Java code for Arabic wordnet AWN

http://www.talp.upc.edu/index.php/technology/resources/multilingual-lexicons-and-machine-translation-resources/multilingual-lexicons/72-awn

awnMain.java is the main class

There two ways to use API Java code

1- **With Harkat/Tashkeel**
   In the main method in main class write
   AWN awn=new AWN (XMLfilepath,true);

2- **Without Harkat/Tashkeel**
   In this way the code will remove the 'Harkat/Tashkeel' from the source XML file and from the user input
   In the main method in main class write
   AWN awn=new AWN (XMLfilepath,false);

Where XMLfilepath is the path to the XML source in your computer

The code was tested on Windows 7 , netBeanse IDE 7.3 , JDK 7.

---

**A list of 35 methods can be used by the API are declared below.**
**The java code example of these methods are written by green color**

1. **Get_The_Spesfic_Link_Type(linkEnum linkType)**

   Return the list of a list of two items that share the specific link-type provided by the user
   List<List<String>> linktypeList =awn.Get_The_Spesfic_Link_Type(linkEnum.causes);

2. **Get_Item_Id_From_Name(String name_Of_Item)**

   Return a list of ItemId attribute by providing the name of Item attribute
   List<String> ItemID= awn.Get_Item_Id_From_Name("صَوْت");

3. **Get_Item_Id_From_Offset(String Offset_Of_Item)**

   Return ItemId attribute by providing offset of Item attribute
   String ItemId = awn.Get_Item_Id_From_Offset("106919436");

4. **Get_List_Of_Item_Id_From_POS(pos pos_Of_Item)**

Return a list of ItemId attribute of a specfic POS

```
List<String> List_of_item = awn.Get_List_Of_Item_Id_From_POS(pos.v);
```

5. **Get_List_Of_Links_Between_Two_Items(String itemId_1, String itemId_2)**

Return a list of relation between two items where ItemId_1 is "link1" and itemId_2 is "link2"

```
List<String> listOfRelation = awn.Get_List_Of_Links_Between_Two_Items("Eamiyq_a2AR",
    "saToHy_a1AR");
```

6. **Get_List_Of_Links_In_For_Item(String itemId)**

Return a list of Link for Item Id where itemId is the "link2" of the relation

```
 List<List<String>> listOfrealtion =
awn.Get_List_Of_Links_In_For_Item("dawolap_|soyawiy~ap_n1AR");
```

7. **Get_List_Of_Links_Out_For_Item(String itemId)**

Return a list of Link for Item Id where itemId is the "link1" of the relation

```
 List<List<String>> listOfrealtion =
    awn.Get_List_Of_Links_Out_For_Item("dawolap_|soyawiy~ap_n1AR");
```

8. **Get_List_Of_Links_Out_For_Item_With_Specfic_Relation(String itemId,linkEnum link )**

return a list of Link for Item Id where itemId in the *left* "link1" side of the specific relation

```
List<String> temp =
 awn.Get_List_Of_Links_Out_For_Item_With_Specfic_Relation("kymyA'_n1AR",
 linkEnum.category_term);
```

9. **Get_List_Of_Links_In_For_Item_With_Specfic_Relation(String itemId,linkEnum link )**

return a list of Link for Item Id where itemId in the *right* "link2" side of the specific relation

```
List<String> temp = awn.Get_List_Of_Links_In_For_Item_With_Specfic_Relation("njrAn_n1AR",
 linkEnum. has_instance);
```

10. **Get_Name_Of_Item_From_Item_Id(String itemId)**

Return a name attribute from specific item id

```
String name =awn.Get_Name_Of_Item_From_Item_Id("bi$akolK_nawoEiy~_r1AR");
```

11. **Get_Pos_Of_Item_From_Item_Id(String itemId)**

Return a POS attribute for a specific item

```
String currentPOS =awn.Get_Pos_Of_Item_From_Item_Id("baronaAmaj_n1AR");
```

12. **Get_Type_Of_Item_From_Item_Id(String itemId)**

Return a type of parm Item Id
String type =awn.Get_Type_Of_Item_From_Item_Id("baronaAmaj_n1AR");

13. **Get_Offset_Of_item_From_Item_Id(String itemId)**

Return an offset of a parm item id
String offset =awn. Get_Offset_Of_item_From_Item_Id("baronaAmaj_n1AR");

14. **Get_Number_Of_items()**
15. **Get_Number_Of_Words()**
16. **Get_Number_Of_Form()**

17. **Get_List_Word_Id_From_Value()**

Return a list of Word ID from a specific word-value
List<String> listWordId= awn.Get_List_Word_Id_From_Value("اِشْتَرَى");

18. **Get_List_Word_Id_From_Synset_ID(String SynsetId_Of_Word)**

Return a list of Word ID from a specific SynsetId Of Word
List<String> listWordId=awn.Get_List_Word_Id_From_Synset_ID("Ai$otaraY_v1AR");

19. **Get_Synset_ID_From_Word_Id(String wordId)**

Return a SynsetID from a specific Word ID
String SynsetID= awn.Get_Synset_ID_From_Word_Id("Ai$otaraY_1");

20. **Get_Word_Value_From_Word_Id(String wordId)**

Return a Word Value from a specific Word ID
String wordValue =awn.Get_Word_Value_From_Word_Id("Ai$otaraY_1");

21. **Get_List_Of_Forms_From_Word_Id(String wordId)**

Return a list of form from provided wordID for each element in the list has two entry 1-formValue and 2-formType
List<List<String> listOfForm=awn.Get_List_Of_Forms_From_Word_Id("Ai$otaraY_1");

22. **Get_List_Of_Words_From_Form_Value(String form_Value)**

Return a list of word from provided form_Value, for each element in the list has two entry 1-wordId and 2-wordType
List<List<String>> ListWordID=awn.Get_List_Of_Words_From_Form_Value("شري");

23. **Get_All_Item_Id()**
24. **Get_All_Word_Id()**
25. **Get_All_Form_Value()**

# IS-A relation tree

We build 61 *IS-A* relation (has_hyponym) tree from the AWN XML source to compute the semantic similarity between two concepts. A value of the node in a tree is "ItemId".
The name of the tree is value of the root node. The list of names of all trees is in "Tree.java"

**Note1:**

Some concepts can have more than one parent in the one tree depends on the XML source file, *for simplicity we only allow one parent to each concept.*

**Note2:**

Some concepts can be a node in more than one tree.

**Note3:**

Java doesn't accept Some characters (> , ~,*) in the predefined constants enumeration
So some characters are removed from enumeration constant, the correct tree name in the string value of the enumeration.
The list of 61 constants in enumeration tree is below

- **$aEara_v2AR("$aEara_v2AR"),**
- **aHodava_v1AR(">aHodava_v1AR"),**
- **aTolaqa(">aTolaqa_v1AR"),**
- **araAda(">araAda_v1AR"),**
- **arisotuqoraATiy(">arisotuqoraATiy~_n1AR"),**
- **axa(">axa*a_v1AR"),**
- **AisotawolY("AisotawolY_v1AR"),**
- **Aisotaxodama("Aisotaxodama_v1AR"),**
- **AistaloqaY("AistaloqaY_v1AR"),**
- **EaAlaja("EaAlaja_v1AR"),**
- **Eabara("Eab~ara_v1AR"),**
- **HaAlap("HaAlap_n1AR"),**
- **HaSala_EalaY("HaSala_EalaY_v1AR"),**
- **Hadav("Hadav_n2AR"),**
- **Hadada("Had~ada_v3AR"),**
- **Halala("Hal~ala_v1AR"),**
- **Haraka("Har~aka_v1AR"),**
- **Hasaba("Hasaba_v2AR"),**
- **Hawala("Haw~ala_v1AR"),**
- **SanaEa("SanaEa_v1AR"),**
- **ZaAhirap("ZaAhirap_n1AR"),**
- **badala("bad~ala_v2AR"),**
- **daEama("daEama_v1AR"),**
- **daxala("daxala_v1AR"),**
- **fahima("fahima_v1AR"),**
- **fakara("fak~ara_v2AR"),**
- **faraga("far~aga_v3AR"),**

- **fiEol("fiEol_n1AR"),**
- **gaAdara("gaAdara_v1AR"),**
- **jamaAEap("jamaAEap_n5AR"),**
- **kaAna_v1AR("kaAna_v1AR"),**
- **kaAna_v3AR("kaAna_v3AR"),**
- **kayonuwnap_n1AR("kayonuwnap_n1AR"),**
- **lAqaY_v2AR("lAqaY_v2AR"),**
- **lamasa_v1AR("lamasa_v1AR"),**
- **lamasa_v2AR("lamasa_v2AR"),**
- **malaka_v1AR("malaka_v1AR"),**
- **manaEa("manaEa_v1AR"),**
- **masaka_v2AR("masaka_v2AR"),**
- **masaka_v3AR("masaka_v3AR"),**
- **milok("milok_n1AR"),**
- **miyozap("miyozap_nafosiy~ap_n1AR"),**
- **na$ara_v2AR("na$ara_v2AR"),**
- **naAfasa_v1AR("naAfasa_v1AR"),**
- **nabiylap_n1AR("nabiylap_n1AR"),**
- **nabaha("nab~aha_v1AR"),**
- **naqal("naqala_v3AR"),**
- **qaAtala_v1AR("qaAtala_v1AR"),**
- **rabaTa("rabaTa_v1AR"),**
- **saAfara("saAfara_v1AR"),**
- **saAwaY("saAwaY_v3AR"),**
- **samaY("sam~aY_v2AR"),**
- **taakara("ta*ak~ara_v2AR"),**
- **taHak("taHak~ama_v1AR") ,**
- **taHaraka("taHar~aka_v2AR"),**
- **taSarafa("taSar~afa_v1AR"),**
- **tagayara("tagay~ara_v1AR"),**
- **tajoriy("tajoriyd_n1AR"),**
- **talafaZa("talaf~aZa_v1AR"),**
- **txlSv1AR("txlS_v1AR"),**
- **waAfaqa_v1AR("waAfaqa_v1AR");**

**26. are_Items_related(List<String> itemIds)**

if all element in the list of parameter are subset of one tree, it will return true ,otherwise false

**27. Is_item_In_Tree(String itemId ,Tree spacficTree)**

if itemId parameter is subset of spacficTree parameter , it will return true ,otherwise false

```
Boolean t=awn.Is_item_In_Tree("juzo'_n3AR", Tree.malaka);
```

**28. Get_Root_Item_ID(String itemId )**

Return a list of root Value of the trees where itemId is node of these trees. If itemId isn't node of any tree the list size =0

```
List<String> rootItemId =awn.Get_Root_Item_ID("juzo'_n3AR");
```

29. **Get_All_Node_In_Tree(Tree spacficTree)**
    Return a list of itemId that are subset of specific Tree parameter
    List<String> temp = awn.Get_All_Node_In_Tree(Tree.ZaAhirap);

30. **returnPathToRoot (String ItemId)**
    return a list of list of itemId for a path from ItemId paramater to the root of its trees (because some itemId are in more than one tree) where the first element is the ItemId parameter node and the last element is the root node.
    List<List<String>> temp= awn.returnPathToRoot ("gayomap_n1AR");

31. **Get_Shard_ancestor(String itemId1, String itemId2)**
    If itemId1 and itemId2 are in same tree it will return the nearest common ancestor to these two nodes. Otherwise it will return ""

32. **Get_word_similirty_edge_counting(String itemId1 ,String itemId2)**
    Look in [1] for more detail

33. **Get_word_similirty_wuP(String itemId1 ,String itemId2)**
    Look in Equation 1 in [2]

34. **Get_word_similirty_LeakcockChodorow(String itemId1 ,String itemId2)**
    Look in Equation 2 in [2]

35. **Get_word_similirty_Li (String itemId1 ,String itemId2 ,double alpha,double beta)**
    Look in Equation 3 in [2]

    **Note:**
    For the above methods to compute similarity score, it will return -1.0 if the two concepts are in different tree

[1]D. Yang and D. M. Powers, “Word similarity on the taxonomy of WordNet.”
**[2]**L. Meng and J. Gu, “A New Method for Calculating Word Sense Similarity in WordNet.,” *International Journal of Signal Processing, Image Processing & Pattern Recognition*, vol. 5, no. 3, 2012.