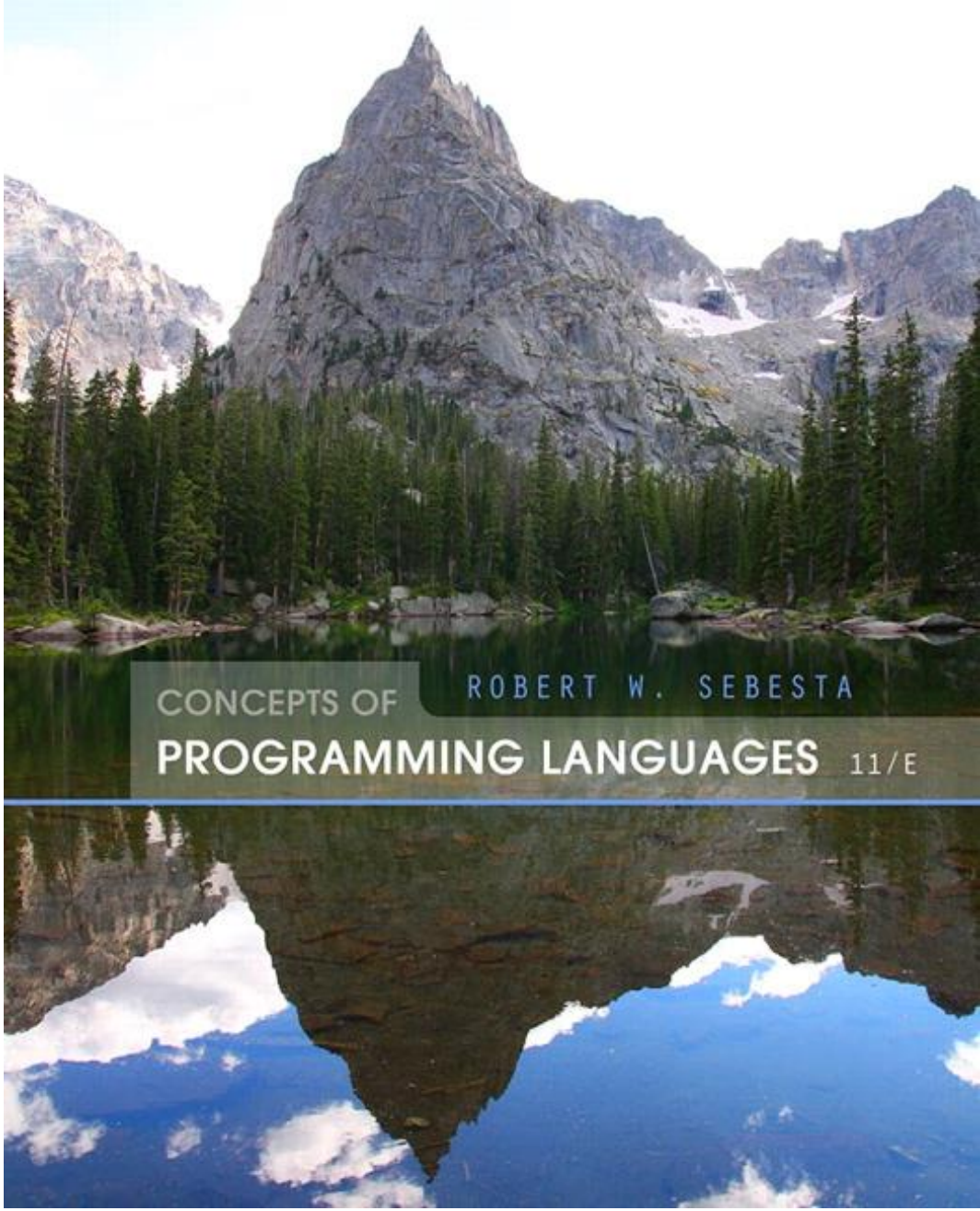
Κεφάλαιο 14

Χειρισμός εξαιρέσεων και χειρισμός συμβάντων



ISBN 0-321-49362-1

Περιεχόμενα κεφαλαίου 14

- Εισαγωγή στο χειρισμό εξαιρέσεων
- Χειρισμός εξαιρέσεων στην C++
- Χειρισμός εξαιρέσεων στην Java
- Χειρισμός εξαιρέσεων στην Python και στην Ruby
- Εισαγωγή στο χειρισμό γεγονότων
- Χειρισμός γεγονότων στην Java
- Χειρισμός γεγονότων στην C#

