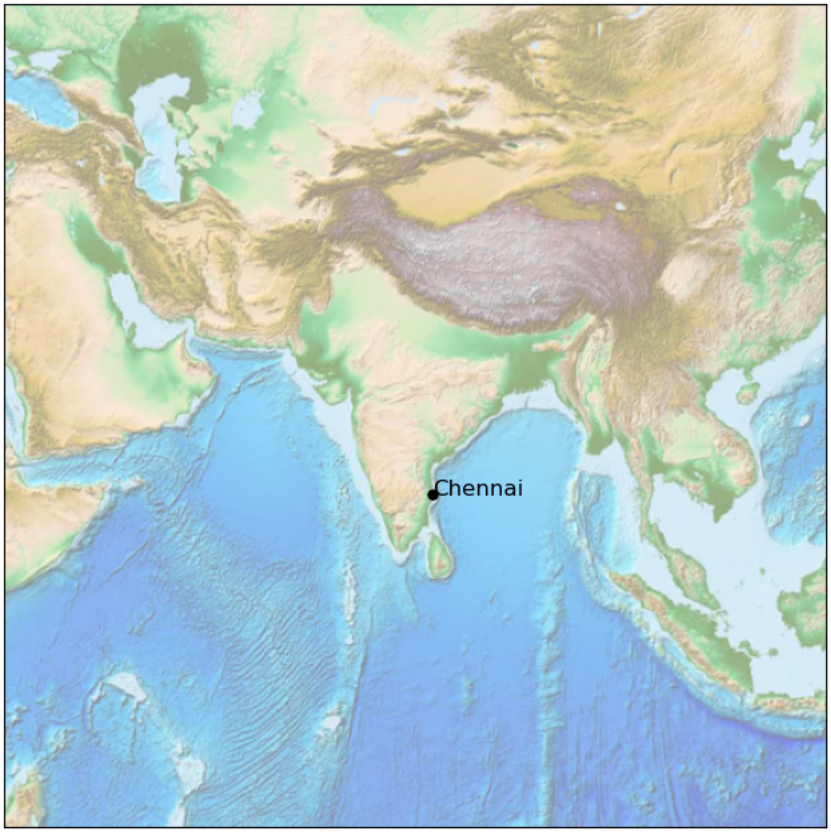
In [81]:
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.basemap import Basemap
```

In [141]:
```
#plt.figure(figsize=(8,8))
#m=Basemap(projection="ortho",lat_0=0,lon_0=0)
#m.bluemarble(scale=0.5)
```

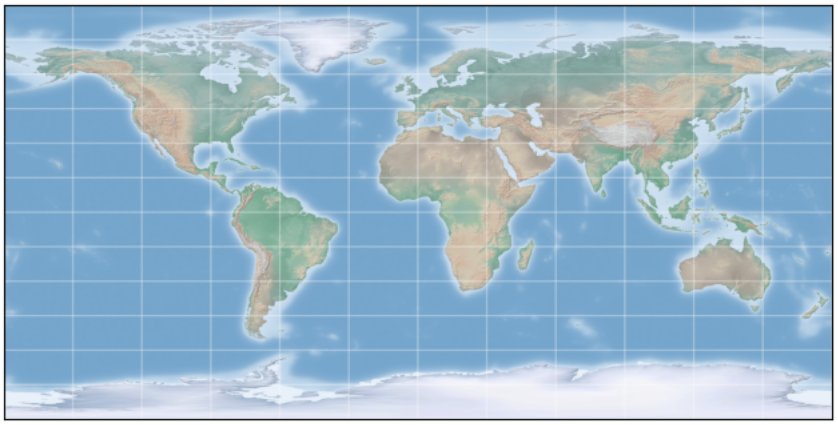
In [142]:
```
fig = plt.figure(figsize=(8, 8))
m=Basemap(projection="lcc",resolution=None,width=8E6,height=8E6,lat_0=20,lon_0=78.6569)
#lcc is a conic map projection eg used are Chennai coordinates
m.etopo(scale=0.5,alpha=0.5)#etopo is used for displaying map with topological description
# Map(long, lat) to (x, y) for plotting
x,y=m(80.2707,13.0827)
plt.plot(x,y,"ok",markersize=5)
plt.text(x,y,"Chennai",fontsize=12)
```

Out[142]: Text(4176049.1929835128, 3229824.522019725, 'Chennai')



In [143]:
```
from itertools import chain
def draw_map(m, scale=0.2):
    # draw a shaded-relief image
    m.shadedrelief(scale=scale)
    # lats and longs are returned as a dictionary
    lats = m.drawparallels(np.linspace(-90, 90, 13))
    lons = m.drawmeridians(np.linspace(-180, 180, 13))
    # keys contain the plt.Line2D instances
    lat_lines = chain(*(tup[1][0] for tup in lats.items()))
    lon_lines = chain(*(tup[1][0] for tup in lons.items()))
    all_lines = chain(lat_lines, lon_lines)
    # cycle through these lines and set the desired style
    for line in all_lines:
        line.set(linestyle='-', alpha=0.3, color='w')
```

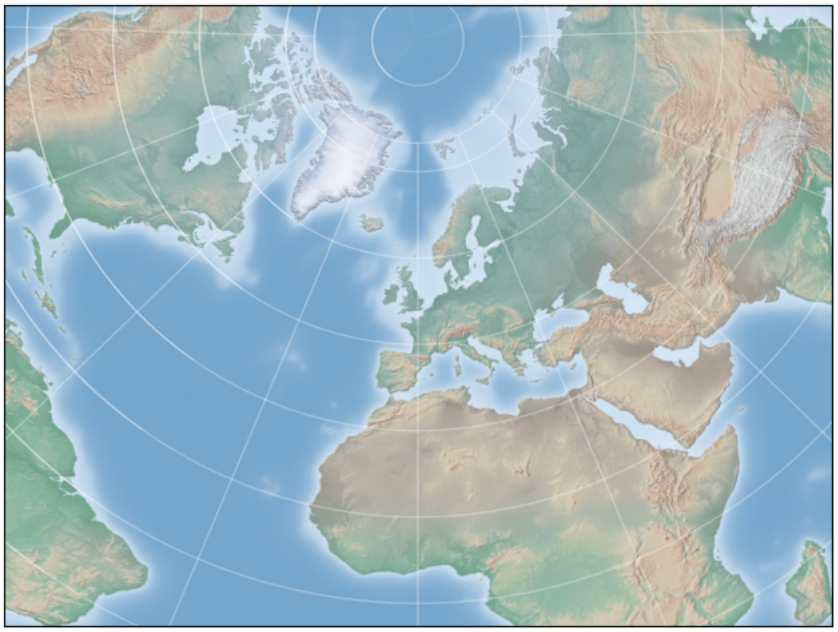
In [144]:
```
fig=plt.figure(figsize=(8, 6),edgecolor='w')
m=Basemap(projection="cyl",resolution=None,llcrnrlat=-90,urcrnrlat=90,llcrnrlon=-180,urcrnrlon=180)
#llcrnlat->lower left corner latitude
#urcrnlon->upper right corner longitude
draw_map(m)
#drawback in cyclindrical is it shows extreme distortions at pole regions
```



In [145]:
```
#pseudo cylindrical projections
#fig=plt.figure(figsize=(8,6),edgecolor="w")
#m=Basemap(projection="moll",resolution=None,lat_0=0,lon_0=0)#molleweide projection,lat_0 ad lon_0 are central coordinates
#draw_map(m)
#other pseudo cyclindrical projections are sinu for sine and robin for robinson projection
```

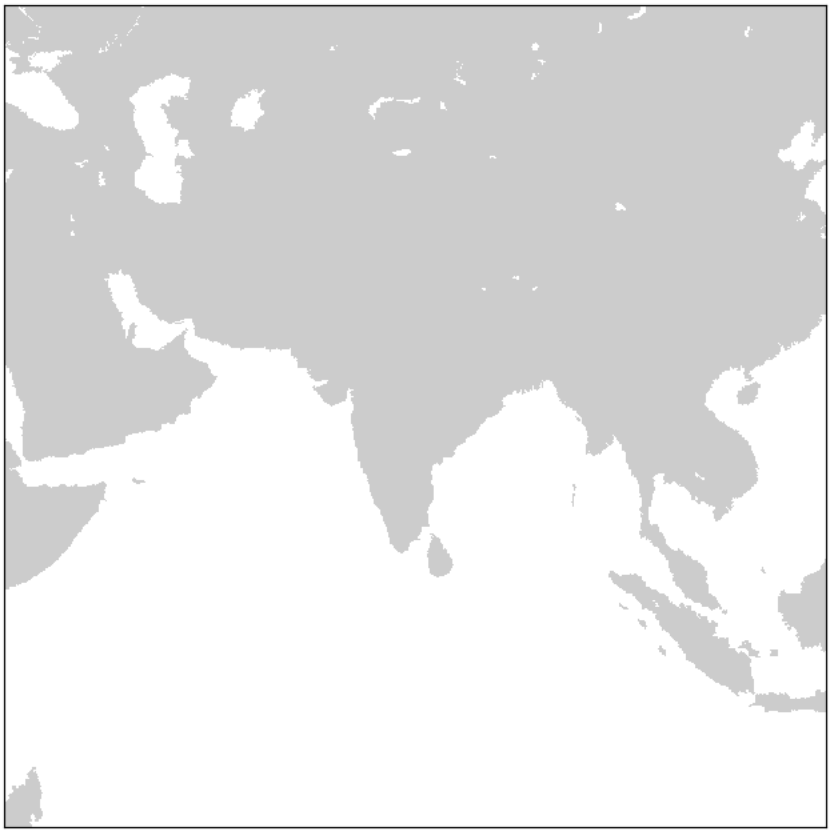
In [146]:
```
#persepective projections
#fig = plt.figure(figsize=(8, 8))
#m=Basemap(projection="ortho",lat_0=50,lon_0=-100)
#draw_map(m)'''
```

In [147]:
```
fig = plt.figure(figsize=(8, 8))
m = Basemap(projection="eqdc", resolution=None,lon_0=0, lat_0=50, lat_1=45, lat_2=55,width=1.6E7,height=1.2E7)
draw_map(m)#equidistant cone
```



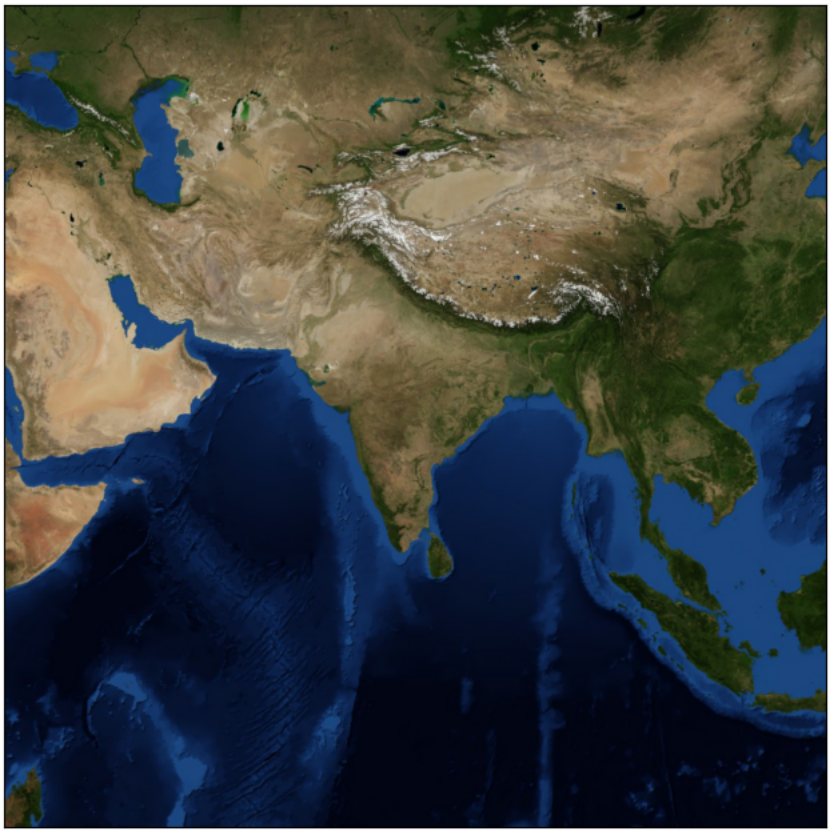
In [148]:
```
#drawing map backgorund
fig = plt.figure(figsize=(8, 8))
m=Basemap(projection="lcc",resolution=None,width=8E6,height=8E6,lat_0=20,lon_0=78.6569)
m.drawlsmask()#drawing mask between land and sea
```

Out[148]: <matplotlib.image.AxesImage at 0x29f3640ac40>



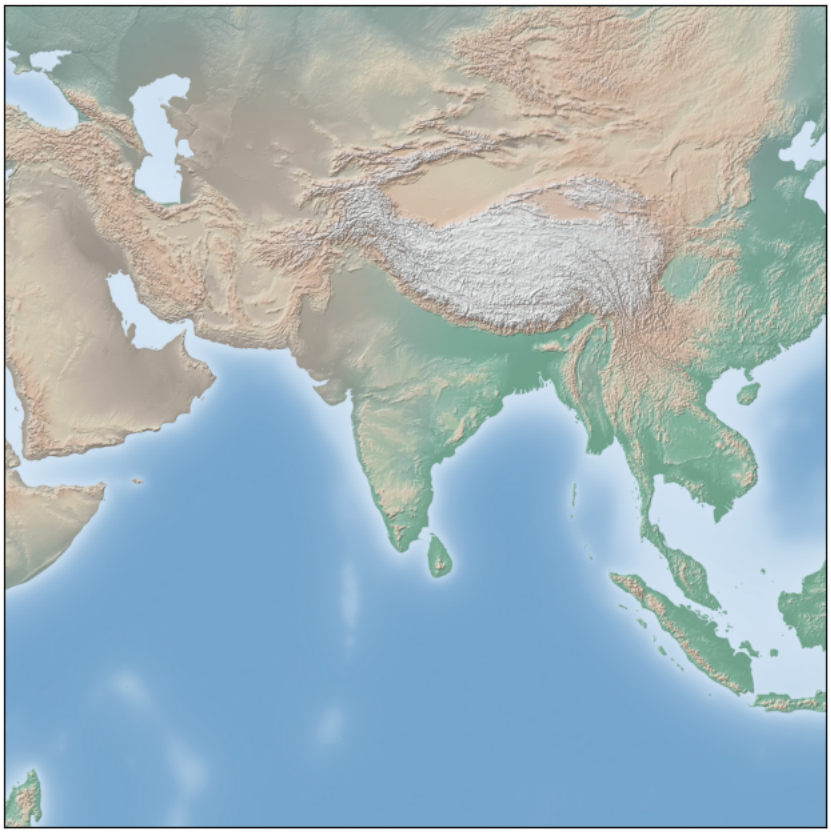
In [149]:
```
fig = plt.figure(figsize=(8, 8))
m=Basemap(projection="lcc",resolution=None,width=8E6,height=8E6,lat_0=20,lon_0=78.6569)
m.bluemarble()#project nasa blue marble image
```

Out[149]: <matplotlib.image.AxesImage at 0x29f37dbc370>



In [150]:
```
fig = plt.figure(figsize=(8, 8))
m=Basemap(projection="lcc",resolution=None,width=8E6,height=8E6,lat_0=20,lon_0=78.6569)
m.shadedrelief()
```

Out[150]: <matplotlib.image.AxesImage at 0x29f31a1eaf0>



In [151]:
```
#resolution options are crude(c),low(l),high(h),intermediate(i),full(f) or None
```

In [152]:
```
import pandas as pd
cities = pd.read_csv('california_cities.csv')
# Extract the data we're interested in
lat = cities['latd'].values
lon = cities['longd'].values
population = cities['population_total'].values
area = cities['area_total_km2'].values
```

In [153]:
```
#draw map background
fig=plt.figure(figsize=(8,8))
m=Basemap(projection="lcc",resolution="c",lat_0=37.5,lon_0=-119,width=1E6,height=1.2E6)
m.shadedrelief()
m.drawcoastlines(color="gray")#draws coastal lines
m.drawcountries(color="gray")#draws all countries with borders in gray
m.drawstates(color="gray")#draws us states with gray color
m.scatter(lon,lat,latlon=True,c=np.log10(population),s=area,cmap='Reds',alpha=0.5)#scatter plot based on population
#create colorbar and legend
plt.colorbar(label=r'$\log_{10}({\rm population})$')
plt.clim(3, 7)
#creating dummy points legend
for i in [100,300,500]:
    plt.scatter([],[],c="k",s=i,alpha=0.5,label=str(a) +' km$^2$')
plt.legend(scatterpoints=1,frameon=False,labelspacing=1,loc="lower left")
```

Out[153]: <matplotlib.legend.Legend at 0x29f37e5f310>