Powering Innovation That Drives Human Advancement

Session 1: Overview of PyAnsys Documentation, Source Codes, and Installations

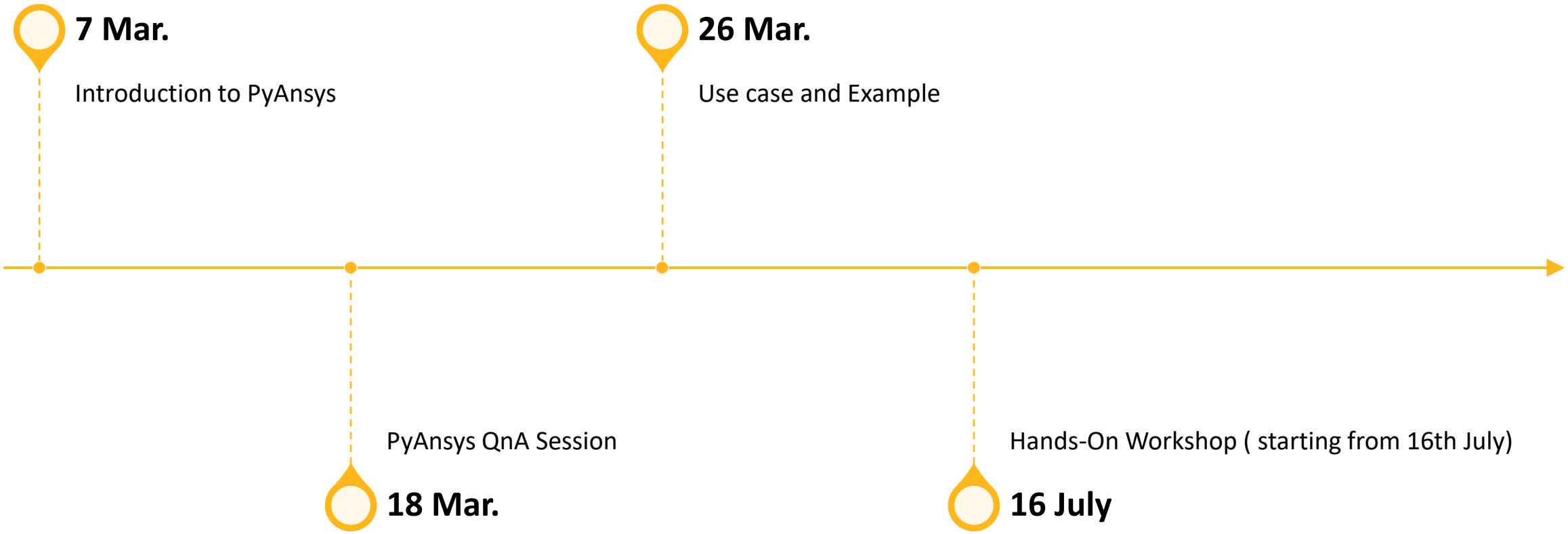
Rajesh Meena

Who am I?

- Automation and Customization of Ansys Products
- Customized Workflow Implementations
- Structural Simulation Expertise
- 9 years in the industry

Rajesh Meena
Sr. Technology Specialist
Ansys Pune, India

PyAnsys Training Series for Channel Partners



PyAnsys Workshop Series

Session	Topic	Coverage	Instructor	Date	Time					Session Duration
					India	Japan/Korea	China/ Singapore	Korea	Australia (Melbourne)	
Session 1	**Overview of PyAnsys Documentation, Source Codes, and Installations**	* Introduction to PyAnsys Documentation * Accessing and Navigating Source Codes * Step-by-Step Installation Guide	Rajesh Meena	**16th July 2024**	11:00 AM	2:30 PM	1:30 PM	2:30 PM	3:30 PM (AEDT)	2 Hours (No Break)
Session 2	**PyAnsys Workshop for Structural Engineers**	* PyAnsys Modules for Structural Simulations (Preprocessing, Setup and Post-Processing) * End-to-End PyAnsys Workflow Demonstrations * Best Practices and Tips * Hands-on Exercises	Rajesh Meena Akshay Jawale Akhil KS	**17th July 2024**	11:00 AM	2:30 PM	1:30 PM	2:30 PM	3:30 PM (AEDT)	4.5 Hours (2 Hr - 30 min Break - 2 Hr)
Session 3	**PyAnsys Workshop for Electronics Engineers**	*PyAEDT and PyEDB Overview * API Structure * Tips to Develop and Debug * Hands-on Exercises	Lindo Ouseph	25th July 2024	11:00 AM	2:30 PM	1:30 PM	2:30 PM	3:30 PM (AEDT)	4.5 Hours (2 Hr - 30 min Break - 2 Hr)
Session 4	**PyAnsys Workshop for Fluids Engineers**	* PyFluent Overview - PyFluent Core - PyFluent Visualization - PyConsole -Meshing Workflows * Hands-on Exercises	Sravan Kumar Nallamothu	6th August	11:00 AM	2:30 PM	1:30 PM	2:30 PM	3:30 PM (AEDT)	4.5 Hours (2 Hr - 30 min Break - 2 Hr)
Session 5	**PyAnsys Auxiliary Modules and Q&A**	* Overview of Auxiliary Modules - PyPrimeMesh - PyOptiSlang - PySimAI * Common Issues and Solutions * Q&A Session for Participants	Rajesh Meena	7th August	11:00 AM	2:30 PM	1:30 PM	2:30 PM	3:30 PM (AEDT)	4.5 Hours (2 Hr - 30 min Break - 2 Hr)

Agenda

- Brief Overview of PyAnsys Project
- PyAnsys: Getting Started
  - Virtual Environment
  - Ansys Python Manager
  - PyAnsys Module Installations
- Documentation Exploration
- Source Code Explorations
- Debugging in Visual Studio Code
- Open Discussion

Pre-requisites

- Basic Knowledge of Python (Intro to Python | Ansys Courses)
- Participant should have attended previous sessions (recordings)
- Ansys 2024R1 software installed on windows machine.
- Any preferred code editor (PyCharm, VS Code, Spyder etc.)
- Install git bash (Steps in next slide)

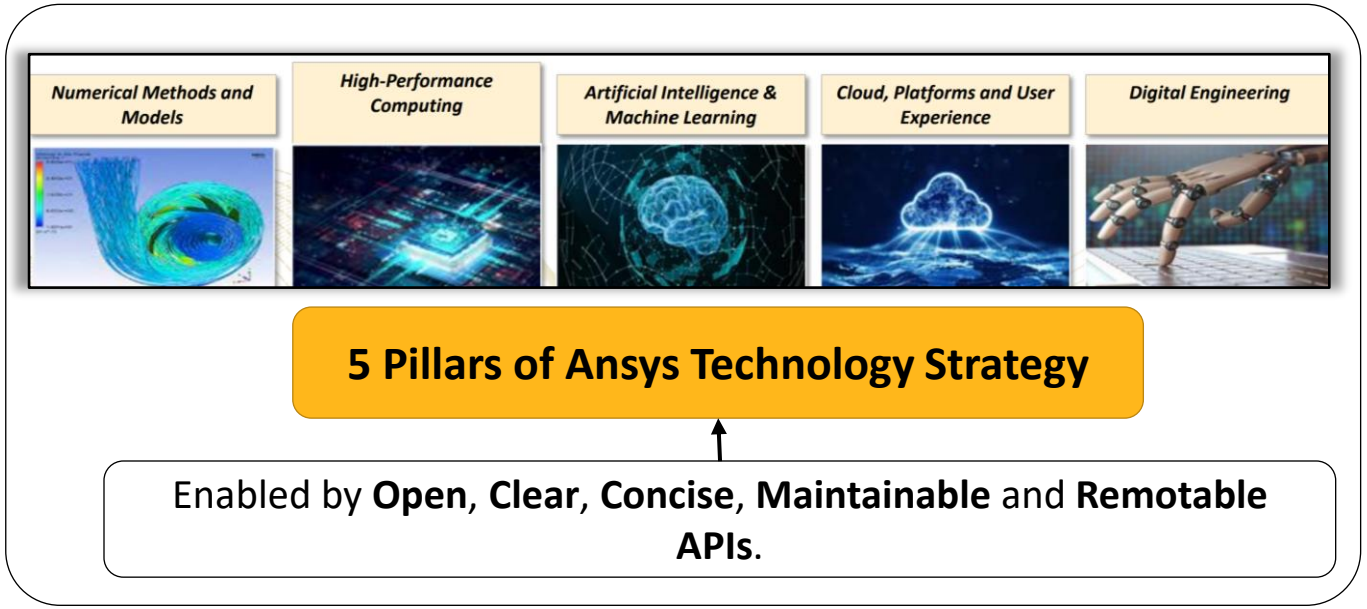
Need for APIs?

