# Waltham Users Forum: Unleashing the Power of PyAnsys

Sandeep Medikonda

September 20<sup>th</sup>, 2023

Ansys

© Copyright 2023 ANSYS, Inc.



- What is PyAnsys?
- Why use it & innovation offered?
- What's Available?
- Popular PyAnsys Workflows
  - Machine Learning Example
  - WebApps
- How to install it?
- Documentation, Help & Other Resources
- Conclusions





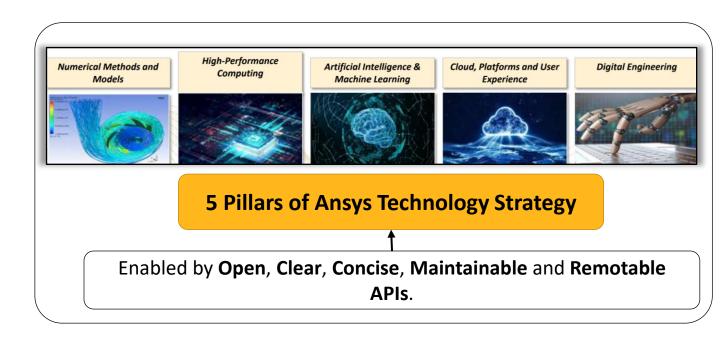


#### **Evolving Ansys Customer Needs:**

- Democratization of simulation (including nonengineers)
- 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

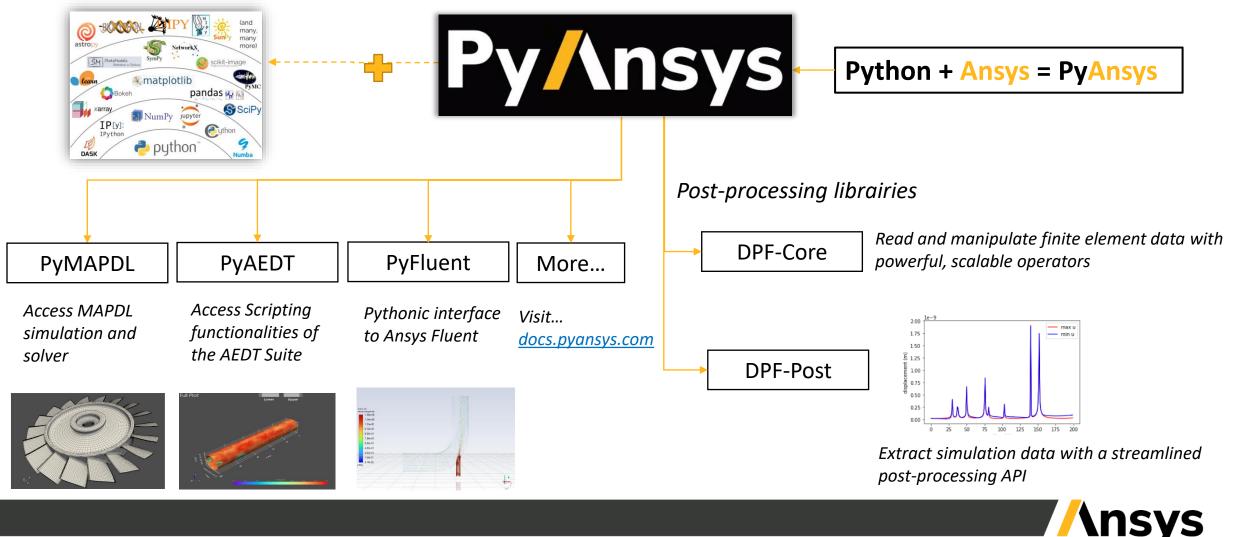


#### It's all about increasing the value of our tools for customers...



## What is PyAnsys?

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



# Putting the Py in PyAnsys

- Py is short for python, the language, not the snake
  - nor pie which is a shame.
- Python is becoming the *lingua Franca* of engineering software languages especially data investigation (e.g AI/ML).



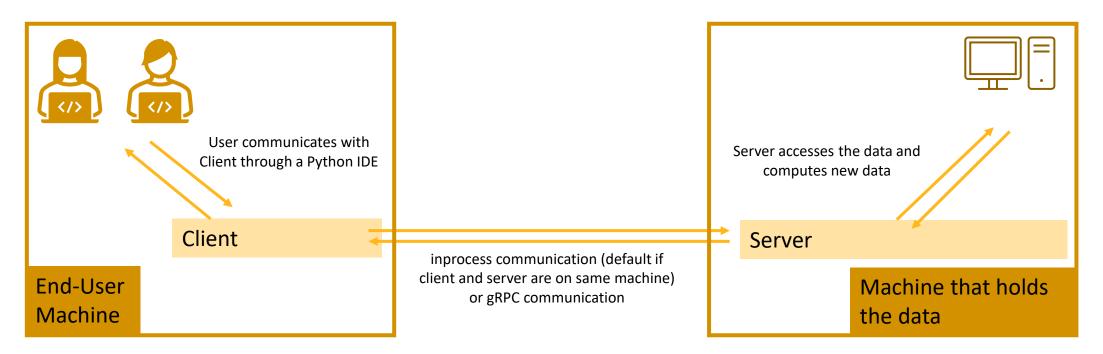






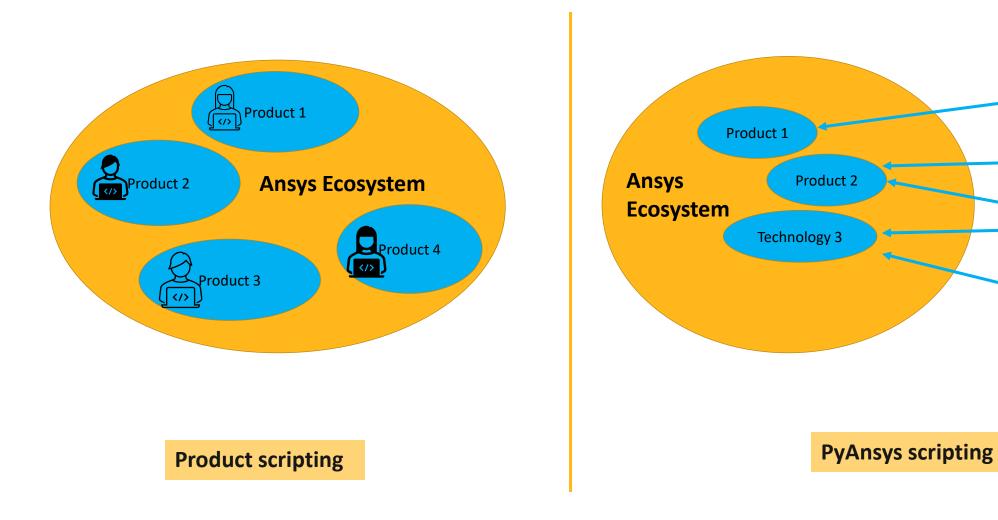
## Client/Server architecture

<u>PyAnsys project</u> offers several packages that are usually based on a Client/Server architecture. Client and Server can be on the same machine, or on separate machines. Below is a typical configuration:



- Where and how to find the files?
  - The Client part comes is provided through Python packages (pip install)
  - The Server part is shipped with the standard Ansys installation

## Two possible approaches to scripting

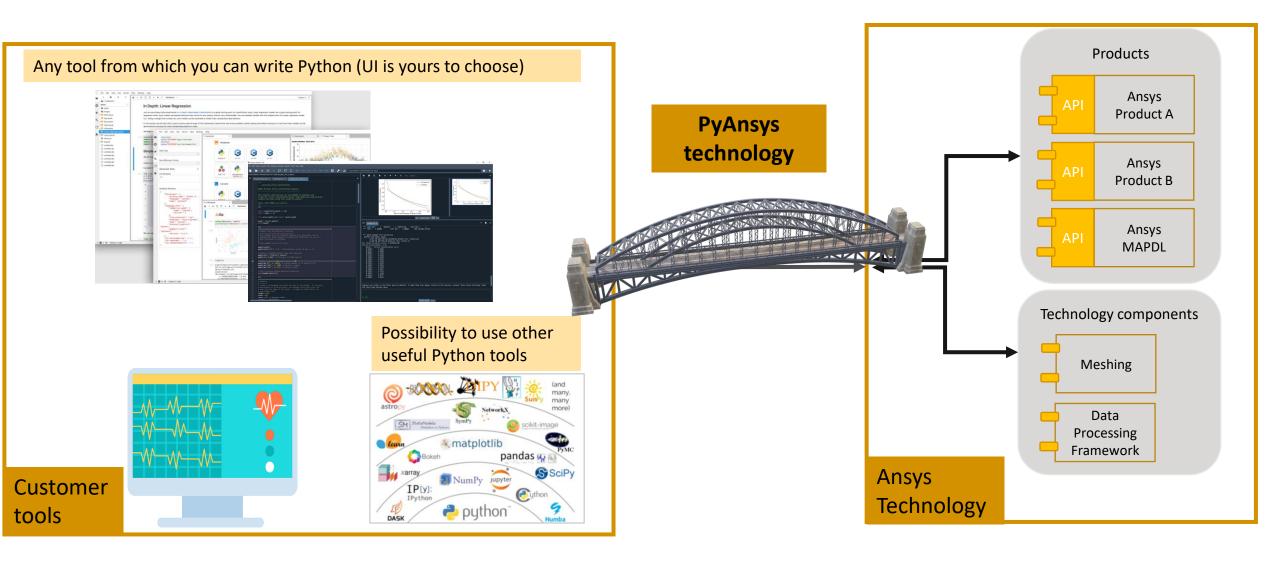




Ξ </>

©2023 ANSYS, Inc.

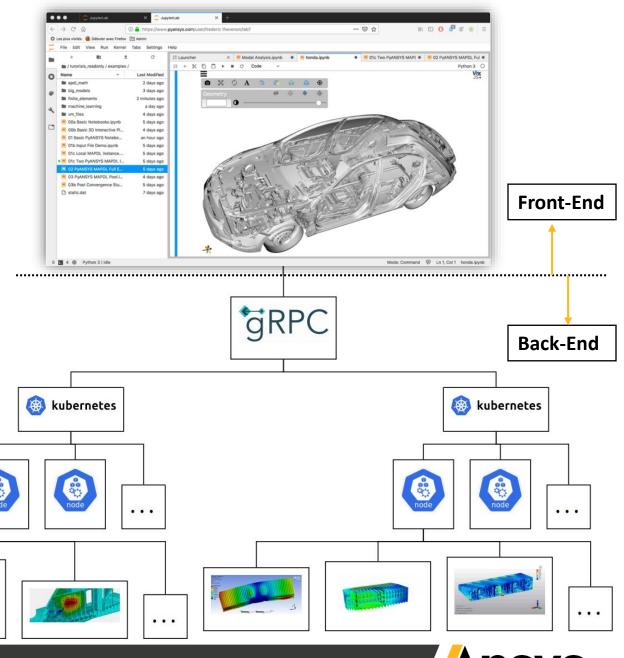
#### Scripting from within the customer tools – PyAnsys



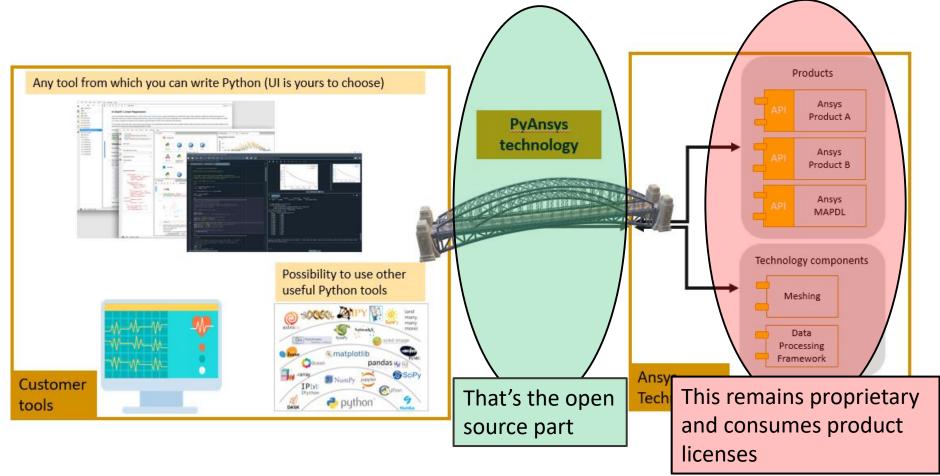


# Why use it & Innovation offered?

- The PyAnsys project is Ansys's commitment to opensource where we provide Python libraries that expose Ansys technologies in the Python ecosystem through APIs and interfaces that are clear, concise, and maintainable. This allows Ansys Customers to do:
  - Flexible Automation: Democratize powerful capabilities offered by Ansys through scripting
  - Flexible Distribution : Connect Ansys and Open-Source technologies in a seamless manner
  - Broader Technology Integration: Integrate Ansys physics capabilities easily with AI/ML
- Built on cutting edge technologies such as:
  - gRPC
  - Kubernetes Framework (or any HPC)
  - Headless Docker Containers



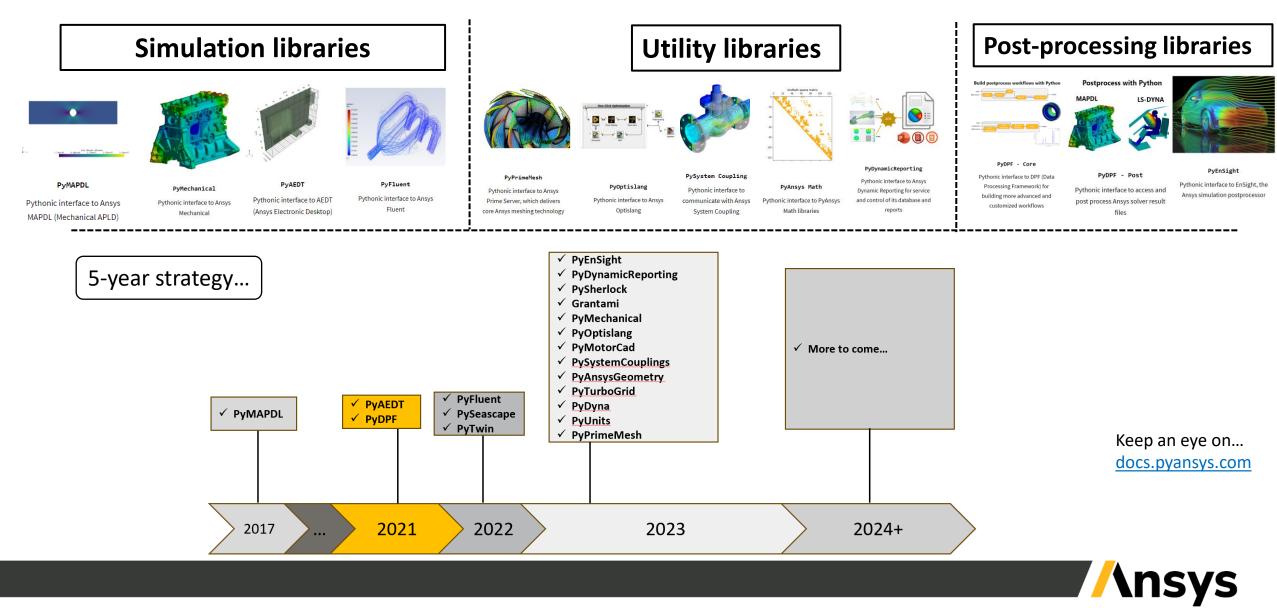
## Open source ?! What's in it for us?



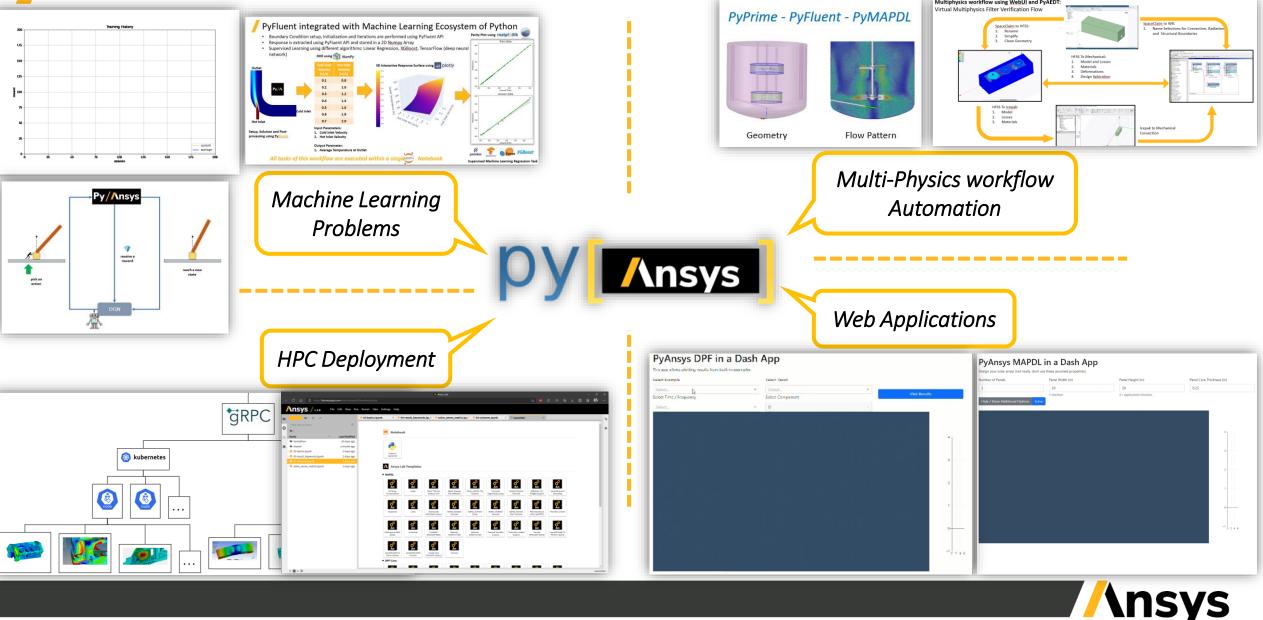
We're just offering a new way of using our products!

This will allow existing users to use our products differently, effectively and efficiently.

