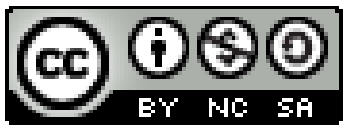
ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

ΑΝΟΙΚΤΑ
ΑΚΑΔΗΜΑΙΚΑ
ΜΑΘΗΜΑΤΑ

Αντικειμενοστρεφής Προγραμματισμός

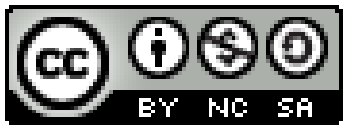
Ενότητα 12: Είσοδος από & Έξοδος σε Ρεύματα & Αρχεία

Γρηγόρης Τσουμάκας, Επικ. Καθηγητής
Τμήμα Πληροφορικής



Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.



Χρηματοδότηση

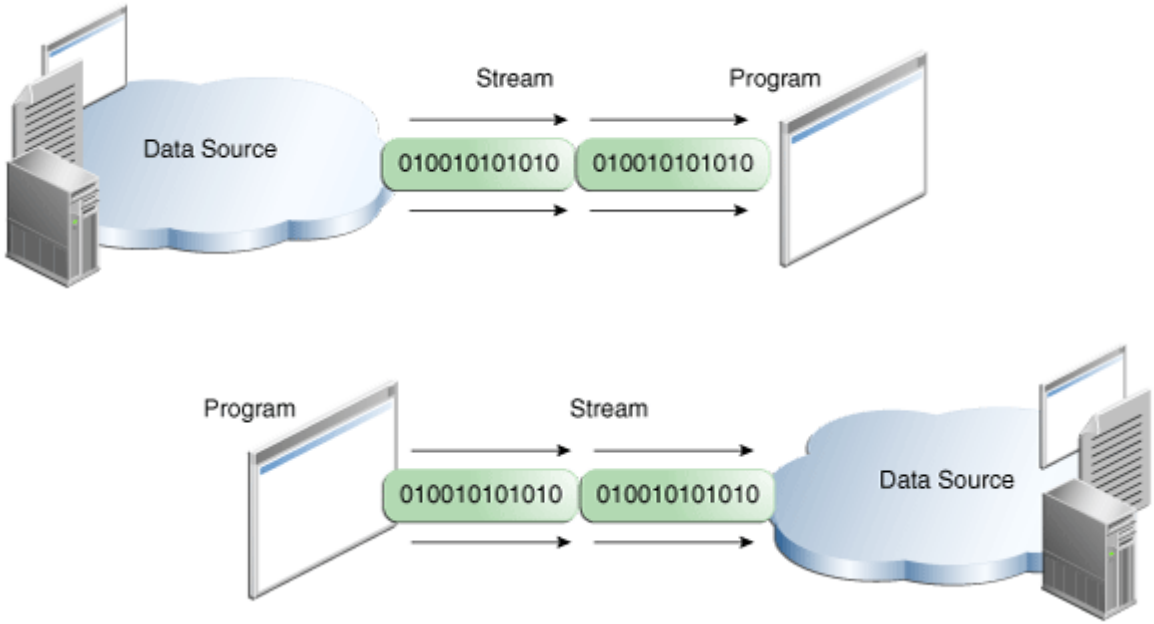
- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «Ανοικτά Ακαδημαϊκά Μαθήματα στο Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Είσοδος από & Έξοδος σε Ρεύματα & Αρχεία

Τα παραδείγματα κώδικα που χρησιμοποιούνται σε κάποιες από τις ακόλουθες διαφάνειες μπορούν να βρεθούν στον παρακάτω σύνδεσμο:
http://users.auth.gr/greg/oop.zip

Ρεύμα (Stream)

- Αφαιρετική έννοια για κάτι το οποίο παράγει (είσοδος) ή καταναλώνει (έξοδος) δεδομένα.



Είδη Ρευμάτων

- Ρεύματα byte.
  - Είσοδος και έξοδος πληροφορίας ανά 8 bit.
  - Χαμηλού επιπέδου είσοδος & έξοδος, στην οποία στηρίζονται οι υπόλοιπες κλάσεις εισόδου/εξόδου.
  - Απόφευγουμε την απευθείας χρήση τους.
- Ρεύματα χαρακτήρων.
  - Είσοδος και έξοδος χαρακτήρων.
  - Μετατροπή από/προς Unicode (UTF-8) μέσα στη Java (16 bit) προς/από το τοπικό σύνολο χαρακτήρων συστήματος (8 bit).

Ρεύματα Byte

- Abstract κλάσεις *InputStream*, *OutputStream.*
  - *read*(), επιστρέφει το επόμενο byte (ως int), ή -1 αν έφτασε το τέλος του ρεύματος.
  - *write(int* b), γράφει ένα byte (ως int).
  - *close*(), κλείσιμο ρεύματος για ελευθέρωση πόρων.
- Αρχεία στο δίσκο.
  - Κλάσεις *FileInputStream*, *FileOutputStream.*
  - Το μονοπάτι προς το αρχείο δίνεται ως παράμετρος στον κατασκευαστή.

Παράδειγμα

```
public static void main(String[] a) throws IOException
{
  FileInputStream in = new FileInputStream("i.doc");
  FileOutputStream out = new FileOutputStream("o.doc");
  int c;
  while ((c = in.read()) != -1) {
    out.write(c);
  }
  in.close();
  out.close();
}
```

CopyBytes.java

Ρεύματα Χαρακτήρων

- Abstract κλάσεις *Reader*, *Writer.*
  - *read*(), διαβάζει τον επόμενο χαρακτήρα και τον επιστρέφει ως int, ή -1 αν έφτασε στο τέλος.
  - *write(int* b), γράφει ένα χαρακτήρα (μέσω int),
  - *close(), κλείσιμο ρεύματος για ελευθέρωση πόρων.*
  - *write(String s)*, γράφει μια συμβολοσειρά.
  - *write(char[] array)*, γράφει πίνακα χαρακτήρων.
- Αρχεία στο δίσκο με τις *FileReader*, *FileWriter.*
  - Το μονοπάτι προς το αρχείο δίνεται ως παράμετρος στον κατασκευαστή.

Παράδειγμα (1/4)

```
try {
   FileWriter writer = new FileWriter("file.txt");
   writer.write("Γειά σου δίσκε!\n");
   writer.close();
} catch (IOException e) {
   System.out.println("Παρουσιάστηκε πρόβλημα");
}
```

Το αρχείο θα παραμείνει ανοιχτό σε περίπτωση σφάλματος κατά την εγγραφή

WritingText1.java

Παράδειγμα (2/4)

```
try {
  FileWriter writer = new FileWriter("file.txt");
  writer.write("Γειά σου δίσκε!\n");
} catch (IOException e) {
  System.out.println("Παρουσιάστηκε πρόβλημα");
} finally {
  writer.close();

}
```

Δεν μεταγλωττίζεται καθώς και η close() μπορεί να πετάξει IOException

Παράδειγμα (3/4)

```
try {
  FileWriter writer = new FileWriter("file.txt");
  writer.write("Γειά σου δίσκε!\n");
} catch (IOException e) {
  System.out.println("Παρουσιάστηκε πρόβλημα");
} finally {
  if (writer != null) {
    try {
      writer.close();
    } catch (IOException e) {
      System.out.println("Παρουσιάστηκε πρόβλημα");
    }
  }

}
```

WritingText2.java

Παράδειγμα (4/4)

```
public static void main(String[] args) {

   try {
     write("file.txt");
   } catch (IOException e) {
     System.out.println("Παρουσιάστηκε πρόβλημα");
   }
}
-----------------------------------------------------
public static void write(String f) throws IOException {
   FileWriter writer = null;
   try {
     writer = new FileWriter(filename);
   } finally {
     if (writer != null)
       writer.close();
   }
}
```

WritingText3.java

Try με Πόρους

- Αυτόματο κλείσιμο πόρου.
  - Δυνατότητα που προστέθηκε στη Java 7.
- Πως λειτουργεί.
  - Άνοιγμα πόρου αμέσως μετά το *try* μέσα σε παρενθέσεις.
  - Θα κλείσει αυτόματα στο τέλος του try χωρίς να χρειαστεί να κάνουμε κάτι.

Παράδειγμα

```
try (FileWriter w = new FileWriter("file.txt")) {
  w.write("Γεια σου δίσκε!\n");
} catch (IOException e) {
  System.out.println("Παρουσιάστηκε πρόβλημα");
}
```

WritingTextTry.java

Ενδιάμεση Μνήμη (1/2)

- Είσοδος/έξοδος χωρίς ενδιάμεση μνήμη.
  - Το λειτουργικό σύστημα πρέπει να διαχειριστεί την κάθε κλήση για είσοδο και έξοδο ξεχωριστά.
  - Σπατάλη πόρων.
- Είσοδος/έξοδος με ενδιάμεση μνήμη.
  - Χώρος στη μνήμη που γεμίζει/αδειάζει με μία μόνο κλήση στο λειτουργικό για είσοδο/έξοδο.
  - Καλύτερη διαχείριση πόρων.

Ενδιάμεση Μνήμη (2/2)

- Κλάσεις για bytes και χαρακτήρες.
  - BufferedInputStream, BufferedOutputStream.
  - BufferedReader, BufferedWriter.
- Αρχικοποίηση.
  - Αντικείμενο κλάσης χωρίς ενδιάμεση μνήμη ως παράμετρος στον κατασκευαστή.
  - Προαιρετικά το μέγεθος buffer ως $2^{η}$ παράμετρος.
- Ανάγνωση γραμμής κειμένου.
  - Με τη μέθοδο *readLine()* της *BufferedReader.*

WritingTextBuffered.java

Παράδειγμα

```
public static void main(String[] args) throws IOException
{
  try (BufferedReader in = new BufferedReader(
                           new FileReader("in.txt"));
       BufferedWriter out = new BufferedWriter(
                            new FileWriter("out.txt"));)
  {
    String l;
    while ((l = in.readLine()) != null) {
      out.write(l + "\n");
    }
  }
}
```

CopyLines1.java

Μορφοποίηση με PrintWriter

- Αρχικοποίηση.
  - Αντικείμενο κλάσης *FileWriter* ή *BufferedFileWriter* ως παράμετρος στον κατασκευαστή.
- Μέθοδοι μορφοποίησης εξόδου.
  - print, println, format, printf.

Παράδειγμα

```
public static void main(String[] args) throws IOException
{
  try (BufferedReader in = new BufferedReader(
                           new FileReader("in.txt"));
       BufferedWriter out = new PrintWriter(
                            new BufferedWriter(
                            new FileWriter("out.txt")));)
  {
    String l;
    while ((l = in.readLine()) != null) {
      out.println(l);
    }
  }
}
```

CopyLines2.java

Ρεύματα Δεδομένων

- Είσοδος και έξοδος δυαδικών δεδομένων.
  - Θεμελιώδεις τύποι, String ως δυαδικά δεδομένα.
- Κλάσεις.
  - *DataInputStream*, *DataOutputStream.*
  - Αντικείμενο τύπου *OutputStream* ως παράμετρος στον κατασκευαστή.
- Μέθοδοι.
  - writeInt(), writeDouble(), writeUTF(), …
  - readInt(), readDouble(), readUTF(), …
  - Η ανάγνωση γίνεται μέχρι να προκύψει EOFException.

Data.java

Ρεύματα Αντικειμένων (1/3)

- Serialization.
  - Μετατροπή αντικειμένου σε μια σειρά από byte με στόχο την είσοδο/έξοδο του από/σε ένα ρεύμα.
- Διασύνδεση Serializable.
  - Δεν ορίζει μεθόδους, απλά δηλώνει σημασιολογικά ότι επιτρέπεται το serialization.
- Για να γίνει serialize ένα αντικείμενο πρέπει,
  - Η κλάση του να υλοποιεί τη διεπαφή Serializable.
  - Να περιέχει μόνο θεμελιώδεις τύπους ή και κλάσεις που επίσης υλοποιούν τη διεπαφή αυτή.

Ρεύματα Αντικειμένων (2/3)

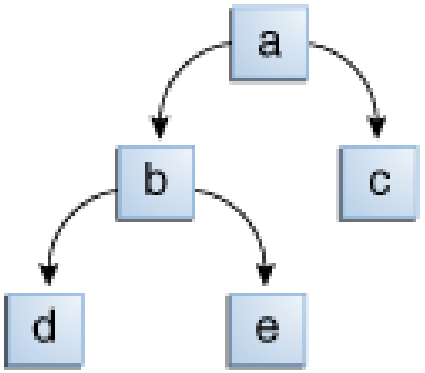
- Κλάσεις που χρησιμοποιούμε.
  - ObjectInputStream, ObjectOutputStream.
- Αρχικοποίηση.
  - Αντικείμενο τύπου *OutputStream* ως παράμετρος στον κατασκευαστή.
- Μέθοδοι.
  - *writeObject().*
  - *readObject()*, η οποία απαιτεί type casting.

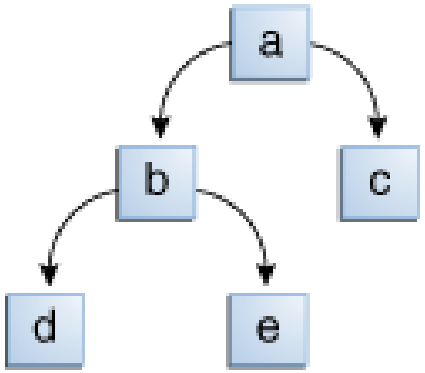
PersisentTime.java, FlattenTime.java, InflateTime.java

Ρεύματα Αντικειμένων (3/3)

- transient.
  - Λέξη κλειδί που μπαίνει εμπρός από πεδία για να δηλώσει ότι δεν θέλουμε την αποθήκευσή τους.
- serialVersionUID.
  - Πεδίο που καθορίζει την έκδοση της κλάσης, προκειμένου να διατηρείται η συμβατότητα με τα αποθηκεύμενα αντικείμενα.

