In [7]:
```
!pip install cutecharts
```

```
Requirement already satisfied: cutecharts in d:\anaconda\lib\site-packages (1.2.0)
Requirement already satisfied: jinja2 in d:\anaconda\lib\site-packages (from cutecharts) (2.11.3)
Requirement already satisfied: MarkupSafe>=0.23 in d:\anaconda\lib\site-packages (from jinja2->cutecha
rts) (1.1.1)
```

In [8]:
```
import cutecharts.charts as ctc
import pandas as pd
```

In [9]:
```
df = pd.read_csv('data/tmdb-movies.csv')
df.head(2)
```

Out[9]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	hom
0	135397	tt0369610	32.985763	150000000	1513528810	Jurassic World	Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...	http://www.jurassicworl
1	76341	tt1392190	28.419936	150000000	378436354	Mad Max: Fury Road	Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...	http://www.madmaxmovi

2 rows × 21 columns

In [10]:
```
df['release_date'] = pd.to_datetime(df['release_date'])
df['popularity'] = round(df['popularity'], 2)
'''
我们先清理数据，然后我们会看到可爱的图表。

将特征更改为正确的日期时间格式并将流行功能四舍五入到小数点后两位，以获得更多内容。
'''
```

Out[10]:
```
'\n我们先清理数据，然后我们会看到可爱的图表。\n\n将特征更改为正确的日期时间格式并将流行功能四舍五入到小数点后两位，以获得更多内容。\n'
```

In [11]:
```
df.drop(['imdb_id', 'homepage', 'budget_adj', 'revenue_adj'],
        axis=1,
        inplace=True)  #删除不必要的特征
```

In [12]:
```
#用missing替换特征nan值
df['tagline'].fillna('missing', inplace=True)
df['keywords'].fillna('missing', inplace=True)
df['production_companies'].fillna('missing', inplace=True)
df['cast'].fillna('missing', inplace=True)
df['director'].fillna('missing', inplace=True)
df['genres'].fillna('missing', inplace=True)
df['overview'].fillna('missing', inplace=True)
```

In [14]:
```
#从预算和收入中删除等于 0 的值。
df.drop(df[(df['budget'] == 0) & (df['revenue'] == 0)].index, inplace=True)
```

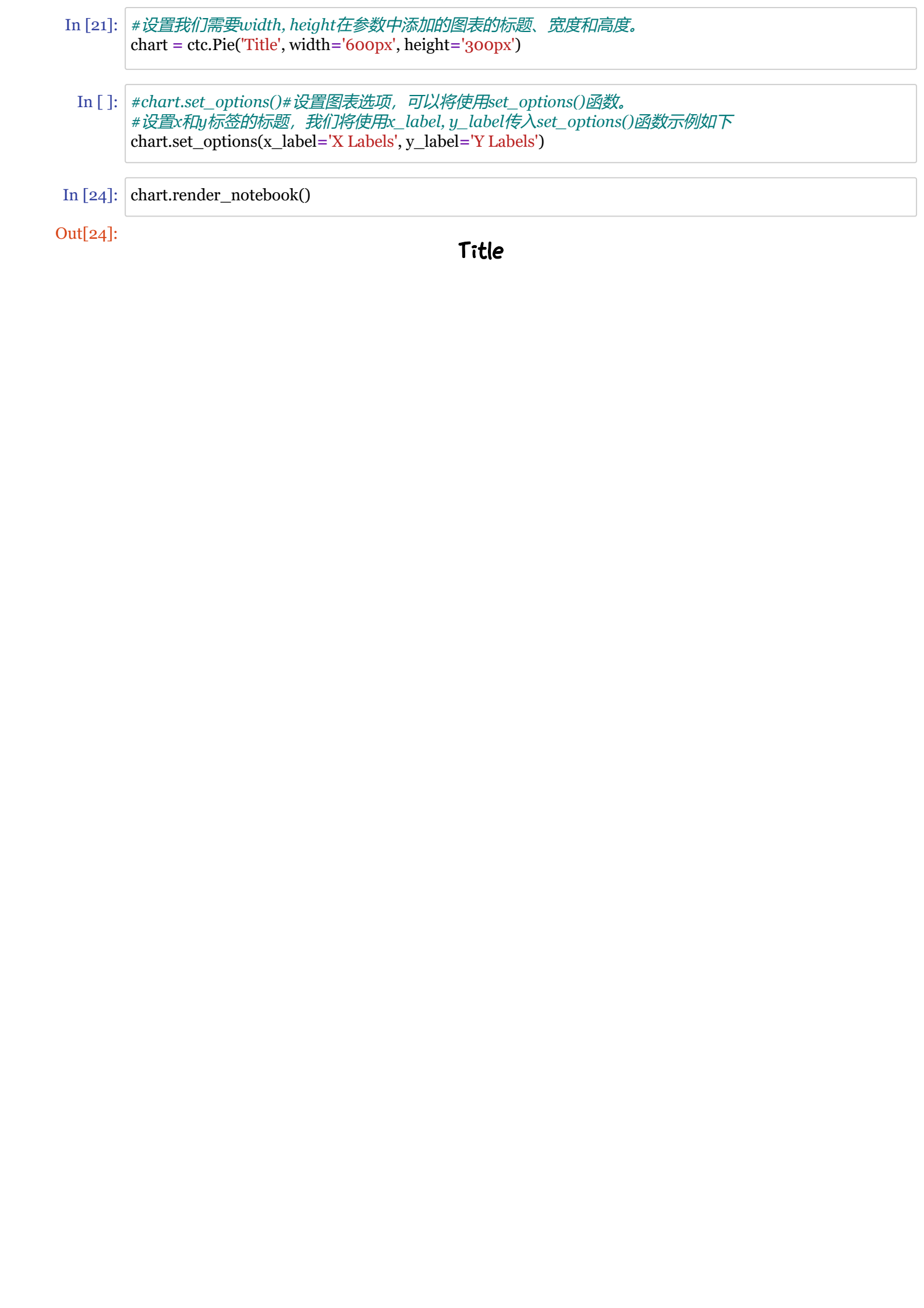
In [20]:
```
chart = ctc.Pie()  #分配你想要的图表名称，例如，你想要一个饼图然后运行下面的代码。
```

In [21]:
```
#设置我们需要width, height在参数中添加的图表的标题、宽度和高度。
chart = ctc.Pie('Title', width='600px', height='300px')
```

In [ ]:
```
#chart.set_options()#设置图表选项，可以将使用set_options()函数。
#设置x和y标签的标题，我们将使用x_label, y_label传入set_options()函数示例如下
chart.set_options(x_label='X Labels', y_label='Y Labels')
```

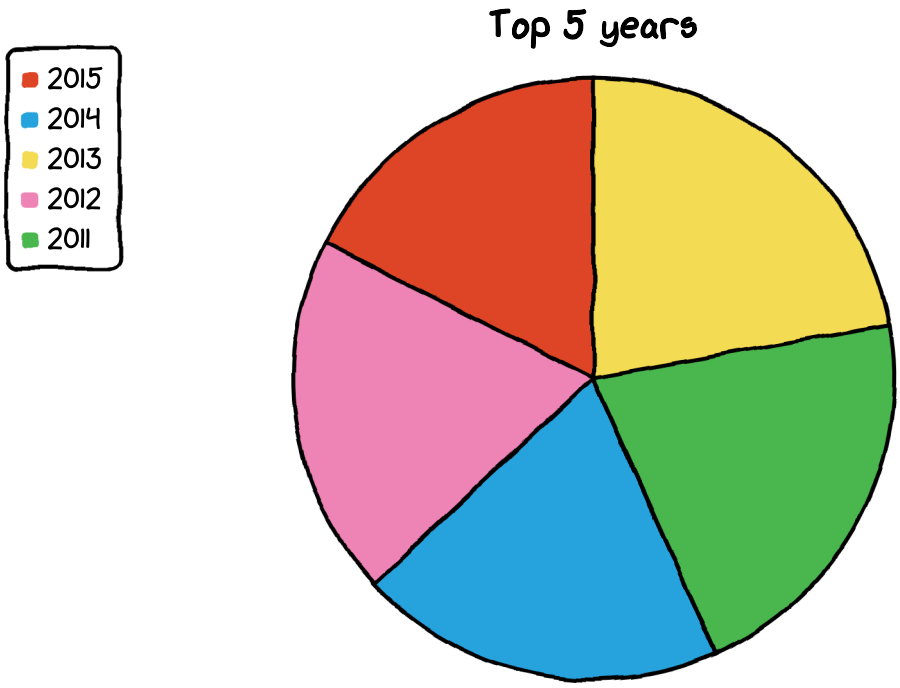
In [24]:
```
chart.render_notebook()
```

Out[24]:



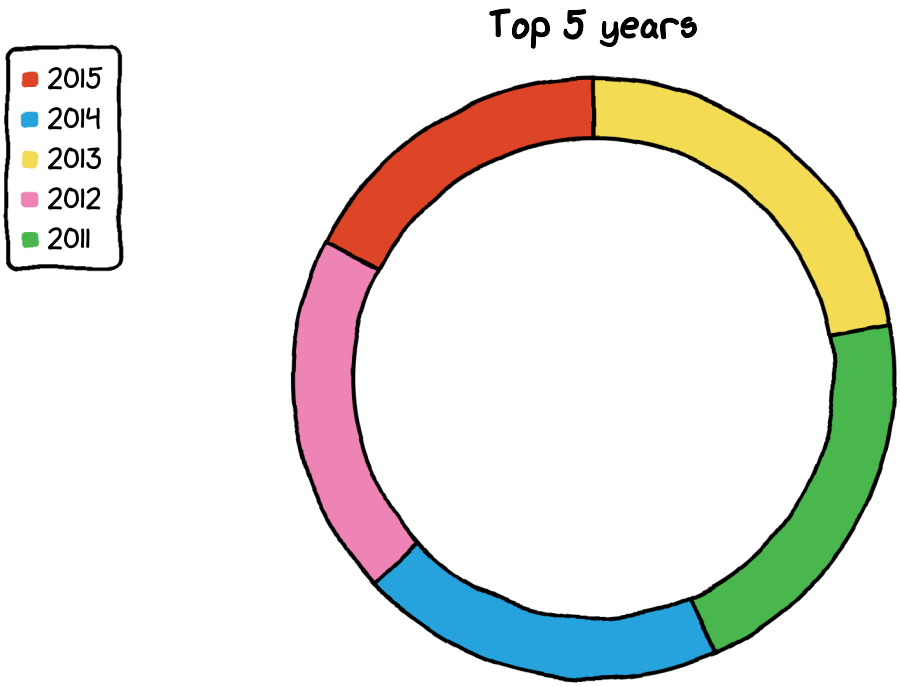
In [25]:
```
#发行量最高的电影的前 5 年
df_year = df['release_year'].value_counts().reset_index().sort_values(
   by='index', ascending=False)[:5].rename(columns={
      'index': 'release_year',
      'release_year': 'Count'
   })
chart = ctc.Pie('Top 5 years', width='600px', height='300px')
chart.set_options(labels=list(df_year['release_year']), inner_radius=0)
chart.add_series(list(df_year['Count']))
chart.render_notebook()
```

Out[25]:



In [26]:
```
df_year = df['release_year'].value_counts().reset_index().sort_values(
   by='index', ascending=False)[:5].rename(columns={
      'index': 'release_year',
      'release_year': 'Count'
   })
chart = ctc.Pie('Top 5 years', width='600px', height='300px')
chart.set_options(labels=list(df_year['release_year']), inner_radius=0.8)
chart.add_series(list(df_year['Count']))
chart.render_notebook()
```

Out[26]:



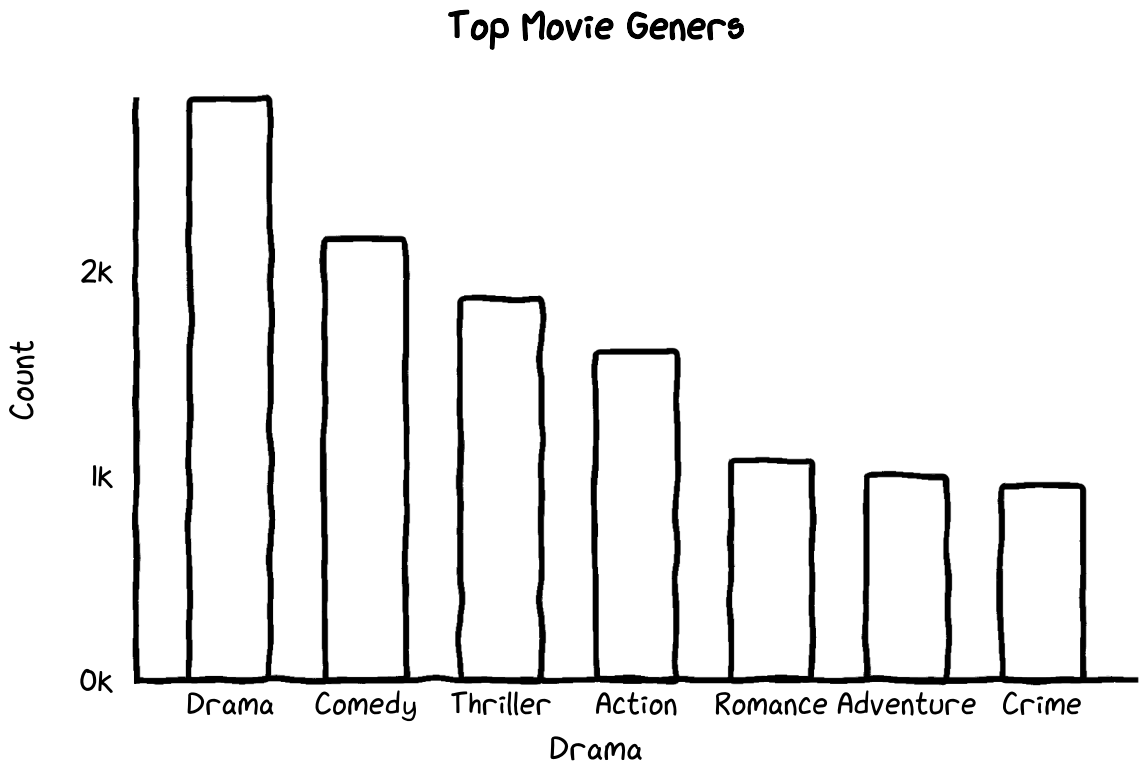
In [30]:
```
#该函数将拆分字符串并返回每个类型的计数。
def count_genre(x):
   data_plot = df[x].str.cat(sep = '|')
   data = pd.Series(data_plot.split('|'))
   info = data.value_counts(ascending=False)
   return info
#调用函数来计算每个类型的电影。
df_genre_movies = count_genre('genres')
df_genre_movies = pd.DataFrame(df_genre_movies).reset_index().rename(columns={'index':'Drama', 0:'C
```

In [31]:
```
#条形图代码
#这里我们通过导入 faker 库为不同的条使用颜色。
from cutecharts.faker import Faker

chart = ctc.Bar('Top Movie Geners', width='600px', height='200px')
chart.set_options(labels=list(df_genre_movies['Drama'][:7]),
x_label='Drama',
y_label='Count',
colors=Faker.colors
)

chart.add_series('Geners',list(df_genre_movies['Count'][:7]))
chart.render_notebook()
```

Out[31]:



