Installations, if needed

In [ ]:
```
!pip install spacy==2.0.13 # Above 2.0.13 doesn't work with the neuralcoref resol
ution
!pip install https://github.com/huggingface/neuralcoref-models/releases/download/e
n_coref_md-3.0.0/en_coref_md-3.0.0.tar.gz # This is the coref language model
!pip install networkx
!pip install pydot # To draw our graphs in graphviz
!pip install graphviz
```

Imports & Loading Files

In [9]:
```
import spacy
from spacy import displacy
from collections import Counter
import re
import os
import pandas as pd
import networkx as nx
import sys
import pydot
import matplotlib.pyplot as plt
import graphviz

TEXT_FILENAME = 'TheOrange.txt'

with open(TEXT_FILENAME, 'rb') as raw_text:
    text = raw_text.read().strip().decode('utf-8',errors='replace')
```

Cleaning text

In [10]:
```
# Can uncomment the below if Harry Potter & removing title.
#text = text[39:]

cleaned_text = re.sub(r'(?:[A-Z]{2,}\s+)','',text).strip()
cleaned_text=re.sub(r'\s+',' ',cleaned_text)

# Can uncomment if lowercase is desired
#cleaned_text = text.lower()
```

Loading and running SpaCy English Medium-Sized Pipeline

In [11]:
```
# Download the english medium-sized pipeline
! python -m spacy download en_core_web_md
```

```
Requirement already satisfied: en_core_web_md==2.0.0 from https://github.com/exp
losion/spacy-models/releases/download/en_core_web_md-2.0.0/en_core_web_md-2.0.0.
tar.gz#egg=en_core_web_md==2.0.0 in /anaconda/envs/ear/lib/python3.6/site-packag
es (2.0.0)

    Linking successful
    /anaconda/envs/ear/lib/python3.6/site-packages/en_core_web_md -->
    /anaconda/envs/ear/lib/python3.6/site-packages/spacy/data/en_core_web_md

    You can now load the model via spacy.load('en_core_web_md')
```

In [12]:
```
# Use that file to process the text into a doc.
nlp = spacy.load('en_core_web_md')
doc = nlp(cleaned_text)
```

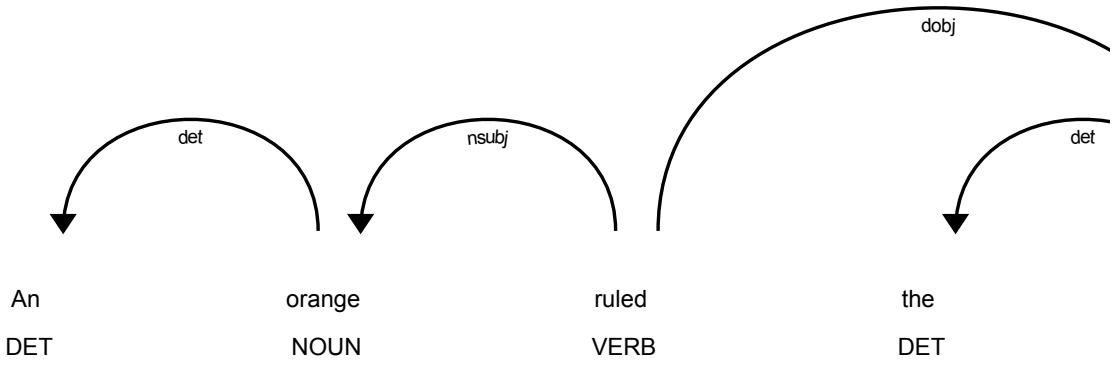
Viewing named entities

In [14]:
```
displacy.render(doc, jupyter=True, style='ent')
```

An orange ruled the world. It was an unexpected thing, the temporary abdication of Heavenly Providence, entrusting the whole matter to a simple orange. The orange, in a grove in Florida **GPE**, humbly accepted the honor. The other oranges, the birds, and the men in their tractors wept with joy; the tractors' motors rumbled hymns of praise. Airplane pilots passing over would circle the grove and tell their passengers, "Below us is the grove where the orange who rules the world grows on a simple branch." And the passengers would be silent with awe. The governor of Florida **GPE** declared every day **DATE** a holiday. On summer afternoons **TIME** the Dalai Lama would come to the grove and sit with the orange, and talk about life. When the time came for the orange to be picked, none of the migrant workers would do it: they went on strike. The foremen wept. The other oranges swore they would turn sour. But the orange who ruled the world said, "No, my friends; it is time." Finally a man from Chicago **GPE**, with a heart as windy and cold as Lake Michigan **LOC** in wintertime, was brought in. He put down his briefcase, climbed up on a ladder, and picked the orange. The birds were silent and the clouds had gone away. The orange thanked the man from Chicago **GPE**. They say that when the orange went through the national produce processing and distribution system, certain machines turned to gold, truck drivers had epiphanies, aging rural store managers called their estranged lesbian daughters on Wall Street and all was forgiven. I bought the orange who ruled the world for 39 cents **MONEY** at Safeway **ORG** three days ago **DATE**, and for three days **DATE** he sat in my fruit basket and was my teacher. Today **DATE**, he told me, "it is time," and I ate him. Now we are on our own again.

Viewing dependencies

In [16]:
```
sentence_spans = list(doc.sents)
displacy.render(sentence_spans[0], jupyter=True, style='dep')
```



Extracting Triples with ReVerb

From http://reverb.cs.washington.edu/ (http://reverb.cs.washington.edu/) (and used in the paper "Information retrieval in folktales using natural language processing": https://arxiv.org/pdf/1511.03012.pdf (https://arxiv.org/pdf/1511.03012.pdf))

In [17]:
```
# Preparing the filename for the tab-separated values output of ReVerb
tsv_filename = TEXT_FILENAME.split('.')[0] + "_ReVerb.tsv"

# Run the Java package. Java JDK is required to be installed from
# https://www.oracle.com/technetwork/java/javase/downloads/jdk11-downloads-5066655.html
os.system("java -Xmx512m -jar reverb-latest.jar -a {0} > {1}".format(TEXT_FILENAME, tsv_filename))

# Reading the results from the Java executable into a Pandas dataframe.
reverb_results = pd.read_csv(tsv_filename, header=None, sep='\t')
reverb_results.columns = ['filename','Sentence_Num','Arg1','Rel','Arg2','Arg1_StartInd','Arg1_EndInd','Rel_StartInd','Rel_EndInd', 'Arg2_StartInd','Arg2_EndInd', 'Confidence','Sent_Text', 'Sent_POS','Sent_ChunkTags', 'Arg1_Norm', 'Rel_Norm', 'Arg2_Norm']

# Set the option to not truncate the text in longer cells
pd.set_option('display.max_colwidth', -1)
```

In [18]:
```
# Exploring the dataframe
reverb_results.head()
```

Out[18]:

	filename	Sentence_Num	Arg1	Rel	Arg2	Arg1_StartInd	Arg1_EndInd	Rel_Sta
0	/Users/andrewlarimer/Dropbox/Berkeley/W266_NLP/FinalProject/TheOrange.txt	1	An orange	ruled	the world	0	2	2
1	/Users/andrewlarimer/Dropbox/Berkeley/W266_NLP/FinalProject/TheOrange.txt	2	It	was	an unexpected thing	0	1	1
2	/Users/andrewlarimer/Dropbox/Berkeley/W266_NLP/FinalProject/TheOrange.txt	2	It	was	the temporary abdication of Heavenly Providence	0	1	1
3	/Users/andrewlarimer/Dropbox/Berkeley/W266_NLP/FinalProject/TheOrange.txt	3	The orange, in a grove in Florida	humbly accepted	the honor	0	8	9
4	/Users/andrewlarimer/Dropbox/Berkeley/W266_NLP/FinalProject/TheOrange.txt	4	the tractors' motors	rumbled	hymns of praise	17	21	21

Graphing

Converting triples to graph nodes and edges

In [19]:
```
def create_nodes_and_edge(row):
    G.add_node(row['Arg1'])
    G.add_node(row['Arg2'])
    G.add_edge(row['Arg1'], row['Arg2'], label=row['Rel'])
```

In [20]:
```
# Establish our graph using Networkx
G = nx.Graph()
_ = reverb_results.apply(lambda x: create_nodes_and_edge(x), axis=1)
```

In [21]:
```
# Write our graph to DOT format to be read and visualized by GraphViz
nx.drawing.nx_pydot.write_dot(G,'graph_dot.txt')
```

Draw the graph

In [22]:
```
from graphviz import Source

graph_filename = TEXT_FILENAME.split('.')[0] + '_Graph'

# Load the saved DOT format
graph_visualized = Source.from_file('graph_dot.txt')

# Save it to a png
graph_visualized.render(filename=graph_filename, format='png')

# View it in the notebook
graph_visualized
```

Out[22]:

Coreference Resolution

Using the NeuralCoref library: https://github.com/huggingface/neuralcoref (https://github.com/huggingface/neuralcoref) (If I can ever get it to work, which so far I cannot.)

In [ ]:
```
nlp_coref = spacy.load('en_coref_sm')
doc = nlp_coref(cleaned_text)
```

In [ ]:
```
doc._.has_coref
doc._.coref_clusters
```