Cheatography

spaCy Cheat Sheet

by LoiVyen via cheatography.com/163788/cs/34314/

Base Initialization

```
import spacy
```

```
nlp = spacy.blank("en")
nlp =
spacy.load("en_core_web_sm")
nlp =
spacy.load("en_core_web_md")
nlp =
spacy.load("en_core_web_lg")
nlp = spacy.l oa d("e n_c ore -
_we b_t rf")
```

```
doc = nlp("Insert your text
here. This string will be used
to create a doc object. For
natural language proces sin g.")
```

```
print( " tokens: ", [token.text
for token in doc])
```

output:

tokens: ['Insert', 'your', 'text', 'here', '.', 'This', 'string', 'will', 'be', 'used', 'to', 'create', 'a', 'doc', 'object', '.', 'For', 'natural', 'language', 'processing', '.']

Token Attributes applied

Remove punctuations for tokens

```
words = [token.text for token in
doc if token.i s_stop != True
and token.i s_ punct != True]
print (words)
```

output:

['Insert', 'text', 'string', 'create', 'doc', 'object', 'natural', 'language', 'processing']

https://spacy.io/api/token

Lemma

```
for token in doc:
    print("token: ", token, "
lemma: ", token.l emma_)
```

token: token object within object
lemma: 'base' form of token
(ex. going --> go; was --> be)

displaCy Visualizer

Sentence

```
list(d oc.s ents)
print(list(text.sents))
```

output:

[Insert your text here., This string will be used to create a doc object., For natural language processing.]

Word Frequency

```
from collec tions import Counter
```

```
words = [token.text for token in
doc if token.i s_stop != True
and token.i s_ punct != True]
word_freq = Counte r(w ords)
common_words = word_f req.mo -
st_ com mon(5)
print (commo n_w ords)
```

output:

[('Insert', 1), ('text', 1), ('string', 1), ('create', 1), ('doc', 1)]

Using textrank component ("custom")

```
nlp.ad d_p ipe ("te xtr ank ")
doc = nlp("Insert your text
here. This string will be used
to create a doc object. For
natural language proces sin g.")
```

```
for index, token in enumer ate -
(do c._.ph rases):
    print("index: ", index, " -
\ntext: ", token.t ext)
    print("rank:", token.r ank,
" count: ", token.c ount)
    print("chunks: ", token.c -
hunks)
```

output:

index: 0
text: natural language processing
rank: 0.22805281953000428 count: 1
chunks: [natural language processing]
index: 1
text: a doc object
rank: 0.1481208925234081 count: 1
chunks: [a doc object]
index: 2
text: your text
rank: 0.073534568107334 count: 1
chunks: [your text]
index: 3
text: This string

Textacy

```
pip install textacy
```

```
import textacy
```

```
metadata = {
    "title": " Nat ura l-l -
anguage proces sin g",
    "url": " htt ps: //e n.w -
iki ped ia.o rg /wi ki/ Nat -
ura l-l ang uag e_p roc ess -
ing ",
    "source": " wik ipe dia ",
}doc = textac y.m ake _sp acy -
_do c(( text, metadata),
lang="en_core_web_sm")
print(doc._.meta["title"])
```

output: Natural-language processing

"**textacy** is a Python library for performing a variety of natural language processing (NLP) tasks, built on the highperformance spaCy library"
https://github.com/chartbeat-labs/textacy
https://textacy.readthedocs.io/en/latest/

latest version documentation:
https://buildmedia.readthedocs.org/media/pdf/textacy/stable/textacy.pdf
https://textacy.readthedocs.io/en/0.12.0/api_reference/datasets_resources.html

```
import spacy
from spacy import displacy

nlp = spacy.l oa d("e n_c ore -
_we b_s m")
doc = nlp("This is a senten -
ce." )
displacy.render(doc,
style= 'de p', jup yte r=True)
```

```
rank: 0.054063724281900864 count: 1
chunks: [This string]
```

By **LoiVyen**
cheatography.com/loivyen/

Not published yet.
Last updated 2nd October, 2022.