NAME

noun.*suffix*, verb.*suffix*, adj.*suffix*, adv.*suffix* − WordNet lexicographer files that are input to **grind**(1WN)

DESCRIPTION

WordNet's source files are written by lexicographers. They are the product of a detailed relational analysis of lexical semantics: a variety of lexical and semantic relations are used to represent the organization of lexical knowledge. Two kinds of building blocks are distinguished in the source files: word forms and word meanings. Word forms are represented in their familiar orthography; word meanings are represented by synonym sets (*synsets*) − lists of synonymous word forms that are interchangeable in some context. Two kinds of relations are recognized: lexical and semantic. Lexical relations hold between word forms; semantic relations hold between word meanings.

Lexicographer files correspond to the syntactic categories implemented in WordNet − noun, verb, adjective and adverb. All of the synsets in a lexicographer file are in the same syntactic category. Each synset consists of a list of synonymous words or collocations (eg. "**fountain pen**", "**take in**"), and pointers that describe the relations between this synset and other synsets. These relations include (but are not limited to) hypernymy/hyponymy, antonymy, entailment, and meronymy/holonymy. A word or collocation may appear in more than one synset, and in more than one part of speech. Each use of a word in a synset represents a sense of that word in the part of speech corresponding to the synset.

Adjectives may be organized into clusters containing head synsets and satellite synsets. Adverbs generally point to the adjectives from which they are derived.

See **wngloss**(7WN) for a glossary of WordNet terminology and a discussion of the database's content and logical organization.

Lexicographer File Names

The names of the lexicographer files are of the form:

pos.*suffix*

where *pos* is either **noun**, **verb**, **adj** or **adv**. *suffix* may be used to organize groups of synsets into different files, for example **noun.animal** and **noun.plant**. See **lexnames**(5WN) for a list of lexicographer file names that are used in building WordNet.

Pointers

Pointers are used to represent the relations between the words in one synset and another. Semantic pointers represent relations between word meanings, and therefore pertain to all of the words in the source and target synsets. Lexical pointers represent relations between word forms, and pertain only to specific words in the source and target synsets. The following pointer types are usually used to indicate lexical relations: Antonym, Pertainym, Participle, Also See. The remaining pointer types are generally used to represent semantic relations.

A relation from a source to a target synset is formed by specifying a word from the target synset in the source synset, followed by the *pointer_symbol* indicating the pointer type. The location of a pointer within a synset defines it as either lexical or semantic. The **Lexicographer File Format** section describes the syntax for entering a semantic pointer, and **Word Syntax** describes the syntax for entering a lexical pointer.

Although there are many pointer types, only certain types of relations are permitted between synsets of each syntactic category.

The *pointer_symbol*s for nouns are:

!	Antonym
@	Hypernym

~	Hyponym
#m	Member holonym
#s	Substance holonym
#p	Part holonym
%m	Member meronym
%s	Substance meronym
%p	Part meronym
=	Attribute

The *pointer_symbol*s for verbs are:

!	Antonym
@	Hypernym
~	Hyponym
*	Entailment
>	Cause
^	Also see
$	Verb Group

The *pointer_symbol*s for adjectives are:

!	Antonym
&	Similar to
<	Participle of verb
\	Pertainym (pertains to noun)
=	Attribute
^	Also see

The *pointer_symbol*s for adverbs are:

!	Antonym
\	Derived from adjective

Many pointer types are reflexive, meaning that if a synset contains a pointer to another synset, the other synset should contain a corresponding reflexive pointer. **grind**(1WN) automatically inserts missing reflexive pointers for the following pointer types:

Pointer	Reflect
Antonym	Antonym
Hyponym	Hypernym
Hypernym	Hyponym
Holonym	Meronym
Meronym	Holonym
Similar to	Similar to
Attribute	Attribute
Verb Group	Verb Group

Verb Frames

Each verb synset contains a list of generic sentence frames illustrating the types of simple sentences in which the verbs in the synset can be used. For some verb senses, example sentences illustrating actual uses of the verb are provided. (See **Verb Example Sentences** in **wndb**(5WN).) Whenever there is no example sentence, the generic sentence frames specified by the lexicographer are used. The generic sentence frames are entered in a synset as a comma-separated list of integer frame numbers. The following list is the text of the generic frames, preceded by their frame numbers:

1	Something ----s
2	Somebody ----s

3	It is ----ing
4	Something is ----ing PP
5	Something ----s something Adjective/Noun
6	Something ----s Adjective/Noun
7	Somebody ----s Adjective
8	Somebody ----s something
9	Somebody ----s somebody
10	Something ----s somebody
11	Something ----s something
12	Something ----s to somebody
13	Somebody ----s on something
14	Somebody ----s somebody something
15	Somebody ----s something to somebody
16	Somebody ----s something from somebody
17	Somebody ----s somebody with something
18	Somebody ----s somebody of something
19	Somebody ----s something on somebody
20	Somebody ----s somebody PP
21	Somebody ----s something PP
22	Somebody ----s PP
23	Somebody's (body part) ----s
24	Somebody ----s somebody to INFINITIVE
25	Somebody ----s somebody INFINITIVE
26	Somebody ----s that CLAUSE
27	Somebody ----s to somebody
28	Somebody ----s to INFINITIVE
29	Somebody ----s whether INFINITIVE
30	Somebody ----s somebody into V-ing something
31	Somebody ----s something with something
32	Somebody ----s INFINITIVE
33	Somebody ----s VERB-ing
34	It ----s that CLAUSE
35	Something ----s INFINITIVE

Lexicographer File Format

Synsets are entered one per line, and each line is terminated with a newline character. A line containing a synset may be as long as necessary, but no newlines can be entered within a synset. Within a synset, spaces or tabs may be used to separate entities. Items enclosed in italicized square brackets may not be present.

The general synset syntax is:

{ *words* *pointers* **(** *gloss* **)** }

Synsets of this form are valid for all syntactic categories except verb, and are referred to as basic synsets. At least one *word* and a *gloss* are required to form a valid synset. Pointers entered following all the *words* in a synset represent semantic relations between all the words in the source and target synsets.

For verbs, the basic synset syntax is defined as follows:

{ *words* *pointers* *frames* **(** *gloss* **)** }

Adjective may be organized into clusters containing one or more head synsets and optional satellite synsets. Adjective clusters are of the form:

[
head synset
[satellite synsets]
[−]
[additional head/satellite synsets]
]

Each adjective cluster is enclosed in square brackets, and may have one or more parts. Each part consists of a head synset and optional satellite synsets that are conceptually similar to the head synset's meaning. Parts of a cluster are separated by one or more hyphens (−) on a line by themselves, with the terminating square bracket following the last synset. Head and satellite synsets follow the syntax of basic synsets, however a "Similar to" pointer must be specified in a head synset for each of its satellite synsets. Most adjective clusters contain two antonymous parts. See **wngloss**(7WN) for a discussion of adjective clusters, and **Special Adjective Syntax** for more information on adjective cluster syntax.

Synsets for relational adjectives (pertainyms) and participial adjectives do not adhere to the cluster structure. They use the basic synset syntax.

Comments can be entered in a lexicographer file by enclosing the text of the comment in parentheses. Note that comments **cannot** appear within a synset, as parentheses within a synset have an entirely different meaning (see **Gloss Syntax**). However, entire synsets (or adjective clusters) can be "commented out" by enclosing them in parentheses. This is often used by the lexicographers to verify the syntax of files under development or to leave a note to oneself while working on entries.

Word Syntax

A synset must have at least one word, and the words of a synset must appear after the opening brace and before any other synset constructs. A word may be entered in either the simple word or word/pointer syntax.

A simple word is of the form:

word[**(** *marker* **)** *][lex_id]* **,**

word may be entered in any combination of upper and lower case unless it is in an adjective cluster. A collocation is entered by joining the individual words with an underscore character (_). Numbers (integer or real) may be entered, either by themselves or as part of a word string, by following the number with a double quote (").

See **Special Adjective Syntax** for a description of adjective clusters and markers.

word may be followed by an integer *lex_id* from **1** to **15**. The *lex_id* is used to distinguish different senses of the same word within a lexicographer file. The lexicographer assigns *lex_id* values, usually in ascending order, although there is no requirement that the numbers be consecutive. The default is **0**, and does not have to be specified. A *lex_id* must be used on pointers if the desired sense has a non-zero *lex_id* in its synset specification.

Word/pointer syntax is of the form:

[*word[* **(** *marker* **)** *][lex_id]* **,** *pointers* **]**

This syntax is used when one or more pointers correspond only to the specific word in the word/pointer set, rather than all the words in the synset, and represents a lexical relation. Note that a word/pointer set appears within a synset, therefore the square brackets used to enclose it are treated differently from those used to define an adjective cluster. Only one word can be specified in each word/pointer set, and any number of pointers may be included. A synset can have any number of word/pointer sets. Each is treated by **grind**(1WN) essentially as a *word*, so they all must appear before any synset *pointers* representing semantic relations.

For verbs, the word/pointer syntax is extended in the following manner to allow the user to specify generic sentence frames that, like pointers, correspond only to a specific word, rather than all the words in the synset. In this case, *pointers* are optional.

[*word* **,** *[pointers]* *frames* **]**

Pointer Syntax

Pointers are optional in synsets. If a pointer is specified outside of a word/pointer set, the relation is applied to all of the words in the synset, including any words specified using the word/pointer syntax. This indicates a semantic relation between the meanings of the words in the synsets. If specified within a word/pointer set, the relation corresponds only to the word in the set and represents a lexical relation.

A pointer is of the form:

[lex_filename: *]word[lex_id]*,*pointer_symbol*

or:

[lex_filename: *]word[lex_id]^word[lex_id]*,*pointer_symbol*

For pointers, *word* indicates a word in another synset. When the second form of a pointer is used, the first *word* indicates a word in a head synset, and the second is a word in a satellite of that cluster. *word* may be followed by a *lex_id* that is used to match the pointer to the correct target synset. The synset containing *word* may reside in another lexicographer file. In this case, *word* is preceded by *lex_filename* as shown.

See **Pointers** for a list of *pointer_symbol*s and their meanings.

Verb Frame List Syntax

Frame numbers corresponding to generic sentence frames must be entered in each verb synset. If a frame list is specified outside of a word/pointer set, the verb frames in the list apply to all of the words in the synset, including any words specified using the word/pointer syntax. If specified within a word/pointer set, the verb frames in the list correspond only to the word in the set.

A frame number list is entered as follows:

frames: *f_num*[,*f_num...]*

Where *f_num* specifies a generic frame number. See **Verb Frames** for a list of generic sentences and their corresponding frame numbers.

Gloss Syntax

A gloss is included in all synsets. The lexicographer may enter a text string of any length desired. A gloss is simply a string enclosed in parentheses with no embedded carriage returns. It provides a definition of what the synset represents and/or example sentences.

Special Adjective Syntax

The syntax for representing antonymous adjective synsets requires several additional conditions.

The first word of a head synset **must** be entered in upper case, and can be thought of as the head word of the head synset. The *word* part of a pointer from one head synset to another head synset within the same cluster (usually an antonym) must also be entered in upper case. Usually antonymous adjectives are entered using the word/pointer syntax described in **Word Syntax** to indicate a lexical relation. There is no restriction on the number of parts that a cluster may have, and some clusters have three parts, representing antonymous triplets, such as **solid**, **liquid**, and **gas**.

A cross-cluster pointer may be specified, allowing a head or satellite synset to point to a head synset in a different cluster. A cross-cluster pointer is indicated by entering the *word* part of the pointer in upper case.

An adjective may be annotated with a syntactic marker indicating a limitation on the syntactic position the adjective may have in relation to noun that it modifies. If so marked, the marker appears between the word and its following comma. If a *lex_id* is specified, the marker immediately follows it. The syntactic markers are:

- **(p)** predicate position
- **(a)** prenominal (attributive) position
- **(ip)** immediately postnominal position

EXAMPLES

(Note that these are hypothetical examples not found in the WordNet lexicographer files.)

Sample noun synsets:

```
{ canine, [ dog1, cat,! ] pooch, canid,@ }
{ collie, dog1,@ (large multi-colored dog with pointy nose) }
{ hound, hunting_dog, pack,#m dog1,@ }
{ dog, }
```

Sample verb synsets:

```
{ [ confuse, clarify,! frames: 1 ] blur, obscure, frames: 8, 10 }
{ [ clarify, confuse,! ] make_clear, interpret,@ frames: 8 }
{ interpret, construe, understand,@ frames: 8 }
```

Sample adjective clusters:

```
[
{ [ HOT, COLD,! ] lukewarm(a), TEPID,^ warm,& (hot to the touch) }
{ warm, }
–
{ [ COLD, HOT,! ] frigid, freezing,& (cold to the touch) }
{ freezing, }
]

[
{ [ TEPID, ICY,! ] warm,& HOT,^ }
{ warm, TEPID,& }
–
{ [ ICY, TEPID,! ] COLD,& }
]
```

Sample adverb synsets:

```
{ [ basically, adj.all:essential^basic,\ ] [ essentially1, adj.all:essential,\ ] }
{ pointedly, adj.all:pungent^pointed,\ }
{ [ well, adj.all:good1,\ ]}
{ [ badly, adj.all:bad,\ well,! ] ill, ("He was badly prepared") }
```

SEE ALSO

grind(1WN), **wnintro**(5WN), **lexnames**(5WN), **wndb**(5WN), **uniqbeg**(7WN), **wngloss**(7WN).

Fellbaum, C. (1998), ed. "*WordNet: An Electronic Lexical Database*". MIT Press, Cambridge, MA.