

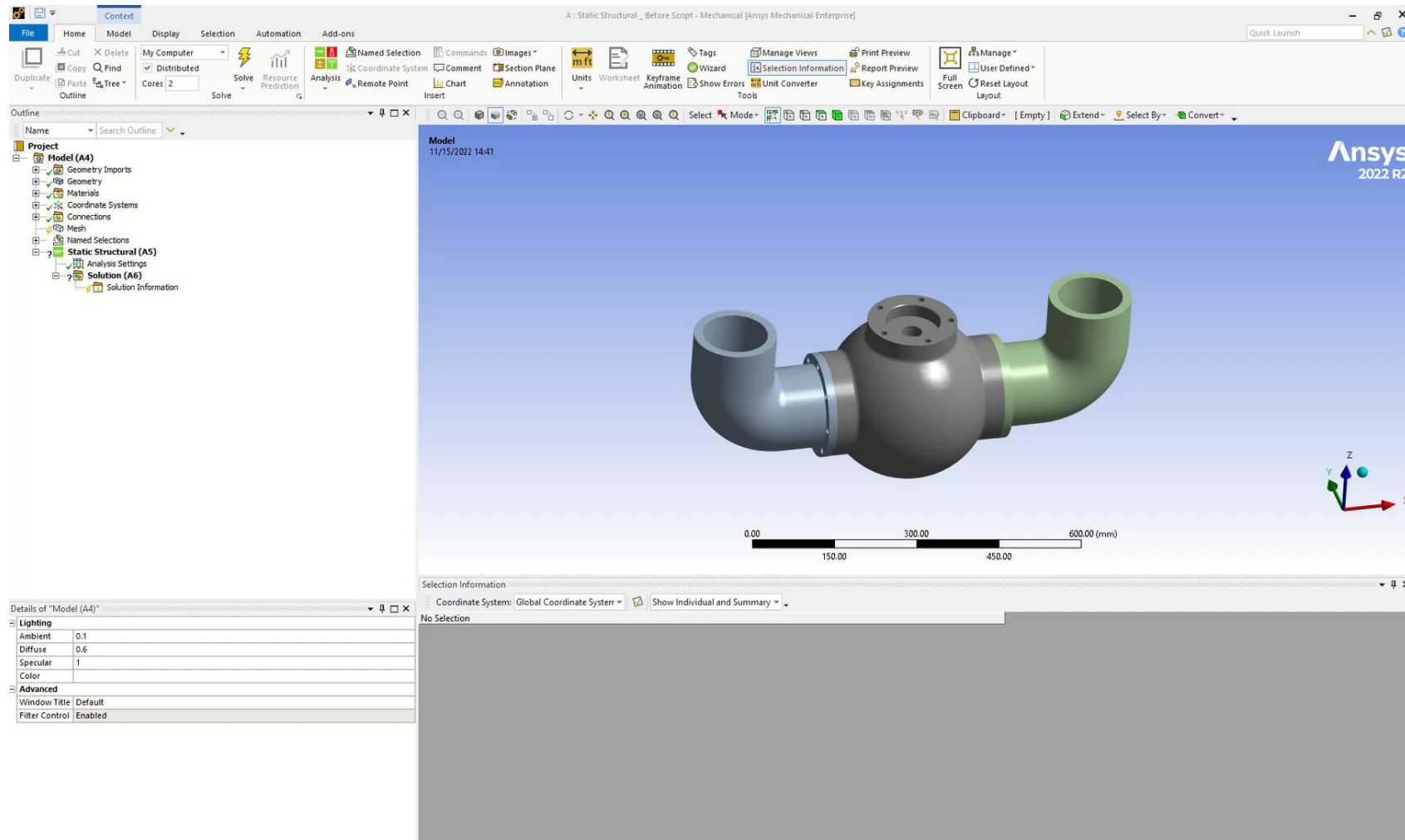


Powering Innovation That Drives Human Advancement

---

# Getting started with Mechanical scripting

# What we are going to learn today: complete model setup automation





# Python 101

# Python in one slide

- **Easy to learn/write/read**

- No variable declaration
- No variable initialization
- Indentation defines where loops/functions end

- **Syntax**

- Case sensitive
- # 1-line comment
- """ Multi line comment """

- **Operators**

- +, -, =, /
- Logical operators: and, or, not
- Relational operators: >, <, ==, !=
- Membership: in, not in
- Identity: is, is not

- **Flow control**

- if, if...else, elif
- while loop
- for loop

```
count = 0
while (count < 9):
    print 'The count is:', count
    count += 1 #same as: count = count + 1

print "Good bye!"
```

```
for letter in 'Python': # First Example
    print 'Current Letter :', letter

fruits = ['banana', 'apple', 'mango']
for fruit in fruits: # Second Example
    print 'Current fruit :', fruit

for index in range(len(fruits)): # Third Example
    print 'Current fruit :', fruits[index]
```

- **Functions**

- Begin with keyword: def

```
# Function definition is here
def printinfo( name, age ):
    "This prints a passed info into this function"
    print "Name: ", name
    print "Age ", age
    return

# Now you can call printinfo function
printinfo( age=50, name="miki" );
printinfo( "Raj", 30 );
```

- **Data types**

- Numbers, strings, lists, dictionaries, tuples

```
list1 = ['physics', 'chemistry', 1997, 2000];
list2 = [1, 2, 3, 4, 5];
list3 = ["a", "b", "c", "d"];

print "list1[0]: ", list1[0] # list1[0]: physics
print "list2[1:5]: ", list2[1:5] # list2[1:5]: [2, 3, 4, 5]
```

```
dict0 = {'Alice': '2341', 'Beth': '9102', 'Cecil': '3258'}
dict1 = {'abc': 456};
dict2 = {'abc': 123, 'xyz': 37};

Ndict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'};
print "dict['Name']: ", dict['Name']; # dict['Name']: Zara
print "dict['Age']: ", dict['Age']; # dict['Age']: 7
```

# Learning Python

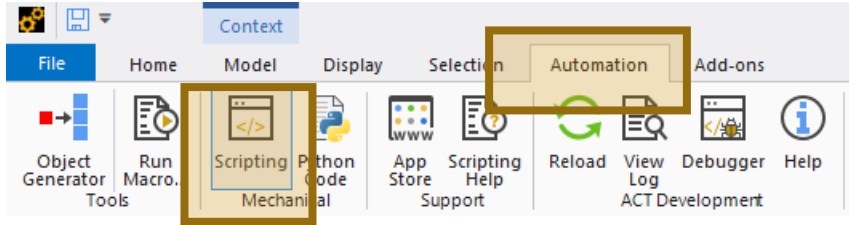
- Many resources available online (LinkedIn Learning, W3Schools, GeekforGeeks, ...)
- One training offered by Ansys on the [Ansys Innovation Space](#)



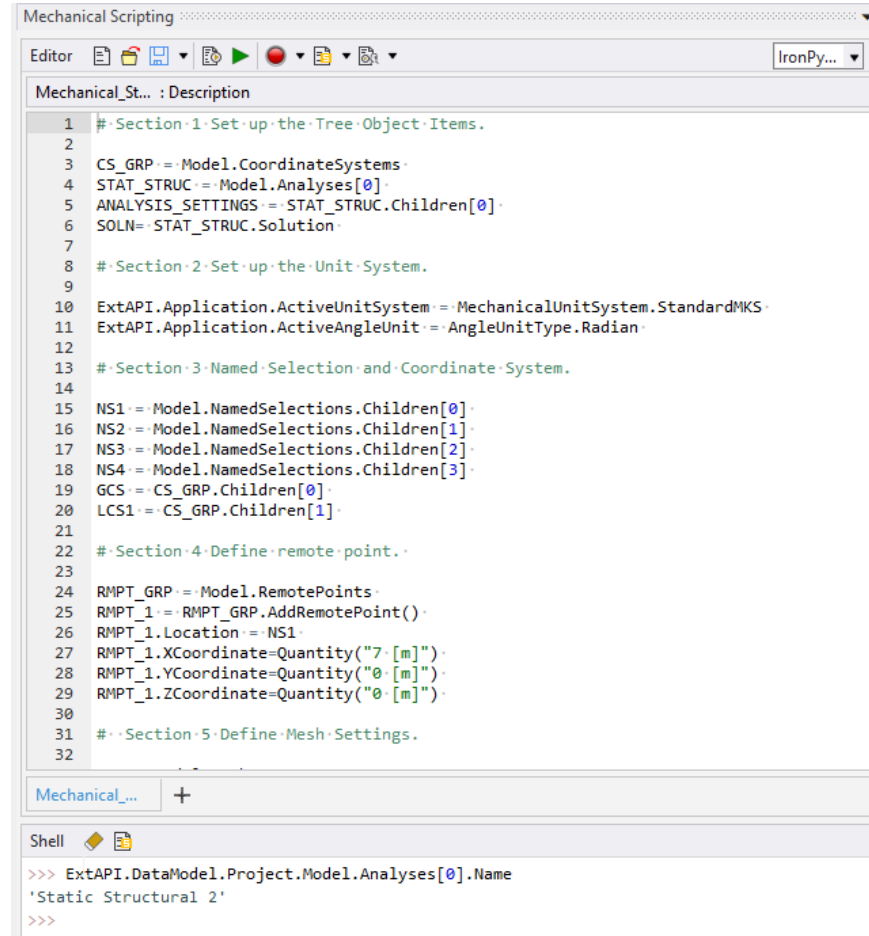


# Using the Mechanical scripting console

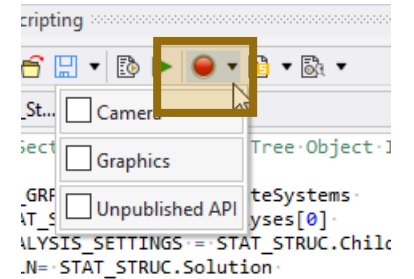
# Mechanical scripting console



*Access Mechanical scripting console*

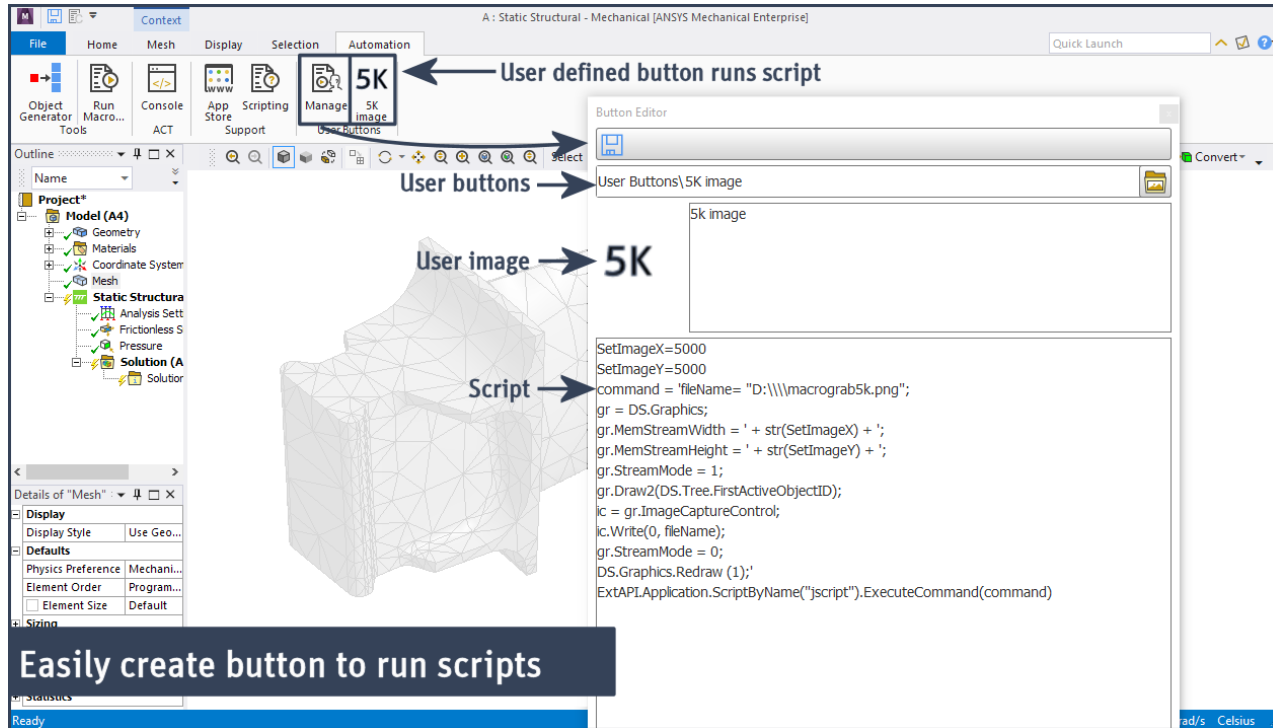


*Script editor and Shell command prompt*



*Recording capabilities*

# More advanced usage



Save script to button  
(Script executed when clicked on)



