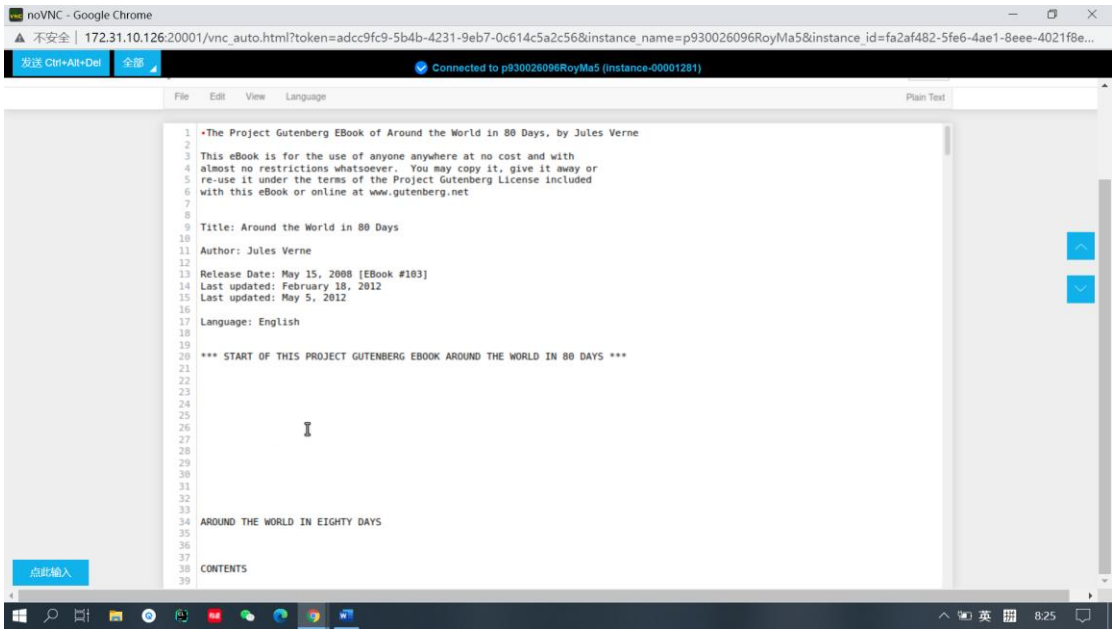
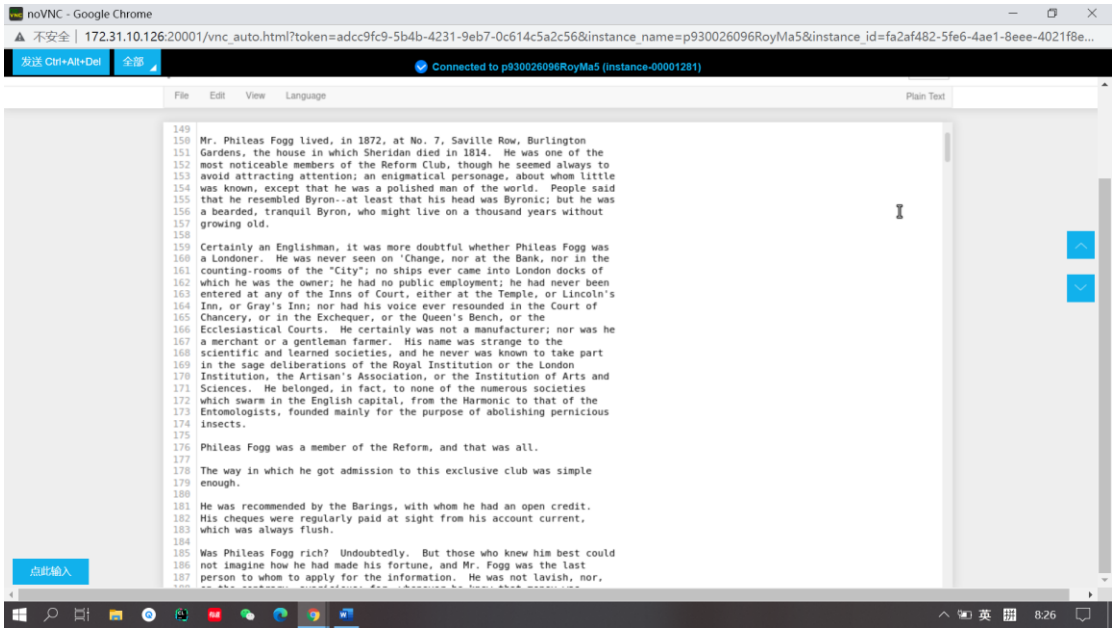
Intermediate



The English text file.



The content of English text file.

Using Spark, Stopwords, RegularExpression, Matplotlib for Word Cloud

In [2]:
```
import matplotlib.pyplot as plt
import re
from wordcloud import WordCloud, STOPWORDS
from pyspark import SparkConf, SparkContext
```

Using Spark, stop words and regular expression, matplotlib for word cloud.

Create Spark Context and load text file

```
In [3]: sc = SparkContext.getOrCreate(SparkConf())
        text_file = sc.textFile('103.txt')
        print(text_file)
```

```
103.txt MapPartitionsRDD[1] at textFile at NativeMethodAccessorImpl.java:0
```

Create Spark context and load text file.

Word Count, sort by key

```
In [4]: counts = text_file.flatMap(lambda x: ((v) for v in re.split('\\W',x))) \
                    .map(lambda word: (word, 1)).reduceByKey(lambda a, b: a + b)
        output = counts.sortByKey().collectAsMap()
        print(output)
```

```
{'': 18494, '0': 1, '000': 3, '1': 48, '103': 4, '10th': 1, '11': 2, '117': 2, '11th': 7, '12th': 3, '13': 2, '13th
': 3, '14th': 4, '15': 1, '1500': 1, '158': 1, '15th': 1, '16th': 1, '170': 1, '1756': 1, '17th': 2, '18': 1, '1814
': 1, '1825': 1, '1837': 1, '1839': 1, '1842': 1, '1843': 1, '1845': 1, '1849': 1, '1853': 1, '1862': 1, '1867': 1,
'1872': 2, '1887': 1, '18th': 1, '19th': 1, '2': 4, '20': 3, '2001': 1, '2008': 1, '2012': 2, '20th': 5, '21st': 1
2, '22': 1, '22nd': 2, '23': 1, '23rd': 4, '25th': 3, '280': 1, '28th': 1, '2nd': 6, '3': 11, '30': 1, '30th': 2, '
35': 1, '3rd': 3, '4': 7, '40': 2, '400': 1, '43': 2, '45': 1, '4557': 1, '48': 1, '4th': 2, '5': 6, '50': 1, '501
': 2, '596': 1, '5th': 3, '6': 4, '60': 1, '6221541': 1, '64': 1, '6th': 3, '7': 9, '7th': 3, '8': 6, '80': 6, '801
': 1, '809': 1, '80th': 1, '84116': 1, '8th': 1, '9': 4, '90': 2, '99712': 1, '9th': 3, 'A': 106, 'ABOUT': 2, 'ACCE
PT': 2, 'ACROBATIC': 1, 'ACROSS': 2, 'ACTUAL': 1, 'AGREE': 2, 'AGREEMENT': 1, 'AIDS': 2, 'AK': 1, 'AMERICAN': 2, 'A
N': 2, 'AND': 17, 'ANTIPODES': 2, 'ANY': 3, 'ANYBODY': 2, 'ANYTHING': 1, 'APPEARS': 2, 'ARE': 4, 'AROUND': 5, 'AS':
7, 'ASCII': 2, 'ASTOUNDS': 2, 'AT': 12, 'ATTRACTION': 1, 'Abandoning': 1, 'About': 4, 'Above': 2, 'Abraham': 1, 'Ab
solute': 1, 'Absolutely': 1, 'According': 2, 'Additional': 1, 'Aden': 8, 'Admitted': 1, 'Admitting': 1, 'Africa':
3, 'African': 1, 'After': 17, 'Ages': 1, 'Agile': 1, 'Agra': 1, 'Agreed': 2, 'Ah': 22, 'Aha': 1, 'Ahmehnagara': 1,
'Alabama': 2, 'Alas': 1, 'Albemarle': 3, 'All': 15, 'Allahabad': 16, 'Alongside': 1, 'Already': 1, 'Although': 1, '
Always': 2, 'America': 15, 'American': 30, 'Americans': 7, 'Among': 3, 'Amphion': 1, 'An': 15, 'Anam': 1, 'And': 6
4, 'Andaman': 2, 'Andrew': 14, 'Angelica': 1, 'Anglo': 2, 'Another': 1, 'Any': 1, 'Aouda': 136, 'Apiece': 2, 'Arabi
c': 2, 'Arabs': 1, 'Archive': 13, 'Are': 7, 'Arkansas': 2, 'Armenian': 1, 'Armenians': 1, 'Around': 4, 'Arrived':
3, 'Articles': 1, 'Artisan': 1, 'Arts': 1, 'As': 42, 'Asia': 4, 'Asian': 1, 'Asphaltite': 1, 'Ass': 1, 'Association
': 1, 'Assurghur': 1, 'At': 66, 'Athens': 1, 'Atlantic': 15, 'Attired': 1, 'Auburn': 1, 'Aureng': 1, 'Aurungabad':
```

