Basic idea of what the parsers do

- Philosophy:

The job of parsing the Wikipedia articles is actually accomplished by 3 parsers.

– A preliminary parser (called Parser 0), not documented here, filters out any characters which are not in ISO/IEC 8859-15 and therefore wouldn't be recognized by FreeLing.
– The "first" parser removes the bulk of wiki markup and adapts the formatting to NLP purposes.
– The second parser deals with the wiki markup structures from which text must be extracted, such as links.

- Pros and Cons, Advantages and Disadvantages:

– These parsers would probably have been written much better if I had previously learnt to work with JavaCC, or at least had access to a manual describing in detail how to use it.
– Despite the amateurish aspect, the parsers accomplish fairly well the job of extracting the text in Natural Language of Wikipedia articles, removing wiki markup and elements which are not part of the actual text, such as image filenames etc.
– Parsers 1 and 2 could probably be combined, so as to obtain a one-pass parser, but even with two passes, the parsers do the job quite rapidly, so it's probably not worth the trouble; not to mention that the resulting grammar would be frighteningly complicated.
– If any obscure questions need to be answered... if any details need to be further explained... I can be contacted at samuel.reese@supaero.org and will attempt to help if I possibly can.

- Useful source for Wiki markup:

http://en.wikipedia.org/wiki/Help:Wikitext_examples
http://www.mediawiki.org/wiki/Markup_spec

- 1st parser:

– remove HTML tags

in case of *ref* or *gallery* or *math* or *source* tags, remove any text between opening and closing tags

– remove HTML comments (<!-- … -->)

– HTML symbols: replace

&amp; &nbsp; … and the like with "&" or " "

– Apostrophes:

' or '''' → '
'' or ''' or ''''' → remove

– Equals: pairs of groups

= → =
== or === or ==== or ===== or ====== → . in place of 2nd group (nothing for 1st group)

– formatting at beginning of lines: add ; at end of line if no punctuation (. , ; ? !) is present there

– remove functions of the form: _ _ … _ _ (where … is a sequence of capital letters)

• <u>2nd parser:</u>

– remove all {*} structures (Tables, Templates; the content is retrieved in the case of quotations, like {{cita|*}}, {{cquote|*}}, {{quote|*}})

– treat all link structures: [[*]]

– treat all remaining bracket structures: [*]