embarcadero®

PyTorch for Delphi

with the

Python **Data Science** Libraries

June 9th, 2022

jim.mckeeth@embarcadero.com

Slides, links, replay: https://blogs.embarcadero.com/?p=145025

Description

PyTorch for Delphi with the Python Data Sciences Libraries

The next installment of the Embarcadero Open Source Live Stream takes a look at the Delphi side of the Python Ecosystem with the new Python Data Sciences Libraries and related projects that make it super easy write Delphi code against Python libraries and easily deploy on Windows, Linux, MacOS, and Android. Specific examples with the Python Natural Language Toolkit and PyTorch, the library that powers projects like Tesla Autopilot, Uber's Pyro, Hugging Face's Transformers.

This is part of a series of regular live streams discussing the latest in Embarcadero open source projects. Hosted by Jim McKeeth and joined by members of the community and developers involved in these open source projects, as well as members of Embarcadero and Idera’s Product Management. A great opportunity to see behind the scenes and help shape the future of Embarcadero’s Open Source projects.

If you are interested in machine learning, artificial intelligence, or data sciences then you want to join this webinar!

Slides, replay, links, etc. blogs.embarcadero.com/?p=145025

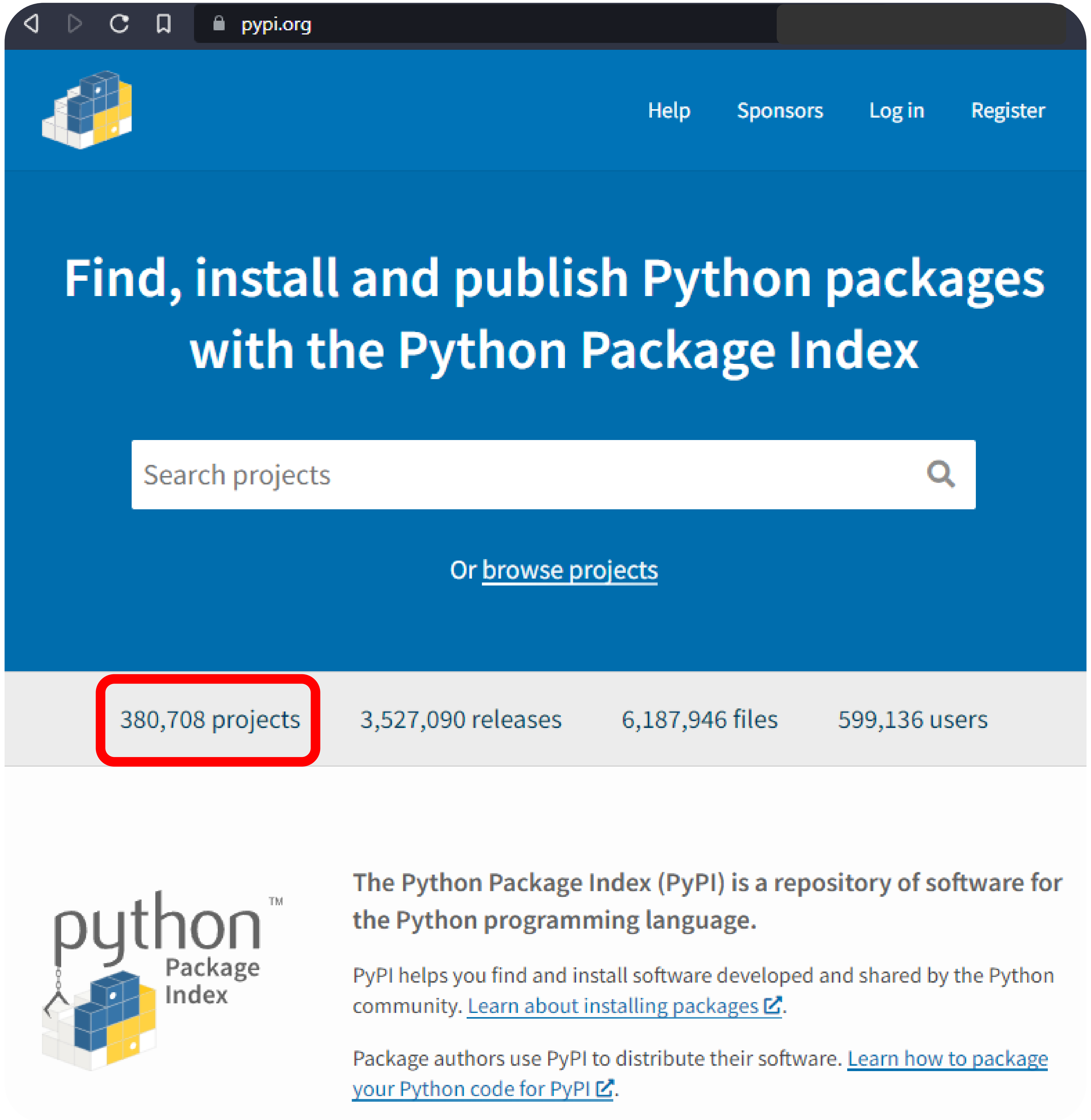
Who’s on the Stream?

- Jim McKeeth - Embarcadero Developer Relations
- Kunarapu Priyatham - Python Developer & Tech Writer
- Lucas Moura Belo - Delphi & Python Developer
- Kiriakos Vlahos - Maintainer of PyScripter and Python4Delphi
- Chris Gardner - Whole Tomato Visual Assist Lead Developer
- Marco Cantu - RAD Studio & Delphi PM
- *And others who may or may not chime in . . .*

We Need More Components!

- We are more productive when there is an existing component or library already
- Delphi and C++Builder has a huge rich ecosystem of Tech Partners and open source
- But wouldn't be great to have ***EVEN MORE?***
- Python has one of the richest ecosystems, especially for data sciences, AI, & ML
- What if Delphi & C++Builder could easily

USE THEM ALL?



Find, install and publish Python packages with the Python Package Index

Or browse projects

380,708 projects 3,527,090 releases 6,187,946 files 599,136 users

The Python Package Index (PyPI) is a repository of software for the Python programming language.

PyPI helps you find and install software developed and shared by the Python community. Learn about installing packages.

Package authors use PyPI to distribute their software. Learn how to package your Python code for PyPI.