Εισαγωγή στο χειρισμό εξαιρέσεων

- Σε μια γλώσσα χωρίς χειρισμό εξαιρέσεων
  - Όταν συμβαίνει μια εξαίρεση, ο έλεγχος μεταβιβάζεται στο λειτουργικό σύστημα, όπου εμφανίζεται ένα μήνυμα και το πρόγραμμα τερματίζεται
- Σε μια γλώσσα με χειρισμό εξαιρέσεων
  - Τα προγράμματα μπορούν να συλλαμβάνουν ορισμένες εξαιρέσεις, οπότε δίνεται η δυνατότητα να επιδιορθωθεί το πρόβλημα και το πρόγραμμα να συνεχίσει να εκτελείται

Βασικές έννοιες

- Πολλές γλώσσες επιτρέπουν στα προγράμματα να συλλαμβάνουν λάθη εισόδου/εξόδου (συμπεριλαμβανομένου του EOF)
- Μια εξαίρεση *exception* είναι ένα μη συνηθισμένο γεγονός, που μπορεί να έχει προκύψει από κάποιο σφάλμα, που ανιχνεύεται είτε από το υλικό είτε από το λογισμικό και που μπορεί να απαιτεί εξειδικευμένο χειρισμό
- Ο εξειδικευμένος χειρισμός που μπορεί να απαιτείται μετά την ανίχνευση μιας εξαίρεσης ονομάζεται χειρισμός εξαιρέσεων (*exception handling)*
- Η μονάδα κώδικα χειρισμού των εξαιρέσεων ονομάζεται χειριστής εξαιρέσεων (*exception handler)*

Εναλλακτικοί μηχανισμοί έναντι του Χειρισμού Εξαιρέσεων

- Μια εξαίρεση προκαλείται (γίνεται raise) όταν συμβαίνει το συσχετιζόμενο με αυτή γεγονός
- Ακόμα και μια γλώσσα που δεν έχει δυνατότητες χειρισμού εξαιρέσεων μπορεί να ορίσει, να ανιχνεύσει, να προκαλέσει και να χειριστεί εξαιρέσεις (που ορίζονται από τον χρήστη και που ανιχνεύονται από το λογισμικό)
- Εναλλακτικές:
  - Αποστολή μιας βοηθητικής παραμέτρου ή χρήση της τιμής επιστροφής για να υποδηλώσει την κατάσταση επιστροφής ενός υποπρογράμματος
  - Πέρασμα μιας παραμέτρου ετικέτας σε όλα τα υποπρογράμματα (error return is to the passed label)
  - Πέρασμα ενός υποπρογράμματος χειρισμού εξαιρέσεων σε όλα τα υποπρογράμματα

Πλεονεκτήματα Built-in Χειρισμού Εξαιρέσεων

- Ο κώδικας εντοπισμού λαθών είναι κουραστικός να γραφεί και μπερδεύει το πρόγραμμα
- Ο χειρισμός λαθών ενθαρρύνει τους προγραμματιστές να λάβουν υπόψη πολλά διαφορετικά πιθανά σφάλματα
- Η διάδοση εξαιρέσεων επιτρέπει την υψηλού επιπέδου επαναχρησιμοποίηση του κώδικα χειρισμού λαθών