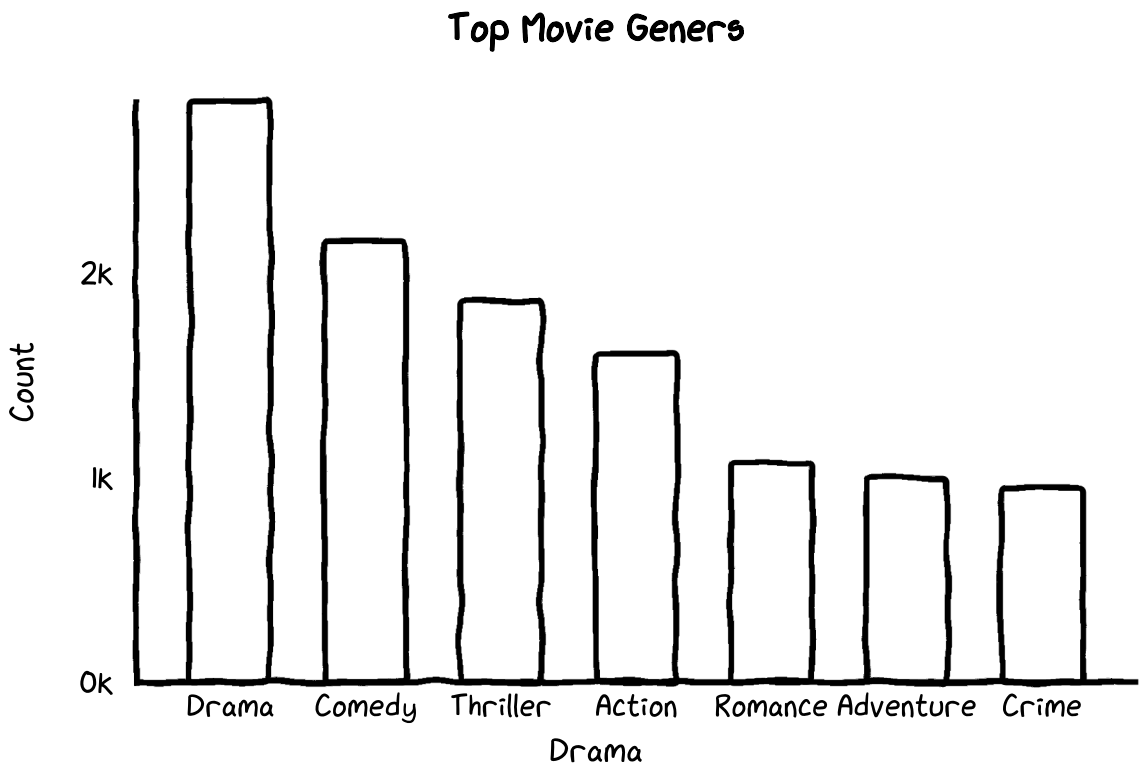
In [33]:
```
#这里我们在可以明确为条形设置不同颜色的地方添加了另一个参数颜色set_options()
chart = ctc.Bar('Top Movie Geners', width='600px', height='200px')

chart.set_options(labels=list(df_genre_movies['Drama'][:7]),
x_label='Drama',
y_label='Count',

colors=['#FFF1C9', '#F7B7A3', '#EA5F89', '#9B3192', '#57167E', '#47B39C','#00529B'])

chart.add_series('Geners',list(df_genre_movies['Count'][:7]))
chart.render_notebook()
```

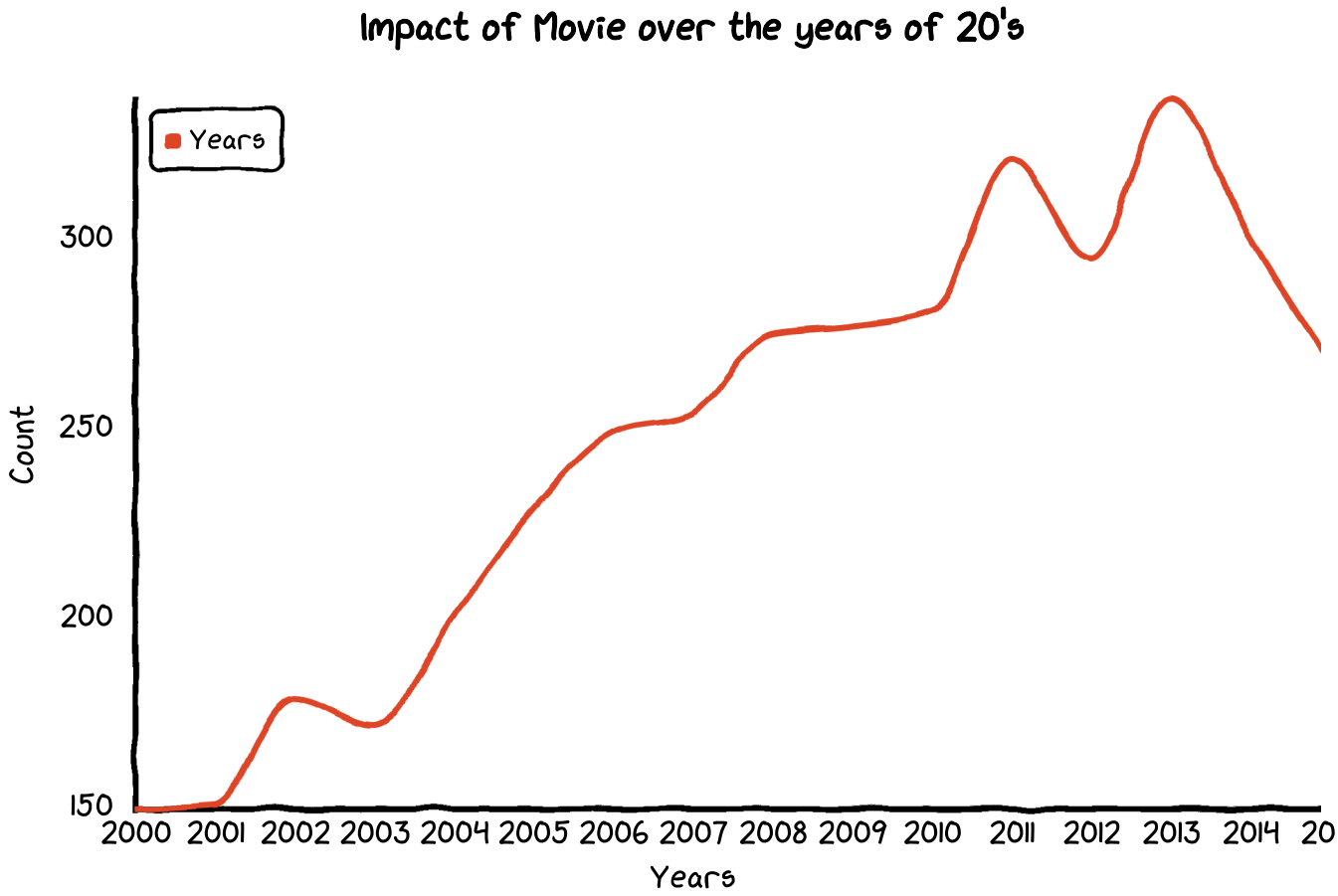
Out[33]:



In [34]:
```
#绘制折线图
#我们将计算多年来发行的电影数量，并仅绘制 20 年代的电影。
data = df.groupby('release_year').count()['id'].reset_index().tail(16)
```

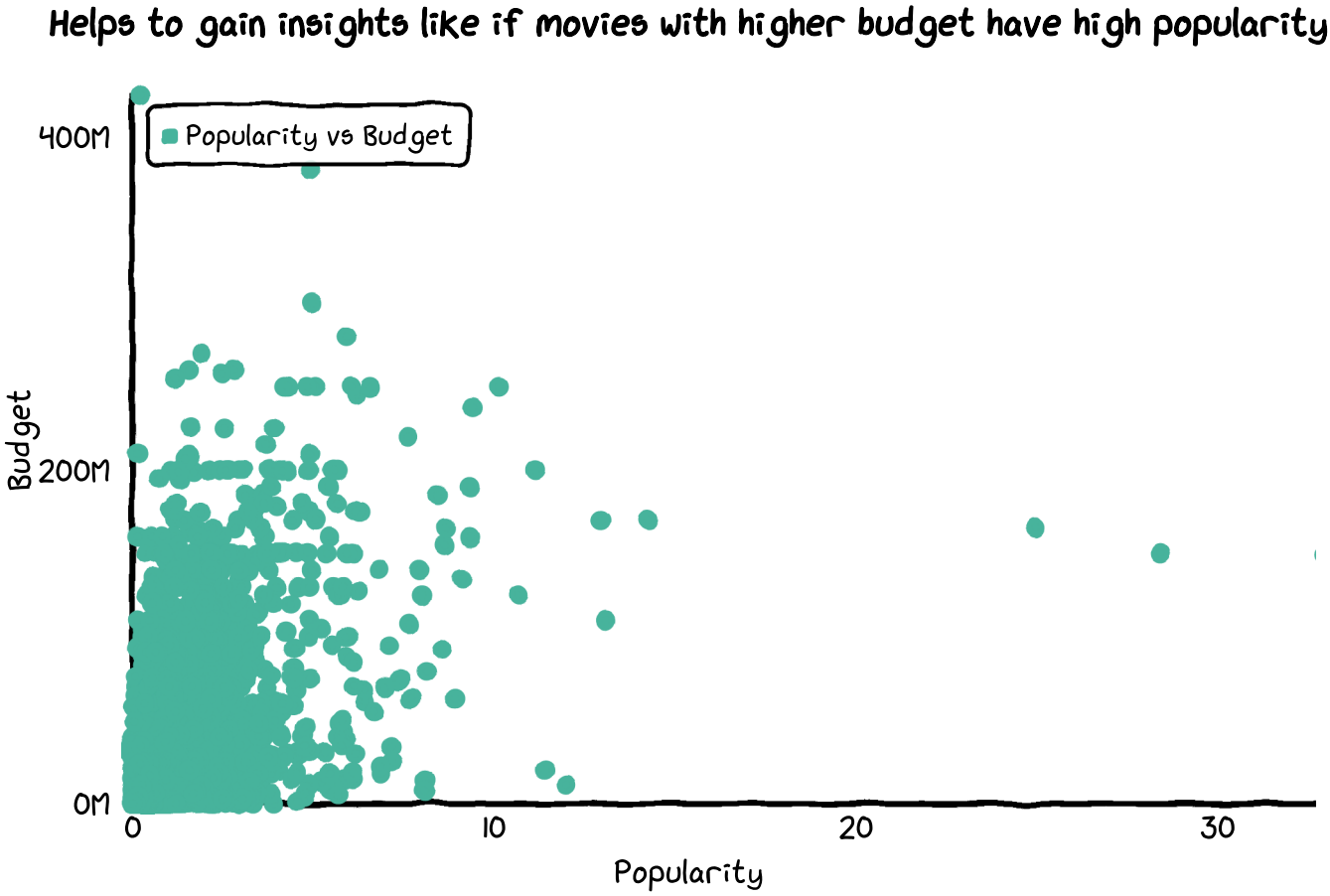
In [35]:
```
chart = ctc.Line("Impact of Movie over the years of 20's", width='700px', height='200px')
chart.set_options(labels=list(data['release_year']),
                  x_label='Years',
                  y_label='Count',
 )
chart.add_series('Years',list(data['id']))
chart.render_notebook()
#多年来，电影发行量呈指数增长。
```

Out[35]:



In [36]:
```
#绘制散点图
#我们看看受欢迎程度与预算的关系，并找出预算是否大于电影的受欢迎程度。
chart = ctc.Scatter("Helps to gain insights like if movies with higher budget have high popularity", width='70
chart.set_options(x_label='Popularity',
                  y_label='Budget',
                  dot_size=1,
                  colors=['#47B39C']
 )
chart.add_series('Popularity vs Budget', [(z[0], z[1]) for z in list(zip(df['popularity'],df['budget']))])
chart.render_notebook()
```

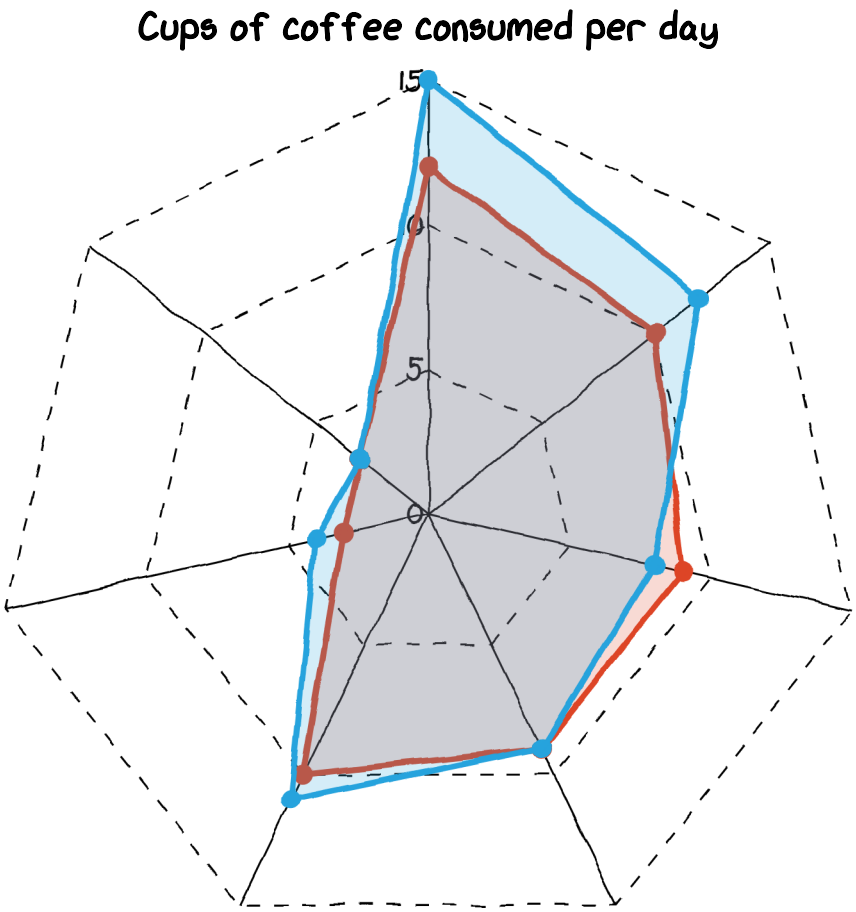
Out[36]:



In [37]:
```
#绘制雷达图
#雷达图也称为蜘蛛图，主要用于显示多变量值。对于雷达图，我们将获取咖啡消费的虚拟数据，其中特征为天
data = {'Day': ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'],
      'This week': [12, 10, 9, 9, 10, 3, 3],
      'Last week': [15, 12, 8, 9, 11, 4, 3]
     }
df_coffee = pd.DataFrame(data, columns = ['Day', 'This week', 'Last week'])
```

In [38]:
```
chart = ctc.Radar('Cups of coffee consumed per day')
chart.set_options(
   labels=list(df_coffee['Day']),
   is_show_legend=True,
   legend_pos='upRight'
   )
chart.add_series('This Week', list(df_coffee['This week']))
chart.add_series('Last Week', list(df_coffee['Last week']))
chart.render_notebook()
```

Out[38]:



In [ ]: