Problem Statement

In this particular project, we are going to work on the inaugural corpora from the nltk in Python. We will be looking at the following speeches of the Presidents of the United States of America:

- President Franklin D. Roosevelt in 1941
- President John F. Kennedy in 1961
- President Richard Nixon in 1973

Importing required libraries

In [1]:
```
# Pandas and Numpy libraries
import pandas as pd
import numpy as np

import nltk                                                                # Used fo
from nltk.corpus import stopwords                                          # Used fo
from nltk.stem.porter import PorterStemmer                                 # Used fo
import re, string                                                          # Used fo
from wordcloud import WordCloud                                            # Used fo

# libaries to help with data visualization
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings("ignore")
```

Understanding the structure of data

In [2]:
```
df = pd.read_excel('Project_Speech.xlsx') # Importing the data
```

In [3]:
```
df.head() # Returns first 5 rows
```

Out[3]:

	Name	Speech
0	Roosevelt	On each national day of inauguration since 178...
1	Kennedy	Vice President Johnson, Mr. Speaker, Mr. Chief...
2	Nixon	Mr. Vice President, Mr. Speaker, Mr. Chief Jus...

Number of rows and columns in the dataset

In [4]:
```
# checking shape of the data

rows = str(df.shape[0])
```

```
columns = str(df.shape[1])

print(f"There are \033[1m" + rows + "\033[0m rows and \033[1m" + columns + "\033[0m
```

```
There are 3 rows and 2 columns in the dataset.
```

Datatypes of the different columns in the dataset

In [5]:
```
df.info() # Concise summary of dataset
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3 entries, 0 to 2
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Name    3 non-null      object
 1   Speech  3 non-null      object
dtypes: object(2)
memory usage: 176.0+ bytes
```

There are 2 columns in the dataset. Both are having object data type.

Finding number of Characters, Words & Sentences in all three speeches

Number of Characters - including spaces in each speech

In [6]:
```
# Number of Characters - including spaces

df['char_count'] = df['Speech'].str.len()
```

In [7]:
```
df.head() # Returns first 5 rows
```

Out[7]:

	Name	Speech	char_count
0	Roosevelt	On each national day of inauguration since 178...	7651
1	Kennedy	Vice President Johnson, Mr. Speaker, Mr. Chief...	7673
2	Nixon	Mr. Vice President, Mr. Speaker, Mr. Chief Jus...	10106

In [8]:
```
print(f"There are \033[1m" + str(df.iloc[0,2]) + "\033[0m characters in President \
print(f"There are \033[1m" + str(df.iloc[1,2]) + "\033[0m characters in President \
print(f"There are \033[1m" + str(df.iloc[2,2]) + "\033[0m characters in President \
```

```
There are 7651 characters in President Roosevelt speech.
There are 7673 characters in President Kennedy speech.
There are 10106 characters in President Nixon speech.
```

Number of Words in each speech

In [9]:
```
# Get word count of all three speeches
```

```
df['total_words'] = [len(x.split()) for x in df['Speech'].tolist()]
```

In [10]:
```
df.head() # Returns first 5 rows
```

Out[10]:

	Name	Speech	char_count	total_words
0	Roosevelt	On each national day of inauguration since 178...	7651	1323
1	Kennedy	Vice President Johnson, Mr. Speaker, Mr. Chief...	7673	1364
2	Nixon	Mr. Vice President, Mr. Speaker, Mr. Chief Jus...	10106	1769

In [11]:
```
print(f"There are \033[1m" + str(df.iloc[0,3]) + "\033[0m words in President \033[1
print(f"There are \033[1m" + str(df.iloc[1,3]) + "\033[0m words in President \033[1
print(f"There are \033[1m" + str(df.iloc[2,3]) + "\033[0m words in President \033[1
```

```
There are 1323 words in President Roosevelt speech.
There are 1364 words in President Kennedy speech.
There are 1769 words in President Nixon speech.
```

Number of Sentences in each speech

In [12]:
```
# Get sentence count of all three speeches

df['total_sentences'] = df['Speech'].str.count('[\w][\.!\?]')
```

In [13]:
```
df.head() # Returns first 5 rows
```

Out[13]:

	Name	Speech	char_count	total_words	total_sentences
0	Roosevelt	On each national day of inauguration since 178...	7651	1323	68
1	Kennedy	Vice President Johnson, Mr. Speaker, Mr. Chief...	7673	1364	54
2	Nixon	Mr. Vice President, Mr. Speaker, Mr. Chief Jus...	10106	1769	72

In [14]:
```
print(f"There are \033[1m" + str(df.iloc[0,4]) + "\033[0m sentences in President \0
print(f"There are \033[1m" + str(df.iloc[1,4]) + "\033[0m sentences in President \0
print(f"There are \033[1m" + str(df.iloc[2,4]) + "\033[0m sentences in President \0
```

```
There are 68 sentences in President Roosevelt speech.
There are 54 sentences in President Kennedy speech.
There are 72 sentences in President Nixon speech.
```

Stopword removal - Stemming - Finding 3 most common words used in all three speeches

Lowercase - each speech

```
In [15]: # Converting each speech in lower case

         df['Speech'] = df['Speech'].apply(lambda x: " ".join(x.lower() for x in x.split()))
```

```
In [16]: df.head() # Returns first 5 rows
```

Out[16]:

	Name	Speech	char_count	total_words	total_sentences
0	Roosevelt	on each national day of inauguration since 178...	7651	1323	68
1	Kennedy	vice president johnson, mr. speaker, mr. chief...	7673	1364	54
2	Nixon	mr. vice president, mr. speaker, mr. chief jus...	10106	1769	72

Special characters removal - each speech

```
In [17]: # Remove special characters from each speech

         df['Speech'] = df['Speech'].str.replace('\n',"").str.replace('--',"").str.replace('
```

```
In [18]: df.head() # Returns first 5 rows
```

Out[18]:

	Name	Speech	char_count	total_words	total_sentences
0	Roosevelt	on each national day of inauguration since 178...	7651	1323	68
1	Kennedy	vice president johnson, mr. speaker, mr. chief...	7673	1364	54
2	Nixon	mr. vice president, mr. speaker, mr. chief jus...	10106	1769	72

Stopword removal - each speech

```
In [19]: stopwords = nltk.corpus.stopwords.words('english') + list(string.punctuation)
```

```
In [20]: df['Speech'] = df['Speech'].apply(lambda x: " ".join(x for x in x.split() if x not
```

```
In [21]: df.head() # Returns first 5 rows
```

Out[21]:

	Name	Speech	char_count	total_words	total_sentences
0	Roosevelt	national day inauguration since 1789, people r...	7651	1323	68
1	Kennedy	vice president johnson, mr. speaker, mr. chief...	7673	1364	54
2	Nixon	mr. vice president, mr. speaker, mr. chief jus...	10106	1769	72

Stemming - each speech

In [22]:
```
st = PorterStemmer()
```

In [23]:
```
df['Speech'] = df['Speech'].apply(lambda x: " ".join([st.stem(word) for word in x.s
```

In [24]:
```
df.head() # Returns first 5 rows
```

Out[24]:

	Name	Speech	char_count	total_words	total_sentences
0	Roosevelt	nation day inaugur sinc 1789, peopl renew sens...	7651	1323	68
1	Kennedy	vice presid johnson, mr. speaker, mr. chief ju...	7673	1364	54
2	Nixon	mr. vice president, mr. speaker, mr. chief jus...	10106	1769	72

Find 3 most common words - each speech

In [25]:
```
# Get all words in President Roosevelt speech

all_words_rs = [x for x in pd.Series(''.join(df.iloc[0,1]).split())]
```

In [26]:
```
# Finding 3 most common words in President Roosevelt speech

print(f"3 most common words in President \033[1m" + df.iloc[0,0] + "\033[0m speech

print(nltk.FreqDist(all_words_rs).most_common(3))
```

```
3 most common words in President Roosevelt speech are:
[('nation', 10), ('know', 9), ('us', 8)]
```

In [27]:
```
# Get all words in President Kennedy speech

all_words_kn = [x for x in pd.Series(''.join(df.iloc[1,1]).split())]
```

In [28]:
```
# Finding 3 most common words in President Kennedy speech
```

```
print(f"3 most common words in President \033[1m" + df.iloc[1,0] + "\033[0m speech
print(nltk.FreqDist(all_words_kn).most_common(3))
```

```
3 most common words in President Kennedy speech are:
[('let', 11), ('us', 11), ('power', 7)]
```

In [29]:
```
# Get all words in President Nixon speech

all_words_ni = [x for x in pd.Series(''.join(df.iloc[2,1]).split())]
```

In [30]:
```
# Finding 3 most common words in President Nixon speech

print(f"3 most common words in President \033[1m" + df.iloc[2,0] + "\033[0m speech
print(nltk.FreqDist(all_words_ni).most_common(3))
```

```
3 most common words in President Nixon speech are:
[('us', 25), ('new', 15), ('let', 13)]
```

In [31]:
```
# Get all words in all speeches

all_Words = [x for x in pd.Series(''.join(df['Speech']).split())]
```

In [32]:
```
# Finding 3 most common words in all three speeches

print(f"3 most common words in all three speeches are:\033[1m")
print(nltk.FreqDist(all_Words).most_common(3))
```

```
3 most common words in all three speeches are:
[('us', 44), ('new', 26), ('let', 25)]
```

Word cloud of all three speeches

Word cloud for President Roosevelt speech

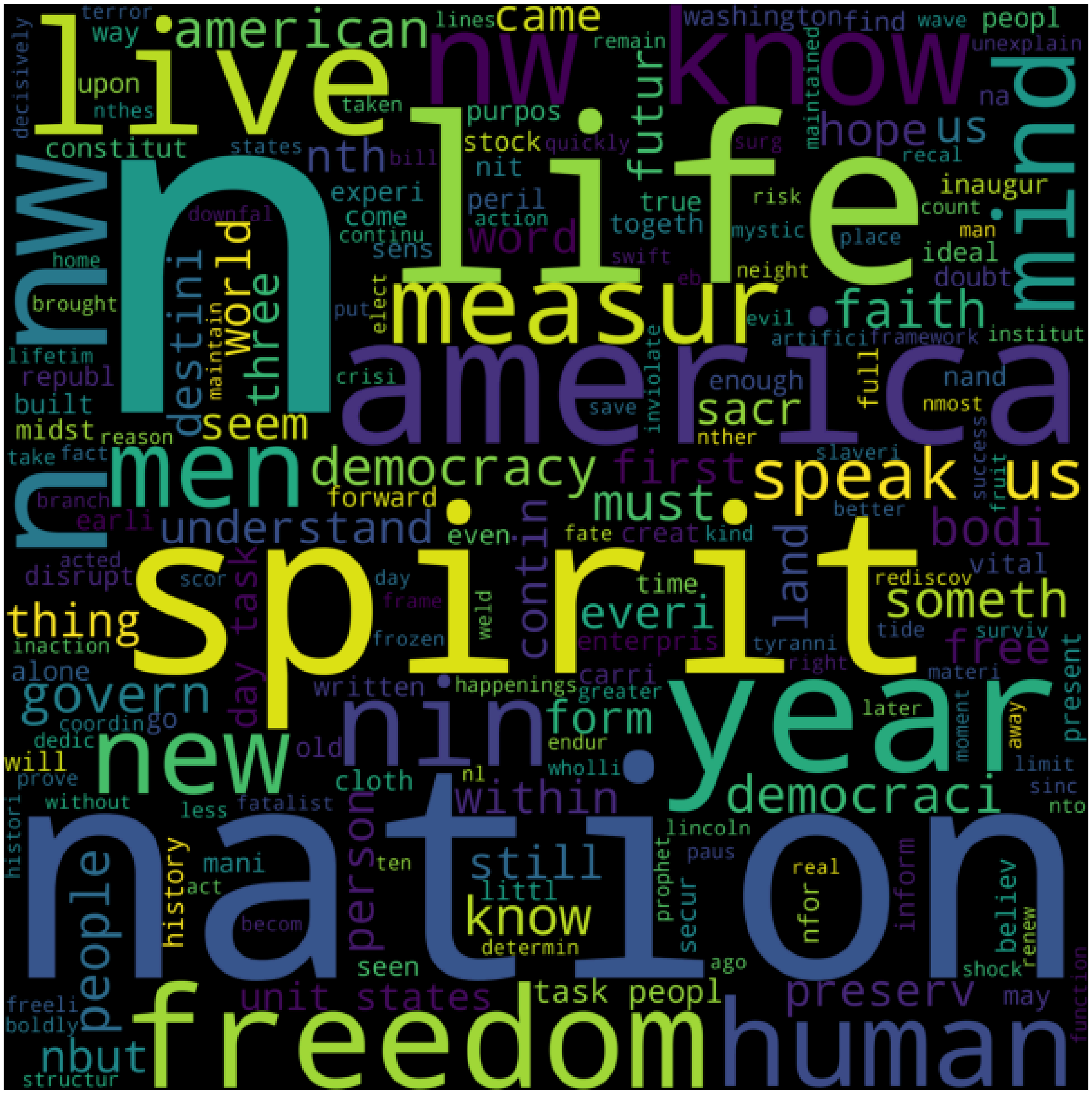
In [33]:
```
wc_rs = ''.join(df.iloc[0,1])
```

