In [1]:
```
import spacy
```

In [2]:
```
nlp = spacy.blank("en")
nlp
```

Out[2]:
```
<spacy.lang.en.English at 0x2090f7235f0>
```

In [3]:
```
text1 = nlp("Dr. Doomsday is most powerful than avengers team")
```

In [4]:
```
for token in text1:
    print(token)
```

```
Dr.
Doomsday
is
most
powerful
than
avengers
team
```

In [5]:
```
text2 =nlp( "harirkishnasanna@gmail.com account has credited $2000 form Harikris
for token in text1:
    print(token)
```

```
Dr.
Doomsday
is
most
powerful
than
avengers
team
```

In [6]:
```
text2[0]
```

Out[6]:
```
harirkishnasanna@gmail.com
```

In [7]:
```
# checking is it MailID or not
text2[0].like_email
```

Out[7]:
```
True
```

In [23]:
```
#now cheking Number
text2[5]
```

Out[23]:
```
2000
```

In [9]:
```
text2[5].is_digit
```

Out[9]:
```
True
```

In [10]:
```
text2[5].like_num
```

Out[10]:
```
True
```

In [11]:
```
# Lets check is curreny or not
text2[4]
```

Out[11]: $

In [12]:
```
text2[4].is_currency
```

Out[12]: True

In [13]:
```
# Lets check all
text2.is_sentenced
```

```
C:\Users\sriha\AppData\Local\Temp\ipykernel_7028\1066314796.py:2: DeprecationWarn
ing: [W107] The property `Doc.is_sentenced` is deprecated. Use `Doc.has_annotatio
n("SENT_START")` instead.
  text2.is_sentenced
```

Out[13]: False

In [14]:
```
for token in text2:

    print(token, token.is_alpha,token.is_currency,token.is_digit,token.like_emai
```

```
harirkishnasanna@gmail.com False False False True
account True False False False
has True False False False
credited True False False False
$ False True False False
2000 False False True False
form True False False False
Harikrishna True False False False
```

In [ ]:

Language change -- Telugu

In [15]:
```
nlp= spacy.blank("te")
text3 = nlp("హరికృష్ణ అనే వాడు నెలా కి ₹300000 సంపాదిస్తున్నాడు హరికృష్ణసన్న@జిమెయి
for token in text3:
    print(token)
```

```
హరికృష్ణ
అనే
వాడు
నెలా
కి
₹
300000
సంపాదిస్తున్నాడు
హరికృష్ణసన్న@జిమెయిల్.com
,
హరికృష్ణసన్న@gmail.com
```

In [16]:
```
# Lets checking Curenncy.
text3[5]
```

Out[16]: ₹

In [17]:
```
text3[5],text3[5].is_currency
```

Out[17]: (₹, True)

In [18]:
```
# lets check integerrs
text3[6],text3[6].is_digit
```

Out[18]: (300000, True)

In [19]:
```
# lets check mail.id
text3[8],text3[8].like_email
```

Out[19]: (హరికృష్ణసన్న@జిమెయిల్.com, False)

In [20]:
```
text3[10],text3[10].like_email
```

Out[20]: (హరికృష్ణసన్న@gmail.com, False)

In [24]:
```
for token in text2:
    print(token)
```

```
harirkishnasanna@gmail.com
account
has
credited
$
2000
form
Harikrishna
```

In [27]:
```
# Lets I want to separate Harrikrishna as "Hari" and "krishna"

from spacy.symbols import ORTH
nlp.tokenizer.add_special_case("Harikrishna",[{ORTH : "Hari"},{ORTH : "krishna"}

text2 =nlp( "harirkishnasanna@gmail.com account has credited $2000 form Harikris
tokens =[token.text for token in text2]
tokens
```

Out[27]:
```
['harirkishnasanna@gmail.com',
 'account',
 'has',
 'credited',
 '$',
 '2000',
 'form',
 'Hari',
 'krishna']
```

Customizing tokenizer

In [30]:
```
for token in text2:
    print(tokens,"==>","index:",token.i,
          "is_alpha",token.is_alpha,
           "is_punct:", token.is_punct,
```

```
        "is_currency",token.is_currency,
        "like_email",token.like_email,)
```

```
['harirkishnasanna@gmail.com', 'account', 'has', 'credited', '$', '2000', 'form',
'Hari', 'krishna'] ==> index: 0 is_alpha False is_punct: False is_currency False
like_email True
['harirkishnasanna@gmail.com', 'account', 'has', 'credited', '$', '2000', 'form',
'Hari', 'krishna'] ==> index: 1 is_alpha True is_punct: False is_currency False l
ike_email False
['harirkishnasanna@gmail.com', 'account', 'has', 'credited', '$', '2000', 'form',
'Hari', 'krishna'] ==> index: 2 is_alpha True is_punct: False is_currency False l
ike_email False
['harirkishnasanna@gmail.com', 'account', 'has', 'credited', '$', '2000', 'form',
'Hari', 'krishna'] ==> index: 3 is_alpha True is_punct: False is_currency False l
ike_email False
['harirkishnasanna@gmail.com', 'account', 'has', 'credited', '$', '2000', 'form',
'Hari', 'krishna'] ==> index: 4 is_alpha False is_punct: False is_currency True l
ike_email False
['harirkishnasanna@gmail.com', 'account', 'has', 'credited', '$', '2000', 'form',
'Hari', 'krishna'] ==> index: 5 is_alpha False is_punct: False is_currency False
like_email False
['harirkishnasanna@gmail.com', 'account', 'has', 'credited', '$', '2000', 'form',
'Hari', 'krishna'] ==> index: 6 is_alpha True is_punct: False is_currency False l
ike_email False
['harirkishnasanna@gmail.com', 'account', 'has', 'credited', '$', '2000', 'form',
'Hari', 'krishna'] ==> index: 7 is_alpha True is_punct: False is_currency False l
ike_email False
['harirkishnasanna@gmail.com', 'account', 'has', 'credited', '$', '2000', 'form',
'Hari', 'krishna'] ==> index: 8 is_alpha True is_punct: False is_currency False l
ike_email False
```

Sentence Tokenization or Segmentation

In [ ]:

In [32]:
```
text2 =nlp( "harirkishnasanna@gmail.com account has credited $2000 form Harikris
for sentance in text2.sents:
    print(sentance)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[32], line 2
      1 text2 =nlp( "harirkishnasanna@gmail.com account has credited $2000 form H
arikrishna")
----> 2 for sentance in text2.sents:
      3     print(sentance)

File ~\AppData\Local\Programs\Python\Python312\Lib\site-packages\spacy\tokens\do
c.pyx:926, in sents()

ValueError: [E030] Sentence boundaries unset. You can add the 'sentencizer' compo
nent to the pipeline with: `nlp.add_pipe('sentencizer')`. Alternatively, add the
dependency parser or sentence recognizer, or set sentence boundaries by setting `
doc[i].is_sent_start`.
```

In [ ]:

In [33]:
```
# But Before sentance we create pipelines for that

nlp.pipeline
```

Out[33]:
```
[]
```

In [35]:
```
nlp.add_pipe('sentencizer')
```

Out[35]:
```
<spacy.pipeline.sentencizer.Sentencizer at 0x2091186f810>
```

In [36]:
```
nlp.pipe_names
```

Out[36]:
```
['sentencizer']
```

In [37]:
```
text2 =nlp( "harirkishnasanna@gmail.com account has credited $2000 form Harikris
for sentance in text2.sents:
    print(sentance)
```

```
harirkishnasanna@gmail.com account has credited $2000 form Harikrishna
```

In [ ]:

In [ ]:

Exercise

In [ ]:

In [38]:
```
text='''
Look for data to help you address the question. Governments are good
sources because data from public research is often freely available. Good
places to start include http://www.data.gov/, and http://www.science.
gov/, and in the United Kingdom, http://data.gov.uk/.
Two of my favorite data sets are the General Social Survey at http://www3.norc.o
and the European Social Survey at http://www.europeansocialsurvey.org/.
'''
```

In [39]:
```
# seapareing all differnt Websites
mails = nlp(text)
data_mails = [token.text for token in mails if token.like_url]
data_mails
```

Out[39]:
```
['http://www.data.gov/',
 'http://www.science',
 'http://data.gov.uk/.',
 'http://www3.norc.org/gss+website/',
 'http://www.europeansocialsurvey.org/.']
```

In [ ]:

In [44]:
```
# Figure out all transactions from this text with amount and currency

transactions = "Tony gave two $ to Peter, Bruce gave 500 € to Steve"
amount = nlp(transactions)
for token in amount:
```

```
    if token.like_num and amount[token.i+1].is_currency:
        print(token.text, amount[token.i+1].text)
```

```
500 €
```

In [ ]: