Explore the Gapminder Dataset with Plotly Express

About the Data: [Data Source (https://www.gapminder.org/tools/#$state$time$value=2007;;&chart-type=bubbles)](https://www.gapminder.org/tools/#$state$time$value=2007;;&chart-type=bubbles)

Task 1: Loading the Data

In this task, we will first demo Hans Rosling's visualization of the Gapminder data set. The interactive, animated visualization was shown to the audience of Hans' Ted talk in 2007.

It has gone one to become one of the most watched Ted talks of all time, and is a testament to the power of beautiful and informative data visualizations.

We will then be introduced to the project goals and learning outcomes.

Once we are familiarized with the Rhyme interface, we begin working with Jupyter Notebooks, a web-based interactive computational environment for creating notebook documents.

In [1]:
```
import plotly.offline as py
py.init_notebook_mode(connected=True)
import plotly.graph_objs as go
import pandas as pd
import numpy as np
```

In [2]:
```
import plotly.express as px
from plotly.figure_factory import create_table
```

```
In [3]: gapminder = px.data.gapminder()

        table = create_table(gapminder.head(10))
        py.iplot(table)
```

country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_num
Afghanistan	Asia	1952	28.801	8425333	779.4453145	AFG	4
Afghanistan	Asia	1957	30.331999999999997	9240934	820.8530296	AFG	4
Afghanistan	Asia	1962	31.997	10267083	853.1007099999999	AFG	4
Afghanistan	Asia	1967	34.02	11537966	836.1971382	AFG	4
Afghanistan	Asia	1972	36.088	13079460	739.9811057999999	AFG	4
Afghanistan	Asia	1977	38.438	14880372	786.11336	AFG	4
Afghanistan	Asia	1982	39.854	12881816	978.0114388000001	AFG	4
Afghanistan	Asia	1987	40.821999999999996	13867957	852.3959447999999	AFG	4
Afghanistan	Asia	1992	41.674	16317921	649.3413952000001	AFG	4
Afghanistan	Asia	1997	41.763000000000005	22227415	635.341351	AFG	4

Task 2: Quick Visualizations with Custom Bar Charts

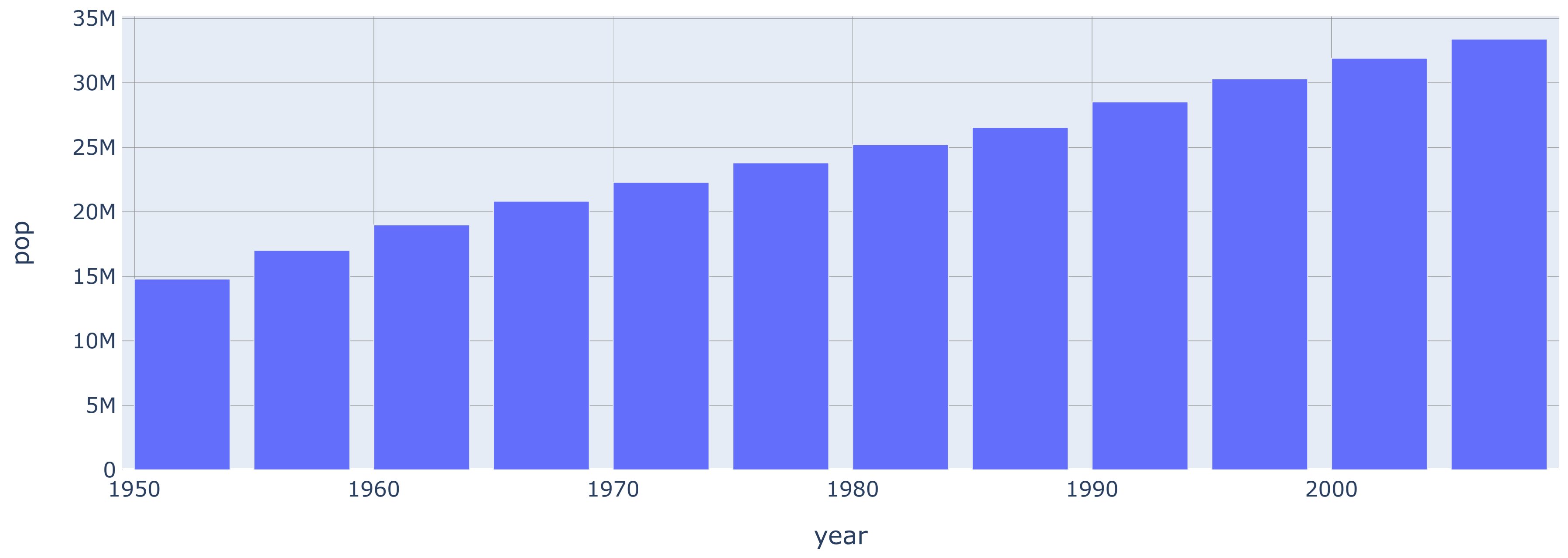
Now that we have imported the data, we are free to use plotly express to explore various facets of the rich data set.

Plotly Express functions accept tidy Pandas data frames.

