Python programming and data visualisation

Project II

DOCUMENTATION

# 

[**Authors**](#_9i3opo2m7xil) **1**

[**Introduction**](#_o0ar09u84mo9) **1**

[**Technologies**](#_ia5nr2fffvvw) **2**

[**OpenWeatherMap API**](#_hmf3mb1sf4yn) **2**

[**Data Model**](#_ltdz4hf39xj6) **3**

[**Application Usage**](#_sbws02y3cjhm) **4**

[**Conclusions**](#_qnntearkvkuz) **5**

# 

## Authors

* Piotr Wachulec, index number 269965
* Stanisław Pawlak, index number 279064

## Introduction

Project goal was to create application, which makes weather forecast visualization for different localizations in Poland. The starting command should be:

**python3** showforecast.py input\_data\_config.json

where showforecast.py should be main Python script, and input\_data\_config.json should contain all data needed to initialize program (e.g. chosen city names, ids and localizations).

## Technologies

* Python 3.6
* PyQt5
* matplotlib
* vtk
* requests

## OpenWeatherMap API

We used free plan from <https://openweathermap.org/api>.

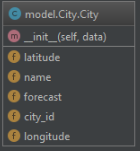
It allows to pull data from different cities for [current moment](https://openweathermap.org/current) and 5 days [forecast data](https://openweathermap.org/forecast5).

It contains structured prediction data for every 3 hours for next 5 days:

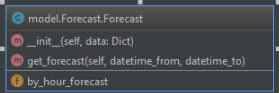
* list.main
  + list.main.temp Temperature. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
  + list.main.temp\_min Minimum temperature at the moment of calculation. This is deviation from 'temp' that is possible for large cities and megalopolises geographically expanded (use these parameter optionally). Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
  + list.main.temp\_max Maximum temperature at the moment of calculation. This is deviation from 'temp' that is possible for large cities and megalopolises geographically expanded (use these parameter optionally). Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
  + list.main.pressure Atmospheric pressure on the sea level by default, hPa
  + list.main.sea\_level Atmospheric pressure on the sea level, hPa
  + list.main.grnd\_level Atmospheric pressure on the ground level, hPa
  + list.main.humidity Humidity, %
* list.weather (more info Weather condition codes)
  + list.weather.id Weather condition id
  + list.weather.main Group of weather parameters (Rain, Snow, Extreme etc.)
  + list.weather.description Weather condition within the group
  + list.weather.icon Weather icon id
* list.clouds
  + list.clouds.all Cloudiness, %
* list.wind
  + list.wind.speed Wind speed. Unit Default: meter/sec, Metric: meter/sec, Imperial: miles/hour.
  + list.wind.deg Wind direction, degrees (meteorological)
* list.rain
  + list.rain.3h Rain volume for last 3 hours, mm
* list.snow
  + list.snow.3h Snow volume for last 3 hours

## Data Model

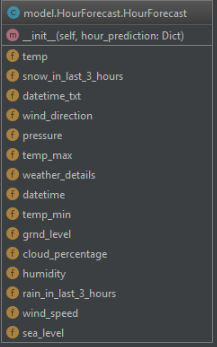
* City - mapped from API to store add process information about localizations. Especially useful when city name or id was needed.



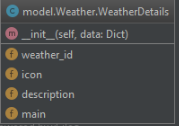
* Forecast - used for filtering data by time.



* HourForecast - mapped from API, used to store information about city weather prediction in specified hour.



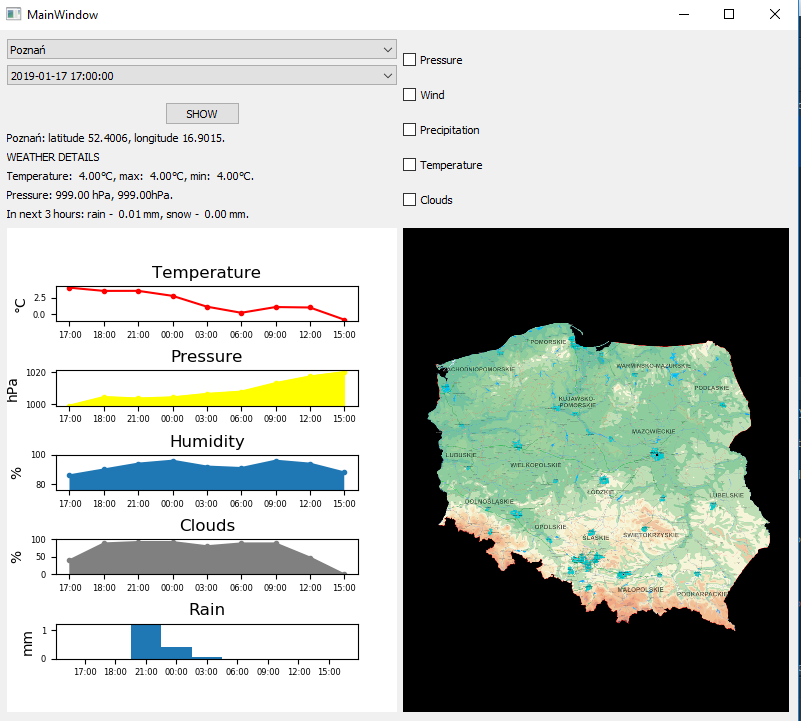
* WeatherDetails - details about weather in specified localizations and hour.



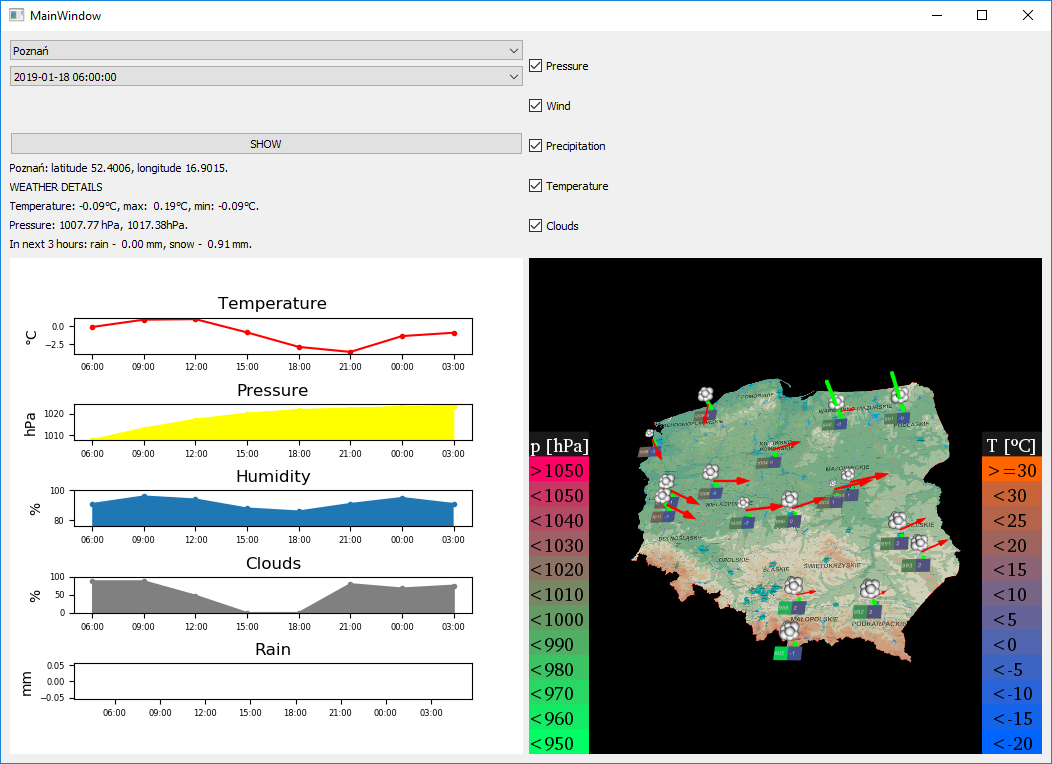
## Application Usage

You can change data plotted either by changing the city or by changing the date.

The forecast notes are shown below. On the right you can check one of checkbox to add this type of visualization for cities on map of Poland.



Example of map:



## Conclusions

1. OpenWeatherMap API is really simple and easy to use in QtApplication, but it has also some drawbacks. You must register to get a key to API, and the data structure varies between request, which logically should has the same structure. Sometimes the response is different than it is described in documentation, which is annoying. In some smaller cities like Świebodzin, which we also used the data is not updated with the same time period, so it also could lead to errors.
2. Matplotlib is also easy to start, but drawing customized plots require more time and focus.
3. VTK is really fast working tool, also easy to combine with QtApp. The drawback could be need for some boiler plate code every time you want add new object to scene.