Αντικειμενοστρεφής Προγραμματισμός

Παναγιώτης Αδαμίδης
adamidis@it.teithe.gr

Εξαιρέσεις (Exceptions)

Χειρισμός Λαθών

- Κύριο πρόβλημα είναι η εμπιστοσύνη στο ότι ο προγραμματιστής ακολουθεί κάποιες γενικές αρχές οι οποίες όμως δεν επιβάλλονται από την γλώσσα.
- Στιβαρό (robust) πρόγραμμα είναι αυτό που συνεχίζει να λειτουργεί με προκαθορισμένο τρόπο, ανεκτά ακόμη και με την παρουσία λαθών
- Ο μηχανισμός χειρισμού εξαιρέσεων ενσωματώνει τον χειρισμό λαθών στην γλώσσα προγραμματισμού ή ακόμη και στο λειτουργικό σύστημα.

Χειρισμός Εξαιρέσεων

- Μία εξαίρεση είναι ένα αντικείμενο το οποίο «ρίχνεται» από το σημείο που γίνεται το λάθος και το οποίο μπορεί να «πιάσει» ο κατάλληλος χειριστής εξαιρέσεων, ο οποίος σχεδιάστηκε για να χειρίζεται τον συγκεκριμένο τύπο λάθους. Είναι σαν να υπάρχει ένας παράλληλος δρόμος εκτέλεσης ο οποίος ακολουθείται όταν συμβαίνει κάποιο λάθος.
- Μία εξαίρεση δεν μπορεί να αγνοηθεί, όπως μία τιμή λάθους που επιστρέφει μία συνάρτηση ή μία σημαία που χρησιμοποιείται για αυτό το σκοπό.
- Οι εξαιρέσεις παρέχουν έναν αξιόπιστο τρόπο με τον οποίο μπορούμε να ξεπεράσουμε προβλήματα εκτέλεσης και να επαναφέρουμε την εκτέλεση του προγράμματος αφού επιληφθούμε των προβλημάτων.
- Ο μηχανισμός χειρισμού εξαιρέσεων της Java είναι απαραίτητο να χρησιμοποιηθεί κατά την συγγραφή προγραμμάτων.

Καθορισμοί

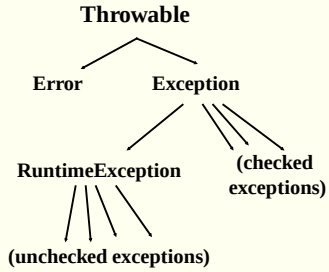
- Μία διαδικασία η οποία μπορεί να τελειώσει με εξαίρεση έχει μία πρόταση `throws`.

```
public static int fact (int n) throws NonPositiveException
// EFFECTS: If n is non-positive throws NonPositiveException,
// else returns the factorial of n
```

- Μία μέθοδος μπορεί να «ρίξει» περισσότερες από μία εξαιρέσεις:

```
public static int search (int[ ] a, int x) throws
                NullPointerException,  NotFoundException
// REQUIRES: a is sorted
// EFFECTS: if a is null throws NullPointerException, else if x
// is not in a throws NotFoundException, else returns k such
// that   a[k]=x
```

Τύποι Εξαιρέσεων



Διαφορές χρήσης

- Εάν μία διαδικασία μπορεί να «ρίξει» μία ελεγχόμενη εξαίρεση, η Java απαιτεί να αναφέρεται η εξαίρεση στη δήλωσή της. Αλλιώς θα υπάρξει λάθος κατά την μεταγλώττιση. Οι μη ελεγχόμενες εξαιρέσεις δεν χρειάζεται να αναφερθούν στην κεφαλίδα της δήλωσης της μεθόδου.
- Εάν ο κώδικας καλεί μία μέθοδο η οποία μπορεί να «ρίξει» μία ελεγχόμενη εξαίρεση, η Java απαιτεί να χειριστεί την εξαίρεση, αλλιώς θα υπάρξει λάθος κατά την μεταγλώττιση. Οι μη ελεγχόμενες εξαιρέσεις δεν χρειάζεται να χειριστούν στον κώδικα που τις καλεί.
- Θα παρεκκλίνουμε από τους κανόνες της Java *προτείνοντας* την αναφορά όλων των πιθανών εξαιρέσεων στην κεφαλίδα της δήλωσης μιας μεθόδου.

Ορισμός Τύπων Εξαιρέσεων

- Η δήλωση μίας εξαίρεσης δείχνει εάν είναι ελεγχόμενη ή μη-ελεγχόμενη
- Παράδειγμα δήλωσης:

```
public class NewKindOfException extends Exception {
        public NewKindOfException() {super();}
        public NewKindOfException(String s) {super(s);}
}
```

- Παραδείγματα χρήσης:

```
Exception e1 = new NewKindOfException
                                ("reason for the except");
Exception e2 = new NewKindOfException();
```

- Απόκτηση της τιμής του String:

```
String s = e1.toString();
s --> "NewKindOfException: reason for the except"
```

Παράγοντας μία εξαίρεση

- Μία μέθοδος μπορεί να τελειώσει παράγοντας (throwing) μία εξαίρεση με την χρήση της εντολής `throw`
- Παράδειγμα:

```
if (n <= 0)
  throw new NonPositiveException("Num.fact");
```

Χειρισμός Εξαιρέσεων

- Όταν μία μέθοδος τελειώνει παράγοντας μία εξαίρεση ο έλεγχος μεταφέρεται στον κώδικα χειρισμού της συγκεκριμένης εξαίρεσης
- Δύο τρόποι χειρισμού: εντολή `try` ή διάδοση της εξαίρεσης
- Παράδειγμα:

```
if (n <= 0)
  throw new NonPositiveException("Num.fact");
```

Εξαιρέσεις - Παράδειγμα-1

```
class ExcDemo1 {
  public static void main(String[] args) {
        int nums[] = new int[4];
        try {
           System.out.println("Before exception is generated");
           nums[7] = 10;
           System.out.println("This won't be displayed");
        }
        catch (ArrayIndexOutOfBoundsException exc) {
           System.out.println("Index out-of-bounds");
        }
        System.out.println("After catch statement");
  }
}
```

Εξαιρέσεις - Παράδειγμα-2

```
class ExcTest {
  static void genException() {
        int nums[] = new int[4];
        System.out.println("Before exception is generated");
        nums[7] = 10;
        System.out.println("This won't be displayed");
  }
}
class ExcDemo2 {
  public static void main(String[] args) {
        try { ExcTest.genException(); }
        catch (ArrayIndexOutOfBoundsException exc) {
           System.out.println("Index out-of-bounds");
        }
        System.out.println("After catch statement");
  }
}
```

Εξαιρέσεις - Παράδειγμα-3

```
class ExcDemo3 {
  public static void main(String[] args) {
        int numer[] = {4, 8, 16, 32, 64, 128};
        int denom[] = {2, 0,  4,  4,  0,  8};
        for (int i=0; i<numer.length; i++) {
           try {
              System.out.println(numer[i]+" /"+ denom[i]+
                                 " is "+ numer[i]/denom[i]);
           }
           catch (ArithmeticException exc) {
              System.out.println("Can't divide by zero!");
           }
        }
  }
}
```

Εξαιρέσεις - Παράδειγμα-4

```
class ExcDemo4 {
  public static void main(String[] args) {
        int numer[] = {4, 8, 16, 32, 64, 128, 256, 512};
        int denom[] = {2, 0,  4,  4,  0,  8};
        for (int i=0; i<numer.length; i++) {
          try {
            System.out.println(numer[i]+" /"+ denom[i]+
                               " is "+numer[i]/denom[i]);
          }
          catch (ArithmeticException exc) {
            System.out.println("Can't divide by zero!");
          }
          catch (ArrayIndexOutOfBoundsException exc) {
            System.out.println("No matching element found!");
          }
        }
  }
}
```

Εξαιρέσεις - Παράδειγμα-5

```
class ExcDemo4 {
  public static void main(String[] args) {
        int numer[] = {4, 8, 16, 32, 64, 128, 256, 512};
        int denom[] = {2, 0,  4,  4,  0,  8};
        for (int i=0; i<numer.length; i++) {
          try {
            System.out.println(numer[i]+" /"+ denom[i]+
                               " is "+numer[i]/denom[i]);
          }
          catch (ArithmeticException exc) {
            System.out.println("Can't divide by zero!");
          }
          catch (ArrayIndexOutOfBoundsException exc) {
            System.out.println("No matching element found!");
          }
         catch (Exception exc) {  System.out.println("Friday!");  }
         catch (Throwable exc) {  System.out.println("Tuesday!");  }
        }
  }
}
```

Παράδειγμα: Πρόκληση εξαίρεσης

```
class ThrowDemo {
  public static void main(String[] args) {
    try {
      System.out.println("Before throw.");
      throw new ArithmeticException();
    }
    catch (ArithmeticException exc) {
          System.out.println ("Exception caught.");
    }
    System.out.println("After try/catch block. ");
  }
}
```