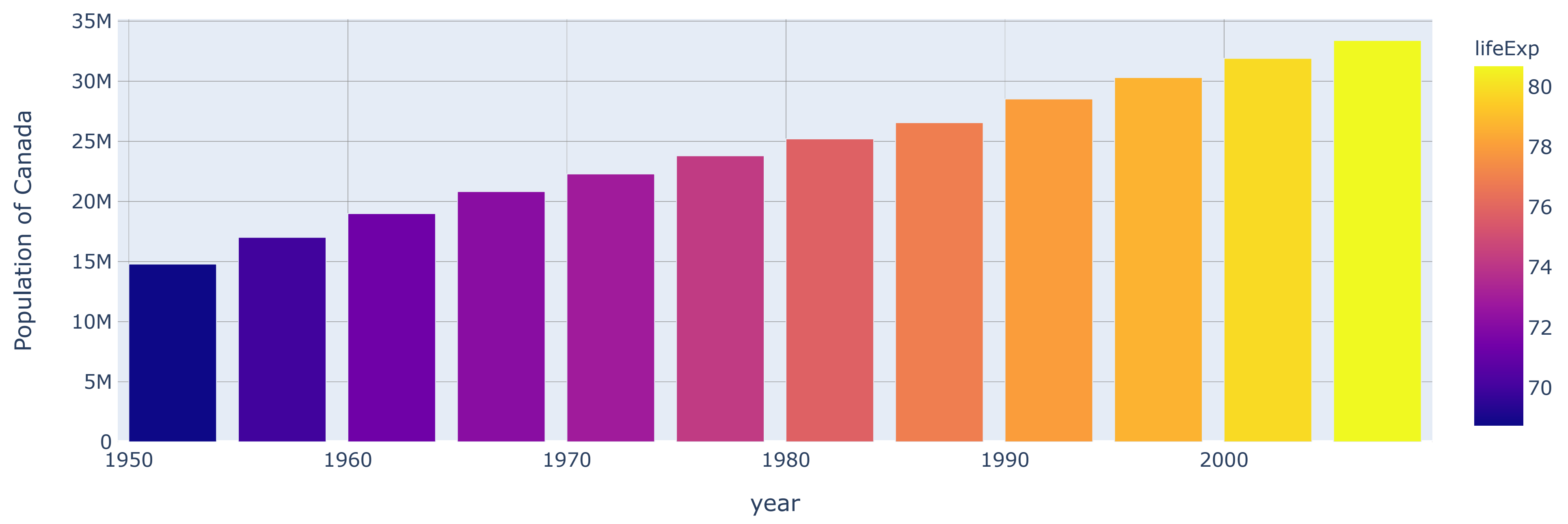
In this task, we will graph the population of Canada by year using a bar plot. In a bar plot, each row of the DataFrame is represented as a rectangular mark.

We will also customize the bar plot using keyword arguments to color the bars according to the average life expectancy.

```
In [4]: data_canada = px.data.gapminder().query("country == 'Canada'")
        fig = px.bar(data_canada, x='year', y='pop', height=400)
        fig.show()
```



```
In [5]: fig = px.bar(data_canada, x='year', y='pop',
                     hover_data = ['lifeExp', 'gdpPercap'], color='lifeExp',
                     labels={'pop': 'Population of Canada'}, height=400)
        fig.show()
```



```
In [ ]:
```

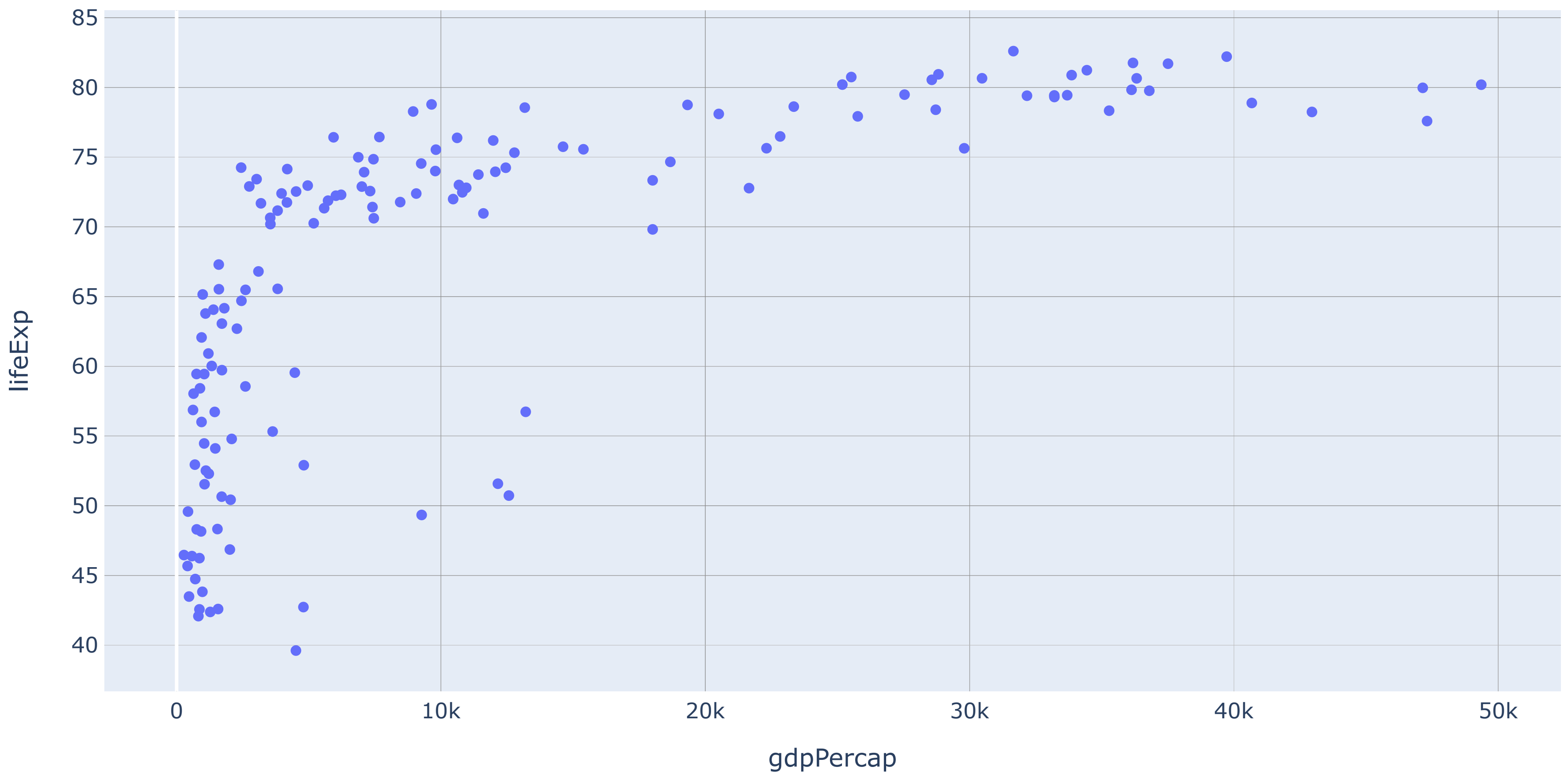
Task 3: Plot Life Expectancy vs GDP per Capita

Create a basic scatter plot showing life expectancy vs GDP per captita by country for 2007.

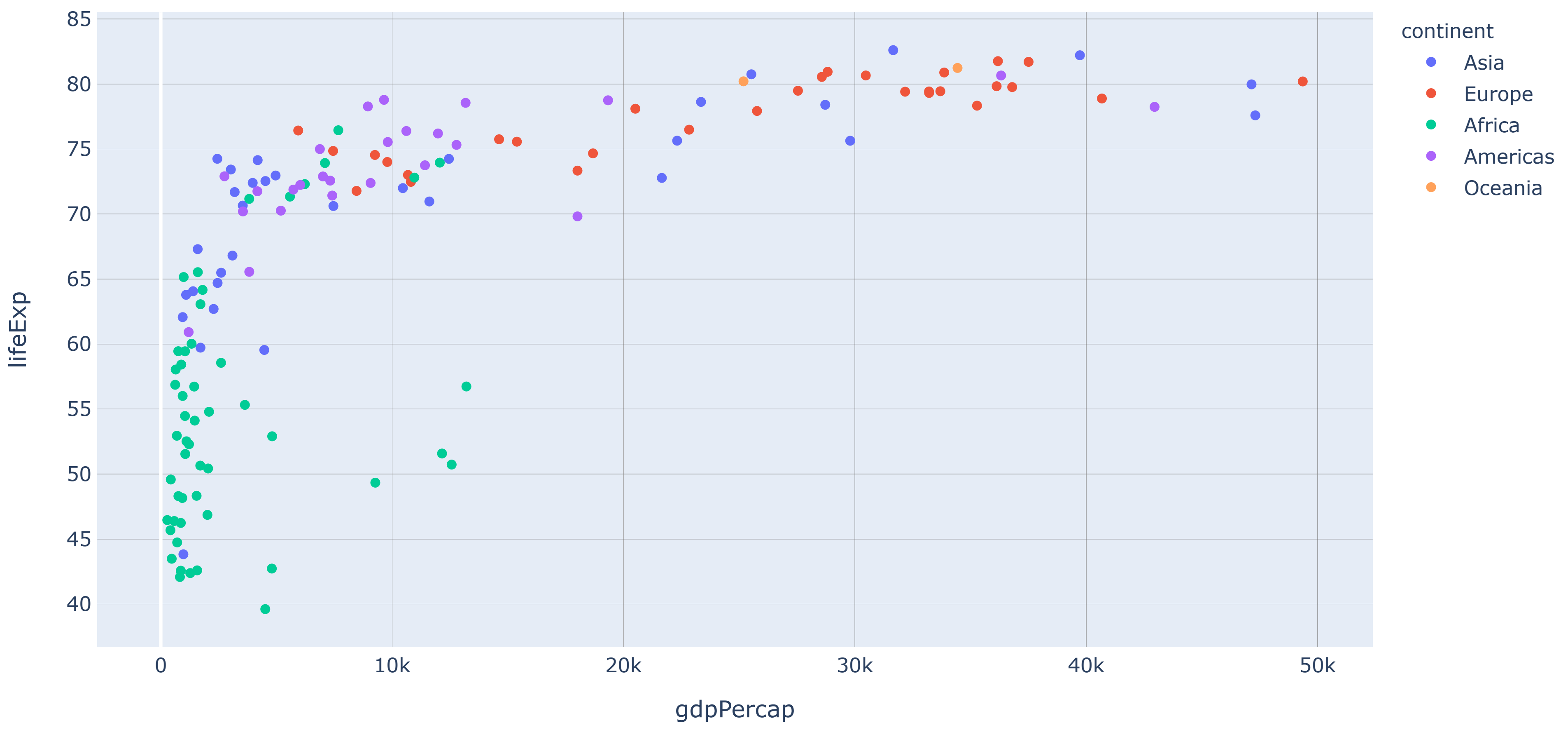
Next, break that down by continent by coloring the points using the color argument, while Plotly Express takes care of assigning the default colors, setting up the legend, etc.

```
In [6]: gapminder2007 = gapminder.query("year == 2007")

        px.scatter(gapminder2007, x="gdpPercap", y="lifeExp")
```



```
In [7]: px.scatter(gapminder2007, x="gdpPercap", y="lifeExp", color='continent')
```



