Java et les annotations

Cnam Paris
jean-michel Douin
version du 6 Octobre 2014

Notes de cours sur les annotations

Sommaire

- **Les prédéfinies**
- **Annotation pour une meilleure productivité**
  - **Annotations comme un nouveau langage**
- **Annotation installée par le programmeur**
- **Annotation et introspection**
- **Annotation et MVC**

Principale bibliographie

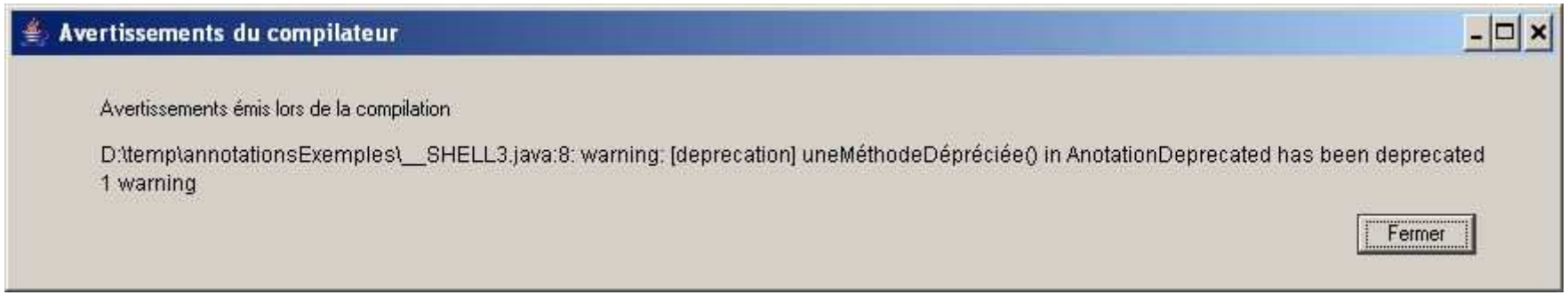
- Le guide
  - **http://java.sun.com/j2se/1.5.0/docs/guide/language/annotations.html**
  - http://java.sun.com/docs/books/tutorial/java/javaOO/annotations.html
- JavaWorld les indispensables
  - **http://www.javaworld.com/javaworld/jw-07-2004/jw-0719-tiger3_p.html**
  - **http://www.javaworld.com/javaworld/jw-08-2005/jw-0801-annotations_p.html**
  - **http://www.javaworld.com/javaworld/jw-10-2005/jw-1003-mvc_p.html**
- AOP et les annotations
  - **http://www.onjava.com/pub/a/onjava/2004/08/25/aoa.html?page=1**
- JUnit4
  - **http://www-128.ibm.com/developerworks/java/library/j-junit4.html**
  - **http://www.java201.com/resources/browse/14-2005.html**
- Stamps
  - **http://sourceforge.net/projects/stamps-mvc/**

Les prédéfinies

Interaction avec le compilateur

@Deprecated
le compilateur affichera un « warning »

```
public class AnnotationDeprecated{
   @Deprecated
   public void uneMéthodeDépréciée(){
   }
}
```



Les prédéfinies

@Override cette méthode devrait être masquée, redéfinie

```
public class AnnotationOverride extends Object{

   @Override
   public boolean equals(Object o){
      return …;
   }
}
```

- **Le compilateur vérifiera qu'il s'agit bien d'une méthode « masquée »**

Les prédéfinies

- **@SuppressWarning**

```
public class AnnotationSuppressWarning{

   @SuppressWarnings({"unchecked"})
   public …….
}
}
```

Voir javac –Xlint
all,deprecation,unchecked,fallthrough,path,serial,finally

Annotations ?

- **Au juste qu'est-ce que c'est ?**

- Des metadonnées du programme – 'décorant' le code
  - Un nouveau langage ?
  - Une prise en charge des traitements fastidieux ?
  - Une aide à la productivité ?

- Les annotations sont interprétées par,
  - les pré-compilateurs,
  - le compilateur,
  - le chargeur de classes et la JVM

Exemple cossu d'emblée

```
@Entity
public class Animal implements Serializable {

    @Column(name="animalName")
    String name;

    String kind;
    String weight;

    @ManyToOne
    Pavilion pavilion;

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;

    ...
```

Annotated as “Entity”

@Id denotes primary key

Génération automatique de fichiers de configuration, xml, script sql, ….
Productivité, nouveau langage, lisibilité, le fastidieux est supprimé

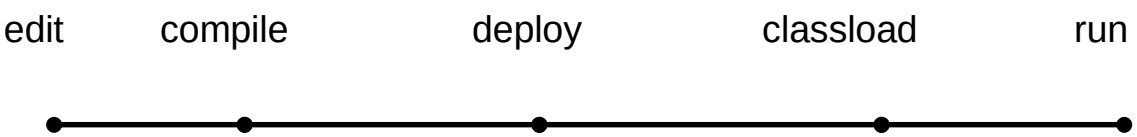
Un autre exemple, page79 livre d'antonio

- **Chapitre Object/Relational Mapping**

```
@Entity
public class Book{
   @Id
   @GeneratedValue(strategy=GenerationType.AUTO)
   private long Id;
   private Float price;
   …
   @ElementCollection(fetch = FetchType.LAZY)
   @CollectionTable(name= "tag")
   @Column(name= "Value")
   private ArrayList<String> tags;
…
```

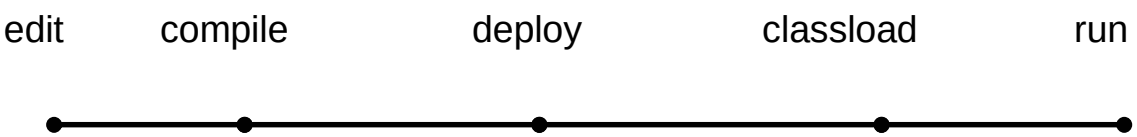
Productivité ?, nouveau langage ?, lisibilité ?, suppression du fastidieux , **discussion**

Annotations



- **Annotations**
  - **proposées par un fournisseur de logiciels, EJB3.0, JUnit4, Modern Jass, JML**

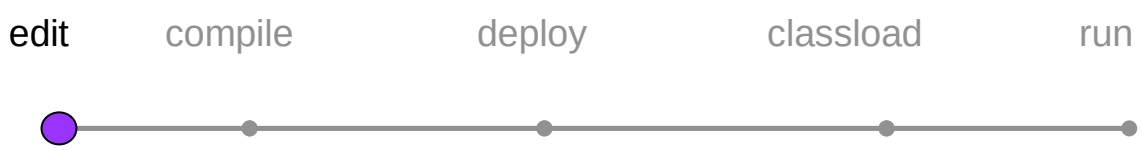
Comment et quand les annotations sont elles exploitées?



Plusieurs possibilités

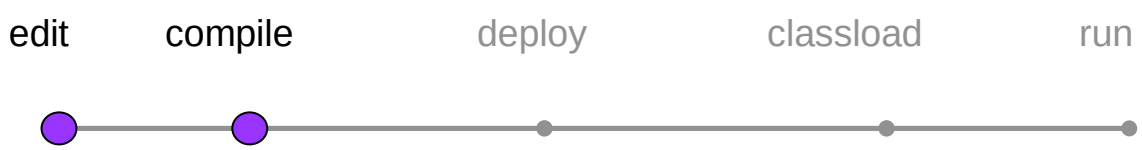
- **Lisibles par l'homme comme commentaire du code source**
- **Meta données d'aide à l' IDE**
- **Annotation processing API (JSR-269) pendant compilation**
- **Lisibles au chargement**
  - **(BCEL-*Byte Code Engineering Library* – une API Java permettant d'analyser, de créer et de manipuler des fichiers .class)**
- **Lisibles par introspection au runtime**

Annotations comme commentaires



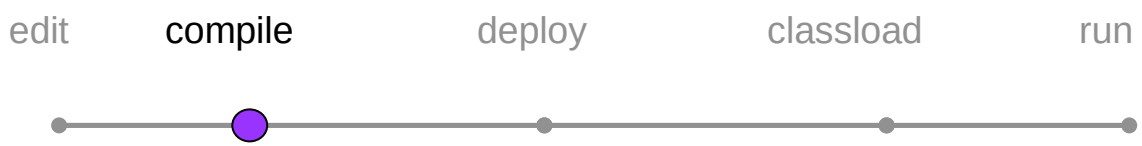
- **Annotations comme “commentaires typés”**
  - **@Deprecated, // déjà vue : les prédéfinies**
  - **versus “/* évitez d'utiliser cette méthode */”**

annotations et AGL



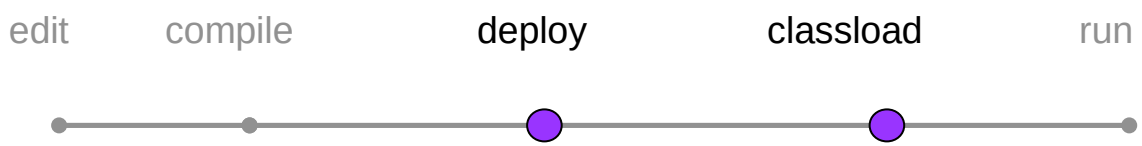
- **Renseigner l'AGL et/ou le compilateur**
  - **au delà des possibilités du langage**

Annotation traitées à la compilation



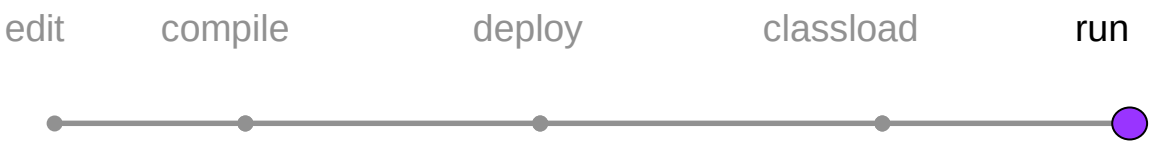
- **Peut générer du source**
  - **interfaces, objets, procédures, code, fichiers xml, ..**
  - **RPC stubs, LocalHome interfaces, deployment descriptors**
- **exemples: EJB**

annotations et fichiers .class



- **exemple:**
  - **@TransactionAttribute(REQUIRED)**
  - **Insère du code transactionnel autour du code de la procédure ainsi annotée.**
  - **Le patron décorateur, proxy …**

annotations et runtime (JUnit 4)

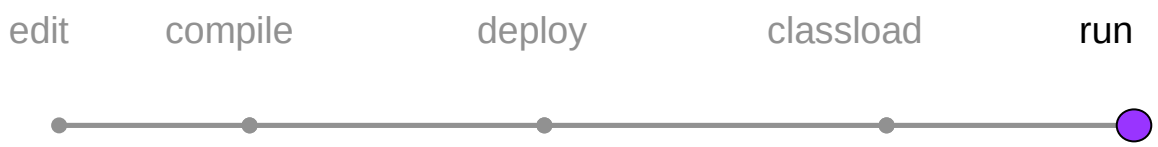


- **JUnit 4 test runner découvre les classes annotées, les instancie, execute les méthodes annotées**
- **Les classes de Test ne doivent plus hériter de TestCase, les méthodes peuvent ne plus commencer par test …**

```
@Test(expected = IndexOutOfBoundsException.class)
public void empty() {
    List l = new ArrayList<Object>();
    l.get(0);  // should throw exception
}
```

```
/* Avant en junit3 */
public void testEmpty() {
  try{
    List l = new ArrayList<Object>();
    l.get(0);  // should throw exception
    fail("should throw exception");
  }catch(Exception e){
    assertTrue(e instanceof IndexOutOfBoundsException);
  }}
```

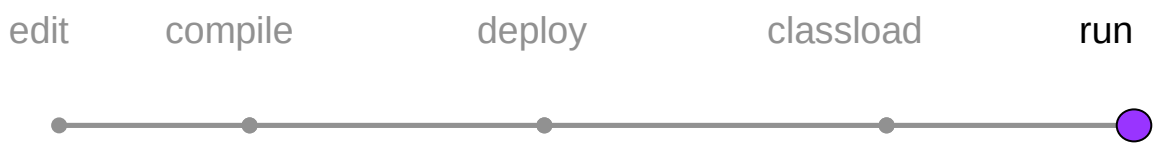
annotations et runtime (Hibernate)



- **Vérification de contraintes dans un conteneur**
  - **Exemple: Hibernate**

```
@NotEmpty
@Length(min = 2, max = 50)
public String getLastName () {
    return “”; // runtime exception!
}
```

annotations et runtime (ModernJass)



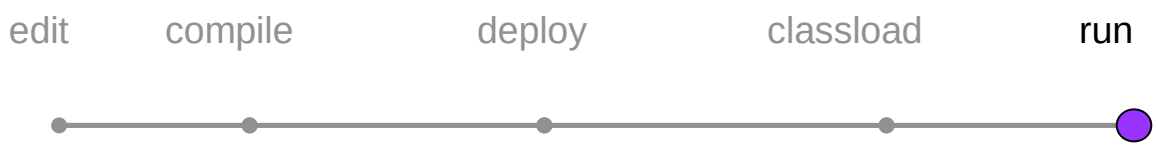
- **Vérification de contraintes à l'exécution,@Pre,@Post,@Invariant**

```
@Invariant("@ForAll(Object o : data;o !=null)" )
public class Liste {

public Object [ ] data ;

@Post ("@Old(size)+1==size")
public void add(Object o ){ . . . }}
```

annotations et runtime
(cofoja + moderne que ModernJass)



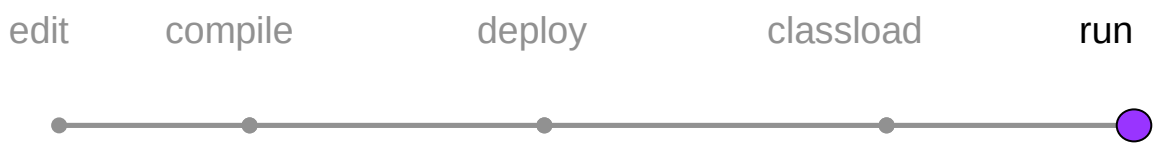
- **Vérification de contraintes à l'exécution,@Requires,@Ensures,@Invariant**

```
import com.google.java.contract.Requires;
public class Liste {
public Object [ ] data ;

@Requires ("o!=null")
public void add(Object o ){ . . . }}
```

- javac -cp .;cofoja-1.1-r150.jar -javaagent:cofoja-1.1-r150.jar contrats.Liste.java
- java -cp .;cofoja-1.1-r150.jar **-javaagent:cofoja-1.1-r150.jar** contrats.Main

annotations et runtime (Google Juice)



- **Injection de code à l'exécution**

```
@Inject public Cours(ServiceI service){
    this.service = service;

}
private @Inject ServiceI service;
```

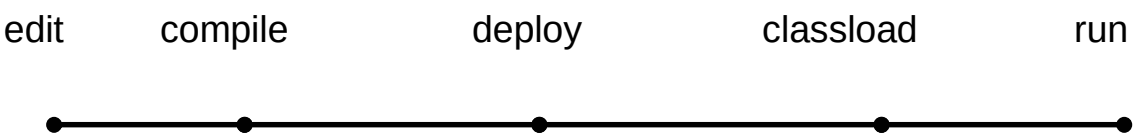
Guice3.0 D'autres annotations

- **D'autres annotations existent**
- http://code.google.com/docreader/#p=google-guice&s=google-guice&t=UsersGuide

- **@ImplementedBy**
- **@providedBy**
- **@Singleton**
- **@Provides**
- **@CheckedProvides**
- **@RequestScoped**
- **@SessionScoped**

  - **Chaque API proposerait-elle ses propres annotations ?**
    - **Serions nous envahis par les annotations ?**

Annotation installée par le « programmeur »



- **A quelles fins ?**
  - **Orientation outils, logiciels de base,**
  - **Langage maison ?**
  - **Perdre les programmeurs Java ? ou bien deviennent ils annotés ? (EJB, JPA)**

Un exemple, une nouvelle annotation

declaration →

```
@interface Author {
    String name();
    int year();
}
```

usage →

```
@Author(name =“Walter Harley”,year =2008)
 class MyClass {
}
```

- **Où annoter le code ?**
  - **@Target**
    - **Une méta-annotation**
- **Quand, à quelle phase est-elle utilisée ?**
  - **@Retention**
    - **Édition, compilation, exécution …**

Où annoter le code ?

```
@A class X {
    @A @B(“quux”) public void foo(@C x) { ... }
    @B private String s;
}
```

- **Syntaxiquement ce sont des "modifiers", comme “final”.**
- **Les Annotations peuvent être appliquées à toutes les déclarations: types (y compris enums et annotation), attributs, constructeurs, méthodes, paramètres, enum, packages, et variables locales.**

Annotations pour Annotation

- @Target
  - Quelles sont les structures du langage annotées
    - TYPE, FIELD, METHOD, PARAMETER, CONSTRUCTOR, …
    - c.f. java.lang.annotation Enum ElementType
- @Retention
  - A quel moment ?
    - SOURCE, CLASS, RUNTIME
    - c.f. java.lang.annotation Enum RetentionPolicy
- @Documented
  - L'annotation doit être documentée
- @Inherited
  - L'annotation est héritée

c.f. java.lang.annotation

Exemples

- **@ACompléter**
- **@DBTable**
- **@Pre, @Post**
- **@xxxxDeTypage**

Exemple « RetentionPolicy.SOURCE »

- **@ACompléter… mieux qu'un commentaire ?**
- **Hypothèse : vous avez des TP en Java à faire et des sources à compléter**
  - **(ce n'est qu'une hypothèse)**
  - **Une annotation @ACompléter est_elle la bienvenue ? …** @ADiscuter

```
@ACompléter
public class Liste{
  @ACompléter
  private int xxxxx;

  @ACompléter
  public Liste(@ACompléter("xxxx est à remplacer") int xxxx ){

  }

  @ACompléter({"attention à la division par zéro"," et à la division entière"})
  public int div(int x, int y){
    return 1;
  }
}
```

@ACompléter

```
import java.lang.annotation.Documented;
import java.lang.annotation.Target;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.ElementType;
import static java.lang.annotation.ElementType.*;

/** ACompléter pour les TP ..
 *
 */
@Documented
@Target({TYPE, FIELD, METHOD, PARAMETER, CONSTRUCTOR, LOCAL_VARIABLE})
@Retention(RetentionPolicy.SOURCE)
public @interface ACompléter{

     /** éventuellement avec un message... */
     String[] value() default {""};
}
```

Exemple 1) « RetentionPolicy.SOURCE »

- **Une annotation destinée au compilateur**
  - **Ou plutôt à une instance d'AnnotationProcessor exécutée par le compilateur depuis 1.6… (en 1.5, voir l'outil apt)**

- **Exemple génération d'un script SQL**
  - **Par exemple pour un Mapping Objet/relationnel … c.f. JPA**

```
@DBTable(name = "CUSTOMERS")
   public class Customers {
   ....
}
```

Extrait en partie de http://www.jroller.com/trickymar/entry/how_to_create_files_with

@DBTable

```
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Target(ElementType.TYPE)
@Retention(RetentionPolicy.SOURCE)
public @interface DBTable {
    String name();
}
```

DBTableProcessor…

```
@SupportedAnnotationTypes(value= {"DBTable"})
@SupportedSourceVersion(SourceVersion.RELEASE_6)
public class DBTableProcessor extends AbstractProcessor{

@Override
public boolean process(Set<? extends TypeElement> annotations,RoundEnvironment roundEnv){
    Boolean result = Boolean.TRUE;
    Messager consoleLogger = processingEnv.getMessager();
    try{
     for (Element element : roundEnv.getElementsAnnotatedWith(DBTable.class)){
        Filer filer = processingEnv.getFiler();
        FileObject o = filer.createResource(StandardLocation.SOURCE_OUTPUT,
                                    "generated",element.getSimpleName()+".sql");
        Writer w = o.openWriter();

        w.append("CREATE TABLE "+element.getAnnotation(DBTable.class).name());
        w.flush();
        w.close();
        consoleLogger.printMessage(Kind.NOTE,"File finished");
      }
    } catch (IOException ex){
      consoleLogger.printMessage(Kind.ERROR,ex.getMessage());
    }
    return result;
  }
}
```

DBTableProcessor les *import*

```
import java.io.IOException;
import java.io.Writer;
import java.util.Set;

import javax.annotation.processing.AbstractProcessor;
import javax.annotation.processing.Filer;
import javax.annotation.processing.Messager;
import javax.annotation.processing.RoundEnvironment;
import javax.annotation.processing.SupportedAnnotationTypes;
import javax.annotation.processing.SupportedSourceVersion;

import javax.lang.model.SourceVersion;
import javax.lang.model.element.Element;
import javax.lang.model.element.TypeElement;

import javax.tools.FileObject;
import javax.tools.StandardLocation;
import javax.tools.Diagnostic.Kind;
```

Le fichier ./generated/Customers.sql

- javac -processor DBTableProcessor Customers.java

- Le fichier ./generated/Customers.sql

```
CREATE TABLE CUSTOMERS
```

- Discussion
  - Nouveau langage ?
  - Annotations absconses ?
    - Trop puissantes ?
    - Indispensables aux développeurs ?

Exemple 2)

- **Usage de Pré-Post assertion**
  - **Présence d'annotation au « run-time »**
  - **Ici des contrats sur les constructeurs d'une Table**

```
public class Table implements TableI{
  public final static int TAILLE_PAR_DEFAUT = 10;
 private int[] table;

  @Pre("taille > 0 ")
  @Post("table != null && table.length==taille")
  public Table(int taille){
    table = new int[taille];
  }

  @Post("table != null && table.length==TAILLE_PAR_DEFAUT")
  public Table(){
    this(TAILLE_PAR_DEFAUT);
  }
```

@Pre, @Post « RetentionPolicy.RUNTIME »

```
import java.lang.annotation.ElementType;
import java.lang.annotation.Target;

import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;


@Retention(RetentionPolicy.RUNTIME)
@Target({ElementType.METHOD, ElementType.CONSTRUCTOR})
public @interface Pre {
  String value();
}


@Post (idem)
```

Introspection : accès au Annotations

```
public void testAnnotationDesConstructeurs() throws Exception{

  for(Constructor<?> cons : Table.class.getDeclaredConstructors()){
    for(Annotation a : cons.getAnnotations()){
      System.out.println(a);
    }
    System.out.println(cons.getName()+ "/" + cons.getParameterTypes().length);
  }

 }
```

```
@Pre(value=taille > 0 )
@Post(value=[table != null && table.length==taille])
Table/1
@Post(value=[table != null && table.length==TAILLE_PAR_DEFAUT])
Table/0
```

Annotations et plus

```
@Pre(value=taille > 0 )
@Post(value=[table != null && table.length==taille])
Table/1
@Post(value=[table != null && table.length==TAILLE_PAR_DEFAUT])
Table/0
```

- **Ce sont des String**
- **-> Instrumentation du bytecode, compilation à la volée, DynamicProxy …**

- **Usage de BCEL**
  - **–instrumentation du byte code**
  - **–compilation à la volée**

- **Usage d'un agent** voir la solution choisie par jass-modern
  - **–Java –agent**

@xxxxxDeTypage Google juice

- **Précision sur les paramètres à injecter :**

```
@Inject constructeur(@Service ServiceI service){
    this.service = service;
}
```

- **Dans l'Injecteur**
  - **bind(ServiceI.class).annotatedWith(Service.class).to(ListeEnCSV.class);**

- **L'annotation @Service décrite comme suit** (fichier Service.java)

```
@Retention(RetentionPolicy.RUNTIME)
@Target( {ElementType.PARAMETER} )
@BindingAnnotation   // annotation d'annotation
public @interface Service{
}
```

En savoir plus

- **MVC + annotations**
  - http://www.javaworld.com/javaworld/jw-10-2005/jw-1003-mvc_p.html
- **JUnit4**
  - http://www-128.ibm.com/developerworks/java/library/j-junit4.html
- **Modern Jass**
  - http://modernjass.sourceforge.net/
- **Google Juice**
  - http://code.google.com/p/google-guice/

Conclusion ?

- **Trouvée sur le web**
  - **Java annotations are a welcome unification and standardization of approaches to annotating code that language designers and implementors have been doing for decades**
  - **Annotations do not directly affect the semantics of a program**
  - **It is not hard to learn!**
- **Discussion**
  - **Sur ce qui est au dessus …**
  - **Fonctionnalités dans le « dos » du programmeur ?**
  - **Annotations absconses sont-elles possibles ?**
  - **Vers une meilleure productivité au détriment de la compréhension ?**
    - **Mais avons besoin de comprendre pour être plus productif …**
    - **Appels d'un framework**
  - **Un autre langage au dessus de Java ?**
    - **c.f. JPA, Mapping Objet/Relationnel**