PyHEP.dev 2024 - ”Python in HEP” Developer’s Workshop

Report of Contributions

Contribution ID: **1** Type: **not specified**

Introducing Immortal Objects: building block towards a multi-core Python runtime.

In this presentation, I'll cover memory management in Python starting from the fundamentals. I'll explain the rationale behind the need for PEP 683: "–Immortal Objects, Using a Fixed Refcount" , discussing about this change may be unlocking exciting avenues for true parallelism in python.

Objective of talk is to discuss on how and why we need pure immutable objects and fixing the old time issue with them can make Python more powerful‼ in terms of code pure parallel execution. Certain objects such as 'None', 'True', and 'False' act as global singletons, shared across the interpreter instead of creating fresh copies each time. However there is a catch even though they are considered immutable these objects were not out of editing their reference count for any new reference or de reference.

The takeaway of this presentation would be :
a) Refresher on object(mutable vs immutable and their significance)
b) Refresher on Life cycle of a python object.
c) Refresher on python memory management with garbage collection
d) How overcoming with pseudo immutable object will be unlocking exciting avenues for true parallelism in Python.

The target audience would be intermediate and advanced Pythonista's. However I will present the information in simple and subtle manner for better understanding of python newbies .

Timeline of a talk :
Introduction(1 min)
What is a python object(3 min)
Life cycle of a python object(3 min)
Python object Runtime state(3 min)
Explain Garbage collection based on reference count(4 min)
Motivation for PEP-0683(3 min)
implementation summary for PEP-0683(4 min)

for you reference: This is the article I wrote on the same topic in my weekly newsletter:
https://technikal.substack.com/p/how-instagram-introduction-immortal

Primary author: MEHRA, Aditya

Presenter: MEHRA, Aditya

Contribution ID: **2** Type: **not specified**

Unleashing the Power of Python Decorators: A Hands-On Workshop

-Workshop Overview:
Dive deep into the world of Python decorators with our hands-on workshop designed for intermediate Python developers. This session will demystify the concept of decorators, showcasing their power and versatility in enhancing the functionality of your code. By the end of this workshop, you'll have a solid understanding of how to create, apply, and leverage decorators to write more elegant and efficient Python programs.

-What You Will Learn:
Introduction to Decorators: Understand the fundamentals of decorators, including their syntax and use cases.
Building Blocks: Learn how functions and closures work in Python as a foundation for creating decorators.
Creating Simple Decorators: Step-by-step guidance on writing basic decorators to modify and enhance existing functions.
Practical Applications: Discover real-world applications of decorators, such as logging, access control, memoization, and more.
Debugging and Testing: Tips and techniques for debugging and testing decorated functions to ensure reliability and performance.

-Prerequisites:
Basic to intermediate knowledge of Python programming
Familiarity with functions and basic object-oriented programming in Python
A laptop with Python installed (preferably version 3.x)

-Audience:
This workshop is ideal for intermediate Python developers who have a solid grasp of Python programming and are looking to expand their skill set with advanced concepts. If you're familiar with functions, basic OOP, and have some experience with Python modules, this workshop is perfect for you.

Primary author: MEHRA, Aditya

Presenter: MEHRA, Aditya

Contribution ID: **3** Type: **not specified**

Blazing Speed and Efficiency in Data Analytics with DuckDB and Python

Join us for an in-depth exploration of how DuckDB, a state-of-the-art database management system, can be integrated with Python to revolutionize your data analytics workflow. This talk will demonstrate how DuckDB combines the power of traditional databases with the simplicity and efficiency of dataframes, providing a seamless experience for data scientists and analysts.

In this technical talk , I will be talking about why and How DuckDB can make life easier for a Data Scientist and increase Speed and Efficiency in Data Analytics manifold.

–We will discussing about following data points:

1. Challenges with Common Data management systems:
   Complex Setup and Maintenance: Systems like Postgres and Spark are difficult to set up and maintain.
   Data Transfer Issues: Transferring data into and out of these systems can be cumbersome.
   Integration Difficulties: Integrating these systems into Python workflows is challenging.
2. for overcoming challenges with Data management systems Data Scientists and community responded as below:
   A) Development of Custom Tools: Data scientists have created their own tools, such as Pandas and Polars, to address these challenges.
   B) Ease of Use: These tools are more intuitive and natural for data scientists to use.

However There is a Limitations of Custom Tools like Pandas and Polars as below:
A) Data Processing Capacity: Pandas and Polars are limited in the amount of data they can handle efficiently.
B) Lack of Automatic Optimization: These tools do not offer the same level of automatic optimization found in traditional data management systems.

