

Analytics Jumpstart

folium maps and df.iterrows()

Nashville Software School



For today

- folium maps
- `df.iterrows()`



Folium

- Python package built upon the leaflet javascript library
- Create interactive maps
- Build markers
- Easily customized popups

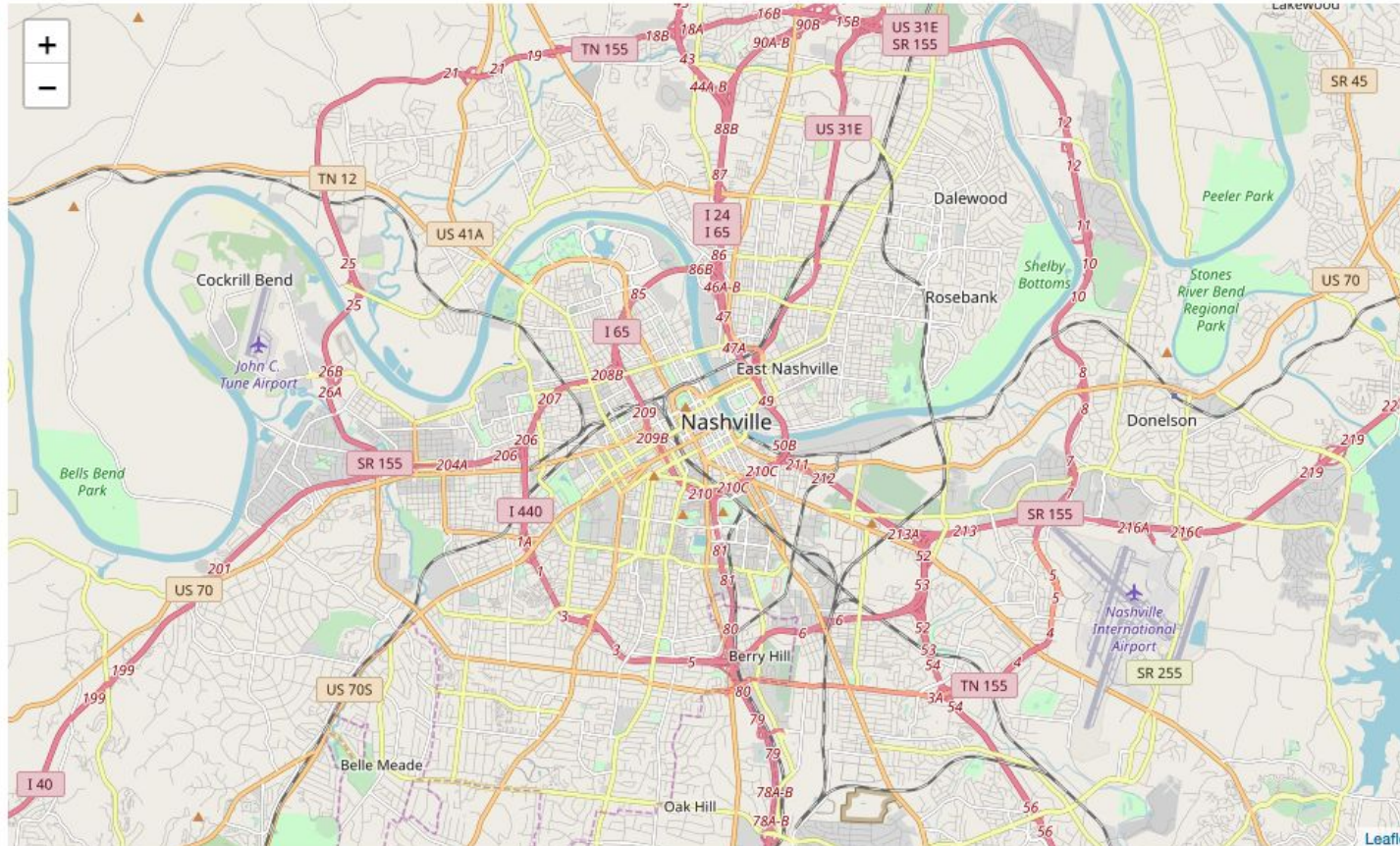


To initialize a map in folium, you need a location + (optionally) a beginning `zoom_level`:

- a folium **location** is a list with latitude first and longitude second
- the **zoom_start** tells folium how close to zoom into the location
- pass **location** and **zoom_start** to the **folium.Map()** constructor
- call the map in jupyter to display it

```
In [68]: nashville = [36.1612, -86.7775]  
nashville_map = folium.Map(location = nashville, zoom_start = 12)  
nashville_map
```

Out[68]:



You will use `iterrows()` to build markers with popups for your folium maps, so let's look at how that works. Here are the first 4 rows of `urban_art`, a data frame of public art in Nashville.

```
In [16]: urban_art.head(4)
```

Out[16]:

	title	last_name	first_name	address	medium	type	desc	lat	lng	loc	geometry	index_right	name
1	[Fourth and Commerce Sculpture]	Walker	Lin	333 Commerce Street, Nashville TN	NaN	Sculpture		36.16234	-86.77774	(36.16234, -86.77774)	POINT (-86.777740000000001 36.16234)	41	Urban Residents
4	A Story of Nashville	Ridley	Greg	615 Church Street, Nashville TN	Hammered copper repousse	Frieze	Inside the Grand Reading Room, this is a serie...	36.16215	-86.78205	(36.16215, -86.78205)	POINT (-86.78205 36.16215)	41	Urban Residents
21	Chet Atkins	Faxon	Russell	Corner of Fifth Avenue North and Union Street,...	Bronze stool and guitar on a granite plynth	Sculpture	A sculpture of a young Chet Atkins seated on a...	36.16466	-86.78102	(36.16466, -86.78102)	POINT (-86.78102 36.16466)	41	Urban Residents
22	Children's Chairs For The Seasons	McGraw	Deloss	615 Church Street, Nashville TN	Mixed Media - wood and paint	Furniture	chairs depicting the four seasons	36.16215	-86.78205	(36.16215, -86.78205)	POINT (-86.78205 36.16215)	41	Urban Residents



Here we use a for-loop with `iterrows()` to iterate through the `urban_art` data frame. We grab each row, print it, then move on to the next row, grab it, print it, until we reach the end of the data frame:

```
In [11]: for row in urban_art.iterrows():  
         print(row)
```

```
(1, title           [Fourth and Commerce Sculpture]  
last_name           Walker  
first_name          Lin  
address             333 Commerce Street, Nashville TN  
medium              NaN  
type                Sculpture  
desc  
lat                 36.1623  
lng                 -86.7777  
loc                 (36.16234, -86.77774)  
geometry            POINT (-86.777740000000001 36.16234)  
index_right         41  
name                Urban Residents  
Name: 1, dtype: object)  
(4, title           A Story of Nashville  
last_name           Ridley  
first_name          Greg  
address             615 Church Street, Nashville TN  
medium              Hammered copper repousse  
type
```



Each data frame row returned with `iterrows()` is a tuple with 2 elements. The first element is the row id and the second element holds all of the values for that row.

```
In [13]: for row in urban_art.iterrows():
          print('first element:', row[0])
          print('second element:', row[1])
          print("-----")
```

```
first element: 1
second element: title          [Fourth and Commerce Sculpture]
last_name          Walker
first_name          Lin
address            333 Commerce Street, Nashville TN
medium             NaN
type               Sculpture
desc
lat                36.1623
lng                -86.7777
loc                (36.16234, -86.77774)
geometry           POINT (-86.777740000000001 36.16234)
index_right        41
name               Urban Residents
Name: 1, dtype: object
-----
first element: 4
second element: title          A Story of Nashville
last_name          Ridley
```



You can drill further down in each data frame row using `iterrows()` by slicing a column by name from the row values. Here we are grabbing and printing the latitude, longitude, and title of each art work in the `urban_art` data frame.

```
In [14]: for row in urban_art.iterrows():
          print('latitude:', row[1]['lat'])
          print('longitude:', row[1]['lng'])
          print('title of work:', row[1]['title'])
          print("-----")

latitude: 36.16234
longitude: -86.77774000000001
title of work: [Fourth and Commerce Sculpture]
-----
latitude: 36.16215
longitude: -86.78205
title of work: A Story of Nashville
-----
latitude: 36.16466
longitude: -86.78102
title of work: Chet Atkins
-----
latitude: 36.16215
longitude: -86.78205
title of work: Children's Chairs For The Seasons
-----
latitude: 36.16215
longitude: -86.78205
title of work: Foliated Scroll
-----
latitude: 36.16298
longitude: -86.78184
title of work: Gone Fishing
-----
latitude: 36.1647
longitude: -86.78043000000001
title of work: Happy Times at The Arcade
-----
latitude: 36.158301
longitude: -86.774955
title of work: Johnny Cash Mural
```



By using `iterrows()` in a for-loop, we create a location from each row's latitude and longitude values and popups from each row's title and description.

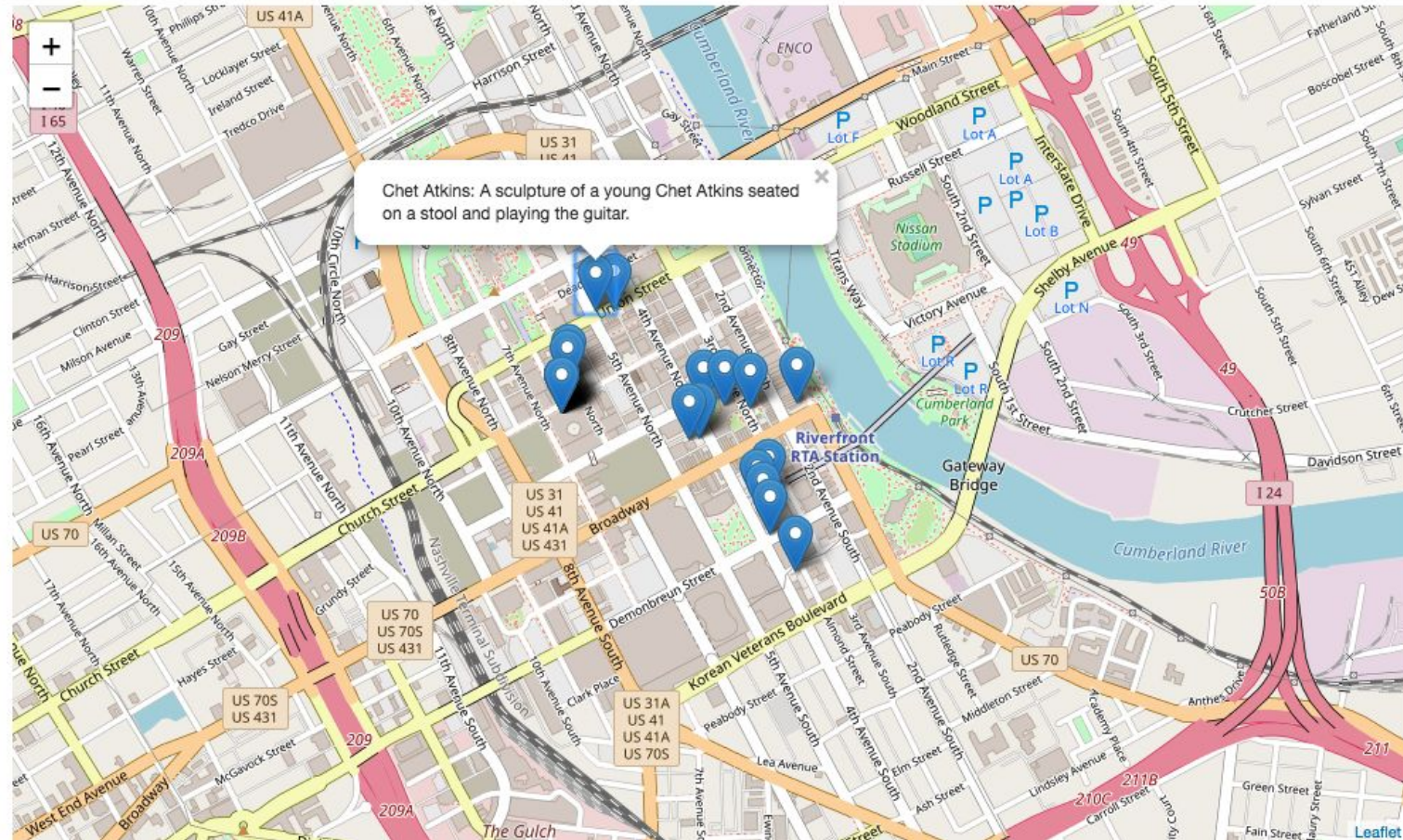
We create a marker each time through the loop by passing the location and the popup to the `folium.Marker()` constructor.

We use the `.add_to()` method to add the marker to the map.

After exiting the loop, we display the map.

```
In [17]: for row in urban_art.iterrows():
         row_values = row[1]
         location = [row_values['lat'], row_values['lng']]
         popup = (str(row_values['title']) + ': ' +
                  str(row_values['desc'])).replace('"', "'")
         marker = folium.Marker(location = location, popup = popup)
         marker.add_to(downtown_map)

display(downtown_map)
```



Questions?

