In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
# importing data to use
df= pd.read_csv('tmdb.movies.csv')
df
```

Out[2]:

	Unnamed: 0	genre_ids	id	original_language	original_title	popularity	release_date	
0	0	[12, 14, 10751]	12444	en	Harry Potter and the Deathly Hallows: Part 1	33.533	2010-11-19	Harry P th Hallov
1	1	[14, 12, 16, 10751]	10191	en	How to Train Your Dragon	28.734	2010-03-26	How to T
2	2	[12, 28, 878]	10138	en	Iron Man 2	28.515	2010-05-07	Ir
3	3	[16, 35, 10751]	862	en	Toy Story	28.005	1995-11-22	
4	4	[28, 878, 12]	27205	en	Inception	27.920	2010-07-16	
...	...	...	...	...	...	...	...	
26512	26512	[27, 18]	488143	en	Laboratory Conditions	0.600	2018-10-13	La C
26513	26513	[18, 53]	485975	en	_EXHIBIT_84xxx_	0.600	2018-05-01	_EXHIBIT
26514	26514	[14, 28, 12]	381231	en	The Last One	0.600	2018-10-01	The
26515	26515	[10751, 12, 28]	366854	en	Trailer Made	0.600	2018-06-22	Tra
26516	26516	[53, 27]	309885	en	The Church	0.600	2018-10-05	Th

26517 rows × 10 columns

In [3]:
```
#printing out first 5 rows
print(df.head())
```

```
   Unnamed: 0            genre_ids     id original_language  \
0           0      [12, 14, 10751]  12444                en   
1           1  [14, 12, 16, 10751]  10191                en   
2           2        [12, 28, 878]  10138                en   
3           3      [16, 35, 10751]    862                en   
4           4        [28, 878, 12]  27205                en   

                                 original_title  popularity release_date  \
0  Harry Potter and the Deathly Hallows: Part 1      33.533   2010-11-19   
1                      How to Train Your Dragon      28.734   2010-03-26   
2                                    Iron Man 2      28.515   2010-05-07   
3                                     Toy Story      28.005   1995-11-22   
4                                     Inception      27.920   2010-07-16   

                                          title  vote_average  vote_count  
0  Harry Potter and the Deathly Hallows: Part 1           7.7       10788  
1                      How to Train Your Dragon           7.7        7610  
2                                    Iron Man 2           6.8       12368  
3                                     Toy Story           7.9       10174  
4                                     Inception           8.3       22186  
```

In [4]:
```
#printing out last 5 rows
print(df.tail())
```

```
       Unnamed: 0          genre_ids      id original_language  \
26512       26512           [27, 18]  488143                en   
26513       26513           [18, 53]  485975                en   
26514       26514       [14, 28, 12]  381231                en   
26515       26515    [10751, 12, 28]  366854                en   
26516       26516           [53, 27]  309885                en   

              original_title  popularity release_date                 title  \
26512  Laboratory Conditions         0.6   2018-10-13  Laboratory Conditions   
26513         _EXHIBIT_84xxx_         0.6   2018-05-01        _EXHIBIT_84xxx_   
26514            The Last One         0.6   2018-10-01           The Last One   
26515            Trailer Made         0.6   2018-06-22           Trailer Made   
26516              The Church         0.6   2018-10-05             The Church   

       vote_average  vote_count  
26512           0.0           1  
26513           0.0           1  
26514           0.0           1  
26515           0.0           1  
26516           0.0           1  
```

In [5]:
```
#getting statistical summary of the data
print(df.describe())
```

```
        Unnamed: 0             id    popularity  vote_average    vote_count
count  26517.00000   26517.000000  26517.000000  26517.000000  26517.000000
mean   13258.00000  295050.153260      3.130912      5.991281    194.224837
std     7654.94288  153661.615648      4.355229      1.852946    960.961095
min        0.00000      27.000000      0.600000      0.000000      1.000000
25%     6629.00000  157851.000000      0.600000      5.000000      2.000000
50%    13258.00000  309581.000000      1.374000      6.000000      5.000000
75%    19887.00000  419542.000000      3.694000      7.000000     28.000000
max    26516.00000  608444.000000     80.773000     10.000000  22186.000000
```

In [6]:
```
# printing out how many rows and columns are there
print(df.shape)
```

```
(26517, 10)
```

In [7]:
```
#getting general information on the columns
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26517 entries, 0 to 26516
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Unnamed: 0         26517 non-null  int64  
 1   genre_ids          26517 non-null  object 
 2   id                 26517 non-null  int64  
 3   original_language  26517 non-null  object 
 4   original_title     26517 non-null  object 
 5   popularity         26517 non-null  float64
 6   release_date       26517 non-null  object 
 7   title              26517 non-null  object 
 8   vote_average       26517 non-null  float64
 9   vote_count         26517 non-null  int64  
dtypes: float64(2), int64(3), object(5)
memory usage: 2.0+ MB
None
```

In [8]:
```
#renaming first column
df.rename(columns= {df.columns[0]:'Number'}, inplace= True)
```

In [9]:
```
#printing out first 5 rows again to confirm column name change
print(df.head())
```

```
   Number             genre_ids     id original_language  \
0       0      [12, 14, 10751]  12444                en
1       1  [14, 12, 16, 10751]  10191                en
2       2        [12, 28, 878]  10138                en
3       3      [16, 35, 10751]    862                en
4       4        [28, 878, 12]  27205                en

                                 original_title  popularity release_date  \
0  Harry Potter and the Deathly Hallows: Part 1      33.533   2010-11-19
1                      How to Train Your Dragon      28.734   2010-03-26
2                                    Iron Man 2      28.515   2010-05-07
3                                     Toy Story      28.005   1995-11-22
4                                     Inception      27.920   2010-07-16

                                          title  vote_average  vote_count
0  Harry Potter and the Deathly Hallows: Part 1           7.7       10788
1                      How to Train Your Dragon           7.7        7610
2                                    Iron Man 2           6.8       12368
3                                     Toy Story           7.9       10174
4                                     Inception           8.3       22186
```

In [10]:
```
#checking for duplicates
duplicates= df[df.duplicated()]
print(len(duplicates))
```

```
0
```

In [11]:
```
# deleting the genre_ids column
df.drop('genre_ids', axis=1, inplace=True)
```

In [12]:
```
#printing first 5 rows to confirm column is deleted
print(df.head())
```

```
   Number     id original_language  \
0       0  12444                en
1       1  10191                en
2       2  10138                en
3       3    862                en
4       4  27205                en

                                 original_title  popularity release_date  \
0  Harry Potter and the Deathly Hallows: Part 1      33.533   2010-11-19
1                      How to Train Your Dragon      28.734   2010-03-26
2                                    Iron Man 2      28.515   2010-05-07
3                                     Toy Story      28.005   1995-11-22
4                                     Inception      27.920   2010-07-16

                                          title  vote_average  vote_count
0  Harry Potter and the Deathly Hallows: Part 1           7.7       10788
1                      How to Train Your Dragon           7.7        7610
2                                    Iron Man 2           6.8       12368
3                                     Toy Story           7.9       10174
4                                     Inception           8.3       22186
```

In [13]:
```
#checking current number of columns
print(df.shape)
```

```
(26517, 9)
```

In [14]:
```
#checking for extraeous values
for col in df.columns:
    print(col, '\n', df[col].value_counts(normalize=True).head(), '\n\n')
```

```
Number
 Number
0        0.000038
17675    0.000038
17685    0.000038
17684    0.000038
17683    0.000038
Name: proportion, dtype: float64


id
 id
380718    0.000113
292086    0.000113
402448    0.000113
192137    0.000113
514791    0.000113
Name: proportion, dtype: float64


original_language
 original_language
en    0.878342
fr    0.019120
es    0.017159
ru    0.011238
ja    0.009994
Name: proportion, dtype: float64


original_title
 original_title
Eden            0.000264
Home            0.000226
Legend          0.000189
Aftermath       0.000189
Truth or Dare   0.000189
Name: proportion, dtype: float64


popularity
 popularity
0.600    0.265377
1.400    0.024475
0.840    0.022137
0.624    0.003922
0.625    0.003469
Name: proportion, dtype: float64


release_date
 release_date
2010-01-01    0.010144
2011-01-01    0.007542
2014-01-01    0.005845
2012-01-01    0.005845
2013-01-01    0.005468
Name: proportion, dtype: float64


title
 title
Eden         0.000264
Home         0.000264
Lucky        0.000189
Legend       0.000189
Aftermath    0.000189
```

```
Name: proportion, dtype: float64


vote_average
 vote_average
6.0     0.073161
7.0     0.058830
5.0     0.056040
10.0    0.047215
8.0     0.046423
Name: proportion, dtype: float64


vote_count
 vote_count
1    0.246672
2    0.114794
3    0.066259
4    0.050798
5    0.036543
Name: proportion, dtype: float64
```

In [15]:
```
df.dtypes
```

Out[15]:
```
Number                 int64
id                     int64
original_language     object
original_title        object
popularity           float64
release_date          object
title                 object
vote_average         float64
vote_count             int64
dtype: object
```

In [16]:
```
df.isna().sum() #Checking for null values
```

Out[16]:
```
Number               0
id                   0
original_language    0
original_title       0
popularity           0
release_date         0
title                0
vote_average         0
vote_count           0
dtype: int64
```

In [17]:
```
df.iloc[1000:1050,2] #  sourcing rows from 1000 to 1500 for the original_langauge colum
```

Out[17]:
```
1000    en
1001    en
1002    en
1003    en
1004    ru
1005    en
1006    nl
1007    en
1008    en
1009    en
1010    en
1011    no
1012    en
1013    en
1014    en
1015    en
1016    en
1017    en
1018    en
1019    de
1020    en
1021    en
1022    en
1023    et
1024    en
1025    en
1026    en
1027    it
1028    da
1029    en
1030    en
1031    en
1032    no
1033    de
1034    en
1035    en
1036    nl
1037    tl
1038    es
1039    en
1040    en
1041    en
1042    fr
1043    en
1044    en
1045    lt
1046    en
1047    fr
1048    en
1049    te
Name: original_language, dtype: object
```

In [18]:
```
#how many times each language appears in the data set
df['original_language'].value_counts()
```

Out[18]:
```
original_language
en    23291
fr      507
es      455
ru      298
ja      265
      ...
bo        1
si        1
sl        1
hz        1
dz        1
Name: count, Length: 76, dtype: int64
```

In [19]:
```
ne_languagecount = (df['original_language'] == 'ne').sum()  # how many times ne languag
ne_languagecount
```

Out[19]:
```
2
```

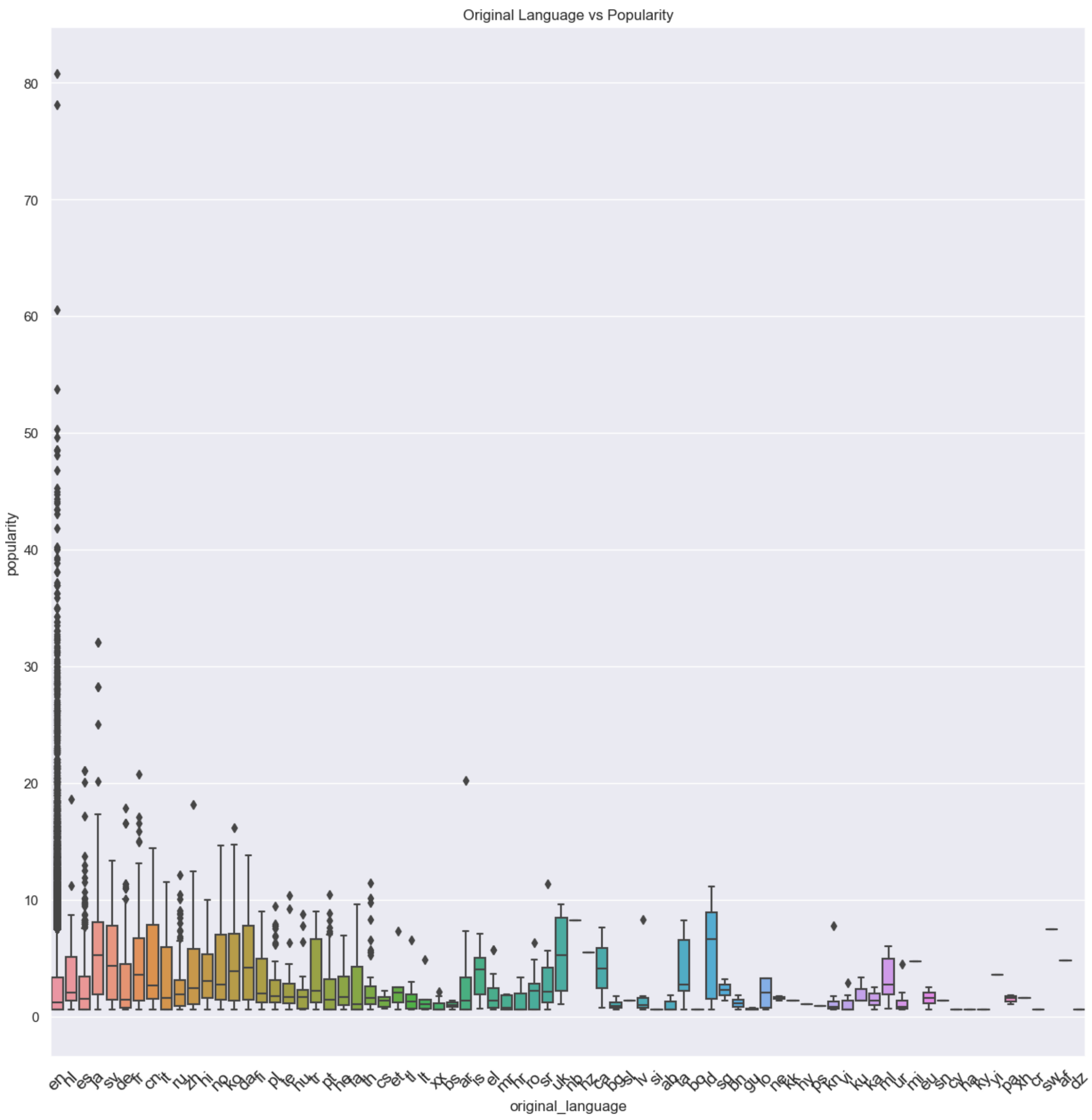
In [20]:
```
sampled_df = df.sample(frac=0.2) #creating a sample data frame from the original
sampled_df
```

Out[20]:

	Number	id	original_language	original_title	popularity	release_date	title	vote_average
20107	20107	521317	en	Hope	0.600	2016-03-07	Hope	7.0
10347	10347	245230	en	I Am Britney Jean	0.600	2013-12-22	I Am Britney Jean	6.8
26082	26082	534169	en	Penny Palabras	0.748	2018-05-19	Penny Palabras	8.0
20277	20277	423342	en	Feeding Time	0.600	2016-10-21	Feeding Time	6.0
11217	11217	243935	en	Rob the Mob	9.175	2014-03-21	Rob the Mob	6.3
...	...	...	...	...	...	...	...	...
22308	22308	456101	en	Aaron's Blood	1.265	2017-06-02	Aaron's Blood	4.8
10264	10264	439755	en	Wake-up Juice	0.600	2013-02-05	Wake-up Juice	7.0
11011	11011	186242	en	Re-Emerging: The Jews of Nigeria	0.600	2013-05-17	Re-Emerging: The Jews of Nigeria	0.5
18467	18467	325186	en	Smothered	2.198	2016-03-29	Smothered	3.3
7379	7379	135847	en	The Transmission	0.600	2012-09-21	The Transmission	6.5

5303 rows × 9 columns

In [25]:
```
#creating box plot for language and popularity
plt.figure(figsize=(15, 15)) #setting desired figure size
sns.boxplot(x='original_language', y='popularity', data=df)
plt.title('Original Language vs Popularity')
plt.xticks(rotation=45, fontsize= 14)  # rotate x values at an angle for them to stop be
plt.show()
```



```
In conclussion, the movies created in English are most popular followed by japanese.
Spanish and french movies follow in close range.
```

In [51]:
```
origlanguage_frequency = df['original_language'].value_counts()
df['origlanguage_frequency'] = df['original_language'].map(origlanguage_frequency)
df.head()
```

Out[51]:

	Number	id	original_language	original_title	popularity	release_date	title	vote_average	vote_cc
0	0	12444	en	Harry Potter and the Deathly Hallows: Part 1	33.533	2010-11-19	Harry Potter and the Deathly Hallows: Part 1	7.7	10
1	1	10191	en	How to Train Your Dragon	28.734	2010-03-26	How to Train Your Dragon	7.7	7
2	2	10138	en	Iron Man 2	28.515	2010-05-07	Iron Man 2	6.8	12
3	3	862	en	Toy Story	28.005	1995-11-22	Toy Story	7.9	10
4	4	27205	en	Inception	27.920	2010-07-16	Inception	8.3	22

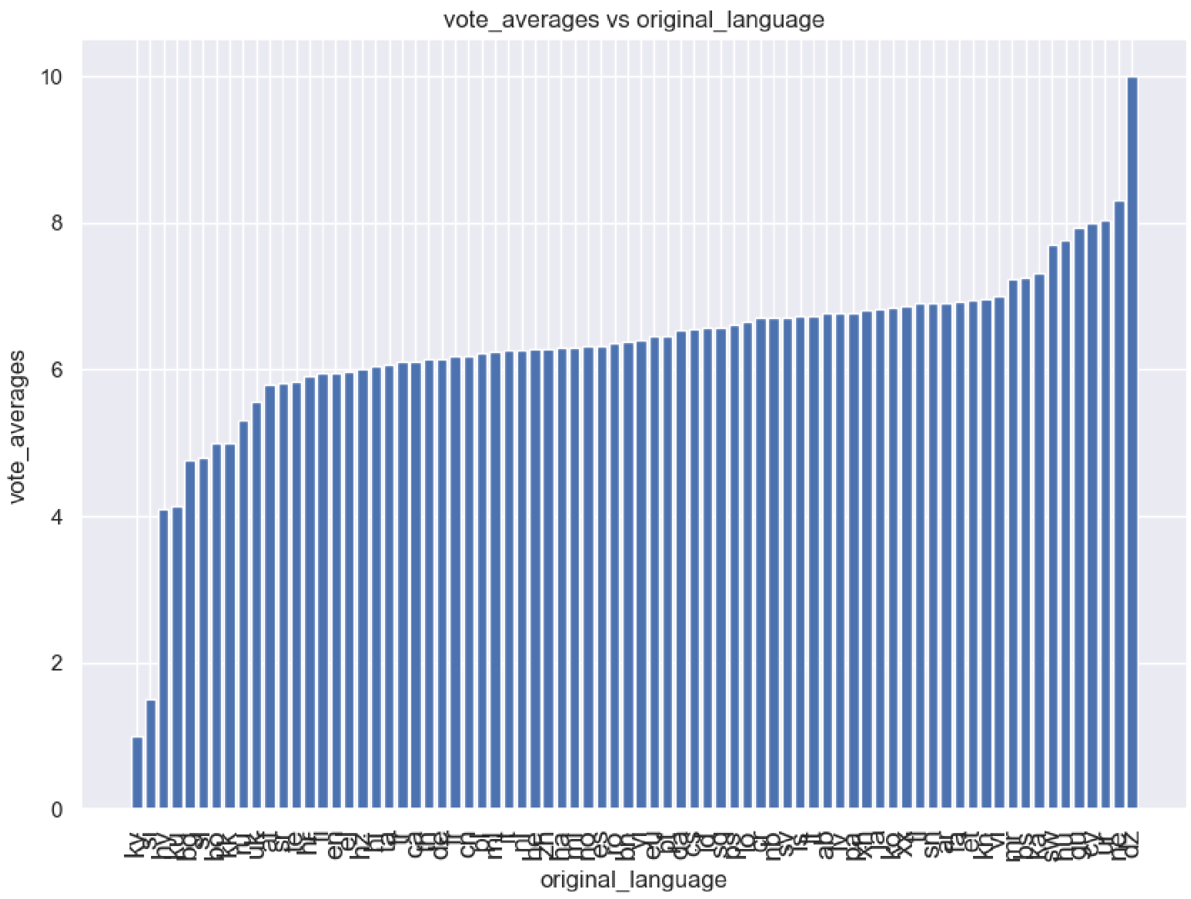
Movies made in English are by far more frequently made than movies in any other language.

In [28]:
```
plt.style.available
```

Out[28]:
```
['Solarize_Light2',
 '_classic_test_patch',
 '_mpl-gallery',
 '_mpl-gallery-nogrid',
 'bmh',
 'classic',
 'dark_background',
 'fast',
 'fivethirtyeight',
 'ggplot',
 'grayscale',
 'seaborn-v0_8',
 'seaborn-v0_8-bright',
 'seaborn-v0_8-colorblind',
 'seaborn-v0_8-dark',
 'seaborn-v0_8-dark-palette',
 'seaborn-v0_8-darkgrid',
 'seaborn-v0_8-deep',
 'seaborn-v0_8-muted',
 'seaborn-v0_8-notebook',
 'seaborn-v0_8-paper',
 'seaborn-v0_8-pastel',
 'seaborn-v0_8-poster',
 'seaborn-v0_8-talk',
 'seaborn-v0_8-ticks',
 'seaborn-v0_8-white',
 'seaborn-v0_8-whitegrid',
 'tableau-colorblind10']
```

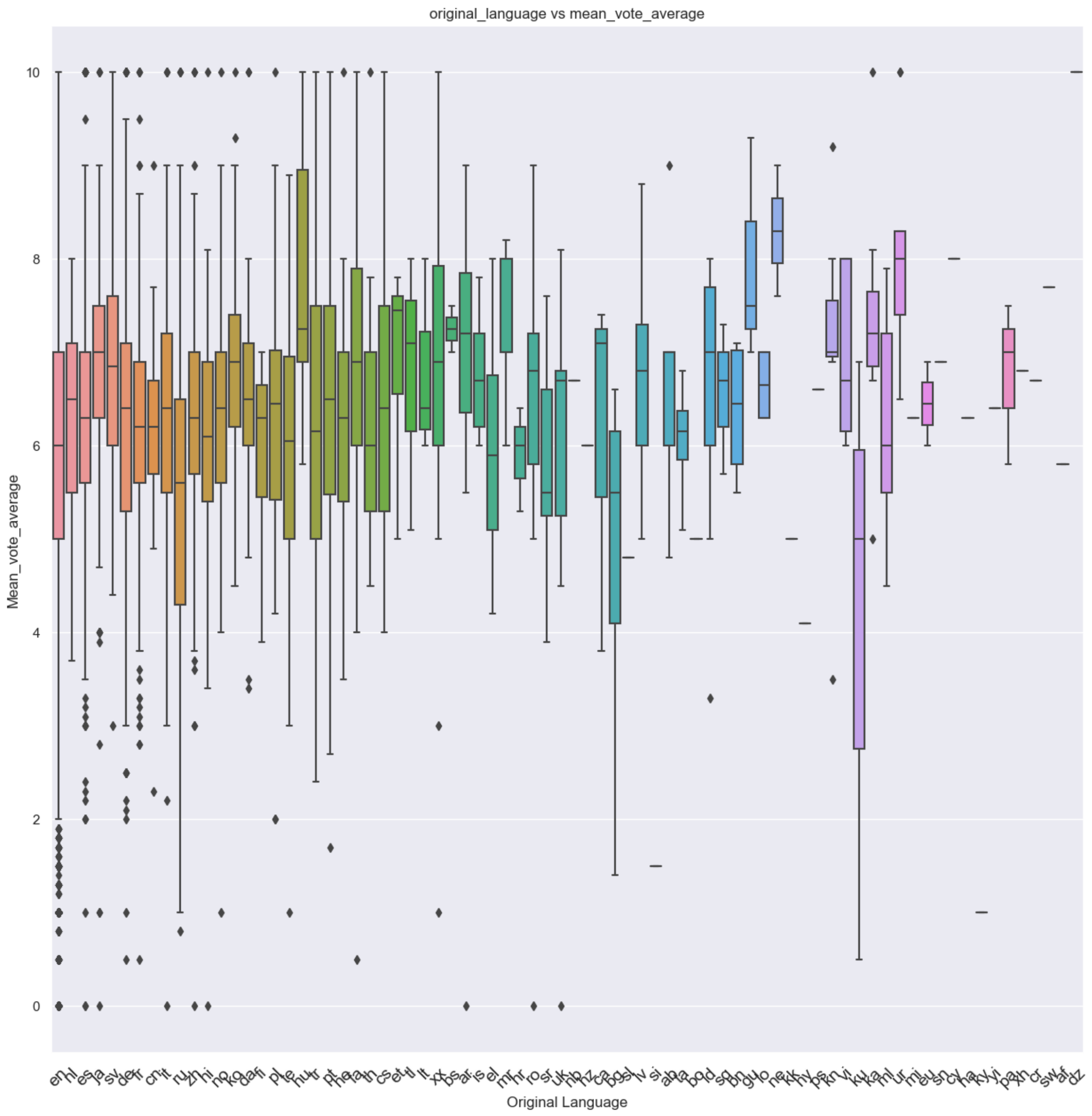
In [40]:
```
#plotting according the the mean of average votes for original_languages
vote_averages = df.groupby('original_language')['vote_average'].mean().reset_index()
vote_averages = vote_averages.sort_values(by='vote_average')
plt.figure(figsize=(10, 7))

plt.bar(vote_averages['original_language'], vote_averages['vote_average'])
plt.xlabel('original_language')
plt.ylabel('vote_averages')
plt.title('vote_averages vs original_language')
plt.xticks(rotation=90, fontsize=14)
plt.show()
```



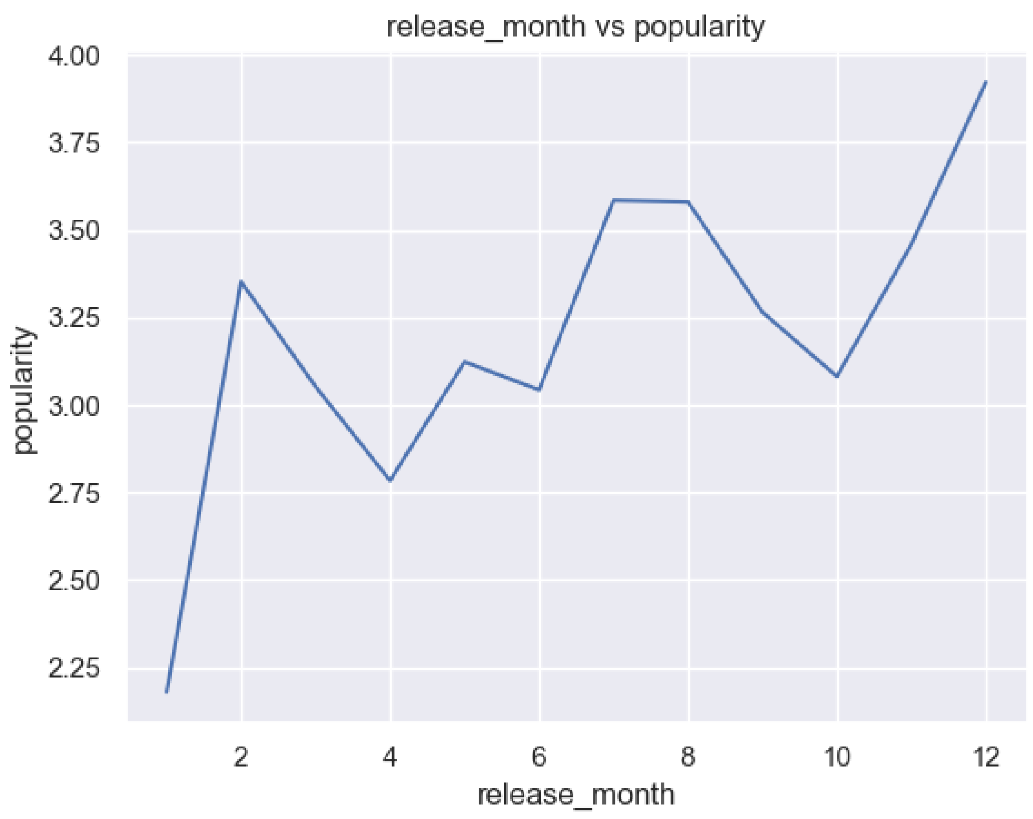
However, this may not be the most accurate way foward as the mean vote_averages tends to be bigger for langauges that appear few times and smaller for languages that appear the more often.

In [31]:
```
# Plot for vote_average mean and language
plt.figure(figsize=(15,15))
sns.boxplot(x='original_language',y='vote_average', data=df)
plt.xlabel('Original Language')
plt.ylabel('Mean_vote_average')
plt.title('original_language vs mean_vote_average')
plt.xticks(rotation=45, fontsize=14)
plt.show()
```



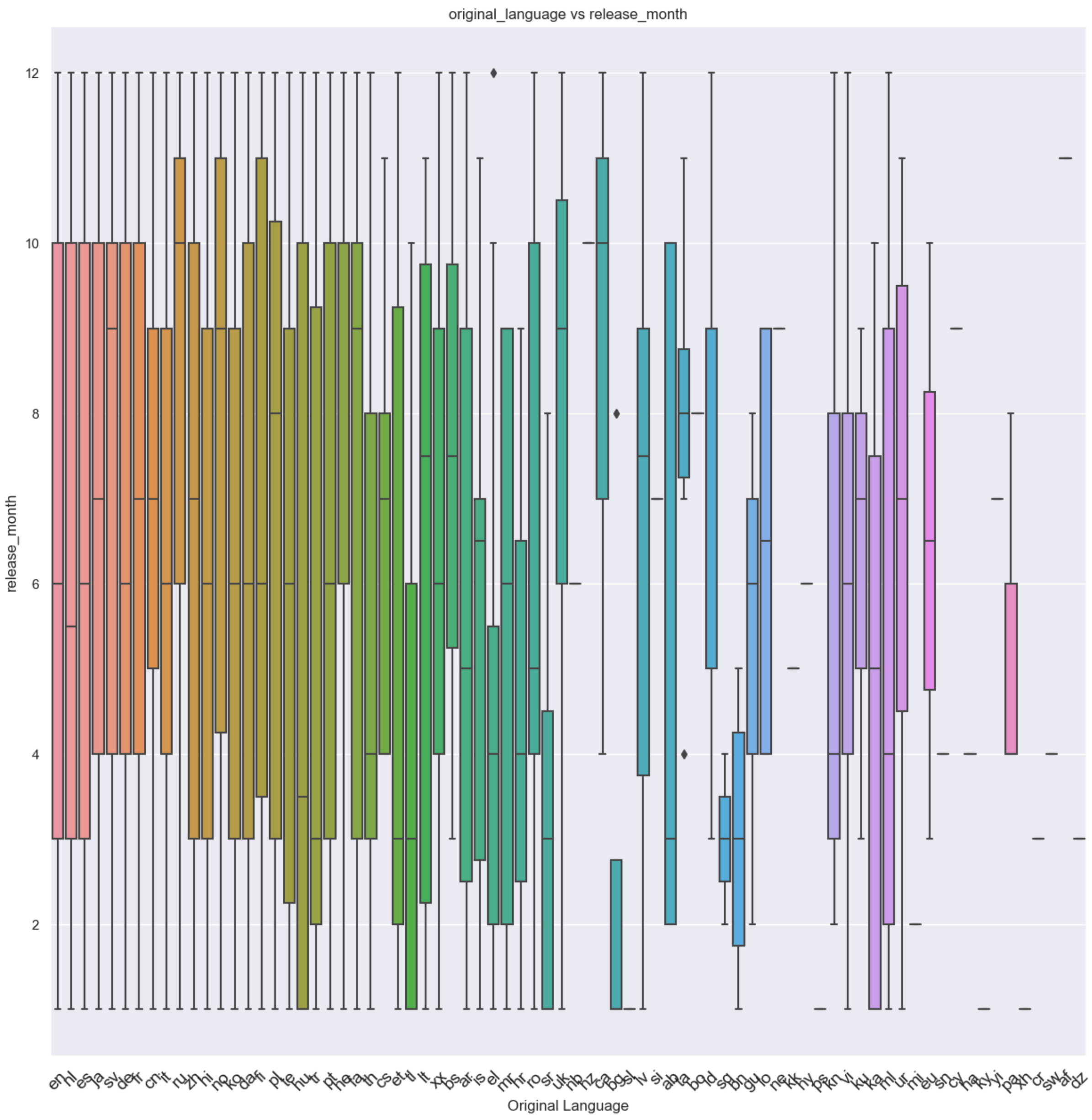
In [32]:
```
df['release_date'] = pd.to_datetime(df['release_date']) # ensuring release_date is in da
df['release_month'] = df['release_date'].dt.month # grouping the release dates by month
release_month_groups = df.groupby('release_month')
monthly_popularity = release_month_groups['popularity'].mean() # finding monthly mean fo

#plotting to see trend
plt.plot(monthly_popularity.index, monthly_popularity.values)
plt.xlabel('release_month')
plt.ylabel('popularity')
plt.title('release_month vs popularity')
plt.show()
```



In [35]:
```
#making sure the release date is in the datetime format
df['release_date'] = pd.to_datetime(df['release_date'])

#changing release_date to monthly format.
df['release_month'] = df['release_date'].dt.month
plt.figure(figsize=(15,15))
sns.boxplot(x='original_language',y='release_month', data=df)
plt.xlabel('Original Language')
plt.ylabel('release_month')
plt.title('original_language vs release_month')
plt.xticks(rotation=45, fontsize=14)
plt.show()
```



For most languages,they peak on the 12th month, December, making it the best time of the year to release movies.