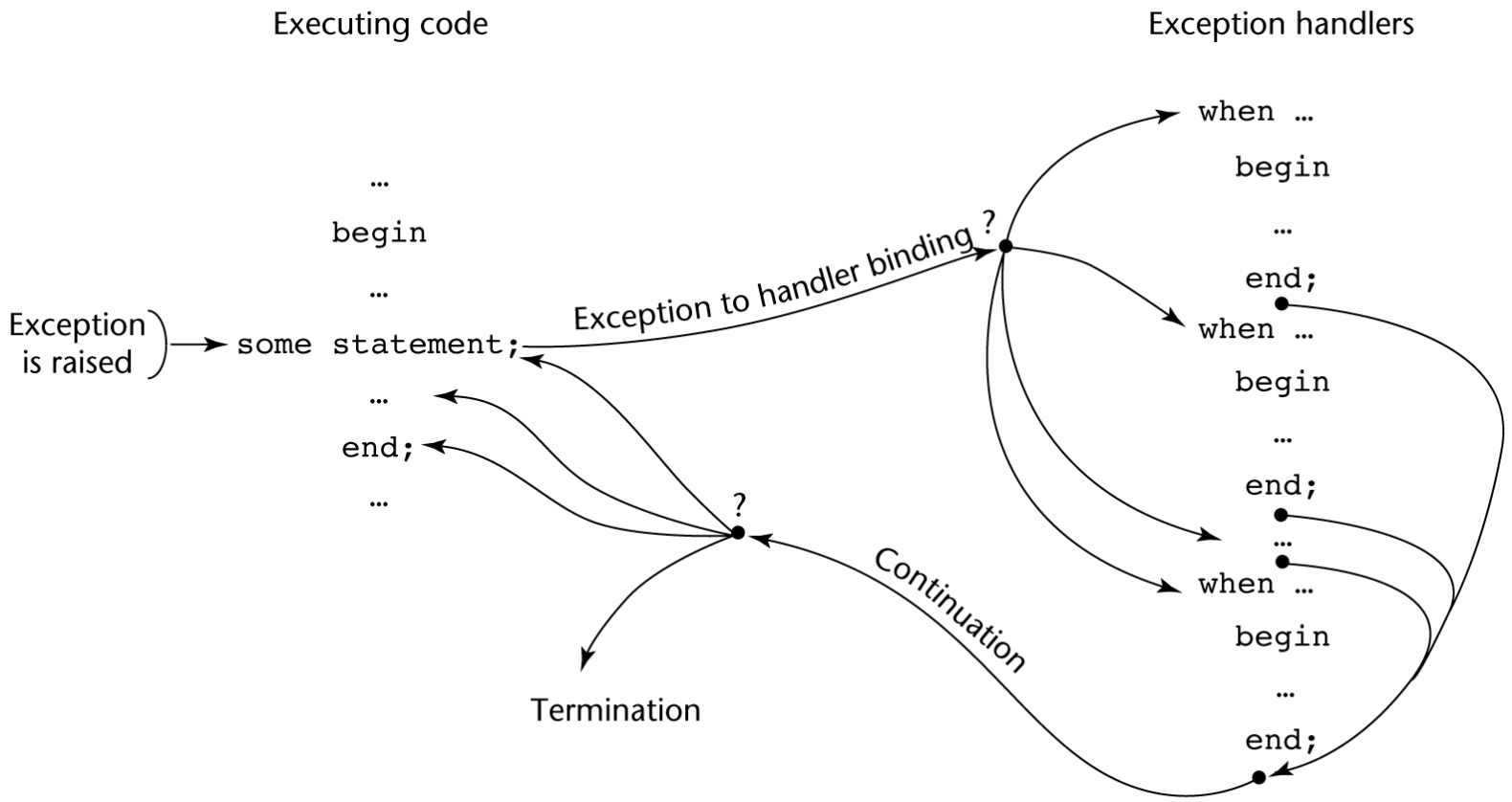
Θέματα Σχεδίασης

- Πως και που ορίζονται οι χειριστές λαθών και ποια είναι η εμβέλειά τους;
- Πως προσδένεται η εμφάνιση μιας εξαίρεσης με τον χειριστή της;
- Μπορεί να περάσει πληροφορία σχετικά με την εξαίρεση στο χειριστή της εξαίρεσης;
- Που συνεχίζει η εκτέλεση του κώδικα, αν αυτό συμβαίνει, όταν ο χειριστής εξαιρέσεων ολοκληρώσει την εκτέλεσή του; (continuation vs. resumption)
- Παρέχεται κάποιας μορφής finalization;

Θέματα Σχεδίασης (συνέχεια)

- Πως διατυπώνονται οι εξαιρέσεις που ορίζονται από τον χρήστη;
- Πρέπει να υπάρχουν προκαθορισμένοι χειριστές εξαιρέσεων για προγράμματα που δεν παρέχουν τους δικούς τους;
- Μπορούν να προκαλούνται σαφώς (explicitly) προκαθορισμένες εξαιρέσεις;
- Μπορούν λάθη ανιχνεύσιμα από το υλικό να αντιμετωπίζονται ως εξαιρέσεις με δυνατότητα χειρισμού;
- Υπάρχουν προκαθορισμένες εξαιρέσεις;
- Πώς μπορούν να απενεργοποιηθούν οι εξαιρέσεις, αν υπάρχει αυτή η δυνατότητα;

Έλεγχος Ροής Χειρισμού Εξαιρέσεων



Χειρισμός εξαιρέσεων στην C++

- Προστέθηκε στην C++ το 1990
- Ο σχεδιασμός τους βασίζεται στον σχεδιασμό των εξαιρέσεων στις CLU, Ada, και ML

Χειριστές εξαιρέσεων στην C++

- Μορφή Χειριστών Εξαιρέσεων:

```
try {
-- κώδικας που αναμένεται να προκαλέσει εξαίρεση
}
catch (formal parameter) {
-- κώδικας χειρισμού εξαίρεσης
}
...
catch (formal parameter) {
-- κώδικας χειρισμού εξαίρεσης
}
```

Η συνάρτηση `catch`

- **`catch`** είναι το όνομα όλων των χειριστών – είναι ένα υπεφορτωμένο όνομα, οπότε το όνομα της τυπικής παραμέτρου κάθε χειριστή πρέπει να είναι μοναδικό
- Η τυπική παράμετρος δεν χρειάζεται να έχει μεταβλητή
  - Μπορεί να είναι απλά ένα όνομα τύπου που διαφοροποιεί τον καθέναν χειριστή από τους άλλους
- Η τυπική παράμετρος μπορεί να χρησιμοποιηθεί για να μεταφέρει πληροφορίες στον χειριστή
- Η τυπική παράμετρος μπορεί να είναι τρεις τελείες (ellipsis), οπότε σε αυτήν την περίπτωση χειρίζεται όλες τις εξαιρέσεις που δεν έχουν χειριστεί ακόμα

Πρόκληση Εξαιρέσεων

- Οι εξαιρέσεις προκαλούνται σαφώς με την εντολή:

  `throw` [*expression*]`;`
- Οι αγκύλες είναι μετασύμβολα
- Ένα **`throw`** χωρίς όρισμα μπορεί να εμφανίζεται σε έναν χειριστή – όταν συμβαίνει αυτό, απλά επανα-προκαλεί την εξαίρεση, που στη συνέχεια χειρίζεται αλλού
- Ο τύπος της έκφρασης προσδιορίζει τον κατάλληλο χειριστή

Εξαιρέσεις άνευ-χειρισμού

- Μια εξαίρεση άνευ-χειρισμού (unhandled exception) προωθείται στη συνάρτηση από την οποία έγινε η κλήση της συνάρτησης που προκάλεσε την εξαίρεση
- Η προώθηση αυτή συνεχίζεται μέχρι τη συνάρτηση main
- Αν δεν βρεθεί χειριστής, τότε καλείται ο προκαθορισμένος χειριστής

Συνέχεια

- Όταν ένας χειριστής εξαιρέσεων ολοκληρώσει την εκτέλεσή του, ο έλεγχος μεταβιβάζεται στην πρώτη εντολή μετά τον τελευταίο χειριστή στη σειρά των χειριστών που ανήκει ο χειριστής
- Άλλες επιλογές σχεδίασης:
  - Όλες οι εξαιρέσεις είναι ορισμένες από τον χρήστη
  - Οι εξαιρέσεις ούτε ορίζονται ούτε δηλώνονται
  - Ο προκαθορισμένος χειριστής, `unexpected`, απλά τερματίζει το πρόγραμμα – το `unexpected` μπορεί να επαναοριστεί από τον χρήστη
  - Οι συναρτήσεις μπορούν να προδιαγράψουν τη λίστα των εξαιρέσεων που θα μπορούσαν να προκαλέσουν
  - Χωρίς σχετική προδιαγραφή, μια συνάρτηση μπορεί να προκαλέσει οποιαδήποτε εξαίρεση (με την πρόταση **`throw`**)

Αποτίμηση εξαιρέσεων στη C++

- Δεν υπάρχουν προκαθορισμένες εξαιρέσεις
- Είναι κάπως περίεργο ότι εξαιρέσεις δεν έχουν ονόματα και ότι εξαιρέσεις που ανιχνεύονται από το υλικό και το λογισμικό συστήματος δεν μπορούν να χειριστούν
- Η πρόσδεση εξαιρέσεων σε διαχειριστές μέσω του τύπου της παραμέτρου δεν διευκολύνει την αναγνωσιμότητα

Χειρισμός Εξαιρέσεων στην Java

- Βασίζεται στον χειρισμό εξαιρέσεων της C++, αλλά είναι περισσότερο κοντά στη OOP φιλοσοφία
- Όλες οι εξαιρέσεις είναι αντικείμενα κλάσεων που είναι απόγονοι της κλάσης `Throwable`

Κλάσεις Εξαιρέσεων

- Η βιβλιοθήκη της Java περιλαμβάνει δύο υποκλάσεις του `Throwable` :
  - `Error`
    - Προκύπτει από τον διερμηνευτή της Java interpreter όπως όταν για παράδειγμα συμβαίνει υπερχείλιση σωρού
    - Ο χειρισμός τους δεν γίνεται από προγράμματα του χρήστη
  - `Exception`
    - Ορισμένες από τον χρήστη εξαιρέσεις που είναι συνήθως υποκλάσεις της Exception
    - Έχει δύο προκαθορισμένες υποκλάσεις την `IOException` και την `RuntimeException` (π.χ., `ArrayIndexOutOfBoundsException` και `NullPointerException`

Χειριστές Εξαιρέσεων στην Java

- Ορίζονται παρόμοια με τη C++, με τη διαφορά ότι κάθε **`catch`** απαιτεί μια παράμετρο στην οποία πρέπει να έχει δοθεί όνομα και όλες οι παράμετροι πρέπει να είναι απόγονοι της `Throwable`
- Το συντακτικό της πρότασης **`try`** είναι ακριβώς όπως και αυτό της C++
- Οι εξαιρέσεις προκύπτουν με το **`throw`**, όπως στη C++, αλλά συχνά η **`throw`** περιλαμβάνει το τελεστή **`new`** για την δημιουργία ενός αντικειμένου, όπως για παράδειγμα:

```
throw new MyException();
```

Πρόσδεση εξαιρέσεων σε χειριστές εξαιρέσεων

- Η πρόσδεση μιας εξαίρεσης σε έναν χειριστή είναι απλούστερη στην Java από ότι στη C++
  - Μια εξαίρεση προσδένεται στον πρώτο χειριστή που εντοπίζεται και έχει παράμετρο στην ίδια κλάση ή κάποιο απόγονό της με το αντικείμενο που έγινε throw στην εξαίρεση
- Μια εξαίρεση μπορεί να χειριστεί και να γίνει throw ξανά μέσα σε ένα χειριστή εξαιρέσεων (ο χειριστής μπορεί να κάνει throw μια διαφορετική εξαίρεση)

Συνέχεια (κώδικα με εξαιρέσεις)

- Αν δεν βρεθεί χειριστής εξαιρέσεων στην **`try`**, η αναζήτηση συνεχίζεται στο πλησιέστερο περιέχων **`try`**, κ.λπ.
- Αν δεν βρεθεί χειριστής εξαιρέσεων στην μέθοδο, η εξαίρεση προωθείται εκεί όπου κλήθηκε η μέθοδος
- Αν δεν βρεθεί χειριστής εξαιρέσεων γενικά (σε όλη τη διαδρομή μέχρι τη `main`), το πρόγραμμα τερματίζεται
- Για να διασφαλιστεί ότι όλες οι εξαιρέσεις συλλαμβάνονται, μπορεί να τοποθετηθεί ένας χειριστής εξαιρέσεων σε κάθε **`try`** που να συλλαμβάνει όλες τις εξαιρέσεις
  - Απλά χρησιμοποιώντας ως παράμετρο κλάσης το `Exception`
  - Βέβαια, θα πρέπει να είναι ο τελευταίος χειριστής εξαιρέσεων του **`try`**

Checked εξαιρέσεις και Unchecked εξαιρέσεις

- Η εντολή **`throws`** της Java είναι αρκετά διαφορετική από την εντολή **`throw`** της C++
- Οι εξαιρέσεις της κλάσης `Error` και της κλάσης `RunTimeException` και όλες οι απόγονοί τους ονομάζονται unchecked exceptions, ενώ όλες οι άλλες εξαιρέσεις ονομάζονται checked exceptions
- Οι checked εξαιρέσεις που προκαλούνται από μια μέθοδο θα πρέπει είτε:
  - Να περιέχονται σε μια λίστα στην πρόταση **`throws`** της μεθόδου, ή
  - Να χειρίζονται από τη μέθοδο

Άλλες Επιλογές Σχεδίασης

- Μια μέθοδος δεν μπορεί να δηλώνει περισσότερες εξαιρέσεις στην **`throws`** πρόταση της από όσες η μέθοδος επαναορίζει (overrides)
- Μια μέθοδος που καλεί μια άλλη μέθοδο που στην πρόταση **`throws`** περιέχει μια συγκεκριμένη checked εξαίρεση έχει τρεις τρόπους χειρισμού της εξαίρεσης:
  - Σύλληψη και χειρισμός της εξαίρεσης μέσα στη μέθοδο
  - Σύλληψη της εξαίρεσης και πρόκληση μιας εξαίρεσης που βρίσκεται στη λίστα της πρότασης **`throws`** της μεθόδου
  - Δήλωση της εξαίρεσης στην πρόταση **`throws`** της μεθόδου, χωρίς να γίνεται χειρισμός της εξαίρεσης στην ίδια την μέθοδο

Η πρόταση `finally`

- Μπορεί να εμφανίζεται στο τέλος ενός try
- Μορφή:

```
finally {
...
}
```

- Σκοπός: Καθορίζει κώδικα που θα εκτελεστεί ανεξάρτητα από το τι συμβαίνει στο κώδικα εντός του **`try`**

Παράδειγμα

- Ένα try με finally (το finally μπλοκ βρίσκεται μετά το μπλοκ του try)

```
try {
    for (index = 0; index < 100; index++) {
          ...
          if (...) {
                return;
          }  //** end of if
}  //** τέλος της πρότασης try
finally {
    ...
}  //** τέλος του try συνολικά
```

Ισχυρισμοί (Assertions)

- Οι ισχυρισμοί είναι εντολές που ορίζουν μια λογική έκφραση που αφορά την τρέχουσα κατάσταση των υπολογισμών σε ένα πρόγραμμα
- Όταν η έκφραση αποτιμάται ως αληθής δεν συμβαίνει τίποτα
- Όταν αποτιμάται ως ψευδής τότε προκαλείται μια εξαίρεση `AssertionError`
- Μπορούν να απενεργοποιηθούν κατά το χρόνο εκτέλεσης χωρίς να απαιτείται αλλαγή του προγράμματος ή επανα-μεταγλώττιση
- Δύο μορφές:
  - **`assert`** *condition*;
  - **`assert`** *condition*: *expression*;

Αποτίμηση Εξαιρέσεων στην Java

- Οι τύποι των εξαιρέσεων στην Java φαίνεται να έχουν περισσότερο νόημα από ότι στην C++
- Η πρόταση **`throws`** είναι είναι καλύτερη από της C++ (Η πρόταση **`throw`** στην C++ λέει λίγα στον προγραμματιστή)
- Η πρόταση **`finally`** είναι συχνά χρήσιμη
- Ο διερμηνευτής της Java προκαλεί πολλές διαφορετικές εξαιρέσεις που μπορούν να χειριστούν από τα προγράμματα του χρήστη

Χειρισμός Εξαιρέσεων στην Python

- Οι εξαιρέσεις είναι αντικείμενα, η βασική κλάση είναι η `BaseException`
- Όλες οι προκαθορισμένες και οι ορισμένες από τον χρήστη εξαιρέσεις προκύπτουν από την `Exception`
- Προκαθορισμένες υποκλάσεις της `Exception` είναι η `ArithmeticError` (με υποκλάσεις τις `OverflowError`, `ZeroDivisionError`, και την `FloatingPointError`) και την `LookupError` (με υποκλάσεις την `IndexError` και την `KeyError`)

Χειρισμός Εξαιρέσεων στην Python (συνέχεια)

```
try:
   - Το μπλοκ try
except Exception1:
   - Ο χειριστής της Exception1
except Exception2:
   - Ο χειριστής της Exception2
...
else:
   - Το μπλοκ else (αν δεν έχει προκύψει εξαίρεση)
finally:
   - το μπλοκ finally (γίνεται σε όλες τις περιπτώσεις)
```

Χειρισμός Εξαιρέσεων στην Python (συνέχεια)

- Οι χειριστές εξαιρέσεων χειρίζονται την επώνυμη εξαίρεση και όλες τις υποκλάσεις αυτής της εξαίρεσης, έτσι αν η επώνυμη εξαίρεση είναι `Exception`, τότε χειρίζεται όλες τις προκαθορισμένες εξαιρέσεις και τις ορισμένες από τον χρήστη εξαιρέσεις
- Οι εξαιρέσεις που δεν έχουν χειριστεί (unhandled exceptions) προωθούνται στο πλησιέστερο περιέχων μπλοκ try, αν δεν βρεθεί χειριστής, τότε καλείται ο προκαθορισμένος χειριστής
- Η `raise IndexError` δημιουργεί ένα στιγμιότυπο
- Η εξαίρεση που προκύπτει μπορεί να «συλληφθεί» ως εξής:

```
except Exception as ex_obj:
```