XKCD on Python

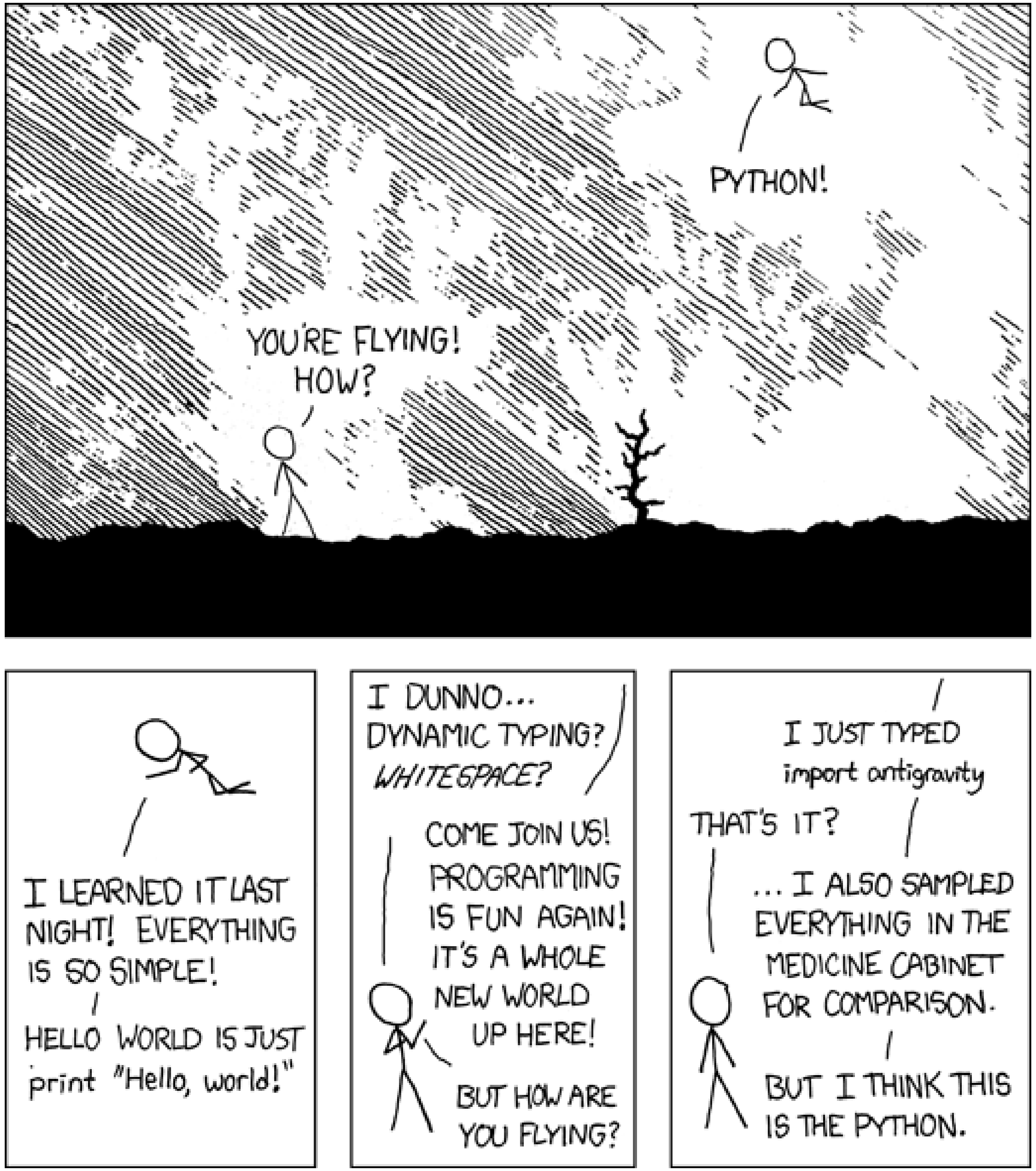
There is a package for everything!

- xkcd.com/353/ from December 5th, 2007
- explainxkcd.com/wiki/index.php/353:_Python

[A friend is talking to Cueball, who is floating in the sky.]

Friend: You're flying! How?
Cueball: Python!
Cueball: I learned it last night! Everything is so simple!
Cueball: Hello world is just `print "Hello, World!"`
Friend: I dunno... Dynamic typing? Whitespace?
Cueball: Come join us! Programming is fun again! It's a whole new world up here!
Friend: But how are you flying?
Cueball: I just typed '`import antigravity`'
Friend: That's it?
Cueball: ...I also sampled everything in the medicine cabinet for comparison.
Cueball: But I think this is the python.

In response to this comic, the Python developers implemented the module `antigravity`. When you import it, the default web browser will open this comic.



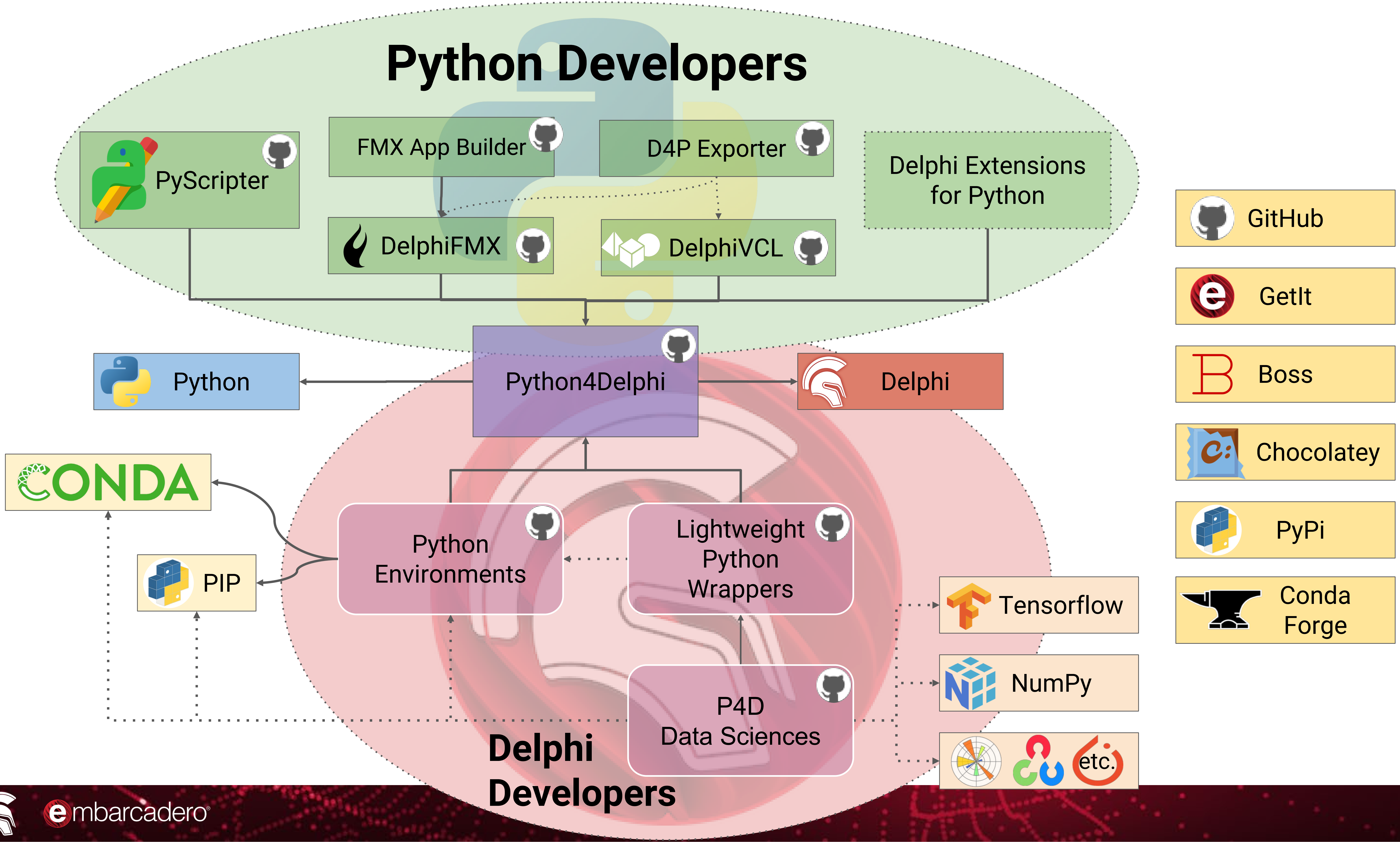
YOU'RE FLYING! HOW?
PYTHON!
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

General Notes and Comments

- Most all projects are open source, MIT licensed, and hosted on GitHub.
- Embarcadero is directly sponsoring the developers and development.
- While the focus is on Delphi, we expect C++Builder will *work eventually*.
  - Once the Delphi side is stable we will look more closely at C++Builder where it makes sense.
  - There are already resources showing how to use Python4Delphi with C++Builder.
- All of these projects are available and mostly working today.
  - They are under active development, so consider them beta or alpha releases….
- Unless otherwise stated, everything works on
  - Android, Windows (32 & 64-bit), MacOS (x86 & ARM), and Linux
- We are looking for feedback, pull requests, contributors, and forks on all Embarcadero Open Source projects!
- ★ Starring a project is a great way to show your interest!

It’s not a Competition

- Language choice isn’t a *zero sum game*.
- Developers have multiple tools on their workbench.
- When tools work together then a developer can use both!
- It is about finding the right tool for each task.
- We use SQL to talk to databases, JavaScript and HTML run the web.
- Having specialized tools for different tasks doesn’t detract from *favorite* tools.
- You can always find a specific task that another tool is better for, but no one tool is that best for *all tasks*.



Python Developers

PyScripter

FMX App Builder

D4P Exporter

Delphi Extensions for Python

DelphiFMX

DelphiVCL

Python

Python4Delphi

Delphi

CONDA

PIP

Python Environments

Lightweight Python Wrappers

P4D Data Sciences

Tensorflow

NumPy

etc.

Delphi Developers

GitHub

GetIt

Boss

Chocolatey

PyPi

Conda Forge

Python and Delphi Ecosystem

1. Free Python IDE and libraries
   - Free give away to Python developers.

PyScripter, DelphiFMX, DelphiVCL

2. Use designer export from Delphi
   - Dipping toe into Delphi CE or a free trial.

Available via GetIt.

3. Import Python libraries into Delphi
   - Starting to explore the power of hybrid apps.

Lightweight Python Wrappers & Data Sciences

4. Build native Delphi modules for Python
   - Use Delphi’s native compiler to add speed and features to Python.

Python4Delphi & Delphi Extensions for Python

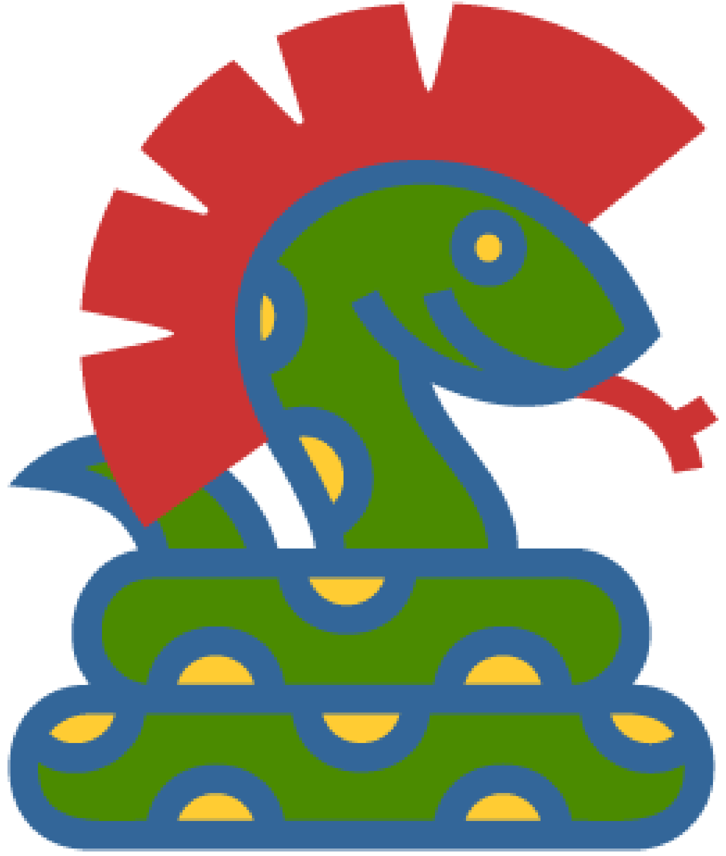
5. Full Hybrid Delphi + Python Power
   - The ultimate stage of development, ushering in the final technological singularity, leading to total universal domination and the enlightenment of all humankind.

Python4Delphi: The Bridge

Embarcadero’s Fork of P4D
github.com/Embarcadero/Python4Delphi
Originally maintained by Kiriakos Vlahos (aka PyScripter)

Python for Delphi (P4D) is a set of components that wrap up the Python Library into Delphi. They let Delphi developers easily execute Python scripts, create new Python modules, and new Python types. You can create Python extension modules and much more. P4D provides different levels of functionality:

- Low-level access to the Python API
- High-level bi-directional interaction with Python
- Access to Python objects using Delphi custom variants (VarPyth.pas)
- Wrapping of Delphi objects for use in Python scripts using RTTI (WrapDelphi.pas)
- Creating Python extension modules with Delphi classes and functions



Supports Android (64-bit ARM), Windows (32 & 64-bit x86), MacOS (x86 & ARM), and Linux (64-bit x86)

Other Pieces in the Ecosystem

- **GitHub** - Where all the projects are hosted: github.com/embarcadero
- **GetIt** - The Embarcadero package manager. When the projects for Delphi developers are ready for general use they can easily be installed via GetIt. getitnow.embarcadero.com
- **Boss** - Delphi dependency manager. Plan to use to make it easy to manage dependencies between projects. github.com/hashload/boss
- **Chocolatey** - General Windows package manager. Simplify deployment and installation of development builds.
- **pip** - Package Installer for Python pip.pypa.io
- **PyPi** - The Python Package Index is the official third-party software repository for Python. Default source for **pip**. pypi.org/user/Embarcadero
- **Conda** - Alternative and very popular Python package manager. docs.conda.io
- **Conda Forge** - Community-led collection of distributions for the conda package manager. conda-forge.org

GitHub

GetIt

Boss

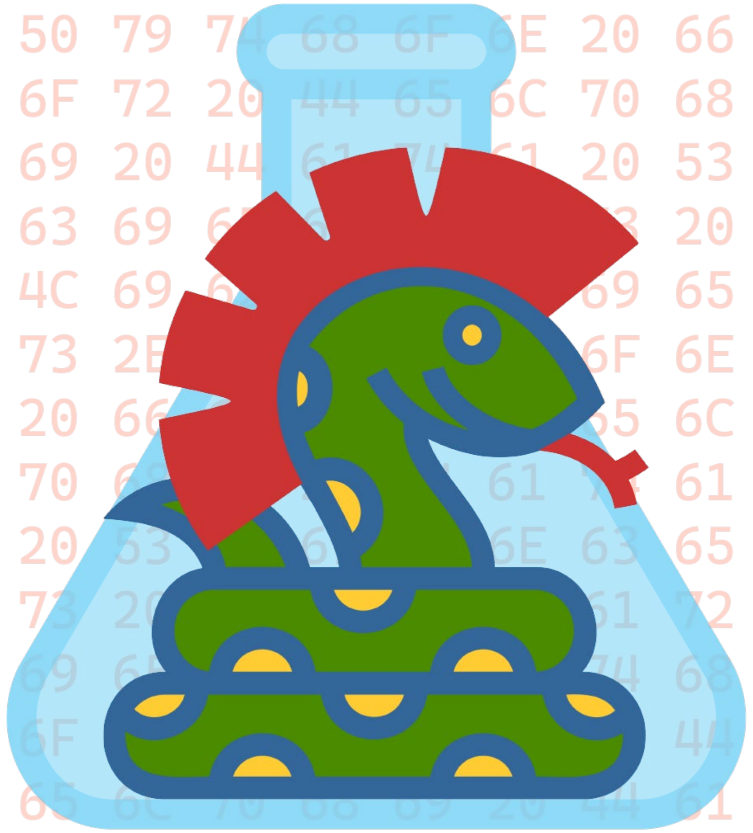
Chocolatey

pip

PyPi

CONDA

Conda Forge



For Delphi Developers

Bringing the best of Python to Delphi developers.

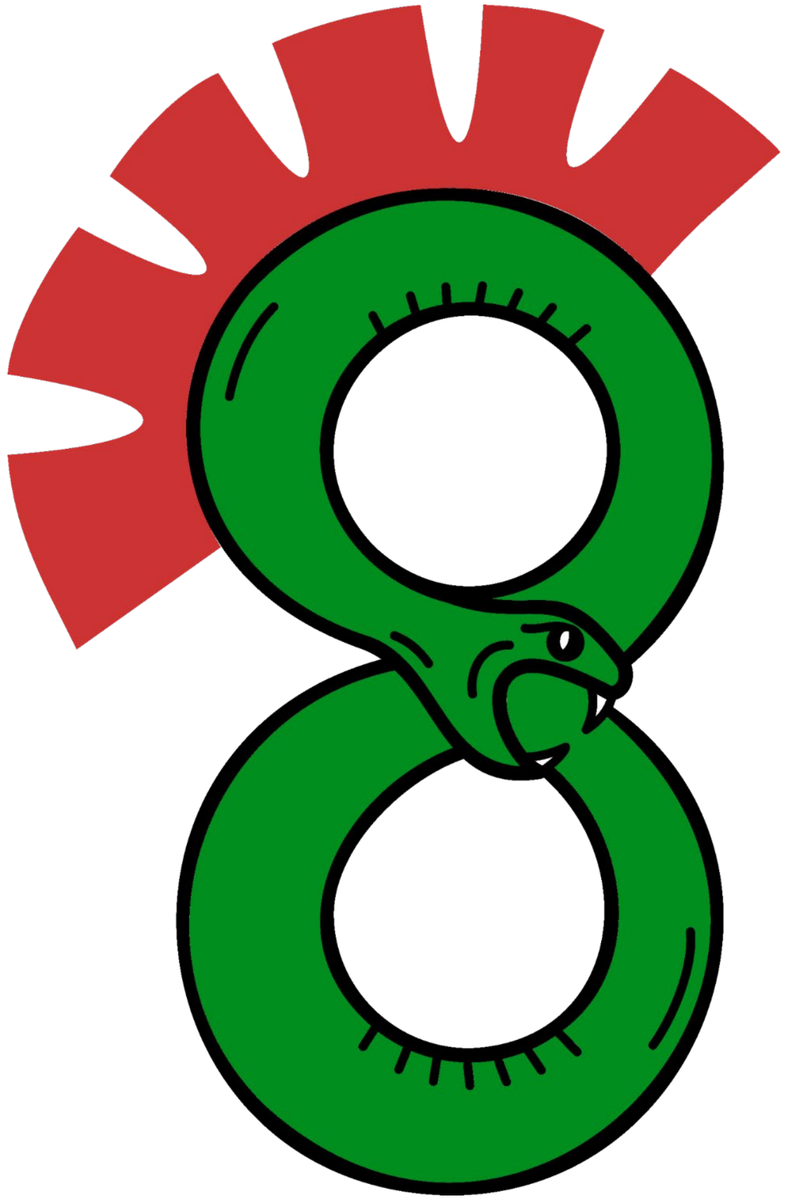
Pure Object Pascal access to Python libraries.

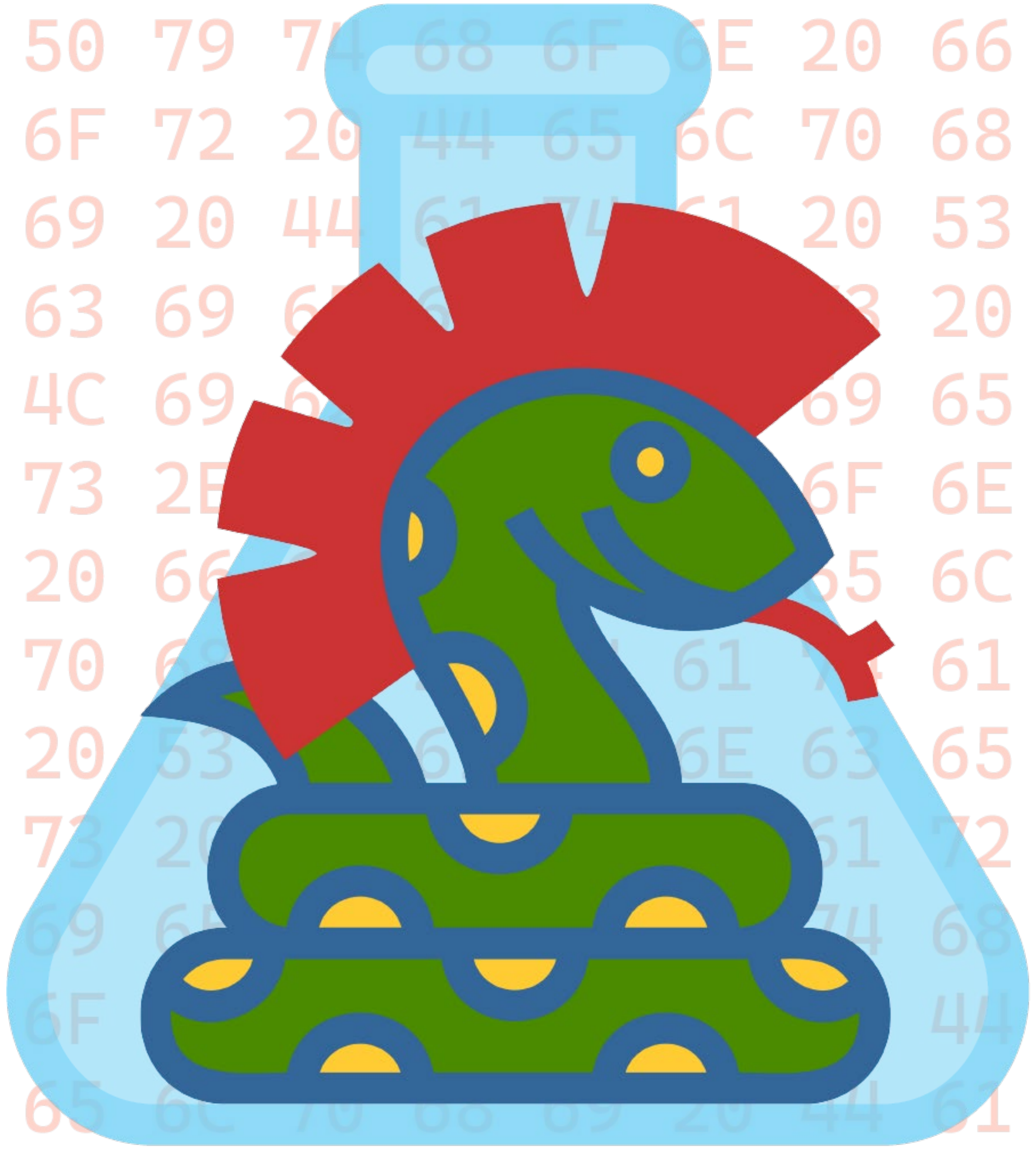
Goals

- Give Delphi (and eventually C++Builder) developers access to all Python libraries
- No Python knowledge or code required
- But still give full access
- Simple development setup
- Simple end user deployment
- No internet connection required
- Cross platform

This is an early access preview, but you are invited to start using and contributing.

Three All New Projects





Powered by Python4Delphi

Python Environments

Early Alpha - Should mostly work, still adding features
github.com/Embarcadero/PythonEnviroments/

- Delphi components that simplify the deployment and configuration of Python environments
- *Freeing Delphi developers from needing to worry about Python*
- Supports Python versions 3.7, 3.8, 3.9, & 3.10
- Currently 3 different options:
  - Use an existing Python environment
  - Redistribute or download a Python environment archive
  - Embed the redistributable within the program
- At runtime it automatically verifies and configures the environment, including PIP or Conda package managers
- The framework is very modular, flexible, and customizable
- Installs necessary Python modules
- Working on Android, Windows (32 & 64-bit), MacOS (x86 & ARM), and Linux

For Delphi Developers

Lightweight Python Wrappers

Early Alpha - Should mostly work, still adding features
github.com/Embarcadero/Lightweight-Python-Wrappers

- Framework for creating a lightweight wrapper around a Python library turning it into a Delphi component
- Makes adding a new Python library super quick and easy
- Manages the necessary Python modules via `pip` or `Conda`
- Allows for a simple translation of Python code into Delphi code
- Uses late binding giving it a more flexible syntax
- Letting Delphi developers use pure Delphi to access Python libraries

To Do:

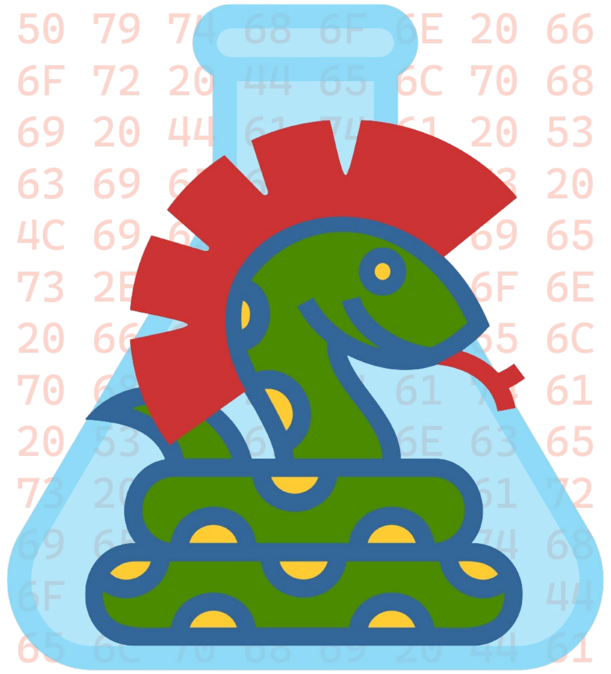
- *Automatic wrapper generation*
- *Python to Delphi code translation*
- *Code completion and better IDE integration*
- *Your suggestions!*

For Delphi
Developers

P4D Data Sciences

Early Alpha - Should mostly work, still adding features
github.com/Embarcadero/P4D-Data-Sciences
Even includes some demos

- Based on the Lightweight Python Wrappers, Python Environments, and Python4Delphi
- Delphi components for popular Python Data Sciences tools
- Currently includes:
  - Tensorflow, NumPy, PyTorch Vision, PyTorch, MatPlotLib, Natural Language Toolkit (NLTK), OpenCV, and Scikit-learn
- Even includes some samples
- Surprisingly easy to make and use
  - Literally a couple hours work to add and test a new library
- *Looking at additional libraries to add….*



For Delphi Developers

Example of adding a library:
Natural Language Toolkit (NLTK)

Natural Language Toolkit (NLTK)

- This is the code to wrap a new Python library for Delphi
- Looking into automatic importer
- All the details and complexity are encapsulated in the *Lightweight Python Wrappers*
- The *Python Environment* handles all the requirements and installation

```
unit NLTK;

interface

uses System.Classes, PyPackage, PyPackage.Model, PythonEngine;

type
  [ComponentPlatforms(pidAllPlatforms)]
  TNLTK = class(TPyManagedPackage)
  private function AsVariant: variant;
  protected
    procedure Prepare(const AModel: TPyPackageModel);
override;
  public property nltk: variant read AsVariant;
  end;

implementation

uses PyPackage.Manager.ManagerKind, PyPackage.Manager.Pip;

function TNLTK.AsVariant: variant;
begin
  Result := inherited;
end;

procedure TNLTK.Prepare( const AModel: TPyPackageModel);
begin
  inherited ;
  with AModel do begin
    PackageName := 'nltk' ;
    PackageManagers.Add(TPyPackageManagerKind. pip ,
      TPyPackageManagerPip.Create( 'nltk' ));
  end;
end;
```

Boilerplate Code

- Name of Python module
- Package manager for installation

For Delphi Developers

NLTK Demo

- Simple code found on nltk.org
- Showing off basic functionality
- Python code running in the REPL, with results displayed inline

Downloads the required data. Only necessary once.

Tokenize and tag some text:

```
>>> import nltk
>>> nltk.download('popular')
>>> sentence = """At eight o'clock on Thursday morning
... Arthur didn't feel very good."""
>>> tokens = nltk.word_tokenize(sentence)
>>> tokens
['At', 'eight', "o'clock", 'on', 'Thursday', 'morning',
'Arthur', 'did', "n't", 'feel', 'very', 'good', '.']
>>> tagged = nltk.pos_tag(tokens)
>>> tagged[0:6]
[('At', 'IN'), ('eight', 'CD'), ("o'clock", 'JJ'), ('on', 'IN'),
('Thursday', 'NNP'), ('morning', 'NN')]
```

Identify named entities:

```
>>> entities = nltk.chunk.ne_chunk(tagged)
>>> entities
Tree('S', [('At', 'IN'), ('eight', 'CD'), ("o'clock", 'JJ'),
           ('on', 'IN'), ('Thursday', 'NNP'), ('morning', 'NN'),
       Tree('PERSON', [('Arthur', 'NNP')]),
           ('did', 'VBD'), ("n't", 'RB'), ('feel', 'VB'),
             ('very', 'RB'), ('good', 'JJ'), ('.', '.')])
```

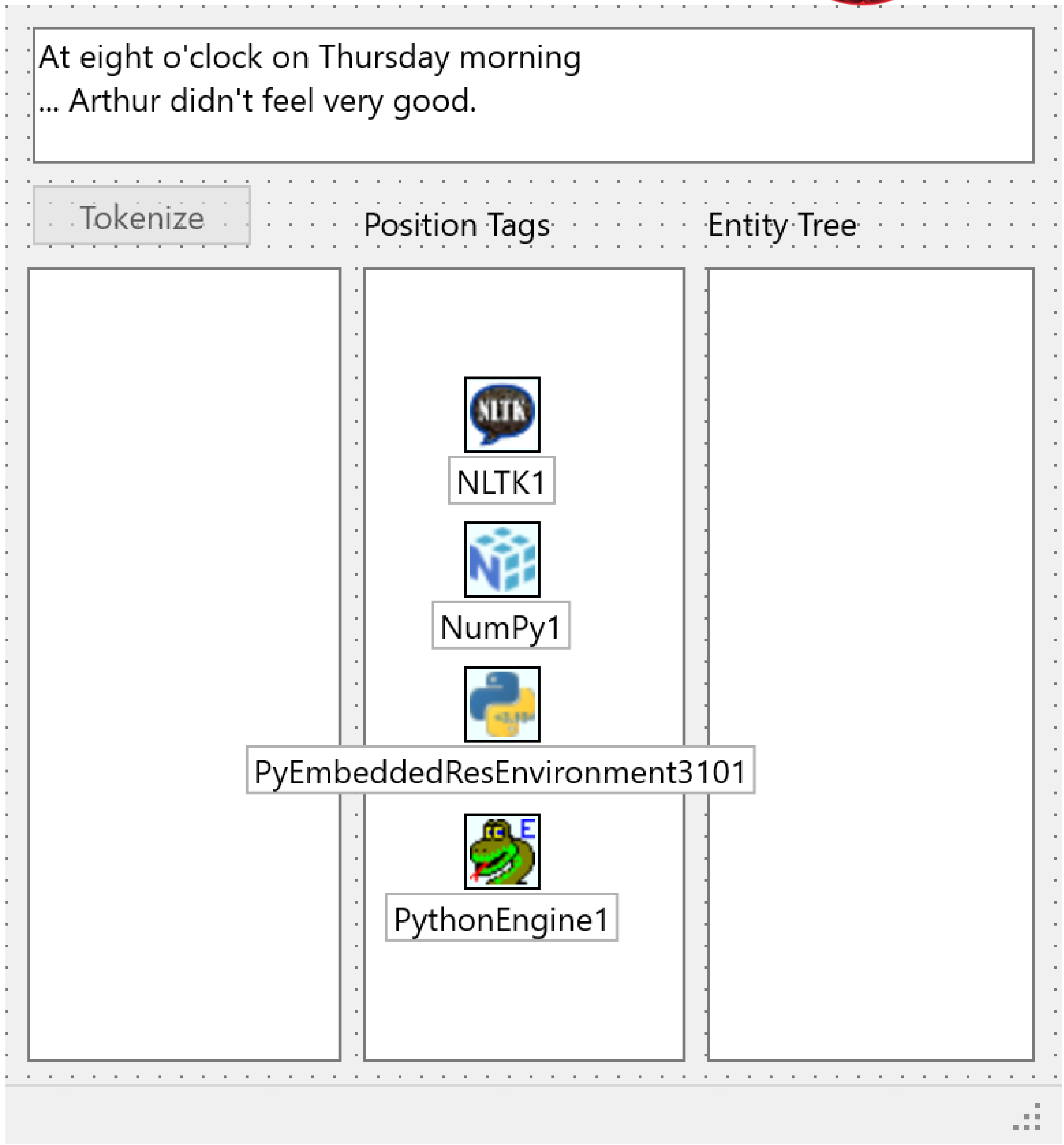
NLTK Demo

- There are a series of event handlers notifying your program of state of the Python Environment.
- Here is the Delphi version of the Python code.
- The output of each step is displayed in the list boxes.
- When compiled there is a single EXE.
- At runtime the Environment sets up a nested Python environment with all the necessary dependencies.

```
with NLTK1 do begin
   nltk.download('popular');

   var tokens :=
     nltk.word_tokenize(memo1.lines.Text);
   VarPyToStrings(tokens,ListBox1.Items);

   var tagged :=
     nltk.pos_tag(tokens);
   VarPyToStrings(tagged,ListBox2.Items);

   var entities :=
     nltk.chunk.ne_chunk(tagged);
   VarPyToStrings(entities, ListBox3.Items);
end;
```

NLTK Demo

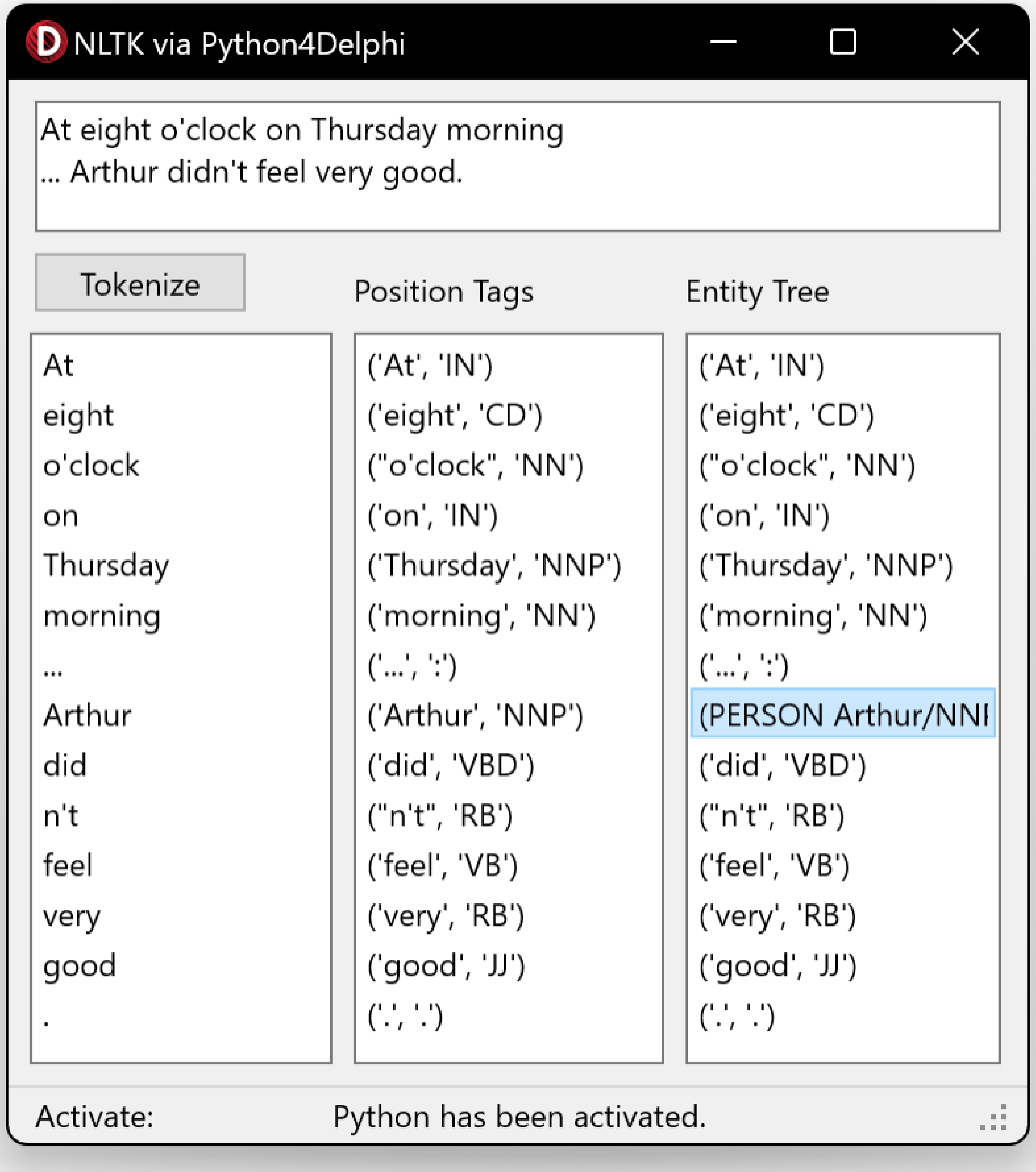
- Add the following components:
  - TPythonEngine
  - TPyEmbeddedResEnvironment310
  - TNumPy
  - TNLTK
- Connect the components
- Make the UI
- Leave the Tokenize button disabled, then use an event handler from the Environment to enable it when everything is setup.



At eight o'clock on Thursday morning
... Arthur didn't feel very good.

NLTK Demo

- One first run the Environment is setup with all dependencies. Usually takes a few seconds.
- On subsequent runs it takes a second to validate the environment, load the libraries, and start the engine.
- The program uses the event handlers to enable the user interface based on the state.



At eight o'clock on Thursday morning
... Arthur didn't feel very good.

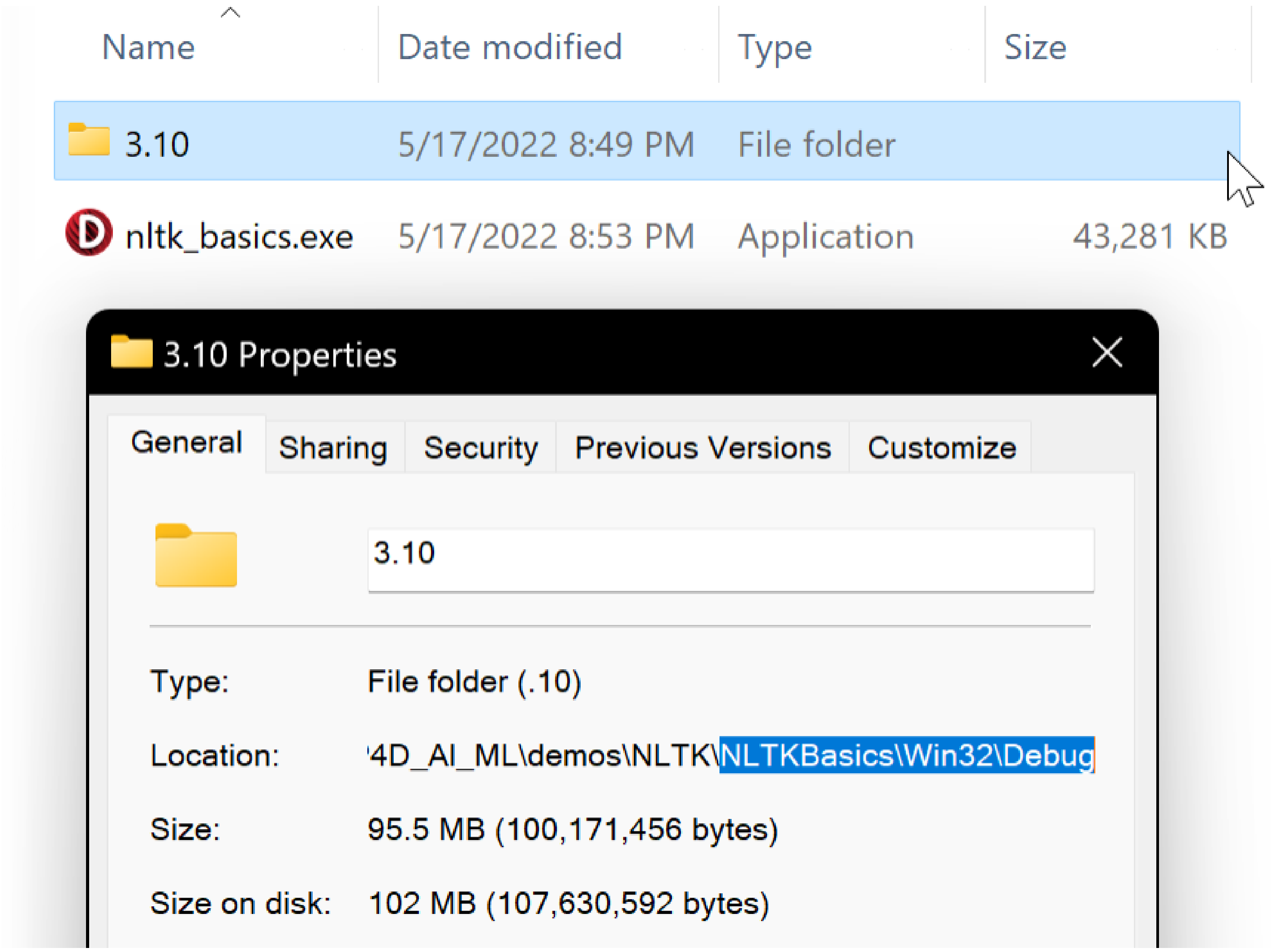
	Position Tags	Entity Tree
At	('At', 'IN')	('At', 'IN')
eight	('eight', 'CD')	('eight', 'CD')
o'clock	("o'clock", 'NN')	("o'clock", 'NN')
on	('on', 'IN')	('on', 'IN')
Thursday	('Thursday', 'NNP')	('Thursday', 'NNP')
morning	('morning', 'NN')	('morning', 'NN')
...	('...', ':')	('...', ':')
Arthur	('Arthur', 'NNP')	(PERSON Arthur/NNI
did	('did', 'VBD')	('did', 'VBD')
n't	("n't", 'RB')	("n't", 'RB')
feel	('feel', 'VB')	('feel', 'VB')
very	('very', 'RB')	('very', 'RB')
good	('good', 'JJ')	('good', 'JJ')
.	('.', '.')	('.', '.')

Activate: Python has been activated.

NLTK Demo

For Delphi Developers

- The Python environment can be installed anywhere (or an existing one can be used) but the default is the EXE folder.
- It can also be created during the installation process, before the program first runs.
- Here we see our sample program in 43 MB.
- And it extracted and created a 95 MB Python environment.
- The same project can compile and run identically on Android, Windows (32 & 64-bit), MacOS (x86 & ARM), and Linux



Name	Date modified	Type	Size
3.10	5/17/2022 8:49 PM	File folder	
nltk_basics.exe	5/17/2022 8:53 PM	Application	43,281 KB

3.10

Type:	File folder (.10)
Location:	'4D_AI_ML\demos\NLTK\NLTKBasics\Win32\Debug
Size:	95.5 MB (100,171,456 bytes)
Size on disk:	102 MB (107,630,592 bytes)

PyTorch

Demo with Delphi

For Delphi
Developers

PyTorch Demo

- PyTorch is an open source machine learning framework based on the Torch library
- Used for applications such as computer vision and natural language processing
- Primarily developed by Facebook's AI Research lab (FAIR)
- Used in projects like Tesla Autopilot, Uber's Pyro, Hugging Face's Transformers, and others
- Delphi demo is part of P4D Data Sciences
    - This is a Thumbs Up and Thumbs Down image classification system with mobile image collection
    - github.com/Embarcadero/P4D-Data-Sciences/tree/main/demos/PyTorch/
        - App - Android App Source
        - Install - Installs dependencies and sets up environment on server
        - Server - WebBroker server
        - Train - Called by server to do the training and process the data
        - Training_Data - Pre collected images

Uber

What’s Next?

It’s still early access, but
what else is possible?

What’s Next

- Still more work - in early stages of development
- Automatic wrapper generation
- More libraries
- Python to Delphi code translation
    - Translate Python sample code
- Documentation for the full stack and for using the imported libraries
- Code completion and better IDE integration
- Your suggestions!

Other Potential Python Libraries

For Delphi Developers

- **Keras** - ML library that runs on both CPU and GPU
- **LightGBM by Microsoft** - Used for ML ranking, classification, etc.
- **Aesara** - Multi-dimensional arrays mathematical expressions
- **BeautifulSoup** - Scrapes data from web pages
- **Pillow** - Image manipulation
- **MoviePy** - Video Editing
- **Seaborn** - Drawing attractive statistical graphics
- ***YOLOv5** - Pretrained object detection architectures and models*

Keras
keras.io

LightGBM
lightgbm.readthedocs.io

aesara
aesara.readthedocs.io

crummy.com/software/BeautifulSoup/
BeautifulSoup

Python Pillow
python-pillow.org

MoviePy
zulko.github.io/moviepy

seaborn
seaborn.pydata.org

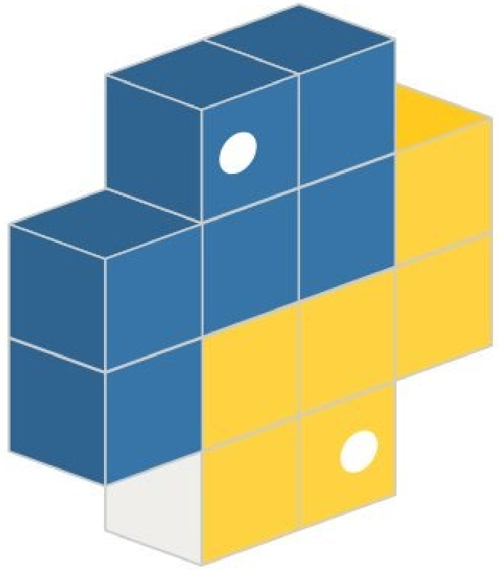
Tip of the Iceberg

- See the list of the most popular Python packages on PyPi in the last 30 days.
- None of these are on the previous slides.
- Can every Python library be available for Delphi developers? One way to find out….

pypistats.org/top - pypi.org

Rank	Library	Downloads
1	boto3	343,267,820
2	urllib3	214,453,654
3	botocore	211,027,562
4	requests	206,502,136
5	idna	176,641,902
6	setuptools	173,382,430
7	s3transfer	171,067,399
8	typing-extensions	165,570,269
9	six	156,492,192
10	certifi	151,478,041
11	python-dateutil	149,953,838
12	pyyaml	142,509,567
13	charset-normalizer	139,528,384
14	awscli	122,370,481
15	click	117,707,552
16	wheel	115,961,643
17	cryptography	110,978,677
18	jinja2	109,342,363
19	pyparsing	103,909,140
20	rsa	103,612,934

For Delphi Developers



We Need Your Help!

- Try the projects
- Star the projects on GitHub
- File bug reports
- Make a fork
- Issue a pull request
- Add features and fix bugs
- Tell your friends
- Write a blog post
- Make a demo

Q & A

General Notes and Comments

- Most all projects are open source, MIT licensed, and hosted on GitHub.
- Embarcadero is directly sponsoring the developers and development.
- While the focus is on Delphi, we expect C++Builder will *work eventually*.
    - Once the Delphi side is stable we will look more closely at C++Builder where it makes sense.
    - There are already resources showing how to use Python4Delphi with C++Builder.
- All of these projects are available and mostly working today.
    - They are under active development, so consider them beta or alpha releases….
- Unless otherwise stated, everything works on
    - Android, Windows (32 & 64-bit), MacOS (x86 & ARM), and Linux
- We are looking for feedback, pull requests, contributors, and forks on all Embarcadero Open Source projects!
- ★ Starring a project is a great way to show your interest!

embarcadero®

PyTorch for Delphi
with the
Python Data Science Libraries