//

Python4Delphi

maXbox Starter86_1 – Code with Python4Delphi

Be yourself; Everyone else is already taken.
    — Oscar Wilde.

In the last Article we have seen that P4D is a set of free components that wrap up the Python DLL into Delphi and Lazarus (FPC). For the next section I want to show how a simple Python evaluator works and to gain low-level access to the Python API.

On Win, the standard Python installer already associates the .py extension with a file type (Python.File) and gives that file type an open command that runs the interpreter (F:\Program Files\Python\python.exe "%1" %*). This is enough to make scripts executable from the command prompt. We use the python-dll as we use a windows dll. Therefore *.pyd files are dll's, but there are a few differences:
So far you have to know 3 different file types:

1. ***.py:** The norm input source code that we've written.

2. ***.pyc:** The compiled bytecode. If you import a module, py will build a *.pyc file that contains bytecode to make importing it again later easier and faster.
3. ***.pyd:** The mentioned windows dll file for Python.

If you have a DLL named foo.pyd, then it must have a function PyInit_foo(). You can then write Python "import foo", and Python will search for foo.pyd (as well as foo.py, foo.pyc) and if it finds it, will attempt to call PyInit_foo() to initialize it. Of course you do not link your .exe with foo.lib, as that would cause Windows to require the DLL to be present, we load it dynamically.

First we check our Python installation. Python provides for all user and current user installations. All user installations place the Py dll in the Windows System directory and write registry info to HKEY_LOCAL_MACHINE.
Current user installations place the dll in the install path and the registry info in HKEY_CURRENT_USER version < py 3.5.
So, for current user installations we need to try and find the install path since it may not be on the system path.

```
$IFDEF MSWINDOWS}
function IsPythonVersionRegistered(PythonVersion : string;
  out InstallPath: string; out AllUserInstall: Boolean) : Boolean;
  // The above convention was changed in Python 3.5.  Now even for all user
  // installations the dll is located at the InstallPath.
  // Also from vers.3.5 onwards 32 bit version have a suffix -32 e.g. "3.6-32"
  // See also PEP 514

var
  key: string;
  VersionSuffix: string;
  MajorVersion : integer;
  MinorVersion : integer;
begin
  Result := False;
  InstallPath := '';
  AllUserInstall := False;
  MajorVersion := StrToInt(PythonVersion[1]);
  MinorVersion := StrToInt(PythonVersion[3]);
  VersionSuffix := '';
{$IFDEF CPUX86}
  if (MajorVersion > 3) or ((MajorVersion = 3)  and (MinorVersion >= 5)) then
    VersionSuffix := '-32';
{$ENDIF}
  key:= Format('\Software\Python\PythonCore\%s%s\InstallPath',
                                            [PythonVersion, VersionSuffix]);

  // First try HKEY_CURRENT_USER as per PEP514
  try
    with TRegistry.Create1(KEY_READ and not KEY_NOTIFY) do
      try
        RootKey := HKEY_CURRENT_USER;
        if OpenKey(Key, False) then begin
          InstallPath := ReadString('');
          Result := True;
          Exit;
        end;
      finally
        Free;
      end;
  except
    writeln(' HKEY_CURRENT_USER except');
  end;

  //Then try for an all user installation
  try
    with TRegistry.Create1(KEY_READ and not KEY_NOTIFY) do
      try
        RootKey := HKEY_LOCAL_MACHINE;
        if OpenKey(Key, False) then begin
          AllUserInstall := True;
          if (MajorVersion > 3) or ((MajorVersion = 3)
                                and (MinorVersion >= 5)) then
            InstallPath := ReadString('');
          Result := True;
        end;
      finally
        Free;
      end;
  except
    writeln(' HKEY__LOCAL_MACHINE except');
  end;
end;
{$ENDIF}
```

In my case the path is on:
C:\Users\max\AppData\Local\Programs\Python\Python36\Lib\

Then we can simple check a first function or load on runtime the PyRun_SimpleString for our next example:

```
//if fileExistst(PYDLLPATH+ 'python37.dll';
  function getCopyRight: PChar;
      external 'Py_GetCopyright@C:\maXbox\EKON25\python37.dll stdcall';


function pyrun(command : pchar):integer;
      external 'PyRun_SimpleString@C:\maXbox\EKON25\python37.dll cdecl';

procedure pyinit;
      external 'Py_Initialize@C:\maXbox\EKON25\python37.dll cdecl';
procedure pyexit(retval: integer);
      external 'Py_Exit@C:\maXbox\EKON24\python37.dll cdecl';
```

Now we use to invoke a Python script as an embedding **const** and use the dll functionality of *Import('PyRun_SimpleString');*

To run python code direct in a maXbox, Free Pascal or whatever script you need to import just the 3 dll functions[1], above all *PyRun_SimpleStringFlags* or without flags:

```
Const PYDLLPATH = 'C:\maXbox\EKON25\';
      PYDLLNAME = 'python37.dll';
      PSCRIPTNAME = 'initpy.py';
```

This is a simplified interface to PyRun_SimpleString leaving the PyCompilerFlags* argument set to NULL. Normally the Python interpreter is initialized by Py_Initialize() so we use the same interpreter as from a shell or terminal:

*int PyRun_SimpleString(const char *command)*

```
  //function pyrun(command :pChar) :integer;
    //writeln('pyinitback: '+itoa
  pyinit();
  //retp:= 'print("hello low")'
  retp:= 'print()';
  //PyRun_SimpleString:   function( str: PAnsiChar): Integer; cdecl;
  //writeln(itoa(pyrun(retp)));
  writeln(itoa(pyrun('print("this is box")')));
  writeln(itoa(pyrun('import sys')));
  writeln(itoa(pyrun('f=open(r"C:\maXbox\maxbox4\pytest.txt","w")')));
  writeln(itoa(pyrun('f.write("Hello PyWorld_, \n")')));
  writeln(itoa(pyrun('f.write("Data will be written on the file.")')));
  writeln(itoa(pyrun('f.close()')));
```

Now check your file system to get the *pytest.txt* also use to invoke a Python script as an embedding **const** and use the dll like above. Next example combines an embedding python script in a

1 Independent from imports and site-packages

pascal script. We’ll use the py-script language and an awesome library called Python OpenWeatherMap (PyOWM) to make it easier to use the OpenWeatherMap API in Python. You'll have to pip install the library first in order to import the module:

```
C:\maXbox>pip3 install pyowm

Collecting pyowm

  Downloading pyowm-2.10.0-py3-none-any.whl (3.7 MB)
```

Then we run a command prompt command ('py '+RUNSCRIPT) with parameters like a command line interface and get the python output back in maXbox with the function *getDosOutput()*.

```
function GetDosOutput(CommandLine: string; Work: string = 'C:\: string;
 Ex.: writeln(GetDosOutput('java -version','C:\'));
      >>>java version "1.8.0_211"
         Java(TM) SE Runtime Environment (build 1.8.0_211-b12)
         Java HotSpot(TM) Client VM (build 25.211-b12, mixed mode)

procedure CaptureConsoleOutput(DosApp : string;AMemo : TMemo);
 Ex.: CaptureConsoleOutput('cmd /C dir *.* ',memo2);
      CaptureConsoleOutput('powershell /C dir *.* ',memo2);
```

This you can see often in P4D (later on), the script itself runs from a memo-control inside a form.

```
program OpenWeatherMap_Py_integrate;
const C=CRLF;
const SCRIPTNAMEP= '1046_openweather.py';
const DETECTOUTFILE= 'openweather_out2.txt';

Const PYSCRIPT6 =
'import pyowm                                               '+C+
'import wget                                                '+C+
'import sys, datetime as dt                                 '+C+
'#nltk.download("vader_lexicon")                            '+C+
'from time import sleep                                     '+C+
'import pandas as pd                                        '+C+
'pd.set_option("max_colwidth", 400)                         '+C+
'import numpy as np                                         '+C+
'print("this first line after config & imports")            '+C+
'                                                           '+C+
'output_path = sys.argv[1]                                  '+C+
'locate = sys.argv[2]                                       '+C+
'                                                                    '+C+
'owm= pyowm.OWM("55013bf3d09cfb0619989a00ed5bed09")                  '+C+
'observation= owm.weather_at_place((locate))                         '+C+
'we= observation.get_weather()                                       '+C+
'temp = we.get_temperature("celsius")                                '+C+
'with open(output_path, "w") as file:                                '+C+
'  file.write("OpenWeatherMap of "+locate+" "+str(dt.datetime.now())+ '+C+
'              "\n"+str(we)+                                         '+C+
'              "\n"+str(temp)+                                       '+C+
'              "\n"+str(dt.datetime.utcnow()))                       '+C+
'                                                                    '+C+
'print("\n")                                                         '+C+
'print("weather today:"+(locate)+" "+str(we)+"\n"+str(temp))         '+C+
'print("integrate weatherreport detector compute ends...")           ';
```

Then we use the parameters from the script as paramstrings. The *ParamStr()* function returns one of the parameters from the command line used to invoke the current script with outputpath for the file and locate, this means the place of the returned weather-report (ParamIndex determines which parameter is returned):

```
Writeln(getDosOutput('py '+RUNSCRIPT+' '+outputpath+' '+locate,
                                                      exePath));
```

```
Const ACTIVESCRIPT = PYSCRIPT6;

var RUNSCRIPT, outputPath, locate: string;
    startTime64, endTime64, freq64: Int64;

begin //@main
  //-init env
  maxform1.console1click(self);
  memo2.height:= 205;
  QueryPerformanceFrequency(freq64);

  //-config
  saveString(exepath+SCRIPTNAMEP, ACTIVESCRIPT);
  sleep(600)
  //outputPath:= '.\crypt\output\'+DETECTOUTFILE;

  if Not fileExists(exepath+DETECTOUTFILE) then
      CreateFileFromString(exepath+DETECTOUTFILE, 'Open_Weather_Data');

  outputPath:= Exepath+DETECTOUTFILE;
  locate:= '"Bern, CH"';

  if fileExists(exepath+SCRIPTNAMEP) then begin
     RUNSCRIPT:= exepath+SCRIPTNAMEP;
     QueryPerformanceCounter(startTime64);
     writeln(getDosOutput('py '+RUNSCRIPT+' '+outputpath+' '+locate, exePath));
     QueryPerformanceCounter(endTime64);
     println('elapsedSeconds:= '+floattostr((endTime64-startTime64)/freq64));
     openFile(outputPath)
    //}
  end;
end.
```

<u>Output:</u>

```
weather today:Bern, CH <pyowm.weatherapi25.weather.Weather - reference
time=2021-07-07 14:43:23+00, status=clouds, detailed status=scattered
clouds>

{'temp': 21.17, 'temp_max': 23.54, 'temp_min': 15.99, 'temp_kf': None}

integrate weatherreport detector compute ends...
```

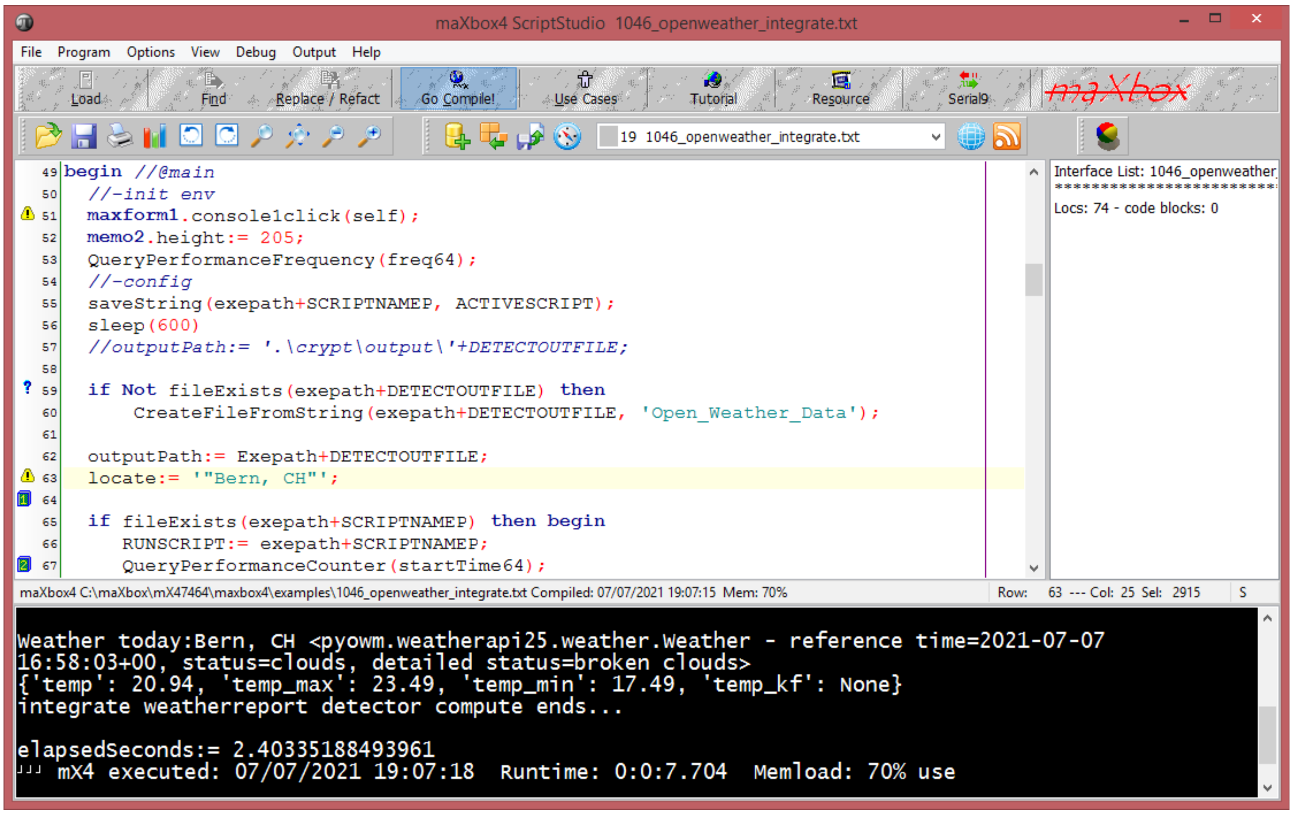
In P4D you do have the mentioned memo with ExeStrings:

```
procedure TForm1.Button1Click(Sender: Tobject);
begin
  PythonEngine1.ExecStrings( Memo1.Lines );
end;
```

This explains best the code behind, to evaluate an internal Python expression. You are responsible for creating one and only one TPythonEngine. Usually you just drop it on your main form.

With the *PythonGUIInputOutput1* you wire the *PythonEngine1* to a memo2, the same as in maXbox with a memo2 as console (from object Form1: TForm1 in Unit1.dfm):

```
object PythonEngine1: TPythonEngine
  IO = PythonGUIInputOutput1
  Left = 32
end
object PythonGUIInputOutput1: TPythonGUIInputOutput
  UnicodeIO = True
  RawOutput = False
  Output = Memo2
  Left = 64
end
```


maXbox4 ScriptStudio 1046_openweather_integrate.txt
File Program Options View Debug Output Help
Load Find Replace / Refact Go Compile! Use Cases Tutorial Resource Serial9
maXbox
19 1046_openweather_integrate.txt
Interface List: 1046_openweather
Locs: 74 - code blocks: 0

```
begin //@main
  //-init env
  maxform1.console1click(self);
  memo2.height:= 205;
  QueryPerformanceFrequency(freq64);
  //-config
  saveString(exepath+SCRIPTNAMEP, ACTIVESCRIPT);
  sleep(600)
  //outputPath:= '.\crypt\output\'+DETECTOUTFILE;

  if Not fileExists(exepath+DETECTOUTFILE) then
      CreateFileFromString(exepath+DETECTOUTFILE, 'Open_Weather_Data');

  outputPath:= Exepath+DETECTOUTFILE;
  locate:= '"Bern, CH"';

  if fileExists(exepath+SCRIPTNAMEP) then begin
     RUNSCRIPT:= exepath+SCRIPTNAMEP;
     QueryPerformanceCounter(startTime64);
```

```
maXbox4 C:\maXbox\mX47464\maxbox4\examples\1046_openweather_integrate.txt Compiled: 07/07/2021 19:07:15 Mem: 70%    Row: 63 --- Col: 25 Sel: 2915    S
```

```
Weather today:Bern, CH <pyowm.weatherapi25.weather.Weather - reference time=2021-07-07
16:58:03+00, status=clouds, detailed status=broken clouds>
{'temp': 20.94, 'temp_max': 23.49, 'temp_min': 17.49, 'temp_kf': None}
integrate weatherreport detector compute ends...

elapsedSeconds:= 2.40335188493961
┘┘┘ mX4 executed: 07/07/2021 19:07:18  Runtime: 0:0:7.704  Memload: 70% use
```

_PIC: 1046_openweather_P4D_2.png

The unit *PythonEngine.pas* is the main core-unit of the framework. Most of the Python/C API is presented as published/public member functions of the engine unit.

```
...
Py_BuildValue              := Import('Py_BuildValue');
Py_Initialize              := Import('Py_Initialize');
PyModule_GetDict           := Import('PyModule_GetDict');
PyObject_Str               := Import('PyObject_Str');
PyRun_String               := Import('PyRun_String');
PyRun_SimpleString         := Import('PyRun_SimpleString');
PyDict_GetItemString       := Import('PyDict_GetItemString');
PySys_SetArgv              := Import('PySys_SetArgv');
Py_Exit                    := Import('Py_Exit');
...
```

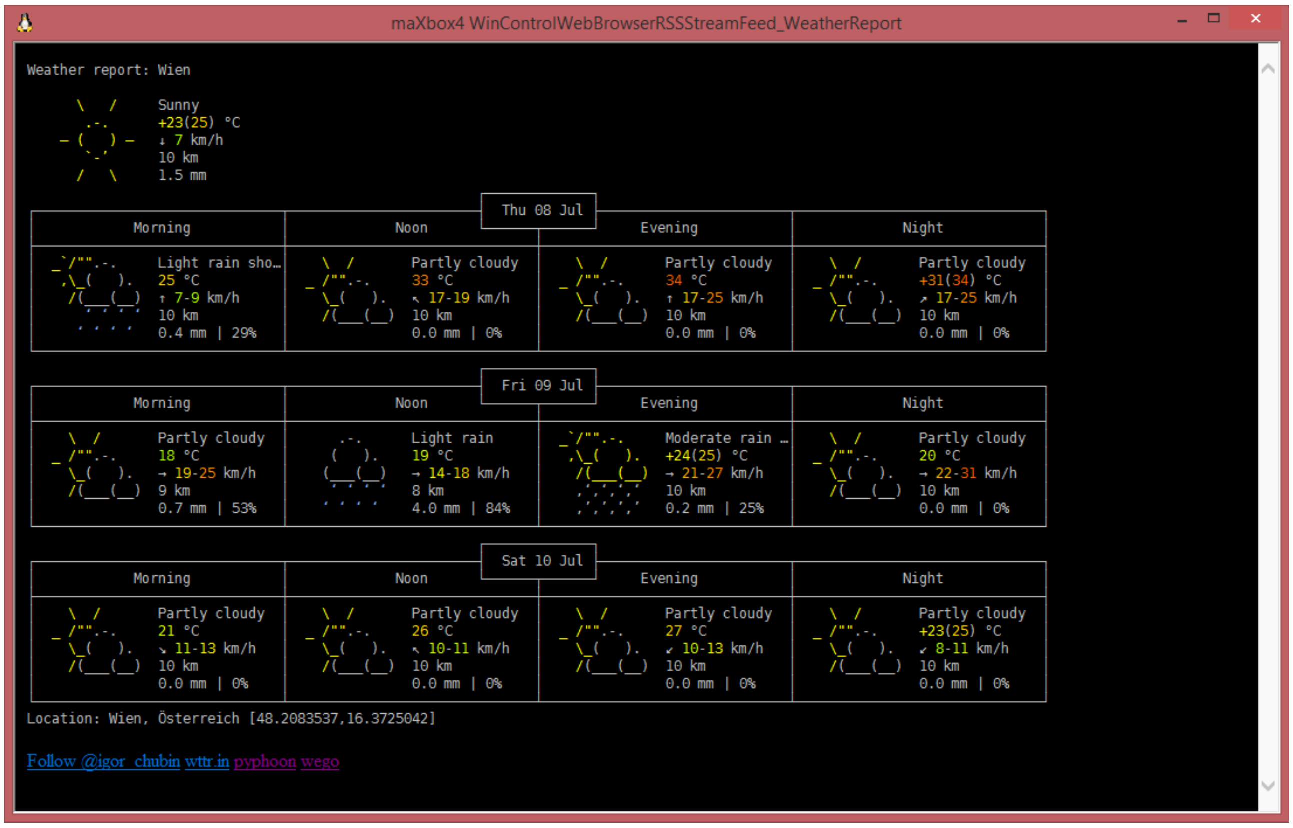
Let's take a last look at the functionality of *PyRun_SimpleString* mentioned first within the openweather const script.

http://www.softwareschule.ch/examples/openweather.txt

or a second larger script:

http://www.softwareschule.ch/examples/1016_newsfeed_sentiment_integrate2.txt

PyRun_SimpleString: function(str: PAnsiChar): Integer; cdecl;


maXbox4 WinControlWebBrowserRSSStreamFeed_WeatherReport

```
Weather report: Wien

      \   /     Sunny
       .-.      +23(25) °C
    ― (   ) ―   ↓ 7 km/h
       `-’      10 km
      /   \     1.5 mm
                                                       ┌─────────────┐
┌──────────────────────────────┬───────────────────────┤  Thu 08 Jul ├───────────────────────┬──────────────────────────────┐
│            Morning           │             Noon      └──────┬──────┘     Evening           │             Night            │
├──────────────────────────────┼──────────────────────────────┼──────────────────────────────┼──────────────────────────────┤
│  _`/"".-.     Light rain sho…│    \  /       Partly cloudy  │    \  /       Partly cloudy  │    \  /       Partly cloudy  │
│   ,\_(   ).   25 °C          │  _ /"".-.     33 °C          │  _ /"".-.     34 °C          │  _ /"".-.     +31(34) °C     │
│    /(___(__)  ↑ 7-9 km/h     │    \_(   ).   ↖ 17-19 km/h   │    \_(   ).   ↑ 17-25 km/h   │    \_(   ).   ↗ 17-25 km/h   │
│      ‘ ‘ ‘ ‘  10 km          │    /(___(__)  10 km          │    /(___(__)  10 km          │    /(___(__)  10 km          │
│     ‘ ‘ ‘ ‘   0.4 mm | 29%   │               0.0 mm | 0%    │               0.0 mm | 0%    │               0.0 mm | 0%    │
└──────────────────────────────┴──────────────────────────────┴──────────────────────────────┴──────────────────────────────┘
                                                       ┌─────────────┐
┌──────────────────────────────┬───────────────────────┤  Fri 09 Jul ├───────────────────────┬──────────────────────────────┐
│            Morning           │             Noon      └──────┬──────┘     Evening           │             Night            │
├──────────────────────────────┼──────────────────────────────┼──────────────────────────────┼──────────────────────────────┤
│    \  /       Partly cloudy  │     .-.      Light rain     │  _`/"".-.     Moderate rain …│    \  /       Partly cloudy  │
│  _ /"".-.     18 °C          │    (   ).    19 °C          │   ,\_(   ).   +24(25) °C     │  _ /"".-.     20 °C          │
│    \_(   ).   → 19-25 km/h   │   (___(__)   → 14-18 km/h   │    /(___(__)  → 21-27 km/h   │    \_(   ).   → 22-31 km/h   │
│    /(___(__)  9 km           │    ‘ ‘ ‘ ‘   8 km           │    ‚‘‚‘‚‘‚‘   10 km          │    /(___(__)  10 km          │
│               0.7 mm | 53%   │   ‘ ‘ ‘ ‘    4.0 mm | 84%   │    ‚’‚’‚’‚’   0.2 mm | 25%   │               0.0 mm | 0%    │
└──────────────────────────────┴──────────────────────────────┴──────────────────────────────┴──────────────────────────────┘
                                                       ┌─────────────┐
┌──────────────────────────────┬───────────────────────┤  Sat 10 Jul ├───────────────────────┬──────────────────────────────┐
│            Morning           │             Noon      └──────┬──────┘     Evening           │             Night            │
├──────────────────────────────┼──────────────────────────────┼──────────────────────────────┼──────────────────────────────┤
│    \  /       Partly cloudy  │    \  /       Partly cloudy  │    \  /       Partly cloudy  │    \  /       Partly cloudy  │
│  _ /"".-.     21 °C          │  _ /"".-.     26 °C          │  _ /"".-.     27 °C          │  _ /"".-.     +23(25) °C     │
│    \_(   ).   ↘ 11-13 km/h   │    \_(   ).   ↖ 10-11 km/h   │    \_(   ).   ↙ 10-13 km/h   │    \_(   ).   ↙ 8-11 km/h    │
│    /(___(__)  10 km          │    /(___(__)  10 km          │    /(___(__)  10 km          │    /(___(__)  10 km          │
│               0.0 mm | 0%    │               0.0 mm | 0%    │               0.0 mm | 0%    │               0.0 mm | 0%    │
└──────────────────────────────┴──────────────────────────────┴──────────────────────────────┴──────────────────────────────┘
Location: Wien, Österreich [48.2083537,16.3725042]

Follow @igor_chubin wttr.in pyphoon wego
```

_PIC: 1046_openweather_ansiview2.png

Wiki & EKON P4D topics

- https://entwickler-konferenz.de/delphi-innovations-fundamentals/python4delphi/

- https://learndelphi.org/python-native-windows-gui-with-delphi-vcl/
- http://www.softwareschule.ch/examples/weatherbox.txt
-

Learn about Python for Delphi

- Tutorials
- Demos
  https://github.com/maxkleiner/python4delphi
- https://raw.githubusercontent.com/wiki/pyscripter/python4delphi/Files/Chapter80Tutorial.pdf
-

Note: You will need to adjust the demos accordingly, to successfully load the Python distribution that you have installed on your computer.

Doc: https://maxbox4.wordpress.com

Appendix: PIP3 Install pyowm:

https://medium.com/nexttech/how-to-use-the-openweathermap-api-with-python-c84cc7075cfc

```
 forecast = owm.three_hours_forecast('mumbai')
 TypeError: 'module' object is not callable

C:\maXbox>pip3 install pyowm
Collecting pyowm
  Downloading pyowm-2.10.0-py3-none-any.whl (3.7 MB)
     |████████████████████████████████| 3.7 MB 819 kB/s
Requirement already satisfied: geojson<3,>=2.3.0 in c:\users\max\appdata\local\p
rograms\python\python36\lib\site-packages (from pyowm) (2.4.0)
Requirement already satisfied: requests<3,>=2.20.0 in c:\users\max\appdata\local
\programs\python\python36\lib\site-packages (from pyowm) (2.24.0)
Requirement already satisfied: chardet<4,>=3.0.2 in c:\users\max\appdata\local\p
rograms\python\python36\lib\site-packages(from requests<3,>=2.20.0->pyowm)(3.0
.4)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in c:\use
rs\max\appdata\local\programs\python\python36\lib\site-packages (from requests<3
,>=2.20.0->pyowm) (1.22)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\max\appdata\local\
programs\python\python36\lib\site-packages (from requests<3,>=2.20.0->pyowm) (20
18.1.18)
Requirement already satisfied: idna<3,>=2.5 in c:\users\max\appdata\local\progra
ms\python\python36\lib\site-packages (from requests<3,>=2.20.0->pyowm) (2.6)
Installing collected packages: pyowm
Successfully installed pyowm-2.10.0
WARNING: You are using pip version 20.1.1; however,version 21.1.3 is available.
```