Sanskrit

Corpora

Use `CorpusImporter()` or browse the CLTK GitHub repository (anything beginning with `sanskrit_`) to discover available Sanskrit corpora.

```
In [1]: from cltk.corpus.utils.importer import CorpusImporter

In [2]: c = CorpusImporter('sanskrit')

In [3]: c.list_corpora
Out[3]:
['sanskrit_text_jnu', 'sanskrit_text_dcs', 'sanskrit_parallel_sacred_texts',
'sanskrit_text_sacred_texts', 'sanskrit_parallel_gitasupersite',
'sanskrit_text_gitasupersite','sanskrit_text_wikipedia','sanskrit_text_sanskrit_documents']
```

Transliterator

This tool has been derived from the IndicNLP Project courtesy of anoopkunchukuttan This tool is made for transliterating Itrans text to Devanagari(Unicode) script. Also, it can romanize Devanagari script.

Script Conversion

Convert from one Indic script to another. This is a simple script which exploits the fact that Unicode points of various Indic scripts are at corresponding offsets from the base codepoint for that script.more.

```
In [1]: from cltk.corpus.sanskrit.itrans.unicode_transliterate import UnicodeIndicTransliterator

In [2]: input_text=u'राजस्थान'

In [3]: UnicodeIndicTransliterator.transliterate(input_text,"hi","pa")
Out[3]: 'ਰਾਜਸ੍ਥਾਨ'
```

Romanization

Convert script text to Roman text in the ITRANS notation

```
In [4]: from cltk.corpus.sanskrit.itrans.unicode_transliterate import ItransTransliterator

In [5]: input_text=u'राजस्थान'

In [6]: lang='hi'

In [7]: ItransTransliterator.to_itrans(input_text,lang)
Out[7]: 'rAjasthAna'
```

Indicization (ITRANS to Indic Script)

Conversion of ITRANS-transliteration to an Devanagari(Unicode) script

```
In [8]: from cltk.corpus.sanskrit.itrans.unicode_transliterate import ItransTransliterator

In [9]: input_text=u'pitL^In'

In [10]: lang='hi'

In [11]: x=ItransTransliterator.from_itrans(input_text,lang)

In [12]: x
Out[12]: 'पितॢन्'
```

Query Script Information

Indic scripts have been designed keeping phonetic principles in nature and the design and organization of the scripts makes it easy to obtain phonetic information about the characters.

```
In [13]: from cltk.corpus.sanskrit.itrans.langinfo import *

In [14]: c = 'क'

In [15]: lang='hi'

In [16]: is_vowel(c,lang)
Out[16]: False

In [17]: is_consonant(c,lang)
Out[17]: True

In [18]: is_velar(c,lang)
Out[18]: True

In [19]: is_palatal(c,lang)
Out[19]: False

In [20]: is_aspirated(c,lang)
Out[20]: False

In [21]: is_unvoiced(c,lang)
Out[21]: True

In [22]: is_nasal(c,lang)
Out[22]: False
```

Other similar functions are here,

```
In [29]: dir(cltk.corpus.sanskrit.itrans.langinfo)
['APPROXIMANT_LIST', 'ASPIRATED_LIST', 'AUM_OFFSET', 'COORDINATED_RANGE_END_INCLUSIVE',
'COORDINATED_RANGE_START_INCLUSIVE', 'DANDA', 'DENTAL_RANGE', 'DOUBLE_DANDA', 'FRICATIVE_LIST',
'HALANTA_OFFSET', 'LABIAL_RANGE', 'LC_TA', 'NASAL_LIST', 'NUKTA_OFFSET', 'NUMERIC_OFFSET_END',
'NUMERIC_OFFSET_START', 'PALATAL_RANGE', 'RETROFLEX_RANGE', 'RUPEE_SIGN', 'SCRIPT_RANGES',
'UNASPIRATED_LIST', 'UNVOICED_LIST', 'URDU_RANGES', 'VELAR_RANGE', 'VOICED_LIST', '__author__',
'__builtins__', '__cached__', '__doc__', '__file__', '__license__', '__loader__', '__name__',
'__package__', '__spec__', 'get_offset', 'in_coordinated_range', 'is_approximant', 'is_aspirated',
'is_aum', 'is_consonant', 'is_dental', 'is_fricative', 'is_halanta', 'is_indiclang_char',
'is_labial', 'is_nasal', 'is_nukta', 'is_number', 'is_palatal', 'is_retroflex', 'is_unaspirated',
'is_unvoiced', 'is_velar', 'is_voiced', 'is_vowel', 'is_vowel_sign', 'offset_to_char']
```

Syllabifier

This tool has also been derived from the IndicNLP Project courtesy of anoopkunchukuttan This tool can break a word into its syllables, this can be applied across 17 Indian languages including Devanagari (all using Unicode) script.

```
In [23]: from cltk.stem.sanskrit.indian_syllabifier import Syllabifier

In [24]: input_text = 'नमस्ते'

In [26]: lang='hindi'

In [27]: x = Syllabifier(lang)

In [28]: current = x.orthographic_syllabify(input_text)
Out[28]: ['न', 'म','स्ते']
```

Tokenizer

This tool has also been derived from the IndicNLP Project courtesy of anoopkunchukuttan This tool can break a sentence into into its constituent words. It works on the basis of filtering out punctuations and spaces.

```
In [29]: from cltk.tokenize.indian_tokenizer import *

In [30]: input_text = "हिन्दी भारत की सबसे अधिक बोली और समझी जाने वाली भाषा है"

In [31]: x = indian_punctuation_tokenize_regex(input_text)
Out[31]: ['हिन्दी', 'भारत', 'की', 'सबसे', 'अधिक', 'बोली', 'और', 'समझी', 'जाने', 'वाली', 'भाषा', 'है']
```

Stopword Filtering

To use the CLTK's built-in stopwords list:

```
In [1]: from cltk.stop.sanskrit.stops import STOPS_LIST

In [2]: from cltk.tokenize.indian_tokenizer import indian_punctuation_tokenize_regex

In [3]: s = "हमने पिछले पाठ मे सीखा था कि “अहम् गच्छामि” का मतलब “मै जाता हूँ” है। आप ऊपर
   ...:  की तालिकाँओ "

In [4]: tokens = indian_punctuation_tokenize_regex(s)

In [5]: len(tokens)
Out[5]: 20

In [6]: no_stops = [w for w in tokens if w not in STOPS_LIST]

In [7]: len(no_stops)
Out[7]: 18

In [8]: no_stops
Out[8]:
['हमने',
 'पिछले',
 'पाठ',
 'सीखा',
 'था',
 'कि',
 '“अहम्',
 'गच्छामि”',
 'मतलब',
 '“मै',
 'जाता',
 'हूँ”',
 'है',
 '।',
 'आप',
 'ऊपर',
 'की',
 'तालिकाँओ']
```