Μεθοδολογία Προγραμματισμού

Εισαγωγή στη Java

Νικόλαος Πεταλίδης

Τμήμα Μηχανικών Πληροφορικής και Επικοινωνιών
Διεθνές Πανεπιστήμιο της Ελλάδος

Εισαγωγή
Εαρινό Εξάμηνο

Η γλώσσα Java

- Είναι μια *αντικειμενοστραφής* γλώσσα προγραμματισμού
- Βασικό της χαρακτηριστικό είναι η φιλοσοφία του *Write once, run anywhere*
- Είναι από τις πιο δημοφιλείς γλώσσες προγραμματισμού με περίπου 9 εκ. προγραμματιστές ανά τον κόσμο

Ιστορικό

- Παρουσιάστηκε το 1995 από την εταιρία Sun Microsystems (έχει συγχωνευθεί με την Oracle)
- Δημιουργήθηκε από τον James Gosling
- Βρίσκεται στην έκδοση 11
- Compilers για τη Java εκδίδονται από την Oracle αλλά υπάρχουν και compilers ελεύθερου λογισμικού
- Από το Μάιο του 2007 το μεγαλύτερο μέρος της τεχνολογίας της Java είναι διαθέσιμο με άδεια ανοιχτού λογισμικού (GNU General Public Licence)

Χαρακτηριστικά της Java

Αντικειμενοστραφής Εφαρμόζονται οι κλασικές έννοιες αντικειμενοστραφούς προγραμματισμού

Κατανεμημένη Δυνατότητα καταμερισμού της εκτέλεσης του κώδικα σε διαφορετικούς υπολογιστές

Πολυνηματική Ταυτόχρονη εκτέλεση πολλών νημάτων (threads)

Μεταφέρσιμη Ο ίδιος κώδικας μπορεί να εκτελεστεί ανεξαρτήτως λειτουργικού.

Ασφαλής Αποτρέπεται η μη εξουσιοδοτημένη πρόσβαση στους σταθμούς πελατών που εκτελούν κώδικα σε Java μέσω δικτύου

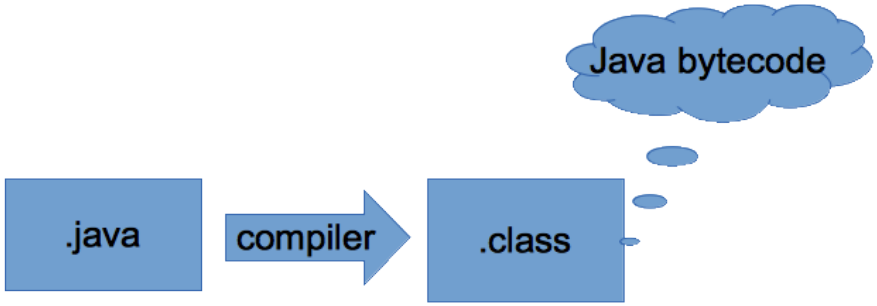
Γιατί Java

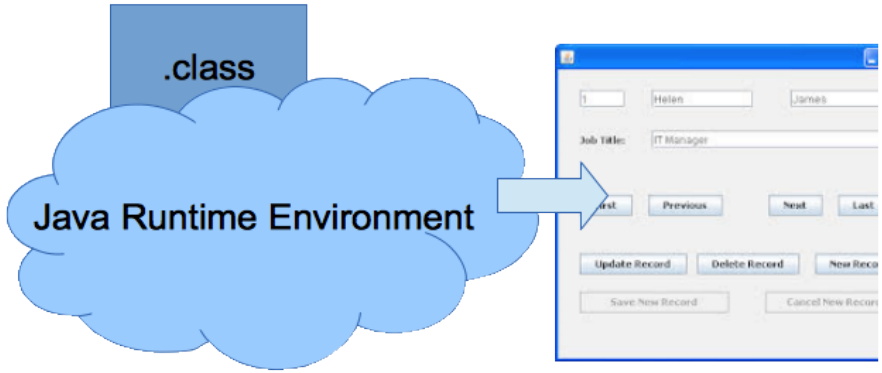
- Είναι μια πολύ διαδεδομένη γλώσσα που αποκτά συνεχώς νέους οπαδούς
- Υπάρχει τεράστια συλλογή από βιβλιοθήκες που μειώνουν πολύ τον απαιτούμενο χρόνο ανάπτυξης
- ... Δεν είναι C++

Write once run everywhere

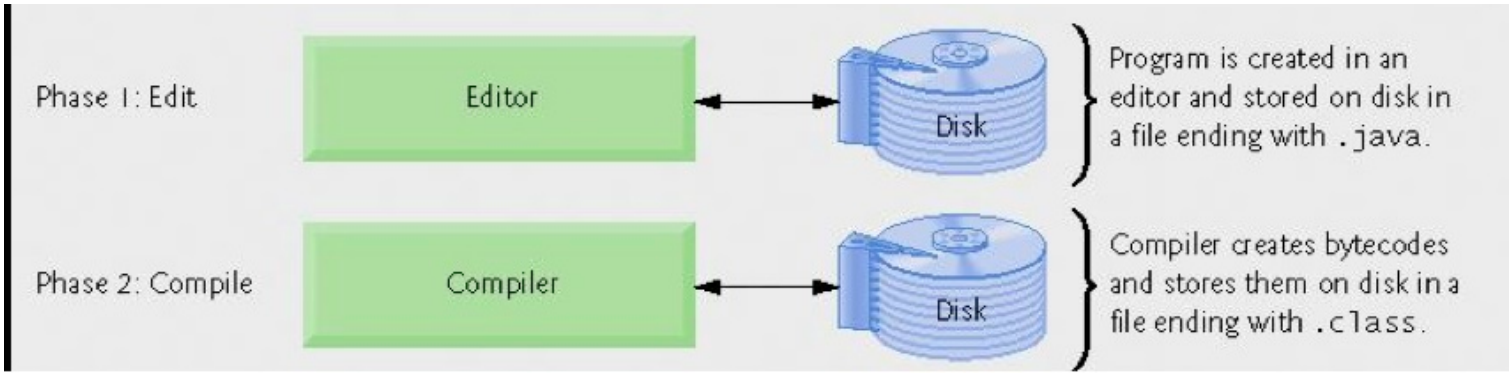
- Ένα πρόγραμμα σε Java μπορεί να τρέχει σε μια πλειάδα συστημάτων χωρίς να γίνεται recompile
- Αυτό είναι δυνατό γιατί το εκτελέσιμο της Java δεν είναι σε γλώσσα μηχανής, αλλά σε μια ενδιάμεση αναπαράσταση γνωστή ως Java bytecode μιας ιδεατής μηχανής, της Java Virtual Machine
- Η μετατροπή του bytecode σε πραγματικό κώδικα μηχανής γίνεται από το περιβάλλον της Java (Java Runtime Environment)
- Αυτή τη στιγμή υπάρχει η δυνατότητα για όλα τα μεγάλα (και μικρά) λειτουργικά συστήματα η μετατροπή από bytecode σε πραγματική γλώσσα μηχανής.

Μεταγλώττιση και εκτέλεση Java

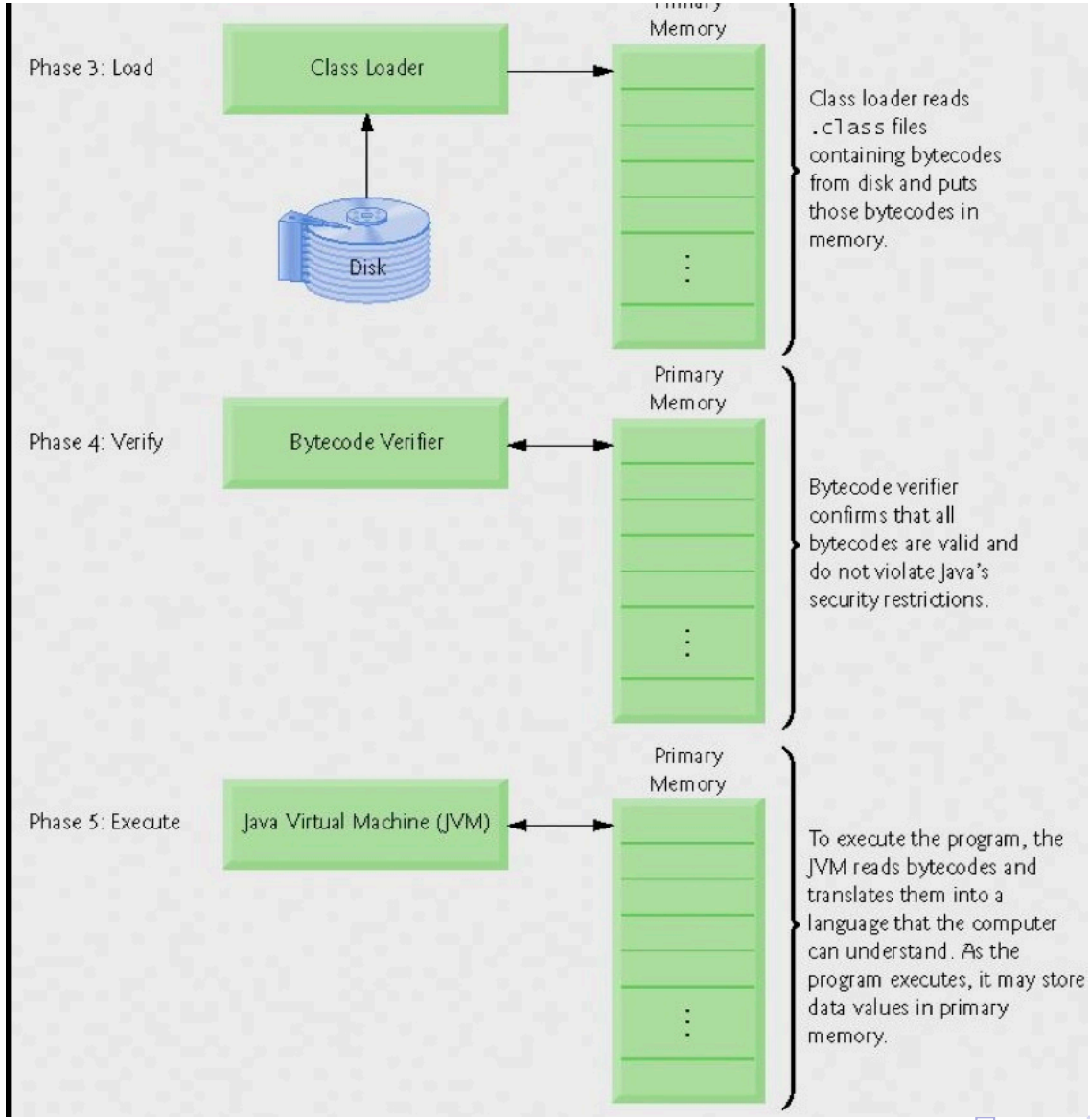




Οι φάσεις μεταγλώττισης πιο αναλυτικά



Οι φάσεις εκτέλεσης πιο αναλυτικά



Η πλατφόρμα της Java

- Αποτελείται από 2 μέρη:
  - Java Virtual Machine
  - Java APIs (οι "βιβλιοθήκες")
- Έρχεται σε τρεις βασικές εκδόσεις
  - Java 2 Standard Edition (SE): APIs ικανά για ανάπτυξη desktop και δικτυακών εφαρμογών
  - Java 2 Enterprise Edition (EE): APIs ικανά για ανάπτυξη web εφαρμογών
  - Java 2 Micro Edition (ME): Για ανάπτυξη εφαρμογών για κινητά

Java Virtual Machine

- «Εκτελεί» μεταγλωττισμένα Java προγράμματα που ονομάζονται bytecode αρχεία.
- Τα bytecode αρχεία είναι ανεξάρτητα πλατφόρμας και μπορούν να εκτελεστούν από οποιονδήποτε υπολογιστή διαθέτει μία JVM.
- Η JVM φορτώνει τις κλάσεις που χρειάζονται για να εκτελεστεί το Java πρόγραμμα (class loader).
- Η JVM «επικυρώνει» (verifies) την εγκυρότητα των bytecode αρχείων πριν τα εκτελέσει (bytecode verifier).

JDK και JRE

JDK είναι το Java Development Kit. Παρέχει τον compiler και άλλα χρήσιμα εργαλεία για την ανάπτυξη της Java, τις βιβλιοθήκες και το JVM

JRE είναι το Java Runtime Environment. Υποσύνολο του JDK που περιέχει τα απαραίτητα αρχεία για να μπορούμε να εκτελούμε προγράμματα Java

Τι να κατεβάσουμε;

Αν θέλετε να κάνετε ανάπτυξη κώδικα και να τρέχετε προγράμματα Java χρειάζεστε το JDK. Αν απλώς θέλετε να τρέχετε προγράμματα Java χρειάζεστε μόνο το JRE

Τα αρχεία της Java

.java πηγαίος κώδικας (source file)

.class bytecode αρχείο (παράγεται από compiler)

.jar Ένα συμπιεσμένο αρχείο που περιέχει πολλές .class αρχεία

Παράδειγμα ενός προγράμματος

```
// This application program prints Welcome
package examples;
public class Welcome {^^I
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
```

Μεταγλώττιση και εκτέλεση του κώδικα

Μεταγλώττιση κώδικα `javac Welcome.java`

Εκτέλεση κώδικα `java Welcome`

Προσέξτε!

- Το αρχείο που θα σώσετε τον κώδικα πρέπει να έχει το ίδιο όνομα όπως η (public) κλάση (`Welcome.java`)
- Συνηθίστε να έχετε μια κλάση ανά αρχείο
- Η μεταγλώττιση θα παράγει ένα αρχείο με το όνομα `Welcome.class`

Η συνάρτηση main

- Πρέπει να υπάρχει μία συνάρτηση main προκειμένου να εκτελεστεί το πρόγραμμά μας
- Η main πρέπει να δηλωθεί ως public
- Η main πρέπει να δηλωθεί ως static
- Η main πρέπει να έχει ως παράμετρο ένα διάνυσμα από String

Βασικοί τύποι δεδομένων στη Java

Type	Size	Default	Literals
`boolean`		`false`	`true`, `false`
`byte`	8 bit	`(byte)0`	
`short`	16 bit	`(short)0`	
`int`	32 bit	`0`	`11`, `1969`, `0xff00`, `017`
`long`	64 bit	`0L`	`11L`, `0x1000L`, `0777L`
`float`	32 bit	`0.0f`	`3.141f`, `1.2e+23f`
`double`	64 bit	`0.0d`	`3.141`, `1e-9`, `0.1e10`
`char`	16 bit	`'\u0000'`	`'a'`, `'?'`,`'\n'`, `'\uFFFF'`

Διαφορές από τη C/C++

- Οι τύποι δεδομένων στη Java έχουν πάντα το ίδιο μέγεθος ανεξάρτητο από την πλατφόρμα στην οποία τρέχει το πρόγραμμα
- Υπάρχει χωριστός τύπος `boolean` ο οποίος είναι `true` ή `false` και όχι `0` και `!0`

Άλλοι τύποι δεδομένων

- Όλοι οι άλλοι τύποι δεδομένων στην Java είναι κλάσεις που κληρονομούν από τη βασική κλάση `java.lang.Object`
- Για να χρησιμοποιήσουμε ένα μη-βασικό τύπο δεδομένων πρέπει πρώτα να δεσμεύσουμε χώρο στη μνήμη χρησιμοποιώντας την εντολή `new`

```
String password = new String("find␣me");
```

Τελεστές

- Μεταξύ των βασικών τύπων μπορούν να χρησιμοποιηθούν όλοι οι βασικοί τελεστές που γνωρίζουμε από τη C/C++. `+,-,*,/` κ.τ.λ.
- Ο τελεστής `+` μπορεί να χρησιμοποιηθεί για τη συνένωση `String`

```
//"Hello Java"
  String helloJava = "Hello " + "Java";
```

final

- Η λέξη κλειδί `final` χρησιμοποιείται για να ορίσει ότι κάτι δε μπορεί να μεταβληθεί αργότερα.
  - **final** **int** x: απαγόρευση αλλαγής τιμής εντός εμβέλειας της μεταβλητής x
  - **final** **int** someMethod(): Η μέθοδος δε μπορεί να γίνει override από υποκλάσεις
  - **final** **public** **class** Person: Η κλάση δε μπορεί να κληρονομηθεί

static

- Η λέξη κλειδί `static` χρησιμοποιείται για να ορίσει ότι μια ιδιότητα ή μέθοδος ανήκει σε όλη την κλάση και όχι σε κάποιο συγκεκριμένο αντικείμενο
  - **static** **int** x; : Για όλα τα αντικείμενα το x έχει την ίδια τιμή
  - **static** **int** someMethod(): Για όλα τα αντικείμενα η μέθοδος επιστρέφει την ίδια τιμή

Import

- Είναι η αντίστοιχη (αλλά όχι ίδια) εντολή με την `#include` στη C
- Δίνει εντολή στον compiler να καταστήσει μια κλάση ή ένα σύνολο κλάσεων (πακέτο) προσβάσιμη από τον κώδικά μας.

Διαφορές #include και import

#include Δίνει εντολή στον προ-επεξεργαστή της C/C++ να αντιγράψει πλήρως τα περιεχόμενα ενός αρχείου σε ένα άλλο

import Απλώς δίνει εντολή στον compiler για το που μπορεί να βρει κατά τη διάρκεια της εκτέλεσης του προγράμματος την επιθυμητή κλάση.

Η μεταβλητή classpath

- Η μεταβλητή `CLASSPATH` στο σύστημά σας (ή σε παραμέτρους που δίνεται στο πρόγραμμα) λέει στην Java που να ψάξει για κλάσεις που έχετε ορίσει στις δηλώσεις `import`
- Αν δε δηλωθεί η Java ψάχνει μόνο στον τρέχοντα φάκελλο

Παραδείγματα

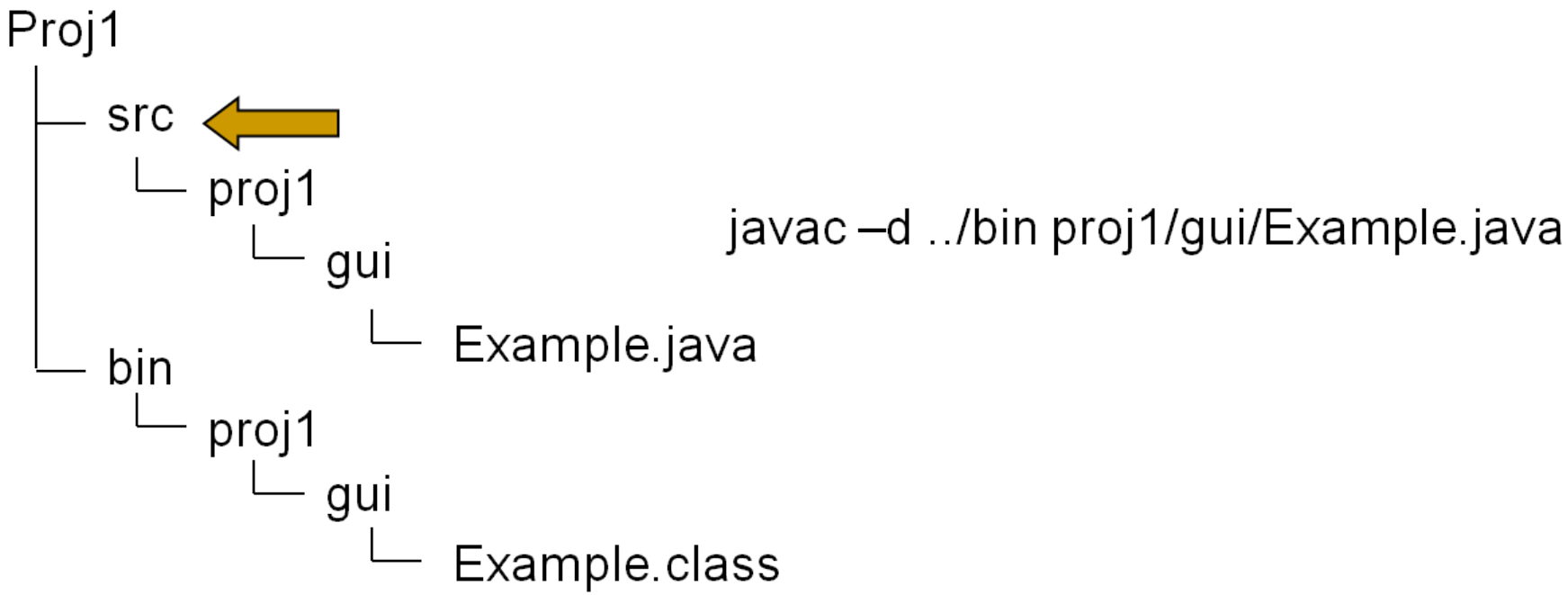
Ψάξε στο `./jardir1` κτλ

```
java -cp ./jardir1/*;./jardir2/* class_with_main
```

Java packages

- Παρόμοια (αλλά όχι ίδια) με το namespace της C++
- Οι κλάσεις μπορούν να ομαδοποιηθούν σε ένα package
- Μια κλάση μπορεί να ανήκει σε ένα μόνο package
- Όταν οι κλάσεις μεταγλωττιστούν θα ομαδοποιηθούν σε ένα φάκελλο ανά package
- Οι κλάσεις μπορούν να ορίσουν ότι ορισμένες ιδιότητες είναι ορατές σε κλάσεις που ανήκουν στο ίδιο package.

- Συνηθίζεται να έχουμε την ίδια δομή τόσο στο πηγαίο κώδικα όσο και στον παραγόμενο εκτελέσιμο.



Συμβάσεις για τα πακέτα

- Συνηθίζεται η ονομασία των πακέτων να αρχίζει με το αντίστροφο του domain μιας εταιρίας
- Αν η εταιρία σας έχει το domain `mycompany.gr` τα πακέτα στα προγράμματα της εταιρίας έχουν ονομασίες της μορφής `gr.mycompany.`...

