//

Synthetic Data Generator

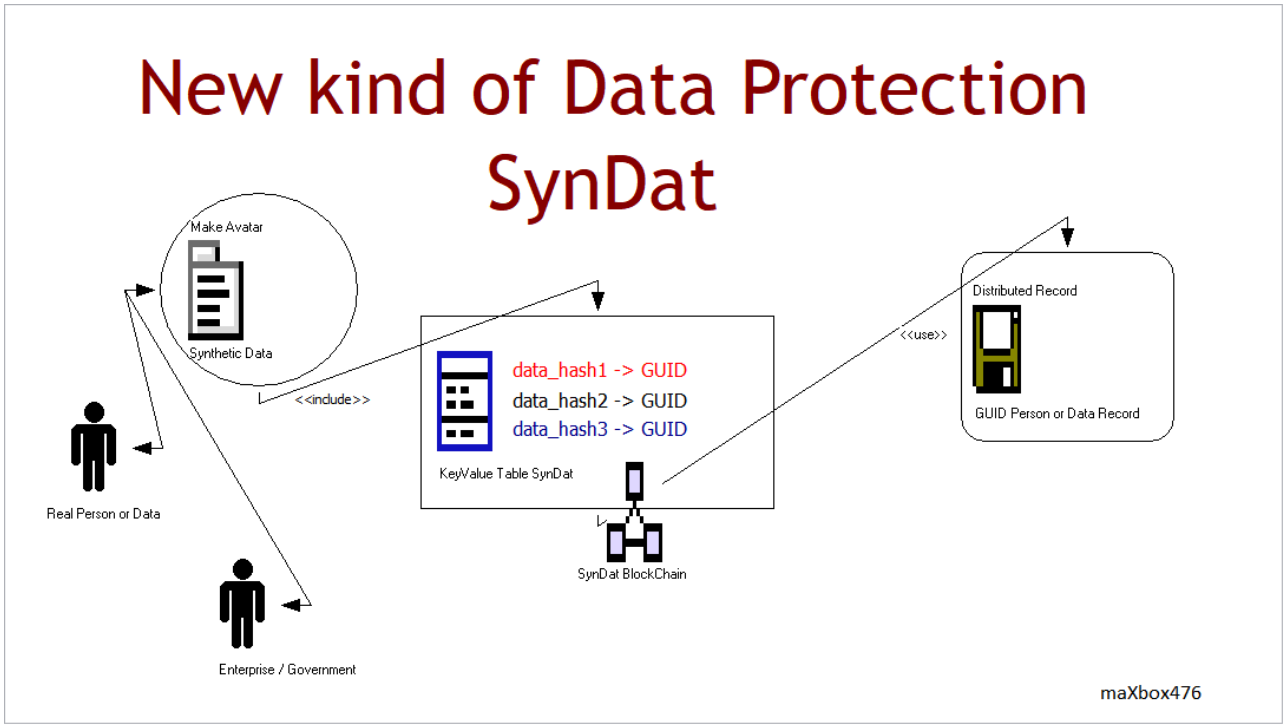
maXbox Starter91 - Build with P4D and SynDat

Make the fake.

Real data, extracted from the real world, is a gold standard for data science and data protection, perhaps for obvious reasons. In such a case, synthetic data producing can be used either in place of real data, protect real user as an avatar or to augment an insufficiently large dataset. With Python4Delphi scripting.

http://www.softwareschule.ch/examples/pydemo32_2.txt

Faker is a Python library[1] that generates fake data. Fake data is often used for testing or filling databases with some dummy data. Faker is strong inspired by PHP's Faker, Perl's Data::Faker, and Ruby's Faker.
We are also able to sample from a model and create synthetic data, hence the name **SynDat**. The most obvious way that the use of synthetic data benefits data science is that it reduces the need to capture data from real-world events, and for this reason it becomes possible to generate data and construct a dataset much more quickly than a dataset dependent on real-world events and in addition you don't misuse data protection.



1 https://pypi.org/project/Faker/

_PIC: 101_syndat_concept.png

Now I want to show almost step by step how we can use the Faker Lib. First you had to install faker package, it can be installed with pip:

```
C:\Users\Max\AppData\Local\Programs\Python\Python36-32>
                                python -m pip install faker
```

Install a 32 bit package module in a 64 bit environment:

1. Change to your 32 bit path with cd:
   C:\Users\Max\AppData\Local\Programs\Python\Python36-32>

2. Call the Pip (e.g. faker module) explicitly with
   python.exe: python -m pip install faker

And it runs:
Downloading
https://files.pythonhosted.org/packages/27/ab/0371598513e8179d9053911e814c4de4ec2d0dd47e725dca40aa664f994c/Faker-9.9.0-py3-none-any.whl (1.2MB)..
You are using pip version 9.0.1, however version 21.3.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip'.
C:\Users\Max\AppData\Local\Programs\Python\Python36-32>
>>>>>>>>>>>

Now we start the program:
The **fake.Faker** (fake = Faker()) creates and initializes a faker generator, which can generate data by accessing properties named after the type of data, whether you need to bootstrap your database, create structured JSON documents or fill-in your storage persistence to stress test.

```
sw:= TStopWatch.Create();
sw.Start;
eg.execStr('from faker import Faker');
eg.execStr('import simplejson as json');  //# instead import json
eg.execStr('import dumper');
eg.execStr('fake = Faker()');
fprofile:= eg.evalStr('(fake.profile())')
fprofile:= StringReplace(fprofile,'\n',CRLF,[rfReplaceAll]);
```

To clean up the data, we will also replace the newlines as \n in the generated addresses with commas or CRLF, and remove the newlines from profile generated text completely.

Faker delegates the data generation to providers. The default provider uses the English locale. Faker supports other locales;

they differ in level of completion, there are lots of ways to artificially manufacture and build data, some of which are far more complex than others and models real-world distribution with descriptive statistics.
Check the output with path and list the profile dictionary, the example outputs a fake name, address, and many more items of a persons profile:

fake person profile:
{'job': 'Manufacturing engineer', 'company': 'Cunningham-Young', 'ssn': '630-62-0344', 'residence': 'PSC 1590, Box 0125
APO AA 42693', 'current_location': (Decimal('-51.8228245'), Decimal('-61.889364')), 'blood_group': 'A+', 'website': ['http://www.jones-clark.net/', 'https://www.fowler.com/'], 'username': 'garciatina', 'name': 'Roger Nichols', 'sex': 'M', 'address': '51574 Combs Alley Apt. 142, Ryanhaven, AL 82796', 'mail': 'andrea31@hotmail.com', 'birthdate': datetime.date(1914, 4, 15)}
creditcard#: 213140049750943
Stop Watch Faker Tester1: 0:0:0.636

This is not json as I first assumed, and we can convert it. I tried first with json and simplejson, got some date and decimals serialize exceptions (Object of type date is not JSON serializable.), then I used dumper lib, but got a next exception Exception: <class 'AttributeError'>: 'NoneType' object has no attribute 'write'.: So the profile is a dict type, the misleading {} trapped me first. Let's generate another avatar:

{'job': 'Nurse, adult', 'company': 'Rogers and Sons', 'ssn': '038-06-4652', 'residence': 'PSC 8856, Box 2882
APO AE 08426', 'current_location': (Decimal('16.4363075'), Decimal('-83.079826')), 'blood_group': 'A-', 'website': ['https://www.white.biz/', 'http://garrett-perez.com/'], 'username': 'xnelson', 'name': 'Ms. Colleen Bowman PhD', 'sex': 'F', 'address': '328 Reeves Estates Apt. 279
Lake Nicholas, MD 31753', 'mail': 'kkhan@yahoo.com', 'birthdate': datetime.date(1936, 6, 3)}

Oh what as surprise a nurse and she holds a PhD and works by Rogers. What if, for instance, I'm interested in generating German or Spanish names and professions of the type one would find in Netherlands, Mexico, Austria or Switzerland?

```
fake = Faker(['de_DE'])
for i in range(10):
      print(fake.name())

eg.execStr('fake = Faker(["es_MX"])')
//for i in range(10):
for it:= 1 to 10 do
  println(UTF8toAnsi(eg.evalStr('fake.name()')));
```

>>> Alma María José Montañez Dávila ...

The Faker constructor takes also a performance-related argument called use_weighting. It specifies whether to attempt to have the frequency of values match real-world frequencies and distribution shape (e.g. the English name Gary would be much more frequent than the name Welson). If use_weighting is False, then all items have an equal chance of being selected, and the selection process is much faster; the default is True.

The next line is a simple demonstration of Faker credit card:

```
   println('creditcard#: '+eg.evalStr('fake.credit_card_number()')); //}
   sw.Stop;
```

Faker also support for dummy hashes and **uuids** for SynDat:

```
#!/usr/bin/env python
from faker import Faker
faker = Faker()
print(f'md5: {faker.md5()}')
print(f'sha1: {faker.sha1()}')
print(f'sha256: {faker.sha256()}')
print(f'uuid4: {faker.uuid4()}')
```

In the end we close and free all the resources of objects, including stop-watcher **sw** and python frame **apd**:

```
  except
    eg.raiseError;
    writeln(ExceptionToString(ExceptionType, ExceptionParam));
  finally
    eg.Free;
    sw.Free;
    sw:= Nil;
    apd.position:= 100;
  end;
```

You can also run the Python Engine script at runtime to get a Faker() object and if something went wrong you got a *raiseError* Py exception. Eval() function accepts a string argument and if the string argument is an expression then eval() will evaluate the expression as a callback with return (*faker.proxy.Faker*):

```
with TPythonEngine.Create(Nil) do begin
  pythonhome:= PYHOME;
  try
    loadDLL;
    Println('Faker Platform: '+
          EvalStr('__import__("faker").Faker()'));
  except
    raiseError;
  finally
    free;
  end;
end;
```

```
>>> <faker.proxy.Faker object at 0x0CAFA850>
```

Conclusion

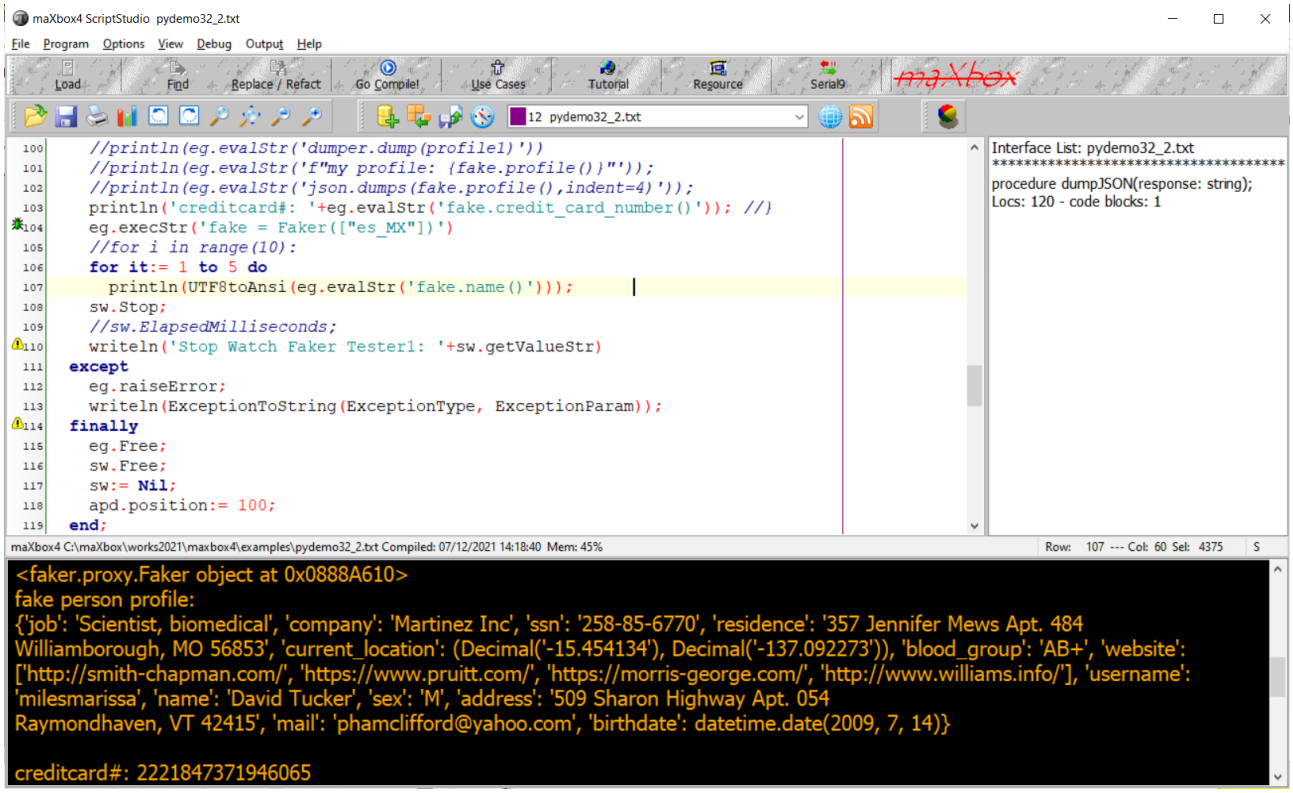
In this report, we used Python Faker to generate fake or synthetic data in Python and maXbox with measuring time behaviour.

Finally, synthetic datasets can minimize privacy concerns. Attempts to anonymize data can be ineffective, as even if sensitive/identifying variables are removed from the dataset, other variables can act as identifiers when they are combined. This isn't an issue with synthetic data, as it was never based on a real person, or real event, in the first place.

A concept could mean, firms, institutes or simply users don't deal with real person data, they got an avatar which makes an relationship between a hash and a guid in a worldwide proxy block-chain (pb1). A real person is protected behind the SynDat proxy with a guid record.

Python for .NET is also a package that gives Python programmers nearly seamless integration with the .NET Common Language Runtime (CLR) and provides a powerful application scripting tool for .NET developers and with Delphi or Lazarus just found that:

https://i2.wp.com/blogs.embarcadero.com/wp-content/uploads/2021/07/demo01_Faker2-2809487.png?ssl=1



_PIC: 101_syndat_gui_profile.png

SynDat topics and script:

- https://pypi.org/project/Faker/
- https://www.kdnuggets.com/2021/11/easy-synthetic-data-python-faker.html
- http://www.softwareschule.ch/examples/pydemo32_2.txt
-
- https://www.unite.ai/what-is-synthetic-data/
- http://www.softwareschule.ch/examples/cheatsheetpython.pdf

```
****************************************************************
Release Notes maXbox 4.7.6.10 II November 2021 mX476
****************************************************************
Add 10 Units + 3 Tutorials
1441 unit uPSI_neuralgeneric.pas; CAI
1442 unit uPSI_neuralthread.pas; CAI
1443 unit uPSI_uSysTools; TuO
1444 unit upsi_neuralsets; mX4
1445 unit uPSI_uWinNT.pas mX4
1446 unit uPSI_URungeKutta4.pas ICS
1447 unit uPSI_UrlConIcs.pas ICS
1448 unit uPSI_OverbyteIcsUtils.pas ICS
1449 unit uPSI_Numedit2 mX4
1450 unit uPSI_PsAPI_3.pas mX4
Total of Function Calls: 35078
SHA1: of 4.7.6.10  D4B0A36E42E9E89642A140CCEE2B7CCDDE3D041A
CRC32: B8F2450F 30.6 MB (32,101,704 bytes)
```