Word count and sorted by key.

Get the list of words

```
In [5]: wordlist = output.keys()
        print(wordlist)
```

```
dict_keys(['', '0', '000', '1', '103', '10th', '11', '117', '11th', '12th', '13', '13th', '14th', '15', '1500', '15
8', '15th', '16th', '170', '1756', '17th', '18', '1814', '1825', '1837', '1839', '1842', '1843', '1845', '1849', '1
853', '1862', '1867', '1872', '1887', '18th', '19th', '2', '20', '2001', '2008', '2012', '20th', '21st', '22', '22n
d', '23', '23rd', '25th', '280', '28th', '2nd', '3', '30', '30th', '35', '3rd', '4', '40', '400', '43', '45', '4557
', '48', '4th', '5', '50', '501', '596', '5th', '6', '60', '6221541', '64', '6th', '7', '7th', '8', '80', '801', '8
09', '80th', '84116', '8th', '9', '90', '99712', '9th', 'A', 'ABOUT', 'ACCEPT', 'ACROBATIC', 'ACROSS', 'ACTUAL', 'A
GREE', 'AGREEMENT', 'AIDS', 'AK', 'AMERICAN', 'AN', 'AND', 'ANTIPODES', 'ANY', 'ANYBODY', 'ANYTHING', 'APPEARS', 'A
RE', 'AROUND', 'AS', 'ASCII', 'ASTOUNDS', 'AT', 'ATTRACTION', 'Abandoning', 'About', 'Above', 'Abraham', 'Absolute
', 'Absolutely', 'According', 'Additional', 'Aden', 'Admitted', 'Admitting', 'Africa', 'African', 'After', 'Ages',
'Agile', 'Agra', 'Agreed', 'Ah', 'Aha', 'Ahmehnagara', 'Alabama', 'Alas', 'Albemarle', 'All', 'Allahabad', 'Alongsi
de', 'Already', 'Although', 'Always', 'America', 'American', 'Americans', 'Among', 'Amphion', 'An', 'Anam', 'And',
'Andaman', 'Andrew', 'Angelica', 'Anglo', 'Another', 'Any', 'Aouda', 'Apiece', 'Arabic', 'Arabs', 'Archive', 'Are',
'Arkansas', 'Armenian', 'Armenians', 'Around', 'Arrived', 'Articles', 'Artisan', 'Arts', 'As', 'Asia', 'Asian', 'As
phaltite', 'Ass', 'Association', 'Assurghur', 'At', 'Athens', 'Atlantic', 'Attired', 'Auburn', 'Aureng', 'Aurungaba
d', 'Author', 'B', 'BAD', 'BAG', 'BANKNOTES', 'BATULCAR', 'BE', 'BEAUTIFUL', 'BECOMES', 'BEFORE', 'BETRAYS', 'BRAVE
', 'BREACH', 'BUSINESS', 'BUT', 'BY', 'Bab', 'Bah', 'Bank', 'Banks', 'Banyans', 'Banyas', 'Baring', 'Barings', 'Bar
num', 'Battery', 'Batulcar', 'Bay', 'Be', 'Because', 'Before', 'Behar', 'Behind', 'Being', 'Below', 'Benares', 'Ben
ch', 'Bengal', 'Benten', 'Besides', 'Bets', 'Between', 'Biblical', 'Birmingham', 'Bitter', 'Blondin', 'Bluffs', 'Bo
mbay', 'Bonds', 'Book', 'Boots', 'Bordeaux', 'Boulevard', 'Bow', 'Bradshaw', 'Brahma', 'Brahmin', 'Brahmins', 'Braz
```

Get the list of words.

Remove stopwords, you can also add your own stopwords

In [6]:
```
stopwords = set(STOPWORDS)
#new_words = ['ebook', 'gutenberg', 'https']
#stopwords.update(new_words)
print(stopwords)
```

```
{'because', 'all', 'could', "how's", "she'd", 'who', 'why', "he'd", 'own', "shan't", 'here', 'herself', 'just', 'the
m', 'his', 'during', 'doing', 'did', "didn't", "shouldn't", 'we', 'about', 'out', "they'd", 'ought', 'its', "she's",
'http', 'by', "when's", 'be', "haven't", 'for', 'our', 'through', 'at', 'then', "isn't", 'down', 'than', 'some', "ha
sn't", 'there', 'whom', 'between', 'further', "you're", "aren't", 'any', 'a', 'each', "mustn't", "wasn't", "we'll",
"weren't", 'how', 'too', 'what', "why's", 'shall', "i'm", 'in', 'an', "couldn't", "who's", "you'd", 'not', "let's",
'otherwise', 'other', 'was', "can't", 'from', 'yourself', 'under', "we're", 'or', 'ourselves', "i'll", 'being', 'the
', 'cannot', "he'll", "i've", 'to', 'themselves', "i'd", "there's", 'had', 'has', "what's", 'are', "doesn't", "they'
ll", 'would', 'www', 'also', 'should', 'have', 'which', 'against', 'i', 'you', 'having', 'like', 'my', 'of', 'am', '
until', 'get', 'same', 'hers', 'so', 'again', 'such', 'hence', "you'll", 'few', 'below', 'however', 'myself', 'while
', "don't", 'as', 'off', 'with', 'once', 'where', "wouldn't", 'do', 'he', 'they', "you've", 'yourselves', 'since', '
can', 'when', "they're", 'itself', "that's", "therefore", "here's", 'ours', "we've", 'that', 'these', 'no', 'r', "he
's", "she'll", 'been', 'if', "it's", 'were', 'on', 'else', 'does', 'more', 'ever', 'com', 'theirs', 'but', 'k', 'nor
', 'both', 'very', 'over', 'only', 'their', 'himself', "we'd", 'me', 'after', 'above', 'him', 'this', 'most', 'is',
'those', 'your', 'she', 'before', 'into', 'up', "won't", 'and', "where's", 'yours', 'it', "hadn't", "they've", 'her
'}
```

Remove stop words, and stop words can be updated by using function update.

Generate and show word cloud

In [7]:
```
wordcloud = WordCloud(stopwords=stopwords, width=1600, height=800, background_color='white')\
            .generate(" ".join(wordlist))

plt.figure(figsize=(20,10))
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
```

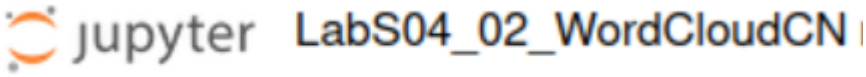
Generate and show word cloud.

You can also include a picture mask for the result

In [8]:
```
import numpy as np
from PIL import Image
from wordcloud import ImageColorGenerator

background_image = np.array(Image.open('heart.jpg'))
#background_image = np.array(Image.open('balloon.jpg'))

wordcloud = WordCloud(stopwords=stopwords, mask=background_image, width=1600, height=800, background_color='white')\
            .generate(" ".join(wordlist))

plt.figure(figsize=(20,10))

image_colors = ImageColorGenerator(background_image)
plt.imshow(wordcloud.recolor(color_func=image_colors))

plt.axis('off')
plt.show()
```

Include a picture mask for the result.

The word cloud included a picture.



Run the word cloud for own Chinese text file.

Using Jieba, Stopwords, Matplotlib for Chinese Word Cloud

In [1]:
```
import jieba
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS
```

Load text file and stopwords file, use Jieba for cutting

In [7]:
```
textFile = 'douduodalu.txt'

text = open(textFile).read()
swFile = open('baidu_stopwords.txt')

wordlist = jieba.cut(text)

stopwords = set(STOPWORDS)
for line in swFile:
    stopwords.add(line.strip('\n'))
```

Import libraries, load text file, stop words file and use Jieba for cutting.

Generate and show word cloud, you may need a Chinese font

```
In [4]: wordcloud = WordCloud(stopwords=stopwords, width=1600, height=800, background_color='white', \
                              font_path='NotoSansCJK-Regular-1.otf').generate(" ".join(wordlist))

        plt.figure(figsize=(20,10))
        plt.imshow(wordcloud)
        plt.axis('off')
        plt.show()
```

```
Building prefix dict from the default dictionary ...
Dumping model to file cache /tmp/jieba.cache
Loading model cost 0.858 seconds.
Prefix dict has been built successfully.
```

Generate and show word cloud by using a Chinese font.

You can also include a picture mask for the result

```
In [8]: import numpy as np
        from PIL import Image
        from wordcloud import ImageColorGenerator

        background_image = np.array(Image.open('heart.jpg'))
        #background_image = np.array(Image.open('balloon.jpg'))

        wordcloud = WordCloud(stopwords=stopwords, mask=background_image, width=1252, height=1144, background_color='white', \
                              font_path='NotoSansCJK-Regular-1.otf').generate(" ".join(wordlist))

        plt.figure(figsize=(20,10))

        image_colors = ImageColorGenerator(background_image)
        plt.imshow(wordcloud.recolor(color_func=image_colors))

        plt.axis('off')
        plt.show()
```

Include a picture for the word cloud.

Word cloud with a included picture.

The word ranking from Jieba analyse should be quite the same, you can include the weights for words

