

# Προγραμματισμός II

## Πρακτικές δόμησης

Διομήδης Σπινέλλης  
Τμήμα Διοικητικής Επιστήμης και Τεχνολογίας  
Οικονομικό Πανεπιστήμιο Αθηνών

dds@aueb.gr  
<http://www.dmst.aueb.gr/dds>  
@CoolSWEng

2024-10-23

### Δόμηση

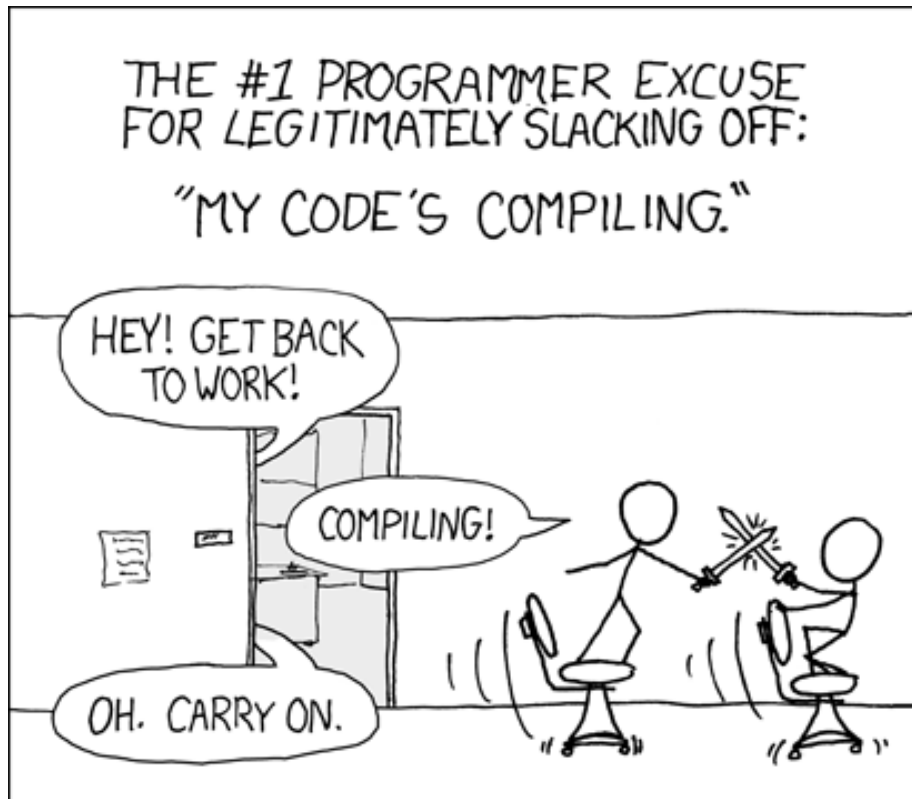


Figure 1: Δόμηση

(XKCD — BY NC 2.5)

## Περιεχόμενα

- Δόμηση
  - Ant
  - Maven
  - Gradle
- Στατική ανάλυση

## Διεργασία δόμησης

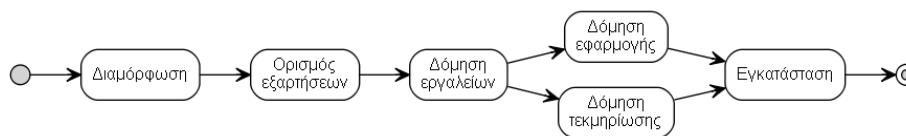


Figure 2: Τυπική διεργασία δόμησης

## Εξαρτήσεις

### Δόμηση με το εργαλείο Ant

- Η διαδικασία εκφράζεται σε XML
- Αρχείο build.xml
- Ορισμός μεταβλητών
- Στόχοι και εξαρτήσεις
- Ο προκαθορισμένος στόχος ορίζεται στην ετικέτα project
- Οι εργασίες είναι τμήμα ή άρθρωμα του ant

### Προκαθορισμένες εργασίες

- mkdir
- copy
- delete
- javac
- jar

### Παράδειγμα αρχείου ant

```
<project name="Jasper" default="deploy" basedir=". ">
  <!-- ===== Initialize Property Values ===== - -
->

  <!-- See "build.properties.sample" in the top level directory for all -
->
```