Execute code automatically during simulation

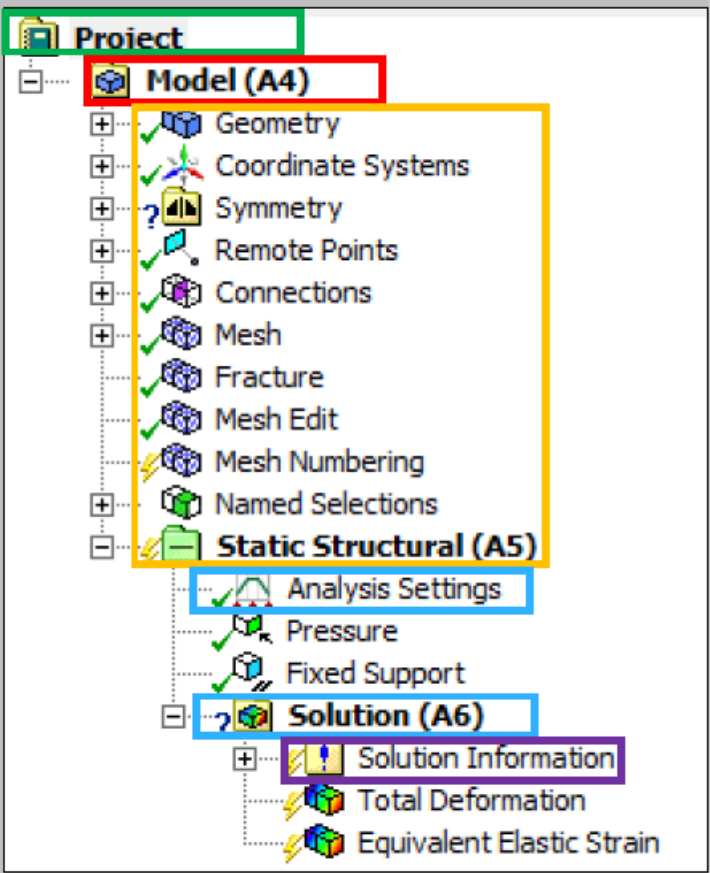
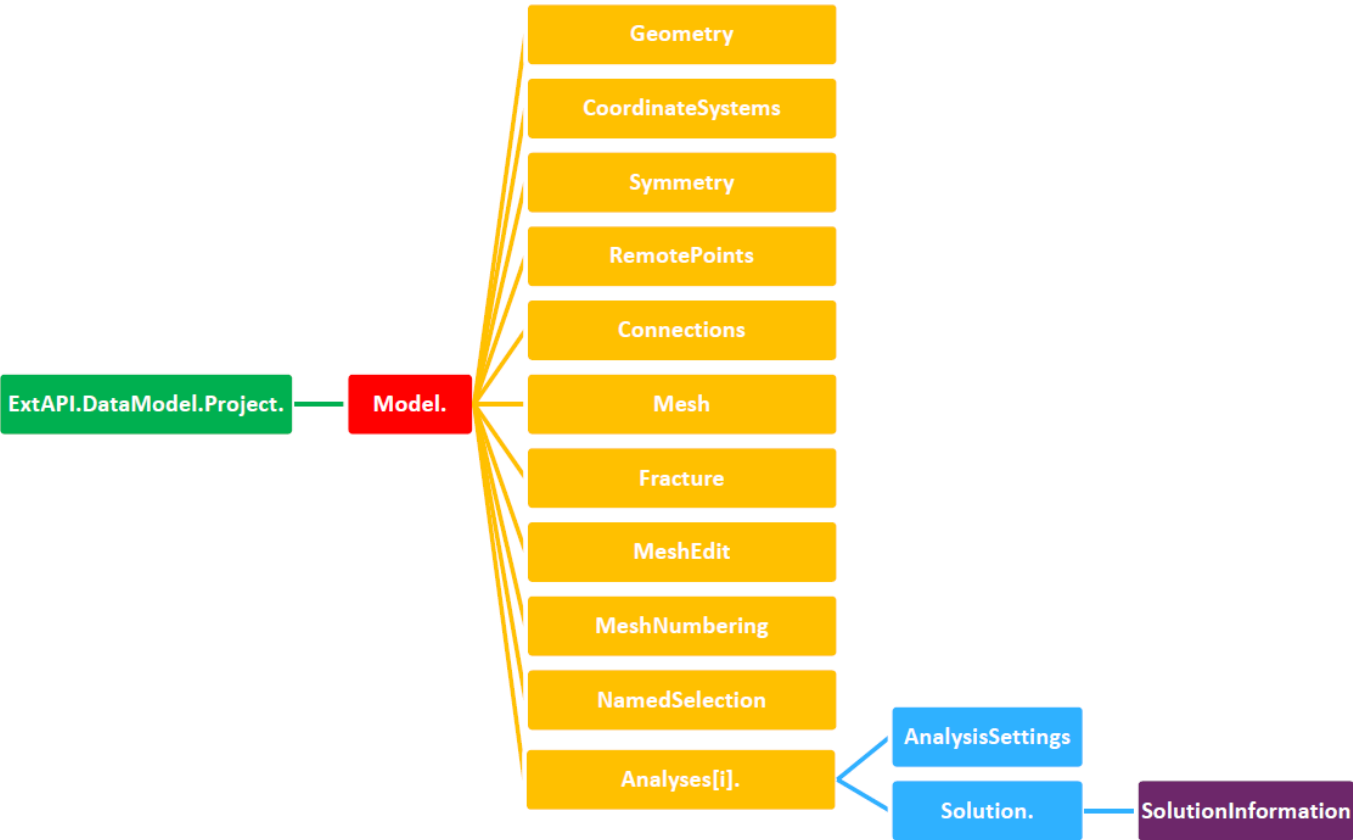




# Understanding Mechanical automation API

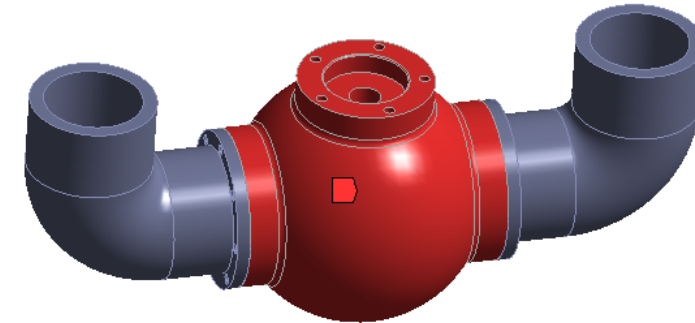
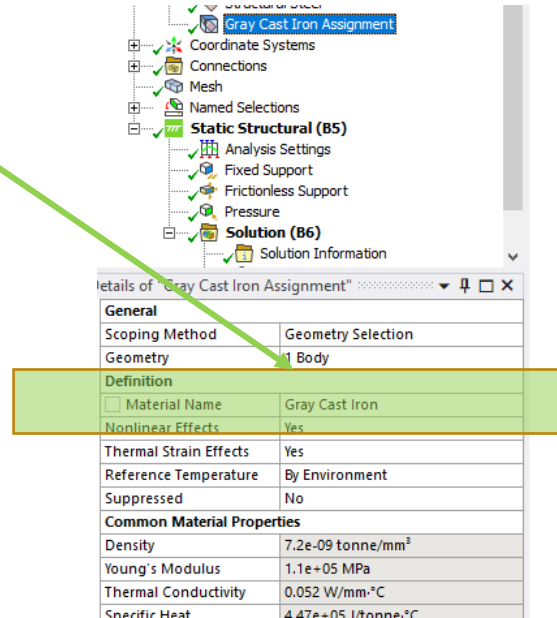
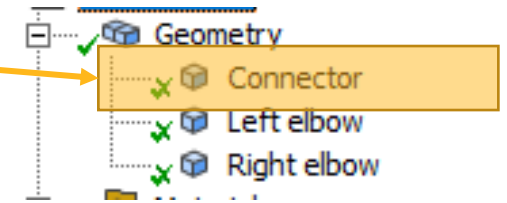
# Mechanical Automation API

```
# Reference model
model = ExtAPI.DataModel.Project.Model
```



# Let's dive into the script: define material assignment

```
# Change material assignment for body named "Connector"
import materials
matAssignment = model.Materials.AddMaterialAssignment()
body = ExtAPI.DataModel.GetObjectsByName("Connector")[0].GetGeoBody()
tempSel = ExtAPI.SelectionManager.CreateSelectionInfo(SelectionTypeEnum.GeometryEntities)
tempSel.Ids = [body.Id]
matAssignment.Location = tempSel
matAssignment.Material = "Gray Cast Iron"
ExtAPI.DataModel.Tree.Refresh()
```



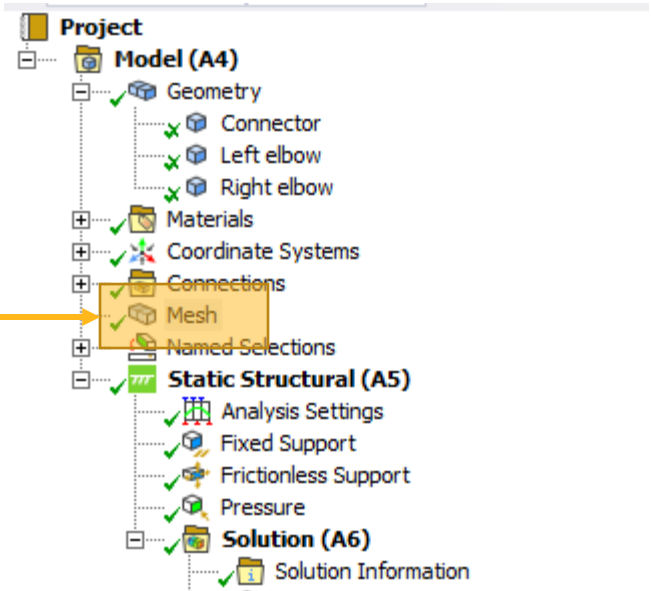
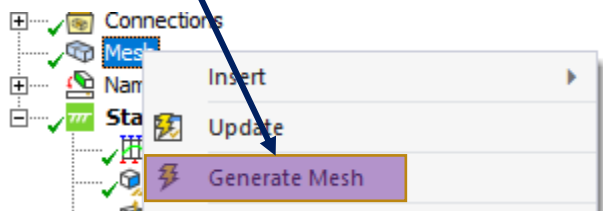
# Let's dive into the script: define mesh settings

```
# Define mesh settings
```

```
mesh = model.Mesh
```

```
mesh.ElementSize = Quantity('25 [mm]')
```

```
mesh.GenerateMesh()
```



Details of "Mesh"	
Display	
Display Style	Use Geometry Setting
Defaults	
Physics Preference	Mechanical
Element Order	Program Controlled
Element Size	25.0 mm
Sizing	
Quality	
Inflation	
Advanced	
Statistics	

# Let's dive into the script: define boundary conditions

```
# Define boundary conditions:
```

```
analysis = model.Analyses[0]
```

```
fixedSupport = analysis.AddFixedSupport()
```

```
fixedSupport.Location = ExtAPI.DataModel.GetObjectsByName("NSFixedSupportFaces")[0]
```

```
frictionlessSupport = analysis.AddFrictionlessSupport()
```

```
frictionlessSupport.Location = ExtAPI.DataModel.GetObjectsByName("NSFrictionlessSupportFaces")[0]
```

```
pressure = analysis.AddPressure()
```

```
pressure.Location = ExtAPI.DataModel.GetObjectsByName("NSInsideFaces")[0]
```

```
pressure.Magnitude.Inputs[0].DiscreteValues = [Quantity("0 [s]"), Quantity("1 [s]")]
```

```
pressure.Magnitude.Output.DiscreteValues = [Quantity("0 [Pa]"), Quantity("15 [MPa]")]
```

Analysis Settings

Fixed Support

Details of "Fixed Support"

Scope	
Scoping Method	Named Selection
Named Selection	NSFixedSupportFaces

Definition	
ID (Beta)	92
Type	Fixed Support
Suppressed	No

Frictionless Support

Details of "Frictionless Support"

Scope	
Scoping Method	Named Selection
Named Selection	NSFrictionlessSupportFaces

Definition	
ID (Beta)	94
Type	Frictionless Support
Suppressed	No

Pressure

Solution (A6)

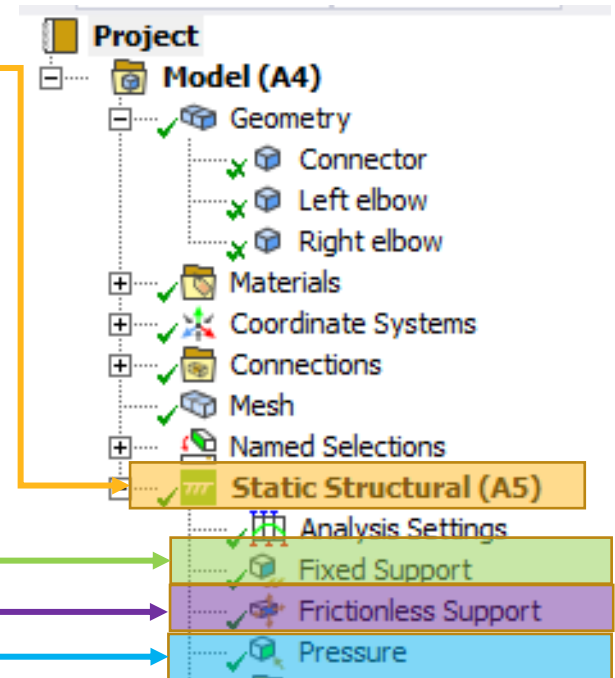
Details of "Pressure"

Scope	
Scoping Method	Named Selection
Named Selection	NSInsideFaces

Definition	
ID (Beta)	96
Type	Pressure
Define By	Normal To
Applied By	Surface Effect
Loaded Area	Deformed
Magnitude	Tabular Data
Suppressed	No

Tabular Data	
Independent Variable	Time

Tabular Data			
Steps	Time [s]	Pressure [MPa]	
1	1	0.	0.
2	1	1.	15.
*			

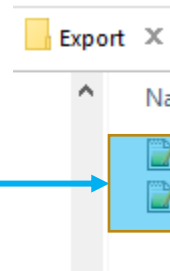
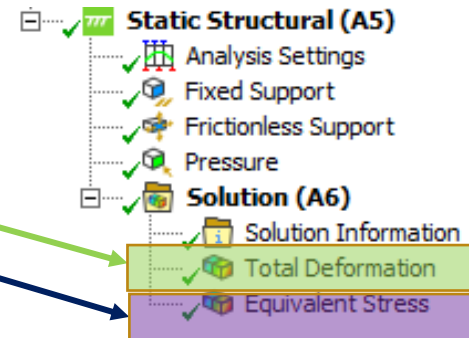
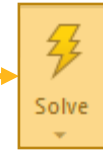


# Let's dive into the script: solve and postprocess

```
# Solve model  
model.Solve()
```

```
# Add results  
solution = analysis.Solution  
solution.AddTotalDeformation()  
solution.AddEquivalentStress()  
solution.EvaluateAllResults()
```

```
# Export result values to a text file  
filePath=r"D:\Export\Result"  
fileExtension=r".txt"  
results = solution.GetChildren(DataModelObjectCategory.Result, True)  
for result in results:  
    fileName = str(result.Name)  
    result.ExportToTextFile(True, filePath+fileName+fileExtension)
```



Node Number	Equivalent (von-Mises) Stress (MPa)
11136	4.9244
11137	5.2469
11138	4.3853
11139	5.9214
11140	5.5261
11141	4.1676
11142	2.8028
11143	3.7985
11144	20.445
11145	22.654
11146	27.851
11147	36.228
.....	..



# Navigating the tree and selecting objects

# Navigating the Mechanical tree

- All objects that exist only once in the tree have a direct access point through `ExtAPI.DataModel.Project.Model`
- For objects that can exist more than once, there can be two methods :

- Analyses are stored in a list:

```
ExtAPI.DataModel.Project.Model.Analyses[0].Name = 'My new analysis name'
```

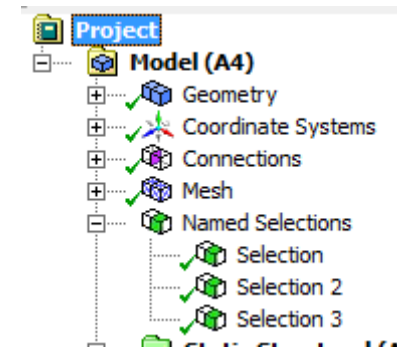
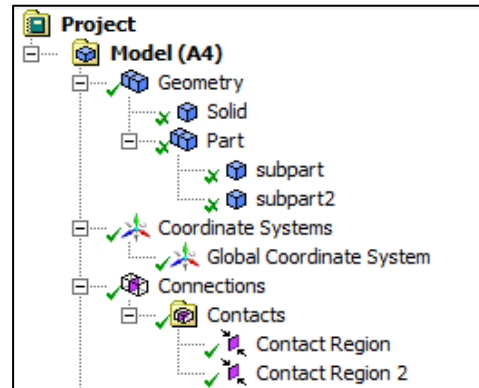
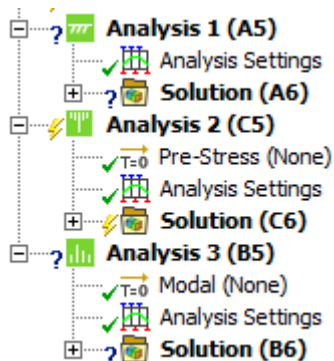
- Other objects: Children objects

```
ExtAPI.DataModel.Project.Model.Geometry.Children[1].Children[0].Name -> subpart
```

```
ExtAPI.DataModel.Project.Model.Connections.Children[0]
```

```
ns = [i for i in ExtAPI.DataModel.Project.Model.NamedSelections.Children if i.Name=="Selection"][0]
```

```
Connections.GetChildren(DataModelObjectCategory.ContactRegion, True)[0]
```

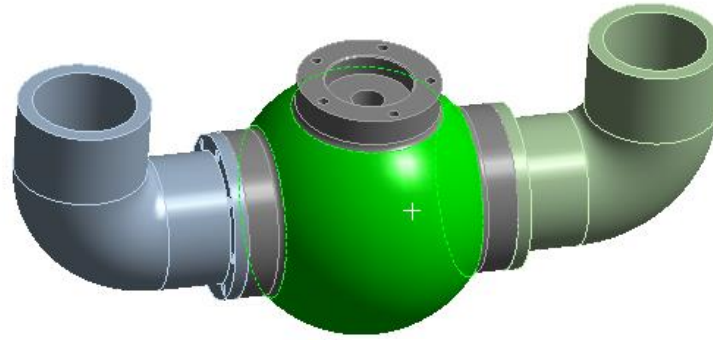




# Selecting entities

- Get IDs of entities currently selected in the UI:

```
ExtAPI.SelectionManager.CurrentSelection
```



- Create a new selection and use it:

```
# create a new empty selection
```

```
tempSel = ExtAPI.SelectionManager.CreateSelectionInfo(SelectionTypeEnum.GeometryEntities)
```

```
# provide list of Ids of entities to select
```

```
tempSel.Ids = [34]
```

```
# create pressure condition
```

```
pressure = ExtAPI.DataModel.Project.Model.Analyses[0].AddPressure()
```

```
# assign location
```

```
pressure.Location=tempSel
```

```
>>> ExtAPI.SelectionManager.CurrentSelection
```