Πολύπλοκα Αντικείμενα





- Το serialization χρησιμοποιείται πολλές φορές για τη "βαθιά" αντιγραφή αντικειμένων.

Complex.java, Person.java, Home.java

Πολλαπλές Εγγραφές

- Έστω ότι γράφουμε πολλές φορές το ίδιο αντικείμενο στο ίδιο ρεύμα.
- Στην πράξη το γράφουμε μόνο μία φορά και η δεύτερη είναι απλή αναφορά στο ίδιο αντικείμενο.

MultipleObjects.java

Αρχεία Τυχαίας Προσπέλασης (1/2)

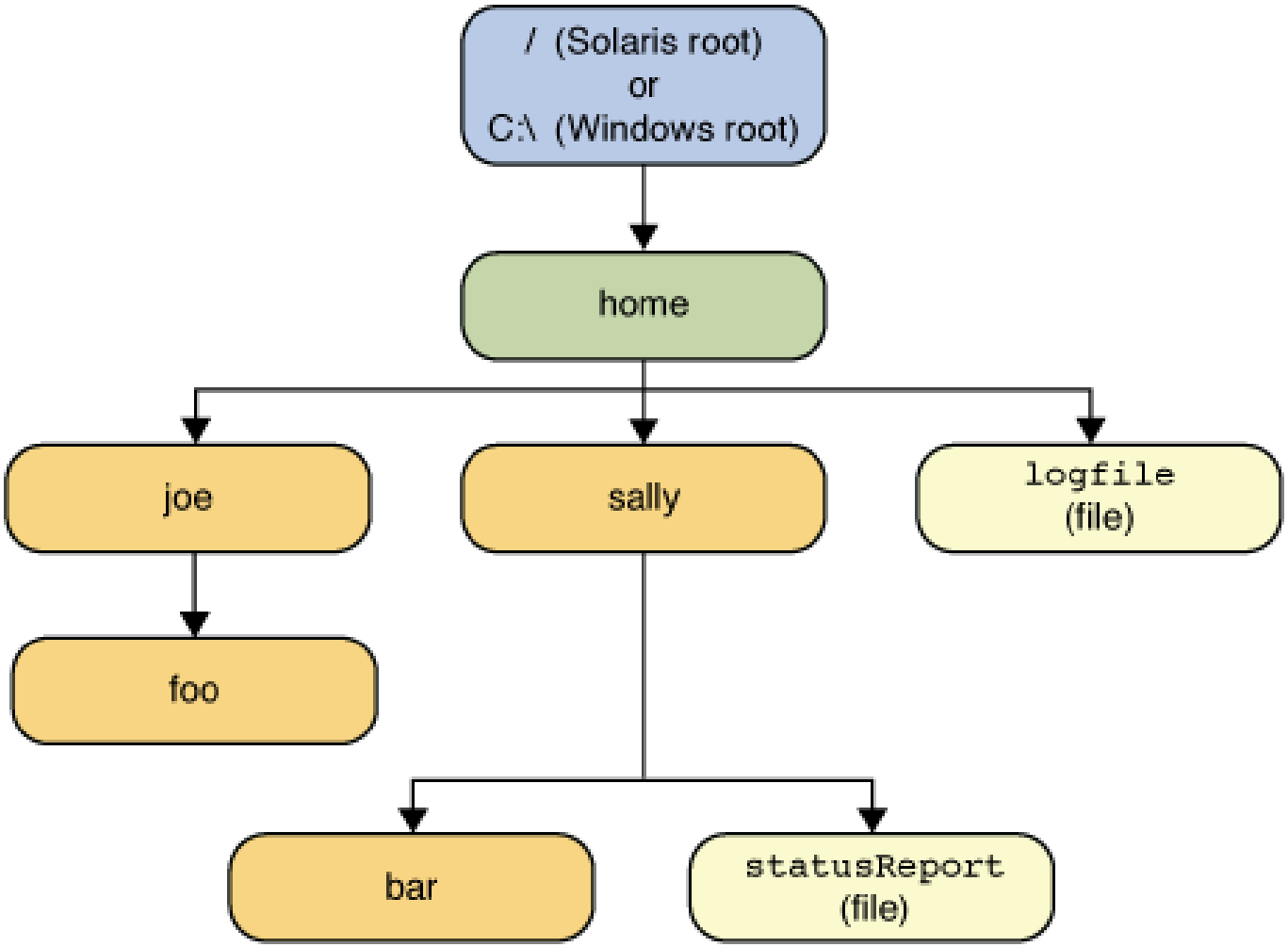
- Γιατί τα χρειαζόμαστε;
  - Για εφαρμογές όπου διαβάζουμε ή γράφουμε συγκεκριμένες εγγραφές και όχι όλο το αρχείο.
- Αρχικοποίηση.
  - RandomAccessFile r;
  - r = new RandomAccessFile("random.dat", "r");
  - r = new RandomAccessFile("random.dat", "rw");
- Μετακίνηση (σε bytes) από την αρχή του αρχείου.
  - long newPos = 100;
  - r.seek(newPos);

