In [1]:
```
import pandas as pd

# Specify the path to your Excel file
excel_file_path = 'C:/Users/juver/Downloads/Aid_Data_Example.xlsx'

# Read the Excel file into a pandas DataFrame
df = pd.read_excel(excel_file_path, skiprows=1)
```

In [2]:
```
# Display the DataFrame
print(df)
```

```
     aiddata_id  year          donor  count of donor              recipient  \
0      12191891  1998      Australia               1              Indonesia
1      27339565  2003      Australia               1              Singapore
2      34378730  2007      Australia               1               Colombia
3      13308187  1999      Australia               1             Timor-Leste
4      38835178  2008      Australia               1  Bilateral, unspecified
..          ...   ...            ...             ...                    ...
494    34085698  2007  United States               1  Bilateral, unspecified
495    26038958  2005  United States               1                  Brazil
496    50081282  2009  United States               1                Pakistan
497    50064569  2009  United States               1                 Ukraine
498    29738512  2006  United States               1                 Somalia

     count of receiver  commitment_amount_usd_constant  \
0                    1                    1.033230e+05
1                    1                    2.979690e+02
2                    1                    1.395560e+05
3                    1                    2.271530e+04
4                    1                    4.468550e+04
..                 ...                             ...
494                  1                    1.422430e+07
495                  1                    3.255560e+05
496                  1                    1.761800e+05
497                  1                    5.495000e+04
498                  1                    4.501060e+05

     coalesced_purpose_code                          coalesced_purpose_name
0                     24040              Informal/semi-formal fin. intermed.
1                     15105  Government and civil society, purpose unspecif...
2                     15130                    Legal and judicial development
3                     12220                                Basic health care
4                     15110          Public sector policy and adm. management
..                      ...                                               ...
494                   13020                        Reproductive health care
495                   41020                            Biosphere protection
496                   43010                                 Multisector aid
497                   15130                    Legal and judicial development
498                   72040                              Emergency food aid

[499 rows x 9 columns]
```

In [3]:
```
print(df.head)
```

```
<bound method NDFrame.head of      aiddata_id  year          donor  count of donor
recipient  \
0      12191891  1998      Australia                1              Indonesia
1      27339565  2003      Australia                1              Singapore
2      34378730  2007      Australia                1               Colombia
3      13308187  1999      Australia                1             Timor-Leste
4      38835178  2008      Australia                1  Bilateral, unspecified
..          ...   ...            ...              ...                     ...
494    34085698  2007  United States                1  Bilateral, unspecified
495    26038958  2005  United States                1                  Brazil
496    50081282  2009  United States                1                Pakistan
497    50064569  2009  United States                1                 Ukraine
498    29738512  2006  United States                1                 Somalia

     count of receiver  commitment_amount_usd_constant  \
0                    1                    1.033230e+05
1                    1                    2.979690e+02
2                    1                    1.395560e+05
3                    1                    2.271530e+04
4                    1                    4.468550e+04
..                 ...                             ...
494                  1                    1.422430e+07
495                  1                    3.255560e+05
496                  1                    1.761800e+05
497                  1                    5.495000e+04
498                  1                    4.501060e+05

     coalesced_purpose_code                          coalesced_purpose_name
0                     24040              Informal/semi-formal fin. intermed.
1                     15105  Government and civil society, purpose unspecif...
2                     15130                    Legal and judicial development
3                     12220                                Basic health care
4                     15110          Public sector policy and adm. management
..                      ...                                               ...
494                   13020                         Reproductive health care
495                   41020                             Biosphere protection
496                   43010                                  Multisector aid
497                   15130                    Legal and judicial development
498                   72040                              Emergency food aid

[499 rows x 9 columns]>
```

In [4]:
```
print(df.columns)
```

```
Index(['aiddata_id', 'year', 'donor', 'count of donor', 'recipient',
       'count of receiver', 'commitment_amount_usd_constant',
       'coalesced_purpose_code', 'coalesced_purpose_name'],
      dtype='object')
```

In [5]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import MinMaxScaler
```

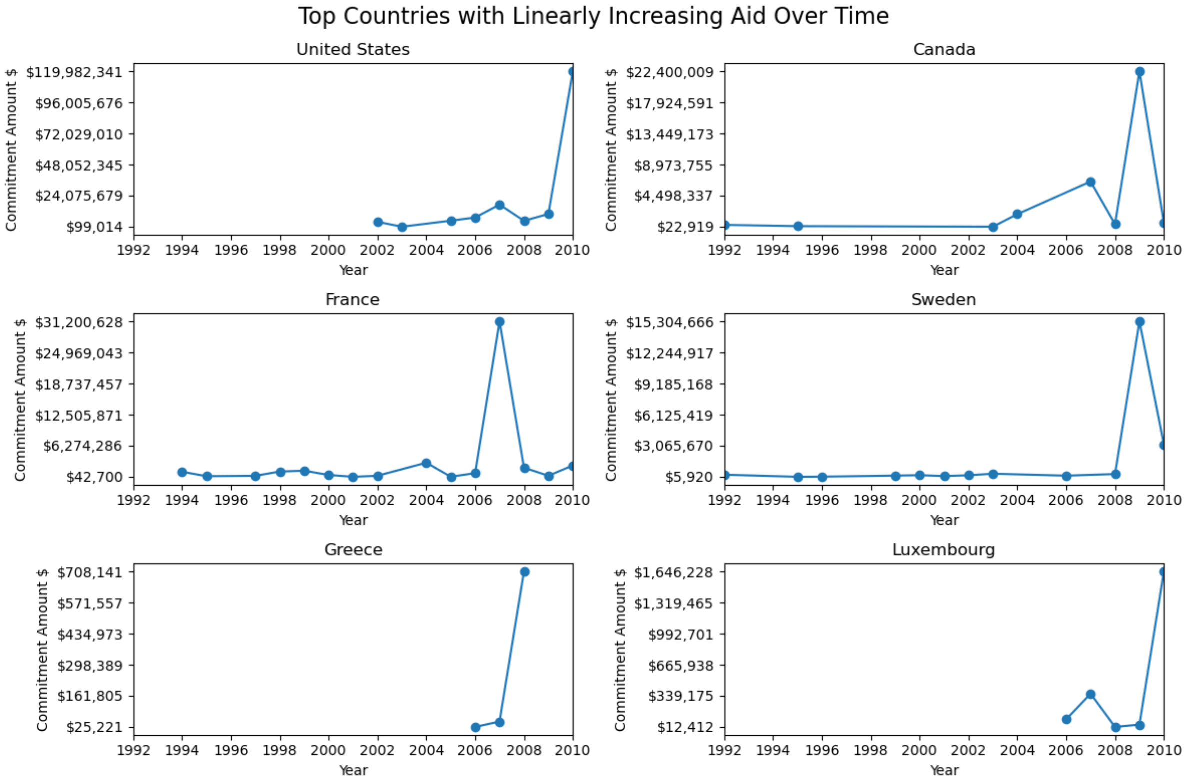
In [6]:
```
# Group by donor country and year by summing the commitment amounts for each donor
grouped_df = df.groupby(['donor', 'year'])['commitment_amount_usd_constant'].sum().

#Linear regression for each country
country_slopes = {}
for country, data in grouped_df.groupby('donor'):
    x = data['year'].values.reshape(-1, 1)
    y = data['commitment_amount_usd_constant'].values
    model = LinearRegression().fit(x, y)
```

```
    slope = model.coef_[0]
    country_slopes[country] = slope
```

In [7]:
```
# We need top 6
top_countries = sorted(country_slopes.items(), key=lambda x: x[1], reverse=True)[:6
top_country_names = [country[0] for country in top_countries]
#print(top_country_names)

# Filter the DataFrame for top countries
filtered_df = grouped_df[grouped_df['donor'].isin(top_country_names)]

# Maintaining same min and max year for all plots
global_min_year = filtered_df['year'].min()
global_max_year = filtered_df['year'].max()
```

In [8]:
```
# subplots
fig, axes = plt.subplots(nrows=3, ncols=2, figsize=(12, 8), sharey=False)  # Update
fig.suptitle('Top Countries with Linearly Increasing Aid Over Time', fontsize=16)

x_axis_min = 0
x_axis_max = 1

# Normalize
scaler = MinMaxScaler(feature_range=(x_axis_min, x_axis_max))

for i, ax in enumerate(axes.flatten()):
    if i < len(top_country_names):
        country = top_country_names[i]
        country_data = filtered_df[filtered_df['donor'] == country].sort_values('ye

        # Scale the y-axis values to the range 0 to 1
        scaled_values = scaler.fit_transform(country_data['commitment_amount_usd_co

        # Plot the data
        ax.plot(country_data['year'], scaled_values, label=country, marker='o')
        ax.set_title(country)
        ax.set_xlabel('Year')
        ax.set_ylabel('Commitment Amount $')

        # same range of years for all subplots
        ax.set_xlim(global_min_year, global_max_year)

        # Normalization to USD values
        inv_transformed_ticks = scaler.inverse_transform(ax.get_yticks().reshape(-1
        ax.set_yticklabels(["${:,.0f}".format(tick) for tick in inv_transformed_tic


plt.tight_layout()
plt.show()

caption_text = "The above subplots of the line graphs depict the linearly increasir
plt.figtext(0.5, 0.01, caption_text, wrap=True, horizontalalignment='center', fonts
```

```
C:\Users\juver\AppData\Local\Temp\ipykernel_38412\2686573009.py:30: UserWarning: FixedFormatter should only be used together with FixedLocator
  ax.set_yticklabels(["${:,.0f}".format(tick) for tick in inv_transformed_ticks])
C:\Users\juver\AppData\Local\Temp\ipykernel_38412\2686573009.py:30: UserWarning: FixedFormatter should only be used together with FixedLocator
  ax.set_yticklabels(["${:,.0f}".format(tick) for tick in inv_transformed_ticks])
C:\Users\juver\AppData\Local\Temp\ipykernel_38412\2686573009.py:30: UserWarning: FixedFormatter should only be used together with FixedLocator
  ax.set_yticklabels(["${:,.0f}".format(tick) for tick in inv_transformed_ticks])
C:\Users\juver\AppData\Local\Temp\ipykernel_38412\2686573009.py:30: UserWarning: FixedFormatter should only be used together with FixedLocator
  ax.set_yticklabels(["${:,.0f}".format(tick) for tick in inv_transformed_ticks])
C:\Users\juver\AppData\Local\Temp\ipykernel_38412\2686573009.py:30: UserWarning: FixedFormatter should only be used together with FixedLocator
  ax.set_yticklabels(["${:,.0f}".format(tick) for tick in inv_transformed_ticks])
C:\Users\juver\AppData\Local\Temp\ipykernel_38412\2686573009.py:30: UserWarning: FixedFormatter should only be used together with FixedLocator
  ax.set_yticklabels(["${:,.0f}".format(tick) for tick in inv_transformed_ticks])
```



Out[8]:

```
Text(0.5, 0.01, 'The above subplots of the line graphs depict the linearly increasing aid provided by the top 6 donor countries over the given years. United States have good linear rate compared to the other donor countries. The graph is presented on an amount-based x-axis scale, providing a clear representation of the increasing aid amounts, while the years are depicted on the y-axis.')
<Figure size 640x480 with 0 Axes>
```

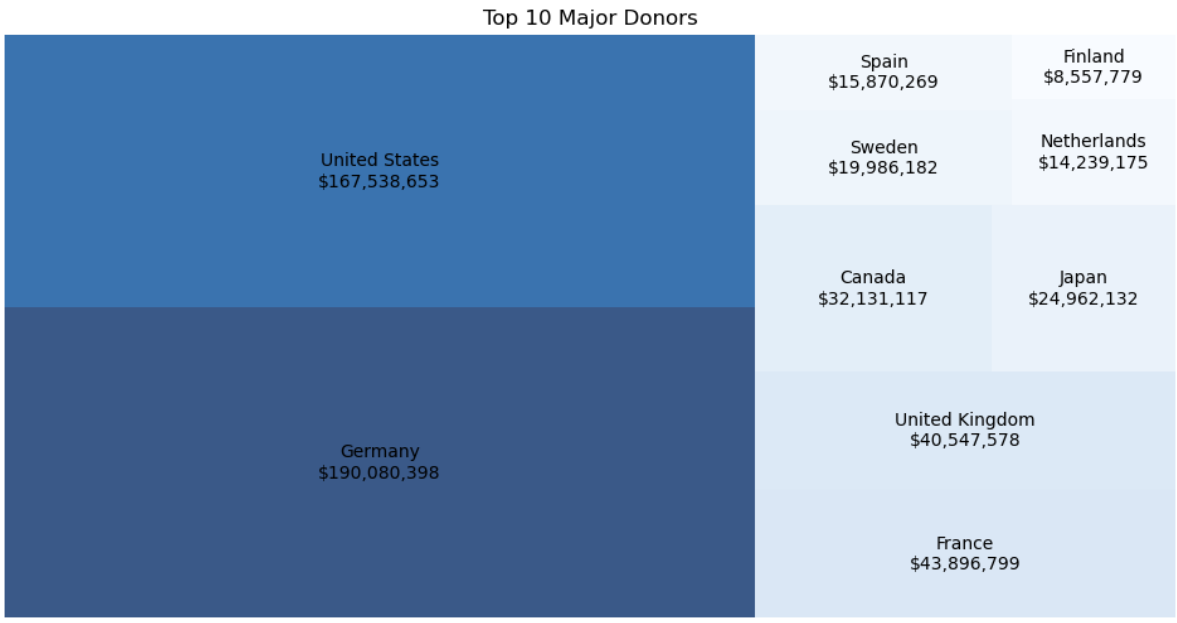
In [9]:

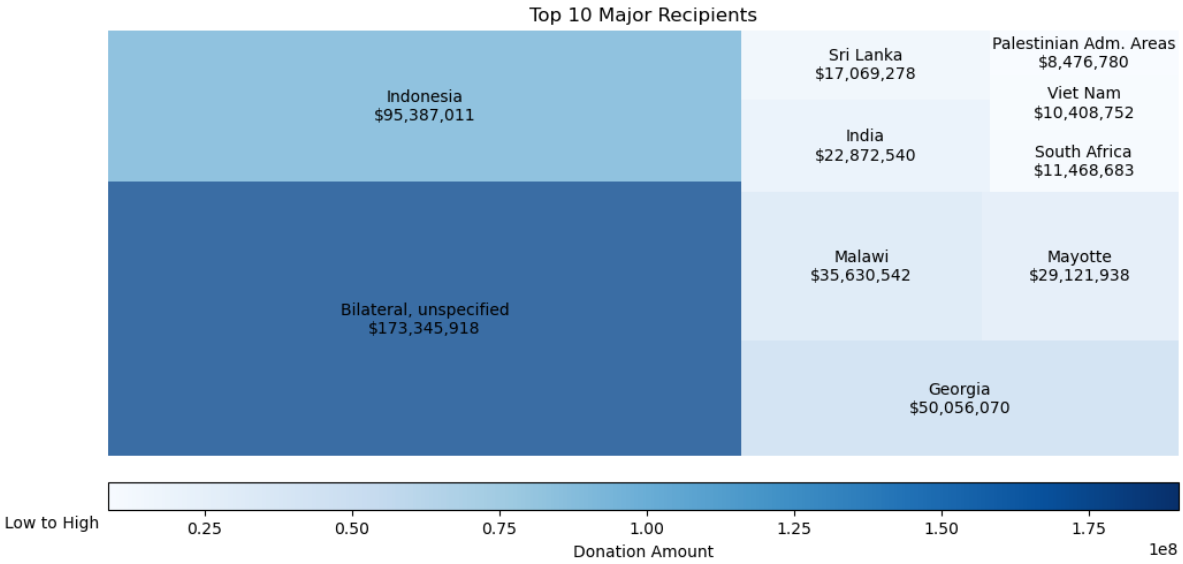
```python
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
import squarify
```

In [10]:

```python
N = 10

# group by donor and amount sum
total_aid_by_donor = df.groupby('donor')['commitment_amount_usd_constant'].sum().nl

# Normalize the data for color mapping
norm = mcolors.Normalize(vmin=min(total_aid_by_donor), vmax=max(total_aid_by_donor)
colors = [plt.cm.Blues(norm(value)) for value in total_aid_by_donor]

# donors
```

```
labels_donor = [f"{index}\n${value:,.0f}" for index, value in total_aid_by_donor.it

plt.figure(figsize=(12, 6))
squarify.plot(sizes=total_aid_by_donor, label=labels_donor, alpha=0.8, color=colors
plt.title(f'Top {N} Major Donors')
plt.axis('off')

total_aid_by_recipient = df.groupby('recipient')['commitment_amount_usd_constant'].

norm_recipient = mcolors.Normalize(vmin=min(total_aid_by_donor), vmax=max(total_aid
colors_recipient = [plt.cm.Blues(norm_recipient(value)) for value in total_aid_by_r

# recipients
labels_recipient = [f"{index}\n${value:,.0f}" for index, value in total_aid_by_reci

plt.figure(figsize=(12, 6))
squarify.plot(sizes=total_aid_by_recipient, label=labels_recipient, alpha=0.8, colo
plt.title(f'Top {N} Major Recipients')
plt.axis('off')

# Create a custom legend
sm = plt.cm.ScalarMappable(cmap=plt.cm.Blues, norm=norm)
sm.set_array([])  # You have to set_array for ScalarMappable.
cbar = plt.colorbar(sm, orientation='horizontal', pad=0.05, aspect=40)
cbar.set_label('Donation Amount')
cbar.ax.text(0, -0.5, ' High', va='center', ha='left')
cbar.ax.text(1, -0.5, 'Low to', va='center', ha='right')

plt.show()

caption_text = "a. [Donor] TreeMap representation of major aid provided by Donor co
plt.figtext(1,1,caption_text, wrap=True,  fontsize=10, color='gray')
```





Out[10]: Text(1, 1, 'a. [Donor] TreeMap representation of major aid provided by Donor countries to the recipient countries for a committed amount in USD. The graph depicts the details of the top donor countries like Germany, United States, France, United Kingdom where Germany as a significant donor. b. [Recipient] The graph depicts the details of the top recipient countries like Indonesia, Georgia, Malawi, Mayotte, India. The legend corresponds to the commitment amount, showcasing the scale of these donations where dark color represents major donations compared to the light color.')

<Figure size 640x480 with 0 Axes>