Περιορισμός πρόσβασης

Modifier	Ίδια κλάση	Ίδιο πακέτο	Υποκλάση	Σύμπαν
`private`	✓			
(άδειο)	✓	✓		
`protected`	✓	✓	✓	
`public`	✓	✓	✓	✓

Διαχείριση μνήμης

- Γίνεται αυτόματα μέσω ενός νήματος: του garbage collector.
- Ένα κομμάτι μνήμης απελευθερώνεται runtime όταν πλέον δεν υπάρχει άλλη αναφορά προς αυτό μέσα στο πρόγραμμα.
- Ο χρήστης δεν ασχολείται (και δεν μπορεί άμεσα να το κάνει) με την απελευθέρωση μνήμης

Καλό παράδειγμα

```
SomeClass someClass = new SomeClass();
//Use someClass here
c = null; //Stop using it.
//Now it can be garbage collected
```

Κακό παράδειγμα

```
SomeClass[] someClass = new SomeClass[10];
someClass[0] = new SomeClass();
someClass[1] = new SomeClass();
int lastIndex = 1; //keeps track of the last element.
//do stuff here that change the last index
lastIndex = 0;
//SomeClass[1] was never nullified.
//It wont be garbage collected
```

Δομές ελέγχου στη Java

- Παρόμοια με C++
- Δομές Επιλογής
  - if-else
  - switch-case
- Δομές Επανάληψης
  - for
  - while
  - do while
  - break, continue, return

Χρήσιμες συντομεύσεις

- Αντί για αυτό

```
for(Iterator<String> i = someList.iterator();
      i.hasNext(); ) {
        String item = i.next();
        System.out.println(item);
}
```

- Δοκιμάστε αυτό

```
for(String item : someList ){
    System.out.println(item);
}
```

Χρήσιμες συντομεύσεις

- Αντί για αυτό

```
for(Iterator<String> i = someList.iterator();
      i.hasNext(); ) {
        String item = i.next();
        System.out.println(item);
}
```

- Δοκιμάστε αυτό

```
for(String item : someList ){
    System.out.println(item);
}
```

Διαφορές από τη C/C++

- Η Java δεν υποστηρίζει πολλαπλή κληρονομικότητα
- Η Java δεν υποστηρίζει δείκτες
- Όλες οι συναρτήσεις της Java είναι `virtual` (methods)
- Δε μπορείς να έχεις συναρτήσεις εκτός κλάσεων στη Java

Ασκησούλα

- Γράψτε ένα πρόγραμμα που αφαιρεί τα HTML tags από ένα αρχείο HTML
- Ποιος ο ρόλος της συνάρτησης `equals()` και `hash()`

Για να γράφετε σωστά κώδικα

- Καλό είναι να ακολουθείτε κάποια πρότυπα και διαδικασίες προγραμματισμού
- Να μάθετε να τεκμηριώνετε τον κώδικά σας

Πρότυπα και διαδικασίες προγραμματισμού

- Η πλειονότητα του λογισμικού αναπτύσσεται από ομάδες
- Τα πρότυπα σας υποχρεώνουν να οργανωθείτε
- Τα πρότυπα βοηθούν την ομάδα σας να καταλάβει
  - τι γράψατε
  - γιατί το γράψατε
  - τι σχέση έχει με το υπόλοιπο έργο

Τα πρότυπα

- Ορίζουν υποχρεωτική συμμόρφωση όπως παρακάτω
  - Χρησιμοποιείτε {} σε όλες τις δομές που αυτά είναι προαιρετικά
  - Τα ονόματα των μεταβλητών έχουν πάντα τη μορφή `firstSecondThird`

Τα πρότυπα

- Ορίζουν ένα στυλ σχολίων

```
/* Statement of function:
 * Component name:
 * Programmer:
 * Version:
 * Procedure Invocation:
 * Input Parameters:
 * Output Parameters:
 */
```

Κατευθυντήριες γραμμές προγραμματισμού

- Ανεξάρτητα από τη γλώσσα προγραμματισμού κάθε συστατικό προγράμματος περιλαμβάνει τουλάχιστον
  - Δομές ελέγχου
  - Αλγορίθμους
  - Δομές δεδομένων

Δομές ελέγχου

- Διατηρήστε τις δομές που έχουν ορισθεί στη σχεδίαση
- Φροντίστε τη ροή του προγράμματος
- Δομήστε σωστά το πρόγραμμα
- Μην κάνετε ειδικές λύσεις
- Ελαχιστοποιείτε τις εξαρτήσεις

Παράδειγμα: Γράψτε πρόγραμμα

- Για τα πρώτα €10.000 εισοδήματος, ο φόρος είναι 10%
- Για τα επόμενα €10.000 είναι 12%
- Για τα επόμενα €10,000 πάνω από τις €20.000 ο φόρος είναι 15%
- Για τα επόμενα €10,000 πάνω από τις €30.000 ο φόρος είναι 18%
- Για οποιοδήποτε εισόδημα πάνω από τις €40.000 ο φόρος είναι 20%

Λύση 1

```
{
tax = 0;
if (taxable_income < 10000) {
   tax = tax + .10*taxable_income;
} else if (taxable_income < 20000) {
   tax = tax + .12*(taxable_income-10000) + 1000;
} else if (taxable_income < 30000) {
   tax = tax + .15*(taxable_income-20000) + 2200;
} else if (taxable_income < 40000) {
   tax = tax + .18*(taxable_income-30000) + 3700;
} else {
   tax = tax + .2*(taxable_income-40000) + 5500;
}
```

Λύση 2

```
for (int i=2; level=1; i <= 5; i++)
  if (taxable_income > bracket[i]) {
    level = level + 1;
    tax = base[level]+percent[level]
            *(taxable_income-bracket[level]);
  }
```

Τεκμηρίωση

- Η τεκμηρίωση ενός προγράμματος εξηγεί τι κάνει το πρόγραμμα και πως το κάνει
- Ως εσωτερική τεκμηρίωση αναφέρεται όλο το περιγραφικό υλικό που γράφεται κατευθείαν στον κώδικα, δηλ. τα σχόλια
- Η εξωτερική τεκμηρίωση είναι το περιγραφικό υλικό που δεν βρίσκεται μαζί με τον κώδικα

Εξωτερική τεκμηρίωση

- Αφορά κυρίως τυχόν διαγράμματα που αναπτύσσονται
- Περιγραφές αλγορίθμων ή πρωτοκόλλων
- Θα δούμε περισσότερα στις επόμενες διαλέξεις

Εσωτερική Τεκμηρίωση

- Περιγράψτε τα συστατικά
- Προσέξτε τα σχόλια σας!

```
//Εδώ αυξάνουμετο i3
i3 = i3 + 1;
```

- Το παραπάνω σχόλιο δε μας δίνει καμιά πληροφορία

Σχόλια

- Φροντίστε τα σχόλια σας να συμφωνούν με τον κώδικα

```
//get rid of trailing newline character
i = 0;
while (date[i] >= ' ') i++;
```

Σχολιάστε τα ... λάθη σας

- Σημειώστε σημεία στον κώδικα που θέλουν βελτίωση:

```
// TODO: use a faster algorithm
```

Επιλέξτε σωστά ονόματα!

- Τι κάνει ο παρακάτω κώδικας;

```
z = (a * b) + (0.5) * (a) * (b - 40);
```

Δείτε τη διαφορά

```
weekWage = (hourRate * hours) +
   (0.5)*(hourRate)*(hours - 40);
```

Προσθέστε μεταβλητές

- Δείτε αυτό

```
if line.split(':')[0].strip() == "root"
```

- Και αυτό

```
username = line.split(':')[0].strip()
if username == "root"
```

Προσθέστε μεταβλητές

- Δείτε αυτό

```
if line.split(':')[0].strip() == "root"
```

- Και αυτό

```
username = line.split(':')[0].strip()
if username == "root"
```

Άλλα άσχημα παραδείγματα

```
final int ONE=1;
final int TWENTY=20;
```

Ονόματα συναρτήσεων

- Επιλέξτε ονόματα που δείχνουν ενέργεια, π.χ. `getTime()` αντί για `time()`
- Προσέξτε τις συναρτήσεις που επιστρέφουν λογικές τιμές (FALSE,TRUE)
  - `if (checkoctal( c ) )` vs
  - `if (isOctal( c ) )`

Ονόματα συναρτήσεων

- Επιλέξτε ονόματα που δείχνουν ενέργεια, π.χ. `getTime()` αντί για `time()`
- Προσέξτε τις συναρτήσεις που επιστρέφουν λογικές τιμές (FALSE,TRUE)
  - `if (checkoctal( c ) )` vs
  - `if (isOctal( c ) )`

Μορφοποίηση

- Η σωστή μορφοποίηση μπορεί να σας βοηθήσει ή να μη σας βοηθήσει να καταλάβετε τον κώδικα

```
for (i=0;i<10;i++);
    printf"(%"d,i);
```

Χρησιμοποιήστε τις βιβλιοθήκες

- Αντί για

```
if (c>=65 && c<=90)
```

- ή

```
if (c >= 'A' && c<='Z')
```

- δώστε

```
if (Character.isUpperCase( c ))
```

Χρησιμοποιήστε τις βιβλιοθήκες

- Αντί για

```
if (c>=65 && c<=90)
```

- ή

```
if (c >= 'A && c'<='Z)
```

- δώστε

```
if (Character.isUpperCase( c ))
```

Χρησιμοποιήστε τις βιβλιοθήκες

- Αντί για

```
if (c>=65 && c<=90)
```

- ή

```
if (c >= 'A && c'<='Z)
```

- δώστε

```
if (Character.isUpperCase( c ))
```

Προσοχή στους τύπους δεδομένων

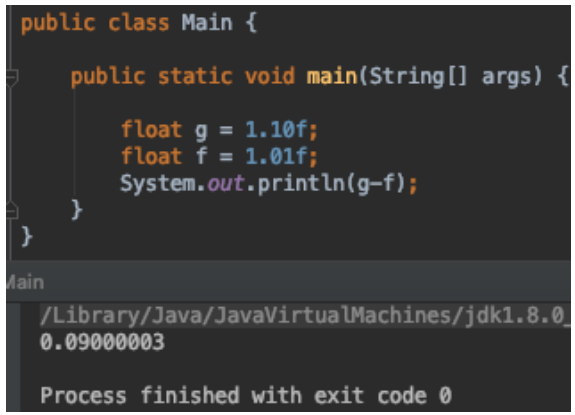
- Τι τυπώνει το παρακάτω πρόγραμμα:

```
public class Main {
    public static void main(String[] args) {
        float g = 1.10f;
        float f = 1.01f;
        System.out.println(g-f);
    }
}
```

Προσοχή στους τύπους δεδομένων

- Τι τυπώνει το παρακάτω πρόγραμμα:

```
public class Main {
    public static void main(String[] args) {
        float g = 1.10f;
        float f = 1.01f;
        System.out.println(g-f);
    }
}
```



Προγραμματίστε αμυντικά

- Μη θεωρείτε ότι οι συναρτήσεις σας θα κληθούν μόνο με σωστές παραμέτρους
- Μην επιτρέπετε σε κανέναν να αλλάξει πράγματα που δε θα έπρεπε
- Αρχικοποιείτε πάντα τις μεταβλητές σας
- Ελέγξτε τις επιστρεφόμενες τιμές

Ένα κακό παράδειγμα

```
double avg(double a[], int n)
{
   int i;
   double sum;
   sum=0.0;
   for (i=0;i<n;i++)
      sum += a[i];
   return sum/n;
}
```

Δημιουργήστε εξαιρέσεις για εξαιρετικές περιστάσεις

```
void gradeStudent() throws InvalidArgumentException
{
  if (state=='Inactive') {
      throw new InvalidArgumentException();
  }
  ...
}
```

Κανόνες τεκμηρίωσης

- Στη γλώσσα Java υπάρχουν εργαλεία τα οποία κατά το δυνατόν αυτοματοποιούν την παραγωγή τεκμηρίωσης για ένα πρόγραμμα.
- Είναι ενδιαφέρον να δούμε πως δουλεύουν
- Ακόμα και αν δε δουλεύετε σε Java μπορείτε να διδαχθείτε από τα παραδείγματα που ακολουθούν

Javadoc

- Ένα ειδικό εργαλείο, το javadoc, δέχεται ως είσοδο ένα πρόγραμμα σε Java και παράγει τεκμηρίωση σε HTML
- Για να δουλέψει σωστά, τα σχόλια στο πρόγραμμα πρέπει να γράφονται με συγκεκριμένο τρόπο

Προδιαγραφές

- Οι προδιαγραφές του κώδικα ορίζονται με σχόλια

```
/**
 * This is the typical format of a simple
 * documentation comment that spans two lines.
 */
/** This comment takes up only one line. */
```

Που μπαίνουν τα σχόλια

- Όλα τα σχόλια μπαίνουν ΑΜΕΣΩΣ ΠΡΙΝ από το ορισμό της κλάσης, του constructor, destructor, μεθόδων κτλ.
- Ανάμεσα στα σχόλια και τον ορισμό δεν επιτρέπεται τίποτα άλλο

```
/**
 * This is the class comment for the class
 * Whatever.
 */
import com.sun; // MISTAKE
public class Whatever {}
```

Ορισμοί με πολλαπλά πεδία

- Αν θέλετε σχόλια σε ορισμούς αυτού του τύπου:
- `int x, y; // Comments`
- Καλύτερα γράψτε το ως

```
int x; //Comments for x
int y; // Comments for y
```

Δομή μιας προδιαγραφής

```
/**
 * Returns the character at the specified index. An index
 * ranges from <code>0</code> to <code>length() - 1</code>.
 *
 * @param     index  the index of the desired character.
 * @return    the desired character.
 * @exception StringIndexOutOfRangeException
 *              if the index is not in the range <code>0</code>
 *              to <code>length()-1</code>.
 * @see       java.lang.Character#charValue()
 */
public char charAt(int index) {
    ...
}
```

Τα σχόλια γράφονται σε HTML

```
/**
 * This is a <b>doc</b> comment.
 * @see java.lang.Object
 */
```

Η πρώτη πρόταση

- Η πρώτη πρόταση στην περιγραφή μιας κλάσης, μεθόδου κτλ
- Πρέπει να είναι μια περίληψη ακριβής και συνοπτική.

Κύρια περιγραφή

- Η κύρια περιγραφή έρχεται μετά από την περίληψη.
- Περιγράφει συνοπτικά τη λειτουργία της κλάσης, μεθόδου κτλ

Tags (ετικέτες)

- Αυτές ακολουθούν μετά την κύρια περιγραφή.
- Περιγράφουν σημαντικά στοιχεία της κλάσης, μεθόδου κτλ.

@param

- Για κάθε μία παράμετρο, ορίζεται και ένα σχόλιο για το ρόλο της

```
/**
   * Tests a character and notifies an observer
   * according to the value of the character
   * @param ch the character to be tested
   * @param obs the observer to be notified
*/
public void testChar( char ch, Observer obs )
```

@return

- Ορίζει μια περιγραφή της τιμής που επιστρέφεται

```
/**
^^I* Tests a character and notifies an observer
^^I* according to the value of the character
^^I* @param ch the character to be tested
^^I* @param obs the observer to be notified
^^I* @return result of the action performed by the
^^I* observer on the character
^^I*/
public int testCharacter(char ch, Observer obs)
```

@throws/@exception class-name description

- Περιγράφει αν η συγκεκριμένη μέθοδος μπορεί να καταλήξει σε μια εξαίρεση (exception) και πότε συμβαίνει αυτό

Παράδειγμα

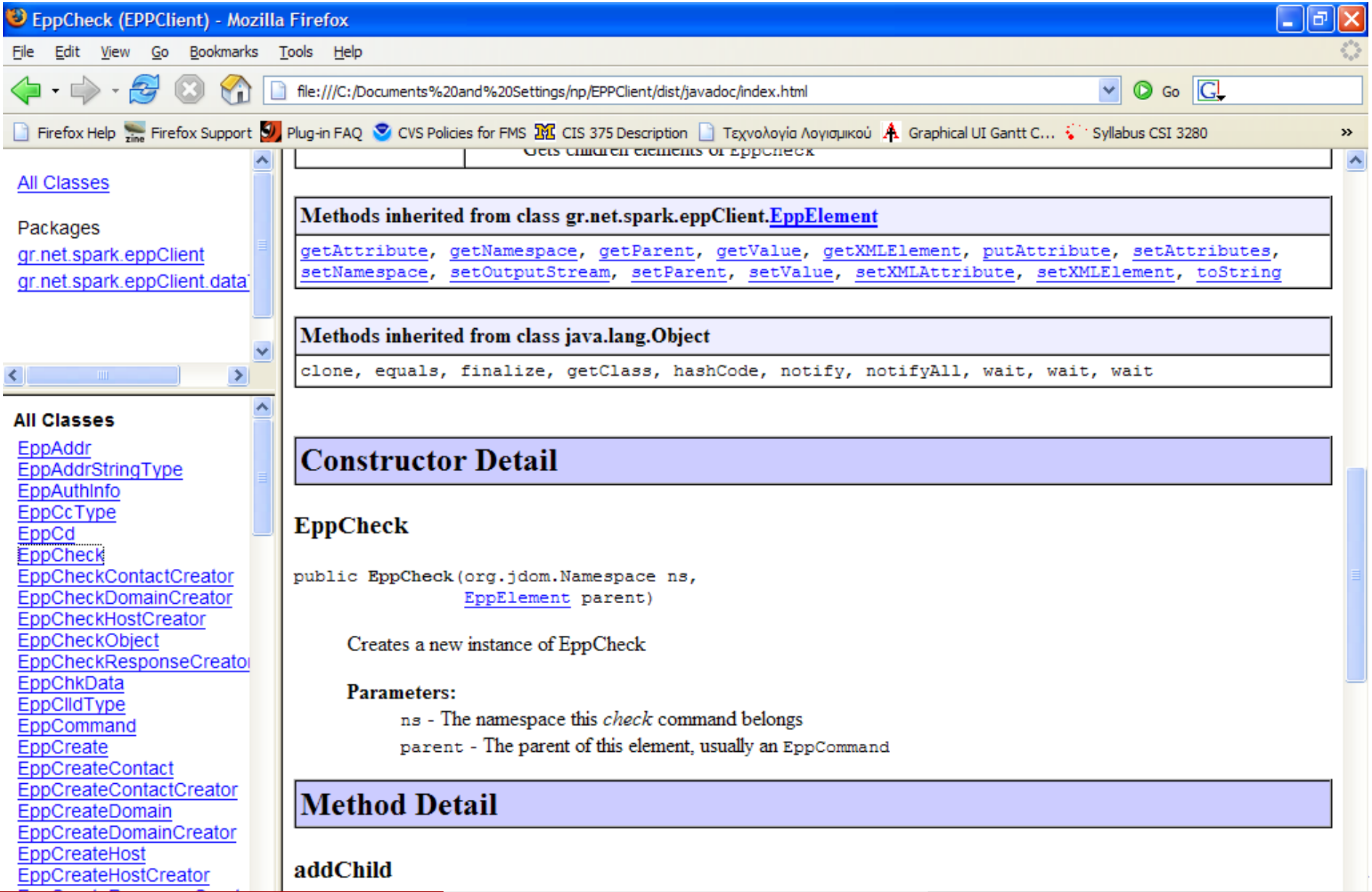
```
/**
^^I* Draws the shape of the object on the screen
^^I* instance given in the parameter.
^^I* @throws NullPointerException If the screen
^^I* object is null.
^^I* @throws ScreenSizeException If the object does
^^I* not fit on the screen.
^^I*/
public boolean drawShape(Screen screen)
^^Ithrows NullPointerException, ScreenSizeException{
...
}
```

Αποτέλεσμα

- Το javadoc τελικά παίρνει ένα σχόλιο που μοιάζει έτσι:

```
/**
   * Creates a new instance of EppCheck
   * @param ns The namespace this <I>check</I> comm
   * @param parent The parent of this element, usua
   */
    public EppCheck(Namespace ns,
                                EppElement parent) {
        super(ns,parent);
        checkObjects = new Vector();
        ...
    }
```

Και παράγει


EppCheck (EPPClient) - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
file:///C:/Documents%20and%20Settings/np/EPPClient/dist/javadoc/index.html
Go
Firefox Help Firefox Support Plug-in FAQ CVS Policies for FMS CIS 375 Description Τεχνολογία Λογισμικού Graphical UI Gantt C... Syllabus CSI 3280
All Classes
Packages
gr.net.spark.eppClient
gr.net.spark.eppClient.data
All Classes
EppAddr
EppAddrStringType
EppAuthInfo
EppCcType
EppCd
EppCheck
EppCheckContactCreator
EppCheckDomainCreator
EppCheckHostCreator
EppCheckObject
EppCheckResponseCreato
EppChkData
EppClIdType
EppCommand
EppCreate
EppCreateContact
EppCreateContactCreator
EppCreateDomain
EppCreateDomainCreator
EppCreateHost
EppCreateHostCreator
Methods inherited from class gr.net.spark.eppClient.EppElement
getAttribute, getNamespace, getParent, getValue, getXMLElement, putAttribute, setAttributes, setNamespace, setOutputStream, setParent, setValue, setXMLAttribute, setXMLElement, toString
Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait
Constructor Detail
EppCheck
public EppCheck(org.jdom.Namespace ns,
EppElement parent)
Creates a new instance of EppCheck
Parameters:
ns - The namespace this check command belongs
parent - The parent of this element, usually an EppCommand
Method Detail
addChild

Περισσότερες πληροφορίες

- Η Java είναι μια γλώσσα με τεράστιο και καλογραμμένο σύστημα πληροφοριών
  - `http://docs.oracle.com/javase/8/docs/`
  - Brian W. Kernighan and Rob Pike, The Practice of Programming, Addison-Wesley, Inc., 1999 (Γενικό για προγραμματισμό)
  - Steve McConnel, Code Complete, A Practical Handbook of Software Construction, Second Edition, Microsoft Pres (Γενικό για προγραμματισμό)