Dear LEGO, what happened?!

I used to love playing with LEGO in the 90s when I was growing up, we built many crazy, moving-flying empires with my dad. But now, when I play with my kids, LEGO frustrates me more than anything. Instead of getting creative on building from the infinit stock of the "4er, 8er and some rods and wheels", there are highly specific bits, annoyingly non-matching shades and long users' manuals everwhere in our flat. Is it just me, or has LEGO fundamentally changed the shapes and colors of their bricks making them more complicated than ever?

To answer this question, I used the data from rebrickable. This database actually updates daily. The data used in this analysis was downloaded today, on the 3 May 2024.

After getting the right data, I will:

- explore the amount of sets and their complexity (parts/set) over time
- find a popular theme (spoiler! it's Star Wars!) and look at the size of the set and unique colors over time
- then the coolest plot: color shades of all parts over time

As always, feedback is very welcome. Enjoy!

Technical Preparation: Imports and Data

In [10]:
```
import csv
import gzip
import os

import numpy as np
import pandas as pd

import matplotlib as mpl
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter  # for percentages
import seaborn as sns

%matplotlib inline
```

In [2]:
```
# Directory of gzipped files
data_dir = "data/"

# List of gzipped file paths
gz_files = ["colors.csv.gz", "elements.csv.gz", "inventories.csv.gz", "inventory_minifig
            "inventory_parts.csv.gz", "inventory_sets.csv.gz", "part_categories.csv.gz",
            "sets.csv.gz", "minifigs.csv.gz", "themes.csv.gz", "parts.csv.gz"]

# Dictionary to store DataFrames
lego_datasets = {}

for file_name in gz_files:
    # construct full file path
    file_path = os.path.join(data_dir, file_name)

    data = []

    # Open the gzipped file
```

```
    with gzip.open(file_path, "rt") as file:
        reader = csv.reader(file) # CSV header
        columns = next(reader) # for header row
        for row in reader:
           data.append(row)

    # Create a pandas DataFrame
    df_name = file_name.split(".")[0]  # Extract filename without extension
    lego_datasets[df_name] = pd.DataFrame(data, columns = columns)
```

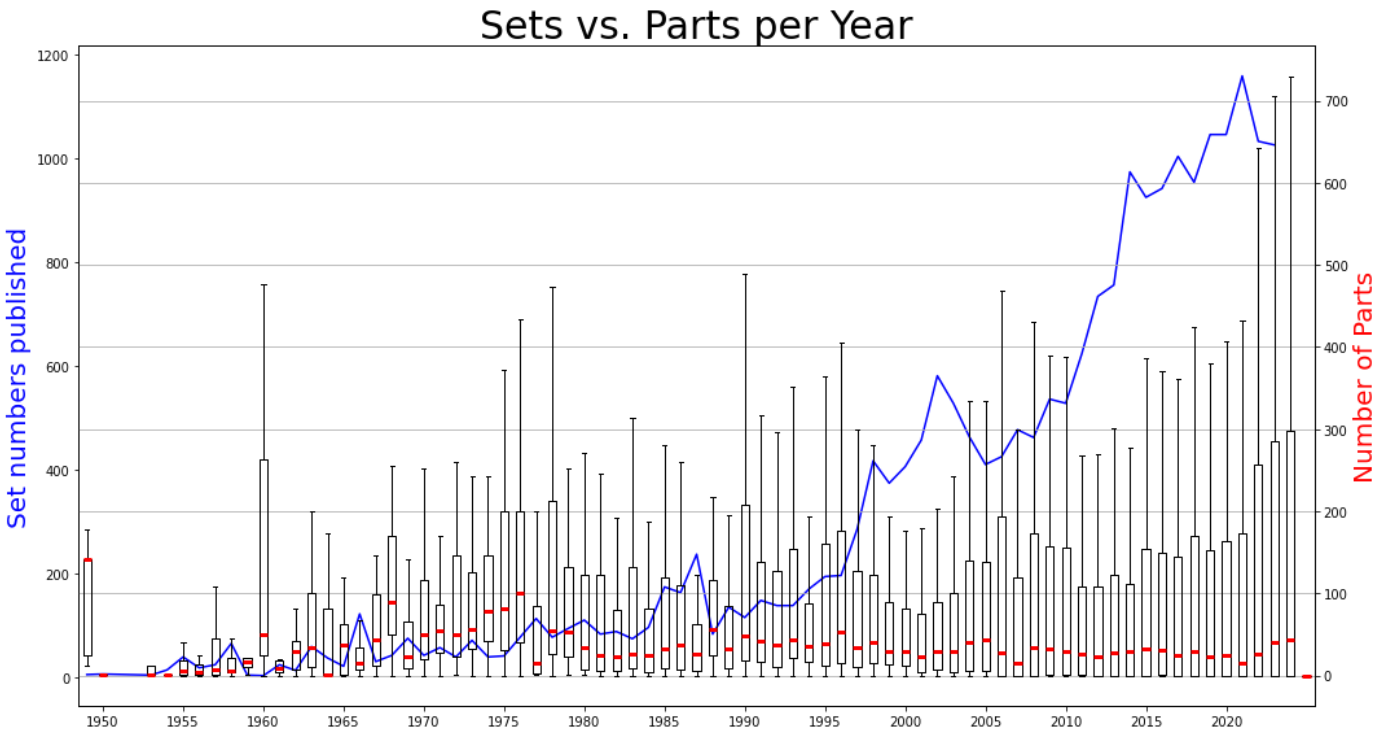
The datasets are related to each other and contain the following variables:

Let's take a look at the available data first. The smallest element is the parts: we have data on their color composition, themes they belong to, and it can all be arranged to sets, and then inventories. Taking a closer look it turns out that this data covers over 75 years (1949-2025)!

Sets and Complexity Over Time

First let's see temporal trends. Is it possible that the reason for the overwhelming complexity of LEGOs is that they published much more sets over time? Or that sets are larger now than how they used to be?

In [18]:
```
sets = lego_datasets["sets"]

# Set plot
plt.style.use('seaborn-v0_8-bright')
fig, ax1 = plt.subplots(figsize=(15, 8), sharex=True)

# Plot 1 - Line plot
sets_num_agg = sets[['year', 'set_num']].groupby(['year']).agg('count')
sets_num_agg.index = sets_num_agg.index.astype(int)
sets_num_agg = sets_num_agg[sets_num_agg.index < 2024] # for completed years only

year = sets_num_agg.index.values
set_number = sets_num_agg['set_num'].values

ax1.plot(year, set_number, color='blue', label='Set numbers published')
ax1.set_ylabel('Set numbers published', color='blue', fontsize = 20)

# Plot 2 - Boxplot
parts_year = sets[['year', 'num_parts']].astype('int')
parts_year = parts_year[['year', 'num_parts']]

years = np.arange(min(parts_year['year']) + 1, max(parts_year['year']) - 4, 5)

ax2 = ax1.twinx()  # Create a twin axis
bp = ax2.boxplot([group['num_parts'] for year, group in parts_year.groupby('year')],
                 positions=sorted(parts_year['year'].unique()), showfliers=False,
                 medianprops=dict(color='red', linewidth=3))

ax2.set_ylabel('Number of Parts', color='red', fontsize = 20)

# Display settings
plt.title('Sets vs. Parts per Year', fontsize = 30)
plt.xlabel('Year')
plt.xticks(years, years)
plt.grid(True)
plt.tight_layout()
plt.show()
```



There are definitely more sets being published every year, there is an approximately 5-fold increase between the 1980s and the 2010s. However, sets do not increase in size (although the number of parts per set seems to vary more recently than before). Unfortunately there is no way from the data to tell if parts are also more complex than before (more specific to the given set), but kind of likely they are.

Most common LEGO themes

I was actually wondering if the complexity was due to publishing more sets or that sets within the same theme became well, more complex. First let's see the Top 10 most LEGO themes (in terms of number of published sets).

In [4]:
```
themes = lego_datasets["themes"]

# merge DFs to get parent theme
themes_common = sets.merge(themes, left_on = 'theme_id', right_on = 'id', how = 'left')
themes_common.rename(columns = {'name_y': 'theme_parent', 'name_x': 'theme_child'},
                     inplace = True)

# get Top 10 themes
themes_common_count = themes_common.groupby('theme_parent').size().reset_index(name='cou
themes_common_count.sort_values(by='counts', ascending=False).head(10).reset_index(drop
```

Out[4]:

	theme_parent	counts
0	Star Wars	941
1	Technic	887
2	Key Chain	706
3	Gear	627
4	Friends	602
5	Ninjago	538
6	Bionicle	442
7	Town	423

8	Classic Town	411
9	Basic Set	386

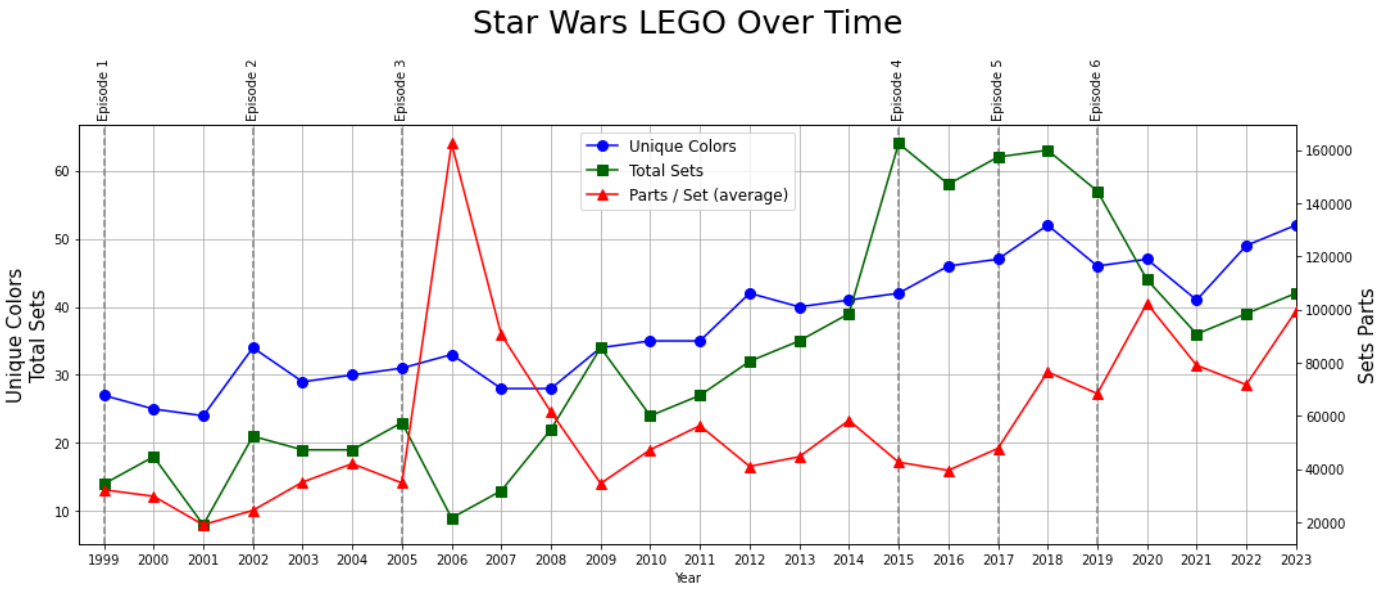
Oh wow, almost a 1000 Star Wars sets were published since 1949! This is about 8 times more than Harry Potter in place 30 (around the 92th percentile). Partially because well, Harry Potter wasn't even born when Han Solo first kissed the mother of his future twins in the Millenial Falcon... So let's see what happened to Star Wars given that they came out (and trending) with multiple movies accross multiple decades. With some luck this can capture LEGO's intentions regarding the complexity of their sets with the same theme.

Star Wars Across Time

In [5]:
```
merged_df = lego_datasets['colors'] \
    .merge(lego_datasets['inventory_parts'], left_on='id', right_on='color_id', how='lef
    .merge(lego_datasets['inventories'], left_on='inventory_id', right_on='id', how='lef
    .merge(lego_datasets['sets'], left_on='set_num', right_on='set_num', how='left') \
    .merge(lego_datasets['themes'], left_on='theme_id', right_on='id', how='left')

star_wars_df = merged_df[merged_df['name'] == 'Star Wars']

# Convert 'num_parts' column to numeric
star_wars_df.loc[:, 'num_parts'] = pd.to_numeric(star_wars_df['num_parts'])

# Group by year and count unique colors, total sets, and total parts
star_wars = star_wars_df.groupby('year').agg(
    unique_colors=('id_x', 'nunique'),  # Count unique colors
    total_sets=('set_num', 'nunique'),  # Count total sets
    total_parts=('num_parts', 'sum')    # Sum of num_parts
)

star_wars['sets_parts'] = star_wars['total_parts'] / star_wars['total_sets']
```

In [25]:
```
fig, ax1 = plt.subplots(figsize=(17, 6))

movies = {
    'Episode 1': 1999,
    'Episode 2': 2002,
    'Episode 3': 2005,
    'Episode 4': 2015,
    'Episode 5': 2017,
    'Episode 6': 2019
}

plt.style.use('seaborn-v0_8-bright')

# Ensure 'years' are integers
years = star_wars.index.astype(int)

# Plotting with higher zorder
unique_colors_line, = ax1.plot(years, star_wars['unique_colors'],
                               color='blue', marker='o', markersize = 8,
                               label='Unique Colors', zorder=3)
total_sets_line, = ax1.plot(years, star_wars['total_sets'],
                            color = 'darkgreen', marker='s', markersize = 8,
                            label='Total Sets', zorder=3)
ax1.set_xlabel('Year')
ax1.set_ylabel('Unique Colors\nTotal Sets', fontsize = 15)
ax1.tick_params(axis='y')
ax1.grid(True)

# Create twin axis for sets_parts
```

```
ax2 = ax1.twinx()
sets_parts_line, = ax2.plot(years, star_wars['sets_parts'],
                            color='red', marker='^', markersize = 8,
                            label='Parts / Set (average)', zorder=3)
ax2.set_ylabel('Sets Parts', fontsize = 15)
ax2.tick_params(axis='y')
ax2.grid(False)

# Add vertical lines for movies
for movie, year in movies.items():
    ax1.axvline(x=year, color='gray', linestyle='--', alpha=1, zorder=2)
    ax1.text(year, ax1.get_ylim()[1] * 1.01, movie,
             rotation=90, verticalalignment='bottom', horizontalalignment='center', zord

# Set labels and legend
ax1.legend([unique_colors_line, total_sets_line, sets_parts_line],
           ['Unique Colors', 'Total Sets', 'Parts / Set (average)'],
           loc='upper center',
           facecolor='white', fontsize = 12)

plt.title('Star Wars LEGO Over Time', y=1.2, fontsize=25)

ax1.set_xticks(years)
ax1.set_xticklabels(years)
ax1.set_xlim(min(years) - 0.5, 2023)  # for completed years only


# Show plot
plt.show()
```



Suprisingly (to me) the first Star Wars LEGO set was published after Episode 1, which means that the initial triology in the 1980s was not commemorated. In the beginning there were on average about 3000 parts per set, until the dark complexity of the siths (Episode 3) called for more volume - all of a sudden sets contained 4 times more parts than before! The new, post-era, non-georgelucasian films also called for more parts apparently, increasing the average part/set number to around 9000. As the success of the Star Wars movies grew, so did the number of sets published, especially during the release of the latest trilogy (Episodes 4-6). There is also a linearly increasing trend of including more unique colors into the set... So yes, Star Wars LEGO sets undoubtedly got more complex over time!

Color Composition Over Time

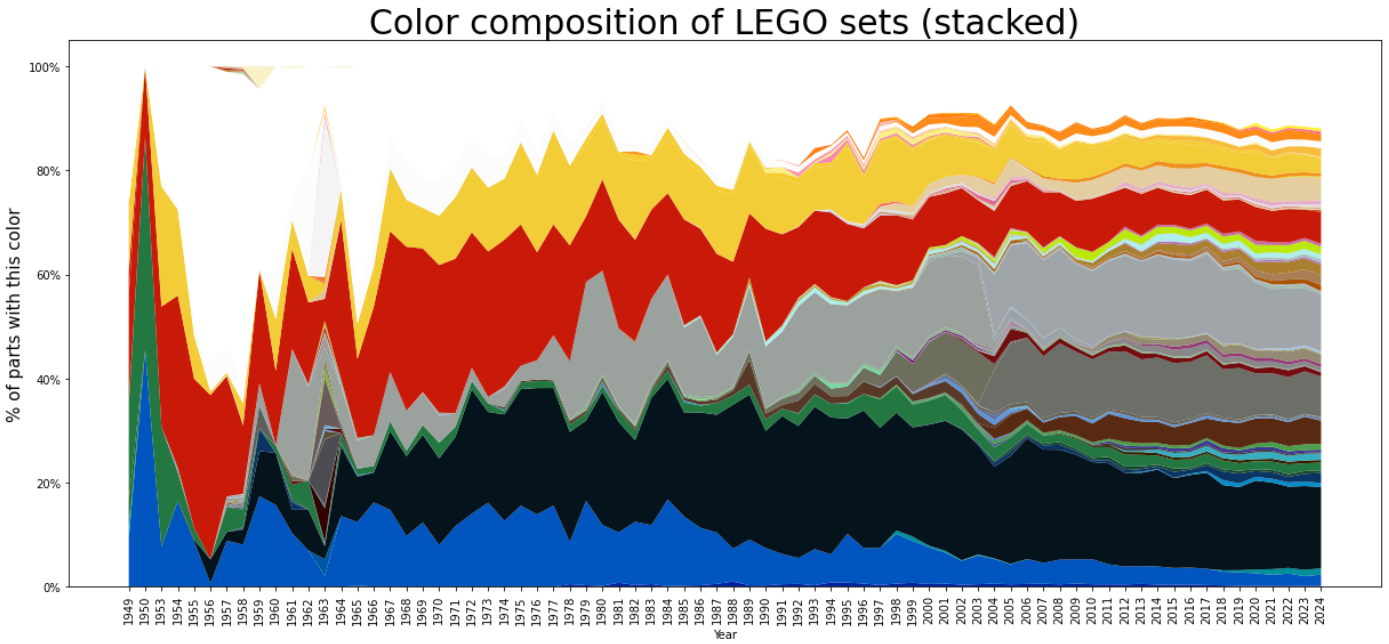
I have a feeling there are just too many colors these days in LEGO sets. So I wanted to see what actually happened to the color composition of sets in the past 70+ years.

This time I'll ignore the sets, and plot the relative occurance of part colors over time. Just like my kids do when they put all their LEGO parts in the middle of our livingroom (maybe they do relative .

By the way, this great idea for visualising shades accross time came from David Robinson at Tidy Tuesday (written in R).

In [7]:
```
sets_colors = lego_datasets['sets']\
                .merge(lego_datasets['inventories'], on='set_num', how = 'left')\
                .merge(lego_datasets['inventory_parts'], left_on = 'id', right_on = 'inv
                .merge(lego_datasets['colors'], left_on = 'color_id', right_on = 'id', h
sets_rgb = sets_colors[['year', 'name_y', 'rgb']].rename(columns = {'name_y':'color'}).s
sets_rgb['rgb'] = '#' + sets_rgb['rgb']

# Create a pivot table to count occurrences of each 'rgb' value per 'year'
pivot_table = sets_rgb.pivot_table(index='year', columns='rgb', aggfunc='size', fill_val

# Normalize the counts per yearS
sets_rgb_normalised = pivot_table.div(pivot_table.sum(axis=1), axis=0)
```

In [26]:
```
rgb_colors = list(sets_rgb_normalised)

years = sets_rgb_normalised.index
colors = []

for color in rgb_colors:
    colors.append(sets_rgb_normalised[color])

plt.style.use('seaborn-v0_8-bright')

plt.figure(figsize=(17, 8))

plt.stackplot(years, colors,
              colors = rgb_colors)

plt.title('Color composition of LEGO sets (stacked)', fontsize = 30)
plt.xlabel('Year')
plt.ylabel('% of parts with this color', fontsize = 15)
plt.xticks(rotation = 90)
plt.gca().yaxis.set_major_formatter(FuncFormatter(lambda y, _: '{:.0%}'.format(y)))  # F
plt.tight_layout()
plt.show()
```



With time more shades of color appear: they started with 5 colors in 1949, and 75 year later there are over 3 dozen shades! This was already a trend in the '80s, but became striking in the 2000s and they didn't stop ever since. Hmm, it does seem like a generational difference after all!

But there are other noteworthy trends too.

Interesing to note, that the color red became much less produced over time, and the same is true for blue and for white, although to a lesser extent. Also, there is a big "confusion" in colors in 1963 - this is apparently due to the fact that LEGO started experimenting with colors. It was also in this year that they started producing the "color" transparent. This color actually accounts for about 30% of their production in that year. It is also interesting to note, that black parts were not really produced before the '60s. Indeed, according to the official LEGO color timeline, they only started producing black bricks in 1958, which apparently had technical resons: it was not possible to color the bricks black before given the composition of plastic. Nevertheless, it seems that once LEGO found a stable color trend in the '60s, they pretty much sticked to this composition relative to each other. I also find it interesing to note that they abruptly changed the shade of grey around 2003, so all of a sudden 'retro' and new grey bricks in the same household stopped being visual friends.

The complexity (the shades of color and size of set) is definitely on the rise by LEGO. It seems to me that they optimise for selling new, color-matched sets, rather than being consistent with their colors in the past 75 years. Shame, really, since plastic bricks are more undestroyable than anything especially accross generations, and it would be more esthetic to work with a single shade of gray instead of 4 different shades with my kids. Well, it is what it is: a first world problem!

In [ ]: