Основи на Програмирането

Лекция 13

Обработка на географски данни в Python. Задачи за геолокация.

Какво ще научите

- ✓ Въведение
- ✓ Системи за глобално позициониране
- ✓ Представяне на географска информация
- ✓ Примери с библиотеки на Python
- ✓ Географски карти и координати
- ✓ Библиотека GeoPandas
- ✓ Примери за онагледяване на географска информация
- ✓ Геопарсинг с Python
- ✓ Създаване на таблици и визуализиране

Основни понятия

- ✓ **Системи за глобално позициониране - сателитни системи, които ви позволяват да определяте координатите на обекти с точност в сантиметри**
- ✓ **Географски системи - сателити или самолети с камера с висока разделителна способност, които събират геограски данни – снимки, карти**
- ✓ **Географски информационни системи (ГИС) - Софтуерни системи с възможност за въвеждане, контрол, анализ и показване на географски данни. Също таказа въвеждане и управление на данните от тези системи.**

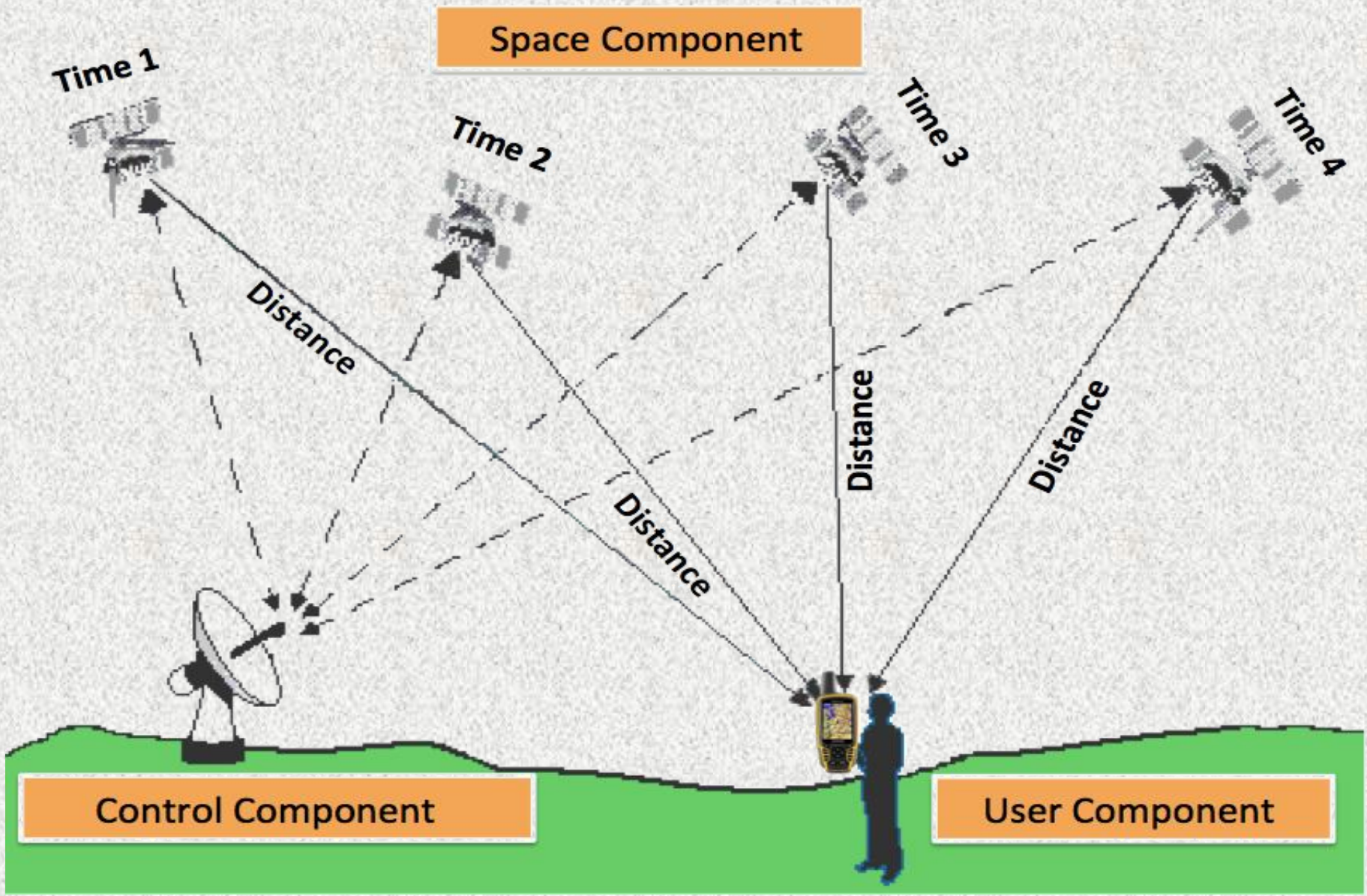
Системи за глобално позициониране

Понятие	Описание
глобално позициониране	За определяне на координатите на всяка точка от земята и точното календарно време.
локално позициониране	Използва се в ограничена територия, ползва по-малко ресурси и е по-лесна за поддръжка.
Географска коорд. система	географска дължина, географска широчина и надморска височина (в метри).
Атомен часовник	Измерва времето. Цезиевия часовник има точност до 1 секунда на 3000 години.
Приемник	изчислява точната позиция (координати).
Наземна станция	Следи сигналите от сателитите, контролира точността на атомните им часовници и орбити на движение, изчислява закъсненията на сигнала.

Определяне координати на точка

В дадената точка приемник улавя сигналите на всички сателити, до които има пряка видимост. Сателитите гарантират видимост за всяка точка от поне 4 различни сателита (чрез 24-30 сателита с подходящи орбити). По времето на излъчване и позицията на сателита се пресмята разстоянието между сателита и точката. Координатите на точката се определят като пресечна точка на 4-те окръжности с центрове - сателитите. Така се получават 4 уравнения с 4 неизвестни - трите координатни на точката, и закъснението на сигналите от сателитите до точката. Чрез решаването на тази система се изчисляват координатите на точката. Точността на този метод е между няколко сантиметра и няколко метра. Приемника получава и точното време.

Определяне координати на точка



Сателити

Всеки сателит има няколко атомни часовника. Така всички сателити са с еднакво време. Това дава услуга точно време за всеки приемник. Всеки сателит излъчва периодично към земята своите координати. Чрез сензори следи какво е състоянието на атмосферата. Чрез система от огледала генерира електроенергия от слънчевите лъчи за захранване системите на сателита. Всеки сателит има период на годност между 15 и 20 години.

Всяка система за глобално пози¬цио¬ниране включва и наземни станции, които следят във всеки момент работата на сате¬ли¬тите, и при нужда извършват корекции в някои от системите им. Те са отговорни и за изчисляване на закъснението на един радиосигнал от сателитите към различните точки от земното кълбо.

Лекция 13

История

Днес има три глобални системи за сателитно позициониране: GPS (американска), ГЛОНАСС (руска) и европейската Галилео. Американската и руската системи се създават в началото на 80-те години от миналия век. Те първоначално се използват само за военни цели. Един южнокорейски самолет през 1983 г. се обърква и лети над територията на днешна Русия (тогава СССР). Той е свален от руската противовъздушна система за защита и загиват много пътници. Този инцидент би бил невъзможен при наличие на система за глобално позициониране в южнокорейския самолет. САЩ решават да предоставят своята система в олекотен вариант за свободно ползване, аналогично решение взима и Русия. Европейският съюз решава да създаде система за свободно ползване с високо качество - Галилео. Тя е значително по-точна от американската и руската система за цивилни нужди. В момента се работи по завършване и на китайска система за позициониране.

Любопитни факти

В авиацията във всеки момент се знае къде точно се намира даден самолет и по какъв маршрут лети. Тази данни се следят от наземните центрове за контрол на полетите с цел избягване на проблеми, катастрофи и други аномалии. Данните за много от самолетите в реално време са налични и на публичния сайт: http://www.flightradar24.com/

Американската система GPS е първата свободно достъпна, и за някои е синоним на система за глобално позициониране. Тя разполага с 24 спътника разположени в 6 различни орбити около земята и 6 резервни. Използва 5 наблюдателни наземни станции разположени по целия свят, и един главен контролен център в Колорадо Спрингс, който контролира работата на всеки спътник. Към края на 2017 г. точността е около 3 метра. Днес е достигната точност под 1 м.

Геотагинг (метаданни)

Всяка снимка, която правите от умен телефон има допълнителна информация, която включва много детайли като ден и час, резолюция и много други. Този тип информация се нарича мета данни и се съхранява съобразно стандарта EXIF заедно със снимката. Част от тези мета данни включват координатите на мястото и този процес се нарича гео-означаване (geotagging). Чрез свободно налични програми можете да откриете тази информация във всяка снимка, аудио или видео, ако е налична.

Използване

Ако искате да сложите снимка върху цифрова карта, трябва последователно:

- да вземете от EXIF мета данните координатите на мястото (GPS position) на снимката
- да намерите тези координати на цифровата карта
- да използвате функциите на цифровата карта за добавяне на снимки (например Добави снимка в Google карти).

Използване на геотагинг

- Когато посочите някаква точка в Google карти, връзката в браузера съдържа и нейните координати
- Използвайте координатите, за да можете точно да посочите всеки един обект върху Google карта Използвайте вградените функции на Google картите за намиране на оптимален маршрут между две точки
- Ако вашата цифрова камера няма гео означаване, вие можете да добавите координатите на всяка точка където сте снимали от вашия телефон или друго крайно GPS устройство.
- Използвайте свободни програми за добавяне на EXIF geotagging метаданни към вашите снимки (отваряте снимката в софтуера и добавяте нейните координати)

Основни понятия

✓ **Географски данни и информация** са дефинирани в стандарта ISO/TC 211 като имащи явна и неявна асоциация с конкретно местоположение на Земята.

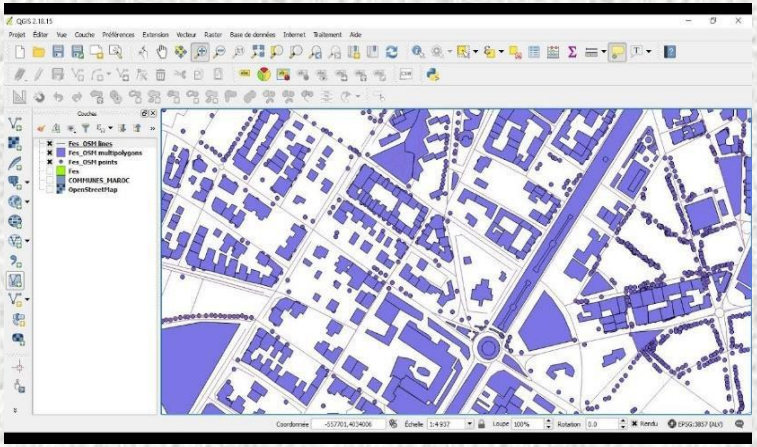
✓ Почти всички данни от официален източник имат компонент свързан с местоположение - https://www.iso.org/committee/54904.html

✓ **ГИС (Географска Информационна Система)** е система проектирана да **определя**, **запазва**, **обработва**, **анализира**, **управлява**, и **представя** пространствени или географски данни. Популярни ГИС са например ArcGIS (ESRI) и QGIS – и двете могат да бъдат използвани чрез езика Python.

Лекция 13

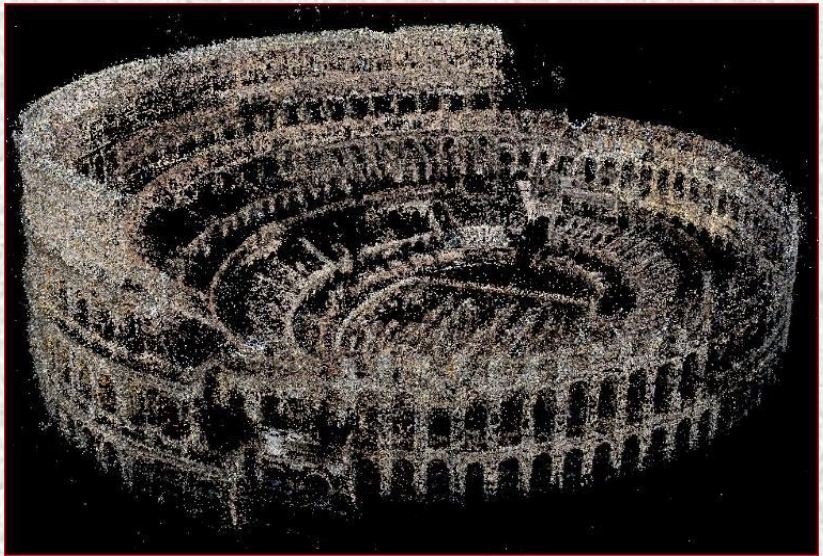
Данни за геолокация

Векторни данни



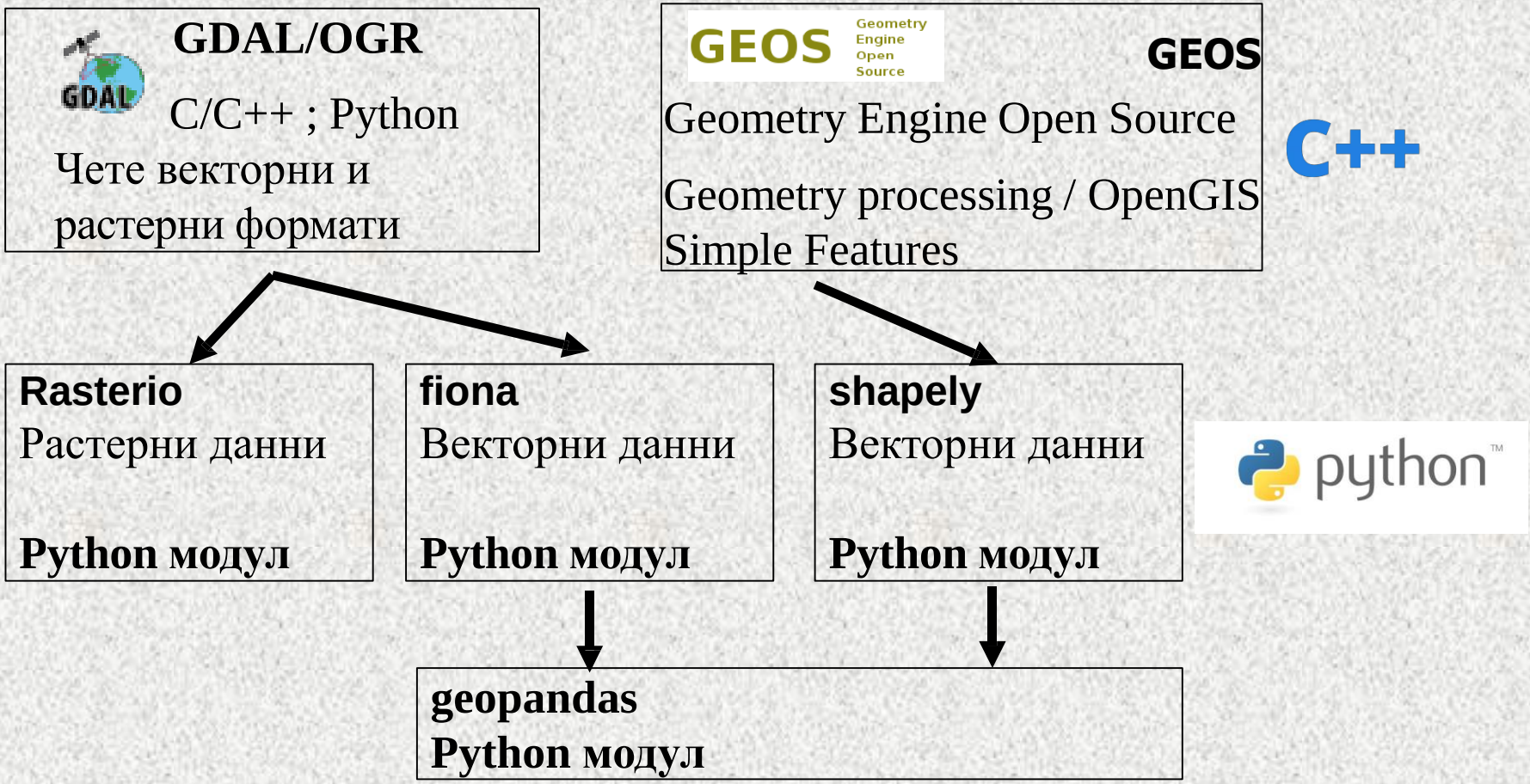
Растерни данни

Точкови облаци



3D-Обекти

Библиотеки и модули с отворен код за векторни и растерни данни



Налични модули за Python

Shapely геометрични обекти https://github.com/Toblerity/Shapely

Fiona векторни данни https://github.com/Toblerity/Fiona

rasterio растерни данни https://github.com/mapbox/rasterio

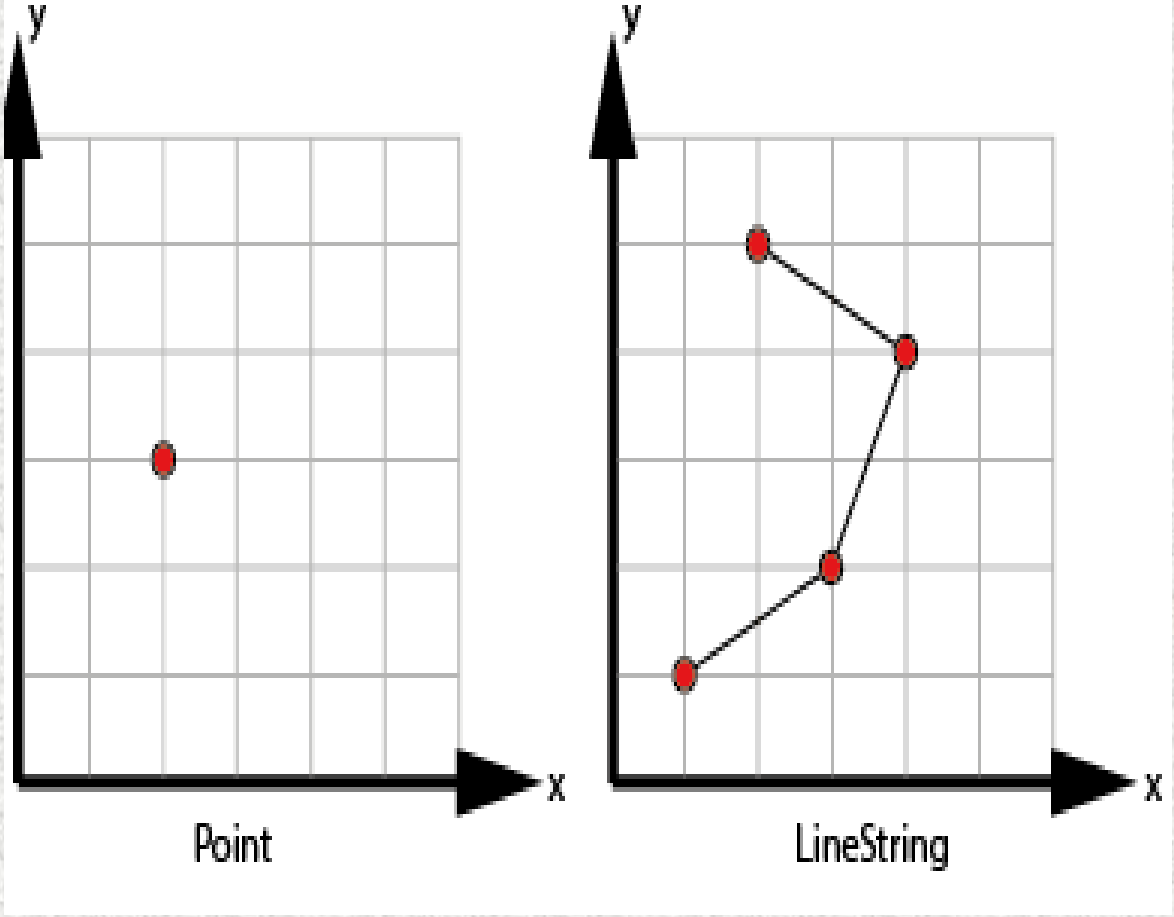
pyproj геолокации https://github.com/jswhit/pyproj

GeoPandas геолокации https://github.com/geopandas/geopandas

Folium създаване на карти https://github.com/python-visualization/folium

ipyleaflet интерактивни карти https://github.com/jupyter-widgets/ipyleaflet

Векторни данни: OpenGIS прости свойства / OCG достъп до тях



В БД за геолокации: (Postgresql/PostGIS), GIS, …

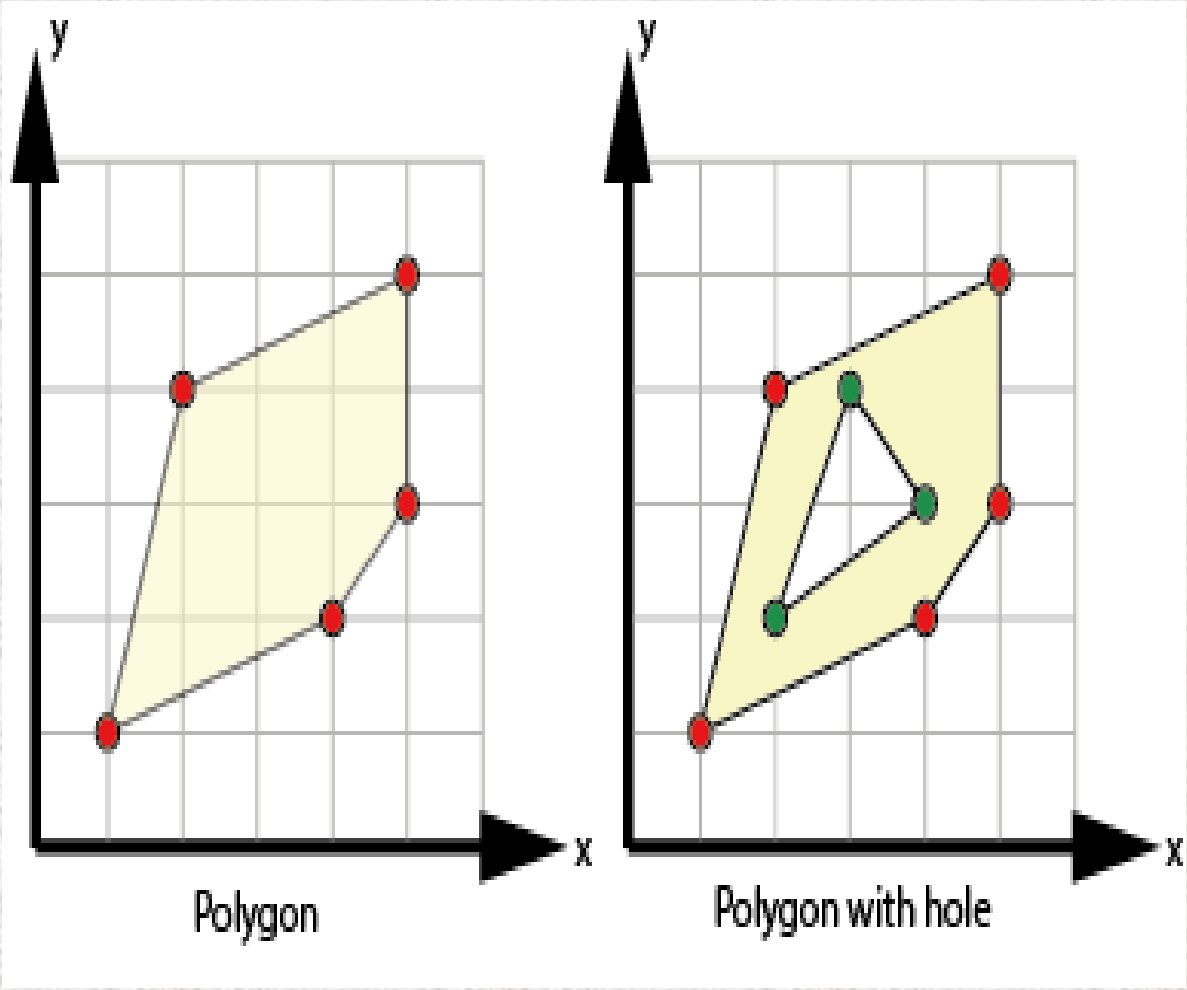
WKT (Well known Text):

- Point, Multipoint
- LineString, MultiLineString

Векторни данни: WKT

WKT (Well known Text):

- Polygon, MultiPolygon
- GeometryCollection
- (TIN, Circle, Curve, Triangle, Surface, PolyhedralSurface, …)



WKT примери

```
POINT (10 20)
MULTIPOLYGON (((30 20, 45 40, 10 40, 30 20)),
   ((15 5, 40 10, 10 20, 5 10, 15 5)))
```

```
from shapely.geometry import Point, Polygon
mypoint = Point(10, 20)
mypoly = Polygon([(30, 10), (40, 40), (20, 35),
(10, 20), (30, 10)])
```

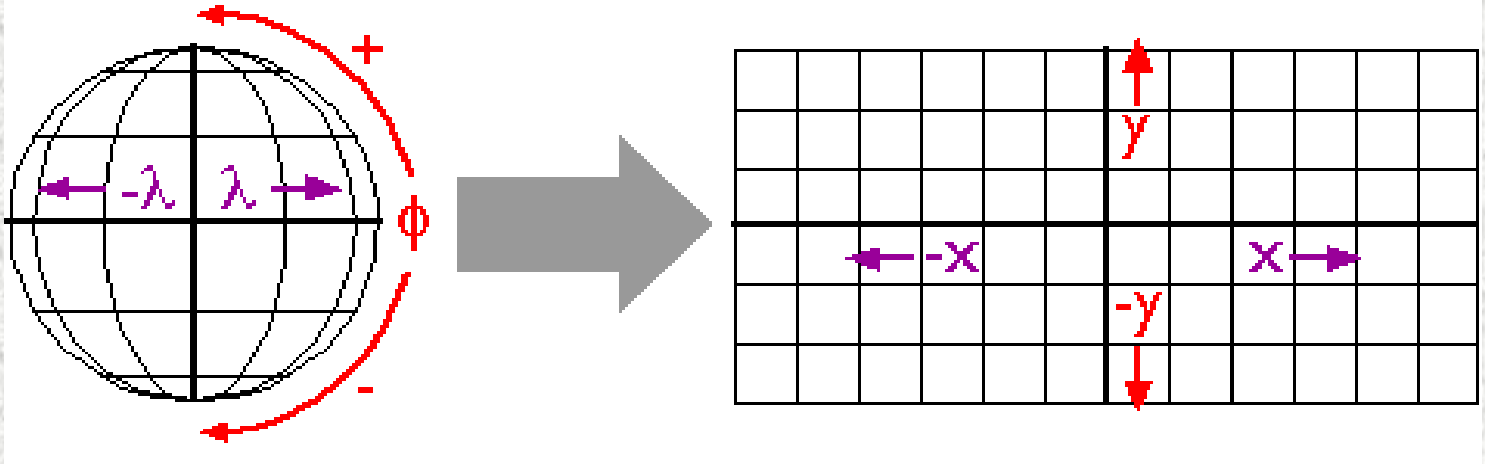
```
import shapely.wkt
myPolygon = shapely.wkt.loads("POLYGON((10 10,
10.1 50, -10 60, 10 80, 80 80, 80 10, 10 10))")
```

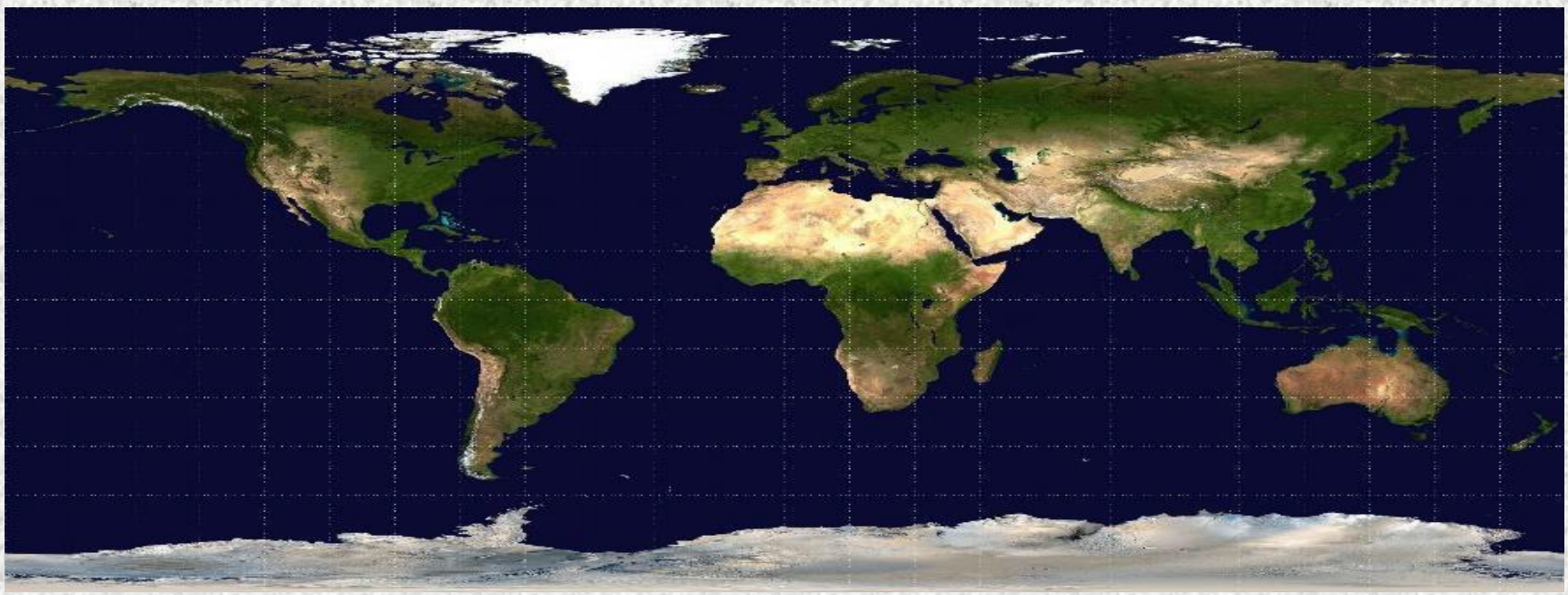
Векторни данни с GeoJSON

Същият принцип, но използва формат JSON

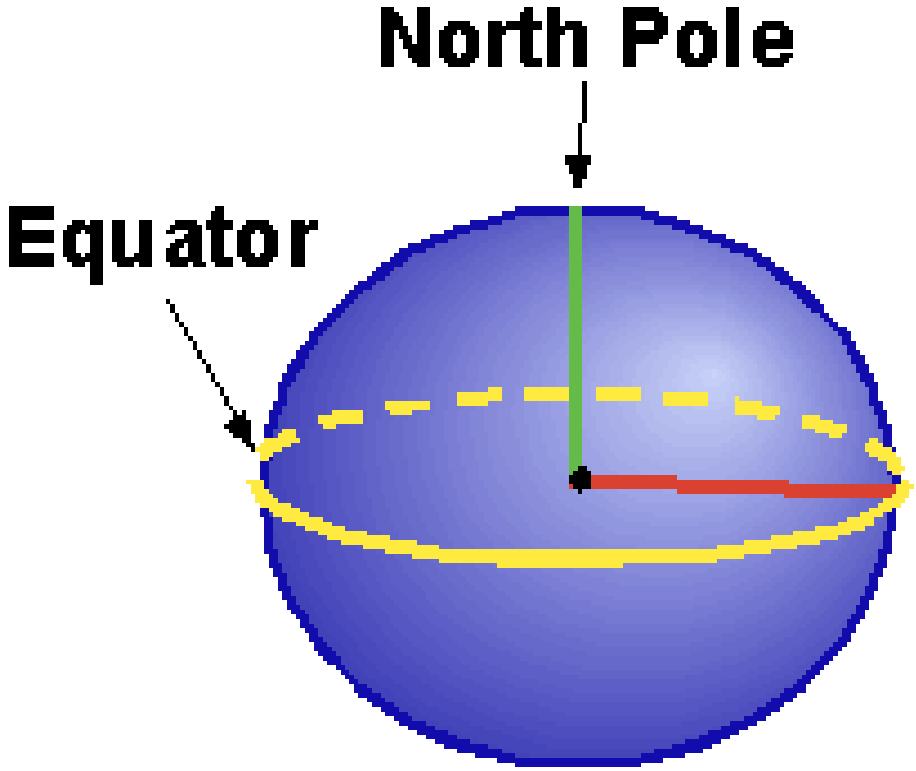
```
{  "type": "Polygon",
   "coordinates":[[[22.731099,54.327537],[22.65 …
}
```

Координати

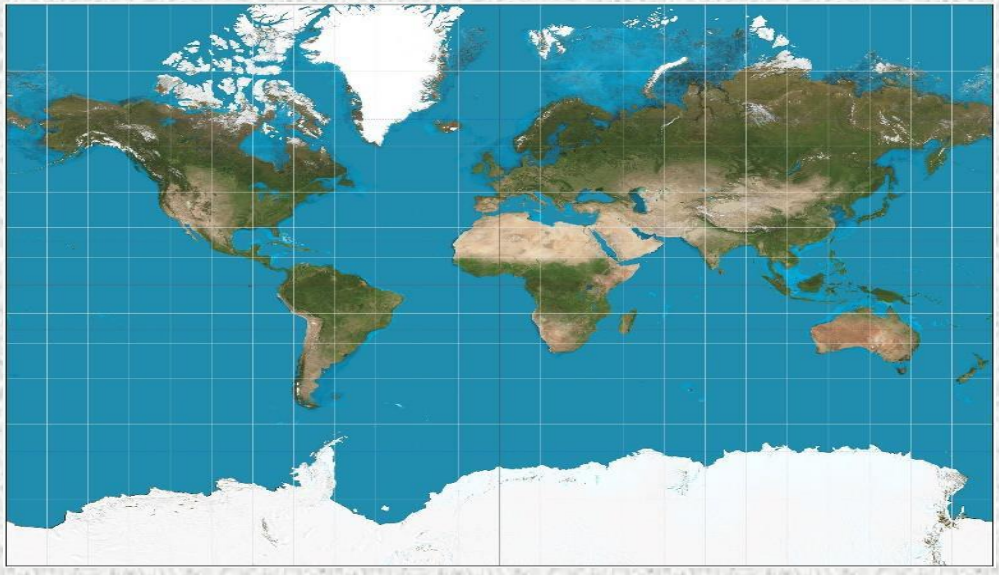




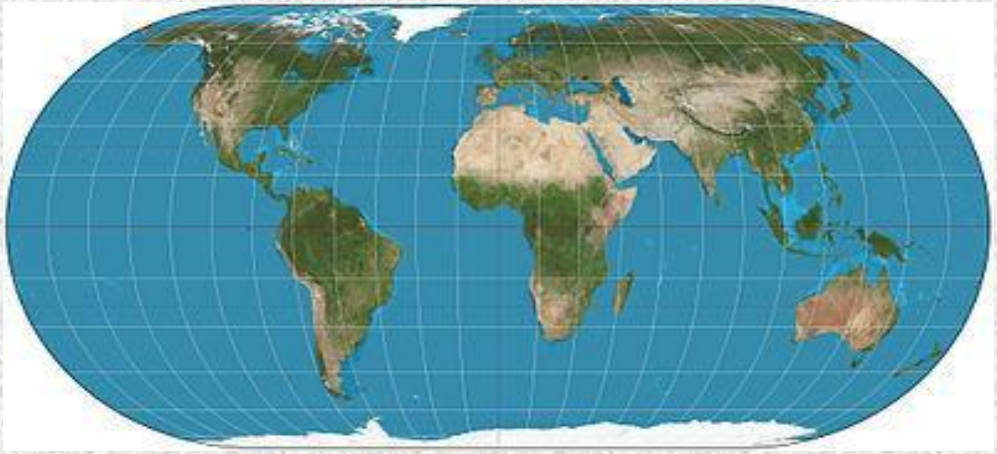
Земята не е кълбо, но принципа е подобен



Примери за проекции на карти



Mercator



Eckert IV

Cassini

Географски координатни системи

Модулът **pyproj** се използва за преобразуване от един формат на координатите в друг. Примери:

Geocentric Cartesian System: Това е стандартната координатна система с три координати: географска дължина, географска ширина, и височина.

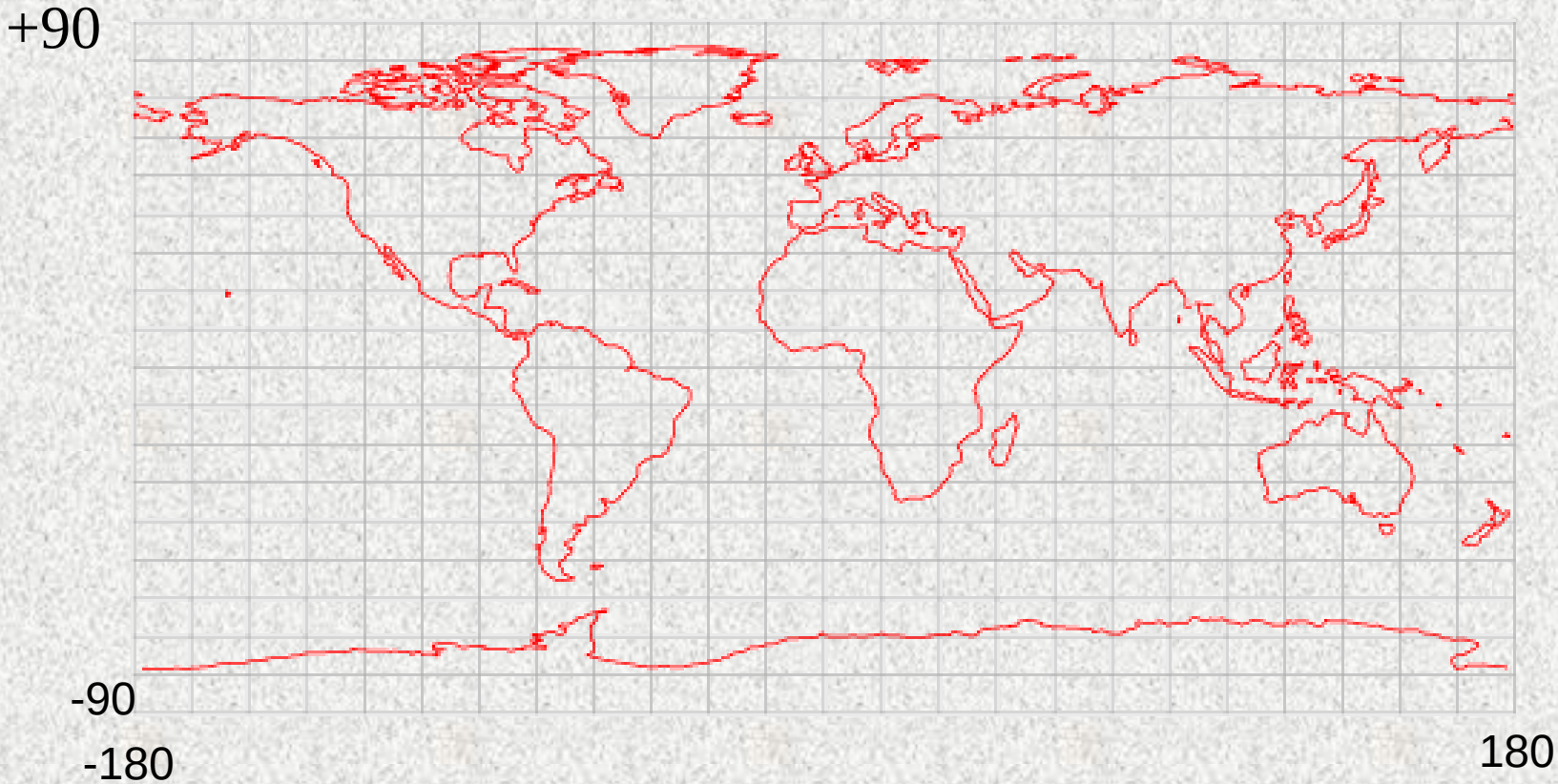
Projected: Взимаме някакво подмножество на земното кълбо, моделираме го като равнина и използваме подходящи координати за това подмножество.

В тази лекция за простота ще ползваме само стандартните географски координати – дължина и широчина (WGS84, EPSG: 4326).

Пример: Швейцария, Базел

Географските му координати: [47.562608, 7.599175]

Това са [дължина - latitude, ширина - longitude]



Анализ на данни с shapely

Пример: дали точка е част от полигон

```
from shapely.geometry import Point

Bassel = Point([47.562608, 7.599175])
Bassel.wkt

'POINT (47.562608  7.599175)'

Bassel.within(switzerland)
False
```

Анализ на данни с shapely

Пример: дали точка е част от полигон

```
from shapely.geometry import Point

Bassel = Point([7.599175, 47.562608])
Bassel.wkt

'POINT (7.599175 47.562608)'

Bassel.within(switzerland)
True
```

Други векторни формати

В практиката са известни много други векторни формати за географски координати. Някои от по-известните са «**Shapefiles**» и «**GeoPackage**».

Ако искате да поддържате повече формати, може да използвате модула Fiona.

Вътрешно в програмата е достатъчно да се поддържат Points, Polygons, …

Анализ с GeoPandas

GeoPandas е Pandas с добавена геометрична колона.
GeoPandas поддържа **пространствени заявки**.

cities5k.csv съдържа всички населени места с население поне 5000 души. [Източник: geonames.org]

A1

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	3039163	Sant Julià de Lòria	Sant Julia de Loria	San Julia,San Julià,Sant Julia de Loria,Sant Julià de	42.46372	1.49129	P	PPLA	AD		6				8022		921	Europe/Andorra	23.11.2013
2	3039678	Ordino	Ordino	Ordino,ao er di nuo,orudino jiao qu,Ордино,オル	42.55623	1.53319	P	PPLA	AD		5				3066		1296	Europe/Andorra	11.12.2009
3	3040051	les Escaldes	les Escaldes	Ehskal'des-Ehndzhordani,Escaldes,Escaldes-Engor	42.50729	1.53414	P	PPLA	AD		8				15853		1033	Europe/Andorra	15.10.2008
4	3040132	la Massana	la Massana	La Macana,La Massana,La Maçana,La-Massana,la	42.54499	1.51483	P	PPLA	AD		4				7211		1245	Europe/Andorra	15.10.2008
5	3040686	Encamp	Encamp	Ehnkam,Encamp,en kan pu,enkanpu jiao qu,Энкам	42.53474	1.58014	P	PPLA	AD		3				11223		1257	Europe/Andorra	13.04.2012
6	3041204	Canillo	Canillo	Canillo,Kanil'o,ka ni e,kaniryo jiao qu,Канильо,カ	42.5676	1.59756	P	PPLA	AD		2				3292		1561	Europe/Andorra	24.12.2012
7	3041563	Andorra la Vella	Andorra la Vella	ALV,Ando-la-Vyey,Andora,Andora la Vela,Andora	42.50779	1.52109	P	PPLC	AD		7				20430		1037	Europe/Andorra	30.05.2010
8	290594	Umm al Qaywayn	Umm al Qaywayn	Oumm al Qaiwain,Oumm al Qaïwaïn,Um al Kawai	25.56473	55.55517	P	PPLA	AE		7				44411		2	Asia/Dubai	07.10.2014
9	291074	Ras al-Khaimah	Ras al-Khaimah	Julfa,Khaimah,RKT,Ra's al Khaymah,Ra's al-Chaim	25.78953	55.9432	P	PPLA	AE		5				115949		2	Asia/Dubai	05.12.2015
10	291279	Muzayri‘	Muzayri`	Mezaira'a,Mezaira’a,Mizeir`ah,Mizeir‘ah,Mozayri`	23.14355	53.7881	P	PPL	AE		1				10000		123	Asia/Dubai	24.10.2013
11	291696	Khawr Fakkān	Khawr Fakkan	Fakkan,Fakkān,Khawr Fakkan,Khawr Fakkān,Khaw	25.33132	56.34199	P	PPL	AE		6				33575		20	Asia/Dubai	25.10.2013
12	292223	Dubai	Dubai	DXB,Dabei,Dibai,Dibay,Doubayi,Dubae,Dubai,Duba	25.0657	55.17128	P	PPLA	AE		3				1137347		3	Asia/Dubai	02.12.2014
13	292231	Dibba Al-Fujairah	Dibba Al-Fujairah	Al-Fujairah,BYB,Dibba Al-Fujairah,dba alfjyrt,الفجيرة	25.59246	56.26176	P	PPL	AE		4				30000		16	Asia/Dubai	12.08.2014
14	292239	Dibba Al-Hisn	Dibba Al-Hisn	BYB,Daba,Daba al-Hisn,Dabā,Dabā al-Ḥiṣn,Diba,Di	25.61955	56.27291	P	PPL	AE		4				26395		4	Asia/Dubai	21.04.2014
15	292672	Sharjah	Sharjah	Al Sharjah,Ash 'Mariqah,Ash Shariqa,Ash Shariqah,	25.33737	55.41206	P	PPLA	AE		6				543733		6	Asia/Dubai	05.03.2013
16	292688	Ar Ruways	Ar Ruways	Ar Ru'ays,Ar Ruways,Ar Ru’ays,Ar-Ruvais,Ruwais,A	24.11028	52.73056	P	PPL	AE	AE	1				16000		16	Asia/Dubai	03.11.2012
17	292878	Al Fujayrah	Al Fujayrah	Al Fujayrah,Al-Fudjayra,Al-Fujayrah' emiraat,FJR,F	25.11641	56.34141	P	PPLA	AE		4				62415		15	Asia/Dubai	18.12.2016
18	292913	Al Ain	Al Ain	AAN,Ainas,Al Ain,Al Ajn,Al Ayn,Al `Ayn,Al Ɛayn,Al ‘A	24.19167	55.76056	P	PPL	AE		1				408733		275	Asia/Dubai	17.03.2013
19	292932	Ajman	Ajman	Ajman,Al Ajman,OAJ,Ujman,ʻjman,عجمان	25.41111	55.43504	P	PPLA	AE		2				226172		4	Asia/Dubai	24.03.2013

Таблица с географски координати

```
import pandas as pd

df = pd.read_csv('data/cities5k.csv', encoding="utf-8", sep=",", header=None, low_memory=False)
df.head(3)
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	3039163	Sant Julià de Lòria	Sant Julia de Loria	San Julia,San Julià,Sant Julia de Loria,Sant J...	42.46372	1.49129	P	PPLA	AD	NaN	6	NaN	NaN	NaN	8022.0
1	3039678	Ordino	Ordino	Ordino,ao er di nuo,orudino jiao qu,Ордино,オルデ...	42.55623	1.53319	P	PPLA	AD	NaN	5	NaN	NaN	NaN	3066.0
2	3040051	les Escaldes	les Escaldes	Ehskal'des-Ehndzhordani,Escaldes,Escaldes-Engo...	42.50729	1.53414	P	PPLA	AD	NaN	8	NaN	NaN	NaN	15853.0

Опростена таблица

```
df2 = df[[1,4,5,14]]
df2.columns = ["name", "lat", "lng", "population"]
df2.head()
```

	name	lat	lng	population
0	Sant Julià de Lòria	42.46372	1.49129	8022.0
1	Ordino	42.55623	1.53319	3066.0
2	les Escaldes	42.50729	1.53414	15853.0
3	la Massana	42.54499	1.51483	7211.0
4	Encamp	42.53474	1.58014	11223.0

Заявка

```
df2.query("name == 'Basel'")
```

	name	lat	lng	type	population
5720	Basel	47.55839	7.57327	PPLA	164488.0

Създаване на GeoPandas таблица

Създаваме обект от **shapely точка** за координатите и наричаме колоната за този обект **geometry**…

```
import geopandas as gpd
from shapely.geometry import Point

geometry = [Point(pos) for pos in zip(df2['lng'], df2['lat'])]
gdf = gpd.GeoDataFrame(df2, geometry=geometry)

gdf.head()
```

	name	lat	lng	population	geometry
0	Sant Julià de Lòria	42.46372	1.49129	8022.0	POINT (1.49129 42.46372)
1	Ordino	42.55623	1.53319	3066.0	POINT (1.53319 42.55623)
2	les Escaldes	42.50729	1.53414	15853.0	POINT (1.53414 42.50729)
3	la Massana	42.54499	1.51483	7211.0	POINT (1.51483 42.54499000000001)
4	Encamp	42.53474	1.58014	11223.0	POINT (1.58014 42.53474)

Актуални към момента данни

Данни за земетресения от USGS:
https://earthquake.usgs.gov/earthquakes/feed/v1.0/geojson.php

```
import requests

url = "https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/2.5_week.geojson"
#url = "https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/significant_month.geojson"
#url = "https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/2.5_month.geojson"

data = requests.get(url)
file = open("earthquakes.geojson","wb")
file.write(data.content)
file.close()
```

Зареждане в Pandas

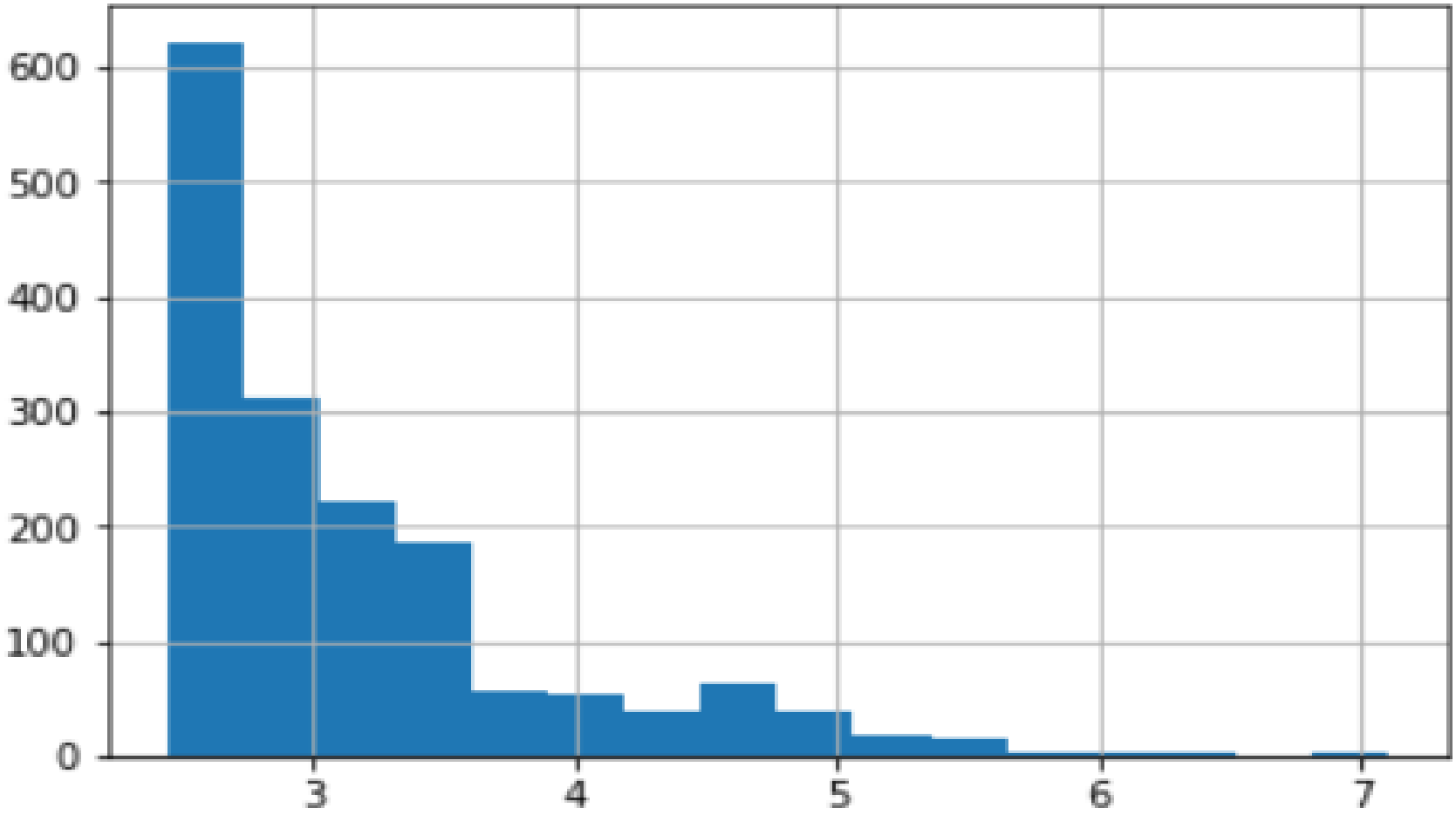
```
import geopandas as gpd

eq_gdf = gpd.read_file("earthquakes.geojson")
eq_gdf.head()
```

```
eq = eq_gdf[["time","mag", "place","geometry"]].copy()
eq.head()
```

	time	mag	place	geometry
0	1562761736970	4.13	8km E of Coso Junction, CA	POINT Z (-117.8568344 36.0499992 1.42)
1	1562760969030	3.70	257km SE of Kodiak, Alaska	POINT Z (-149.3869 56.1625 21.55)
2	1562760618010	2.55	14km ENE of Ridgecrest, CA	POINT Z (-117.5215 35.6498333 2.82)
3	1562760034750	2.74	19km ESE of Little Lake, CA	POINT Z (-117.7089996 35.8751678 3.79)
4	1562760005100	3.81	19km ESE of Little Lake, CA	POINT Z (-117.7068329 35.8764992 4.43)

```
eq.mag.hist(bins=16);
```



Уеднаквяване на дата и час

```
from datetime import datetime, timezone

data = []
for row in range(0,len(eq)):
    time = eq.iloc[row].time
    t = str(datetime.fromtimestamp(time/1000.0, timezone.utc))
    data.append(t)

eq["time_utc"] = data
eq = eq.drop(['time'], axis=1)
eq.head()
```

	mag	place	geometry	time_utc
0	4.13	8km E of Coso Junction, CA	POINT Z (-117.8568344 36.0499992 1.42)	2019-07-10 12:28:56.970000+00:00
1	3.70	257km SE of Kodiak, Alaska	POINT Z (-149.3869 56.1625 21.55)	2019-07-10 12:16:09.030000+00:00
2	2.55	14km ENE of Ridgecrest, CA	POINT Z (-117.5215 35.6498333 2.82)	2019-07-10 12:10:18.010000+00:00
3	2.74	19km ESE of Little Lake, CA	POINT Z (-117.7089996 35.8751678 3.79)	2019-07-10 12:00:34.750000+00:00
4	3.81	19km ESE of Little Lake, CA	POINT Z (-117.7068329 35.8764992 4.43)	2019-07-10 12:00:05.100000+00:00

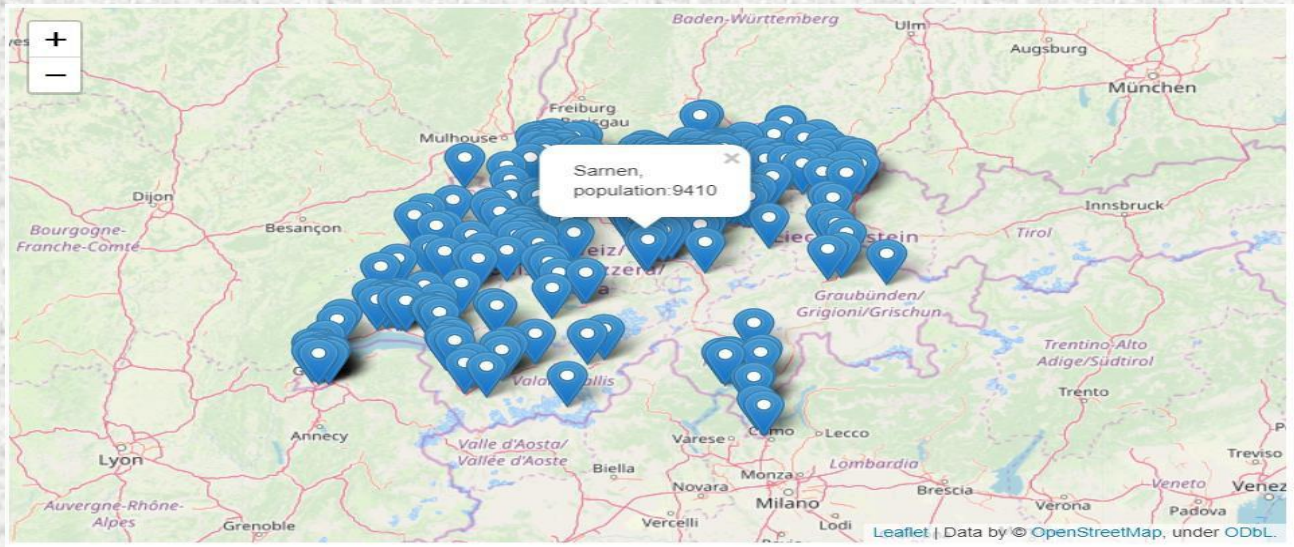
Изчертаване с Folium

```
import folium

map_cities = folium.Map(location=[47.562608, 7.599175], zoom_start=7)

def create_marker(row):
    lng = row["geometry"].x
    lat = row["geometry"].y
    name = row["name"]
    population = str(int(row["population"]))
    folium.Marker([lat, lng], popup=f'{name}, population:{population}').add_to(map_cities)

swiss_cities.apply(create_marker, axis=1)
map_cities
```

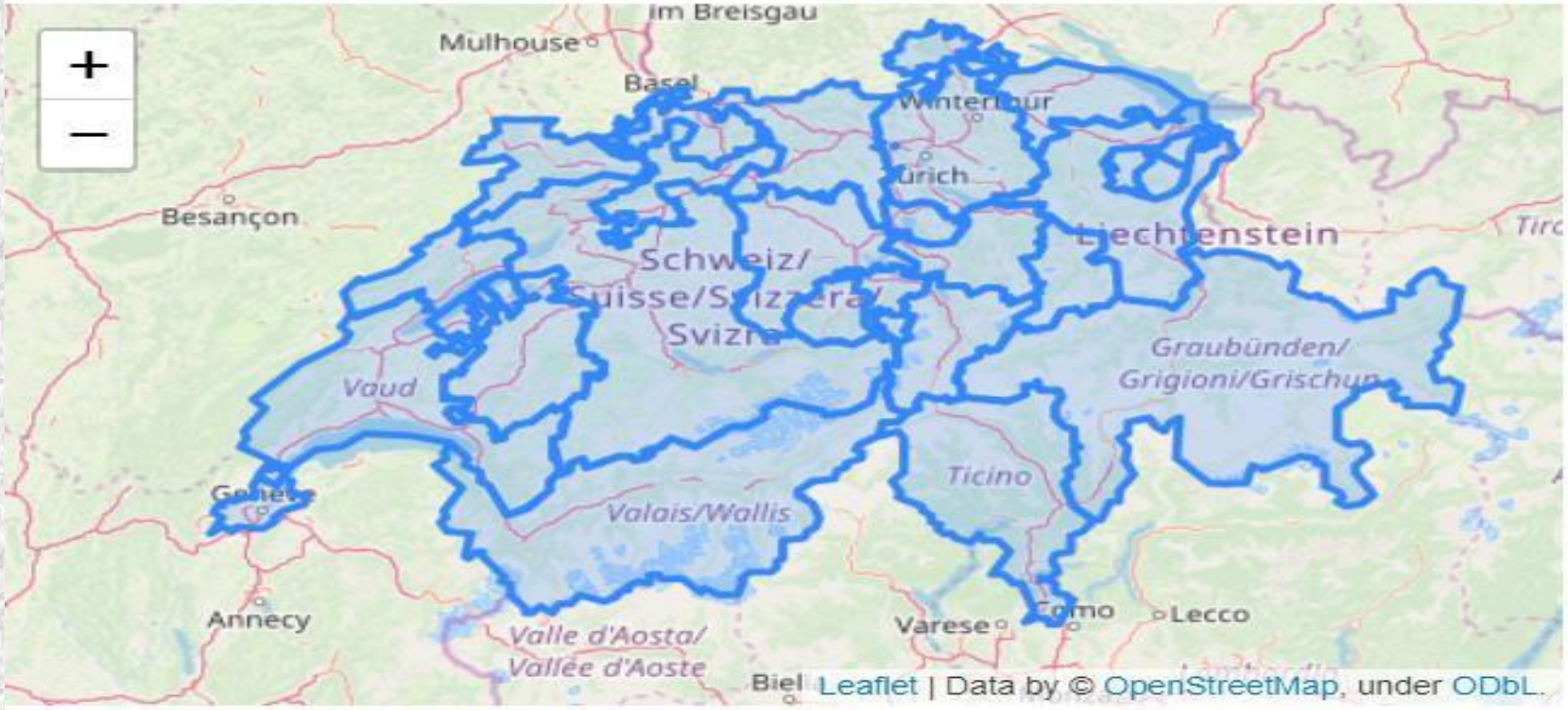


GeoJSON се поддържа от Folium

```
import json
from folium import GeoJson

f = open("data/switzerland.geojson")
switzerland = json.load(f)
f.close()

m = folium.Map(location=[46.889920, 8.305847], zoom_start=8)
GeoJson(switzerland).add_to(m)
m
```

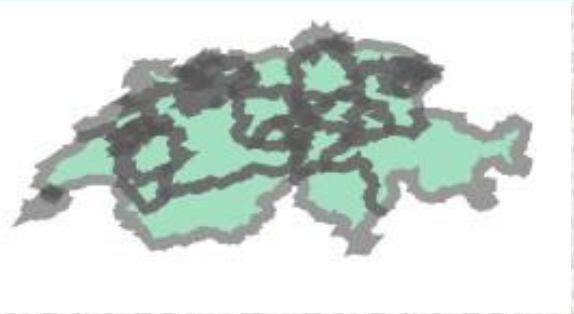


Зареждаме GeoJSON като Shapely GeometryCollection

```
import json
from shapely.geometry import shape, GeometryCollection


with open("data/switzerland.geojson") as f:
    features = json.load(f)["features"]

gc = GeometryCollection([shape(feature["geometry"]).buffer(0) for feature in features])
```



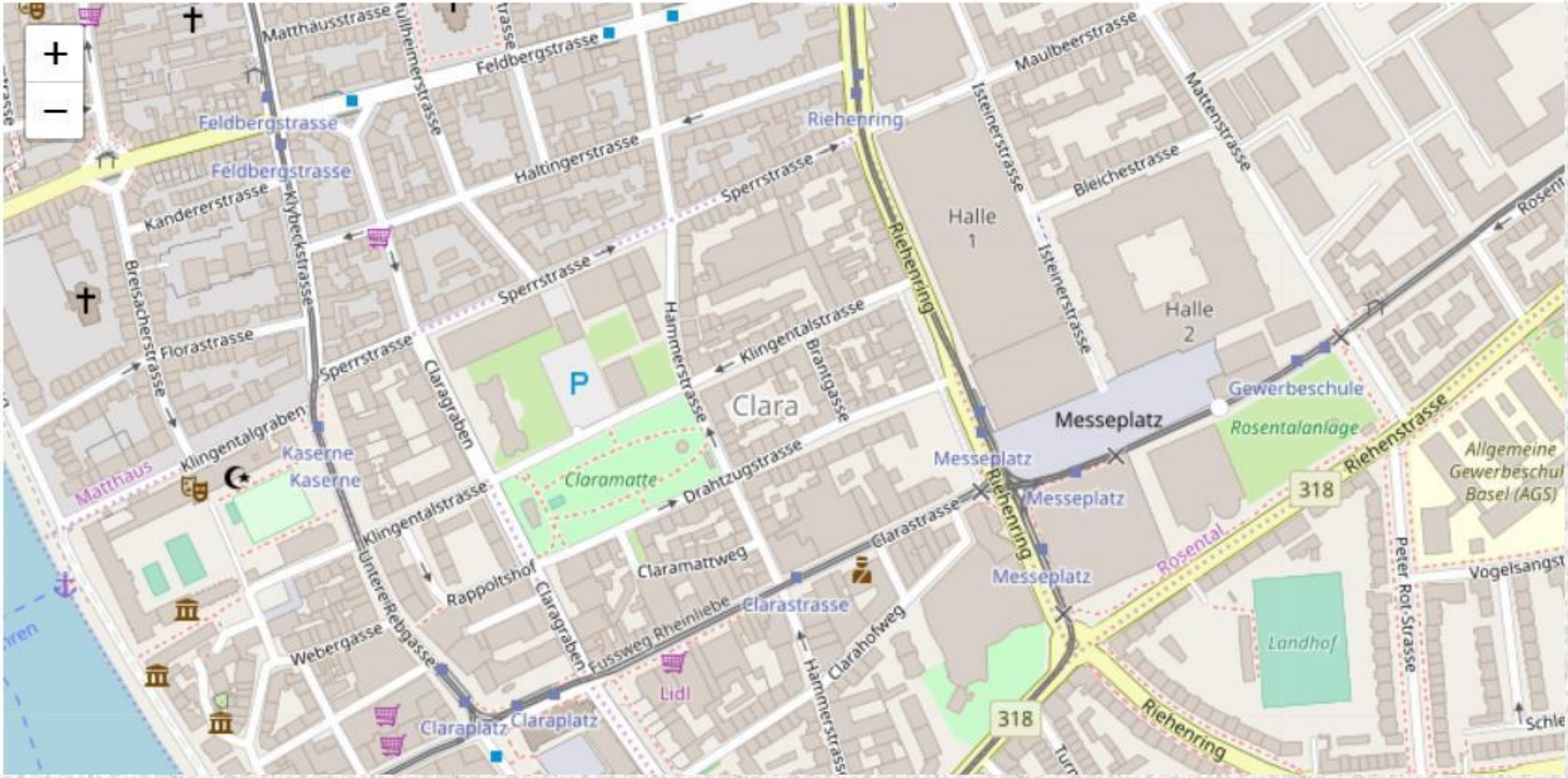
```
from shapely.geometry import MultiPolygon

switzerland = MultiPolygon(gc)
```

Folium: чрез модул pandas и библиотека leaflet.js създава интерактивни карти

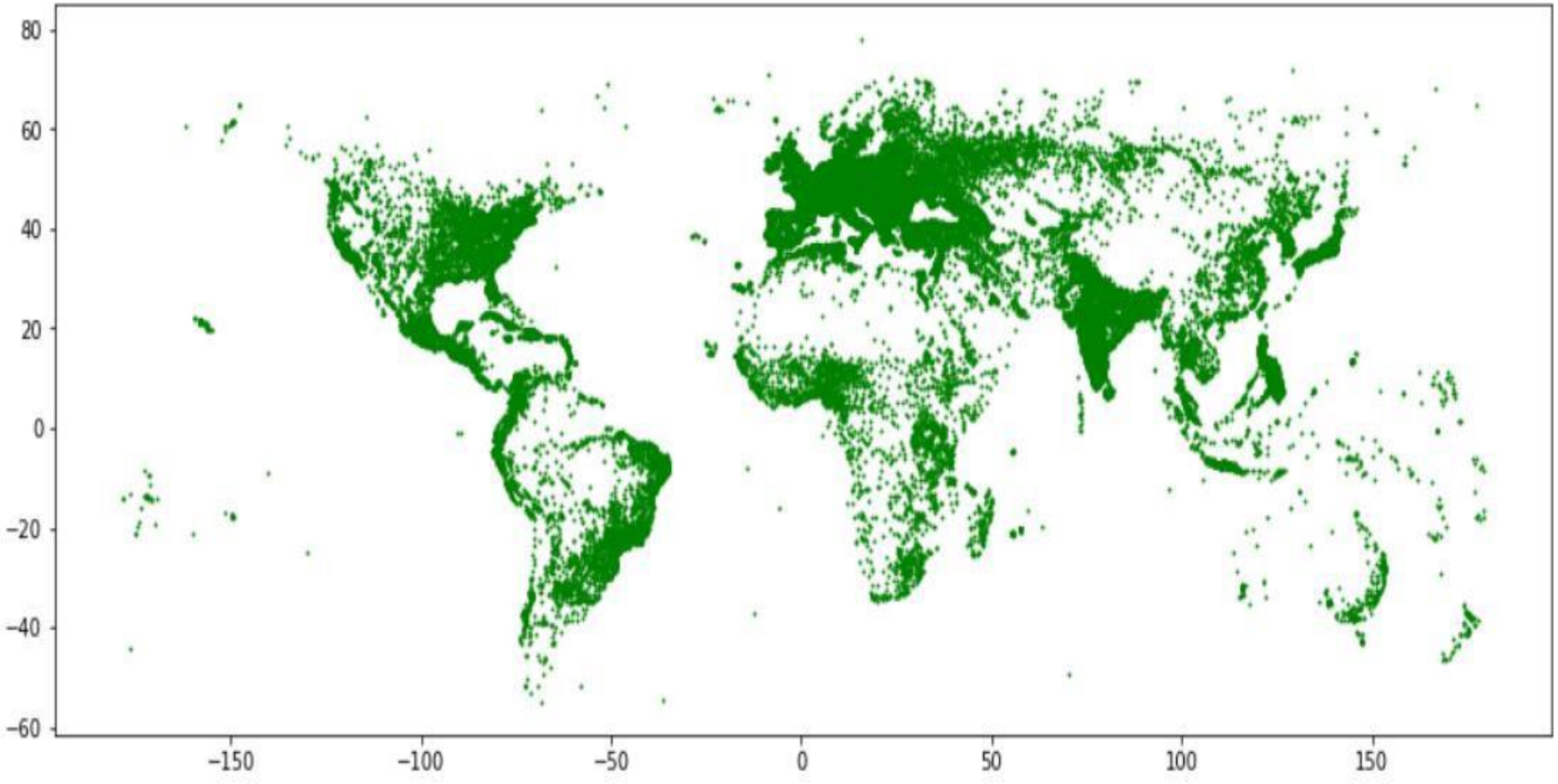
```
import folium

m = folium.Map(location=[47.562608, 7.599175], zoom_start=16)
m
```



Изчертаване на карта с Matplotlib

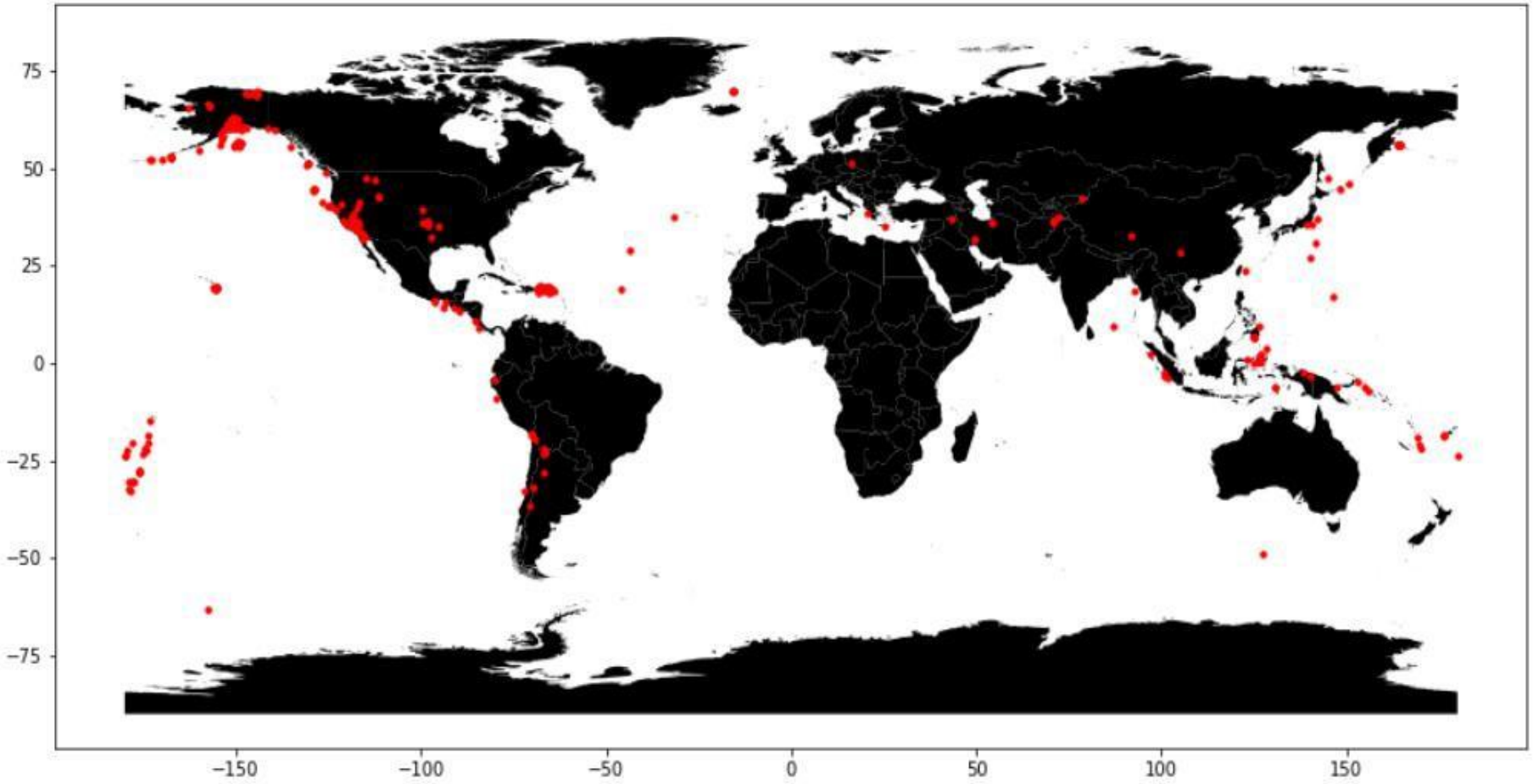
```
%matplotlib inline
gdf.plot(color='green', markersize=1, figsize=(15,9));
```



Изчертаване с fiona и Pandas

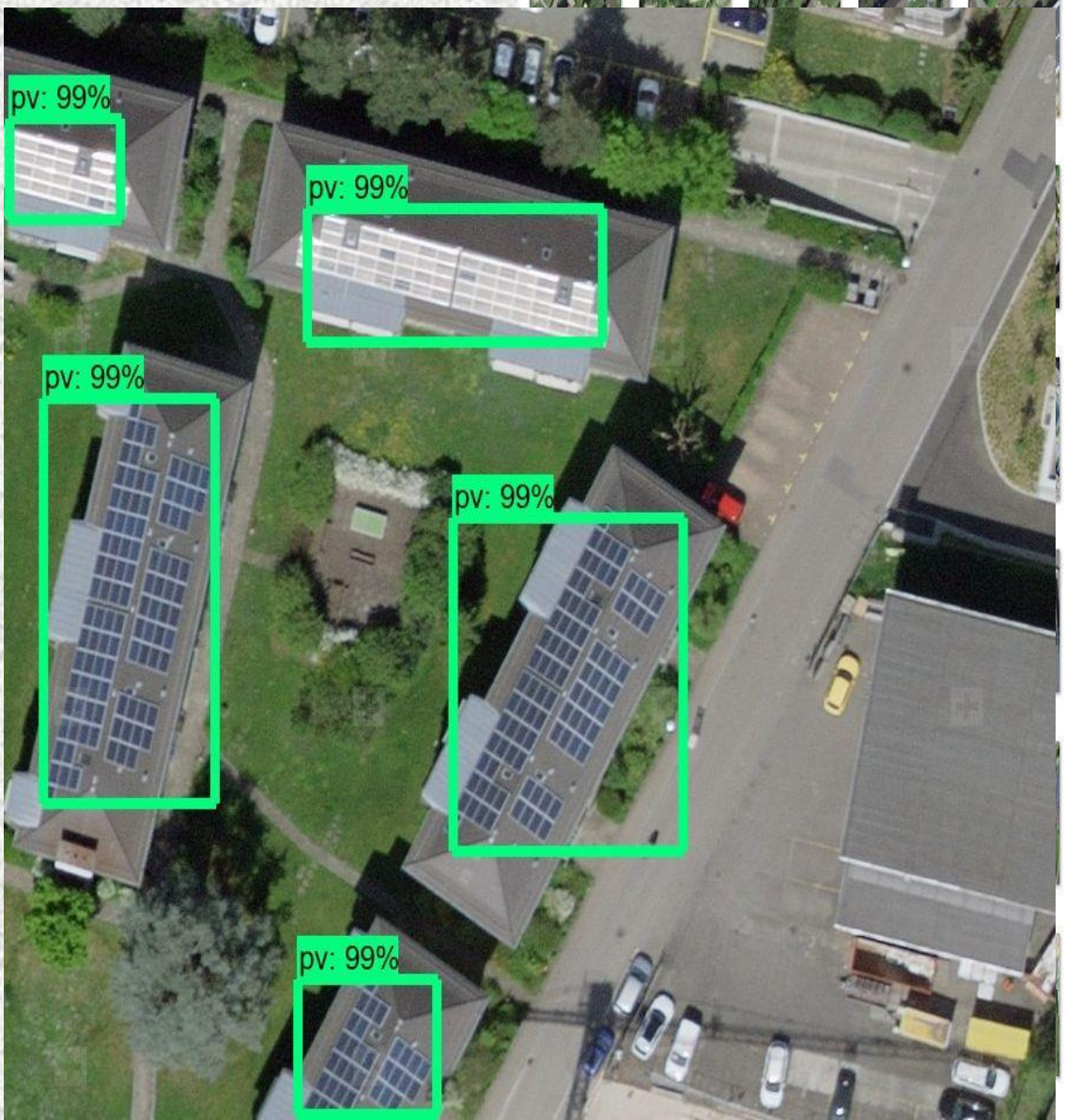
```
gdfAdmin0 = gpd.read_file("data/ne_10m_admin_0_countries/ne_10m_admin_0_countries.shp", encoding="utf-8")
```

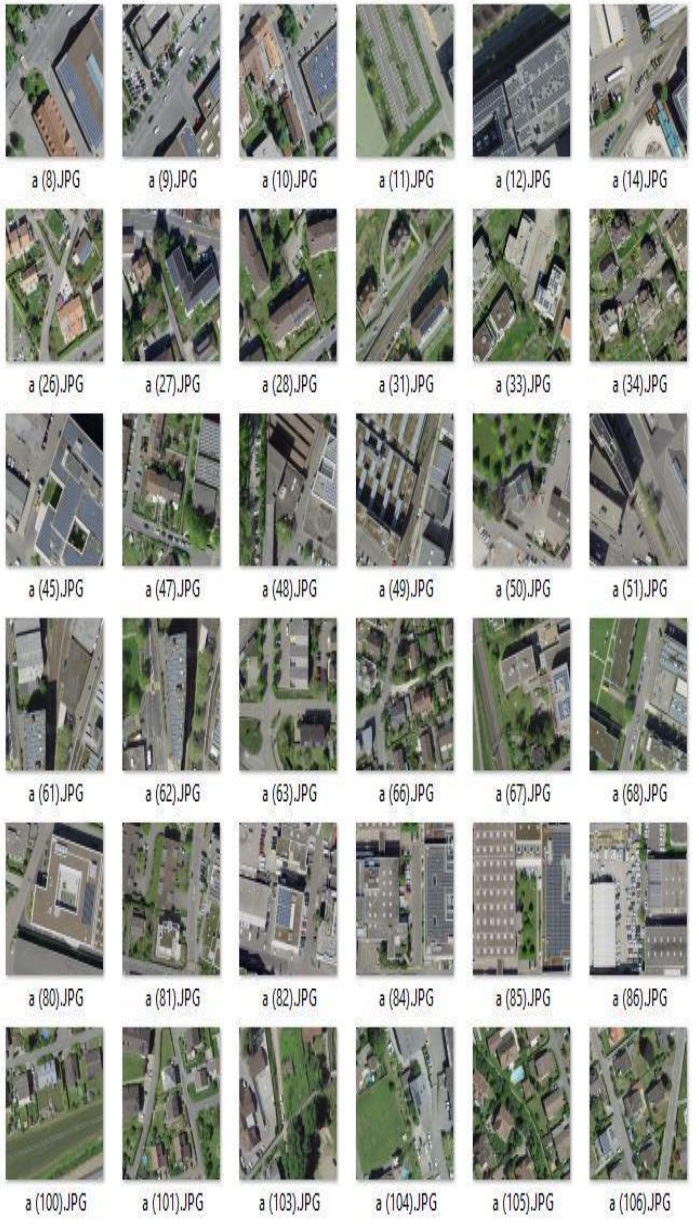
```
countries = gdfAdmin0.plot(figsize=(15,9), color="black")
eq.plot(ax=countries, color="red", markersize=10);
```



Лекция 13

Откриване на слънчеви панели с Tensorflow





Geoparsing (географски разбор)

Гео-разбор (geoparsing) е процес, в който по текстово описание на местоположение, се идентифицира точното географско местоположение на карта. В социалните науки от 2015 година се използва т.нар. Edinburgh Geoparser (http://groups.inf.ed.ac.uk/geoparser/documentation/v1.1/html/intro.html), който се опитва автоматично да определя точното местоположение на всяко географско място по текстовото му описание. По зададен текст, тя се опитва да покаже в Google Maps точното местоположение, отговрящо на текстовото описание.

Geoparsing с Python

Ще демонстрираме как може да се реализира дейност подобна на Edingbourgh Geoparser с програма на Python:

1. Ще покажем как можем да заредим исторически текст за анализиране
2. Как можем да отделим географските наименования (например имена на градове)
3. Как можем да получим точното местоположение на всеки град (неговите географски координати)
4. Как можем да визуализираме местоположението на всеки град върху географска карта

Зареждане на текст

Ще вземем географския пътеводител от М. Твен: The Innocents Abroad, който е свободно наличен в Интернет: http://www.gutenberg.org/files/3176/3176-0.txt
Зареждаме го чрез модула requests от пакета urllib:

```
>>> import urllib.request
>>> url = "http://www.gutenberg.org/files/3176/3176-0.txt"
>>> response = urllib.request.urlopen(url)
>>> raw = response.read().decode('utf8')
>>> print(f'{type(raw)}, \n{len(raw)}, \n{raw[:501]}')
```

Командата print извежда първата страница от ръкописа, който сме получили като един голям символен низ от 1145397 символа.

Зареждане на текст

<class 'str'>,
1145397,
Project Gutenberg's The Innocents Abroad, by Mark Twain (Samuel Clemens)

This eBook is for the use of anyone anywhere at no cost and with almost no restrictions whatsoever. You may copy it, give it away or re-use it under the terms of the Project Gutenberg License included with this eBook or online at www.gutenberg.net

Title: The Innocents Abroad

Author: Mark Twain (Samuel Clemens)

Release Date: August 18, 2006 [EBook #3176]

Last Updated: February 23, 2018

Language: English

Лексикографски анализ

Ще извлечем имената на градовете от текста чрез библиотеката geoText, която разпознава географските места като държави и градове.
Първо ще инсталираме библиотеката с командата от ОС pip install, а после ще я заредим в Python с командата: from geotext import GeoText

С командата cities = list(places.cities) ще получим списък от всички градове с повторения.

(Примерът е на следващият слайд)

Лексикографски анализ

pip install
https://github.com/elyase/geotext/archive/master.zip

```
>>> from geotext import GeoText
>>> places = GeoText(pp_raw)
>>> cities = list(places.cities)
>>> cities
['Tangier', 'Paris', 'Temple', 'Como', 'Garibaldi', 'Rome',
'Roman', 'Naples', 'Naples', ... ( 1091 града с повторения) ]
>>> unique_cities = list(set(cities))
>>> len(unique_cities)
184
```

(184 уникални имена на градове)

Получаване на координатите

Ще използваме модула geopy, който има автоматичен декодер за получаване на географски координати на географски обекти. Базиран е на свободната карта OpenStreetMap (https://www.openstreetmap.org/)
Инсталира се с командата:
pip install geopy

В тази библиотека има свободен метод за търсене Nominatum с който в цикъл ще получим за всеки град по неговото име неговите географски координати.

Подробностите са в програмата на следващия слайд.

Получаване на координатите

```
>>> from geopy.geocoders import Nominatim
>>> geolocator = Nominatim(user_agent="python38")
>>> lat_lon = []
>>> for city in unique_cities:
    try:
        location = geolocator.geocode(city)
        if location:
            print(location.latitude, location.longitude)
            lat_lon.append(location)
    except GeocoderTimeOut as e:
        print("Error: geocode failed on input %s with message %s"%(city, e))
```

Получаване на координатите

На екрана се разпечатват координатите на всички градове:
45.939475900000005 9.149410145408947
44.4970713 34.1586871
37.176059949999996 -3.588110277327934 7
31.778345 35.2250786
38.9819845 -77.12423413116392
43.850374 -79.024658
30.048819 31.243666
38.1937571 15.5542082
39.5695818 2.6500745
36.2452294 -101.8858689
….

Получаване на карта

Ще използваме модула folium за изчертаване на карта и за нанасяне на точките с указаните координати върху картата. Резултатът ще запазим в HTML файл, който може да бъде визуализиран в произволен уеб браузър.

За инсталиране на този модул:

pip install folium

За зареждането му в програмата:

>>> import folium

Допълнително ще ни трябва модула pandas:

>>> import pandas as pd

Получаване на карта

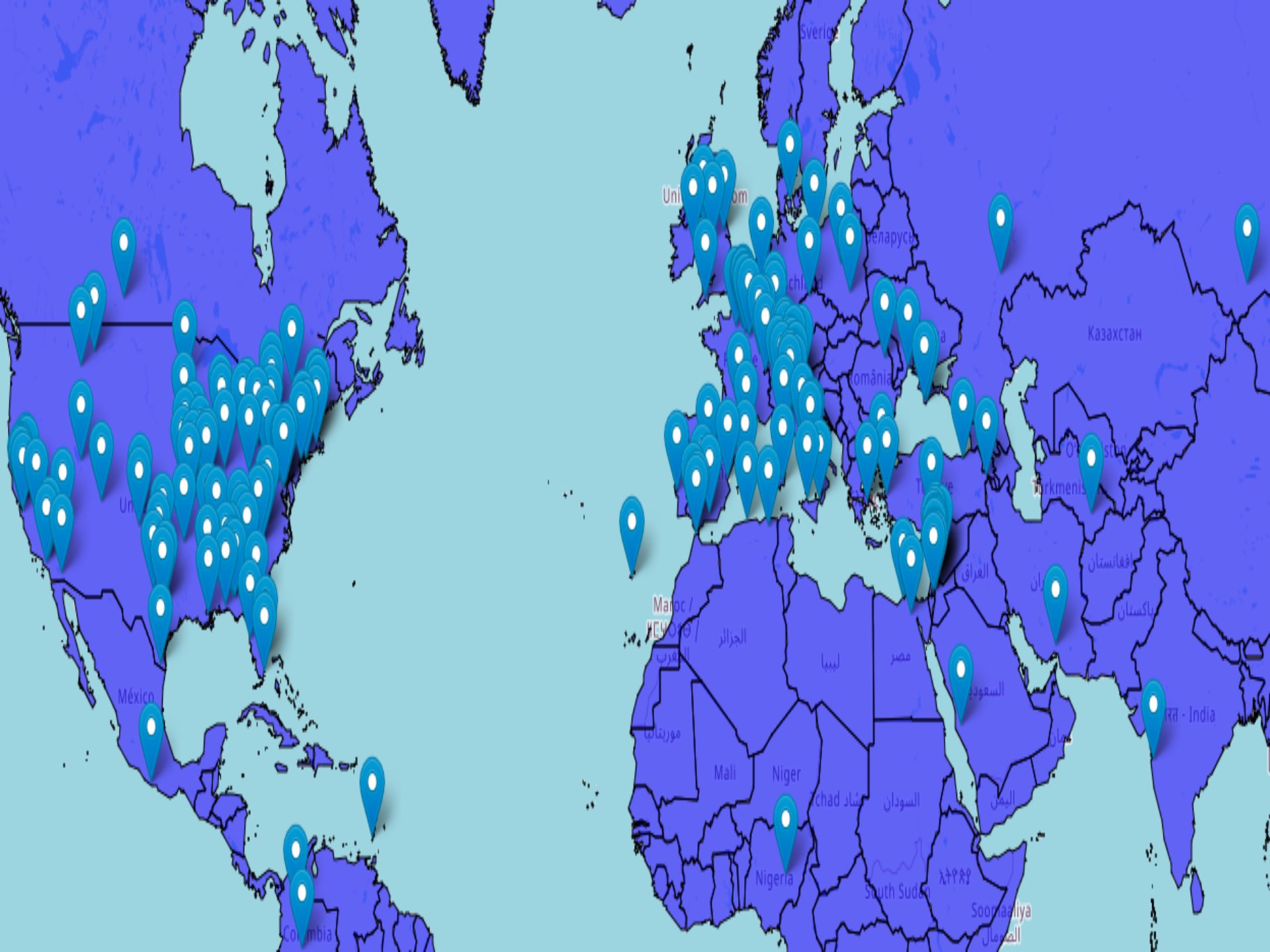
Ще преобразуваме данните в таблица (datframe) от pandas:

```
>>> df = pd.DataFrame(lat_lon, columns=['City Name', 'Coordinates'])
>>> df.head(7)

# Създаваме празна карта
>>> m = folium.Map(location=[20, 0], tiles="Mapbox Bright", zoom_start=2)

# Добавяме по един маркер на картата за всеки град
>>> for i in range(0,len(data)):
    folium.Marker([data.iloc[i]['lon'], data.iloc[i]['lat']],
                  popup=data.iloc[i]['name']).add_to(m)

# Запазваме картата с маркираните градове като html файл
>>> m.save('312_markers_on_folium_map1.html')
```


Sverige
Беларусь
Казахстан
România
Türkmenistan
México
Mali
Niger
Nigeria
Chad
South Sudan
Soomaaliya
India

Създаване на таблица с градове

```
import folium
import pandas as pd
>>> import pandas as pd
>>> df = pd.read_csv('cities15000.txt', sep='\t', encoding='utf-8',
header=None)
>>> df.head()
      0                1   ...          17          18
0  3040051           les Escaldes  ...  Europe/Andorra  2008-10-15
1  3041563        Andorra la Vella  ...  Europe/Andorra  2020-03-03
2   290594  Umm Al Quwain City  ...      Asia/Dubai  2019-10-24
3   291074   Ras Al Khaimah City  ...      Asia/Dubai  2019-09-09
4   291580             Zayed City ...      Asia/Dubai  2019-10-24

[5 rows x 19 columns]
```

Оформяне на таблицата с градове

```
>>> df2 = df[[1, 4, 5, 14]]
>>> df2.columns = ['name', 'lat', 'lng', 'population']
>>> df2.head()
                 name      lat      lng  population
0          les Escaldes  42.50729  1.53414       15853
1     Andorra la Vella  42.50779  1.52109       20430
2  Umm Al Quwain City  25.56473  55.55517       62747
3   Ras Al Khaimah City  25.78953  55.94320      351943
4            Zayed City  23.65416  53.70522       63482
>>>
```

Изчертаване на градовете

```
>>> m = folium.Map(location=[40, 20], zoom_start=3)
>>> state_geo =
'https://raw.githubusercontent.com/dataworkshop/visualization/master/
geo_json/world_geojson_from_ogr.json'
>>> folium.Choropleth(geo_data=state_geo,
                      name='choropleth',
                     legend_name='Biggest cities in the world’
                     ).add_to(m)
<folium.features.Choropleth object at 0x0000000017E3C250>

>>> m.save('folium_map.html')
```

Заключение

- Работа с географска информация в Python е лесна
- Много налични свободни библиотеки
- Различни методи за рисуване на карти
- Различни методи и стандарти за представяне и обработка на географска информация
- Езикът Python е идеален инструмент за решаване на всякакви географски задачи и проблеми. Единственото ограничение е вашето въображение

Какво научихте

- ✓ Въведение
- ✓ Системи за глобално позициониране
- ✓ Представяне на географска информация
- ✓ Примери с библиотеки на Python
- ✓ Географски карти и координати
- ✓ Библиотека GeoPandas
- ✓ Примери за онагледяване на географска информация
- ✓ Геопарсинг с Python
- ✓ Създаване на таблици и визуализиране