I will be explaining breifly about Key features of DuckDB.
Such as:
A) Fast analytical queries: DuckDB runs on a columnar-vectorized query engine, which helps to make efficient use of the CPU cache and speed up response times for analytical query workloads.

B) Supports SQL and integration with Python and other programming languages: DuckDB enables users to run complex SQL queries and provides APIs for Python and other languages.

C) DuckDB has no external dependencies, so you don't have to worry about dependency issues.

And much more…..

Key Takeaways:
A) Introduction to DuckDB: Learn about DuckDB's architecture, key features, and why it stands out among other database management systems.
B) Integration with Python: Discover how DuckDB integrates deeply with Python, allowing for efficient data manipulation and analysis.
C) Dataframe Compatibility: See how DuckDB works with popular Python dataframe libraries such as Pandas, Polars, and Apache Arrow.
D) Performance Benefits: Understand the performance advantages of using DuckDB for querying and data processing compared to traditional methods.
E) Practical Applications: Explore real-world use cases and examples demonstrating how DuckDB can be used for various data analytics tasks.

Target Audience:
A) Data Scientists and Data Engineers: Seeking efficient, scalable data management and processing tools that integrate seamlessly with Python.

B) Analysts and Business Analysts: Working with large datasets, requiring intuitive tools for data wrangling and analysis to drive decision-making.

C) Python Developers: Wanting to enhance their projects with advanced data manipulation and querying capabilities.

D) Researchers and Academics: Needing powerful, user-friendly tools for data analysis and complex queries without extensive setup.

Primary author: MEHRA, Aditya

Presenter: MEHRA, Aditya

Contribution ID: **4** Type: **not specified**

Sightseeing Tour

Thursday, August 29, 2024 3:00 PM (2 hours)

Contribution ID: **5** Type: **not specified**

Workshop Dinner

Wednesday, August 28, 2024 7:00 PM (2h 30m)

Contribution ID: **6** Type: **not specified**

Registration & Reception

Contribution ID: **7** Type: **not specified**

Group photo

Tuesday, August 27, 2024 10:30 AM (30 minutes)

Contribution ID: **8** Type: **not specified**

Welcome slides

Monday, August 26, 2024 9:00 AM (15 minutes)

Contribution ID: **9** Type: **not specified**

Organizing the paper-writing

Friday, August 30, 2024 9:00 AM (15 minutes)

Contribution ID: **10** Type: **not specified**

Organizing the paper-writing

Friday, August 30, 2024 11:00 AM (15 minutes)

Contribution ID: **11** Type: **not specified**

Close-out

Friday, August 30, 2024 12:15 PM (15 minutes)

Contribution ID: **12** Type: **not specified**

Bridging Python and Julia for Enhanced Data Analysis

Let's discuss the exciting world of combining Python and Julia for data analysis for high-energy physics (HEP) and other data-intensive fields.

We'll kick things off with a quick overview of why Python is so popular for data analysis and introduce Julia, which is making waves with its incredible performance and suitability for scientific computing.

Next, I'll show you how we can get the best of both worlds. We'll talk about using PythonCall to bring Python functions and libraries into Julia and how we can embed Julia code right into our Python scripts using JuliaCall. It's easier than you might think!

I'll walk you through some practical examples where mixing Python and Julia really shines. We'll look at real-world scenarios and see how this combination can speed up our data analysis and make our work more efficient.

Of course, there are always some bumps in the road, so I'll share some common challenges you might face and how to overcome them. We'll cover best practices for managing dependencies and keeping everything running smoothly.

Finally, we'll look ahead to the future. There's so much potential for deeper integration and community-driven innovation. I hope to inspire you to explore these possibilities and collaborate with other developers to push the boundaries of what's possible.

By the end of this talk, you'll have a good grasp of how to mix Python and Julia in your projects and leverage the strengths of both languages to supercharge your data analysis.

Primary authors: OSBORNE, Ianna (Princeton University); LING, Jerry ☒ (Harvard University (US)); PIVARSKI, Jim (Princeton University)

Presenter: OSBORNE, Ianna (Princeton University)

Contribution ID: **13** Type: **not specified**

Architectural framework for data analysis Lena

The term «architecture» in software has numerous definitions. Ultimately it defines whether your analysis code will be **extensible** and **maintainable**. We propose an architecture based on the functional style and separation of data, logic and presentation. It is implemented in a free software framework Lena.

Lena is a general data analysis framework in Python, named after a great Siberian river. It allows usage of any Python constructs and functions, but structures the analysis into **reusable** sequences and elements. It natively supports **metadata** (which is important for modern data analysis). It employs lazy evaluation, which makes it suitable for processing data which would not fit into memory, in particular, for **big data** analysis.

The talk will be of primary interest to those who write large programs and face architectural challenges and who need to automatically create many similar plots. The audience will get a powerful tool, which would make their code **structured** and **beautiful**, or understand strengths and weaknesses of an alternative approach to data analysis in Python.

Primary author: NIKITENKO, Yaroslav

Presenter: NIKITENKO, Yaroslav

Contribution ID: **14** Type: **not specified**

File synchronization between Linux systems in Python with yarsync

Yet Another Rsync is a Python wrapper around a well-established Linux tool **rsync** with a **simple** and **familiar interface** of git. Python allows us to create a higher-level instrument, which is safer and sometimes more efficient than the original binary.

While many data analysts today heavily use databases and rely on cloud computing, other approaches have also their benefits. Many data kinds are difficult to represent in relational databases or it takes time to do that. Files in a user-defined format become a simpler and more general solution, which is often less expensive and error prone. Linux servers take a considerable share today, and many data analysts also use Linux as a good programming environment. Our approach is inspired by data analysis workflow in HEP. We shall tell about creating data repositories with yarsync, relevant rsync features and how the tool will assist against possible problems in data synchronization.

Primary author: NIKITENKO, Yaroslav

Presenter: NIKITENKO, Yaroslav

Contribution ID: **15** Type: **not specified**

Case for awkward array - efficient I/O for highly hierarchical data structure

We present from our developer experience, a case for adopting awkward array abstraction in data I/O library for highly hierarchical data set commonly as seen in HEP.

We will discuss both technical, performance-related lessons learned in implementing TTree/RNTuple reading, these are universal principles for all columnar data format I/O.

Specifically, we discuss the the importance of minimizing allocation, and lazy materialization due to sparse access pattern in HEP analysis.

Then we move to high-level, design challenges and how "owning" the entire data representation by using awkward array enables efficient / lazy data I/O that otherwise cannot be easily achieved otherwise.

Specifically, we will use the example of accessing "subset of complex data structures" (e.g. only jets.pt when event.jets isa Vector{LorentzVector}) to demonstrate the weakness of conventional approach and how awkward array provides a systematic way to do exact I/O instead of overtouching.

Primary authors: LING, Jerry (Harvard University (US)); MATO, Pere (CERN)

Presenter: LING, Jerry (Harvard University (US))

Contribution ID: **16** Type: **not specified**

FCCAnalyses: A Framework for Future Circular Collider Physics Performance Studies

Physics performance analyses provide essential input for defining detector requirements in the Future Circular Collider (FCC) project. To streamline these analyses, we employ FCCAnalyses, a software framework built on top of the ROOT DataFrame.

Among the functionalities offered by the FCCAnalyses are:
* *Standard set of RDataFrame EDM4hep functions:* Events can be analysed directly in the EDM4hep event data format with ability to easily work with relationships among the datamodel objects.
* *Multi-stage Analysis Workflow:* Enables running analyses locally or on CERN's HTCondor cluster split into multiple stages.
* *Metadata Management:* Manages metadata associated with centrally produced pre-generated samples produced in Delphes fast simulation and Geant4 full simulation.

The framework can be used equivalently well from Python and C++ and as the analyser functions are written in C++, this gives them possibility to directly employ any of the High Energy Physics (HEP) C++ frameworks.

Primary author: SMIESKO, Juraj (CERN)

Presenter: SMIESKO, Juraj (CERN)

Contribution ID: **17** Type: **not specified**

An overview of the fitting ecosystem

This talk will give a broad overview on the fitting that we're doing in HEP. On one hand, the talk will cover the variety of fits in HEP, the different needs and types of inference as well as efforts for serialization and standardization. On the other hand, the relevant libraries will be covered, that is zfit, pyhf, hepstats iminuit and Python packages like SciPy and how they work together today, as well as the future plans and technical considerations.

Primary authors: PUIG NAVARRO, Albert (Universität Zürich (CH)); KROMMYDAS, Iason (Rice University (US)); ESCHLE, Jonas (Syracuse University (US)); Mr MARINANGELI, Matthieu (EPFL - Ecole Polytechnique Federale Lausanne (CH)); SILVA COUTINHO, Rafael (Syracuse University (US))

Presenter: ESCHLE, Jonas (Syracuse University (US))