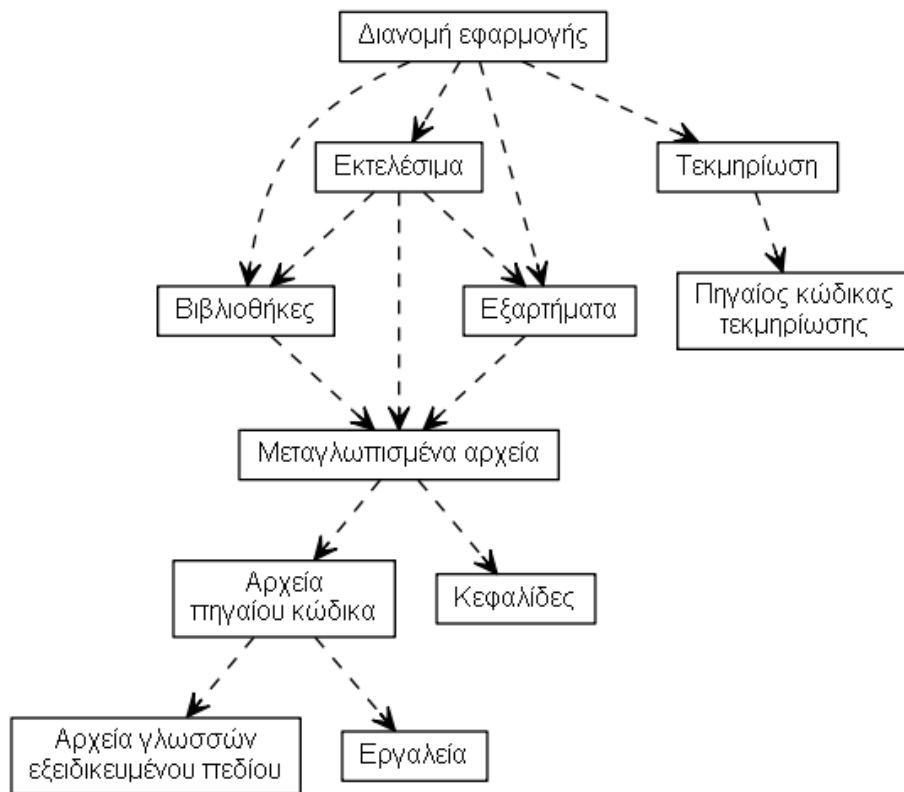


Figure 3: Παράδειγμα εξαρτήσεων

```

<!-- property values you must customize for successful building!!! -
->
<property file="build.properties"/>
<property file="../build.properties"/>
<property file="${user.home}/build.properties"/>

<!-- Build Defaults -->
<property name="build.compiler" value="classic"/>
<property name="copy.crimson.jar" value="../lib/crimson.jar"/>

<!-- ===== BUILD: Create Directories ===== -
->
<target name="build-prepare">

    <available classname="junit.framework.TestCase" property="junit.present" />

    <mkdir dir="${jasper.build}"/>
    <mkdir dir="${jasper.build}/jasper"/>
    <mkdir dir="${jasper.build}/lib"/>

</target>

<!-- ===== BUILD: Copy Static Files ===== -
->
<target name="build-static" depends="build-prepare">

    <!-- Executable Commands -->
    <copy todir="${jasper.build}/bin">
        <fileset dir="src/bin" />
    </copy>
    <fixcrlf srcdir="${jasper.build}/bin" includes="*.sh" eol="lf"/>
    <fixcrlf srcdir="${jasper.build}/bin" includes="*.bat" eol="crlf"/>
    <chmod perm="+x" file="${jasper.build}/bin/jasper.sh"/>
    <chmod perm="+x" file="${jasper.build}/bin/jspc.sh"/>

    <!-- Common Extensions -->
    <copy todir="${jasper.build}/jasper" file="${copy.crimson.jar}"/>
    <copy todir="${jasper.build}/jasper" file="${copy.jaxp.jar}"/>

</target>

<!-- ===== BUILD: Compile Server Components ===== -
->

```

```

<target name="build-main" depends="build-static">

  <!-- Compile internal server components -->
  <javac srcdir="src/share" destdir="${jasper.build}/classes"
        debug="${compile.debug}" deprecation="${compile.deprecation}"
        optimize="${compile.optimize}"
        excludes="**/CVS/**">
    <classpath refid="jasper.classpath" />
  </javac>
</target>
</project>

```

## Το σύστημα Maven

- Η διαδικασία εκφράζεται σε XML
- Project Object Model (POM): συμβάσεις αντί για διαμόρφωση (convention over configuration)
- Αρχείο pom.xml
- Αυτόματη ανάκτηση εξαρτήσεων
- Η δόμηση γίνεται μέσω αρθρωμάτων με λογικές προκαθορισμένες τιμές
- Ο κύκλος ζωής κατευθύνει την εκτέλεση αρθρωμάτων

## Ο κύκλος ζωής του Maven

```

validate
initialize
generate-sources
process-sources
generate-resources
process-resources
compile
process-classes
generate-test-sources
process-test-sources
generate-test-resources
process-test-resources
test-compile
process-test-classes
test
prepare-package
package
pre-integration-test
integration-test
post-integration-test
verify
install

```

deploy

## Δομή καταλόγων έργων Maven

### Παράδειγμα αρχείου pom.xml

```
<project>
  <!-- model version is always 4.0.0 for Maven 2.x POMs -->
  <modelVersion>4.0.0</modelVersion>

  <!-- project coordinates, i.e. a group of values which
        uniquely identify this project -->
  <groupId>gr.aueb.dmst.aueb.dds</groupId>
  <artifactId>linked-list</artifactId>
  <version>1.0</version>

  <!-- Project properties. Make build platform-independent -
->
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>

  <!-- library dependencies -->
  <dependencies>
    <dependency>

      <!-- coordinates of the required library -->
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>

      <!-- this dependency is only used for running and compiling tests -
->
      <scope>test</scope>
    </dependency>
  </dependencies>

  <reporting>
    <plugins>

      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-checkstyle-plugin</artifactId>
        <version>2.13</version>
        <reportSets>
          <reportSet>
```

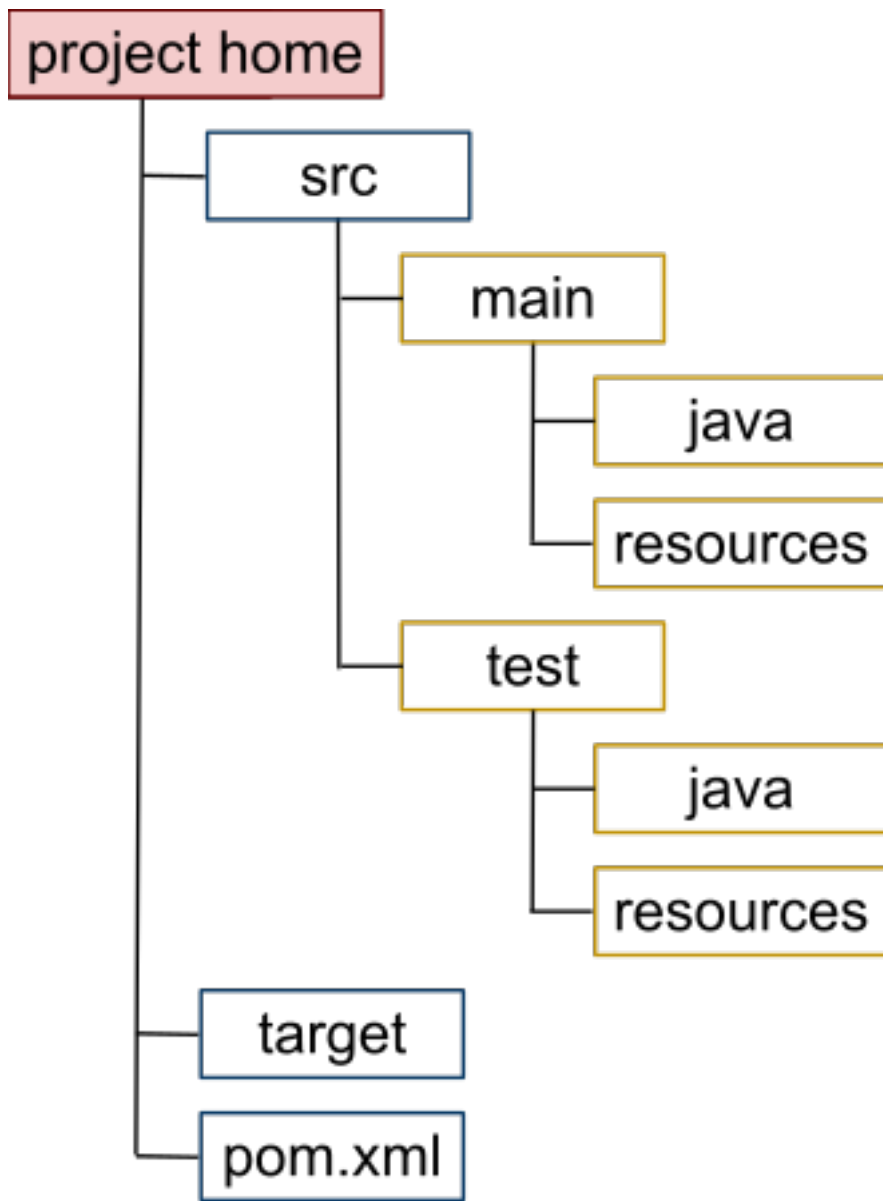


Figure 4: Δομή καταλόγων έργων Maven

```

        <reports>
          <report>checkstyle</report>
        </reports>
      </reportSet>
    </reportSets>
  </plugin>

  <plugin>
    <groupId>org.codehaus.mojo</groupId>
    <artifactId>spotbugs-maven-plugin</artifactId>
    <version>3.0.1-SNAPSHOT</version>
  </plugin>

</plugins>
</reporting>

</project>

```

## Το σύστημα Gradle

- Βασίζεται σε γλώσσα εξειδικευμένου πεδίου (DSL)
  - Groovy (γλώσσα JVM και υποστήριξη δυναμικών τύπων — `build.gradle`)
  - Kotlin (γλώσσα JVM με εφαρμογή στο Android — `build.gradle.kts`)
- Μικρά και ευνόητα αρχεία διαμόρφωσης
- Υποστηρίζει πολλές γλώσσες
- Αυτόματη αρχικοποίηση (`gradle init`)

## Παράδειγμα `build.gradle.kts`

```

plugins {
    // Apply the application plugin to add support for building a CLI application in Java
    application
}

repositories {
    // Use Maven Central for resolving dependencies.
    mavenCentral()
}

dependencies {
    // Use JUnit test framework.
    testImplementation("junit:junit:4.13.2")
}

```



```
application {  
    // Define the main class for the application.  
    mainClass.set("erp.App")  
}
```

## Αξιολόγηση Ant

### Πλεονεκτήματα:

- Ευελιξία
- Προσαρμοστικότητα
- Υποστήριξη για σταδιακές μεταγλωττίσεις
- Κατάλληλο για μικρά και απλά έργα

### Μειονεκτήματα:

- Χειροκίνητη διαχείριση εξαρτήσεων
- Λεπτομερές και πολύπλοκο XML αρχείο
- Έλλειψη προτυποποίησης

## Αξιολόγηση Maven

### Πλεονεκτήματα:

- Αυτόματη διαχείριση εξαρτήσεων
- Προτυποποίηση (convention over configuration)
- Προκαθορισμένος κύκλος ζωής έργου
- Πλούσιο οικοσύστημα πρόσθετων (plugins)
- Ιδανικό για μεγάλα έργα με πολλές εξαρτήσεις

### Μειονεκτήματα:

- Μικρότερη ευελιξία
- Απότομη καμπύλη εκμάθησης
- Αργή εκτέλεση

## Αξιολόγηση Gradle

### Πλεονεκτήματα:

- Γρήγορη δόμηση
- Διαχείριση εξαρτήσεων
- Ευελιξία
- Ιδανικό για μεγάλα έργα
- Ενσωμάτωση με CI/CD

### Μειονεκτήματα:

- Πολύπλοκη διαμόρφωση
- Μεγαλύτερη καμπύλη εκμάθησης
- Λιγότερη καθοδήγηση

### Στατική ανάλυση

- Αυτόματος έλεγχος του κώδικα
- Εντοπίζει:
  - λογικά λάθη
  - κενά ασφάλειας
  - λάθη στυλ
- Μπορεί να περιλαμβάνει συμβολική εκτέλεση
- Μερικά εργαλεία για κώδικα Java:
  - Checkstyle
  - PMD
  - SpotBugs

### Παράδειγμα από το εργαλείο checkstyle

```
<?xml version="1.0" encoding="UTF-8"?>
<checkstyle version="5.7">
<file name="C:\dds\pubs\courses\prog2gp\junit\src\main\java\gr\aub\dmst\dds\exampl
<error line="0" severity="error" message="Missing package-
info.java file." source="com.puppycrawl.tools.checkstyle.checks.javadoc.JavadocPack
<error line="3" severity="error" message="First sentence should end with a period." so
<error line="4" severity="error" message="Type Javadoc comment is missing an @param &l
<error line="5" severity="error" message="First sentence should end with a period." so
<error line="7" severity="error" message="First sentence should end with a period." so
<error line="10" severity="error" message="First sentence should end with a period." s
<error line="11" column="16" severity="error" message="Parameter v should be final." s
<error line="11" column="18" severity="error" message="Expected @param tag for &apos;
<error line="16" severity="error" message="First sentence should end with a period." s
<error line="17" severity="error" message="Expected an @return tag." source="com.pupp
<error line="17" column="5" severity="error" message="Method &apos;add&apos; is not d
needs to be abstract, final or empty." source="com.puppycrawl.tools.checkstyle.checks
<error line="17" column="31" severity="error" message="Parameter v should be final." s
<error line="17" column="33" severity="error" message="Expected @param tag for &apos;
<error line="19" column="15" severity="error" message="&apos;=&apos; is not preceded
<error line="19" column="16" severity="error" message="&apos;=&apos; is not followed
<error line="23" severity="error" message="First sentence should end with a period." s
<error line="24" severity="error" message="Expected an @return tag." source="com.pupp
<error line="24" column="5" severity="error" message="Method &apos;getHead&apos; is n
needs to be abstract, final or empty." source="com.puppycrawl.tools.checkstyle.checks
<error line="25" column="21" severity="error" message="&apos;;&apos; is preceded with
```

```

<error line="28" severity="error" message="First sentence should end with a period." s
<error line="29" severity="error" message="Expected an @return tag." source="com.pupp
<error line="29" column="5" severity="error" message="Method &apos;getTail&apos; is n
needs to be abstract, final or empty." source="com.pupppycrawl.tools.checkstyle.checks
<error line="33" severity="error" message="First sentence should end with a period." s
<error line="34" severity="error" message="Expected an @return tag." source="com.pupp
<error line="34" column="5" severity="error" message="Method &apos;toString&apos; is
needs to be abstract, final or empty." source="com.pupppycrawl.tools.checkstyle.checks
<error line="38" severity="error" message="&apos;if&apos; construct must use &apos;{}
<error line="40" severity="error" message="&apos;else&apos; construct must use &apos;
<error line="44" severity="error" message="First sentence should end with a period." s
<error line="45" severity="error" message="Expected an @return tag." source="com.pupp
<error line="45" column="5" severity="error" message="Method &apos;size&apos; is not
needs to be abstract, final or empty." source="com.pupppycrawl.tools.checkstyle.checks
<error line="47" severity="error" message="&apos;if&apos; construct must use &apos;{}
<error line="49" severity="error" message="&apos;else&apos; construct must use &apos;
<error line="53" severity="error" message="First sentence should end with a period." s
<error line="54" severity="error" message="Expected an @return tag." source="com.pupp
<error line="54" column="5" severity="error" message="Method &apos;contains&apos; is
needs to be abstract, final or empty." source="com.pupppycrawl.tools.checkstyle.checks
<error line="54" column="29" severity="error" message="Parameter e should be final." s
<error line="54" column="31" severity="error" message="Expected @param tag for &apos;
<error line="56" severity="error" message="&apos;for&apos; construct must use &apos;{
<error line="57" severity="error" message="&apos;if&apos; construct must use &apos;{}
<error line="62" severity="error" message="First sentence should end with a period." s
<error line="63" column="12" severity="error" message="&apos;public&apos; modifier ou
<error line="63" column="29" severity="error" message="Parameter args should be final
<error line="63" column="36" severity="error" message="Expected @param tag for &apos;
<error line="63" column="40" severity="error" message="Array brackets at illegal posi
<error line="67" column="25" severity="error" message="&apos;18&apos; is a magic numb
<error line="68" column="25" severity="error" message="&apos;45&apos; is a magic numb
<error line="70" severity="error" message="&apos;for&apos; construct must use &apos;{
<error line="70" column="29" severity="error" message="&apos;5&apos; is a magic numbe
<error line="71" column="33" severity="error" message="&apos;10&apos; is a magic numb
</file>
</checkstyle>

```

## Ο κώδικας πριν τις διορθώσεις

```

package gr.aueb.dmst.dds.example;

/** Implementation of a simple non-empty parametric linked list type */
public class LinkedList <E> {
    /** Node's value */
    private E value;

```

```

/** Next node */
private LinkedList <E> next;

/** Construct a list with a single element v */
LinkedList(E v) {
    value = v;
    next = null;
}

/** Return a list with element n added to it */
public LinkedList <E> add(E v) {
    LinkedList <E> n = new LinkedList <E>(v);
    n.next = this;
    return n;
}

/** Return the element at the list's head */
public E getHead() {
    return value;
}

/** Return the list's tail */
public LinkedList<E> getTail() {
    return next;
}

/** Return a string representation of the list */
public String toString() {
    String me = value.toString();

    /* Recursive implementation */
    if (next == null)
        return me;
    else
        return me + " -> " + next;
}

/** Return the number of elements in the list */
public int size() {
    /* Recursive implementation */
    if (next == null)
        return 1;
    else
        return next.size() + 1;
}

```

```

/** Return true if the specified element exists in the list */
public boolean contains(E e) {
    /* Iterative implementation */
    for (LinkedList <E> i = this; i != null; i = i.next)
        if (i.value.equals(e))
            return true;
    return false;
}

/** Test harness */
static public void main(String args[]) {
    LinkedList <Integer> ilst = new LinkedList <Integer>(0);

    ilst = ilst.add(1);
    ilst = ilst.add(18);
    ilst = ilst.add(45);

    for (int i = 0; i < 5; i++)
        ilst = ilst.add(i * 10);
    System.out.println(ilst);
}
}

```

### Ο κώδικας μετά τις διορθώσεις

```

package gr.aueb.dmst.dds.example;

/**
 * Implementation of a simple non-empty parametric linked list type.
 * @param <E> The type of the values stored in the list.
 */
public class LinkedList <E> {
    /** Node's value. */
    private E value;
    /** Next node. */
    private LinkedList <E> next;

    /**
     * Construct a list with a single element.
     * @param v The value of the list's first element.
     */
    LinkedList(final E v) {
        value = v;
        next = null;
    }
}

```

```

/** Add an element to the list.
 * @param v The value of the element to add.
 * @return A list with element v added to it.
 */
public final LinkedList <E> add(final E v) {
    LinkedList <E> n = new LinkedList <E>(v);
    n.next = this;
    return n;
}

/**
 * Obtain the element at the list's head.
 * @return The value of the first element of the list.
 */
public final E getHead() {
    return value;
}

/**
 * Obtain the list's tail.
 * @return The list starting from the second element.
 */
public final LinkedList<E> getTail() {
    return next;
}

/**
 * Obtain a string representation of the list.
 * @return The list as a series of elements connected with arrows.
 */
public final String toString() {
    String me = value.toString();

    /* Recursive implementation */
    if (next == null) {
        return me;
    } else {
        return me + " -> " + next;
    }
}

/**
 * Obtain the number of elements in the list.
 * @return The number of elements in the list.
 */

```

```

public final int size() {
    /* Recursive implementation */
    if (next == null) {
        return 1;
    } else {
        return next.size() + 1;
    }
}

/**
 * Check whether the specified element exists in the list.
 * @param e The element to search for in the list.
 * @return True if the specified element is in the list, false otherwise.
 */
public final boolean contains(final E e) {
    /* Iterative implementation */
    for (LinkedList<E> i = this; i != null; i = i.next) {
        if (i.value.equals(e)) {
            return true;
        }
    }
    return false;
}
}

```

### Άδεια διανομής

Εκτός αν αναφέρεται κάτι διαφορετικό, όλο το πρωτότυπο υλικό της σελίδας αυτής του οποίου δημιουργός είναι ο Διομήδης Σπινέλλης παρέχεται σύμφωνα με τους όρους της άδειας Creative Commons Αναφορά-Παρόμοια διανομή 3.0 Ελλάδα.

