Altair Exercises

This notebook will explore multiple different visualizations in Altair.

Part 3

The following exercise is based on the article by FiveThirtyEight ["America's Favorite 'Star Wars' Movies (And Least Favorite Characters)," (https://fivethirtyeight.com/features/americas-favorite-star-wars-movies-and-least-favorite-characters/)](https://fivethirtyeight.com/features/americas-favorite-star-wars-movies-and-least-favorite-characters/).

In [1]:
```
import pandas as pd
import numpy as np
import altair as alt
import math
```

In [2]:
```
# enable correct rendering
alt.renderers.enable('default')

# uses intermediate json files to speed things up
alt.data_transformers.enable('json')
```

Out[2]:
```
DataTransformerRegistry.enable('json')
```

In [3]:
```
sw = pd.read_csv('../assets/StarWars.csv', encoding='latin1')

# Some format is needed for the survey dataframe, we provide the forma
sw = sw.rename(columns={'Have you seen any of the 6 films in the Star
                        'Do you consider yourself to be a fan of the S
                        'Which of the following Star Wars films have y
                        'Unnamed: 4' : 'seen_EII',
                        'Unnamed: 5' : 'seen_EIII',
                        'Unnamed: 6' : 'seen_EIV',
                        'Unnamed: 7' : 'seen_EV',
                        'Unnamed: 8' : 'seen_EVI',
                        'Please rank the Star Wars films in order of p
                        'Unnamed: 10' : 'rank_EII',
                        'Unnamed: 11' : 'rank_EIII',
                        'Unnamed: 12' : 'rank_EIV',
                        'Unnamed: 13' : 'rank_EV',
                        'Unnamed: 14' : 'rank_EVI',
```

```
                        'Please state whether you view the following c
                        'Unnamed: 16' : 'Luke Skywalker',
                        'Unnamed: 17' : 'Princess Leia Organa',
                        'Unnamed: 18' : 'Anakin Skywalker',
                        'Unnamed: 19' : 'Obi Wan Kenobi',
                        'Unnamed: 20' : 'Emperor Palpatine',
                        'Unnamed: 21' : 'Darth Vader',
                        'Unnamed: 22' : 'Lando Calrissian',
                        'Unnamed: 23' : 'Boba Fett',
                        'Unnamed: 24' : 'C-3P0',
                        'Unnamed: 25' : 'R2 D2',
                        'Unnamed: 26' : 'Jar Jar Binks',
                        'Unnamed: 27' : 'Padme Amidala',
                        'Unnamed: 28' : 'Yoda',
                       })
sw = sw.drop([0])
```

In [4]:
```
# take a peak to look at the data
sw.sample(5)
```

Out[4]:

	RespondentID	seen_any_movie	fan	seen_EI	seen_EII	seen_EIII	seen_EIV	seen_EV
856	3.289534e+09	Yes	Yes	Star Wars: Episode I The Phantom Menace	Star Wars: Episode II Attack of the Clones	Star Wars: Episode III Revenge of the Sith	Star Wars: Episode IV A New Hope	Star Wars: Episode V The Empire Strikes Back
657	3.289997e+09	Yes	Yes	Star Wars: Episode I The Phantom Menace	Star Wars: Episode II Attack of the Clones	Star Wars: Episode III Revenge of the Sith	Star Wars: Episode IV A New Hope	Star Wars: Episode V The Empire Strikes Back
1175	3.288402e+09	Yes	No	Star Wars: Episode I The Phantom Menace	Star Wars: Episode II Attack of the Clones	Star Wars: Episode III Revenge of the Sith	NaN	NaN
458	3.290532e+09	Yes	Yes	Star Wars: Episode I The Phantom	Star Wars: Episode II Attack of the	Star Wars: Episode III Revenge	Star Wars: Episode IV A New	Star Wars: Episode V The Empire

				Phantom Menace	of the Clones	of the Sith	Hope	Strikes Back
387	3.290686e+09	No	NaN	NaN	NaN	NaN	NaN	NaN

5 rows × 38 columns

In [5]:
```
# We're going to fix the labels a bit so will create a mapping to the
episodes = ['EI', 'EII', 'EIII', 'EIV', 'EV', 'EVI']
names = {
    'EI' : 'The Phantom Menance', 'EII' : 'Attack of the Clones', 'EI
    'EIV': 'A New Hope', 'EV': 'The Empire Strikes Back', 'EVI' : 'The
}

# we're also going to use this order to sort, so names_l will now have
names_l = [names[ep] for ep in episodes]

print("sort order: ",names_l)
```

```
sort order:  ['The Phantom Menance', 'Attack of the Clones', 'Revenge
of the Sith', 'A New Hope', 'The Empire Strikes Back', 'The Return of
the Jedi']
```

In [6]:
```
# let's do some data pre-processing... sw (star wars) has everything

# We want to only use those people who have seen at least one movie, l
# and get the total count

# find people who have at least on of the columns (seen_*) not NaN
seen_at_least_one = sw.dropna(subset=['seen_' + ep for ep in episodes]
total = len(seen_at_least_one)

print("total who have seen at least one: ", total)
```

```
total who have seen at least one:  835
```

In [7]:
```
# for each movie, we're going to calculate the percents and generate a
percs = []

# loop over each column and calculate the number of people who have se
# specifically, filter out the people who are *NaN* for a specific epi
# and divide by the percent
for seen_ep in ['seen_' + ep for ep in episodes]:
    perc = len(seen_at_least_one[~ pd.isna(seen_at_least_one[seen_ep])
    percs.append(perc)

# at this point percs is holding our percentages

# now we're going use a trick to make tuples--pairing names with perce
tuples = list(zip([names[ep] for ep in episodes],percs))
seen_per_df = pd.DataFrame(tuples, columns = ['Name', 'Percentage'])
seen_per_df
```

Out[7]:

	Name	Percentage
0	The Phantom Menance	0.805988
1	Attack of the Clones	0.683832
2	Revenge of the Sith	0.658683
3	A New Hope	0.726946
4	The Empire Strikes Back	0.907784
5	The Return of the Jedi	0.883832

In [8]:
```
# ok, time to make the chart... let's make a bar chart (use mark_bar)
bars = alt.Chart(seen_per_df).mark_bar(size=20).encode(
    # encode x as the percent, and hide the axis
    x=alt.X(
        'Percentage',
        axis=None),
    y=alt.Y(
        # encode y using the name, use the movie name to label the axi
        'Name:N',
         axis=alt.Axis(tickCount=5, title=''),
         # we give the sorting order to avoid alphabetical order
         sort=names_l
    )
)

# at this point we don't really have a great plot (it's missing the ar
bars
```

Out[8]:

In [9]:

```
text = bars.mark_text(
    align='left',
    baseline='middle',
    dx=3  # Nudges text to right so it doesn't appear on top of the ba
).encode(
    # we'll use the percentage as the text
    text=alt.Text('Percentage:Q',format='.0%')
)

# finally, we're going to combine the bars and the text and do some st
seen_movies = (text + bars).configure_mark(
    # we don't love the blue
    color='#008fd5'
).configure_view(
    # we don't want a stroke around the bars
    strokeWidth=0
).configure_scale(
    # add some padding
    bandPaddingInner=0.2
).properties(
    # set the dimensions of the visualization
    width=500,
    height=180
).properties(
    # add a title
    title="Which 'Star Wars' Movies Have you Seen?"
)

seen_movies

# note that we are NOT formatting this in the Five Thirty Eight Style
```

Out[9]:

What's the best 'Star Wars' Movie?

In [10]:
```
# find people who have seen all of the movies (seen_*) not NaN
seen_all_six = sw.dropna(subset=['seen_' + ep for ep in episodes],how=
total = len(seen_all_six)

# for each movie, calculate percents and generate a new data frame
percs = []

# loop over each column and calculate the number of people who have a
for rank_ep1 in ['rank_' + ep for ep in episodes]:
    perc = len(seen_all_six[seen_all_six[rank_ep1] == '1']) / total
    percs.append(perc)

tuples = list(zip([names[ep] for ep in episodes],percs))
ranked_no1 = pd.DataFrame(tuples, columns = ['Name', 'Percentage'])
ranked_no1
```

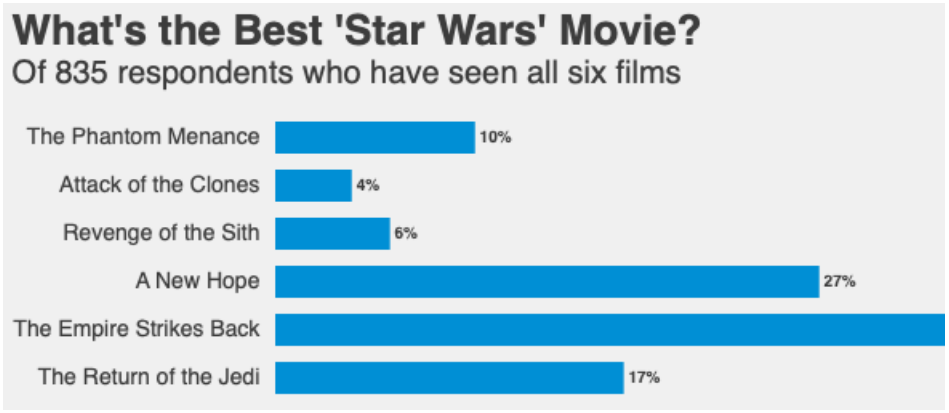
Out[10]:

	Name	Percentage
0	The Phantom Menance	0.099788
1	Attack of the Clones	0.038217
2	Revenge of the Sith	0.057325
3	A New Hope	0.271762
4	The Empire Strikes Back	0.358811
5	The Return of the Jedi	0.174098

In [11]:
```
# Let's make a new chart to match 538
bars2 = alt.Chart(ranked_no1).mark_bar(size=20).encode(
    # encode x as the percent, and hide the axis
    x=alt.X(
        'Percentage',
        axis=None),
    y=alt.Y(
        # encode y using the name, use the movie name to label the axi
        'Name:N',
         axis=alt.Axis(ticks=False, title='',domain=False),
         # we give the sorting order to avoid alphabetical order
         sort=names_l,
    )
)
text2 = bars2.mark_text(
    align='left',
    baseline='middle',
    dx=3,  # Nudges text to right so it doesn't appear on top of the b
    font='Helvetica',
    color='#3C3C3C'
).encode(
    # we'll use the percentage as the text
    text=alt.Text('Percentage:Q',format='.0%')
)
```

```
# finally, we're going to combine the bars and the text and do some st
best_movies = (text2 + bars2).configure(background='#F0F0F0').configur
    # we don't want a stroke around the bars
    strokeWidth=0
).configure_scale(
    # add some padding
    bandPaddingInner=0.2
).configure_axisY(
    labelPadding= 10,
    labelFontSize=14,
    labelColor='#3C3C3C'
).configure_view(
    strokeWidth=0
).properties(
    title={
      "text": ["What's the Best 'Star Wars' Movie?"],
      "subtitle": ["Of 835 respondents who have seen all six films"],
      "color": '#3C3C3C',
      "subtitleColor": '#3C3C3C',
        "subtitleFontSize":20,
    },
    # set the dimensions of the visualization
    width=500,
    height=180
).configure_mark(
    font='Helvetica',
    fontWeight='bold',
    fontSize=10,
    color='#008fd5'
).configure_title(
    anchor='start',
    offset=15,
    font='Helvetica',
    fontWeight='bold',
    fontSize=26,
)

best_movies
```

Out[11]:



How people rate the Star Wars Movies

In [12]:

```
# reorganize the data to separate top, middle, and bottom thirds
total = len(seen_all_six)

perc_top = []
perc_middle = []
perc_bottom = []

for rank in ['rank_' + ep for ep in episodes]:
    perc_top_df = seen_all_six.loc[seen_all_six[rank].isin(['1','2'])]
    per_top = len(perc_top_df) / total
    perc_top.append(float(per_top))
```

```
    perc_middle_df = seen_all_six.loc[seen_all_six[rank].isin(['3','4'
    per_middle = len(perc_middle_df) / total
    perc_middle.append(float(per_middle))

    perc_bottom_df = seen_all_six.loc[seen_all_six[rank].isin(['5','6'
    per_bottom = len(perc_bottom_df) / total
    perc_bottom.append(float(per_bottom))

tups = list(zip([names[ep] for ep in episodes],perc_top,perc_middle,pe
thirds = pd.DataFrame(tups, columns = ['Name', 'Top Third','Middle Thi
thirds
```

Out[12]:

	Name	Top Third	Middle Third	Bottom Third
0	The Phantom Menance	0.163482	0.373673	0.462845
1	Attack of the Clones	0.138004	0.288747	0.573248
2	Revenge of the Sith	0.129512	0.401274	0.467091
3	A New Hope	0.498938	0.309979	0.191083
4	The Empire Strikes Back	0.641189	0.220807	0.138004
5	The Return of the Jedi	0.428875	0.405520	0.165605

In [13]:
```
# create top third chart (bars)
top_third_bars = alt.Chart(thirds).mark_bar(size=20).encode(
    x=alt.X('Top Third:Q',
            axis=None,
           ),
    y=alt.Y('Name:N',
            axis=alt.Axis(tickCount=5, title=''),
            sort=names_l
           ),
    color=alt.value('#77AB43')
).properties(
    width=75,
    height=180,
).properties(
    title='Top Third'
)

# create top third chart (text)
top_third_text = top_third_bars.mark_text(
    align='left',
    baseline='middle',
    dx=3
).encode(
    text=alt.Text('Top Third:Q', format='.0%')
)
```

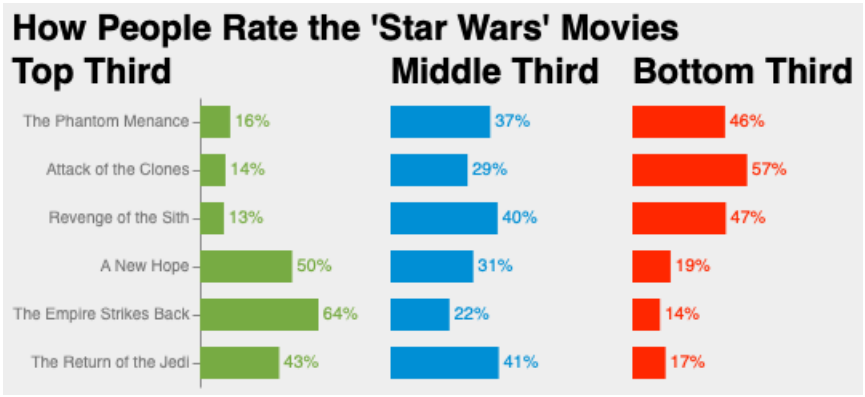
In [14]:
```
# create middle third chart (bars)
mid_third_bars = alt.Chart(thirds).mark_bar(size=20).encode(
    x=alt.X('Middle Third:Q',
            axis=None),
    y=alt.Y('Name:N',
            axis=None,
            sort=names_l
           ),
    color=alt.value('#008FD5')
```

```
    color=alt.value('#008FD5')
).properties(
    width=75,
    height=180,
).properties(
    title='Middle Third',
    #titleFontSize=10 # These keep erroring out as not a part of hconc
)

# create middle third chart (text)
mid_third_text = mid_third_bars.mark_text(
    align='left',
    baseline='middle',
    dx=3
).encode(
    text=alt.Text('Middle Third:Q', format='.0%')
)
```

In [15]:
```
# create bottom third chart (bars)
bot_third_bars = alt.Chart(thirds).mark_bar(size=20).encode(
    x=alt.X('Bottom Third:Q',
            axis=None),
    y=alt.Y('Name:N',
            axis=None,
            sort=names_l
           ),
    color=alt.value('#FF2700')
).properties(
    width=75,
    height=180,
).properties(
    title='Bottom Third'
)

# create bottom third chart (text)
bot_third_text = bot_third_bars.mark_text(
    align='left',
    baseline='middle',
    dx=3
).encode(
    text=alt.Text('Bottom Third:Q', format='.0%')
)
```

In [16]:
```
# create final chart
ratings = ((top_third_text + top_third_bars) | (mid_third_text + mid_t
    background='#eeeeee'
).configure_axis(
    labelFontSize=10,
    labelFont='Helvetica',
    labelOpacity=0.5
).configure_view(
    strokeWidth=0
).configure_scale(
    bandPaddingInner=0.05 # this isn't taking for some reason. not su
).configure_title(
    anchor='start',
    font='Helvetica',
    fontSize=22,
    fontWeight='bold'
).properties(
    title={
        "text":"How People Rate the 'Star Wars' Movies"
    }
)

ratings
```

Out[16]:

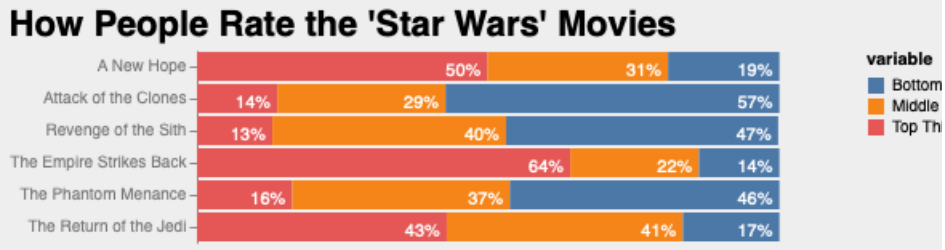


Alternative Encoding of the same concept

In [17]:
```
thirds_2 = thirds.melt(id_vars=['Name'])

thirds_2_bars = alt.Chart(thirds_2).mark_bar().encode(
    x=alt.X(
        'value:Q',
        stack='zero',
        axis=None),
    y=alt.Y(
        'Name:N',
        axis=alt.Axis(tickCount=5,title='')),
    color=alt.Color('variable')
)

thirds_2_text = alt.Chart(thirds_2).mark_text(dx=-15, dy=3, color='wh
    x=alt.X(
        'value:Q',
        stack='zero',
        axis=None),
    y=alt.Y(
        'Name:N'),
    detail='variable:N',
    text=alt.Text(
        'value:Q',
        format='.0%'
    )
)

thirds_2_final = (thirds_2_bars + thirds_2_text).configure(
```

```
    background='#eeeeee'
).configure_axis(
    labelFontSize=10,
    labelFont='Helvetica',
    labelOpacity=0.5
).configure_view(
    strokeWidth=0
).configure_title(
    anchor='start',
    font='Helvetica',
    fontSize=22,
    fontWeight='bold'
).properties(
    title="How People Rate the 'Star Wars' Movies"
)

thirds_2_final
```

Out[17]:



Character Favorability Ratings

Finally, we took a boilerplate format used by political favorability polls — "Please state whether you view the following characters favorably, unfavorably, or are unfamiliar with him/her" — and asked respondents to rate characters in the series.

In [18]:

```
# Data Pre-processing
char_list = ["Luke Skywalker", "Han Solo", "Princess Leia Organa", "Ob
             "C-3P0", "Anakin Skywalker", "Darth Vader", "Lando Calris
             "Emperor Palpatine", "Jar Jar Binks"]

perc_fav = []
perc_neutral = []
perc_unfav = []
perc_unfam = []

# filter down to only those who voted on all characters (recommended i
# voted = sw.dropna(subset=[c for c in char_list],how='any')
# total = len(voted)

for c in char_list:
    voted = seen_at_least_one.dropna(subset=[c], how='any').copy()
    c_total = len(voted)

    favorable_df = voted.loc[voted[c].isin(['Very favorably', 'Favorab
    favorable = len(favorable_df) / c_total
    perc_fav.append(float(favorable))

    neutral_df = voted.loc[voted[c].isin(['Neither favorably nor unfav
    neutral = len(neutral_df) / c_total
    perc_neutral.append(float(neutral))

    unfavorable_df = voted.loc[voted[c].isin(['Very unfavorably', 'Som
    unfavorable = len(unfavorable_df) / c_total
    perc_unfav.append(float(unfavorable))

    unfamiliar_df = voted.loc[voted[c].isin(['Unfamiliar (N/A)'])]
    unfamiliar = len(unfamiliar_df) / c_total
    perc_unfam.append(float(unfamiliar))

tuples = list(zip([c for c in char_list], perc_fav, perc_neutral, perc
favorability = pd.DataFrame(tuples, columns=['Name', 'Favorable', 'Neut
```

```
favorability = pd.DataFrame(tuples, columns=['Name', 'Favorable', 'Neutral
favorability
```

Out[18]:

	Name	Favorable	Neutral	Unfavorable	Unfamiliar
0	Luke Skywalker	0.927711	0.045783	0.019277	0.007229
1	Han Solo	0.919082	0.053140	0.010870	0.016908
2	Princess Leia Organa	0.912048	0.057831	0.021687	0.008434
3	Obi Wan Kenobi	0.910194	0.052184	0.018204	0.019417
4	Yoda	0.907879	0.061818	0.019394	0.010909
5	R2 D2	0.901086	0.068758	0.019300	0.010856
6	C-3P0	0.851090	0.095642	0.036320	0.016949
7	Anakin Skywalker	0.625304	0.164234	0.148418	0.062044
8	Darth Vader	0.581818	0.101818	0.304242	0.012121
9	Lando Calrissian	0.445122	0.287805	0.086585	0.180488
10	Padme Amidala	0.431734	0.254613	0.113161	0.200492
11	Boba Fett	0.358816	0.305795	0.173859	0.161529
12	Emperor Palpatine	0.311193	0.261993	0.236162	0.190652
13	Jar Jar Binks	0.295122	0.200000	0.373171	0.131707

In [19]:

```
# create favorable chart (bars)
favorable_bars = alt.Chart(favorability).mark_bar(size=20).encode(
    x=alt.X('Favorable:Q',
            axis=None,
            scale=alt.Scale(domain=(0,1))
           ),
    y=alt.Y('Name:N',
            axis=alt.Axis(tickCount=5, title=''),
            sort=names_l
           ),
    color=alt.value('#77AB43')
).properties(
    width=100,
    height=400,
    title=alt.TitleParams(text = 'Favorable',
                          font='Helvetica Neue',
                          fontWeight='bold',
                          fontSize=16,
                          color = '#3C3C3C',
                          anchor='end'
                         )
)

# create favorable chart (text)
favorable_text = favorable_bars.mark_text(
    align='left',
    baseline='middle',
    dx=3,
    font='Helvetica Neue',
    color='#3C3C3C'
).encode(
    text=alt.Text('Favorable:Q', format='.0%')
)
```

In [20]:
```
# create neutral chart (bars)
neutral_bars = alt.Chart(favorability).mark_bar(size=20).encode(
    x=alt.X('Neutral:Q',
            axis=None,
            scale=alt.Scale(domain=(0,1)),
           ),
    y=alt.Y('Name:N',
            axis=None,
            sort=names_l
           ),
    color=alt.value('#008FD5')
).properties(
    width=100,
    height=400,
    title=alt.TitleParams(text = 'Neutral',
                          font='Helvetica Neue',
                          fontWeight='bold',
                          fontSize=16,
                          color = '#3C3C3C',
                          anchor='start')
)

# create neutral chart (text)
neutral_text = neutral_bars.mark_text(
    align='left',
    baseline='middle',
    dx=3,
    font='Helvetica Neue',
    color='#3C3C3C'
).encode(
    text=alt.Text('Neutral:Q', format='.0%')
)
```

In [21]:
```
# create unfavorable chart (bars)
unfavorable_bars = alt.Chart(favorability).mark_bar(size=20).encode(
    x=alt.X('Unfavorable:Q',
            axis=None,
            scale=alt.Scale(domain=(0,1)),
           ),
    y=alt.Y('Name:N',
            axis=None,
            sort=names_l
           ),
    color=alt.value('#FF2700')
).properties(
    width=100,
    height=400,
    title=alt.TitleParams(text = 'Unfavorable',
                          font='Helvetica Neue',
                          fontWeight='bold',
```

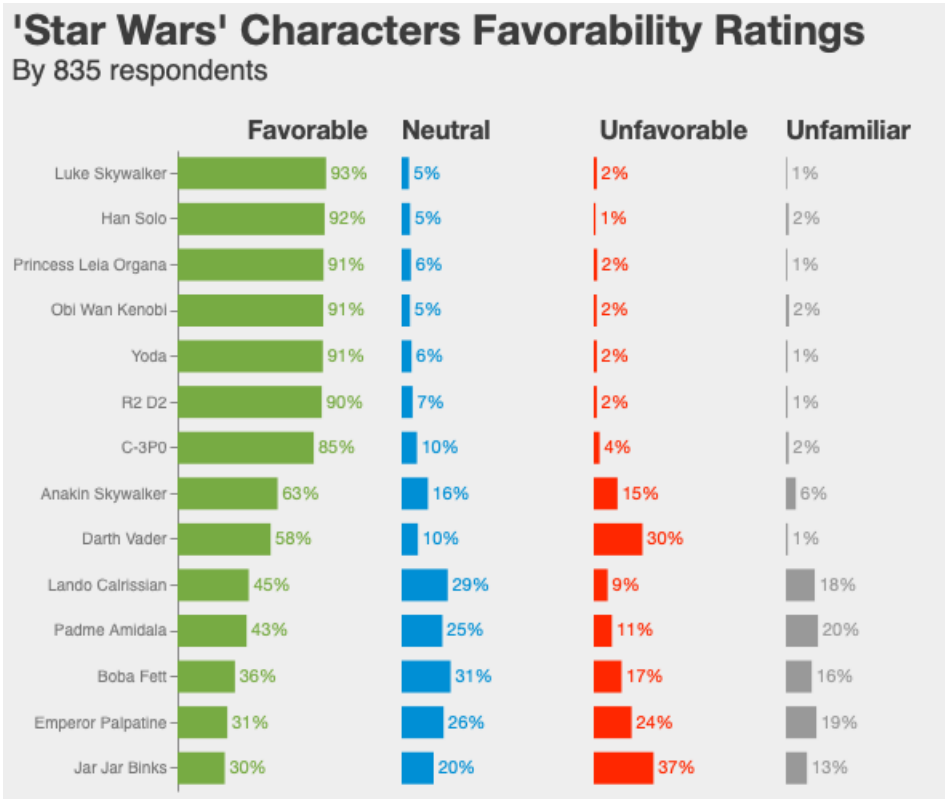
```
                            fontSize=16,
                            color = '#3C3C3C',
                            anchor='middle')
)

# create favorable chart (text)
unfavorable_text = unfavorable_bars.mark_text(
    align='left',
    baseline='middle',
    dx=3,
    font='Helvetica Neue',
    color='#3C3C3C'
).encode(
    text=alt.Text('Unfavorable:Q', format='.0%')
)
```

In [22]:
```
# create unfamiliar chart (bars)
unfamiliar_bars = alt.Chart(favorability).mark_bar(size=20).encode(
    x=alt.X('Unfamiliar:Q',
            axis=None,
            scale=alt.Scale(domain=(0,1)),
           ),
    y=alt.Y('Name:N',
            axis=None,
            sort=names_l
           ),
    color=alt.value('#999999')
).properties(
    width=100,
    height=400,
    title=alt.TitleParams(text = 'Unfamiliar',
                          font='Helvetica Neue',
                          fontWeight='bold',
                          fontSize=16,
                          color = '#3C3C3C',
                          anchor='start')
)

# create unfamiliar chart (text)
unfamiliar_text = unfamiliar_bars.mark_text(
    align='left',
    baseline='middle',
    dx=3,
    font='Helvetica Neue',
    color='#3C3C3C'
).encode(
    text=alt.Text('Unfamiliar:Q', format='.0%')
)
```

In [23]:
```
favorability_chart = ((favorable_text + favorable_bars) | (neutral_tex
```

```
    background ='#eeeeee'
).configure_axis(
    labelFontSize=10,
    labelFont='Helvetica',
    labelOpacity=0.5
).configure_view(
    strokeWidth=0
).configure_scale(
    bandPaddingInner=0.5
).configure_title(
    anchor='start',
    font='Helvetica',
    fontSize=22,
    fontWeight='bold'
).properties(
    title = alt.TitleParams(text = "'Star Wars' Characters Favorabili
                            subtitle = "By 835 respondents",
                            font = 'Helvetica Neue',
                            fontSize = 26,
                            color = '#3C3C3C',
                            subtitleFontSize = 18,
                            subtitleColor='#3C3C3C',
                            anchor='start',
                            offset=20,
                            )
)

favorability_chart
```

Out[23]:



You read that correctly. Jar Jar Binks has a lower favorability rating than the actual personification of evil in the galaxy.

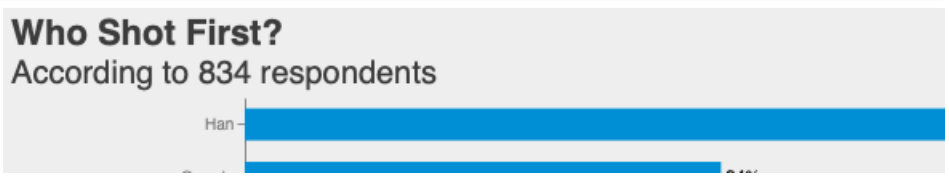
Who Shot First

And for those of you who want to know the impact that historical revisionism (http://en.wikipedia.org/wiki/Han_shot_first) can have on a society...

In [24]:
```
# Recreate this image using altair here (10 POINTS)

wsf = sw.dropna(subset=['Which character shot first?'])
```

```
total = len(wsf)

percs = []

responses = ["Han", "Greedo", "I don't understand this question"]
for r in responses:
    perc = len(wsf[wsf['Which character shot first?'] == r]) / total
    percs.append(perc)

tuples = list(zip(responses,percs))
resp_per = pd.DataFrame(tuples, columns=['Response', 'Percentage'])
#resp_per

# create bars portion of chart
bars = alt.Chart(resp_per).mark_bar(size=20).encode(
    x=alt.X(
        'Percentage',
        axis=None),
    y=alt.Y(
        'Response:N',
        axis=alt.Axis(tickCount=5, title=''),
        sort=responses)
)

# create text portion of chart
text = bars.mark_text(
    align='left',
    baseline='middle',
    dx=3
).encode(
    text=alt.Text('Percentage:Q', format='.0%')
)

# combine text + bars portion of chart
shot_first = (text + bars).configure(
    background='#eeeeee'
).configure_axis(
    labelFontSize=10,
    labelFont='Helvetica',
    labelOpacity=0.5
).configure_mark(
    color='#008fd5'
).configure_view(
    strokeWidth=0
).configure_scale(
    bandPaddingInner=0.2
).configure_title(
    anchor='start',
    font='Helvetica',
    fontSize=22,
    fontWeight='bold',
).properties(
    width=500,
    height=100
).properties(
    title={
      "text": ["Who Shot First?"],
      "subtitle": ["According to 834 respondents"],
      "color": '#3C3C3C',
      "subtitleColor": '#3C3C3C',
        "subtitleFontSize":20,
    },
)

shot_first
```

Out[24]:



Exercise adapted and modified from UMSI homework assignment for SIADS 522.