Evolving Ansys Customer Needs:

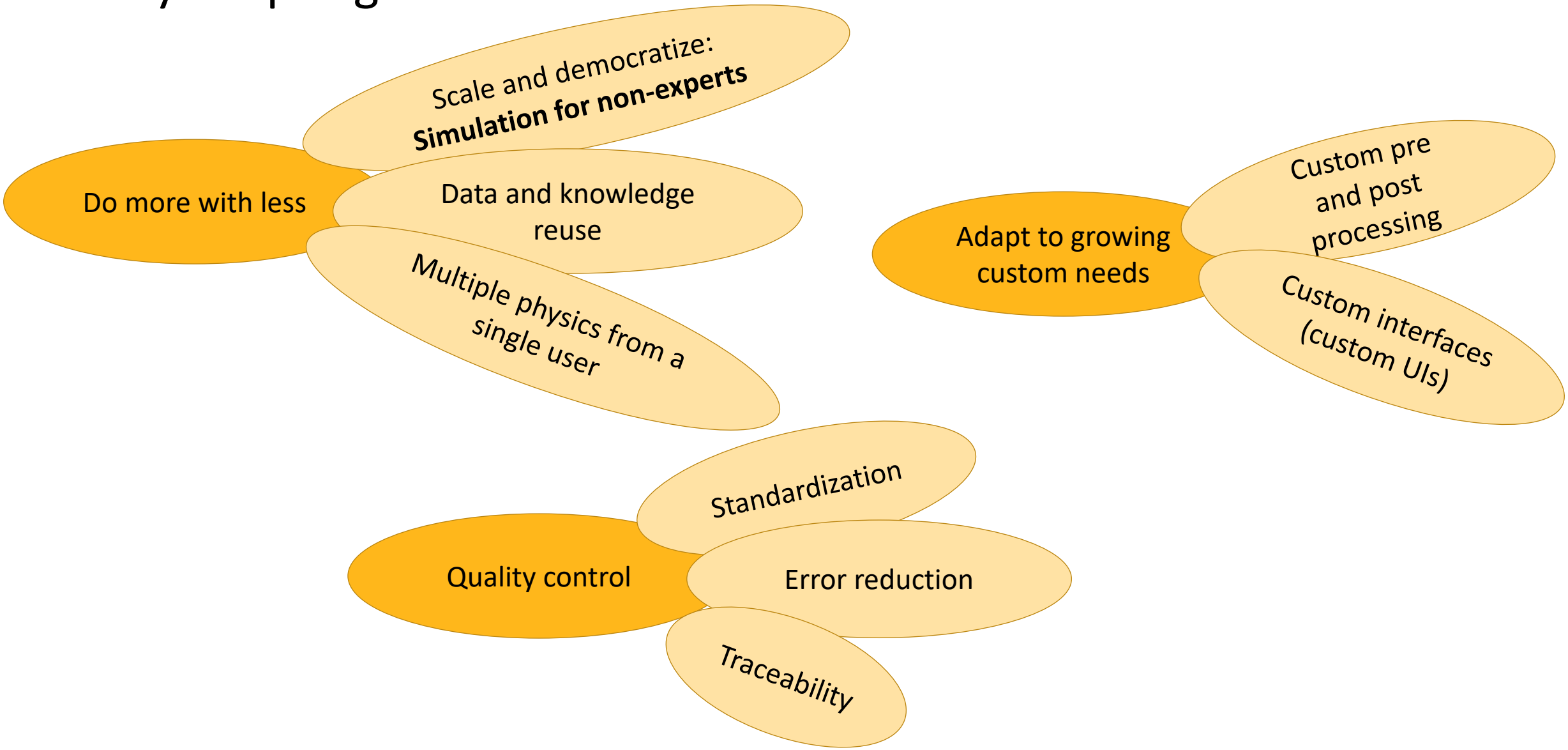
- Democratization of simulation (including non-engineers)
- Accelerate predictable insights
- Connect Ansys technologies to wider enabling technologies such as AI/ML

Ansys Assessment:

- Requirements can be classified into 5 Pillars: **Numerical Solvers**, **High-Performance Computations**, **AI/ML**, **Cloud** and **Digital** Engineering.
- Need for strong and open APIs



Why scripting?



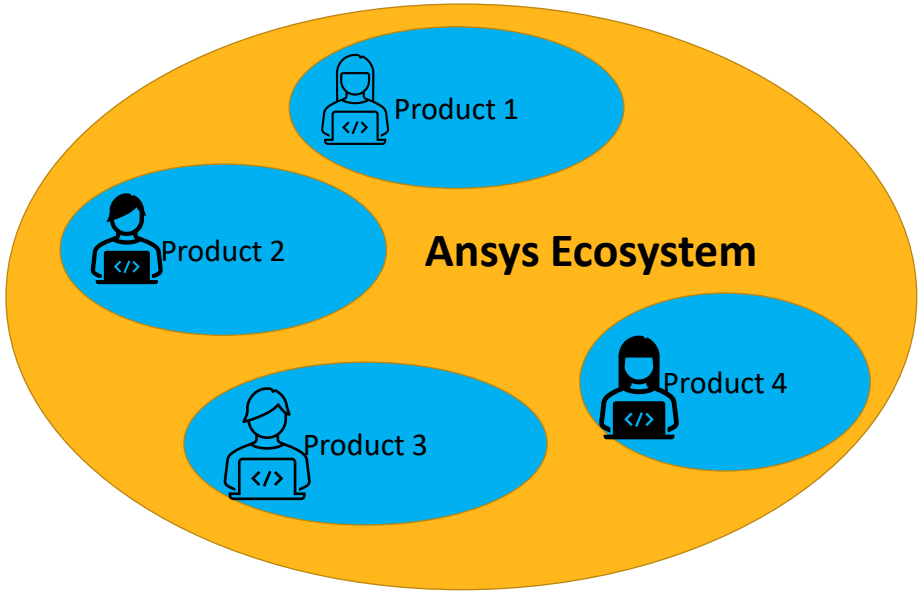
Why scripting?

It’s all about increasing the value of our tools for customers (*)

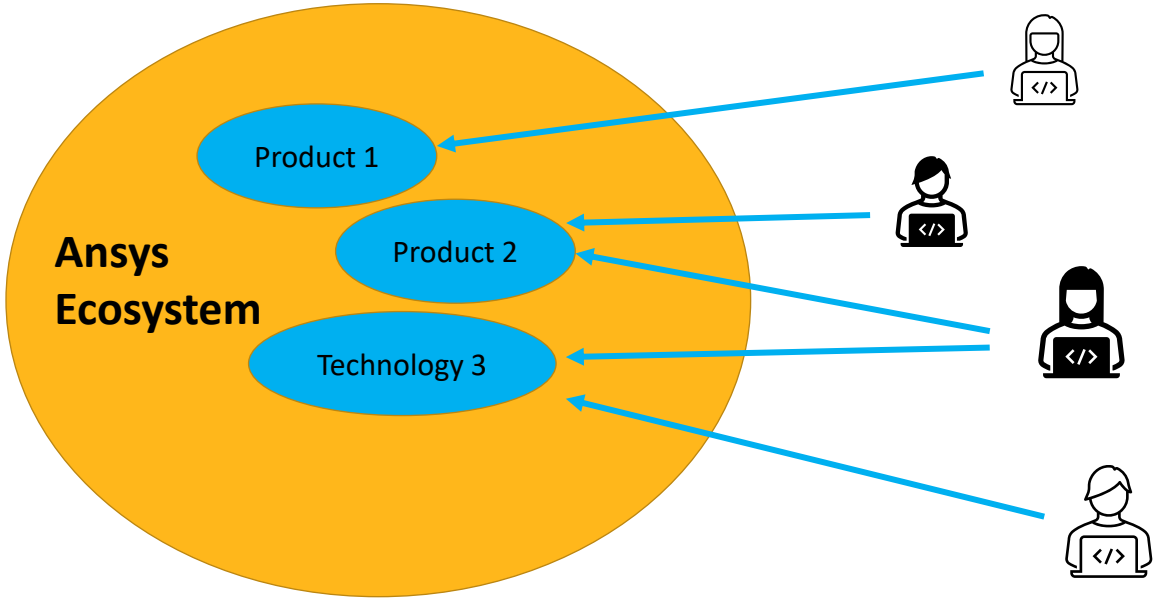
(*) and saving ourselves time as well ☺

Everyone buys the same software from us but uses it in a different way.
Not every usage is covered by “standard” features.

Two possible approaches to scripting with Ansys

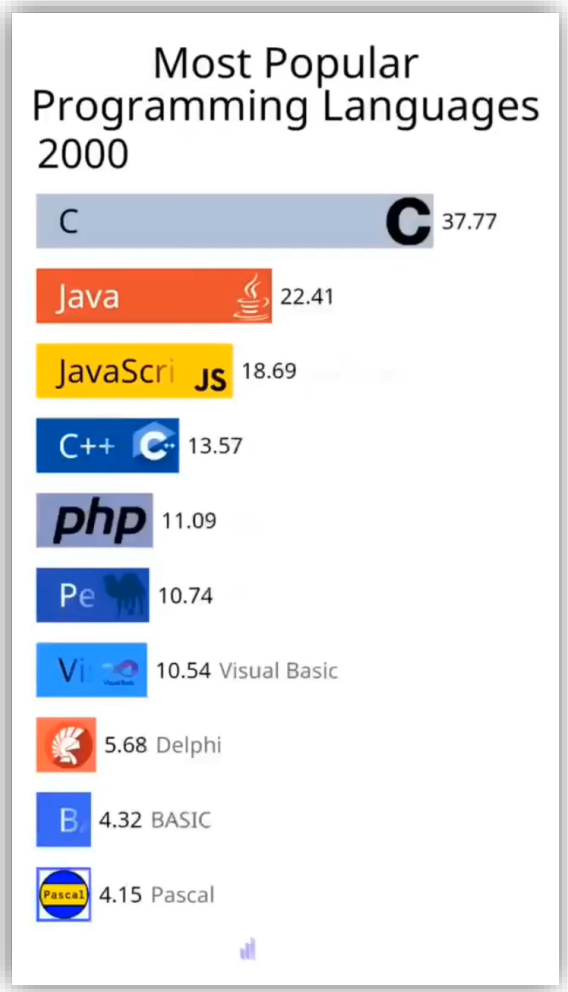


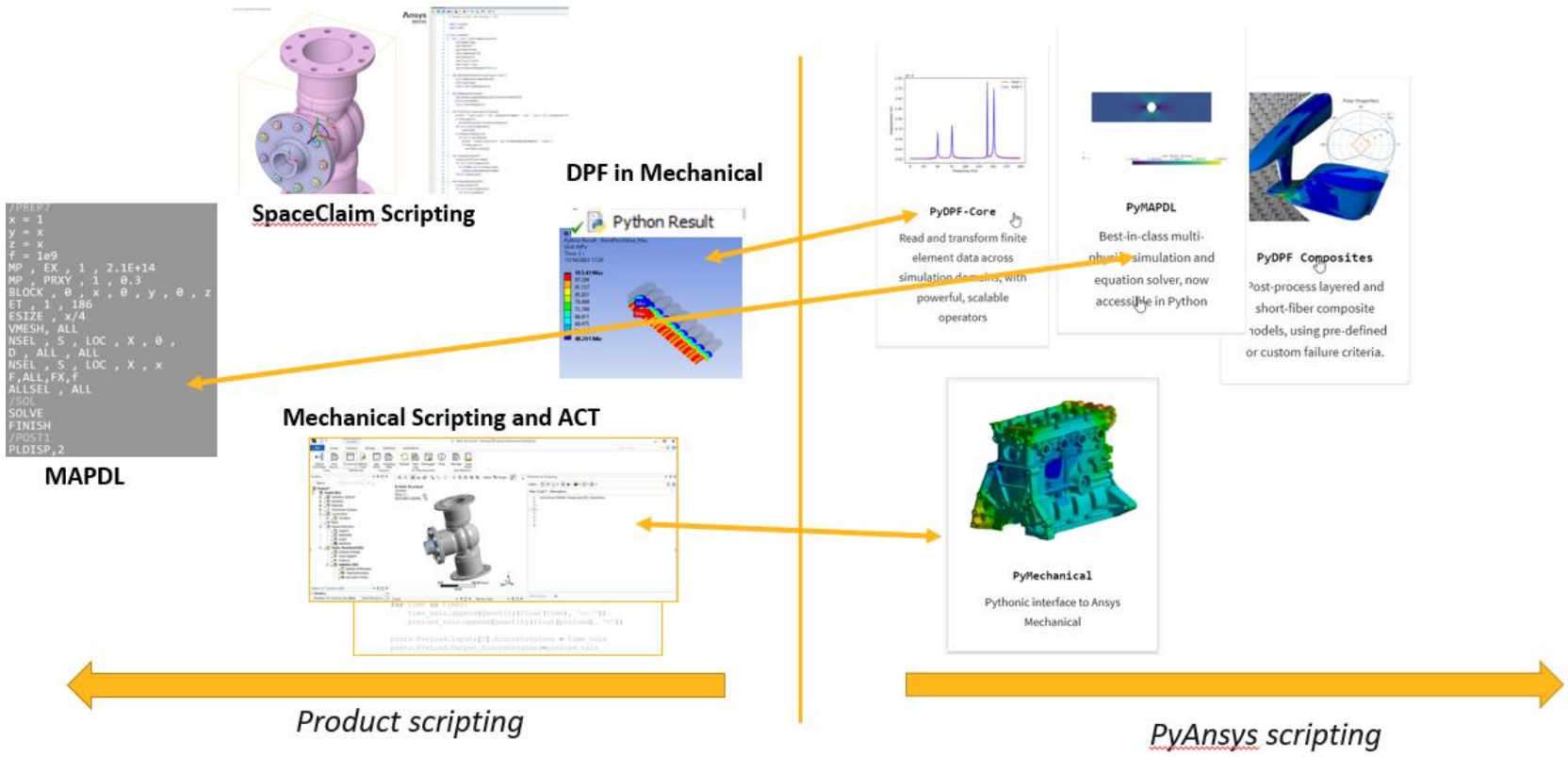
Product scripting



PyAnsys scripting

With a lot of similarities:





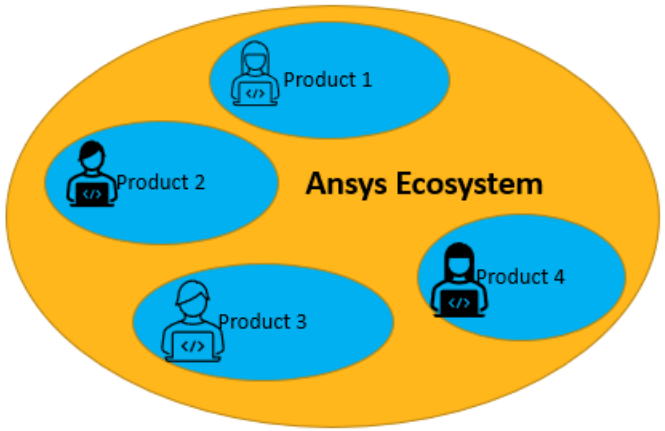
PyAnsys

Ansys

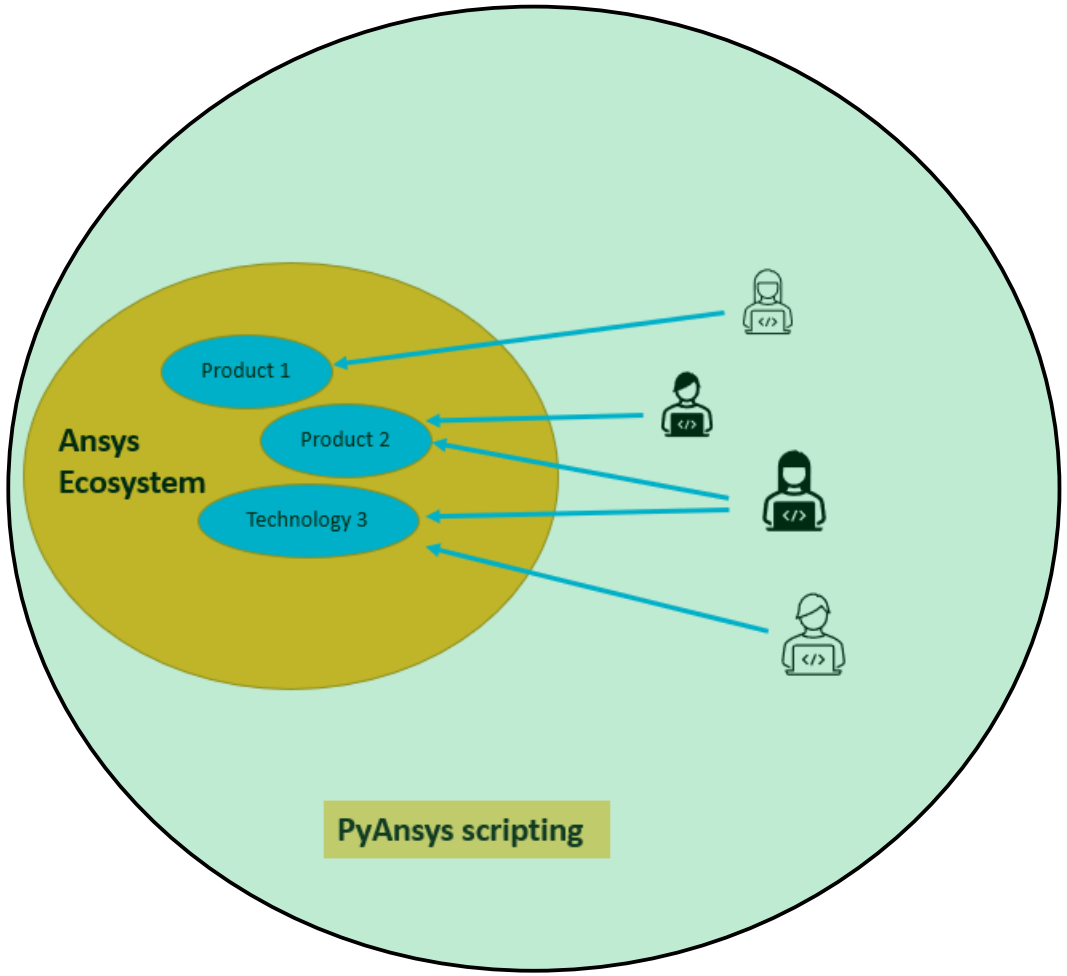
What is PyAnsys?

Set of technologies that allow the user to interface with Ansys products pythonically

PyAnsys scripting



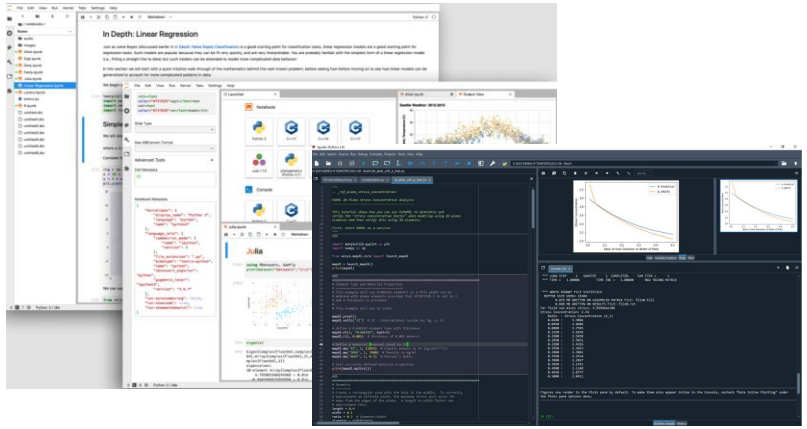
Product scripting



PyAnsys scripting

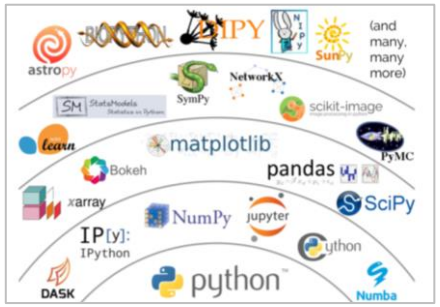
Scripting from within **YOUR** tools

Any tool from which you can write Python (UI is yours to choose)



Customer tools

Possibility to use other useful Python tools



PyAnsys technology

Products

- API — Ansys Product A
- API — Ansys Product B
- API — Ansys MAPDL

Technology components

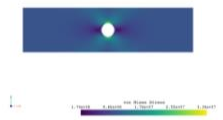
- Meshing
- Data Processing Framework

Ansys Technology

What Packages Are Available?

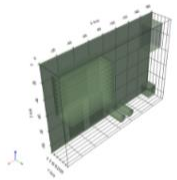
Check out all available packages in the PyAnsys doc

Simulation Libraries



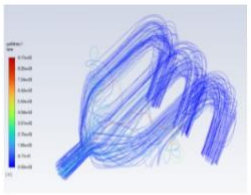
PyMAPDL

Pythonic interface to Ansys MAPDL (Mechanical APDL)



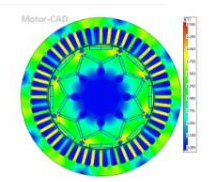
PyAEDT

Pythonic interface to AEDT (Ansys Electronic Desktop)



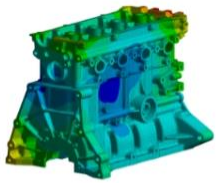
PyFluent

Pythonic interface to Ansys Fluent



PyMotorCAD

Pythonic interface to Ansys Motor-CAD

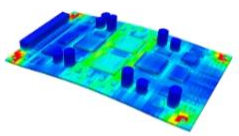


PyMechanical

Pythonic interface to Ansys Mechanical

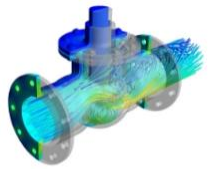
PyOptislang

Pythonic interface to Ansys Optislang



PySherlock

Pythonic interface to communicate with Ansys Sherlock



PySystem Coupling

Pythonic interface to communicate with Ansys System Coupling

PyDyna

Pythonic interface to build the Ansys DYNA input deck, submit it to the Ansys LS-DYNA solver and postprocess its results

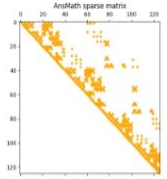
Utility Libraries

PyPrimeMesh

Pythonic interface to Ansys Prime Server, which delivers core Ansys meshing technology

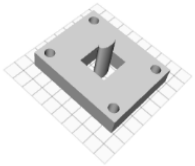
PyDynamicReporting

Pythonic interface to Ansys Dynamic Reporting for service and control of its database and reports



PyAnsys Math

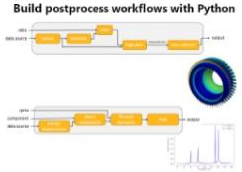
Pythonic interface to PyAnsys Math libraries



PyAnsys Geometry

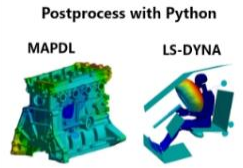
Pythonic interface to the Ansys Geometry service

Postprocessing Libraries



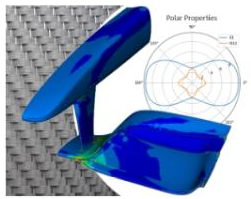
PyDPF - Core

Pythonic interface to DPF (Data Processing Framework) for building more advanced and customized workflows



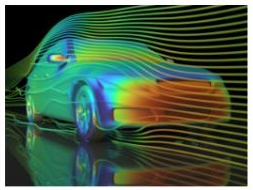
PyDPF - Post

Pythonic interface to access and post process Ansys solver result files



PyDPF - Composites

Pythonic interface to post-process layered and short-fiber composite models

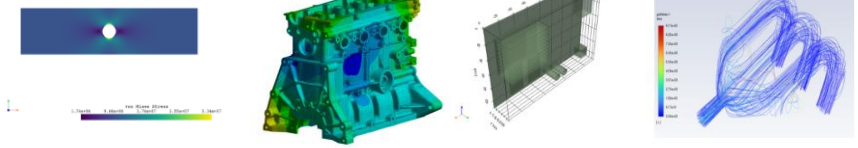


PyEnSight

Pythonic interface to EnSight, the Ansys simulation postprocessor

PyAnsys: What's available?

Simulation libraries



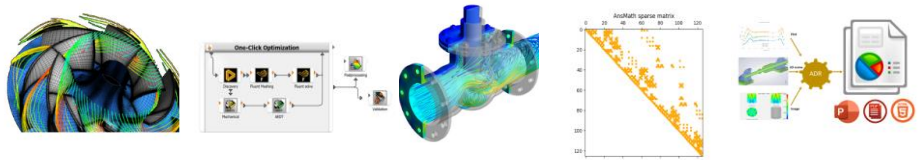
PyMAPDL
Pythonic interface to Ansys MAPDL (Mechanical APLD)

PyMechanical
Pythonic interface to Ansys Mechanical

PyAEDT
Pythonic interface to AEDT (Ansys Electronic Desktop)

PyFluent
Pythonic interface to Ansys Fluent

Utility libraries



PyPrimeMesh
Pythonic interface to Ansys Prime Server, which delivers core Ansys meshing technology

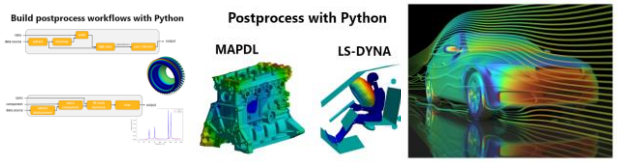
PyOptislang
Pythonic interface to Ansys Optislang

PySystem Coupling
Pythonic interface to communicate with Ansys System Coupling

PyAnsys Math
Pythonic interface to PyAnsys Math libraries

PyDynamicReporting
Pythonic interface to Ansys Dynamic Reporting for service and control of its database and reports

Post-processing libraries

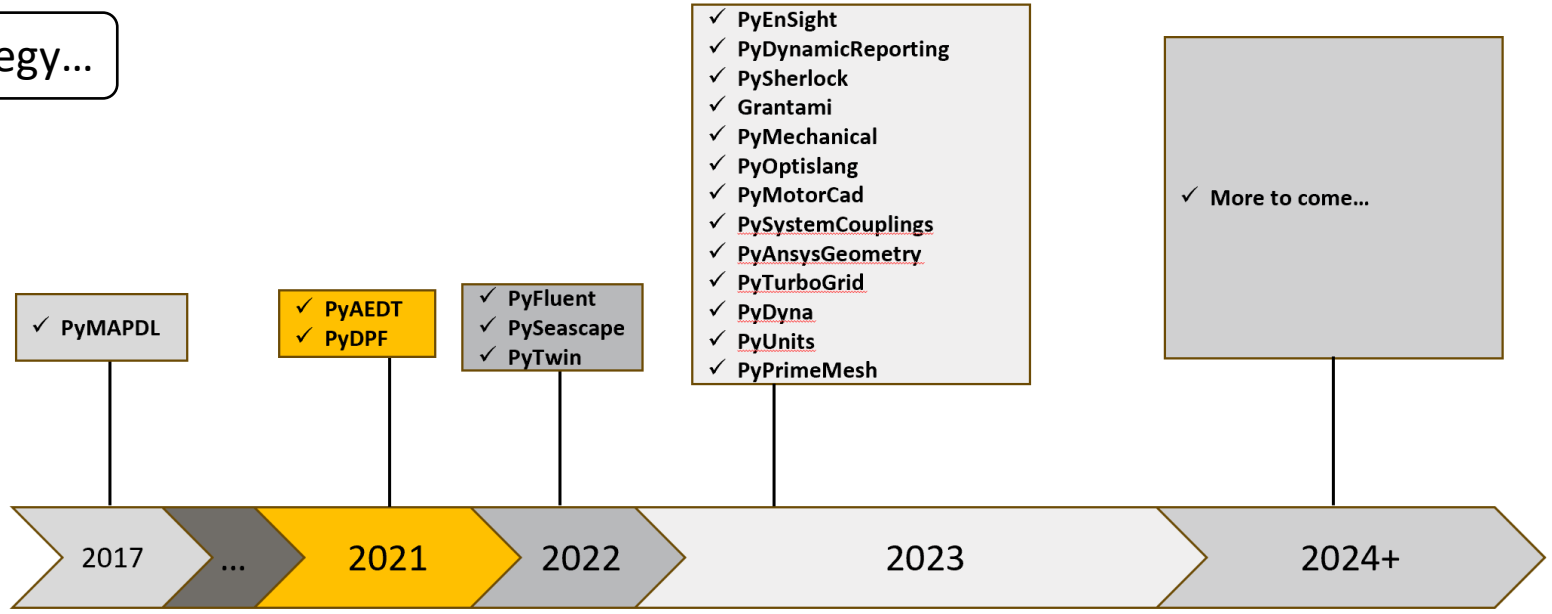


PyDPF - Core
Pythonic interface to DPF (Data Processing Framework) for building more advanced and customized workflows

PyDPF - Post
Pythonic interface to access and post process Ansys solver result files

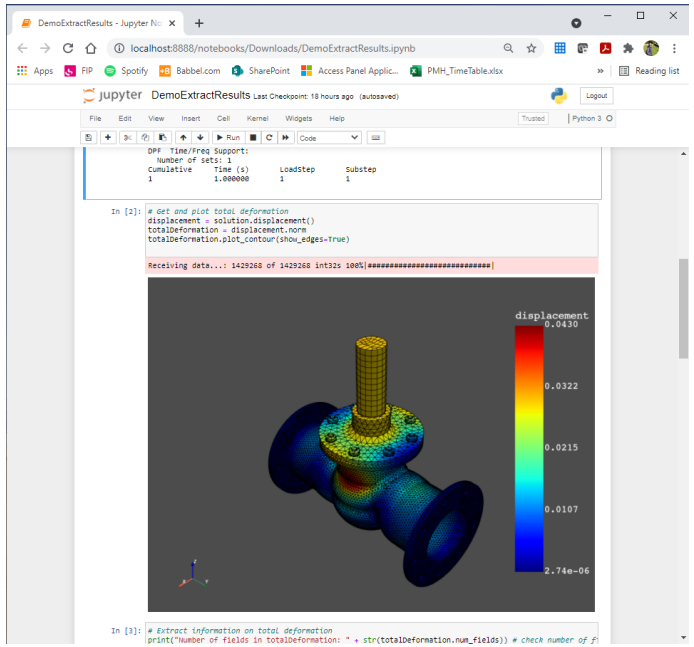
PyEnSight
Pythonic interface to EnSight, the Ansys simulation postprocessor

5-year strategy…

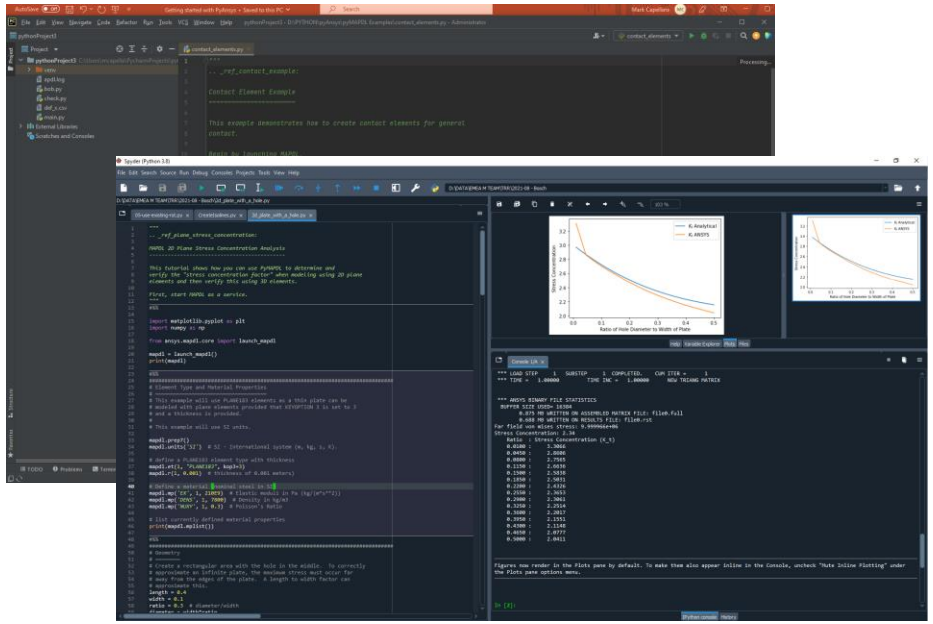


Keep an eye on…
docs.pyansys.com

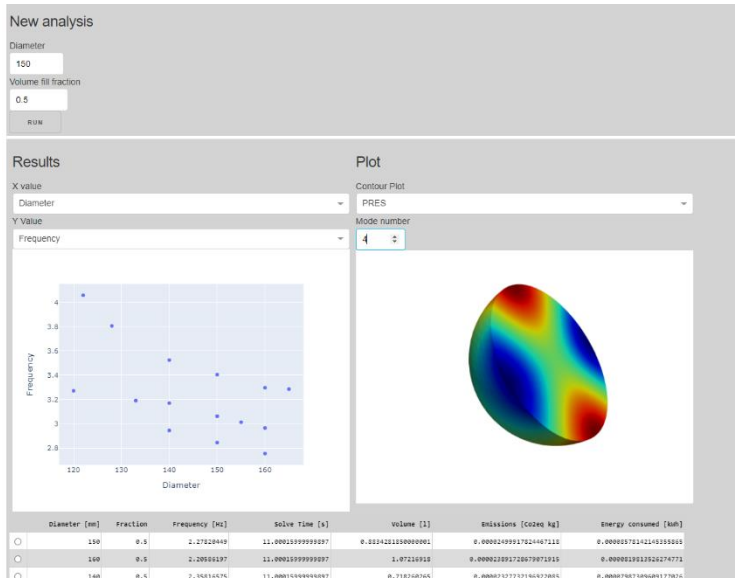
Is there a user interface (UI)?



Code can be written from a web-page thanks to Jupyterlab

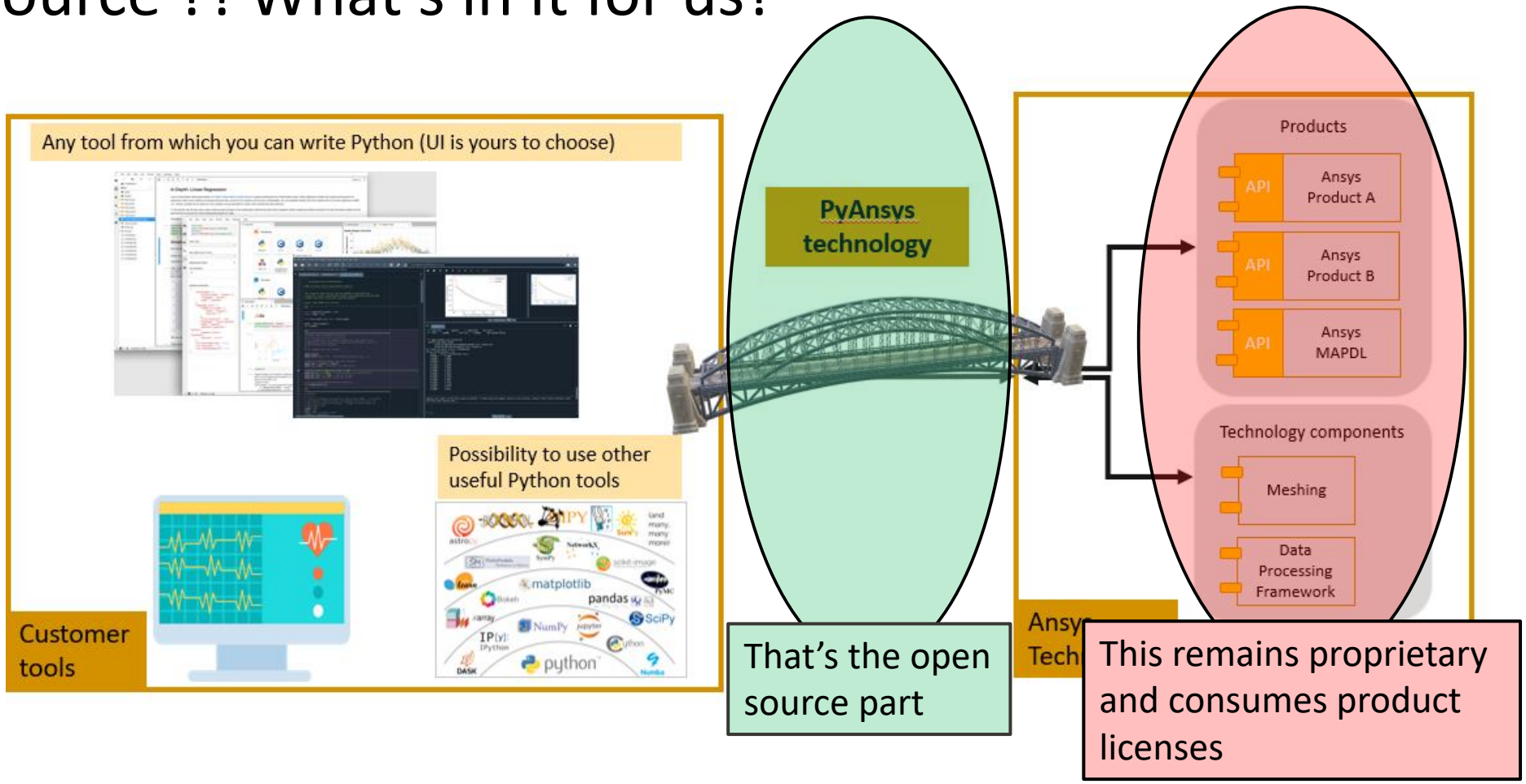


An IDE can be used (PyCharm, Spyder, …)



User can also develop apps by coding both the front end and the back end

Open source ?! What's in it for us?



We're just offering a new way of using our products!
This will allow existing users to use our products differently, effectively and efficiently.

PyAnsys: Getting Started

How do I start?

Python

Ansys

Install Python

Python

https://www.python.org/downloads/

What should be done after installing Python?

Python

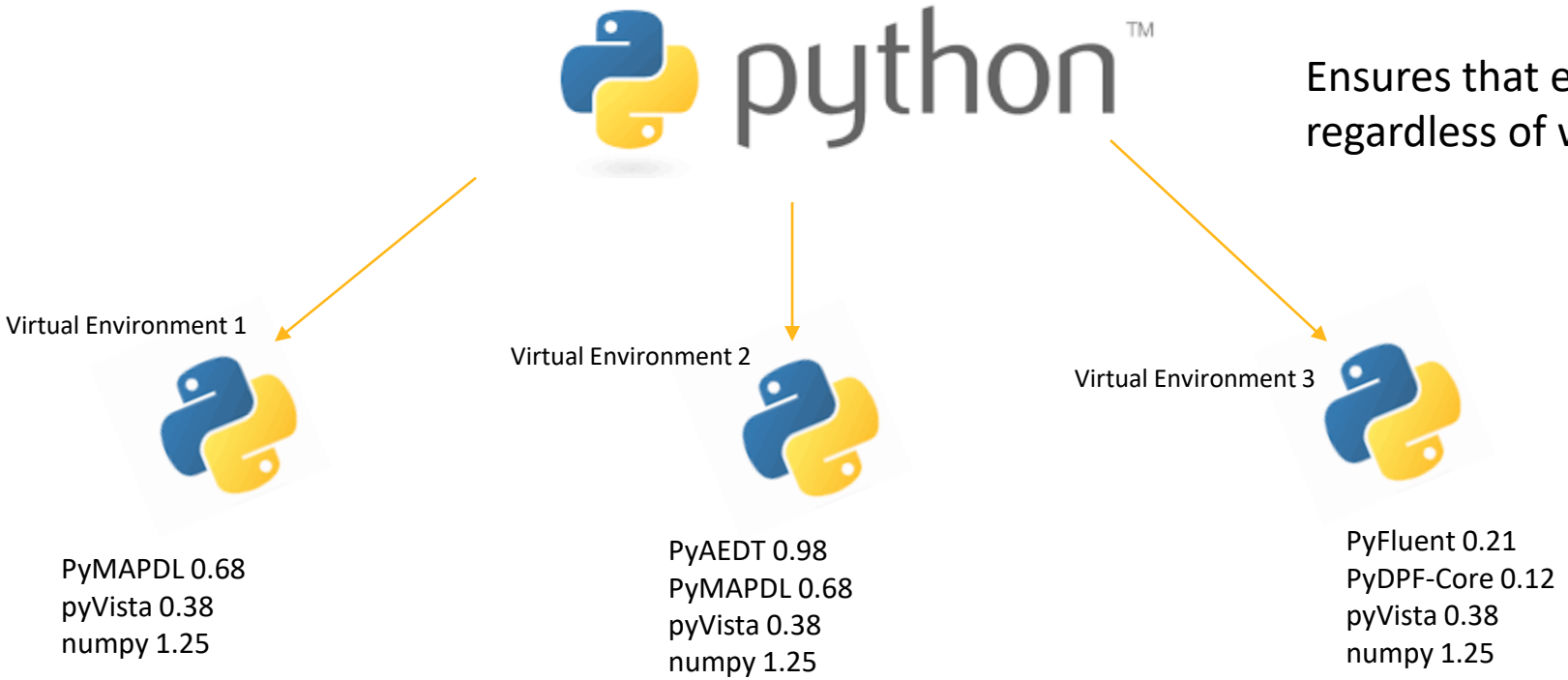
Question: What is next step?

B) Create Virtual Environment

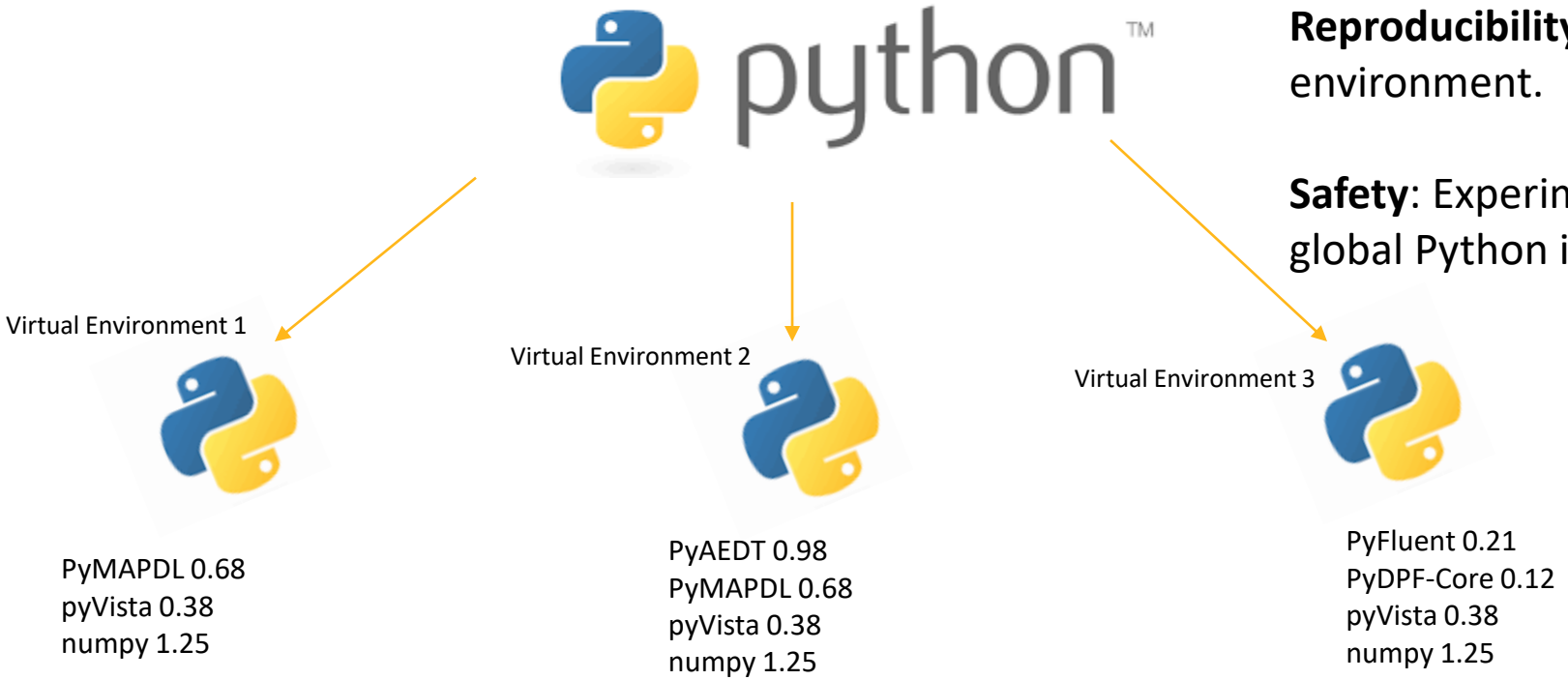
What is Virtual Environment?

A self-contained directory that contains a Python **installation for a particular version of Python**, plus a number of additional packages.

Ensures that each project can **have its own dependencies**, regardless of what dependencies every other project has.



What is Virtual Environment?



Isolation: Keep dependencies required by different projects in separate places.

Compatibility: Avoid version conflicts between packages.

Reproducibility: Ensure that others can reproduce your environment.

Safety: Experiment with packages without affecting the global Python installation

Whenever you are facing strange issue in Python Scripts

Create a fresh virtual environment and retry

How do you create environment and use it?

- Pipenv
- Venv
- Virtualenv

- Official Python documentation on virtual environments: Python Docs – venv
- Tutorials and guides: Real Python - Virtual Environments

Is there an easier way?

Use Ansys Python Manager – Graphical Installation (windows & linux)

Need:

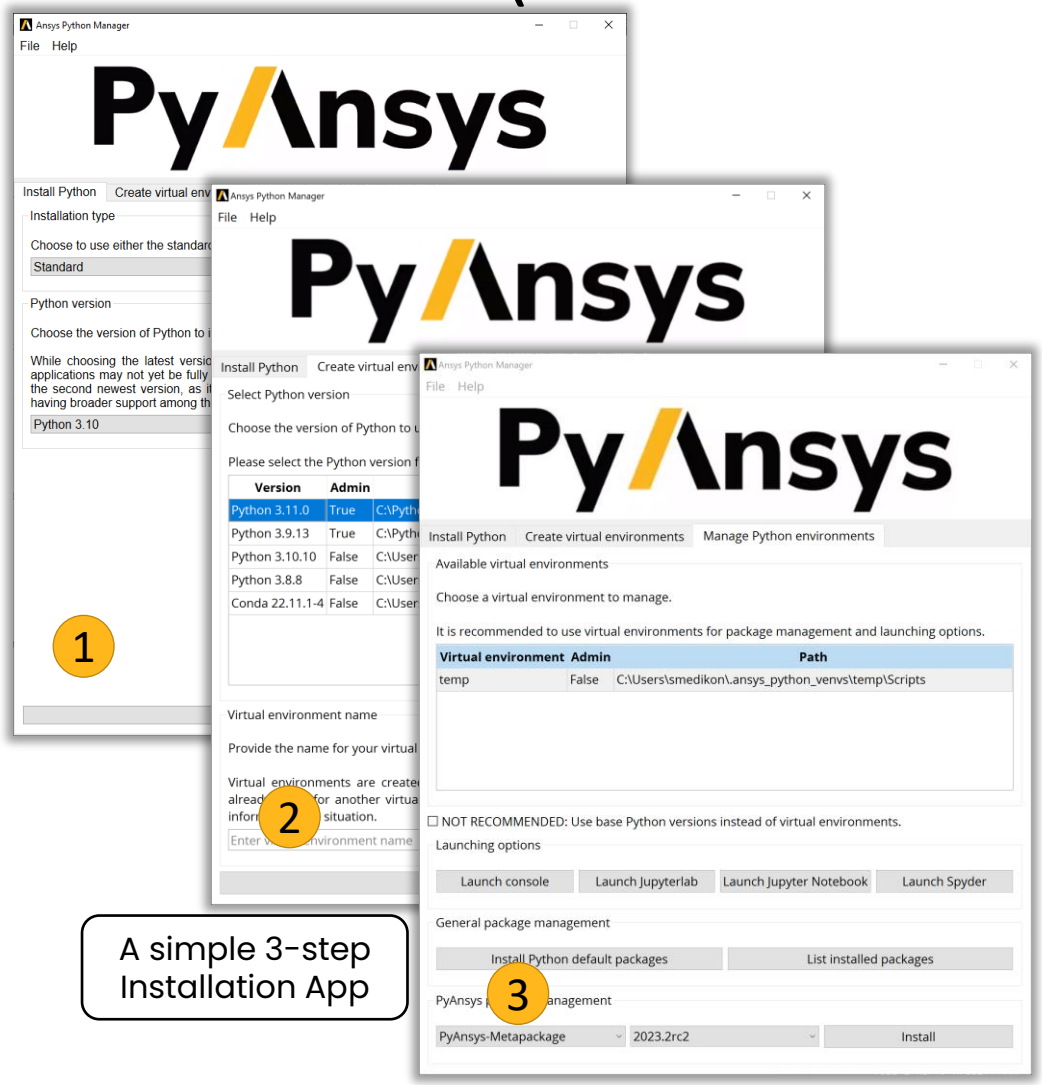
- Similar to the **Anaconda GUI installer**: Transition to a **Commercial Business Model** was a major roadblock for Ansys Customers.
- Reduce Impact for Beginner & Intermediate Python users.

Solution/Tool Created:

- A simple 3-step installer app.

- Using Ansys Python Manager:
  Download it from:
  [Releases · ansys/python-installer-qt-gui (github.com)]

https://developer.ansys.com/ansys-python-manager



How to install modules?

- pip install module_name
  - e.g pip install ansys-dpf-core
- Is that all?
  - There are additional optional dependencies.
  - e.g pip install ansys-dpf-core[plotting]

Exercise

- Find command for optional dependencies for any two Python modules

Code Editor

Let's Set up Code Editor.

- Which Code Editor is installed on your machine?

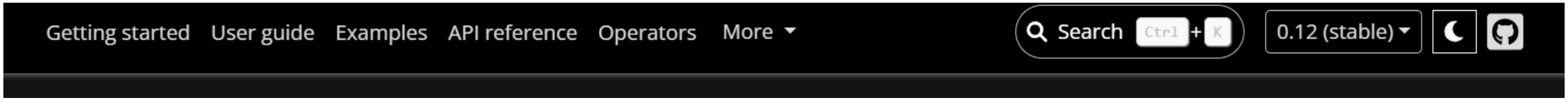
Visual Studio Code

PyCharm

PyAnsys Documentation

PyAnsys Documentation

- Documentation Link: https://docs.pyansys.com/
- Features:
  - Automatically Built from Source Codes (Sphinx)
  - Common Theme Across Modules



PyAnsys Source Code Exploration

What is important on GitHub Repository?