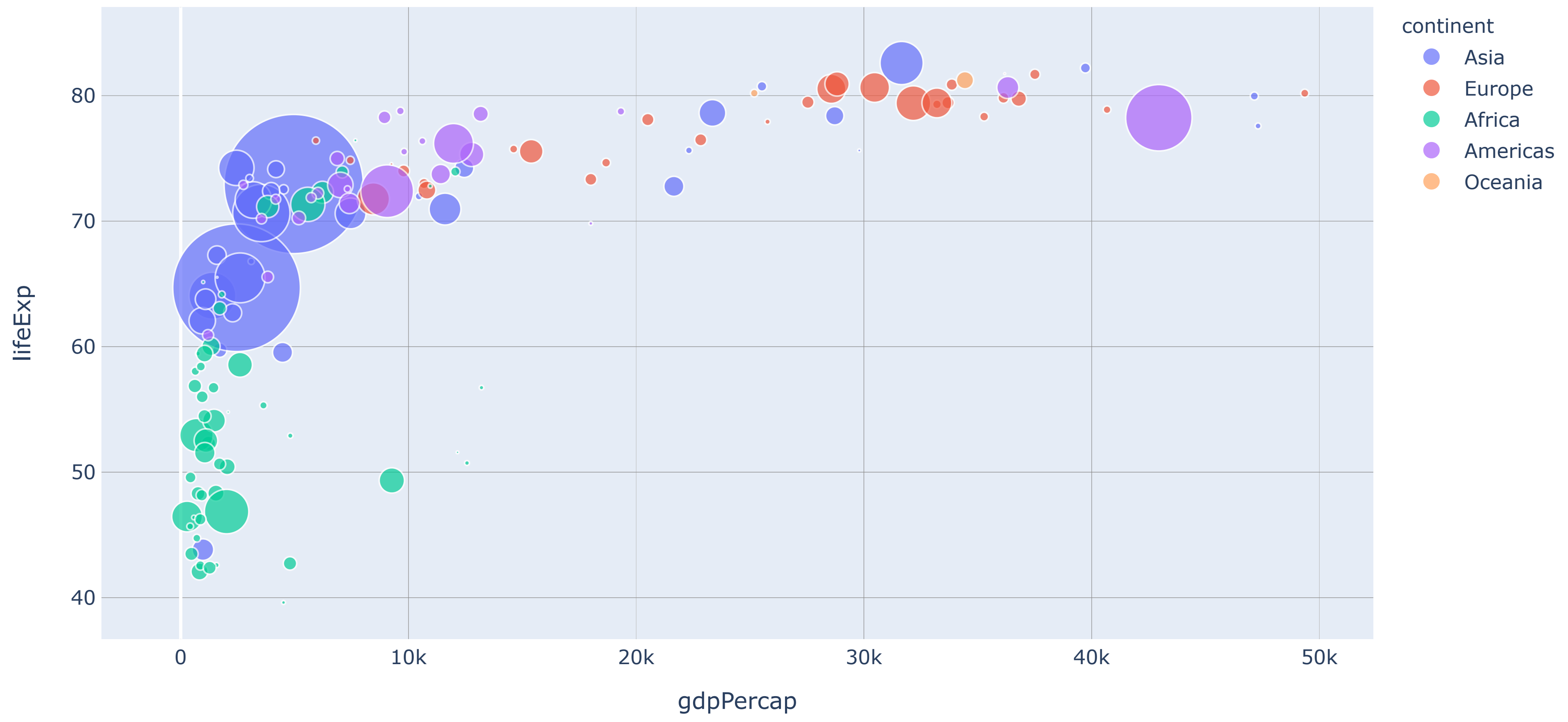
```
In [ ]:
```

Task 4: Customize Interactive Bubble Charts

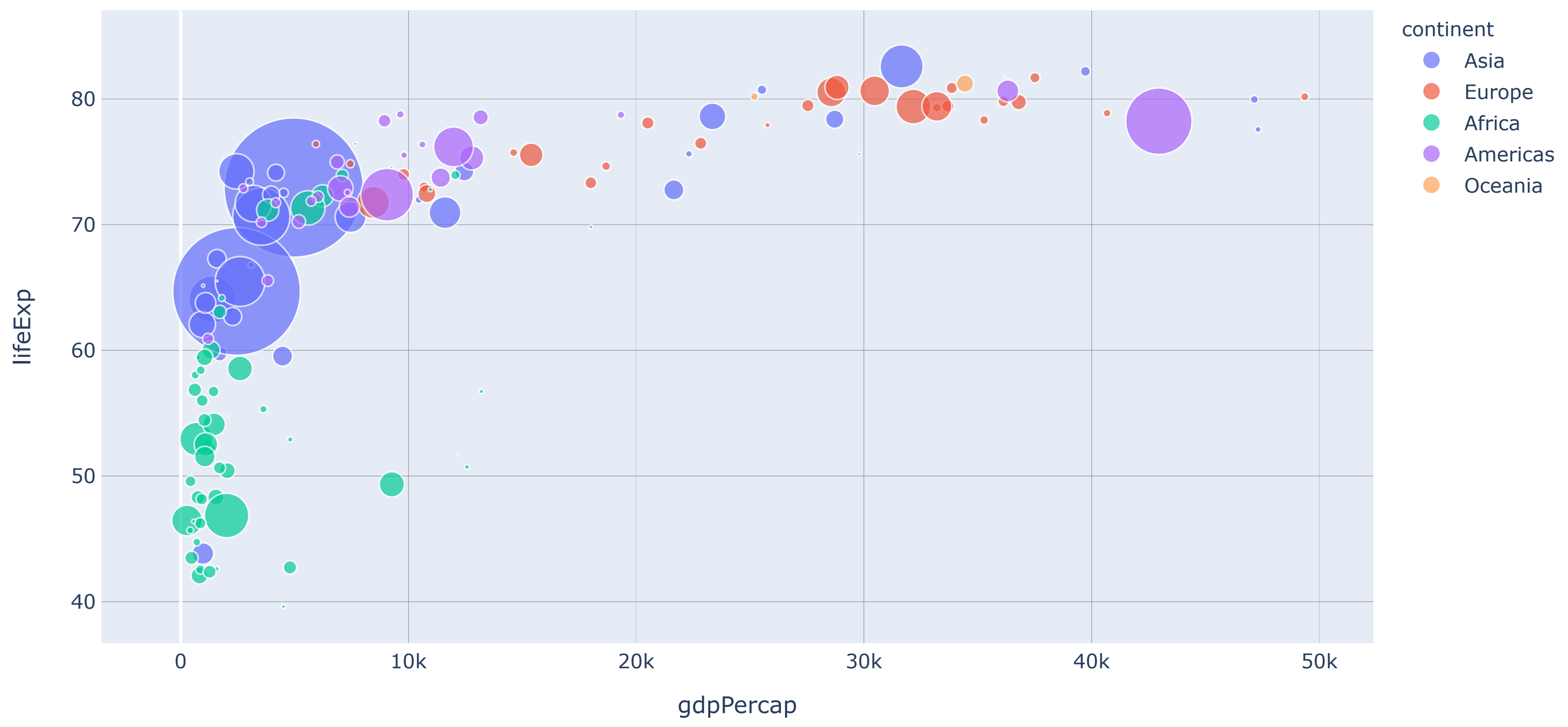
Each point in the previous plot is a country. To scale the points by the country population, simply pass in the size argument!

If you're curious about the identity of a particular point, you can add a hover_name to display the country name. (You will never again have to worry about what a particular outlier represents. You can simply mouse over the point of interest and the hover_name will identify it.)

In [8]:
```
px.scatter(gapminder2007, x="gdpPercap", y="lifeExp", color='continent',
           size="pop", size_max=60)
```



In [9]:
```
px.scatter(gapminder2007, x="gdpPercap", y="lifeExp", color='continent',
          size="pop", size_max=60, hover_name="country")
```



In [ ]:
```
```

Task 5: Create Interactive Animations and Facet Plots

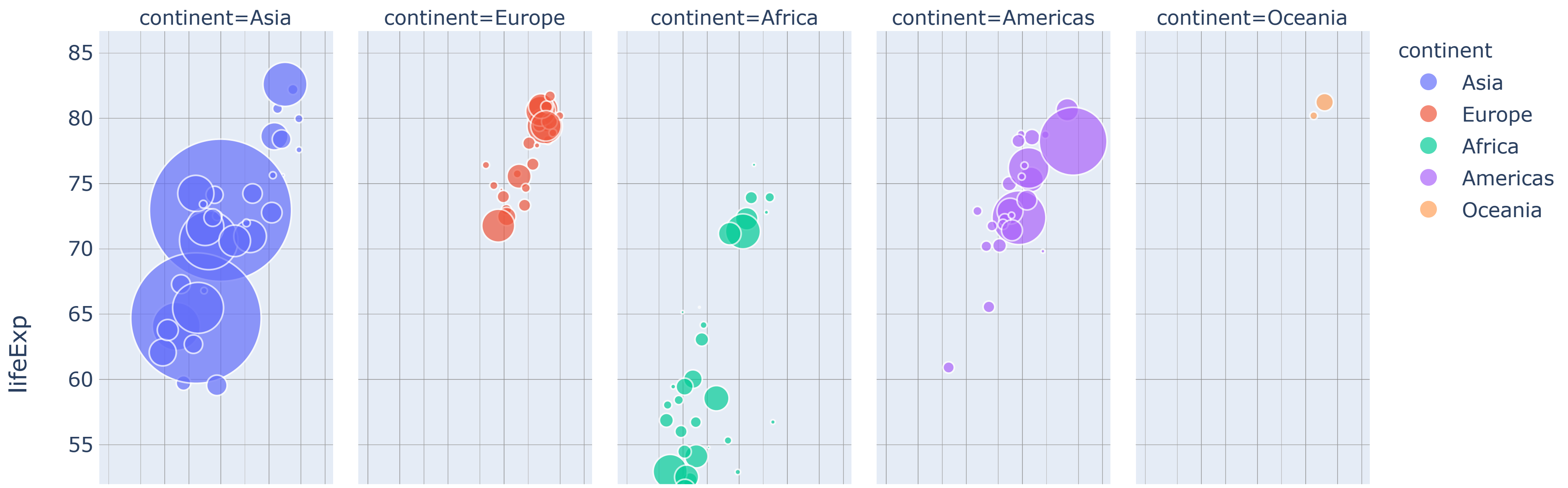
We are able to easily interact with the plots we have created so far. Try mousing over points, clicking or double-clicking on legend items, or using the modebar that appears when you move your mouse into the frame to control the behaviour of click-drag interactions (zoom, pan, select).

Create facet plots to pick apart the continents, just as easily as coloring your points, with facet_col="continent", and make the x-axis logarithmic to see things more clearly.

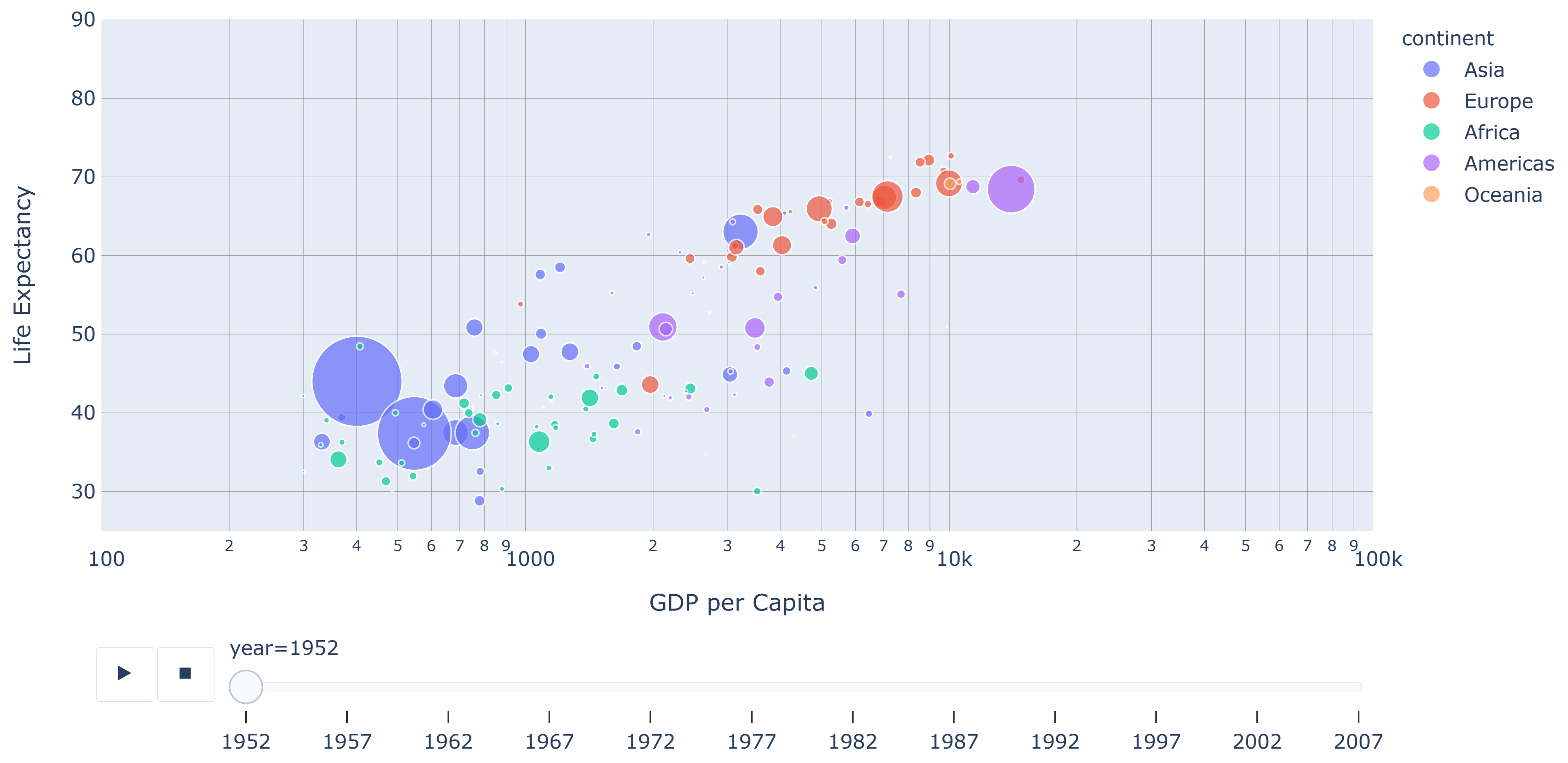
Maybe you're interested in more than just 2007 want to see how the chart evolved over time. You can animate it by setting animation_frame="year" and animation_group="country" to identify which circles match which ones across frames.

Provide prettier labels that get applied throughout the figure, in legends, axis titles and hovers. Also provide some manual bounds so the animation looks nice throughout.

```
In [10]: px.scatter(gapminder2007, x="gdpPercap", y="lifeExp", color='continent',
                    size="pop", size_max=60, hover_name="country", facet_col="continent",log_x=True)
```



```
In [11]: px.scatter(gapminder, x="gdpPercap", y="lifeExp", color='continent',
                    size="pop", size_max=60, hover_name="country", animation_frame='year',
                    animation_group="country", log_x=True, range_x=[100, 100000], range_y=[25, 90],
                    labels=dict(pop="Population", gdpPercap="GDP per Capita", lifeExp="Life Expectancy"))
```

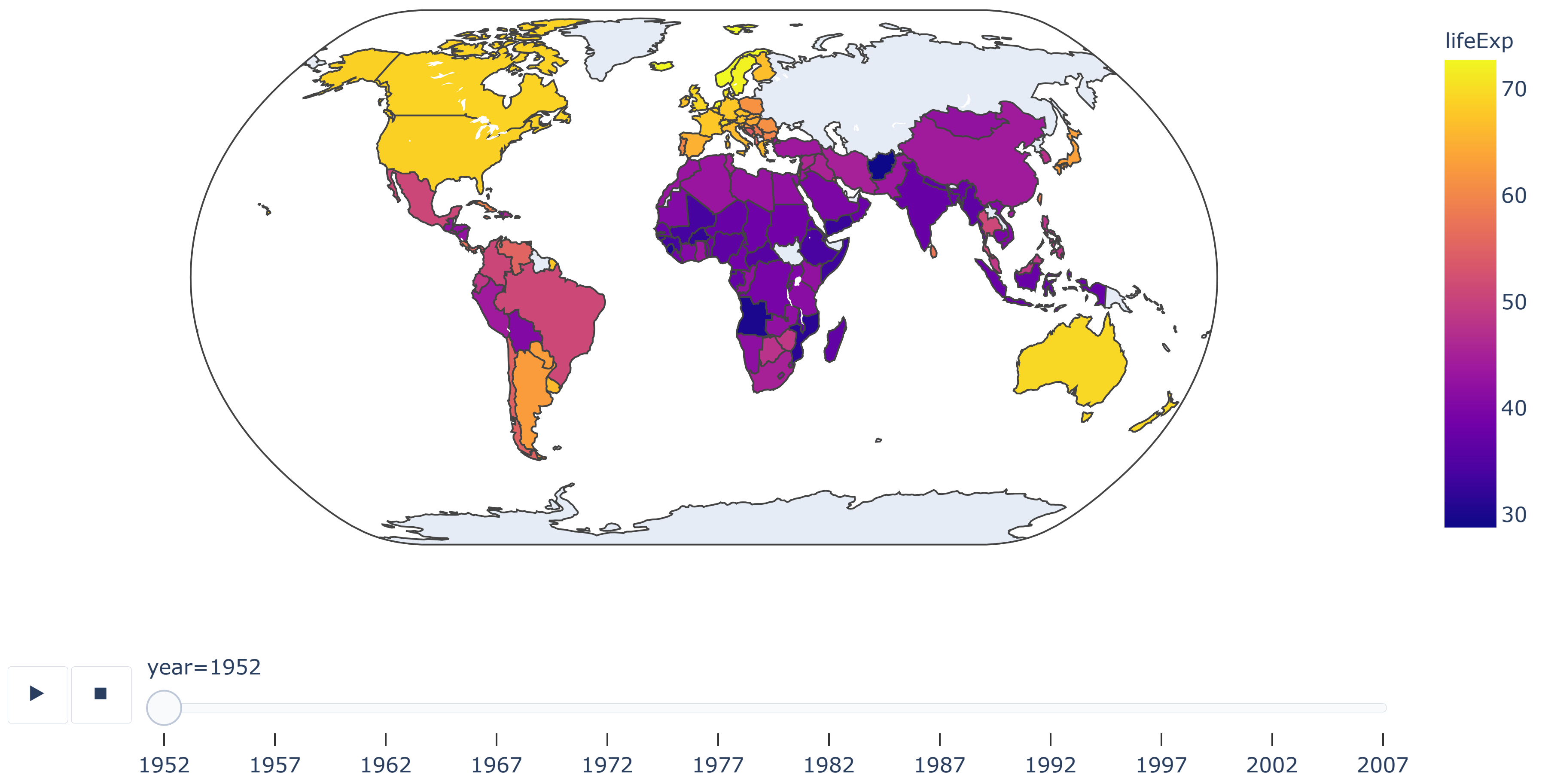


```
In [ ]:
```

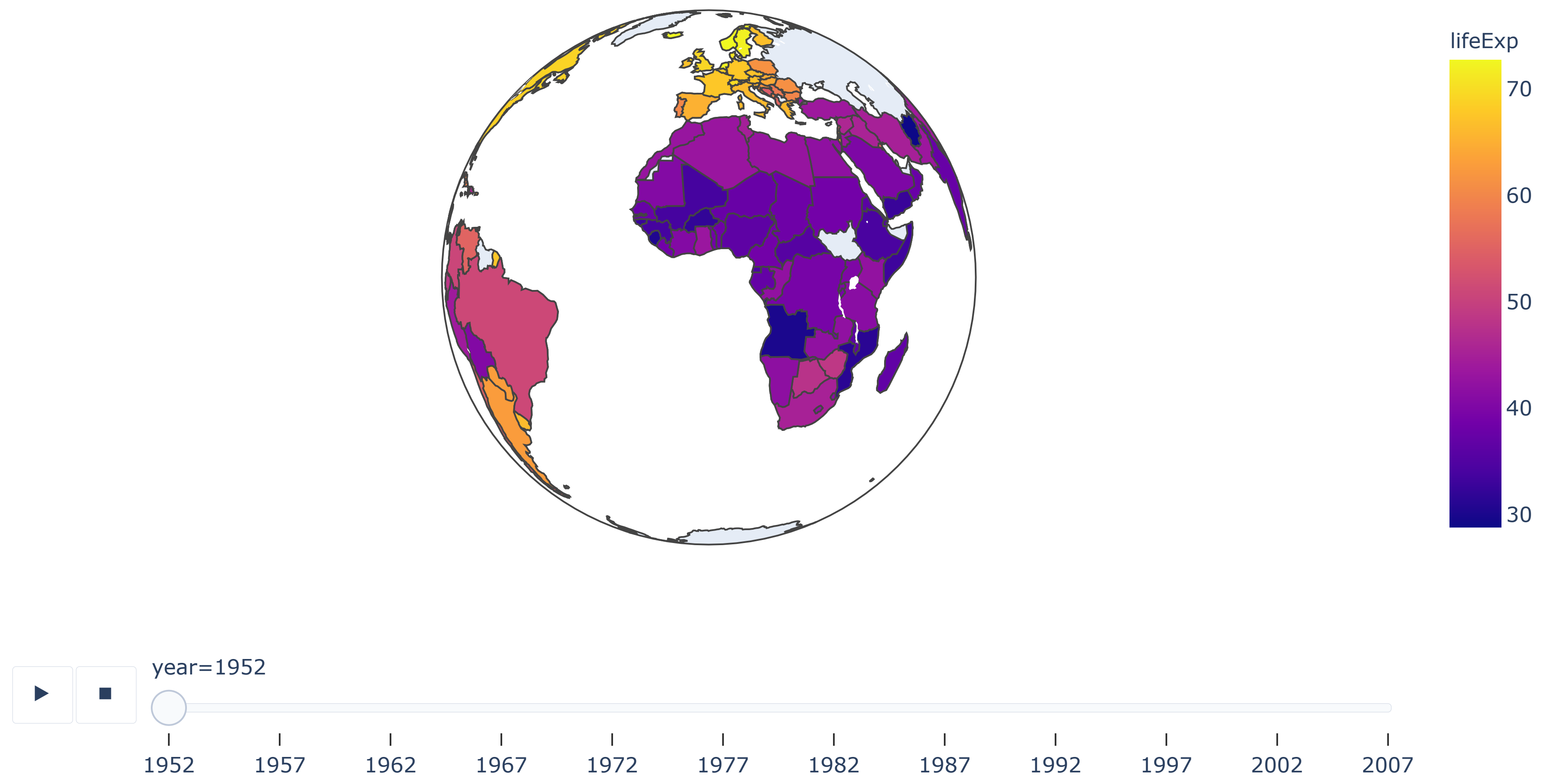
Task 6: Represent Geographic Data as Animated Maps

As this is geographic data, you can also represent it as an animated map, which makes it clear that Plotly Express can make a lot more than just scatter plots, and that this dataset is missing data for the former Soviet Union.

```
In [12]: px.choropleth(gapminder, locations="iso_alpha", color="lifeExp", hover_name="country",
                      animation_frame="year", color_continuous_scale = px.colors.sequential.Plasma,
                      projection = "natural earth")
```



```
In [13]: px.choropleth(gapminder, locations="iso_alpha", color="lifeExp", hover_name="country",
                       animation_frame="year", color_continuous_scale = px.colors.sequential.Plasma,
                       projection = "orthographic")
```



```
In [13]:
```