Αρχεία Τυχαίας Προσπέλασης (2/2)

- Μέθοδοι για εγγραφή και ανάγνωση byte.
  - int read(), int read(byte[] b).
  - write(int b), write(byte[] b).
- Υλοποιεί τη διεπαφή DataInput, DataOutput.
  - writeInt(), writeDouble(), writeUTF().
  - readInt(), readDouble(), readUTF().

RandomAccessDemo.java

Μονοπάτια


/ (Solaris root)
or
C:\ (Windows root)
home
joe
sally
logfile
(file)
foo
bar
statusReport
(file)

Η Κλάση Path

- Απόλυτο ή σχετικό μονοπάτι αρχείου ή φακέλου σε ένα σύστημα αρχείων.
  - Path p1 = Paths.get("c:\\input.txt");
  - Path p2 = Paths.get("../examples");
- Χρησιμότητα.
  - Όχι για άνοιγμα ή επεξεργασία αρχείων.
  - Για ανάκτηση πληροφοριών σχετικά με τα αρχεία.
  - Για διαγραφή και μετονομασία αρχείων, δημιουργία φακέλων.
- Εξαρτάται από το λειτουργικό σύστημα.
  - Περιέχει στοιχεία, όπως οι χαρακτήρες διαχωρισμού φακέλων και διαδρομών.

Πληροφορίες Μονοπατιών

- Έστω εκτέλεση του.

```
Path a = Paths.get("C:\\home\\joe\\foo"); // windows.

Path a = Paths.get("/home/joe/foo"); // linux.
```

Μέθοδος	Windows	Linux
a.toString()	c:\home\joe\foo	/home/joe/foo
a.getFileName()	foo	foo
a.getName(0)	home	home
a.getNameCount()	3	3
a.subPath(0,2)	home\joe	home/joe
a.getParent()	c:\home\joe	/home/joe
a.getRoot()	c:\	/

Έλεγχος Ύπαρξης

- exists().
  - Το path υπάρχει αν επιστρέψει true.
- notExists().
  - Το path δεν υπάρχει αν επιστρέψει true.
- Άγνωστη κατάσταση αρχείου.
  - Αν και τα δύο επιστρέψουν false.
  - Μπορεί να συμβεί όταν το πρόγραμμα δεν έχει πρόσβαση στο αρχείο/φάκελο.

Μέθοδοι Διαχείρισης

- Files.delete(Path p).
  - Διαγραφή αρχείου ή φακέλου (αν είναι άδειος).
- Files.createDirectory(Path p).
  - Δημιουργία φακέλου.
- Files.createDirectories(Path p)
  - Δημιουργία φακέλου και όλων όσων αναφέρονται στη διαδρομή και δεν υπάρχουν.
- Files.move(Path source, Path target).
  - Μετακίνηση αρχείου ή φακέλου.
- Files.copy(Path source, Path target).
  - Αντιγραφή αρχείου ή φακέλου.

Μέθοδοι Πληροφόρησης

- Files.size(Path p).
- Files.isDirectory(Path p).
- Files.getLastModifiedTime(Path p).
- Περιεχόμενα καταλόγου.
  - DirectoryStream<Path> stream = Files.newDirectoryStream(Path p);
- Φάκελοι ρίζας του συστήματος.
  - Iterable<Path> dirs = FileSystems.getDefault().getRootDirectories();

FileStuff.java

ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

ΑΝΟΙΚΤΑ
ΑΚΑΔΗΜΑΙΚΑ
ΜΑΘΗΜΑΤΑ

Τέλος Ενότητας

Επεξεργασία: Εμμανουήλ Ρήγας

Θεσσαλονίκη, Εαρινό Εξάμηνο 2013-2014

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης