folium

June 11, 2020

O ***Folium*** é um pacote que possibilita a criação de mapas *online*, facilitando a visualização dos dados manipulados no *Python* em um mapa que usa a biblioteca JavaScript ***Leaflet***.

A biblioteca possui vários conjuntos de blocos internos do OpenStreetMap, Mapbox e Stamen, além de suportar conjuntos de blocos customizados com as chaves da API Mapbox ou Cloudmade. ***Folium*** suporta sobreposições de imagem, vídeo, GeoJSON e TopoJSON.

{: .box-warning} **Aviso:** Esse *post* tem a finalidade de mostrar os comandos básicos e me deixar com uma "cola" rápida para meu uso cotidiano. Todas os códigos são exemplificativos e podem/devem ser alterados, indicando o nome dos arquivos e diretórios corretamente.

{: .box-note} **Nota:** É possível acessar esse *post* em formato ***pdf***, diretamente por meio do **repositório do GitHub** ou ainda

```
#, de maneira interativa, usando o [![Binder](https://mybinder.org/badge_logo.
↪svg)](https://mybinder.org/v2/gh/michelmetran/package_folium/master).
```

1 Importando Bibliotecas

As bibliotecas básicas, ou *packages*, necessárias para criação do mapa são: - O ***Pandas***, que tem a missão de trabalhar com dados, criar *subsets*, selecionar e filtros dados e; - O ***Folium***, que é a biblioteca que cria, na prática, o mapa!

```
import os
import json
import folium
import pandas as pd
import geopandas as gpd
from datetime import date
```

2 Criando um mapa

Basta um par de coordenadas – que pode ser obtida facilmente no *link* de qualquer endereço usando ***Google Maps*** – e um nível de zoom que o mapa já está criado.

```
folium.Map(
    location=[-23.9619271, -46.3427499],    # Define coordenadas iniciais
    #min_zoom=6,                            # Define qual o menor zoom
```

```
    #max_zoom=14,                              # Define qual o maior zoom
    #no_wrap=True,
    max_bounds=True,
    zoom_start=12,                             # Define o zoom do início
    control_scale=True,                        # Define se terá barra de escala
    #width=width,
    #height=height,
)
```

Utilizando um conjunto de dados apresentado em **Jessica Temporal**, contendo coordenadas geográficas de empresas, podemos extrair uma empresa específica e plotar no mapa, ou ainda trabalhar de outras maneiras com esses dados.

```
# Lendo e filtrando dados
df = pd.read_csv('data/empresas.xz')
df = df[df['state'] == 'SP']
df = df[df['city'] == 'SANTOS']

df.dtypes
#df.head(10)
```

2.1 Inserindo algumas coordenadas

```
# Cria o mapa
m = folium.Map(
    location=[0,0],
    zoom_start=15,
    min_zoom = 10,
    max_zoom = 15,
    max_bounds = True
    #min_lat = min_lat,
    #max_lat = max_lat,
    #min_lon = min_long,
    #max_lon = max_long
)

# Extrai informações de duas empresas
df1 = df.iloc[0]
df2 = df.iloc[1]

# Adiciona no mapa tais empresas
folium.Marker(
    location=[df1['latitude'], df1['longitude']],
).add_to(m)

folium.Marker(
    location=[df2['latitude'], df2['longitude']],
```

```
).add_to(m)

# Limite o zoom às feiçoes inseridas
m.fit_bounds(m.get_bounds())
m.get_bounds()[0]
#min_lat, min_long = m.get_bounds()[1]
#max_lat, max_long = m.get_bounds()[0]

# Apresenta o mapa
m
```

2.2 Inserindo multiplas coordenadas

```
# Cria o mapa
m = folium.Map(
    location=[-23.9619271,-46.3427499],
    zoom_start=12
)

# Adiciona todas as empresas selecionadas
for index, row in df.iterrows():
    folium.Marker(
        location=[row['latitude'], row['longitude']],
        tooltip=row['neighborhood'],
    ).add_to(m)

# Apresenta o mapa
m
```

3 Inserindo feições

As feições que são possiveis de apresentar são àquelas típicas do geoprocessamento: - Pontos; - Linhas; - Polígonos

Abaixo são apresentados alguns tipos de marcadores.

PS: Um site relevante para obter as cores em formato hexadecimal (bastante utilizado na definição dos *styles* do **folium** é usando o site ColorBook.io).

3.1 Pontos Simples

```
# Cria o mapa
m = folium.Map(
    location=[-23.9619271,-46.3427499],
    zoom_start=12
)
```

```
# Cria cores para as tags
colors = {
    'PONTA DA PRAIA': 'pink',
    'CENTRO': 'blue',
    'GONZAGA': 'green',
    'JOSÉ MENINO': 'red',
    'EMBARE': 'beige',
    'MACUCO': 'blue',
    'VILA MATHIAS': 'lightblue',
    'POMPEIA': 'red',
    'APARECIDA': 'purple'
}

# Adiciona as diferentes empresas com cores por bairros
for index, row in df.iterrows():
    if row['neighborhood'] in colors.keys():
        folium.Marker(
            location=[row['latitude'], row['longitude']],
            popup=row['name'],
            tooltip=row['neighborhood'],
            icon=folium.Icon(color=colors[row['neighborhood']], icon='leaf')
        ).add_to(m)

# Apresenta o mapa
m
```

3.2 Ponto com Buffer

```
# Cria o mapa
m = folium.Map(
    location=[-23.9619271,-46.3427499],
    zoom_start=12
)

# Adiciona as diferentes empresas com cores por bairros
for index, row in df.iterrows():
    if row['neighborhood'] in colors.keys():
        folium.CircleMarker(
        location=[row['latitude'], row['longitude']],
        radius=10,
        popup='<strong>'+row['neighborhood']+'</strong>',
        tooltip='Dica',
        fill=True,
        #fill_color='#428bca'
        fill_color=colors[row['neighborhood']]
    ).add_to(m)
```

```
# Apresenta o mapa
m
```

3.3 Custom Icon

https://www.w3schools.com/icons/default.asp

https://fontawesome.com/v4.7.0/icons/ (não funciona o 5.0) https://stackoverflow.com/questions/58607693/how-to-use-folium-icon-with-fontawesome

```
pointIcon_url = 'http://maps.google.com/mapfiles/kml/shapes/shaded_dot.png'
icon = folium.features.CustomIcon(pointIcon_url, icon_size=(15, 15))
icon
```

3.4 Vegas

O *folium* tem o vegas https://vega.github.io/vega/ como *default*.

```
# Cria o mapa
m = folium.Map(
    location=[-23.9619271,-46.3427499],
    zoom_start=12
)

# Importa bibliotecas e lê o json
vis = os.path.join('data', 'vis.json')

# Adiciona as diferentes empresas com gráficos no popup
for index, row in df.iterrows():
    if row['neighborhood'] in colors.keys():
        folium.Marker(
            location=[row['latitude'], row['longitude']],
            popup=folium.Popup(max_width=450).add_child(folium.Vega(json.
 ↪load(open(vis)), width=450, height=250))
).add_to(m)

# Display Map
m
```

3.5 Geojson

É possível também inserir desenhos em formato ***GeoJson***, o que abre grandes possibilidades. Para rabiscos aleatórios, anotações etc, é possivel criar o arquivo usando http://geojson.io.

```
# Cria o mapa
m = folium.Map(
    location=[-23.9619271,-46.3427499],
    zoom_start=12,
```

```
)

# Importa bibliotecas e lê o json
shp = os.path.join('data', 'trajetos.json')

# Adiciona as diferentes empresas com gráficos no popup
a = folium.GeoJson(shp, name='Trajetos')
folium.GeoJson(shp, name='Trajetos').add_to(m)

# Apresenta o mapa
m
```

3.6 Geojson in URL

```
# Map Object
m = folium.Map(location=(0,0),
               zoom_start=2)

# Data: Link to geojson
json_url = 'https://raw.githubusercontent.com/datasets/
 ↪geo-boundaries-world-110m/master/countries.geojson'

# Adiciona GeoJson
a = folium.GeoJson(shp, name='Trajetos').add_to(m)

# Folium Object
folium.Choropleth(
    geo_data=json_url,
    fill_color='YlGn',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Limites').add_to(m)

# Ordena Layers
m.keep_in_front(a)

# Display Map
m
```

3.7 GeoDataFrame

```
# Cria geopandas Object
gdf = gpd.read_file(os.path.join('data', 'Piracicaba.shp'))

# Transform Coordenate and Format
gjson = gdf.to_crs(epsg='4326').to_json()
```

```
# Cria mapa
m = folium.Map([-15.783333, -47.866667],
               min_zoom=9,
               tiles='cartodbpositron')

shp = folium.features.GeoJson(gjson)
m.add_child(shp)
m.fit_bounds(m.get_bounds())

# Plota Mapa
m
```

3.8 WMS

```
m = folium.Map(location=[28, -81], zoom_start=6)

folium.raster_layers.WmsTileLayer(
    url='http://mesonet.agron.iastate.edu/cgi-bin/wms/nexrad/n0r.cgi',
    name='test',
    fmt='image/png',
    layers='nexrad-n0r-900913',
    attr=u'Weather data © 2012 IEM Nexrad',
    transparent=True,
    overlay=True,
    control=True,
).add_to(m)

m
```

```
m = folium.Map(location=[-22, -55], zoom_start=6)

folium.raster_layers.WmsTileLayer(
    url='http://datageo.ambiente.sp.gov.br/geoserver/ows?SERVICE=WMS&',
    name='test',
    fmt='image/png',
    layers='vwm_car_propriedades_publico_ugrhi_5_pol',
    attr=u'Weather data © 2012 IEM Nexrad',
    transparent=True,
    overlay=True,
    control=True,
).add_to(m)

#m

from folium.plugins import MousePosition
```

```
#m = folium.Map()
MousePosition().add_to(m)
m
```

3.9 *Join* e Categorias

```
# Data
states = os.path.join('data', 'us-states.json')
unemployement_data = os.path.join('data', 'us_unemployment.csv')
df = pd.read_csv(unemployement_data)

#
print(df.head())

# Map Object
m = folium.Map(location=[48, -102],
               zoom_start=3)

# Folium Object
folium.Choropleth(
    geo_data=states,
    name='States',
    data=df,
    columns=['State', 'Unemployment'],
    key_on='feature.id',                    # Campo, no geojson, que tem o␣
 ↪identificador
    fill_color='YlGn',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Unemployment Rate %'
).add_to(m)

# Add Features
folium.LayerControl().add_to(m)

# Display Map
m
```

```
#json.load(open(states))
```

4 Basemap

4.1 Por nome

O mapa pode ter diferentes *basemaps*, que são, na essência, o mapa de fundo renderizado em *titles*. O *folium* utiliza, por *default*, o basemap do *OpenStreetMap*, contudo existe a possibilidade

de adicionar outros serviços, conforme se vê abaixo.

- https://github.com/nextgis/quickmapservices_contrib/tree/master/data_sources
- https://xyz.michelstuyts.be/
- https://www.trailnotes.org/FetchMap/TileServeSource.html

```
folium.Map(location=[-23.9619271,-46.3427499],
           #tiles='Mapbox Bright',
           #tiles='Mapbox Control Room',
           #tiles='Stamen Toner',
           #tiles='Stamen Terrain',
           #tiles='OpenStreetMap',
           zoom_start=12
          )
```

Um outro jeito de inserir *basemaps* é utilizado o MapBox, onde é possível customizar um *basemap* personalizado, bem como utilizar outros *basemaps* pré-existentes, incluindo imagens de satélite de alta resolução, etc.

Para melhor utilização, com a possiblidade de disponibilizar códigos, é necessário estudar a melhor maneira de ocultar a *API key*. Um início: - http://www.blacktechdiva.com/hide-api-keys/ - https://www.quora.com/How-do-you-hide-your-API-customer-key-token-when-youre-pushing-code-to-Github

```
#folium.Map(location=[-23.9619271,-46.3427499],
#           tiles='Mapbox',
#           API_key='your.API.key',
#           zoom_start=12
#          )
```

4.2 Por endereço do title (x, y, z)

Por fim, é possivel ainda inserir *basemaps* personalizados, disponibilizados em algum servidor.

```
folium.Map(location=[-23.9619271,-46.3427499],
           zoom_start=3,
           #tiles='http://{s}.tile.osm.org/{z}/{x}/{y}.png',
           tiles='http://prettymaps.stamen.com/201008/tiles/isola/{z}/000/{x}/
 ↪000/{y}.png',
           attr=' '
          )
```

```
folium.raster_layers.TileLayer(
    tiles='http://{s}.google.com/vt/lyrs=m&x={x}&y={y}&z={z}',
    attr='google',
    name='google street view',
    max_zoom=20,
    subdomains=['mt0', 'mt1', 'mt2', 'mt3'],
```

```
    overlay=False,
    control=True,
).add_to(m)

m
```

```
def create_tiles_folium(tile_service=2, location=[-23.9619271, -46.3427499],␣
↪zoom_start=10):
    """
    Function to create map using tiles... a list of them
    - https://xyz.michelstuyts.be/
    :param tile_service:
    :param location:
    :param zoom_start:
    :return:
    """
    # Import Packages
    import pandas as pd
    import folium

    # Read table with all tiles servers
    #tiles_services = pd.read_csv('data/tiles.csv', index_col=0)
    df = pd.read_csv('https://raw.githubusercontent.com/michelmetran/
↪package_folium/master/data/tiles.csv', index_col=0)
    #print(df)
    #Mais serviços! https://leaflet-extras.github.io/leaflet-providers/preview/

    # Create reference to attribution
    ref = ('<a href="' +
           df.loc[tile_service, 'attribution'] +
           '" target="blank">' +
           df.loc[tile_service, 'name'] +
           '</a>')

    m = folium.Map(location=location,
                   zoom_start=zoom_start,
                   tiles=[],
                  )

    folium.TileLayer(tiles=df.loc[tile_service, 'link'],
                     attr=ref,
                     name=df.loc[tile_service, 'name'],
                    ).add_to(m)

    folium.LayerControl().add_to(m)

    return m
```

```
create_tiles_folium(4)
```

4.3 Multiplos

Ainda, é possível criar um objeto que não tenha nenhum mapa por padrão. Daí, em seguinda, se inserem diversos mapas.

```
m = folium.Map(location=[-23.9619271,-46.3427499],
               zoom_start=12,
               tiles=None,)

folium.TileLayer('openstreetmap').add_to(m)
folium.TileLayer('stamentoner').add_to(m)
folium.TileLayer('stamenTerrain').add_to(m)
folium.TileLayer('stamenwatercolor', name='Qualquer coisa').add_to(m)
folium.TileLayer('cartodbpositron').add_to(m)
folium.TileLayer('cartodbdark_matter').add_to(m)
folium.LayerControl().add_to(m)

m
```

```
def create_map_multitiles(location=[-23.9619271, -46.3427499], zoom_start=10):
    """
    :param tile_service:
    :param location:
    :param zoom_start:
    :return:
    """
    # Import Packages
    import pandas as pd
    import folium

    # Create Maps
    m = folium.Map(location=location,
                   zoom_start=zoom_start,
                   tiles=None,
                   )

    # Read table with all tiles servers
    #df = pd.read_csv('data/tiles.csv', index_col=0)
    df = pd.read_csv('https://raw.githubusercontent.com/michelmetran/
↪package_folium/master/data/tiles.csv', index_col=0)
    #print(df)

    # Filter some tiles
    #df = df[2:4]
```

```
    #df = df.loc[(df['name'] == 'ESRI Satelite') | (df['name'] == 'ESRI␣
↪Street')]
    df = df[df['name'].str.startswith(('ESRI'))]

    # For
    for index, row in df.iterrows():
        # Create reference to attribution
        ref = ('<a href="' +
               row['attribution'] +
               '" target="blank">' +
               row['name'] +
               '</a>')

        # Create multiples tiles layers
        folium.TileLayer(tiles=row['link'],
                         attr=ref,
                         name=row['name'],
                         ).add_to(m)

    #  Add Legend
    folium.LayerControl().add_to(m)

    # Results
    return m
```

```
m = create_map_multitiles(location=[-23.9619271, -46.3427499], zoom_start=10)
#folium.LayerControl('topright', collapsed=False).add_to(m)
m.save('maps/map.html')
m
```

5 Outros elementos do *WebMap*

5.1 Logo

```
import base64
import folium
import requests
from folium.plugins import FloatImage

# Cria o mapa
m = folium.Map(location=[-20,-50])

# Dados do Logo
#src = 'https://raw.githubusercontent.com/michelmetran/michelmetran.github.io/
↪master/img/mm.png'
src = 'https://avatars3.githubusercontent.com/u/10374538?v=4&s=60'
```

```
#src = 'https://i.imgur.com/XRdd9RL.png'
site = 'https://michelmetran.github.io'
alt = 'MichelMetran'
size = '60px'

# Works, but huge!
FloatImage(src, bottom=1.3, left=1).add_to(m)

# HTML
html = "<a href={}><img alt={} src={} width={}></a>".format(site, alt, src,␣
 ↪size)
html = "<img src={}>".format(src)
html = '<img src=https://michelmetran.github.io/img/mm.png width=40px>'

# Not Work
#FloatImage(html, bottom=40, left=10).add_to(m)

# Imagem Encode e Decode
#png = base64.b64encode(requests.get(src).content).decode()
#png = '<img src="{}" width=90px>'.format(png)
#FloatImage(png, bottom=40, left=1).add_to(m)

# Print HTML
print(html)

m
```

```
from PIL import Image
import requests
from io import BytesIO

response = requests.get(src)
img = Image.open(BytesIO(response.content))

basewidth = 70
wpercent = (basewidth/float(img.size[0]))
hsize = int((float(img.size[1])*float(wpercent)))
img = img.resize((basewidth,hsize), Image.ANTIALIAS)

type(img)
```

5.2 Adiciona legenda

```
folium.LayerControl('topright', collapsed=False).add_to(m)
m
```

5.3 A pontos *on-the-fly*

```
m.add_child(folium.ClickForMarker(popup='Waypoint'))
m
```

5.4 Adiciona coordenadas no clique

```
m.add_child(folium.LatLngPopup())
m
```

5.5 Localização Mouse

```
from folium.plugins import MousePosition

m = folium.Map()

#MousePosition().add_to(m)

formatter = "function(num) {return L.Util.formatNum(num, 3) + ' º';};"

MousePosition(
    position='topright',
    separator=' | ',
    empty_string='NaN',
    lng_first=True,
    num_digits=20,
    prefix='Coordinates:',
    lat_formatter=formatter,
    lng_formatter=formatter,
).add_to(m)

m
```

5.6 Fit map

```
m.fit_bounds(m.get_bounds())
m
```

5.7 Full Screen

```
# Cria o mapa
m = folium.Map(
    location=[-23.9619271,-46.3427499],
    zoom_start=12
)
```

```
# Maximiza/Minimiza o mapa
from folium import plugins
plugins.Fullscreen(
    position='topleft',
    title='Clique para Maximizar',
    title_cancel='Mininizar',
    force_separate_button=True).add_to(m)

m
```

5.8 Medição

```
# Cria o mapa
m = folium.Map(
    location=[-23.9619271,-46.3427499],
    zoom_start=12
)

# Adiciona ferramenta de medição
from folium.plugins import MeasureControl
m.add_child(MeasureControl())

m
```

6 Mapas Complexos

6.1 Heat Map

```
# HeatMap https://minerandodados.com.br/
 ↪visualizando-mapas-interativos-com-python/
from folium.plugins import HeatMap

# Cria o mapa
m = folium.Map(
    location=[-23.9619271,-46.3427499],
    zoom_start=12
)

# Lendo e filtrando dados
df = pd.read_csv('data/empresas.xz')

# HeapMap
hm_wide = HeatMap(list(zip(df['latitude'].values,
                           df['longitude'].values)),
                  min_opacity=0.10,
```

```
                    radius=10,
                    blur=15,
                    max_zoom=1
                   )

m.add_child(hm_wide)

# Apresenta o mapa
m
```

6.2 Dual Map

```
# DualMap accepts the same arguments as Map:
from folium.plugins import DualMap

m = DualMap(location=(0, 0), tiles='cartodbpositron', zoom_start=5)

# Add the same marker to both maps:
folium.Marker((0, 0)).add_to(m)

# The individual maps are attributes called `m1` and `m2`:
folium.Marker((0, 1)).add_to(m.m1)

folium.LayerControl().add_to(m)

m
```

```
m = folium.plugins.DualMap(location=(52.1, 5.1), tiles=None, zoom_start=8)

folium.TileLayer('openstreetmap').add_to(m.m1)
folium.TileLayer('cartodbpositron').add_to(m.m2)

folium.LayerControl(collapsed=False).add_to(m)
m
```

```
m = folium.plugins.DualMap(layout='vertical')
m
```

```
m = folium.plugins.DualMap(location=(52.1, 5.1), tiles='cartodbpositron',␣
 ↪zoom_start=8)

fg_both = folium.FeatureGroup(name='markers_both').add_to(m)
fg_1 = folium.FeatureGroup(name='markers_1').add_to(m.m1)
fg_2 = folium.FeatureGroup(name='markers_2').add_to(m.m2)

icon_red = folium.Icon(color='red')
```

```
folium.Marker((52.0, 5.0), tooltip='both', icon=icon_red).add_to(fg_both)
folium.Marker((52.4, 5.0), tooltip='1').add_to(fg_1)
folium.Marker((52.0, 5.4), tooltip='2').add_to(fg_2)

folium.LayerControl(collapsed=False).add_to(m)
m
```

7 Salvar o mapa em HTML

A grande vantagem é salvar o mapa como um arquivo *.html*, bastante possivel para dar um *embed* em qualquer página. Para salvar o resultado em um dado local, criei uma função que pode contribuir, avaliando se determinadas pastas estão criadas e, em caso negativo, cria as mesmas. Em uma destas pastas que ficará salvo o arquivo *.html* criado

```
# %load '../codes/files/create_folders.py'
def create_folders(path, folders=['data', 'docs', 'maps']):
    """
    :param folders: Name os folders that you want create; E.g.: ['folder1',␣
 ↪'folder2']
    :return: Create directories if not exist
    """
    # Import Packages
    import os
    for folder in folders:
        directory=os.path.join(path, folder)
        try:
            if not os.path.exists(directory):
                os.makedirs(directory)
                print('Directory "', directory, '" created!', sep='')
            else:
                print('Directory "', directory, '" already exists...', sep='')
        except OSError:
            print('Error: Creating directory "', directory, '" fail.', sep='')
```

```
create_folders('')
```

```
m.save('maps/map.html')
```

O mapa em *.html*, que é possivel acessar por usando o ***githack.com***. conforme segue:

8 Referências

- https://www.freecodecamp.org/news/real-world-data-science-project-traffic-accident-analysis-e5a36775ee11/
- Muita coisa interessante em https://www.youtube.com/watch?v=4RnU5qKTfYY
- https://jtemporal.com/folium/

- https://www.kaggle.com/rachan/how-to-folium-for-maps-heatmaps-time-analysis
- Muitos exemplos de ipynb https://nbviewer.jupyter.org/github/python-visualization/folium/tree/master/examples/

```
%%javascript

var kernel = IPython.notebook.kernel;
var body = document.body, attribs = body.attributes;
var command = 'inp = ' + '"' + attribs['data-notebook-name'].value + '"';
kernel.execute(command);
```

```
#inp
```