

# Zápočtová úloha z předmětu KIV/ZSWI

## **UŽIVATELSKÁ PŘÍRUČKA**

pro Transformaci javovských anotací  
do technologií sémantického webu

16.5.2011

Tým: Hokishushodikukosida team

Členové:

Jan	Hrbek	hrbekj@students.zcu.cz
Jakub	Krauz	jakub.krauz@volny.cz
Miroslav	Mašek	miroslam@students.zcu.cz
Miroslav	Vozábal	vozami@hotmail.cz
Libor	Váchal	vachall@students.zcu.cz

## Obsah

1. ÚVOD .....	2
2.IMPLEMENTOVANÉ ANOTACE .....	2
3.SPUŠTĚNÍ .....	3

## 1.Úvod

Uživatelská příručka popisuje implementované anotace a jejich spuštění .  
Anotace slouží k zpřesnění informací o třídách, attributech atp.

## 2.Implementované anotace

**EquivalentClass, EquivalentProperty, sameAs,differentFrom, allDifferent, label, comment, seeAlso, isDefinedBy,Label, comment, seeAlso, isDefinedBy.**

### Annotation properties:

#### **Label**

Anotace slouží pro nastavení názvu anotovaného elementu ve tvaru, který je čitelný pro lidského uživatele.

#### **Comment**

Anotace slouží pro nastavení komentáře k anotovanému elementu..

#### **IsDefinedBy**

Anotace slouží pro nastavení odkazu na nadřazený element k anotovanému elementu, ve kterém je definován.

#### **SeeAlso**

Anotace slouží pro nastavení odkazu na související element k anotovanému elementu.

### **(In)Equality:**

#### **Equivalent Class**

Annotace informuje o stejnosti instancí dvou tříd

#### **Property**

Annotace slouží pro nastavení informace o vlastnostech anotovaného elementu.

#### **SameAs**

Annotace informuje o totožnosti dvou tříd.

#### **DifferentFrom**

Annotace popisuje odlišnost dvou a více elementů.

#### **AllDiferent**

Annotace popisuje odlišnost všech elementů.

Obrázek č.1 ukazuje implementaci anotací pro třídu , obrázek č.2 demonstruje implementaci pro atribut. Ukázky zdrojových kódů jsou pořízeny z testovací třídy Person.java. Funkčnost anotací lze ověřit ve výstupním .owl souboru, který se ukládá do složky projektu.

Obr. č.1

```

12  /**
13   * Person generated by hbm2java
14   */
15   //@Namespace("http://www.w3c.org/prostornik/")
16   //@RdfType("Osoba")
17   //@Transitive
18   //@Symmetric
19   //@Inverse("http://www.kiv.zcu.cz/eeg/Weather")
20   //@VersionInfo("Stara verze")
21   //@OnProperty("http://www.kiv.zcu.cz/#hasMaker-Person23")
22   //@AllValuesFrom("http://www.kiv.zcu.cz/#Person23")
23   //@Comment("http://www.kiv.zcu.cz/commenttridy")
24   //@SeeAlso ("http://www.kiv.zcu.cz/koukniTaky trida")
25   //@Label("http://www.kiv.zcu.cz/labelnadtridou")
26   //@Cardinality(4)
27   //@SomeValuesFrom("http://www.kiv.zcu.cz/nejakyHodnotyProPerson")
28   //@EquivalentClass("http://www.kiv.zcu.cz/ekvivalentnitridaktrideHuman")
29   //@EquivalentProperty("http://www.kiv.zcu.cz/ekvivalentnipropertyktrideHuman")
30   //@SameAs("http://www.kiv.zcu.cz/stejnatridajakoHuman")
31   //@DifferentFrom("http://www.kiv.zcu.cz/jina trida Person")
32   @AllDifferent({"http://www.kiv.zcu.cz/Ruzne tridy 1",
33                "http://www.kiv.zcu.cz/Ruzne tridy 2",
34                "http://www.kiv.zcu.cz/Ruzne tridy 3"})
35   public class Person implements java.io.Serializable {
36

```

Na obrázku č.1 je vidět použití anotace @AllDifferent pro třídu Person.java. Ostatní anotace jsou zakomentované.

Obr. č.2

```

39   //@Comment("http://www.kiv.zcu.cz/comment atributu")
40   //@Label("http://www.kiv.zcu.cz/givename")
41   //@SeeAlso("http://www.kiv.zcu.cz/koukniTaky atribut")
42   @IsDefinedBy("http://www.kiv.zcu.cz/isdefinedby")
43   private String givename;





```

Na obrázku č.2 je vidět použití anotace @IsDefinedBy pro atribut givename nacházející se ve třídě Person.java.

### 3.Spuštění

Aby bylo možné aplikaci spustit, musí uživatel nejprve připojit následující knihovny: JenaBeanExtension\_1. 0.jar, a všechny ostatní knihovny ze složky lib, které se nachází v adresáři projektu. Obrázek č.3 ukazuje strukturu adresáře projektu.

Obr. č.3.

	lib	18.5.2011 23:42	Složka souborů	
	source	18.5.2011 23:42	Složka souborů	
	ZSWI_ukazka	18.5.2011 23:42	Složka souborů	
	JenaBeanExtension_1.0	18.5.2011 23:42	Executable Jar File	121 kB

Jako je to ukázáno ve třídě Hlavní.java musí uživatel v klientské třídě vytvořit instanci objektů nad kterými jsou definovány anotace ze kterých chce výstupní .rdf, nebo .owl soubor. Vloží je do seznamu ArrayList a poté zvolí jednu ze dvou transformačních metod - metodu toOWL(), nebo toRDF().

OOMtoSemanticWeb.toRDF(ArrayList<Object>, String) -> pro transformaci do RDF

OOMtoSemanticWeb.toOWL(ArrayList<Object>, String) -> pro trans. do OWL

Vše je situováno na obrázku č.4.

Obr. č.4.

```
1 import java.util.*;
2
3
4 public class Hlavni {
5
6
7
8     public static void main(String[] args){
9         ArrayList<Object> osoby = new ArrayList<Object>();
10         Person franta = new Person(1, "Vokurka", 'M');
11         Person zdenek = new Person(2, "Slepička", 'M');
12
13         osoby.add((Object) franta);
14         osoby.add((Object) zdenek);
15         OOMtoSemanticWeb.toOWL(osoby, "www.kiv.zcu.cz/eeg");
16     }
17
18 }
19
```

Po spuštění proběhne transformace a vytvoří se výstupní .owl, nebo .rdf soubor.

