε καλώς ορισμένα τμήματα, που μπορούν να αναπτυχθούν και να ελεγχθούν ξεχωριστά, και αλληλεπιδρούν με σαφώς ορισμένους τρόπους.
- Παράδειγμα: κατασκευή αυτοκινήτου
- Με άλλα λόγια: *Διαίρει και Βασίλευε*

68

Ψηφιακό Ρολόι

11:03

- Ώρες και λεπτά ωω:λλ από 00:00 έως 23:59

69

Κατάτμηση του Ψηφιακού Ρολογιού

11:03 Μια οθόνη τεσσάρων ψηφίων;

Ή δύο οθόνες δύο ψηφίων; 11 03

Στον αντικειμενοστρεφή προγραμματισμό,
τα τμήματα είναι αντικείμενα.
Πολλά αντικείμενα μπορεί να συνδυάζονται σε ένα.

70

Κλάση Διψήφιας Οθόνης

Πηγαίος κώδικας clock-display-gigas

```
/**
 * Διψήφια ψηφιακή οθόνη
 */
public class NumberDisplay
{
    // δείχνει από 00 ως limit-1
    private int limit;
    // τρέχουσα τιμή
    private int value;
    ...
}
```

71

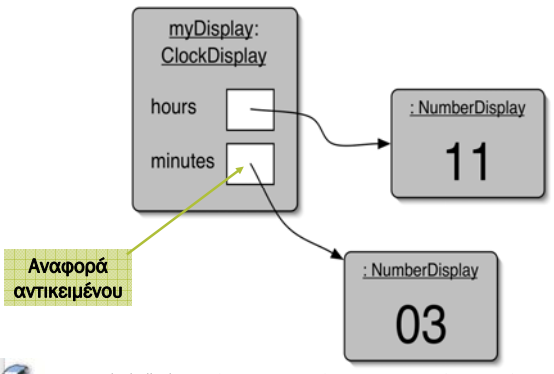
Κλάση Οθόνης Ρολογιού

```
/**
 * Η οθόνη του ψηφιακού ρολογιού
 * αποτελείται από δύο διψήφιες οθόνες
 * που προβάλλουν την ώρα και τα λεπτά
 */
public class ClockDisplay
{
    private NumberDisplay hours;
    private NumberDisplay minutes;
    ...
}
```

Η κλάση που ορίσαμε προηγουμένως

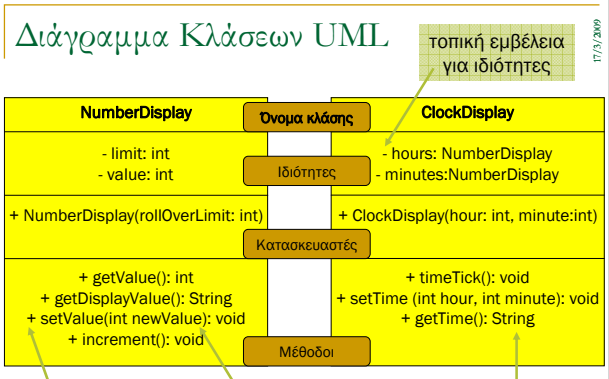
72

Διάγραμμα Αντικειμένων



Project: **clock-display**: εκτέλεση και παρατήρηση συμπεριφοράς και κατάστασης

Διάγραμμα Κλάσεων UML



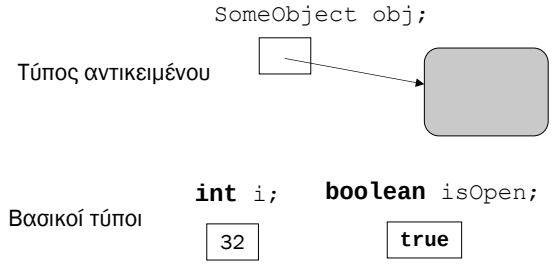
Διαγράμματα

- **Διάγραμμα κλάσεων** (class diagram)
  - Δείχνει τις κλάσεις μιας εφαρμογής και τις συσχετίσεις μεταξύ τους
  - Δίνει πληροφορίες για τον πηγαίο κώδικα
  - Στατική όψη του προγράμματος
- **Διάγραμμα αντικειμένων** (object diagram)
  - Δείχνει αντικείμενα και τις συσχετίσεις του κατά μια συγκεκριμένη στιγμή της εκτέλεσης του προγράμματος
  - Δίνει πληροφορίες για τα αντικείμενα κατά το χρόνο εκτέλεσης
  - Δυναμική όψη του προγράμματος

Είδη Τύπων Δεδομένων

- Η Java έχει δύο είδη τύπων δεδομένων:
  - **βασικοί τύποι** (primitive data types)
  - **τύποι αντικειμένων** (object data types)
- Οι βασικοί τύποι, όπως **`int`**, **`boolean`**, είναι προκαθορισμένοι από τη γλώσσα.
  - Η αναπαράσταση και η συμπεριφορά τους δεν αλλάζει.
  - Πραγματικοί: **`double`** Χαρακτήρες: **`char`**
- Οι τύποι αντικειμένων προέρχονται από κλάσεις.
  - Κάποιοι ορίζονται από το σύστημα, πχ `String`
  - Ο προγραμματιστής μπορεί να ορίσει κι άλλους, πχ `NumberDisplay`, ή να τροποποιήσει κάποιους από τους υπάρχοντες.

Παραδείγματα Ειδών Τύπων Δεδομένων



Project: **clock-display**: επιθεώρηση κώδικα

Υλοποίηση ClockDisplay

```
public class NumberDisplay
{
    private int limit; // όριο που ποτέ δεν φτάνει
    private int value; // τρέχουσα τιμή οθόνης

    public NumberDisplay(int rollOverLimit) { //κατασκευαστή
        limit = rollOverLimit; value = 0; }

    public int getValue() { return value; } //ανάγνωση τιμής

    public String getDisplayValue() { // ανάγνωση τιμής
        if (value < 10) return "0" + value;
        else            return ""  + value;
    }

    public void setValue(int newValue) { // θέτει νέα τιμή
        if ((newValue >= 0) && (newValue < limit))
            value = newValue;
    }

    public void increment() { // προχωράει την τιμή
        value = (value + 1) % limit;    }
}
```

Λογική σύζευξη ΚΑΙ

Υπόλοιπο ακέραιας διαίρεσης

Ιωάννης Γαβιώτης
gaviotis@aegean.gr
http://www.syros.aegean.gr/users/gaviotis/tmp2

Υλοποίηση ClockDisplay

```
public class ClockDisplay{
    private NumberDisplay hours;
    private NumberDisplay minutes;

    public ClockDisplay(int hour, int minute) {
        hours = new NumberDisplay(24);
        minutes = new NumberDisplay(60);
        setTime(hour, minute);
    }

    public void timeTick() {
        minutes.increment();
        if (minutes.getValue() == 0) hours.increment();
    }

    public void setTime(int hour, int minute) {
        hours.setValue(hour); minutes.setValue(minute);
    }

    public String getTime() {
        return hours.getDisplayValue() + ":" +
               minutes.getDisplayValue();
    }
}
```

Αναφορά αντικειμένου

Κλήση κατασκευαστή

79

Τελεστές και Ορίσματα

- **Τελεστής** (operator) είναι το σύμβολο μιας πράξης.
- **Ορίσματα** (operands) είναι οι παραστάσεις πάνω στις οποίες εφαρμόζονται οι τελεστές.
  - Πχ στην παράσταση `4 + 5` τα ορίσματα είναι οι αριθμοί `4, 5` και ο τελεστής είναι το `+`
- Συνηθισμένοι τελεστές είναι οι **αριθμητικοί**:
  `+  -  *  /`
- Οι **λογικοί τελεστές** (logic operators) στη Java είναι:

`&&`	σύζευξη	λογικό ΚΑΙ
`||`	διάζευξη	λογικό Ή
`!`	άρνηση	λογικό ΟΧΙ

80

Υπερφόρτωση Τελεστών

- Ένας τελεστής είναι **υπερφορτωμένος** (overloaded operator), όταν εκτελεί διαφορετικές πράξεις ανάλογα με τον τύπο των ορισμάτων που δέχεται.
- Ο τελεστής + εκτελεί πρόσθεση όταν εφαρμόζεται σε αριθμούς και συνένωση όταν εφαρμόζεται σε συμβολοσειρές.
  - `3+2` → `5`, ενώ
  - `"Καλή" + "μέρα"` → `"Καλήμέρα"`
- Ο τελεστής / εκτελεί **ακέραια διαίρεση**, όταν εφαρμόζεται σε ακεραίους.
  - `5 / 2` → `2` ενώ
  - `5.0 / 2` → `2.5` (μετατρέπει και το 2 σε πραγματικό 2.0)

81

Ο Τελεστής Υπόλοιπο

- Ο τελεστής `%` επιστρέφει το **υπόλοιπο** (modulo)της ακέραιης διαίρεσης.
  - `10 % 4` → `2`
  - `101 % 50` → `1`
  - `10 % 5` → `0`
- Τα ορίσματα του `%` πρέπει να είναι ακέραιοι.
- Αν κ, λ είναι ακέραιοι, τι τιμές μπορεί να πάρει η παράσταση κ `%` λ ;

82

Δημιουργία Στιγμιότυπων

- Μέχρι τώρα δημιουργούσαμε αντικείμενα με δεξί κλικ πάνω στην κλάση τους και επιλέγοντας *new ...* από το αναδυόμενο μενού.
- Με κώδικα αυτό γίνεται καλώντας το κατασκευαστή μαζί με τη δεσμευμένη λέξη **new** σε μια εντολή ανάθεσης.

```
hours = new NumberDisplay( 24 );
```

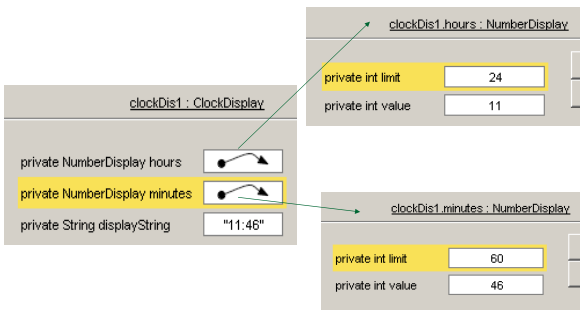
Μεταβλητή τύπου αντικειμένου

Όνομα κατασκευαστή

Πραγματική παράμετρος

83

Η Κατάσταση των Αντικειμένων



84

Πολλαπλοί Κατασκευαστές

- Μέχρι τώρα κάθε κλάση έχει έναν κατασκευαστή που τον χρησιμοποιούμε για να δημιουργούμε στιγμιότυπα.
- Μπορούμε να έχουμε πολλούς κατασκευαστές
  - Όλοι θα έχουν το ίδιο όνομα – το όνομα της κλάσης
  - Θα έχουν διαφορετικές παραμέτρους για να ξεχωρίζουν κατά την κλήση τους.
- Αυτό ονομάζεται **υπερφόρτωση** (overloading).
  - Πού ξανασυναντήσαμε κάτι ανάλογο;

Παράδειγμα Υπερφόρτωσης Κατασκευαστών

```
public class ClockDisplay {

  private NumberDisplay hours;
  private NumberDisplay minutes;

  /** δημιουργεί ρολόι που έχει ώρα 00:00 */
  public ClockDisplay() {
    hours = new NumberDisplay(24);
    minutes = new NumberDisplay(60);
  }

  /** δημιουργεί ρολόι που έχει συγκεκριμένη
    * ώρα hour και λεπτά minute */
  public ClockDisplay(int hour, int minute) {
    hours = new NumberDisplay(24);
    minutes = new NumberDisplay(60);
    setClock(hour, minute);
  }
}
```


Κλήση Μεθόδου της Ίδιας Κλάσης

```
public ClockDisplay(int hour, int minute)
{
  hours = new NumberDisplay(24);
  minutes = new NumberDisplay(60);
  setTime(hour, minute);
}
...
public void setTime(int hour, int minute)
{
  hours.setValue(hour);
  minutes.setValue(minute);
}
```

κλήση μεθόδου

Σύνταξη: *όνομαΜεθόδου* **(** *λίσταΠαραμέτρων* **);**

Κλήση Μεθόδου Άλλης Κλάσης

Σύνταξη:
αντικείμενο **.** *όνομαΜεθόδου* **(** *λίσταΠαραμέτρων* **);**

```
public void tickTick()
{
  minutes.increment();
  if (minutes.getValue() == 0) {
    hours.increment();
  }
}
```

κλήση μεθόδου

```
public class NumberDisplay {
  ...
  public void increment() { ... }
}
```

ορισμός σε άλλη κλάση

Να η Αφαίρεση!

- Ξεκινήσαμε να φτιάξουμε ένα ψηφιακό ρολόι.
- Διαπιστώσαμε ότι μπορεί να αποτελείται από δύο διψήφιες οθόνες για τις ώρες και τα λεπτά.
- Φτιάξαμε κώδικα κλάσης για μια «γενική» οθόνη.
  - Μετράει μέχρι όπου της πει ο κατασκευαστής της.
- Το ρολόι είναι σύνθεση δύο οθόνων.
  - Φτιάξαμε δύο στιγμιότυπα και καλέσαμε μεθόδους τους.

Δεν χρειάζεται να γνωρίζουμε πώς λειτουργούν οι μέθοδοι για να τις καλέσουμε!

Η δεσμευμένη λέξη **this**

```
public class Auto {

  //μεταβλητές στιγμιότυπου
  private int kyvika ;
  private String montelo ;

  public Auto (int kyvika, String montelo) {
    this.kyvika = kyvika ;
    this.montelo = montelo ;
  }
}
```

Στο σώμα του κατασκευαστή υπάρχουν δύο μεταβλητές με το όνομα `kyvika`: η τυπική παράμετρος και η μεταβλητή στιγμιότυπου (**υπερφόρτωση ονόματος**).

Ιωάννης Γαβιώτης
gaviotis@aegean.gr
http://www.syros.aegean.gr/users/gaviotis/tmp2

Αποσφαλμάτωση

17/3/2009

- Σπάνια γράφεται λογισμικό δίχως **σφάλματα** (bugs).
- Υπάρχουν δύο είδη σφαλμάτων:
  - **Συντακτικά**, που τα συλλαμβάνει ο μεταγλωττιστής
    ```
    string onoma;
    onoma = 'αννα";
    ```
  - **Λογικά**, που τα βρίσκει (;) ο προγραμματιστής
    ```
    int a = 0; int b = 10 / a;
    ```
- Εντοπίζουμε σφάλματα:
  - Διαβάζοντας τον κώδικα (& κατανοώντας πώς λειτουργεί)
  - Παρακολουθώντας την εκτέλεση του προγράμματος με χρήση του debugger

91

Τεχνικές Αποσφαλμάτωσης

17/3/2009

- **Σημείο διακοπής** (breakpoint): η εκτέλεση διακόπτεται όταν φτάσει στο σημείο αυτό.
  - Τότε μπορούμε να εξετάσουμε τις τιμές των μεταβλητών.
- Εκτέλεση κώδικα γραμμή – προς – γραμμή
- Κατά την κλήση μιας μεθόδου, έχουμε δύο επιλογές:
  - Να παρακολουθήσουμε την εκτέλεση μέσα στο σώμα της μεθόδου (step into)
  - Να την αφήσουμε να εκτελεστεί και να προχωρήσουμε στην επόμενη εντολή (step over)

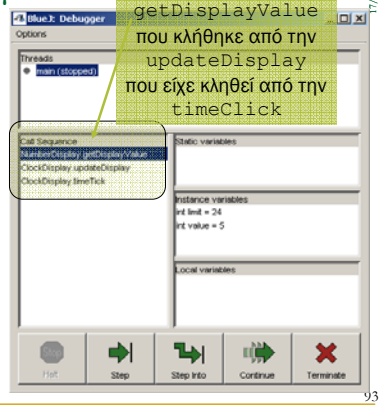
92

Ακολουθία Κλήσεων

17/3/2009

- Κατά την εκτέλεση του προγράμματος μπορούμε να δούμε τις μεθόδους που εκτελούνται (call sequence).
- Έτσι μπορούμε να δούμε ποιος κάλεσε ποιον και ποιες μέθοδοι είναι 'ενεργές'.

Εκτελείται η `getDisplayValue` που κλήθηκε από την `updateDisplay` που είχε κληθεί από την `timeClick`



93

Όροι Ενότητας 3

17/3/2009

- Αφαίρεση
- Αρθρωσιμότητα
- Διάγραμμα κλάσεων
- Διάγραμμα αντικειμένων
- Βασικοί τύποι δεδομένων
- Τύποι δεδομένων αντικειμένων
- Τελεστές
- Ορίσματα
- Λογικοί τελεστές
- Υπερφορτωμένος τελεστής
- Υπόλοιπο ακέραιης διαίρεσης
- Δημιουργία αντικειμένων
- Κλήση μεθόδου
- Υπερφόρτωση ονόματος μεθόδου
- Αποσφαλμάτωση
- Σημείο διακοπής

Τεκμήρια ενότητας 3

`& &` `| |` `!`
`%`
`new`
`.`
`( )`
`this`

94