In [9]:
```
import jieba.analyse

with open(textFile, 'r', encoding = 'utf-8') as file:
    text = file.readlines()
    keywords = jieba.analyse.extract_tags(str(text), topK = 300, withWeight=True)
    print(keywords)
```

```
[('唐三', 0.16566952239628238), ('武魂', 0.07118377983166513), ('小舞', 0.05890037250405411), ('海神', 0.040239491751961454), ('自己', 0.03567626478989935), ('史莱克', 0.03558323190887187), ('魂力', 0.034059723927515904), ('魂师', 0.032573196926085 27), ('已经', 0.0310156129341668), ('戴沐白', 0.030056532792084287), ('身体', 0.02980345642503064), ('魂环', 0.027996258526943586), ('斗罗', 0.026092460788269267), ('大师', 0.024943238374206374), ('学院', 0.022924335418680577), ('他们', 0.02247165145154045), ('宁荣荣', 0.02176327688936602), ('没有', 0.021366778791257986), ('昊天', 0.021064724839033502), ('魂技', 0.02010723365093014), ('奥斯卡', 0.01993579139128269), ('马红俊', 0.017942641701478516), ('此时', 0.017421378766133658), ('弗兰德', 0.017280809640077353), ('身上', 0.01679001401229908), ('实力', 0.016273788891385116), ('之中', 0.015766772089122295), ('七怪', 0.015152143646161364), ('一个', 0.015089682092282479), ('能够', 0.014894811006991361), ('我们', 0.014814870094246545), ('虽然', 0.014652743002929198), ('蓝银草', 0.014565335667191243), ('光芒', 0.014526513041754894), ('攻击', 0.014445427005279336), ('唐昊', 0.014226324197901935), ('比比', 0.0135688968569727), ('天斗', 0.01331354446018137), ('泰坦', 0.013254260403845017), ('封号', 0.0131047079178764), ('修炼', 0.013103624927074923), ('时候', 0.013098308277221674), ('感觉', 0.012749979565214896), ('千仞', 0.012739797196471302), ('就是', 0.012720044337170977), ('什么', 0.012675617080577 49), ('金色', 0.012604788771194 54), ('帝国', 0.012415713212802312), ('知道', 0.012319430005421048), ('对手', 0.012314615655125366), ('赵无极', 0.012176524352498537), ('独孤', 0.012173848220848126), ('三叉戟', 0.012097924132069313), ('只是', 0.012035270544755 71), ('不是', 0.011913469211821811), ('你们', 0.011744592717822855), ('瞬间', 0.011420520655162593), ('朱竹清', 0.011253270195040669), ('同时', 0.011222398611023586), ('技能', 0.011019882368696625), ('精神力', 0.010761406573249914), ('有些', 0.010637714957137294), ('那么', 0.010543467668175877), ('如果', 0.010174915552760 84), ('魂兽', 0.010157934509775991), ('竟然', 0.009931406960742226), ('这个', 0.009906674777603396), ('蓝银', 0.00983194174630436), ('现在', 0.009763649351145782), ('释放', 0.009685136733104591), ('看着', 0.009663241987111213), ('唐门', 0.009662425509299114), ('出现', 0.009640985283827 28), ('就算', 0.009599807961181 33), ('胡列娜', 0.009597226956604787), ('看到', 0.009514215067541723), ('还是', 0.009396104347926443), ('顿时', 0.009389765798605273), ('魂骨', 0.009349472456366349), ('心中', 0.009255658194767653), ('变得', 0.009189667579567223), ('十万年', 0.009036683017714936), ('完全', 0.008912186457741944), ('能量', 0.008796481688203277), ('领域', 0.008571857376910396), ('眼中', 0.008403705643011213), ('强大', 0.008163617107051442), ('无法', 0.008149477110505052), ('蓝银皇', 0.008136779376251886), ('八蛛', 0.008110699955174154), ('暗器', 0.008089435062387027), ('一声', 0.008025813432463981), ('风致', 0.008024554686730542), ('起来', 0.007989247876761104), ('一下', 0.007887503356724 53), ('只有', 0.007864808059371137), ('气息', 0.007855008758038005),
```

Rank the word by Jieba Analyse.

You can also generate WordCloud from weights/keywords of JIeba

In [10]:
```
word_dict = {}
for i in keywords:
    word_dict[i[0]]=i[1]

wordcloud = WordCloud(stopwords=stopwords, width=1600, height=800, background_color='white', \
                      font_path='NotoSansCJK-Regular-1.otf').fit_words(word_dict)

plt.figure(figsize=(20,10))
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
```

Generate word cloud from weights of JIeba.

Word cloud with weights