In [1]:
```
# Data Cleaning for the dataframe: https://data.cityofnewyork.us/Public-Safe
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
```

In [2]:
```
# Read csv file to create dataframe
file = pd.read_csv("Motor_Vehicle_Collisions_-_Crashes.csv")
```

```
/var/folders/gf/945vyt3n2m3flxbfw360lmb40000gn/T/ipykernel_2279/268181324.p
y:2: DtypeWarning: Columns (3) have mixed types. Specify dtype option on im
port or set low_memory=False.
  file = pd.read_csv("Motor_Vehicle_Collisions_-_Crashes.csv")
```

In [3]:
```
# Check missing values
file.isna().sum()
```

Out[3]:
```
CRASH DATE                          0
CRASH TIME                          0
BOROUGH                        628067
ZIP CODE                       628305
LATITUDE                       229727
LONGITUDE                      229727
LOCATION                       229727
ON STREET NAME                 425008
CROSS STREET NAME              755889
OFF STREET NAME               1686327
NUMBER OF PERSONS INJURED          18
NUMBER OF PERSONS KILLED           31
NUMBER OF PEDESTRIANS INJURED       0
NUMBER OF PEDESTRIANS KILLED        0
NUMBER OF CYCLIST INJURED           0
NUMBER OF CYCLIST KILLED            0
NUMBER OF MOTORIST INJURED          0
NUMBER OF MOTORIST KILLED           0
CONTRIBUTING FACTOR VEHICLE 1    6351
CONTRIBUTING FACTOR VEHICLE 2  308070
CONTRIBUTING FACTOR VEHICLE 3 1875772
CONTRIBUTING FACTOR VEHICLE 4 1986822
CONTRIBUTING FACTOR VEHICLE 5 2010288
COLLISION_ID                        0
VEHICLE TYPE CODE 1             12684
VEHICLE TYPE CODE 2            377220
VEHICLE TYPE CODE 3           1880762
VEHICLE TYPE CODE 4           1987893
VEHICLE TYPE CODE 5           2010550
dtype: int64
```

In [4]:
```
file.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2018963 entries, 0 to 2018962
Data columns (total 29 columns):
 #   Column                          Dtype  
---  ------                          -----  
 0   CRASH DATE                      object 
 1   CRASH TIME                      object 
 2   BOROUGH                         object 
 3   ZIP CODE                        object 
 4   LATITUDE                        float64
 5   LONGITUDE                       float64
 6   LOCATION                        object 
 7   ON STREET NAME                  object 
 8   CROSS STREET NAME               object 
 9   OFF STREET NAME                 object 
 10  NUMBER OF PERSONS INJURED       float64
 11  NUMBER OF PERSONS KILLED        float64
 12  NUMBER OF PEDESTRIANS INJURED   int64  
 13  NUMBER OF PEDESTRIANS KILLED    int64  
 14  NUMBER OF CYCLIST INJURED       int64  
 15  NUMBER OF CYCLIST KILLED        int64  
 16  NUMBER OF MOTORIST INJURED      int64  
 17  NUMBER OF MOTORIST KILLED       int64  
 18  CONTRIBUTING FACTOR VEHICLE 1   object 
 19  CONTRIBUTING FACTOR VEHICLE 2   object 
 20  CONTRIBUTING FACTOR VEHICLE 3   object 
 21  CONTRIBUTING FACTOR VEHICLE 4   object 
 22  CONTRIBUTING FACTOR VEHICLE 5   object 
 23  COLLISION_ID                    int64  
 24  VEHICLE TYPE CODE 1             object 
 25  VEHICLE TYPE CODE 2             object 
 26  VEHICLE TYPE CODE 3             object 
 27  VEHICLE TYPE CODE 4             object 
 28  VEHICLE TYPE CODE 5             object 
dtypes: float64(4), int64(7), object(18)
memory usage: 446.7+ MB
```

In [5]:
```
# Convert 'CRASH DATE' and 'CRASH TIME' to datetime
```

In [6]:
```
file['CRASH DATE'] = pd.to_datetime(file['CRASH DATE'])
```

In [7]:
```
file["CRASH TIME"] = pd.to_datetime(file["CRASH TIME"])
```

In [8]:
```
file.columns
```

Out[8]:
```
Index(['CRASH DATE', 'CRASH TIME', 'BOROUGH', 'ZIP CODE', 'LATITUDE',
       'LONGITUDE', 'LOCATION', 'ON STREET NAME', 'CROSS STREET NAME',
       'OFF STREET NAME', 'NUMBER OF PERSONS INJURED',
       'NUMBER OF PERSONS KILLED', 'NUMBER OF PEDESTRIANS INJURED',
       'NUMBER OF PEDESTRIANS KILLED', 'NUMBER OF CYCLIST INJURED',
       'NUMBER OF CYCLIST KILLED', 'NUMBER OF MOTORIST INJURED',
       'NUMBER OF MOTORIST KILLED', 'CONTRIBUTING FACTOR VEHICLE 1',
       'CONTRIBUTING FACTOR VEHICLE 2', 'CONTRIBUTING FACTOR VEHICLE 3',
       'CONTRIBUTING FACTOR VEHICLE 4', 'CONTRIBUTING FACTOR VEHICLE 5',
       'COLLISION_ID', 'VEHICLE TYPE CODE 1', 'VEHICLE TYPE CODE 2',
       'VEHICLE TYPE CODE 3', 'VEHICLE TYPE CODE 4', 'VEHICLE TYPE CODE
5'],
      dtype='object')
```

In [9]:
```
# Delete columns to create easy an dataframe with only the columns needed

df = file[['CRASH DATE', 'CRASH TIME', 'BOROUGH', 'ZIP CODE','LOCATION','ON
       'NUMBER OF PEDESTRIANS KILLED', 'NUMBER OF CYCLIST INJURED','NUMBER O
```

In [34]:
```
# At what time do most accidents occur?

df['hour'] = df['CRASH TIME'].dt.hour

# Group by hours and accident count over the years:
accident_counts = df.groupby('hour')['hour'].count()

# Create line graph with custom styling:
sns.set_style("white")
plt.figure(figsize=(7, 2))

# Adjust line style (thicker line), color, and markers
plt.plot(accident_counts.index,
         accident_counts.values,
         color='blue',
         linewidth=4,
         label='Accidents')
plt.xlabel("Time of Day", color='gray')
plt.ylabel("Accident Count", color='gray')

# Find the hour with the maximum accident count
max_hour = accident_counts.idxmax()

# Modify the title style
plt.title("Most accidents occur during rush hour",
          color='black')

# Set x-axis and y-axis tick color to gray
plt.xticks(range(24), color='gray')
plt.yticks(color='gray')


# Change the color of the lines next to the y-axis and x-axis labels to gray
ax = plt.gca()
ax.spines['left'].set_color('gray')
ax.spines['bottom'].set_color('gray')


# Highlight the maximum point with a red dot
plt.plot(max_hour,
         accident_counts[max_hour],
         'ro',
         markersize=9,
         label='Maximum')

# Get rid of the grid
sns.despine()
plt.grid(False)

# Save the graph as a PNG file
plt.savefig('accidents_by_hour_small.png', dpi=300, bbox_inches='tight')
plt.show()
```

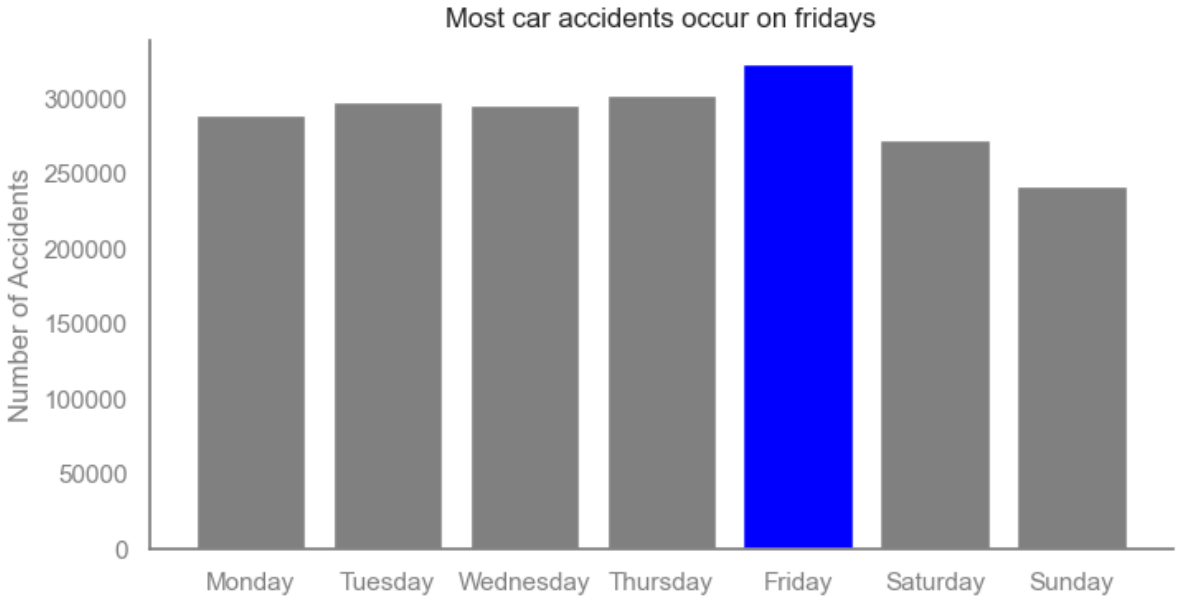
```
/var/folders/gf/945vyt3n2m3flxbfw360lmb40000gn/T/ipykernel_2279/4075929296.
py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['hour'] = df['CRASH TIME'].dt.hour
```

In [35]:
```
# In what month do most accidents occur?

# Get the month from CRASH DATE in new column:
df['month'] = df['CRASH DATE'].dt.month

# Groupby month and accident count over the years:
accident_counts_month = df.groupby('month')['month'].count()
sns.set_style("white")

# create linegraph:
plt.figure(figsize=(7,2))
plt.plot(accident_counts_month.index,
         accident_counts_month.values,
         color='blue',
         linewidth=4,
         label='Accidents')

# Change the color of the lines next to the y-axis and x-axis labels to gray
ax = plt.gca()
ax.spines['left'].set_color('grey')
ax.spines['bottom'].set_color('grey')

# Get rid of the grid
sns.despine()
plt.grid(False)

# Customize labels and titles
plt.xlabel("Month of Year", color = "grey")
plt.ylabel("Accidents", color = "grey")
plt.title("Most car accidents occur in the summer")
plt.xticks(range(1,13), color = "grey")
plt.yticks(color = "grey")

plt.savefig('accidents_by_month_small.png', dpi=300, bbox_inches='tight')
plt.show()
```

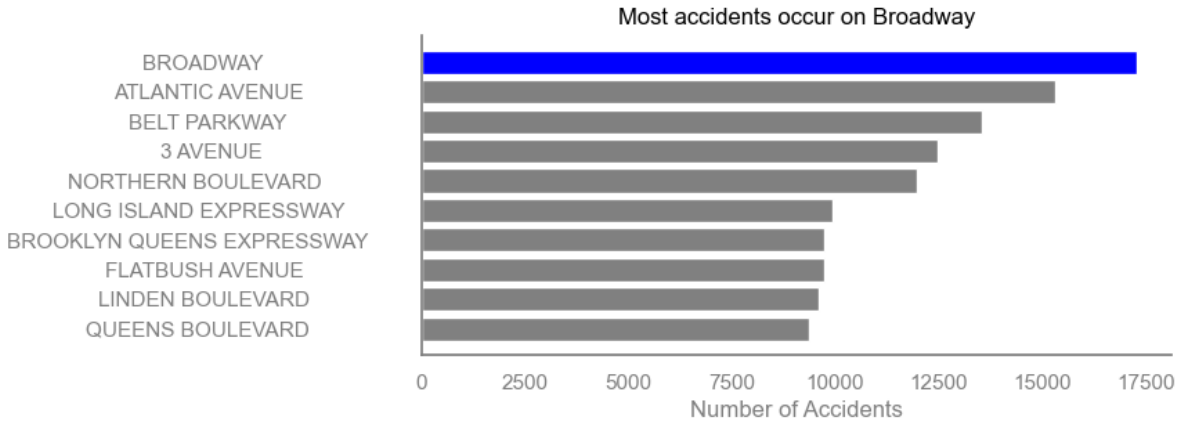
```
/var/folders/gf/945vyt3n2m3flxbfw360lmb40000gn/T/ipykernel_2279/1427170387.
py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['month'] = df['CRASH DATE'].dt.month
```

In [37]:
```
# On what day of the week do most accidents occur?
df['day of the week'] = df['CRASH DATE'].dt.dayofweek + 1

# Calculate the number of accidents per day of the week
weekday_counts = df['day of the week'].value_counts().sort_index()

# Define the names of the days of the week
days_of_week = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Sat

# Create a bar chart with all bars in grey
plt.figure(figsize=(8, 4))
colors = ['grey' if day != days_of_week[weekday_counts.idxmax() - 1] else 'b
plt.bar(days_of_week, weekday_counts, color=colors)
plt.ylabel("Number of Accidents",
           color = "grey")

# Change the color of the lines next to the y-axis and x-axis labels to gray
ax = plt.gca()
ax.spines['left'].set_color('grey')
ax.spines['bottom'].set_color('grey')

# Customize labels and titles
plt.xticks(color = "grey")
plt.yticks(color = "grey")
plt.title("Most car accidents occur on fridays")
sns.despine()

plt.savefig('accidents_by_weekday_small.png', dpi=300, bbox_inches='tight')
plt.show()
```

```
/var/folders/gf/945vyt3n2m3flxbfw360lmb40000gn/T/ipykernel_2279/3326852988.
py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['day of the week'] = df['CRASH DATE'].dt.dayofweek + 1
```



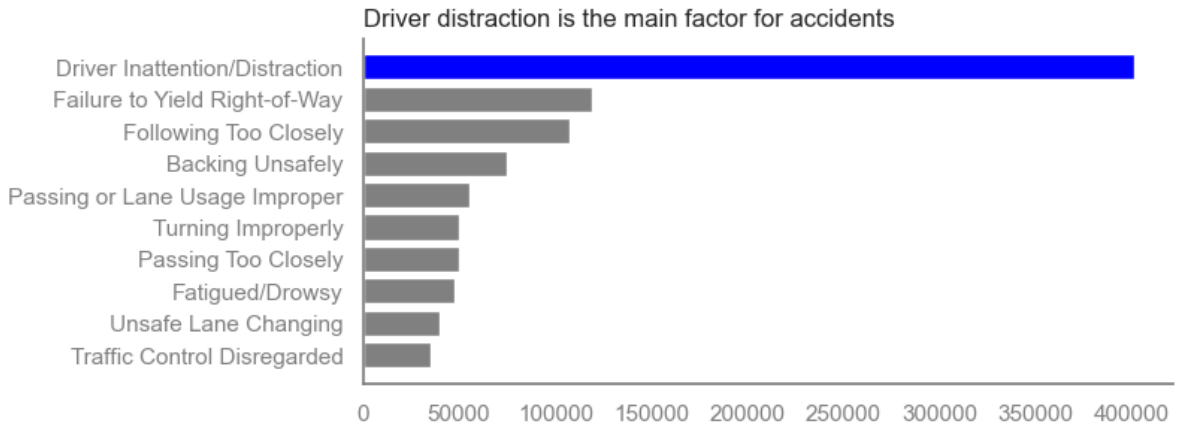
In [38]:
```
# On what street do most accidents happen?

# Calculate the number of accidents per street
street_counts = df['ON STREET NAME'].value_counts()

# Select the top 10 streets to display on the chart
top_streets = street_counts.head(10)

colors = ['blue' if street == top_streets.index[0] else 'grey' for street in

# Create a horizontal bar chart
plt.figure(figsize=(7, 3))
plt.barh(top_streets.index, top_streets.values, color=colors)

# Customize labels and titles
plt.xlabel("Number of Accidents", color="grey")
plt.title("Most accidents occur on Broadway", color="black")
plt.gca().invert_yaxis()  # Invert the y-axis to display the most dangerous
plt.yticks(color="grey")
plt.xticks(color="grey")

# Customize the axis lines
ax = plt.gca()
ax.spines['right'].set_visible(False)
ax.spines['top'].set_visible(False)
ax.spines['left'].set_color('grey')
ax.spines['bottom'].set_color('grey')

plt.savefig('accidents_by_streets_small.png', dpi=300, bbox_inches='tight')
plt.show()
```



In [39]:
```
# What are the main contributing factors for accidents?
contributing_factor = df['CONTRIBUTING FACTOR VEHICLE 1'].value_counts()
contributing_factor.head(12)
```

Out[39]:
```
Unspecified                     692913
Driver Inattention/Distraction  401420
Failure to Yield Right-of-Way   119209
Following Too Closely           107510
Backing Unsafely                 75053
Other Vehicular                  62709
Passing or Lane Usage Improper   55474
Turning Improperly               49928
Passing Too Closely              49873
Fatigued/Drowsy                  47346
Unsafe Lane Changing             39722
Traffic Control Disregarded      35235
Name: CONTRIBUTING FACTOR VEHICLE 1, dtype: int64
```

In [40]:
```
# Visualise results without "Unspecified" and "Other Vehicular", because it

reasons = ["Driver Inattention/Distraction", "Failure to Yield Right-of-Way'
counts = [401409, 119204, 107510, 75053, 55474, 49925, 49873, 47344, 39721,3
colors = ['blue' if reason == 'Driver Inattention/Distraction' else 'grey' f

plt.figure(figsize=(7, 3))
plt.barh(reasons, counts, color=colors)

plt.title("Driver distraction is the main factor for accidents", loc = "left
plt.yticks(color="grey")
plt.xticks(color="grey")

# Customize the axis lines
ax = plt.gca()
ax.spines['right'].set_visible(False)
ax.spines['top'].set_visible(False)
ax.spines['left'].set_color('grey')  # Set the left spine to grey
ax.spines['bottom'].set_color('grey')  # Set the bottom spine to grey

plt.gca().invert_yaxis()  # Invert the y-axis for better readability
plt.savefig('contributing_factors_small.png', dpi=300, bbox_inches='tight')

plt.show()
```



In [41]:
```
# How many persons have been killed in accidents from 2012-2022?
total_persons_killed = (killed_persons["NUMBER OF PERSONS KILLED"] * killed_

print("Total persons killed:", total_persons_killed)
```

```
Total persons killed: 2923.0
```

In [42]:
```
# Who is most likely to die in an accident?
most_likely_to_die = df[['NUMBER OF CYCLIST KILLED',"NUMBER OF PEDESTRIANS K
most_likely_to_die
```

Out[42]:
```
NUMBER OF CYCLIST KILLED         226
NUMBER OF PEDESTRIANS KILLED    1464
NUMBER OF MOTORIST KILLED       1194
dtype: int64
```