In [34]:
```
# Word Cloud
wordcloud = WordCloud(width = 3000, height = 3000,
                background_color ='black',
                min_font_size = 10, random_state=100).generate(wc_rs)

# plot the WordCloud image
plt.figure(figsize = (6, 6), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.xlabel('Word Cloud')
plt.tight_layout(pad = 0)

plt.title("Fig 1: Word Cloud for President Roosevelt speech")
plt.show()
```

Fig 1: Word Cloud for President Roosevelt speech



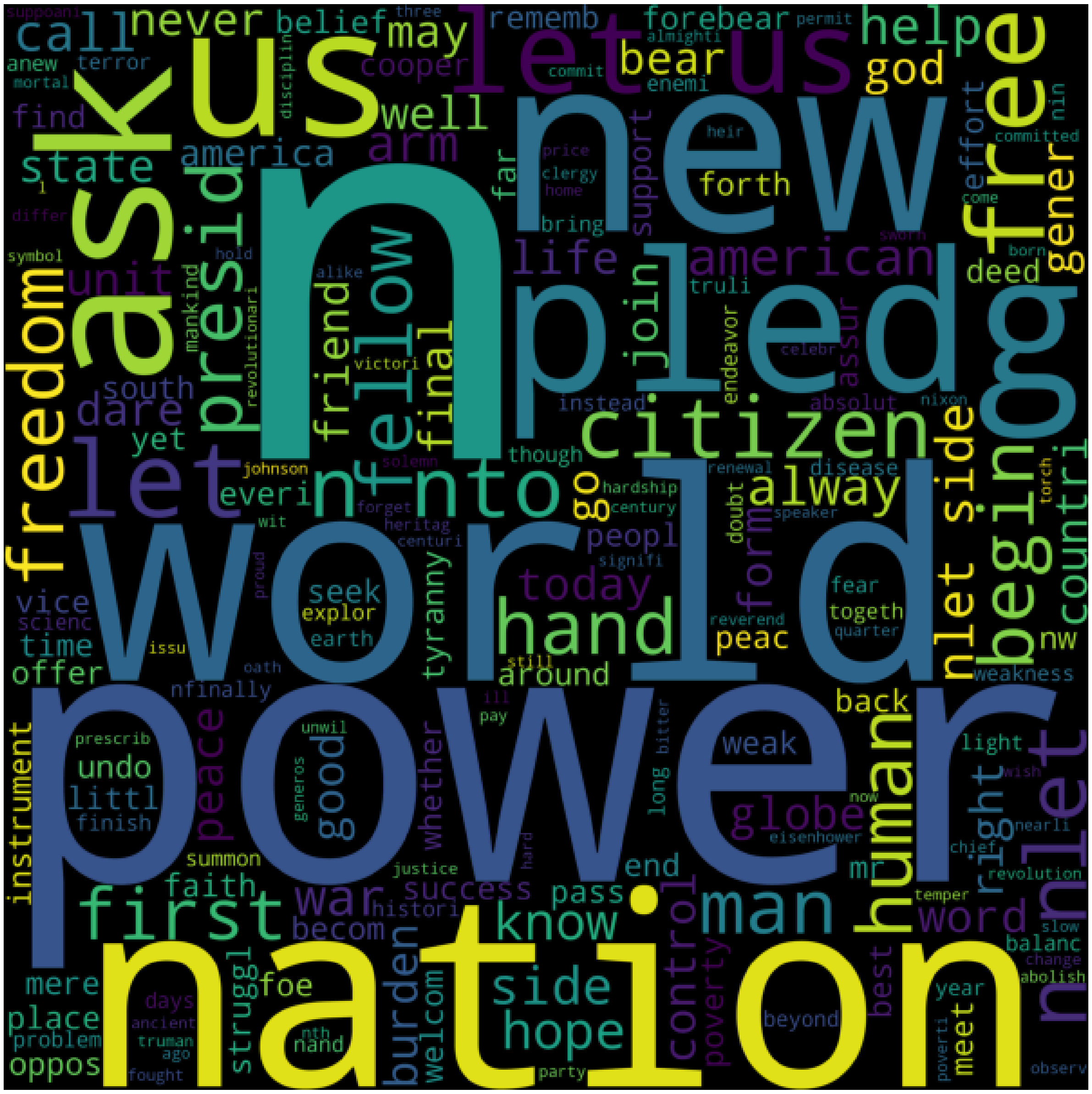
Word cloud for President Kennedy speech

```
In [35]: wc_kn = ''.join(df.iloc[1,1])
```

```
In [36]: # Word Cloud
         wordcloud = WordCloud(width = 3000, height = 3000,
                         background_color ='black',
                         min_font_size = 10, random_state=100).generate(wc_kn)

         # plot the WordCloud image
         plt.figure(figsize = (6, 6), facecolor = None)
         plt.imshow(wordcloud)
         plt.axis("off")
         plt.xlabel('Word Cloud')
         plt.tight_layout(pad = 0)
         plt.title("Fig 2: Word Cloud for President Kennedy speech")
         plt.show()
```

Fig 2: Word Cloud for President Kennedy speech



Word cloud for President Nixon speech

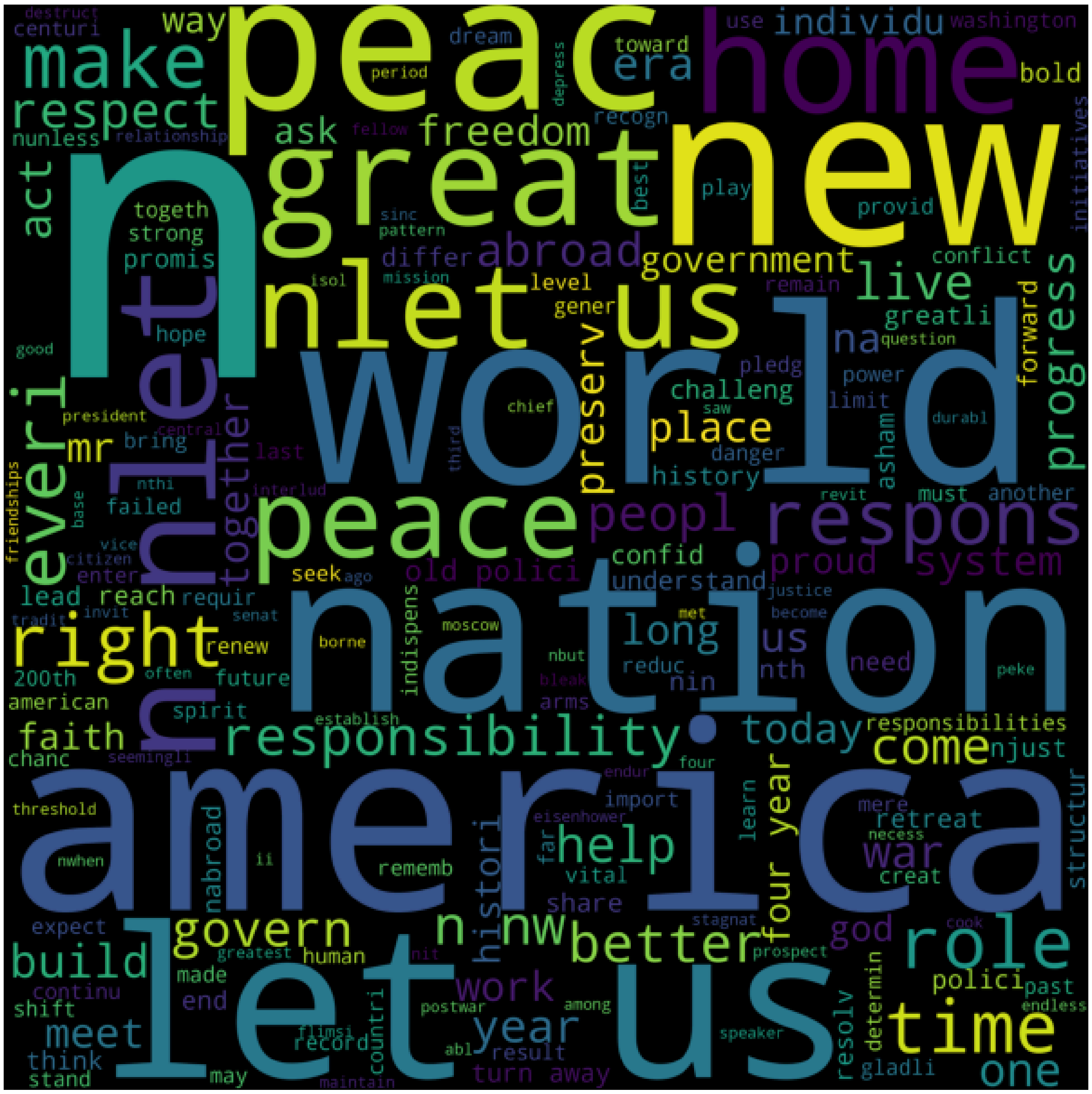
In [37]:
```
wc_ni = ''.join(df.iloc[2,1])
```

In [38]:
```
# Word Cloud
wordcloud = WordCloud(width = 3000, height = 3000,
                background_color ='black',
                min_font_size = 10, random_state=100).generate(wc_ni)

# plot the WordCloud image
plt.figure(figsize = (6, 6), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.xlabel('Word Cloud')
plt.tight_layout(pad = 0)

plt.title("Fig 3: Word Cloud for President Nixon speech")
plt.show()
```

Fig 3: Word Cloud for President Nixon speech



Word cloud for all three speeches

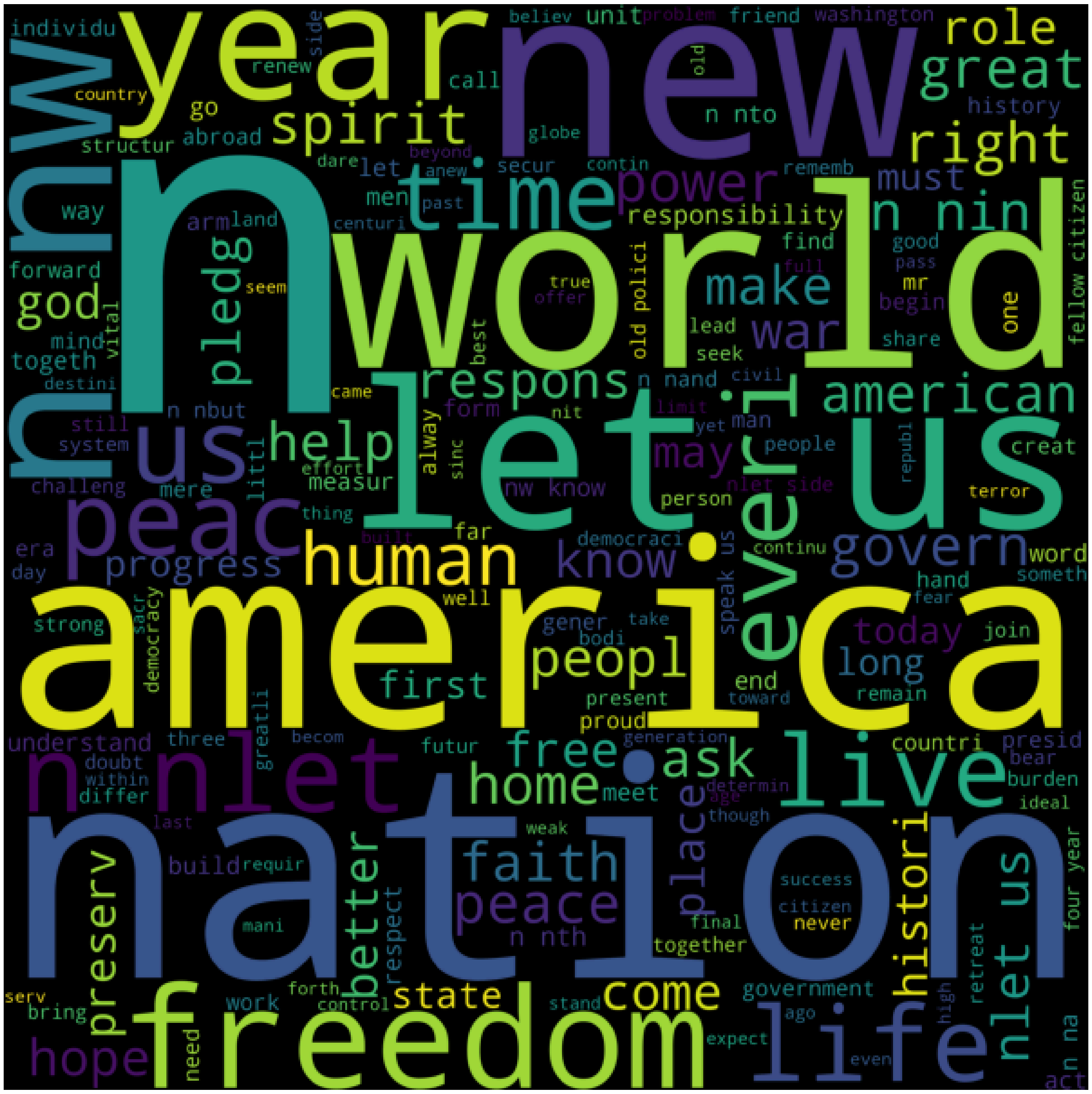
In [39]:
```
wc_all = ''.join(df['Speech'])
```

In [40]:
```
# Word Cloud
wordcloud = WordCloud(width = 3000, height = 3000,
                background_color ='black',
                min_font_size = 10, random_state=100).generate(wc_all)

# plot the WordCloud image
plt.figure(figsize = (6, 6), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.xlabel('Word Cloud')
plt.tight_layout(pad = 0)

plt.title("Fig 4: Word Cloud for all three speeches")
plt.show()
```

Fig 4: Word Cloud for all three speeches



```
In [ ]:
```