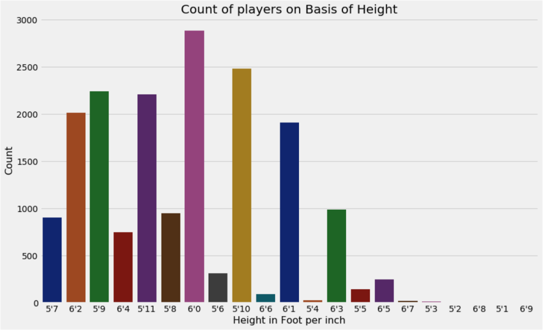
In [1]:

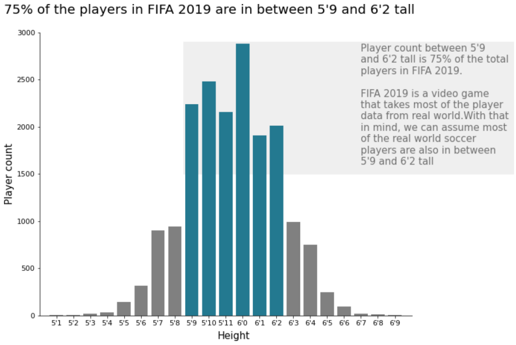
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as patches
from PIL import Image

%config InlineBackend.figure_format = 'retina'
```

FIFA GRAPH

In [2]:

```
fig, ax = plt.subplots(1,2,figsize=(25,15))

for i, image in enumerate(['fifa_bad.png', 'fifa.png']):

    graph = Image.open(image)
    ax[i].imshow(graph)
    ax[i].axis('off')

plt.show()
```



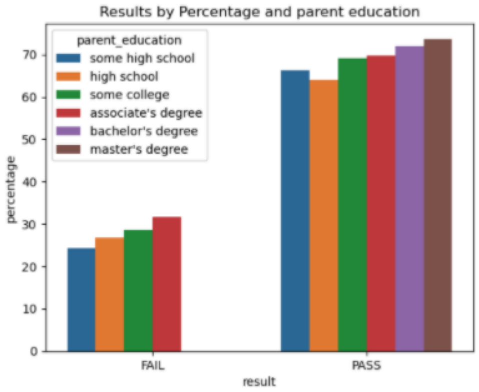


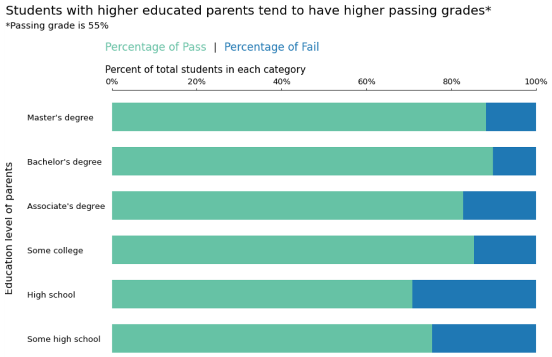
- Reordered categories from low to high since height is a quantitative(ratio) data
- Renamed axis labels to be more concise and understandable
- Changed colors to the same color to be consistent and less distracting
- Changed title to summarize what's found about the data
- Enclosed the 5'9, 6'2 height area(gestalt principle) and changed color to create contrast, since findings and the story we are telling is about that range
- Added text to point out what's possible about real world

PARENT EDUCATION GRAPH

In [3]:

```
fig, ax = plt.subplots(1,2,figsize=(25,15))

for i, image in enumerate(['education_bad.png', 'education.png']):

    graph = Image.open(image)
    ax[i].imshow(graph)
    ax[i].axis('off')

plt.show()
```



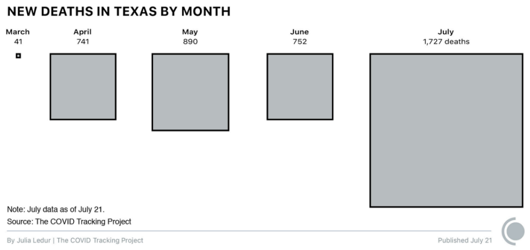


- Changed graph type to stacked horizontal bar chart because it's hard to find the relationship of fail percentage and pass percentage+
- In the old graph, both percentages of pass and fail increase as the education level becomes higher, which conflicts in itself. Created the new graph with cleaned dataset to make new graph reasonable
- The old graph has too many colors that is unnecessary and causing distractions. So assign two contrasting colors to passing and failing students for every category
- Color "percentage of pass" and "percentage of fail" texts as the same color as their corresponding bar colors for easier understanding of what colors represent
- Arrange y-axis categories from highest education to lowest edication
- Added descriptive title, subtitle, text and axis labels for easier understanding

COVID GRAPH

In [4]:

```
fig, ax = plt.subplots(1,2,figsize=(25,15))

for i, image in enumerate(['covid_bad.png', 'covid.png']):

    graph = Image.open(image)
    ax[i].imshow(graph)
    ax[i].axis('off')

plt.show()
```



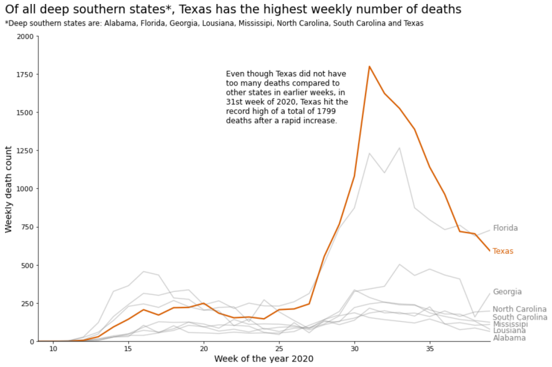
NEW DEATHS IN TEXAS BY MONTH

Note: July data as of July 21.

Source: The COVID Tracking Project

By Julia Ledur | The COVID Tracking Project

Published July 21



Of all deep southern states*, Texas has the highest weekly number of deaths

*Deep southern states are: Alabama, Florida, Georgia, Lousiana, Mississipi, North Carolina, South Carolina and Texas

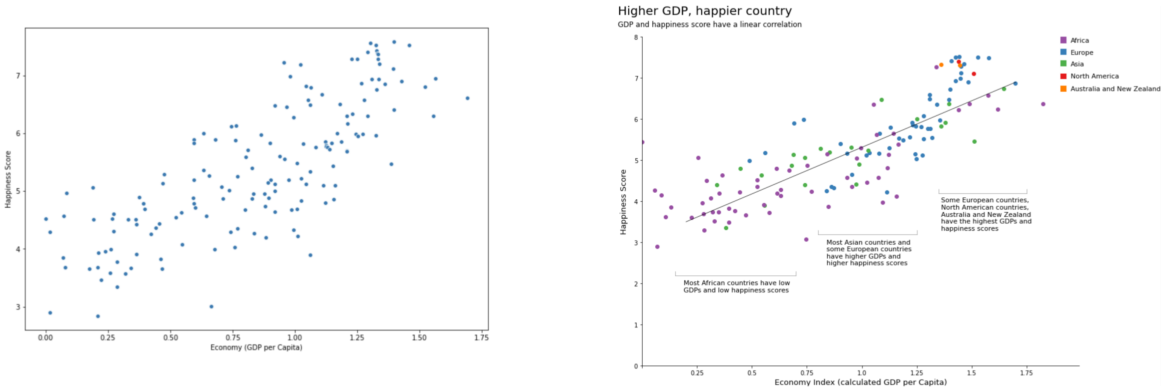
Even though Texas did not have too many deaths compared to other states in earlier weeks, in 31st week of 2020, Texas hit the record high of a total of 1799 deaths after a rapid increase.

- Get rid of square area graph idea fully, and change it to a line plot since covid death data is time dependent
- Use weeks on x-axis instead of months to be able to show more detailed graph
- Plot Texas and other Southern States on the graph for comparison
- Use color to bring Texas infront and send other states back, since our main point of interest is Texas's plot
- Add state names near related lines
- Color state names accordingly to be consistent and bring interest on Texas
- Add a title that summarizes the finding from the data
- Add a subtitle to explain which states are considered deep southern for people who might not know
- Add text to explain findings about Texas's weekly covid death data

HAPPINESS SCORE GRAPH

In [5]:

```
fig, ax = plt.subplots(1,2,figsize=(28,15))

for i, image in enumerate(['happiness_bad.png', 'happiness.png']):

    graph = Image.open(image)
    ax[i].imshow(graph)
    ax[i].axis('off')

plt.show()
```



- Remove top and right spines, since they are not adding any value
- Add a descriptive title and a subtitle to explain what's found in the data
- Make the baseline start from 0 for more intuitive comparison between happiness scores
- Add a line to show the linear correlation between GDP and happiness score
- Assign qualitative color scheme to different regions to highlight how GDP and happiness score changes depending on the region (Colors are picked from colorbrewer)
- Add texts to explain findings from the data

FIFA CODE

In [6]:

```
df_fifa = pd.read_csv('fifa2019.csv')
df_fifa.head()
```

Out[6]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	http

5 rows × 89 columns

In [7]:

```
df_fifa.columns
```

Out[7]:

```
Index(['Unnamed: 0', 'ID', 'Name', 'Age', 'Photo', 'Nationality', 'Fla
g',
       'Overall', 'Potential', 'Club', 'Club Logo', 'Value', 'Wage', 'S
pecial',
       'Preferred Foot', 'International Reputation', 'Weak Foot',
       'Skill Moves', 'Work Rate', 'Body Type', 'Real Face', 'Positio
n',
       'Jersey Number', 'Joined', 'Loaned From', 'Contract Valid Unti
l',
       'Height', 'Weight', 'LS', 'ST', 'RS', 'LW', 'LF', 'CF', 'RF', 'R
W',
       'LAM', 'CAM', 'RAM', 'LM', 'LCM', 'CM', 'RCM', 'RM', 'LWB', 'LD
M',
       'CDM', 'RDM', 'RWB', 'LB', 'LCB', 'CB', 'RCB', 'RB', 'Crossing',
       'Finishing', 'HeadingAccuracy', 'ShortPassing', 'Volleys', 'Drib
bling',
       'Curve', 'FKAccuracy', 'LongPassing', 'BallControl', 'Accelerati
on',
       'SprintSpeed', 'Agility', 'Reactions', 'Balance', 'ShotPower',
       'Jumping', 'Stamina', 'Strength', 'LongShots', 'Aggression',
       'Interceptions', 'Positioning', 'Vision', 'Penalties', 'Composur
e',
       'Marking', 'StandingTackle', 'SlidingTackle', 'GKDiving', 'GKHan
dling',
       'GKKicking', 'GKPositioning', 'GKReflexes', 'Release Clause'],
      dtype='object')
```

In [8]:

```
df_height = df_fifa.groupby('Height').count()['Name'] # Groups by height and does
n't count missing values!
x, y = df_height.sum(), df_height.loc[["5'9","5'10","5'11","6'0","6'1","6'2"]].sum
()
percentage = y/x*100
percentage
```

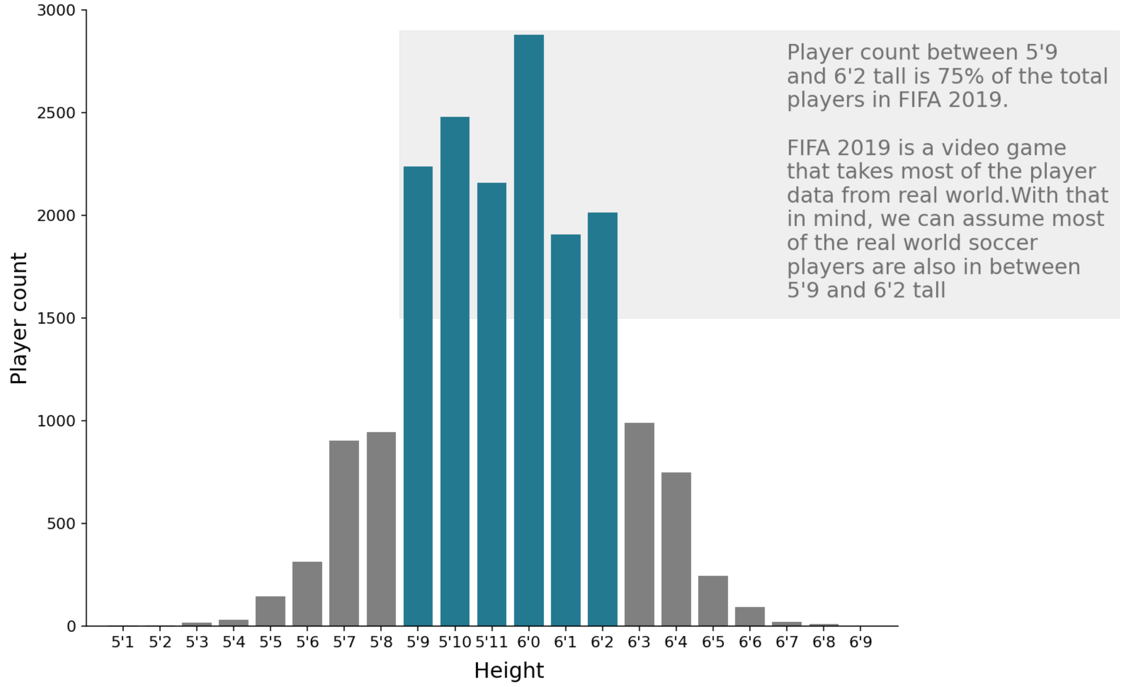
Out[8]:

```
75.33454485379151
```

In [9]:

```
fig, ax = plt.subplots(figsize=(14,8))

ax.set_ylim(0,3000)
ax.set_xlim(-1,29)

heights=["5'1","5'2","5'3","5'4","5'5","5'6","5'7","5'8","5'9","5'10","5'11",
         "6'0","6'1","6'2","6'3","6'4","6'5","6'6","6'7","6'8","6'9"]

rect = patches.Rectangle(xy=(7.5,1500), width=19.5, height=1400,
                         facecolor='gray', linewidth=.5, edgecolor="gray", alpha =
0.12)
ax.add_patch(rect)

for i, height in enumerate(heights):
    color = ('#237990' if (height=="5'9")|(height=="5'10")|(height=="5'11")|(height
=="6'0")|(height=="6'1")|(height=="6'2") else 'gray')
    ax.bar(i,df_height.loc[height], color = color)

ax.text(-3.1,3200,"75% of the players in FIFA 2019 are in between 5'9 and 6'2 tall"
, fontsize = 20)
ax.text(18,1600,"Player count between 5'9\nand 6'2 tall is 75% of the total\nplayer
s in FIFA 2019.\n\nFIFA 2019 is a video game\nthat takes most of the player\ndata f
rom real world.With that\nin mind, we can assume most\nof the real world soccer\npl
ayers are also in between\n5'9 and 6'2 tall",fontsize=15,color='#707070')
ax.text(9.5,-250,'Height', fontsize=15)
ax.set_ylabel('Player count', fontsize=15)

ax.set_xticks(range(21))
ax.set_xticklabels(heights,fontsize=11)
plt.yticks(fontsize= 11)

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_bounds(-1, 21)

plt.show()
```

75% of the players in FIFA 2019 are in between 5'9 and 6'2 tall



Player count between 5'9 and 6'2 tall is 75% of the total players in FIFA 2019.

FIFA 2019 is a video game that takes most of the player data from real world.With that in mind, we can assume most of the real world soccer players are also in between 5'9 and 6'2 tall

EDUCATION CODE

In [10]:

```
df_student = pd.read_csv('StudentsPerformance.csv')
df_student.head()
```

Out[10]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

In [11]:

```
df_student['total_score'] = (df_student['math score'] + df_student['reading score']
+ df_student['writing score'])
```

In [12]:

```
df_student.head()
```

Out[12]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score	total_sco
0	female	group B	bachelor's degree	standard	none	72	72	74	21
1	female	group C	some college	standard	completed	69	90	88	24
2	female	group B	master's degree	standard	none	90	95	93	27
3	male	group A	associate's degree	free/reduced	none	47	57	44	14
4	male	group C	some college	standard	none	76	78	75	22

In [13]:

```
df_student['total_average_score'] = df_student['total_score']/3
df_student.head()
```

Out[13]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score	total_sco
0	female	group B	bachelor's degree	standard	none	72	72	74	21
1	female	group C	some college	standard	completed	69	90	88	24
2	female	group B	master's degree	standard	none	90	95	93	27
3	male	group A	associate's degree	free/reduced	none	47	57	44	14
4	male	group C	some college	standard	none	76	78	75	22

In [14]:

```
def result(total_average):
    if total_average >=55:
        return "PASS"
    else:
        return "FAIL"
```

In [15]:

```
df_student['result'] = df_student['total_average_score'].apply(result)
```

In [16]:

```
df_student.head()
```

Out[16]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score	total_sco
0	female	group B	bachelor's degree	standard	none	72	72	74	21
1	female	group C	some college	standard	completed	69	90	88	24
2	female	group B	master's degree	standard	none	90	95	93	27
3	male	group A	associate's degree	free/reduced	none	47	57	44	14
4	male	group C	some college	standard	none	76	78	75	22

In [17]:

```
df_summary = df_student[['parental level of education', 'total_average_score', 'res
ult']]
```

In [18]:

```
df_summary
```

Out[18]:

	parental level of education	total_average_score	result
0	bachelor's degree	72.666667	PASS
1	some college	82.333333	PASS
2	master's degree	92.666667	PASS
3	associate's degree	49.333333	FAIL
4	some college	76.333333	PASS
...	...	...	...
995	master's degree	94.000000	PASS
996	high school	57.333333	PASS
997	high school	65.000000	PASS
998	some college	74.333333	PASS
999	some college	83.000000	PASS

1000 rows × 3 columns

In [19]:

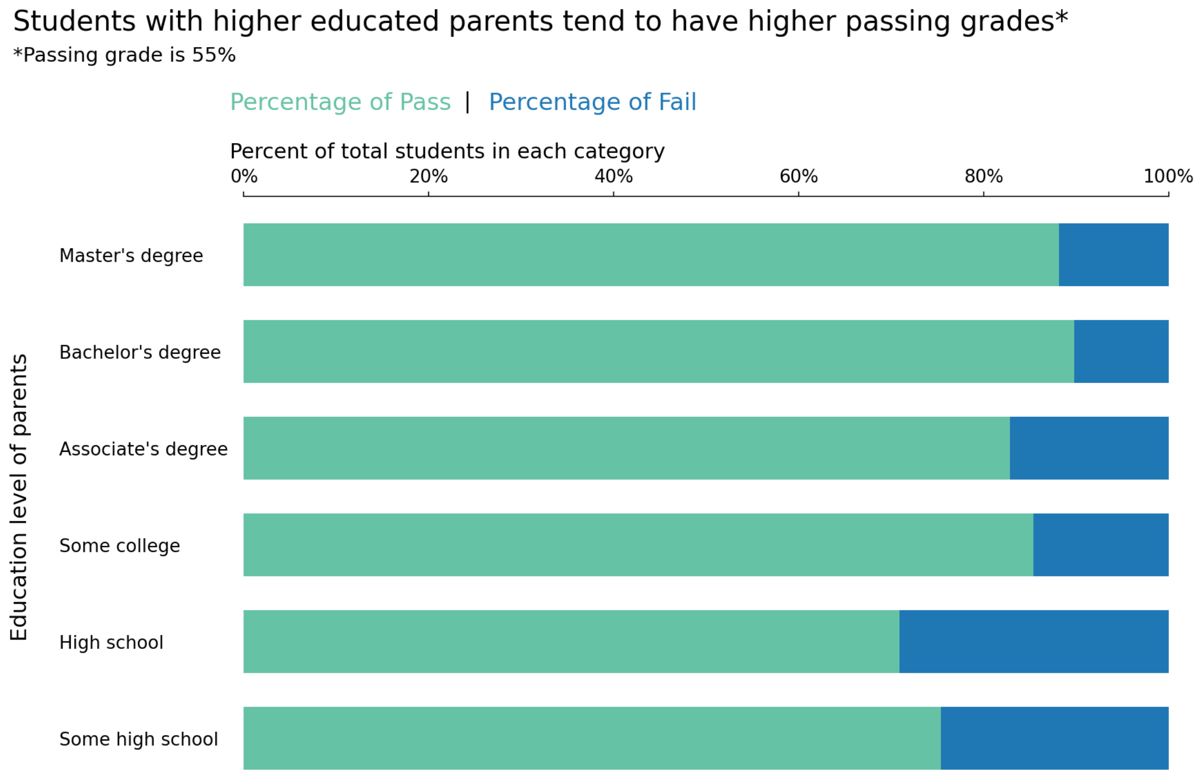
```
df_pass_fail = df_summary.groupby(['parental level of education', 'result']).count
()
df_pass_fail
```

Out[19]:

		total_average_score
parental level of education	**result**	
associate's degree	**FAIL**	38
	PASS	184
bachelor's degree	**FAIL**	12
	PASS	106
high school	**FAIL**	57
	PASS	139
master's degree	**FAIL**	7
	PASS	52
some college	**FAIL**	33
	PASS	193
some high school	**FAIL**	44
	PASS	135

In [20]:

```
fig, ax = plt.subplots(figsize=(12,8))

ax.barh(np.arange(6), 1,  height=0.65, color='#1f78b4')
i = '#66c2a5'
ax.barh(5, 52/59, height=0.65, color=i)
ax.barh(4, 106/118, height=0.65, color=i)
ax.barh(3, 184/222, height=0.65, color=i)
ax.barh(2, 193/226, height=0.65, color=i)
ax.barh(1, 139/196, height=0.65, color=i)
ax.barh(0, 135/179, height=0.65, color=i)

ax.spines['top'].set_visible(True)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(False)
ax.spines['left'].set_visible(False)


ax.set_yticklabels([None, 'Some high school', 'High school', 'Some college', '''Associate's degree''', '''Bachelor's degree''', '''Master's degree'''], ha = 'left',fontsize=13)
ax.set_xticklabels(['0%','20%','40%','60%','80%','100%'], fontsize=13)
ax.tick_params(axis='y', pad=130)
ax.set_xlim(0,1)

ax.text(-.25,7.32,"Students with higher educated parents tend to have higher passing grades*", fontsize=20)
ax.text(-.25,7,"*Passing grade is 55%", fontsize=14.5)
ax.text(0-.015,6.5,"Percentage of Pass", color="#66c2a5", fontsize=17)
ax.text(0.25-.012,6.52,"|", fontsize=16)
ax.text(0.28-.015,6.5,"Percentage of Fail", fontsize=17, color="#1f78b4")
ax.text(0-.015,6,"Percent of total students in each category", fontsize=15)
ax.set_ylabel('Education level of parents',fontsize=16, labelpad=20)

plt.tick_params(labeltop=True,labelleft=True,labelright=False,labelbottom=False)
plt.tick_params(top=True,bottom=False,left=False,right=False)

plt.show()
```



COVID CODE

In [21]:

```
df_covid = pd.read_csv('US_covid.csv') # data from cdc
df_covid.head()
```

Out[21]:

	submission_date	state	tot_cases	conf_cases	prob_cases	new_case	pnew_case	tot_death
0	01/22/2020	CO	0	NaN	NaN	0	NaN	0
1	01/23/2020	CO	0	NaN	NaN	0	NaN	0
2	01/24/2020	CO	0	NaN	NaN	0	NaN	0
3	01/25/2020	CO	0	NaN	NaN	0	NaN	0
4	01/26/2020	CO	0	NaN	NaN	0	NaN	0

In [22]:

```
df_covid.shape
```

Out[22]:

```
(14940, 15)
```

In [23]:

```
df_covid.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14940 entries, 0 to 14939
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   submission_date  14940 non-null  object 
 1   state            14940 non-null  object 
 2   tot_cases        14940 non-null  int64  
 3   conf_cases       4847 non-null   float64
 4   prob_cases       4847 non-null   float64
 5   new_case         14940 non-null  int64  
 6   pnew_case        9676 non-null   float64
 7   tot_death        14940 non-null  int64  
 8   conf_death       5168 non-null   float64
 9   prob_death       5168 non-null   float64
 10  new_death        14940 non-null  int64  
 11  pnew_death       9674 non-null   float64
 12  created_at       14940 non-null  object 
 13  consent_cases    11952 non-null  object 
 14  consent_deaths   12201 non-null  object 
dtypes: float64(6), int64(4), object(5)
memory usage: 1.7+ MB
```

In [24]:

```
df_covid['submission_date'] = pd.to_datetime(df_covid['submission_date'])
```

In [25]:

```
df_covid['week'] = df_covid['submission_date'].dt.week
```

In [26]:

```
df_covid = df_covid[['state','week','new_death']]
df_covid.head()
```

Out[26]:

	state	week	new_death
0	CO	4	0
1	CO	4	0
2	CO	4	0
3	CO	4	0
4	CO	4	0

In [27]:

```
df_covid['state'].unique()
```

Out[27]:

```
array(['CO', 'FL', 'AZ', 'SC', 'CT', 'NE', 'IA', 'NM', 'KY', 'WY', 'N
D',
       'WA', 'RMI', 'TN', 'AS', 'MA', 'PA', 'NYC', 'OH', 'AL', 'MI', 'V
A',
       'CA', 'MS', 'NJ', 'IL', 'TX', 'LA', 'WI', 'GA', 'NV', 'PR', 'I
N',
       'OR', 'MD', 'OK', 'NY', 'NC', 'ID', 'UT', 'AR', 'MO', 'DE', 'M
N',
       'WV', 'RI', 'DC', 'SD', 'ME', 'KS', 'NH', 'MT', 'HI', 'AK', 'V
T',
       'GU', 'VI', 'MP', 'FSM', 'PW'], dtype=object)
```

In [28]:

```
df_southern = df_covid.loc[(df_covid['state']=='TX')|
                           (df_covid['state']=='FL')|
                           (df_covid['state']=='GA')|
                           (df_covid['state']=='AL')|
                           (df_covid['state']=='SC')|
                           (df_covid['state']=='MS')|
                           (df_covid['state']=='LA')|
                           (df_covid['state']=='NC')]
df_southern
```

Out[28]:

	state	week	new_death
249	FL	4	0
250	FL	4	0
251	FL	4	0
252	FL	4	0
253	FL	4	0
...	...	...	...
9457	NC	39	39
9458	NC	39	30
9459	NC	39	40
9460	NC	39	53
9461	NC	39	31

1992 rows × 3 columns

In [29]:

```
df_south_grouped = df_southern.groupby(['state','week']).sum()
df_south_grouped.loc['TX']
```

Out[29]:

	new_death
week	
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	5
13	29
14	93
15	144
16	206
17	171
18	219
19	221
20	248
21	183
22	153
23	158
24	146
25	206
26	211
27	244
28	555
29	766
30	1080
31	1799
32	1622
33	1524
34	1387
35	1140

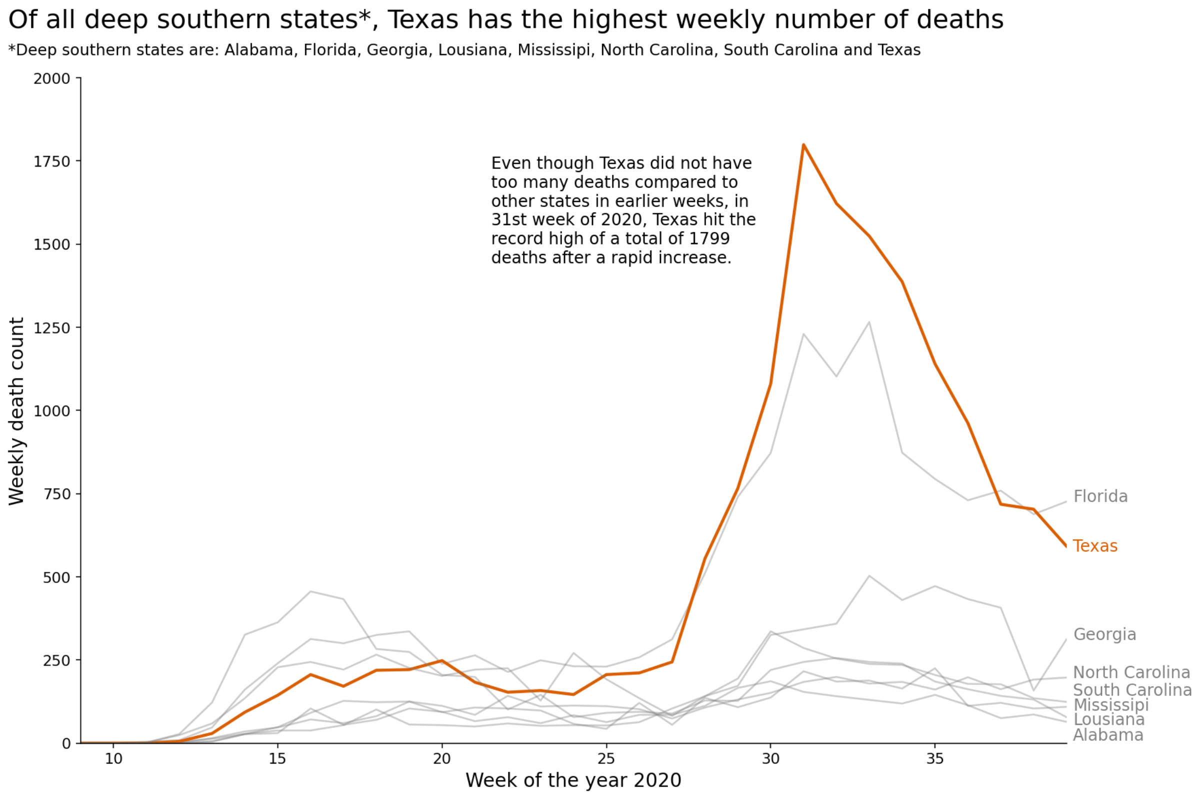
	new_death
week	
36	962
37	718
38	703
39	592

In [30]:

```
df_grouped = df_covid.groupby(['state','week']).sum()
```

In [31]:

```
fig, ax = plt.subplots(figsize = (13,9))

states = df_southern['state'].unique()

ax.set_ylim(0,2000)
ax.set_xlim(9,39)

ax.text(6.8,2150,'Of all deep southern states*, Texas has the highest weekly number of deaths', fontsize = 19)
ax.text(6.8,2070,'*Deep southern states are: Alabama, Florida, Georgia, Lousiana, Mississipi, North Carolina, South Carolina and Texas', fontsize = 11.5)

for index, state in enumerate(states):
    color = ('#d95f02' if state == 'TX' else 'gray')
    lw = (2.2 if state == 'TX' else 1.3)
    alpha = (1 if state == 'TX' else 0.4)
    ax.plot(df_south_grouped.loc[state], color = color, alpha = alpha, linewidth = lw)



ax.text(21.5, 1600, 'Even though Texas did not have\ntoo many deaths compared to\nother states in earlier weeks, in\n31st week of 2020, Texas hit the\nrecord high of a total of 1799\ndeaths after a rapid increase.',
        verticalalignment='center', horizontalalignment='left', fontsize = 12)

ax.text(39.2,df_south_grouped.loc['TX',39].values[0], 'Texas', fontsize = 12, color = '#d95f02', verticalalignment='center')
ax.text(39.2,df_south_grouped.loc['FL',39].values[0], 'Florida', fontsize = 12, alpha = .5)
ax.text(39.2,df_south_grouped.loc['GA',39].values[0], 'Georgia', fontsize = 12, alpha = .5)
ax.text(39.2,df_south_grouped.loc['NC',39].values[0], 'North Carolina', fontsize = 12, alpha = .5)
ax.text(39.2,df_south_grouped.loc['SC',39].values[0]+20, 'South Carolina', fontsize = 12, alpha = .5)
ax.text(39.2,df_south_grouped.loc['MS',39].values[0]-10, 'Mississipi', fontsize = 12, alpha = .5)
ax.text(39.2,df_south_grouped.loc['LA',39].values[0]-22, 'Lousiana', fontsize = 12, alpha = .5)
ax.text(39.2,df_south_grouped.loc['AL',39].values[0]-55, 'Alabama', fontsize = 12, alpha = .5)

# Even though Texas did not have too many deaths compared to other states in earlier weeks, in 31st week of 2020, Texas hit the record high of a total of 1799 deaths after a rapid increase.

ax.set_xlabel('Week of the year 2020', fontsize = 14)
ax.set_ylabel('Weekly death count', fontsize = 14)

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.tick_params(labelsize=11)

plt.show()
```



HAPPINESS SCORE CODE

In [32]:

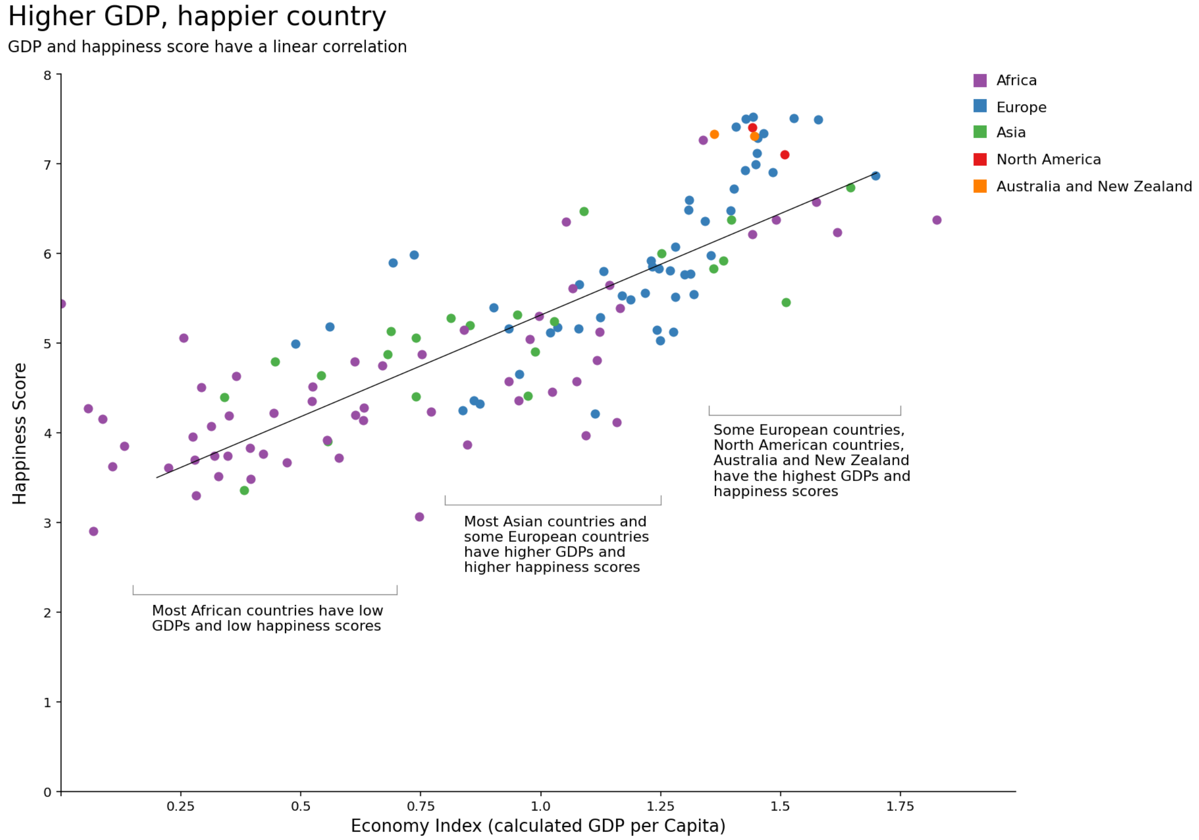
```
happiness_score = pd.read_csv("happiness_score.csv")
```

In [33]:

```
europe = happiness_score[(happiness_score['Region'] == 'Western Europe')|
                         (happiness_score['Region'] == 'Central and Eastern Europe'
)]
north_america  = happiness_score[happiness_score['Region'] == 'North America']
asia = happiness_score[(happiness_score['Region'] == 'Southeastern Asia') |
                       (happiness_score['Region'] == 'Eastern Asia')      |
                       (happiness_score['Region'] == 'Southern Asia')]
africa         = happiness_score[(happiness_score['Region'] == 'Middle East and Nor
thern Africa')|
                                 (happiness_score['Region'] == 'Sub-Saharan Africa'
)]
aus_and_newzeal= happiness_score[happiness_score['Region'] == 'Australia and New Ze
aland']
```

In [34]:

```
fig, ax = plt.subplots(figsize=(13,10))

ax.scatter(europe['Economy (GDP per Capita)'], europe['Happiness Score'], c='#377eb
8')
ax.scatter(north_america['Economy (GDP per Capita)'], north_america['Happiness Scor
e'], c='#e41a1c')
ax.scatter(asia['Economy (GDP per Capita)'], asia['Happiness Score'], c='#4daf4a')
ax.scatter(africa ['Economy (GDP per Capita)'], africa ['Happiness Score'], c='#984
ea3')
ax.scatter(aus_and_newzeal ['Economy (GDP per Capita)'], aus_and_newzeal ['Happines
s Score'], c='#ff7f00')

ax.barh(8,0.03,0.3,1.9,     color='#984ea3',edgecolor='white' )
ax.barh(7.65,0.03,0.162,1.9,color='#377eb8',edgecolor='white' )
ax.barh(7.36,0.03,0.16,1.9, color='#4daf4a',edgecolor='white' )
ax.barh(7.05,0.03,0.16,1.9,  color='#e41a1c',edgecolor='white' )
ax.barh(6.75,0.03,0.16,1.9,  color='#ff7f00',edgecolor='white' )

ax.text(1.95,7.88,'Africa', fontsize=11)
ax.text(1.95,7.58,'Europe', fontsize=11)
ax.text(1.95,7.3,'Asia', fontsize=11)
ax.text(1.95,7,'North America', fontsize=11)
ax.text(1.95,6.7,'Australia and New Zealand', fontsize=11)

ax.plot([0.2,1.7], [3.5,6.9], c='black', lw=0.7)

ax.set_ylim(0,8)
ax.set_xlim(0,1.99)
ax.set_xticklabels([None, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75])
ax.set_xlabel('Economy Index (calculated GDP per Capita)', fontsize=13)
ax.set_ylabel('Happiness Score', fontsize=13, labelpad=10)
ax.text(-.11,8.55,'Higher GDP, happier country', fontsize=20)

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(True)
ax.spines['left'].set_visible(True)

ax.text(-.11,8.25,'GDP and happiness score have a linear correlation', fontsize=12)
ax.plot([0.15,0.7],[2.2,2.2],c='gray',lw=0.6)
ax.plot([0.8,1.25],[3.2,3.2],c='gray',lw=0.6)
ax.plot([1.35,1.75],[4.2,4.2],c='gray',lw=0.6)

ax.plot([0.15,0.15],[2.2,2.3],c='gray',lw=0.6)
ax.plot([0.7,0.7],[2.2,2.3],c='gray',lw=0.6)
ax.plot([0.8,0.8],[3.2,3.3],c='gray',lw=0.6)
ax.plot([1.25,1.25],[3.2,3.3],c='gray',lw=0.6)
ax.plot([1.35,1.35],[4.2,4.3],c='gray',lw=0.6)
ax.plot([1.75,1.75],[4.2,4.3],c='gray',lw=0.6)

ax.text(0.19, 1.8,'Most African countries have low\nGDPs and low happiness scores',
fontsize=11)
ax.text(0.84, 2.45,'Most Asian countries and\nsome European countries\nhave higher
 GDPs and\nhigher happiness scores',fontsize=11)
ax.text(1.36, 3.3,'Some European countries,\nNorth American countries,\nAustralia a
nd New Zealand\nhave the highest GDPs and\nhappiness scores',fontsize=11)


plt.show()
```


Higher GDP, happier country
GDP and happiness score have a linear correlation
Africa
Europe
Asia
North America
Australia and New Zealand
Some European countries,
North American countries,
Australia and New Zealand
have the highest GDPs and
happiness scores
Most Asian countries and
some European countries
have higher GDPs and
higher happiness scores
Most African countries have low
GDPs and low happiness scores
Happiness Score
Economy Index (calculated GDP per Capita)