Τεχνολογίες Εφαρμογών Διαδικτύου

Java Annotations
(Επισημάνσεις στον κώδικα)

Δρ. Ι. Χαμόδρακας
Μέλος του Εργαστηριακού Διδακτικού Προσωπικού
Τμήμα Πληροφορικής και Τηλεπικοινωνιών
Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Java Annotations

- Οι επισημάνσεις (annotations) στον κώδικα εισήχθησαν στην Java 5.
- Οι επισημάνσεις παρέχουν δεδομένα σχετικά με ένα πρόγραμμα: δεν αποτελούν τμήμα του προγράμματος και δεν επηρεάζουν άμεσα την εκτέλεση του κώδικα.
- Χρήση των annotations:
  - **Πληροφορίες για τον μεταγλωττιστή:** παρέχουν στο μεταγλωττιστή πληροφορίες για τον έλεγχο λαθών και την απόκρυψη των μηνυμάτων προειδοποίησης (warnings)
  - **Επεξεργασία κατά το χρόνο μεταγλώττισης / εφαρμογής (deployment):** χρησιμοποιούνται από εργαλεία λογισμικού για την παραγωγή κώδικα, αρχείων XML, κ.λπ.
  - **Επεξεργασία κατά το χρόνο εκτέλεσης**

Java Annotations: εφαρμογή και μορφή

- Τα annotations μπορούν να εφαρμοστούν σε δηλώσεις οποιουδήποτε είδους: κλάσεις, πεδία, constructors, μεθόδους, παραμέτρους, τοπικές μεταβλητές. Π.χ.

```
@Author(
   name = "Benjamin Franklin",
   date = "3/27/2003"
)
class MyClass() { }

@SuppressWarnings(value = "unchecked")
void myMethod() { }
```

- Μετά την εισαγωγή του annotation ακολουθούν ζεύγη στοιχείων και τιμών.

Java Annotations: μορφή

- Αν ένα annotation περιλαμβάνει μόνο ένα στοιχείο τότε το όνομά του μπορεί να παραλείπεται:

```
@SuppressWarnings("unchecked")
void myMethod() { }
```

- Αν κάποιο annotation δεν έχει στοιχεία τότε οι παρενθέσεις παραλείπονται:

```
@Override
void mySuperMethod() { }
```

Java Annotations: τεκμηρίωση

- Τα annotations μπορούν να χρησιμοποιηθούν αντί για σχόλια στον κώδικα.

```
public class Generation3List extends Generation2List {

    // Author: John Doe
     // Date: 3/17/2002
    // Current revision: 6
    // Last modified: 4/12/2004
    // By: Jane Doe
    // Reviewers: Alice, Bill

    // class code goes here

}
```

Java Annotations: τεκμηρίωση

- Για να εισάγουμε την ίδια τεκμηρίωση με annotations πρέπει καταρχάς να δημιουργήσουμε τον κατάλληλο τύπο annotation, του οποίου η δήλωση είναι παρόμοια με τη δήλωση interface. Στην πραγματικότητα τα annotations είναι interfaces ειδικού τύπου:

```
public @interface ClassPreamble {
   String author();
   String date();
   int currentRevision() default 1;
   String lastModified() default "N/A";
   String lastModifiedBy() default "N/A";
   // Note use of array
   String[] reviewers();
}
```

Java Annotations: τεκμηρίωση

- Αφού οριστεί ο τύπος annotation εντός αρχείου ClassPreamble.java όπου πρέπει να έχει δηλωθεί και το package. Μπορούμε εν συνεχεία να το χρησιμοποιήσουμε αν το κάνουμε import.

```
@ClassPreamble (
   author = "John Doe",
   date = "3/17/2002",
   currentRevision = 6,
   lastModified = "4/12/2004",
   lastModifiedBy = "Jane Doe",
   // Note array notation
   reviewers = {"Alice", "Bob", "Cindy"}
)
public class Generation3List extends Generation2List {

// class code goes here
}
```

Java Annotations: τεκμηρίωση

- Προκειμένου οι πληροφορίες που περιλαμβάνονται σε ένα annotation να εμφανιστούν στην τεκμηρίωση που παράγεται από το javadoc θα πρέπει ο ορισμός του τύπου @ClassPreamble να είναι ο ίδιος υπομνηματισμένος με το @Documented annotation.

```
 // import this to use @Documented
import java.lang.annotation.Documented;

@Documented
public @interface ClassPreamble {

   // Annotation element definitions

}
```

Java Annotations: χρήση από τον μεταγλωττιστή

- Παρέχονται τρεις τύποι annotation που παρέχονται από τις βιβλιοθήκες της Java και χρησιμοποιούνται από τον μεταγλωττιστή:

  `@Deprecated`: όταν μία κλάση, μέθοδος ή πεδίο υπομνηματίζονται ως deprecated ο μεταγλωττιστής παράγει κατάλληλη προειδοποίηση.

  `@Override`: όταν μία μέθοδος υπομνηματίζεται ως override ο μεταγλωττιστής ελέγχει αν πράγματι η μέθοδος μιας κλάσης ορίζει εκ νέου κάποια μέθοδο των υπερκλάσεων της. Σε αντίθετη περίπτωση δημιουργείται σφάλμα μεταγλώττισης. Έτσι αποφεύγονται λάθη.

  `@SuppressWarnings`: δίνει οδηγία στον μεταγλωττιστή να μην εμφανίσει προειδοποιήσεις. Υπάρχουν 2 είδη warnings: deprecation και unchecked όταν χρησιμοποιείται κώδικας που δεν χρησιμοποιεί Generics. Αν όλες οι προειδοποιήσεις πρέπει να αποκρυφτούν:

  ```
  @SuppressWarnings({"unchecked", "deprecation"})
  ```

Επεξεργασία Java Annotations

- Οι πιο προχωρημένες χρήσεις των annotations περιλαμβάνουν την ανάπτυξη επεξεργαστών που παράγουν αυτοματοποιημένα βοηθητικό κώδικα, XML αρχεία ρυθμίσεων, κλπ. Η Java παρέχει τον επεξεργαστή apt μετά την έκδοση 5.
- Για να μπορεί να χρησιμοποιείται ένα annotation κατά το χρόνο εκτέλεσης πρέπει να έχει υπομνηματιστεί ο τύπος του:

```
import java.lang.annotation.RetentionPolicy;

@Retention(RetentionPolicy.RUNTIME)
public @interface AnnotationForRuntime {

   // Elements that give information
   // for runtime processing
}
```

Παράδειγμα runtime επεξεργασίας Annotation (Wikipedia)

```
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
// This is the annotation to be processed
// Default for Target is all Java Elements
// Change retention policy to RUNTIME (default is CLASS)

@Retention(RetentionPolicy.RUNTIME)
public @interface TypeHeader
{
        // Default value specified for developer attribute
        String developer() default "Unknown";
        String lastModified();
        String [] teamMembers();
        int meaningOfLife();
}
```

Παράδειγμα runtime επεξεργασίας Annotation (Wikipedia)

```
// This is the annotation being applied to a class
@TypeHeader(developer = "Bob Bee",
lastModified = "2013-02-12",
teamMembers = { "Ann", "Dan", "Fran" },
meaningOfLife = 42)

public class SetCustomAnnotation
{ // Class contents go here }
```

Παράδειγμα runtime επεξεργασίας Annotation (Wikipedia)

```
// This is the example code that processes the annotation
import java.lang.annotation.Annotation;
import java.lang.reflect.AnnotatedElement;

public class UseCustomAnnotation {
    public static void main(String [] args) {
        Class<SetCustomAnnotation> classObject = SetCustomAnnotation.class;
        readAnnotation(classObject);
    }

    static void readAnnotation(AnnotatedElement element) {
        try {
            System.out.println("Annotation element values: \n");
            if (element.isAnnotationPresent(TypeHeader.class)) {
                // getAnnotation returns Annotation type
                Annotation singleAnnotation =
                        element.getAnnotation(TypeHeader.class);
```

Παράδειγμα runtime επεξεργασίας Annotation (Wikipedia)

```
            TypeHeader header = (TypeHeader) singleAnnotation;

            System.out.println("Developer: " + header.developer());
            System.out.println("Last Modified: " + header.lastModified());

            // teamMembers returned as String []
            System.out.print("Team members: ");
            for (String member : header.teamMembers())
                    System.out.print(member + ", ");
            System.out.print("\n");
            System.out.println("Meaning of Life: "+ header.meaningOfLife());
        }
    } catch (Exception exception) {
            exception.printStackTrace();
        }
    }
}
```