In [114]:
```
from spacy.matcher import Matcher
import spacy
import random
nlp=spacy.load('en_core_web_sm')
```

In [115]:
```
TEXTS=['How to preorder the iPhone X',
 'iPhone X is coming',
 'Should I pay $1,000 for the iPhone X?',
 'The iPhone 8 reviews are here',
 'Your iPhone goes up to 11 today',
 'I need a new phone! Any tips?']
TEXTS_TEST_DATA=['Apple is slowing down the iPhone 8 and iPhone X - how to sto
p it',
 "I finally understand what the iPhone X 'notch' is for",
 'Everything you need to know about the Samsung Galaxy S9',
 'Looking to compare iPad models? Here’s how the 2018 lineup stacks up',
 'The iPhone 8 and iPhone 8 Plus are smartphones designed, developed, and mark
eted by Apple',
 'what is the cheapest ipad, especially ipad pro???',
 'Samsung Galaxy is a series of mobile computing devices designed, manufacture
d and marketed by Samsung Electronics with iPhone 5']
doc = list(nlp.pipe(TEXTS))
matcher = Matcher(nlp.vocab)
```

In [116]:
```
# Two tokens whose lowercase forms match 'iphone' and 'x'
pattern1 = [{'LOWER': 'iphone'}, {'LOWER': 'x'}]

# Token whose lowercase form matches 'iphone' and an optional digit
pattern2 = [{'LOWER': 'iphone'}, {'IS_DIGIT': True,"OP":"?"}]

# Add patterns to the matcher
matcher.add('GADGET', None, pattern1, pattern2)
```

In [117]:
```
# Create a Doc object for each text in TEXTS which creates a context for GADGE
TS if any
for doc in list(nlp.pipe(TEXTS)):
    # Find the matches in the doc
    matches = matcher(doc)

    # Get a list of (start, end, label) tuples of matches in the text
    entities = [(start, end, 'GADGET') for match_id, start, end in matches]
    print(doc.text, entities)
```

```
How to preorder the iPhone X [(4, 6, 'GADGET'), (4, 5, 'GADGET')]
iPhone X is coming [(0, 2, 'GADGET'), (0, 1, 'GADGET')]
Should I pay $1,000 for the iPhone X? [(7, 9, 'GADGET'), (7, 8, 'GADGET')]
The iPhone 8 reviews are here [(1, 2, 'GADGET'), (1, 3, 'GADGET')]
Your iPhone goes up to 11 today [(1, 2, 'GADGET')]
I need a new phone! Any tips? []
```

In [124]:
```
#Creation of Training Data
TRAINING_DATA = []

# Create a Doc object for each text in TEXTS
for doc in nlp.pipe(TEXTS):
    # Match on the doc and create a list of matched spans
    spans = [doc[start:end] for match_id, start, end in matcher(doc)]
    # Get (start character, end character, label) tuples of matches
    entities=[]
    print(spans)
    for span in spans:
        for i in entities:
            print(i)
            if i[0]==span.start_char:
                if i[1]<span.end_char:
                    del[entities[entities.index(i)]]
                else:
                    break
        else:
            entities.append((span.start_char, span.end_char, 'GADGET'))
    # Format the matches as a (doc.text, entities) tuple
    training_example = (doc.text, {'entities': entities})
    # Append the example to the training data
    TRAINING_DATA.append(training_example)

#Before you train a model with the data, you always want to double-check that
#your matcher didn't identify any false positives.
#But that process is still much faster than doing everything manually.
print(*TRAINING_DATA, sep='\n')
```

```
[iPhone X, iPhone]
(20, 28, 'GADGET')
[iPhone X, iPhone]
(0, 8, 'GADGET')
[iPhone X, iPhone]
(28, 36, 'GADGET')
[iPhone, iPhone 8]
(4, 10, 'GADGET')
[iPhone]
[]
('How to preorder the iPhone X', {'entities': [(20, 28, 'GADGET')]})
('iPhone X is coming', {'entities': [(0, 8, 'GADGET')]})
('Should I pay $1,000 for the iPhone X?', {'entities': [(28, 36, 'GADGET')]})
('The iPhone 8 reviews are here', {'entities': [(4, 12, 'GADGET')]})
('Your iPhone goes up to 11 today', {'entities': [(5, 11, 'GADGET')]})
('I need a new phone! Any tips?', {'entities': []})
```

In [125]:
```
# Create a blank 'en' model
nlp = spacy.blank('en')
```

In [126]:
```
# Create a new entity recognizer and add it to the pipeline
ner = nlp.create_pipe('ner')
nlp.add_pipe(ner)
```

In [127]:
```
# Add the label 'GADGET' to the entity recognizer
ner.add_label('GADGET')
```

In [128]:
```
# Start the training - initialize weigths to random
nlp.begin_training()

# Loop for 10 iterations
for itn in range(10):
    # Shuffle the training data
    random.shuffle(TRAINING_DATA)
    losses = {}
    for batch in spacy.util.minibatch(TRAINING_DATA, size=2):
        texts = [text for text, entities in batch]
        annotations = [entities for text, entities in batch]
        nlp.update(texts, annotations, losses=losses)
        print(losses)
```

```
{'ner': 8.333333253860474}
{'ner': 20.298118472099304}
{'ner': 33.15032756328583}
{'ner': 10.349002599716187}
{'ner': 16.706289410591125}
{'ner': 21.364170104265213}
{'ner': 3.4739289432764053}
{'ner': 4.880907749757171}
{'ner': 8.365761628840119}
{'ner': 3.137280941242352}
{'ner': 3.9661206960845448}
{'ner': 5.721750619279192}
{'ner': 4.413929186761379}
{'ner': 5.648673966861679}
{'ner': 8.609826355110272}
{'ner': 3.266738541831728}
{'ner': 5.328867996518966}
{'ner': 7.160170043182006}
{'ner': 1.613703683251515}
{'ner': 2.465106178683527}
{'ner': 3.30830936287839}
{'ner': 0.3054005171288736}
{'ner': 0.3778636934909514}
{'ner': 1.7764691235424266}
{'ner': 0.01839688112886506}
{'ner': 0.018933855623856743}
{'ner': 0.7492466795803097}
{'ner': 0.0006282346955259754}
{'ner': 2.232760816175464}
{'ner': 2.2327636593631537}
```

In [129]:
```
# Process each text in TEXTS_TEST_DATA
for doc in nlp.pipe(TEXTS_TEST_DATA):
    # Print the document text and entitites
    print(doc.text)
    print(doc.ents, '\n\n')
```

```
Apple is slowing down the iPhone 8 and iPhone X - how to stop it
(iPhone 8, iPhone X)


I finally understand what the iPhone X 'notch' is for
(iPhone X,)


Everything you need to know about the Samsung Galaxy S9
()


Looking to compare iPad models? Here’s how the 2018 lineup stacks up
()


The iPhone 8 and iPhone 8 Plus are smartphones designed, developed, and marke
ted by Apple
(iPhone 8, iPhone 8)


what is the cheapest ipad, especially ipad pro???
()


Samsung Galaxy is a series of mobile computing devices designed, manufactured
and marketed by Samsung Electronics with iPhone 5
(iPhone 5,)
```

In [ ]: