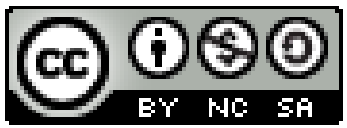
ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

ΑΝΟΙΚΤΑ
ΑΚΑΔΗΜΑΙΚΑ
ΜΑΘΗΜΑΤΑ

Αντικειμενοστρεφής Προγραμματισμός

Ενότητα 14: Γενικός Κώδικας (Generics)

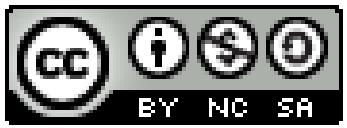
Γρηγόρης Τσουμάκας, Επικ. Καθηγητής
Τμήμα Πληροφορικής



Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

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.

ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

ΑΝΟΙΚΤΑ
ΑΚΑΔΗΜΑΙΚΑ
ΜΑΘΗΜΑΤΑ

Γενικός Κώδικας (Generics)

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

Εισαγωγή (1/2)

- Γενικός κώδικας.
  - Επιτρέπει σε τύπους αναφοράς (κλάσεις και διεπαφές) να είναι παράμετροι κατά τον ορισμό κλάσεων, διεπαφών και μεθόδων.
- Παραδείγματα γενικών κλάσεων.
  - ArrayList<Student>, HashMap<Integer, Student>.
- Παραδείγματα γενικών διεπαφών.
  - List<Student>, Map<Integer, Student>.
  - Comparable<Student>.

Εισαγωγή (2/2)

- Ορισμός γενικών τύπων.
  - Μεταβλητές που αντιστοιχούν στους τύπους που θα δοθούν ως παράμετροι δηλώνονται εντός γωνιακών αγκύλων μετά το όνομα του τύπου.
  - Π.χ. public class ArrayList<E>.
  - Π.χ. public class HashMap<K, V>.
- Συμβάσεις στα ονόματα.
  - (T)ype, S,U,V κτλ = 2ος, 3ος, 4ος τύπος.
  - (E)lement, (K)ey, (V)alue, (N)umber.

Πλεονεκτήματα

- Υλοποίηση γενικών δομών δεδομένων και αλγορίθμων.
- Δεν θα μπορούσαμε να χρησιμοποιήσουμε απλά την Object και κληρονομικότητα;
  - Καλύτερος έλεγχος τύπου.
  - Αποφυγή μετατροπής τύπων.

Δοχείο Ενός Αντικειμένου (1/4)

```
public interface Container1 {
    boolean put(Object object);
    Object get();
}

public interface Container2<T> {
    boolean put(T object);
    T get();
}
```

Δοχείο Ενός Αντικειμένου (2/4)

```
public class Box1 implements Container1 {
    private Object object;

    public boolean put(Object anObject) {
        if (anObject != null && object == null) {
            object = anObject;
            return true;
        } else
            return false;
    }

    public Object get() {
        Object temp = object;
        object = null;
        return temp;
    }
}
```

Δοχείο Ενός Αντικειμένου (3/4)

```
public class Box2<T> implements Container2<T> {
    private ~~Object~~ T object;

    public boolean put(~~Object~~ T anObject) {
        if (anObject != null && object == null) {
            object = anObject;
            return true;
        } else
            return false;
    }

    public ~~Object~~ T get() {
        ~~Object~~ T temp = object;
        object = null;
        return temp;
    }
}
```

Δοχείο Ενός Αντικειμένου (4/4)

```
public static void main(String[] args) {
    Box1 box1 = new Box1();
    box1.put("laptop");
    String content = (String) box1.get();
    box1.put(19);
    content = (String) box1.get();

    Box2<String> box2 = new Box2<>();

    box2.put("laptop");

    //box2.put(19);

    content = box2.get();

}
```

Γενική Κλάση με 2 Παραμέτρους

```
public class Pair<K, V> {

  private K key;
  private V value;

  public Pair(K key, V value) {
    this.key = key;
    this.value = value;
  }

  public static void main(String[] args) {
    Pair<Integer, String> p;
    p = new Pair<>(123, "Γιώργος Παππάς");

  }

}
```

Γενικές Μέθοδοι

- Τύποι ως παράμετροι στον ορισμό μεθόδων.
  - Δεν είναι ανάγκη να ανήκουν σε γενικό τύπο.
  - Μπορεί να είναι και στατικές μέθοδοι.
- Ορισμός.
  - Οι τύποι δίνονται πριν τον τύπο επιστροφής της μεθόδου.

Παράδειγμα

```
public class Util1 {

  public static <T> void conc(List<T> a, List<T> b) {
    for (T e : b)
      a.add(e);
  }

  public static void main(String[] args) {
    List<Integer> list1 = new ArrayList<>();
    list1.add(1); list1.add(2); list1.add(3);
    List<Integer> list2 = new ArrayList<>();
    list2.add(4); list2.add(5); list2.add(6);
    conc(list1, list2);
    for (Integer i : list1)
      System.out.println(i);
  }
}
```

Διαγραφή (Erasure)

- Τι γίνεται κατά τη μεταγλώττιση και εκτέλεση;
  - Ένα και μοναδικό ή πολλά και διαφορετικά αρχεία bytecode (.class) ανάλογα με τους τύπους που δίνονται ως παράμετροι;
- Ένα αρχείο bytecode.
  - Οι τύποι «διαγράφονται» και αντικαθίστανται από τον java.lang.Object
  - Οι τύποι που προκύπτουν καλούνται ακατέργαστοι (raw) και είναι διαθέσιμοι.

Παράδειγμα

```
public static void main(String[] args) {
  Box2<String> box = new Box2<>();

  if (box instanceof Box2<String>)
    System.out.println("true");

  if (box instanceof Box2)
    System.out.println("true");

  // ακατέργαστος τύπος
  Box2 anotherBox = new Box2();
  anotherBox.put("laptop");
  String content = (String) anotherBox.get();
  System.out.println(content);

}
```

Σφάλμα μεταγλώττισης

Erasure.java

Περιορισμός του Τύπου

- Περιορισμός του τύπου.
  - Μετά τον τύπο ακολουθεί η λέξη κλειδί *extends* και ακολουθεί κλάση ή διεπαφή.
  - Μπορεί να γίνει χρήση μεθόδων του τύπου που καθορίζεται ως περιορισμός.

Παράδειγμα 1

```
public class NumberList<T extends Number>
             extends ArrayList<T> {


  public static void main(String[] args) {
    //NumberList<String> listStr = new NumberList<>();
    NumberList<Integer> list = new NumberList<>();
    list.add(5);
    list.add(6);
  }
}
```

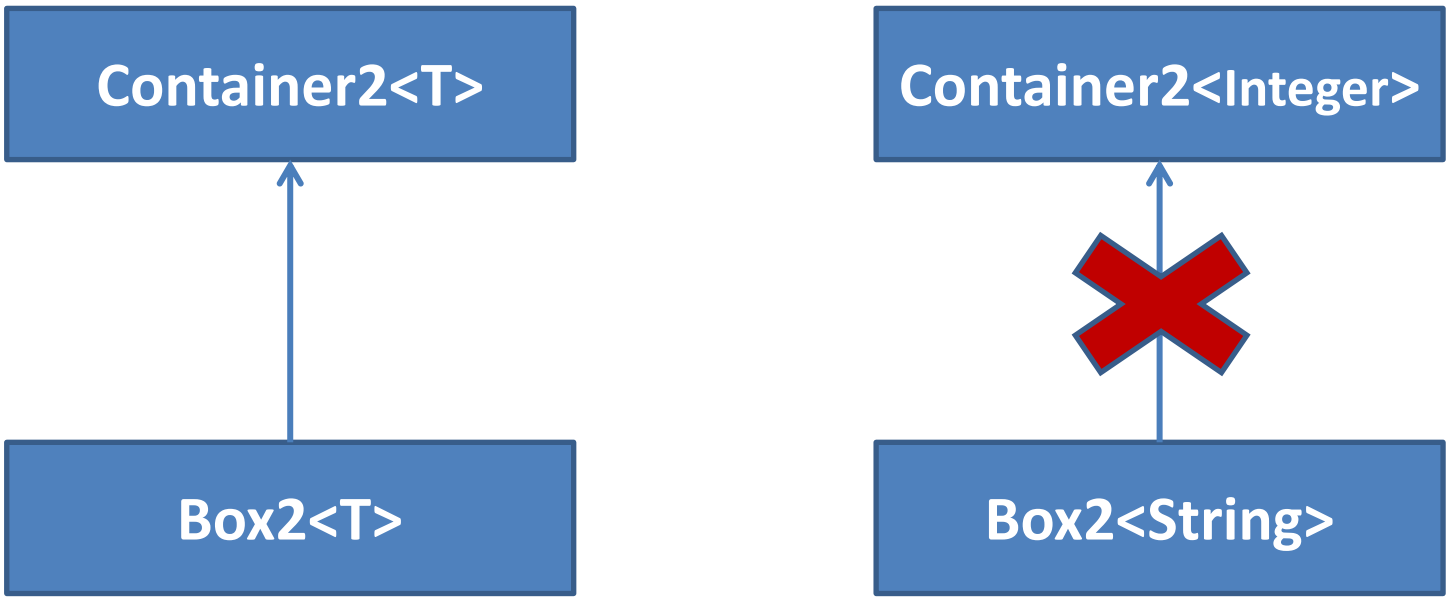
Παράδειγμα 2

```
public class NumBox<T extends Number> extends Box2<T>
{
  public boolean isNegative() {
    return object.doubleValue() < 0;
  }

  public static void main(String[] args) {
    NumberBox<Integer> boxInteger = new NumBox<>();
    boxInteger.put(11);
    System.out.println(boxInteger.isNegative());

    NumberBox<Double> boxDouble = new NumberBox<>();
    boxDouble.put(-4.0);
    System.out.println(boxDouble.isNegative());

    //NumberBox<String> boxStr = new NumberBox<>();

  }
}
```

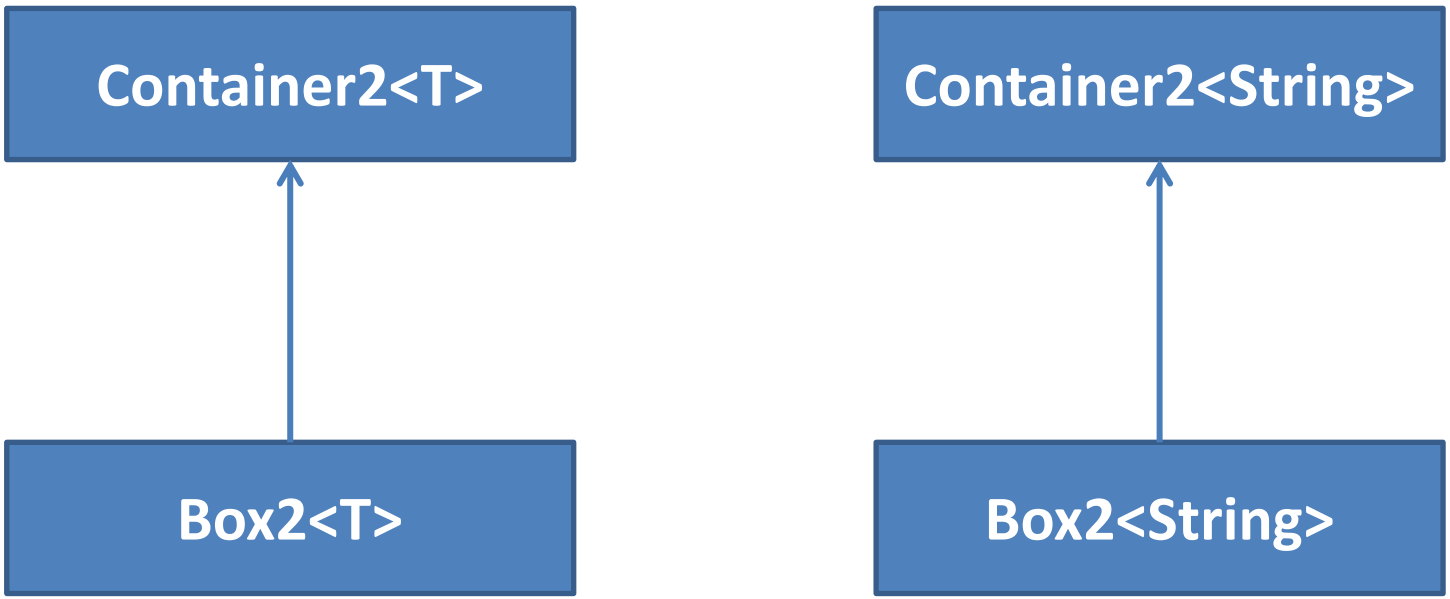
Παράδειγμα 3

```
public class Util2 {
  public static <T extends Comparable<T>>
         int countGreaterThan(List<T> l, T elem) {
    int counter = 0;
    for (T e : l)
      if (e.compareTo(elem) > 0)
        counter++;
    return counter;
  }
  public static void main(String[] args) {
    List<String> list = new ArrayList<>();
    list.add("greg"); list.add("nick");
    int x = countGreaterThan(list, "jim");
    System.out.println(x);

  }
}
```

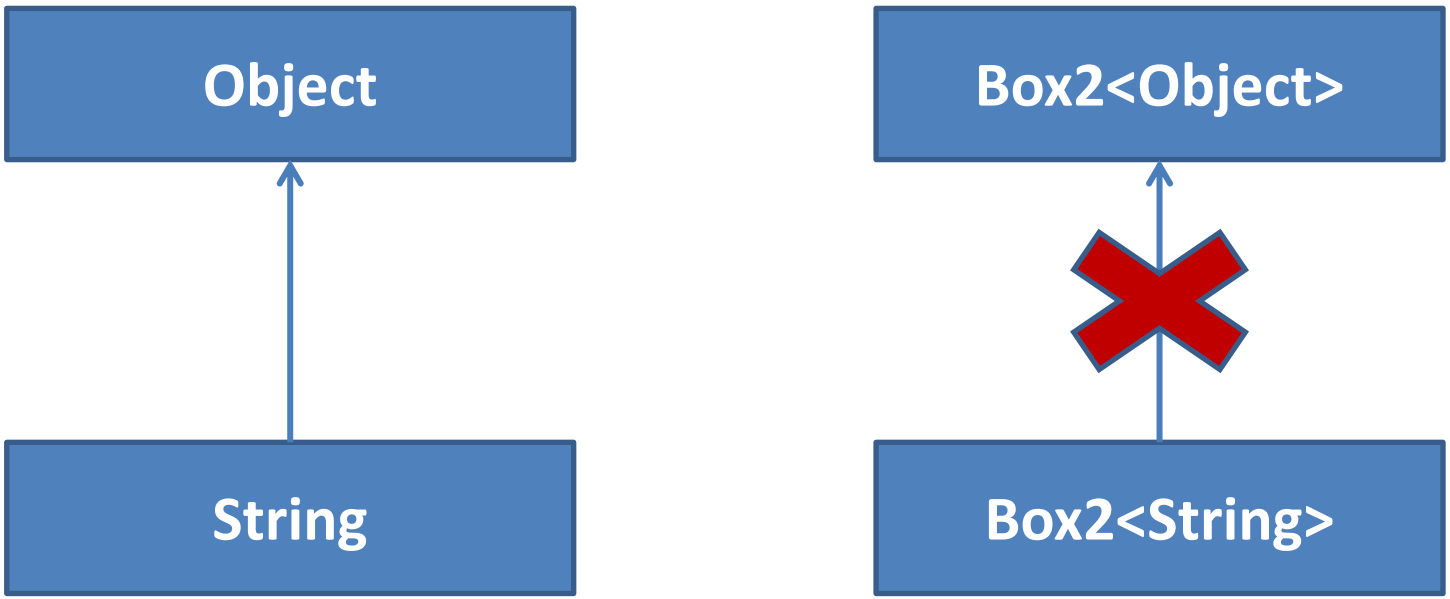
Σχέσεις Μεταξύ Τύπων (1/4)



Σχέσεις Μεταξύ Τύπων (2/4)



Σχέσεις Μεταξύ Τύπων (3/4)



Σχέσεις Μεταξύ Τύπων (4/4)

- Box2<String> και Container2<Integer>
  - Δεν υπάρχει σχέση.
- Box2<String> και Container2<String>
  - Σχέση υποτύπου και υπερτύπου.
- Box2<String> και Box2<Object>
  - Δεν υπάρχει σχέση.

Relationships.java

Μπαλαντέρ (1/2)

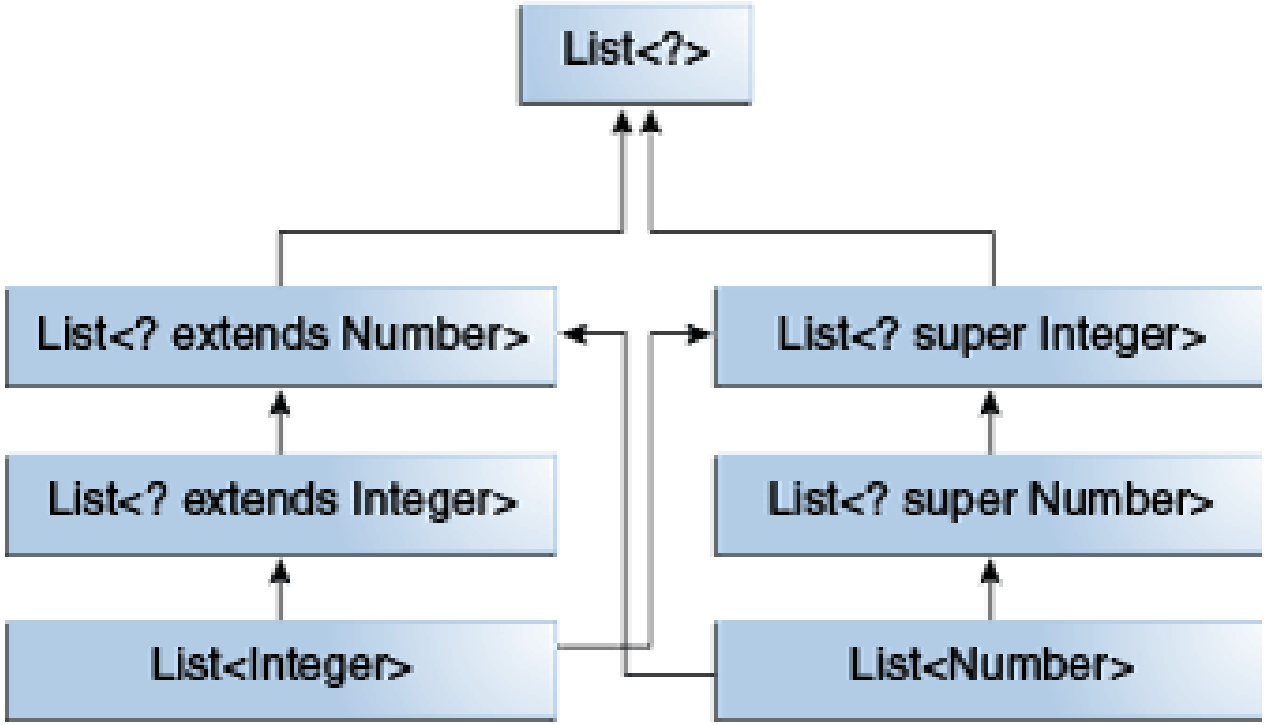
- <?>
  - Δηλώνει οποιονδήποτε τύπο.
  - Αποτελεί το αντίστοιχο του Object για παραμέτρους και λύνει το πρόβλημα με τον πολυμορφισμό στους τύπους-παραμέτρους.

Παράδειγμα

```
public class Wildcards {

   public static void main(String[] args) {
      //Box2<Object> b1 = new Box2<String>();
      Box2<?> b2 = new Box2<String>();
   }
}
```

Μπαλαντέρ (2/2)

- <? extends *Τύπος*>, <? super *Τύπος*>
  - Περιορισμός του τύπου από πάνω, κάτω.



Παράδειγμα

```
public class Util3 {

  public static double sum(List<? extends Number> l) {
    double s = 0;
    for (Number n : list)
      s += n.doubleValue();
    return s;
  }

  public static void main(String[] args) {

    List<Integer> list = new ArrayList<>();
    list.add(5); list.add(6); list.add(7);
    double x = sumOfList(list);
    System.out.printf("%.0f%n", x);
  }
}
```

Παράδειγμα: Collections.sort

```
public static <T extends Comparable<? super T>>
                void sort(List<T> list) {
  Object[] a = list.toArray();
  Arrays.sort(a);
  ListIterator<T> i = list.listIterator();
  for (int j=0; j<a.length; j++) {
    i.next();
    i.set((T)a[j]);
  }
}
```

Γιατί χρησιμοποιείται περιορισμός από κάτω (super) εντός του γενικού τύπου της Comparable;

Ασκήσεις

- Διαχείριση προσωπικού πανεπιστημίου.
  - Κλάσεις Person, Student, Professor.
  - Κλάση Personnel με δυνατότητα προσθήκης μιας λίστας φοιτητών ή μιας λίστας καθηγητών ή γενικότερα μιας λίστας ατόμων.
- Δομές δεδομένων.
  - Στοίβα με μεθόδους push, pop.
  - Ουρά με μεθόδους push, pop και pushMany.
  - Δέντρο με μεθόδους searchBreadth, searchDepth.

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