### PyAnsys: What's available?



#### Key workflows where Ansys customers are using PyAnsys

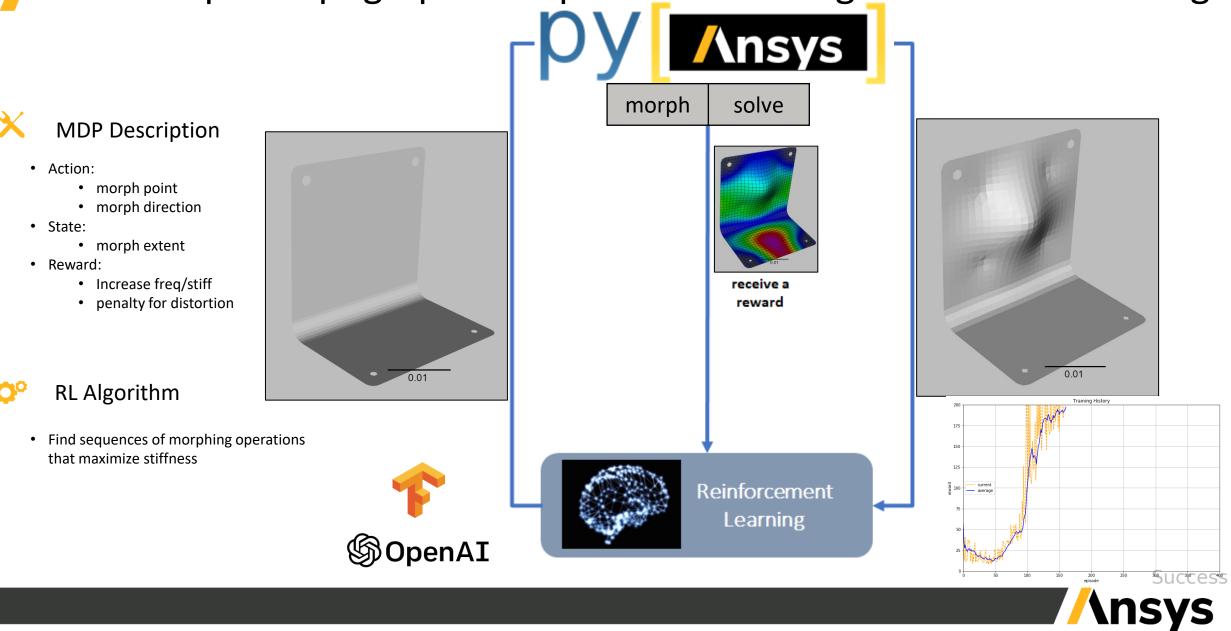


### Angle Bracket Optimization

Angle bracket has "dimple" to increase stiffness



#### ML Example: Topographical Optimization using Reinforced Learning



## Spray Dryer WebApp





## Use Ansys Python Manager – Graphical Installation (windows)

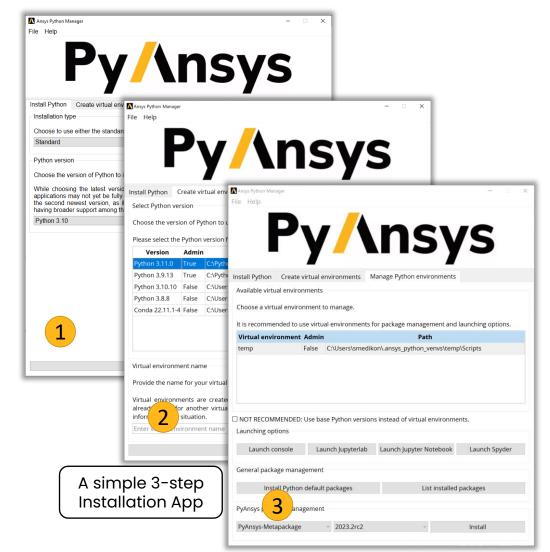
#### **Challenge/Need:**

- 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 created in <1 month.
- Using Ansys Python Manager: Download it from: <u>Releases · ansys/python-installer-qt-gui</u> (github.com)

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





### How to get started with PyAnsys (local installation)?

#### 3 Step Process:

1. Install Ansys

2. Install Python and an IDE of your choice (Spyder, Jupyter, Jupyter Lab, ...)

3. Install PyAnsys modules

Release version	Release date		Click for more	
Python 3.9.0	Oct. 5, 2020	🕹 Download	Release Notes	
Python 3.8.6	Sept. 24, 2020	🕹 Download	Release Notes	
Python 3.5.10	Sept. 5, 2020	🕹 Download	Release Notes	
pvthon	-m venv	PvAnsvs		
		ripts\activ	vate	
pip in	stall jup	yterlab		
pip in	stall ans	ys-mapdl-co	ore	
pip in	stall ans	sys-mapdl-re	eader	2
pip in	stall pya	edt		
pip in	stall ans	sys-dpf-core	9	
pip in	stall ans	sys-dpf-pos	t	
pip in	stall ans	sys-fluent-	core	
pip in	stall ans	- sys-fluent-j	parametri <u>c</u>	
	stall ans	support-ansys.con	visualizatic	

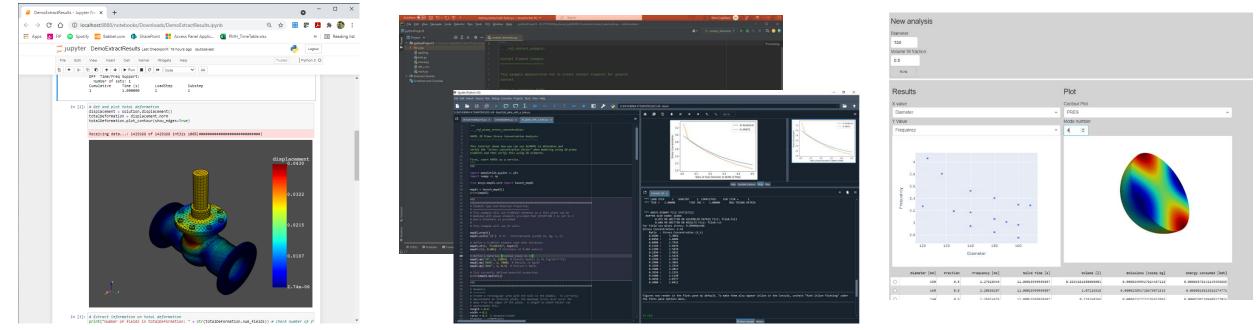
python.org/downloads/



## What's the UI like?

#### No specific UI. User's choice.

- PyAnsys is a set of tools to interact with Ansys products from a Python interface
- Any Python interface can be used (even the Windows Command Prompt !)
- Using an IDE (Integrated Development Environment) is recommended to write and maintain code. Many IDEs are available: <u>Spyder</u>, <u>PyCharm</u>, <u>VS Code</u>, <u>Visual Studio</u>, ....



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



©2023 ANSYS, Inc.

#### **Documentation & Help?**

#### Source Code: <u>https://github.com/ansys/pyansys</u>

PyAnsys PyAnsys - Ansys Python deve Phttps://docs.pyansys.com/				
Coverview ☐ Repositories 44	Packages A People 62 Ax	Teams 1	🛄 Projects 🛛 🗠 Insights	
Pinned				
Pymapdl     Public       Pythonic interface to MAPDL       Python     259     \$ 59	<b>pyaedt</b> AEDT Python Client Package         ● Python ☆ 32 ぷ 7	Public	Public       Data Processing Framework - Python Core       Python     23     26	
Data Processing Framework - Post Processing Module	pymapdl-reader  Legacy binary interface to MAPDL bin	Public nary files.		
● Python 🏠 12 😵 1	● Python 🏠 15 😚 9			

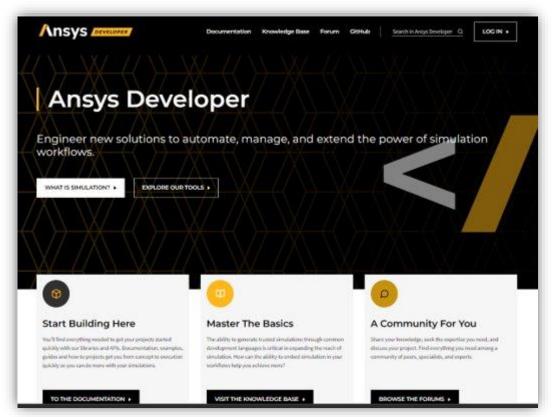
#### Contribute/Post-Issues directly on Github: C C Sign up 📃 🔳 Sign up 📃 📃 Designment of the pyansys / PyAEDT pyansys / pymapdl On Notifications ☆ Star 224 ♥ Fork 53 Ω Notifications ☆ Star 12 ¥ Fork 2 Code O Issues 34 11 Pull requests 3 Discussions O Actions 1 Security Insights ⇔ Code ⊙ Issues 22 1 Pull requests 4 ⊙ Actions I Projects 1 □ Wiki ① Security O ississue iscopen C Labels 10 O Milestones 0 C Labels 18 Milestones 0 New issue ississue iscoper New issue ⊙ 34 Open ✓ 324 Closed ⊙ 22 Open ✓ 26 Closed Author + Label + Projects + Milestones + Assignee + Sort + Author - Label - Projects - Milestones - Assignee - Sort -O MAPDL server connection terminated after error in do loop Bug O Add full doc build to the CI/CD 2 #544 opened 19 days ago by echatziz #372 opened 5 days ago by akaszynski Add Python 3.9 Support Bug ⊙ function Maxwell2d.assign\_current does not work on Magnetostatic solver 🍘 🖓 🖗 2 #543 opened 20 days ago by akaszynski enhancement #371 opened 5 days ago by tizianro O lplot and kplot slow and unresponsive within notebook environment when showing C 3 (2) D 3 number labels. Bug Maxwell 2D example in Example gallery does not work in PyCharm docu #370 opened 5 days ago by tizianro #525 opened 29 days ago by CesarRodriguezPereira 1 D 4 O Assigning design variable to parameter "amplitude" Maxwell2d.assign\_current() does Hello,I want to know something about Solve Sparse Linear Systems Support #524 opened on Jul 10 by Villareally not work. #367 opened 7 days ago by gane O Autosummary for working plane is broken Documentation 8 O Add doc coverage CI/CD #523 opened on Jul 9 by akaszynski #364 opened 11 days ago by akaszyns Include the element Library in PyMAPDL's documentation New feature 2 💭 🕲 UDP Parametrization needs expanding to handle Version and NoOfParameters fields 8 #516 opened on Jul 7 by jad10 #351 opened 14 days ago by voss70 Implement the APDL \*GET shortcuts in PyMAPDL New feature a 🔋 💭 12 ⊙ The returned string is "10WWatt". It should be "10W" or "10Watts" documents #504 opened on Jun 23 by jod10 <a>26 of 51 tasks</a> #349 opened 14 days ago by MaxJPRet ⊙ title, return\_plotter, theme ... are not valid parameters for plot\_nodal\_... New feature • Add Eve Diagrams methods to Circuit e #494 opened on Jun 21 by natter1 #346 opened 14 days ago by maxcapodi78 **P**1 • Visualization of cross sections when plotting New feature O Possibly redundant methods in Q2d.py #485 opened on Jun 17 by CesarRodrig

#### **Documentation:**

- Access from GitHub or use these Direct links:
  - <u>https://docs.pyansys.com/</u>
  - <u>https://mapdl.docs.pyansys.com/</u>
  - <u>https://dpf.docs.pyansys.com/</u>
  - <u>https://aedt.docs.pyansys.com/</u>



#### Need additional help? Use the Ansys Developer Portal and Developer Forum



Ansys Developer Portal: https://developer.ansys.com/

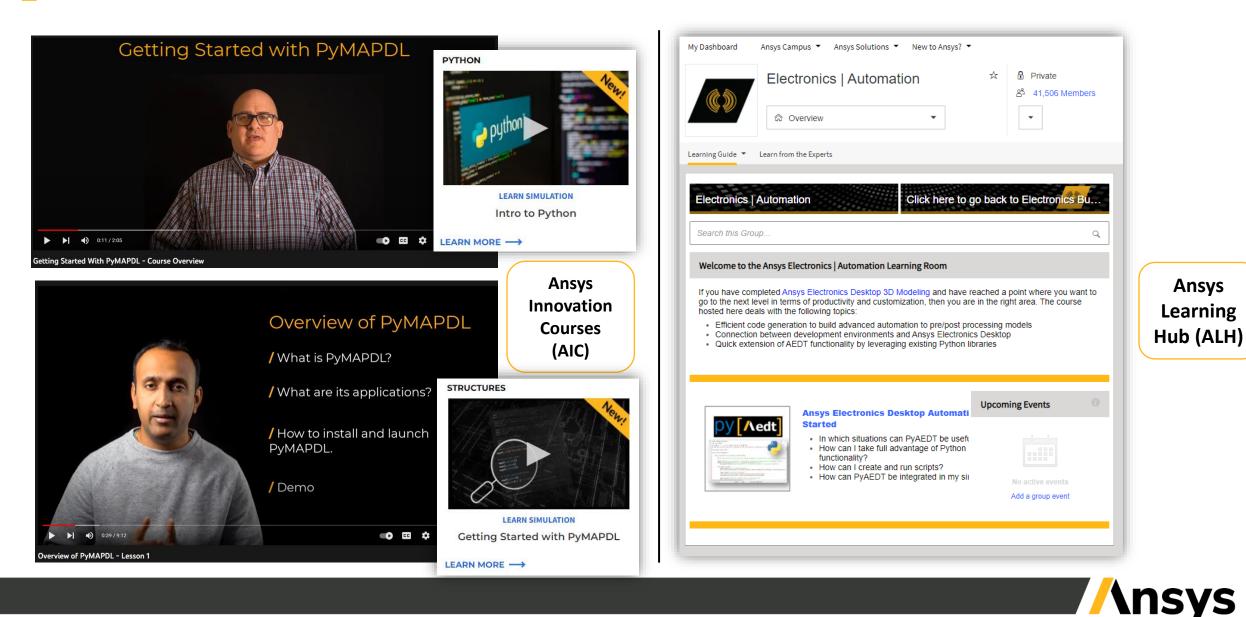
HOME + I	NGINEERING SIMULATION
+	1 2 3 4 5 6 *
how to	control contour in PyDPF plot
	pf.docs.pyansys.com/examples has quite some useful example but I couldn't find anythin on how to manipulate the contour. For banded/smooth, number of bands, their colors, min and max
E	Answered 🗸 (PyDPE) 17 views 1 comments 0 reactions Started by Pavel Most recent by Ayush Kumar Jan 22, 2023. Engineering Simulation
Retriev	e messages from Mechanical
Through	ACT scripting, how can I retrieve the info, warning and error messages in Mechanical ?
•	Answered V AnsysACT 16 views 3 comments 0 reactions Started by Pernelle Marone-Hitz Most recent by Pernelle Marone-Hitz Jan 20, 20 Engineering Simulation
How to	get results with DPF for a time/frequency not available in my rst file?
For insta how can	nce, let's imagine that our harmonic analysis contains results for frequency 12Hz and 15Hz. If I want to get a result for 14Hz and 14.5 I do It?
	OPF) 17 views 1 comments 0 reactions Started by Javier Vigue Most recent by Javier Vigue Jan 17, 2023 Engineering Simulation
How ca	n I read a file using Python?
How car	Fread a file using Python?
a	Answered 🗸 IronPython 53 views & comments 2 reactions Started by Pernelle Marone-Hitz Most recent by Pernelle Marone-Hitz Jan 12, 2

Ansys Developer Forum: https://discuss.ansys.com/



©2023 ANSYS, Inc.

## Trainings (AIC & ALH)



#### **PyAnsys: Module Cheat Sheets**

Ansys

Ansys

Apply solution settings

number of iterations=158

/ Post-Processing

PyFluent allows you to post-pro The following example shows y

surfaces\_list = | "symmetry-xyplane

**Temperature Contour** 

References from PyAnsys Docu

PyFluent allows you to use Solver settings objects to apply solution settings, initialize, and solve.

olver.solution.initialization.hybrid initialize

solver.results.graphics.centour["contour"]
solver.results.graphics.centour["contour"]
print\_state()
solver.results.graphics.centour["contour"]

#### Cheat sheet for PyMAPDL

Converting an existing APDL script to PyMAPDL format	/ Post-Processing Class This class is used for plotting and saving results to arrys. mspfl.est(1, 2) mspfl.est(1, 2) msp	
<pre>pyscript='pyscript.py' mapdl.convert_script(inputfile, pyscript)</pre>		
/ MAPDL Class		
<pre>mapdl.load_table(name,array,var1='', var2='',</pre>		
To access from or write parameters to MAPDL database	<pre># Plot contour legend using dictionary mapdl.allsel() sbar_kwargs=("color": "black",</pre>	
<pre># Save a parameter to a RusPy array nparray nparray=mapdl.parameters["displ_load"]</pre>	1997 wargset (color 1 million) wert (color 1 million) wert (color 1 million) mapell post, processing, plot, nodal, eye, stress edge, color black', shew, deges Tree, ecalar bar_angresher, kwengs, n.color // Plotting Class Potting interpolated with PAViata by swing the stress and storing within the underlying Unitary stress and storing within the underlying th	
<pre># Create a parameter from a NumPy array mapray mapdl.parameters["exp_disp"]=mparray</pre>		
To access information using "GET and "VGET directly to NumPy arrays		
# Runs <get a="" and="" command="" python="" returns="" value<br="">mapdl.get_value(entity='', entnum='',</get>		
item1='', itlnum:'', item2='', it2num:'', **kwargs)		
<pre># Runs +VGIT command and returns a Python array mapdl_get_array(entity='', entnum='', iten2='', iten1='', klopet'', +then2='', it2num='', klopet'', ++kwarga)</pre>	<pre>pl:ppvista.Plotter() pl@mapdl.post_processing.plot_nodal_stress     return_plotter*True) pl.add(pl0.mesh) pl.adws()</pre>	
/ Mesh Class Store the finite element mesh as a VTK UnstructuredGrid data object.	<pre># Plot the currently selected elements mappl.eplot(show.node_numbering, vtk) # Plot the selected volums</pre>	
grid-mapdl.mesh.grid	mapdl.vplot(nv1, nv2, ninc, degen, scale,	
Save element & node numbers to Python arrays.	<pre>npdLaplet(nal, nal, nic, diegen, scile f Display the selected lines without # AMPOL plat symbols napdL.lpid(twi=True, cpos"sy', 1 ine_width # Same png file of time plat with MAPOL # coordinate symbol napdL.lpid(tck=false)</pre>	
nodes=mapdl.mesh.nodes		
node_num~mapdl.mesh.nnum # Save node mathers of all nodes to array		
# Element mumbs, of currently selected elements elem_nummapdl.mesh.enum # All element mumbers incl, those not selected	References from PyMAPDL Document - Getting Started - MAPDL Commands	
	Imperfering inspection (in the inspection of the	

**Cheat sheet for PyFluent** 

Solver Settings Object Interface

initial, name="water-liquid")
ietup.cell\_zone\_conditions.fluid["elhow-fluid
iterial = "water-liquid")

/ Define materials

ample shows how you u

tup.boundar ["].vmag = {

solver.setup.boundary\_condit

fy cell zone conditions

ver.setup.boundary\_conditions.ve

].ke\_spec = "Intensity and Hydrauli solver.setup.boundary\_conditions.ve

].turb\_hydraulic\_diam = "4 [in]"

er.setup.boundary\_conditions.velocit inlet"].t = { "option": "constant or expression", "constant": 293.15,

The examples in this section show how you use 5 settings objects to modify **cell zone** conditions.

solver.setup.cell\_zone\_conditions.fluid["elbow-flu = ("laminar": True)

Getting Started with PyAnsys / PyAnsys on GitHub / Visit

Define boundary conditions

ples in this section show how you up objects to define boundary condition

/ Launch Fluent locally

import ansys.fluent.core as pyfluent solver = pyfluent.launch\_fluent(mode = "solver" show\_gut = True)

The solver object contains attributes such as file, setup, solution, and results, which are also instances of setting

/ Import mesh in launched session

import\_filename = 'example\_file.msh.h5'
solver.file.read(file\_type="case", file\_name="case", file\_name="case"", file\_name="case"", file\_name="case", file\_name="case"", file\_name="case", f

There are other specific APIs available for reading ca and reading case-data files.

import\_filename = 'example\_file.cos.h5'
solver.file.read\_cose(file\_type="case", file\_t
import\_filename)

/ Enable heat transfer physics

solver.setup.models.energy.enabled = True

ng the object state with using pprint // Intercept and sequences of the sequence of the sequenc

an be sent to

import\_filename)

PyAE	EDT EDB-API Cheats	heet //nsys	
/ Launching EDB-API using PyAEDT EDB manager manages the AEDB Database. An AEDB database is a folder that contains the database representing	<pre># List of all mets in available in AEDB file edb.mets.metList T = delets a net</pre>	<pre>setup.add_source_terminal_to_ground("VI", i) solve_edb = edb.solve_sivave()</pre>	
any part of a PCB. It can be opened and edited using the Edb class.	<pre>edb.mets["net_name"].delete()</pre>	/ Simulation Configuration These classes are the containers of simulation configura	
9 To lunch an instance import pyaedt Edb (edbrersion="2023.1", edbpath= aedb_path) edb.aya edb() # Save the edb file	/ Vias and padstacks These containers contains the API references for padstack management. The main padstack object is called directly from main application using the property padstacks.	constructors for the EDB. # AC settings Sim_setup.ac_settings.start_freq = "100Hz" sim_setup.ac_settings.stop_freq = "6GHz"	
edb.close_edb() # exits the edb file	<pre># Creating a via edb.padstacks.place(position = [5e-3, 5e-3], "</pre>	sim_setup.ac_settings.step_freq = "10NHz"	
/ Stackup and Layers These classes are the containers of the layer and stackup man- ager of the EDB APL	NyVis") # To get the pad parameters edb.padstacks.get_pad_parameters()	<pre>sim_setup.edbapp.nev_simulation_configuration sim_setup.solver_type sim_setup.SOLVER_TYPE. Sim_setup.batch.solve_settings.</pre>	
<pre># Adding a stackup layer edb.stakup.add_layer("Name") # To get the names of the all layers edb.stackup.stackup_layers.keys()</pre>	/ Sources and Excitation These classes are the containers of sources methods of the EDB for both HFSS and Siwave	<pre>nlm_crist.and/ani/14_sphiles01 sin_setup_batch_colve_settings. dc_crist.retubescips = True sin_setup_nes_defailt_cristor = File sin_setup_nes_defailt_cristor = File sin_setup_hatch_colve_settings_signal_set sing_the_clist sin_setup_hatch_colve_settings_components component_list sin_setup_hatch_colve_settings_prever_ness prev_retup_list</pre>	
/ Modeler and primitives These classes are the containers of primitives and all relative methods. Primitives are planes, lines, rectangles, and circles.	# To get the Dicticeary of EDB excitations edb.excitations # To create differential port edb.hfms.create.differential_wave_port( positive_printive_id = trace_p[0].id, positive_points.or_edde = bpoints.		
8 Creating a polygon by defining points points = [[0.0, 1e-3], [0.0, 10e-3], [100e-3, 10 e-3], [100e-3, 1e-3], [0.0, 1e-3]] edb.modeler.create.polygon_from_points(points,	<pre>prostive_prints_un_edge = pi_prints, negstive_prints_on_edge = ni_prints, name = "vare_port_i")</pre>	<pre># Saving config file sim_setup.export_json(os.path.join(project_ps</pre>	
layer_same = "Name") / Components	/ Simulation Setup These classes are the containers of setup classes in EDB for both HFSS and Siwave.	/ SiWave Manger Siwave is a specialized tool for power integrity, signal integ	
Component class contains API references for net manage- ment. The main component object is called directly from main application using the property components.	<pre># HFSS simulation setup setup = edb.create_hfss_setup(mane = "my_setup") setup.set_solution_single_frequency()</pre>	and EMI analysis of IC packages and PCB. This tool so power delivery systems and high-speed channels in electr devices. It can be accessed from PyAEDT in Windows on setups can be implemented through EDB API.	
<pre># To get the list of components eds.cemponents.cempsents.keys() # To get the net information of a component edb.cemponents.get_component_net_connection_info (*Q13#=)</pre>	<pre>setup.hfss.solver_settings. enhanced.low_freq_accuracy - True setup.hfss.solver_settings.order_hasis = "first" setup.adsptive_settings. add_adaptive_frequency_data("SGMI", \$, "0.01")</pre>	from pyaedt.sivave import Sivave # this call returns the Kab class initialized 2003 %1 sivave - Sivave(specified_version="2023.1") sivave.ope.project("spyroject.siv")	
/ Nets	<pre>\$ SiVave Simulation setup setup = edb.sivave.add_sivave_dc_analysis(name =</pre>	siwave.export_element_data("mydata.txt") siwave.close_project()	
The Net class contains API references for net management. The main net object is called directly from main application using the property nets.	"myDCIR_4") setup.use_dc_custom_settings = True setup.dc_slider_position = 0	References from PyAEDT Documentation	

letting Started With AFDT / Ansys In

/ Launching PyAEDT To launch HFSS instance locally and exit it	/ Creating boundaries In AEDT, open region is created as	/ Mesh Class Mesh module manages the mesh functions in PyAEDT	
hfss = pysedt .Hfss(specified_version="2023.1", non_graphical=False.	hfss.create_open_region(Frequency='1GEz')	hfss mesh assign_initial_mesh_from_slider(1 =6) # setting the slider level to 6	
	Assigning the radiation boundary to a box is as follows	# assign model resolutions hfor mech assign_model_resolution(names=[obje	
<pre>new_desktop_session=True. projectname="Project_name". designname="Design_name") # To exit the insurance</pre>	hfss.assign_radiation_boundary_to_objects(" airbox")	.name, object2.name], defeature_length=No # assigning the mesh length to the object1 fa hfas mesh assign_length_mesh(names-object1 fa . isinside=False, maxlength=1, mame1=2000	
ht ss. release_deskt op () / Variable Class Creating a local and global variable in HFSS	/ Port definitions Common port types in HFSS are Lumped-port and Wave guide-port. Pythonic way of defining the lumped-port is	Analysis Class     The solution setup (#7.50 testign is analyzed by	
hfss["din"] = "lmm" # design variable hfss["ddin"] = "lmm" # project variable	box1 = hfss.modeler.create_box([0,0,50], [10,10,5], "Box1","copper")	ing the following python command	
Af es writeble_manager is the class handles all the variables. It contains a lot of properties and methods useful for variable handling. // Material Class	bot2 = hfss modeler create_box([0,0,00], [10,01.05]. "Box2", "copper") port_L = hfss.lmmed.port(signal="Box1", reference"Box2", integration.line.hfss. ArisDir.XMsg. inpedance.60, name="LimpedFort ", resortalize=free.deshed=False")	Post-processing Class Post-processing class has modules for creating and ediplots in AEDT. They are accessible through the post library	
ht ss.materials class is used to access the material library. A new material is added in HFSS as	Pythonic way of defining Wave guide-port is	<pre>plotf = hfss post.create_fieldplot_volume(</pre>	
<pre>ny_mat = hfss.materials.add_material("myHat") ny_mat.permittivity = 3.6 ny_mat.conductivity = 450000 ny_mat.permeability = 1.5</pre>	port_W = hfss.wave.port("Borl", "Borl", name=" NavePort", integration_line=1)	intrinsic_dict) # This call return a FisidFlat object my_data = hfss_pest_get_solution_data(express =trace_manes) # This call return a	
/ Geometry Creation M sa modeler contains all properties and methods needed to	Setup Class     Setup class is used to define the solution setup in the PyAEDT	<pre>standard(report = hfss.post.report_by_categ standard(report = hfss.post.report_by_categ standard('db(S(1.1)')') # This call returney standard</pre>	
edit a modeler, including primitives methods and properties. A box is drawn at (x_pos, y_pos, z_pos) position with (x_dim, y_dim, z_dim) dimensions as	setup = hfss.create_setup("HySetup') setup.props("Frequency") = "SONHz" setup("MarinunPasce") = 10	standard_report_create() # This call reate a report sols = standard_report_get_solution_data()	
<pre>box = hfss.modeler.create_box([x_pos_y_pos_z_pos ]. [x_din,y_din,z_din].name="airbox", matname="air")</pre>	<pre>hfss.create_limear_count_sweep(setupname="any"; unit="NNA".freqstart=0.f.freqstop=100, num_of_freq_points=100, sweepname="sweep1", sweep.trpe="Interpolating", save_field="</pre>	/ Calling AEDT-API with PyAEDT Most of the critical functionalities are captured in PyA	
A spiral geometry made of "copper" is created as follows	False)	However, the missing features from AEDT-API method can converted to PyAEDT method.	
<pre>ind = hfss modeler create_spiral( internal_radius=rin, width=width, opacing=spacing, turns=Nr.</pre>	Parametric sweep and optimizations are accessed using following classes	Example: AEDT-API module (Optimetrics), is accessible PyAEDT as	
faces=Np, thickness=thickness, material="copper",mame="Inductor1")	# returns the ParametericsSetups Class hfss.parametrics	omodule = hfss.odesign.GetHodule("Optimetrics	
box, ind are the Object3d objects, containing a lot of properties and methods related to that object including faces, vertices, colors, and materials.	# returns the OptimizationSetupe Class hfss.optimizations	References from PyAEDT Documentation	

PyDynamicReporting Cheat sheet //nsys			
Ansys Dynamic Reporting Service To launch and stop a local ADR Service on a new database:	"key": "child_parent", "name": "& child parent", "value": "Parents can have values",	<pre>server.put_objects(template_1) template_2=server.create_template(name="Plot",</pre>	
<pre>tigort angu.dynamicropering.core as add dw.dir e r(c)trolwy.local durinetry: angu.nu = r(c)tronger files.nu; already dr_service = dw.Service( angu.sinstallationangu.ins, db.directory.db.dir) sestoru.guid = dw.service.start(create_dbwTrue) * To stop the service dr_service.stop()</pre>	<pre>value:: "Find rest can have values", "state" [collapsed]) tree r[ (%re.apped] (%re.iprot, "nose: "Top level", "value": None, "children": troot, "nose: "Top level", "value": None, "children": trait". "expanded")) my_tree = adm_service.treat_ites(obj_nose*Tree") my_tree.iter.tree = tree</pre>	<pre>template_arserver.create_template(name="rist", parentramplate_g, report_type="layout;pace") template_2.set_filter("A(i_type cont table;") server.put_objects(template_g) server.put_objects(template_g)</pre>	
		Visualize items and reports     To search for text items:	
To connect to an already running ADR service on port 8000: Import ansys.dynamicreporting.core as adr	/ Set plot properties To display a table item as a plot:	<pre>all_items = adr_service.query() only_text_items = adr_service.query(filter=</pre>	
<pre>ansys_ins = r*C:\Program Files\Ansys Inc\v241* adr_service = adr_Service(</pre>	<pre># Set visualization to be plot instead of table my_table,plot = 'line' Set &amp; axis and axis formatting my_table.xaris = 'Row 1' my_table.format = 'floatdot1'</pre>	To visualize the first item returned:	
ansys_installation=ansys_ins) adr_service.connect(url="http://localhost:8000") / Create new items		only_text_items[0].visualize()	
		To visualize all the items in the report listed one after the other:	
To create new items in the database of different types:	/ Tag items	adr_service.visualize_report()	
# Create text item	To set tags on an item:	To list all the reports.	
<pre>my_text = adr_service.create_iten(obj_name='Text') my_text,item_text = "<hl>Simple Title</hl>Abc"</pre>	my_text.set_tags("tagl=one tagl=two tagl=three")	all_reports = adr_service.get_list_reports()	
e Create table item Teopri numpy as no	To add or remove tags on an item:	To view a report with a specific name:	
<pre>mpuble = add_service.creats[ten(dg]_name='Table'] wy_table.table(cf("realbls]) = (Thut 1", "Row 2") wy_table.table.ten table = rep.array([ [tit, "tit, "tit, "tit," add_servic(")] [tit, "tit, "tit," add_servic(")] [tit, "tit," add_service.table.table(")] [tit, "tit," add_service.table(tit,")] [tit, "tit," add_service.table(tit,")] [tit, tit," add_service.table(tit,")] [tit," tit," add_service.table(tit,")] [tit," tit," tit," add_service.table(tit,")] [tit," tit," tit," add_service.table(tit,")] [tit," tit," tit," add_service.table(tit,")] [tit,"]]</pre>	<pre>my_text.add_tag(tag*'tag4', value*'four') my_text.rem_tag("tag1")</pre>	<pre>report_by_name = adr_service.get_report(     report_name='Ny Report')</pre>	
	Create report templates To create report templates use the ADR low level API. The first step is to define the server object:	report_by_name.v1sua11ze()  / Get url for items and reports To get the URL from the first item returned:	
<pre># Create 3D item scene = adr_service.create_item(obj_name='3D Scene')</pre>	server = adr_service.serverobj	only_text_items[0].url	
sceme.item_sceme = r'C:\tmp\test_sceme.avz' # Create a tree item via a dictinnary	Then use the low level API to create a new report template:	To get the URL of a report:	
leaves = []	template_0=server.create_template(name="Hy Report",	report_by_name.get_url()	
<pre>for i in range(b):</pre>	parent-None, report_type="Layoutbas(c") server.put.bjects(template_8) template_liserver.creats_template(name="intro", parentemplate_8.reg_type="Layouttpare") template_lises(fiter(r%))type="Layouttpare") server.put befect(template 8)	References from PyAnsys Documentation - Gotting Started - API reference - Exemples	

#### And many more on....

cheatsheets.docs.pyansys.com

or

developer.ansys.com



		<pre>anly_text_items[0].visualize[) To visualize all the items in the report listed one aft other.</pre>	
<pre>ion=ansys_ins) ct(url="http://localhost:8808")</pre>	<pre>my_table.plot = 'line' # Set X axis and axis formatting</pre>		
items	my_table.xaxis = 'Row 1' my_table.format = 'floatdot1'		
in the database of different types:	/ Tag items	adr_service.visualize_report()	
	To set tags on an item:	To list all the reports	
<pre>rice.create_item(obj_name='Text') = "<hl>Simple Title</hl>Abc*</pre>	<pre>my_text.set_tags("tagl=one tag2=two tag3=three")</pre>	all_reports = adr_service.get_list_reports()	
	To add or remove tags on an item:	To view a report with a specific name:	
<pre>rvice.create_item(obj_name='Table') ct("rouble") = ["Bow 1", "Row 2"] is = np.array([     "4", "5"],     "10", "25"]], dtype=" S20") end create_item(obj_name='Image') '(ct)tem(tem_item_op')</pre>	<pre>my_text.add_tag(tag*'tag4', value*'four') my_text.rem_tag("tag1")</pre>	<pre>report_by_name = adr_service.get_report(     report_name='Wy Report')     report_by_name.visualize()</pre>	
	/ Create report templates To create report templates use the ADR low level API. The first step is to define the server object:	Get url for items and reports     To get the URL from the first item returned:	
ce.create_item(obj_name='30 Scene')	server = adr_service.serverobj	only_text_items[0].url	
<pre>r'C:\tep\test_scene.avz' tem via a dictionary</pre>	Then use the low level API to create a new report template:	To get the URL of a report.	
	<pre>template_0=server.create_template(name="Hy Report", parent=None, report_type="Layout:bas(c")</pre>	report_by_name.get_url()	
key": "leaves", "name": f"Leaf (1)", "key": "child", "name": "Boolean lue": True))	parenti-mome_ireger(_type=layoutleast); serve.put_sjocts(teoplate_s) template_irserver.create_template(name*'intro", parenti-momplate_ir.exper(type='layoutlparel') template_irset_fitter('Ais_type[cont]html,string:") server.put_blets(template_s)	References from PyAnsys Documentat - Getting Started - API reference - Examples	

#### ©2023 ANSYS, Inc.

## Testimonials – Research Community

Pyansys tremendously helped by providing an **easy** way to procedurally read and process data directly from Ansys without any human intervention. Combined with Python's enormous data analysis and plotting capability this helped me automate extraction of knowledge from 100s of giant Ansys simulations and saved a lot of expensive human hours

Ph.D. Student

You can **easily** retrieve node stresses and temperatures. Then outside, there is a material database which you can retrieve yield strength based on temperatures, and finally you can calculate factor of safety. And perhaps you can plot it. Think about you have many analyses in different locations and you can automatize this process

Aerospace Engineer

I was trying to avoid exporting my results to csv and then importing back into python. [I] started to look also into the "controller" features, which I came to like a lot.

I also contributed to pyansys once, to support reading transient analyses with a reduced number of DOFs [...]. As far as I remember it was written in C, which I don't "speak".

Ph.D. Student

Quite **easy** to embed the simulation in different environments. I also have to say, how glad I am that the plotting functionality in pyansys is so much faster and intuitive than its APDL counterpart

Undergraduate Student

data extraction data transforms automation solver embedding

orms <mark>plotting</mark>

user contributions



# Conclusions

- PyAnsys introduces a paradigm shift in how Ansys simulation tools will be used going forward. Ansys is the 'first' simulation software provider to introduce such dynamic interaction with its products.
- PyAnsys has been deployed on **GitHub** and is **controlled by ANSYS**.
- Ability to separate **Pre-processing**, **FE model** and **Post-processing** from outside the Ansys environment is a strength that PyAnsys has and will help us **deploy**, **maintain** and **scale** for applications across various industries.
- AI/ML Community can easily integrate ANSYS physics capabilities into their processes.