Χειρισμός Εξαιρέσεων στην Python (συνέχεια)

- Η εντολή `assert` ελέγχει μια λογική έκφραση (η πρώτη παράμετρος) και στέλνει τη δεύτερη παράμετρό της στον κατασκευαστή του αντικειμένου εξαίρεσης που θα προκύψει:

```
assert test, data
```

Χειρισμός Εξαιρέσεων στην Ruby

- Οι εξαιρέσεις είναι αντικείμενα
- Υπάρχουν πολλές προκαθορισμένες εξαιρέσεις
- Όλες οι εξαιρέσεις που χειρίζονται από τον χρήστη είναι είτε της κλάσης `StandardError` ή κάποιας υποκλάσης της
- Η `StandardError` είναι υποκλάση της `Exception`, που έχει δύο μεθόδους τη `message` και τη `backtrace`
- Οι εξαιρέσεις μπορούν να προκαλούνται με τη **`raise`**, που συχνά λαμβάνει τη μορφή:

```
raise "bad parameter" if count == 0
```

Χειρισμός Εξαιρέσεων στην Ruby (συνέχεια)

- Οι χειριστές εξαιρέσεων τοποθετούνται στο τέλος ενός begin-end μπλοκ κώδικα, ξεκινώντας με **`rescue`**

  `begin`

  – Εντολές ενός μπλοκ κώδικα

  `rescue`

  – Χειριστής

  `end`

- Το μπλοκ μπορεί να περιέχει **`else`** και προτάσεις **`ensure`**, που είναι σαν το **`else`** και το **`finally`** της Java

Χειρισμός Εξαιρέσεων στην Ruby (συνέχεια)

- Σε αντίθεση με C++, Java, Python, στην Ruby ο κώδικας που προκαλεί μια εξαίρεση μπορεί να επανα-εκτελεστεί τοποθετώντας την εντολή `retry` στο τέλος του χειριστή εξαιρέσεων

Εισαγωγή στον Χειρισμό Γεγονότων

- Ένα γεγονός *(event)* είναι μια ειδοποίηση ότι κάτι συγκεκριμένο έχει συμβεί, όπως ένα κλικ ποντικιού σε ένα πλήκτρο μιας γραφικής διεπαφής
- Ο χειριστής γεγονότων (*event handler)* είναι ένα τμήμα κώδικα το οποίο εκτελείται ως απόκριση σε ένα γεγονός

Τα Συστατικά του Java Swing GUI

- Το Text box είναι ένα αντικείμενο της κλάσης `JTextField`
- Το Radio button είναι ένα αντικείμενο της κλάσης `JRadioButton`
- Η οθόνη ενός applet είναι ένα frame, και πρόκειται για μια πολύ-επίπεδη δομή
- Το content pane αποτελεί ένα επίπεδο, όπου τα applets τοποθετούν την έξοδό τους
- Τα συστατικά ενός GUI μπορούν να τοποθετηθούν σε ένα frame
- Τα αντικείμενα διαχείρισης τοποθέτησης (layout managers) χρησιμοποιούνται για να ελέγξουν την τοποθέτηση των συστατικών

Το Μοντέλο Γεγονότων (event-model) της Java

- Οι αλληλεπιδράσεις του χρήστη με τα συστατικά ενός GUI components δημιουργούν γεγονότα που μπορούν να «συλληφθούν» από χειριστές γεγονότων, που ονομάζονται *event listeners*
- Μια γεννήτρια γεγονότων ενημερώνει έναν listener για ένα γεγονός στέλνοντας ένα μήνυμα
- Ένα interface χρησιμοποιείται για να κάνει τις μεθόδους χειρισμού γεγονότων να συμφωνούν με ένα στάνταρντ πρωτόκολλο
- Μια κλάση που υλοποιεί έναν listener must θα πρέπει να υλοποιεί το interface του listener

Το Μοντέλο Γεγονότων (event-model) της Java (συνέχεια)

- Μια κλάση γεγονότων είναι η `ItemEvent`, που σχετίζεται με τα γεγονότα του κλικ σε checkbox, σε radio button, ή σε ένα στοιχείο μιας λίστας
- Το interface `ItemListener` προδιαγράφει τη μέθοδο, `itemStateChanged`, που είναι χειριστής των γεγονότων `ItemEvent`
- Ο listener δημιουργείται με το `addItemListener`

Χειρισμός Γεγονότων στην C#

- Ο χειρισμός γεγονότων στην C# (και στις άλλες .NET γλώσσες) είναι παρόμοιος με τον χειρισμό γεγονότων στην Java
- Το .NET έχει δύο προσεγγίσεις, τα Windows Forms και το Windows Presentation Foundation—θα αναφέρουμε μόνο το πρώτο (το οποίο αποτελεί και την αρχική προσέγγιση)
- Μια εφαρμογή αποτελεί υποκλάση της προκαθορισένης κλάσης `Form` (που ορίζεται στο `System.Windows.Forms`)
- Δεν υπάρχει ανάγκη να δημιουργηθεί ένα frame ή ένα panel στο οποίο θα τοποθετηθούν τα GUI συστατικά
- Τα `Label` αντικείμενα χρησιμοποιούνται για να τοποθετηθεί κείμενο στο παράθυρο
- Τα radio buttons είναι αντικείμενα της κλάσης `RadioButton`

Χειρισμός Γεγονότων στην C# (συνέχεια)

- Τα συστατικά τοποθετούνται αναθέτοντας ένα νέο αντικείμενο `Point` στην ιδιότητα Location του κάθε συστατικού

```
private RadioButton plain = new RadioButton();
plain.Location = new Point(100, 300);
plain.Text = "Plain";
controls.Add(plain);
```

- Όλοι οι χειριστές γεγονότων της C# έχουν το ίδιο πρωτόκολλο, ο τύπος επιστροφής είναι `void` και οι δύο παράμετροι είναι τύπου `object` και `EventArgs`

Χειρισμός Γεγονότων στην C# (συνέχεια)

- Ένας χειριστής γεγονότων μπορεί να έχει οποιοδήποτε όνομα
- Ένα radio button μπορεί να ελεγχθεί με την Boolean ιδιότητα Checked του πλήκτρου

```
private void rb_CheckedChanged (object o, EventArgs e) {
   if (plain.Checked) …
   ...
  }
```

- Για να καταχωρηθεί ένα γεγονός, πρέπει να δημιουργηθεί ένα νέο `EventHandler` αντικείμενο και να προστεθεί ως προκαθορισμένο delegate για το γεγονός

Χειρισμός Γεγονότων στην C# (συνέχεια)

- Όταν ένα radio button αλλάζει από unchecked σε checked, τότε προκαλείται το γεγονός `CheckedChanged`
- Το συσχετιζόμενο delegate γίνεται αναφορά με το όνομα του γεγονότος
- Αν ο χειριστής γεγονότων ονομάζονταν `rb_CheckedChanged`, θα μπορούσαμε να το καταχωρήσουμε στο radio button με όνομα `plain` ως:

```
plain.CheckedChanged += new EventHandler(rb_CheckedChanged);
```

Σύνοψη

- Η Ada παρέχει εκτεταμένες δυνατότητες χειρισμού εξαιρέσεων με ένα περιεκτικό σύνολο built-in εξαιρέσεων.
- Η C++ δεν περιέχει προκαθορισμένες εξαιρέσεις
- Οι εξαιρέσεις προσδένονται σε χειριστές εξαιρέσεων συνδέοντας τον τύπο μιας έκφρασης στην εντολή **`throw`** με τον τύπο της τυπικής παραμέτρου της συνάρτησης **`catch`**
- Οι εξαιρέσεις στην Java είναι παρόμοιες με τις εξαιρέσεις της C++ με τη διαφορά ότι οι εξαιρέσεις στην Java exception πρέπει να είναι απόγονοι της κλάσης `Throwable`. Επιπρόσθετα, η Java περιέχει την πρόταση **`finally`**
- Ένα γεγονός είναι μια ειδοποίηση ότι κάτι έχει συμβεί που απαιτεί χειρισμό μέσω ενός χειριστή γεγονότων
- Ο χειρισμός γεγονότων στην Java ορίζεται για τα συστατικά Swing
- Ο χειρισμός γεγονότων στην C# ακολουθεί το μοντέλο .NET, που είναι παρόμοιο με το μοντέλο της Java