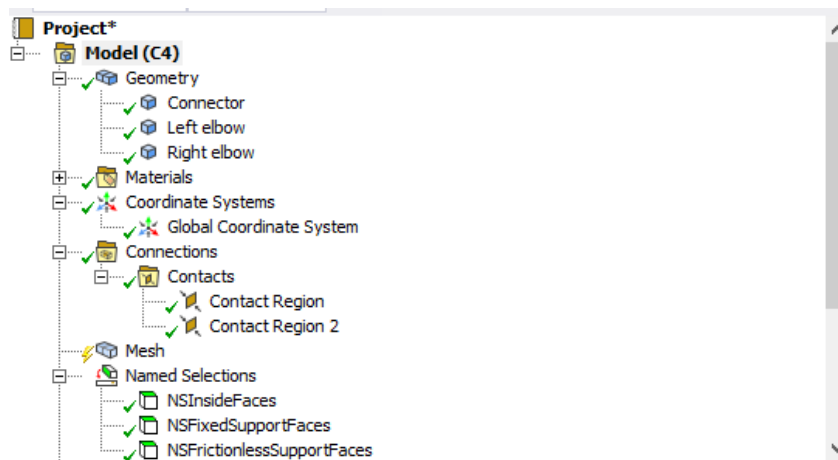
```
[34]
```



# Automate Preprocessing

# Geometry and geodata

`ExtAPI.DataModel.Project.Model.Geometry`

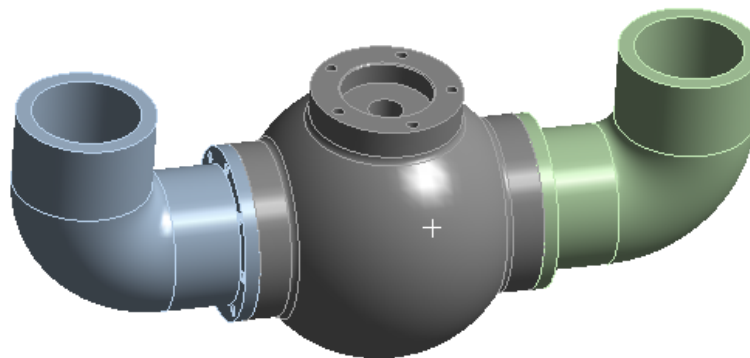


Details of "Connector"	
Graphics Properties	
Definition	
Suppressed	No
ID (Beta)	19
Stiffness Behavior	Flexible
Coordinate System	Default Coordinate System
Reference Temperature	By Environment
Treatment	None
Reference Frame	Lagrangian
Material	
Assignment	Structural Steel
Nonlinear Effects	Yes
Thermal Strain Effects	Yes
Bounding Box	
Properties	
Statistics	
CAD Attributes	
PartTolerance:	0.00000001
Color:	143.143.175

```
>>> geom = ExtAPI.DataModel.Project.Model.Geometry
>>> body = geom.Children[0].Children[0]
>>> body.Name
'Connector'
>>> body.StiffnessBehavior
Flexible
>>> |
```

`ExtAPI.DataModel.Geodata`

Model  
13/04/2021 15:37



```
>>> geodata = ExtAPI.DataModel.GeoData
>>> geobody = geodata.Assemblies[0].Parts[0].Bodies[0]
>>> geobody.Name
'Connector'
>>> geobody.Volume
0.0127583884848715
>>> geobody.Faces.Count
61
```

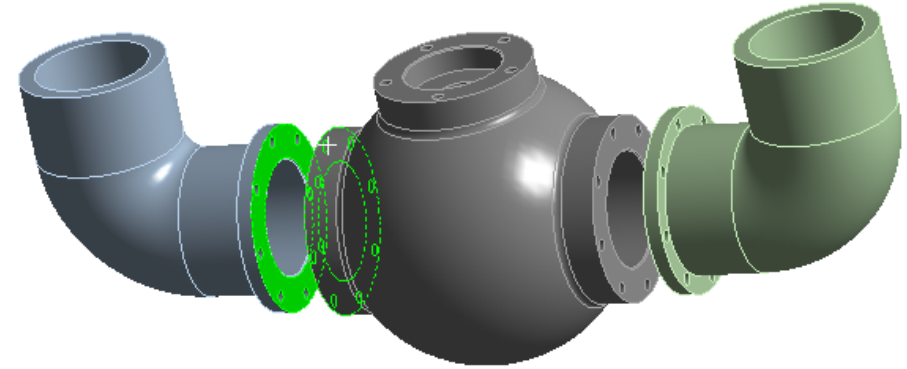
# Connections

- Create a new contact region and define its type:

```
connections = ExtAPI.DataModel.Project.Model.Connections  
contact = connections.AddContactRegion()  
contact.ContactType=ContactType.NoSeparation
```

- Specify contact and target faces (with manual selection of faces):

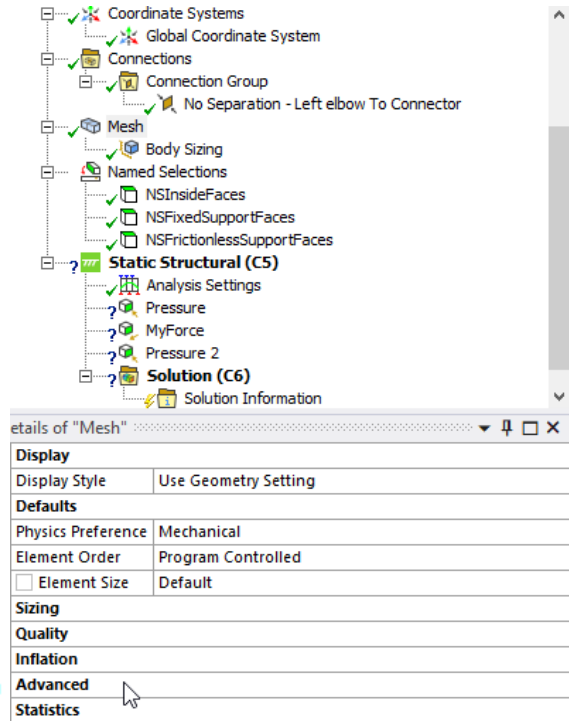
```
contact.SourceLocation = ExtAPI.SelectionManager.CurrentSelection  
contact.TargetLocation = ExtAPI.SelectionManager.CurrentSelection
```



Details of "No Separation - Left elbow To Connector" ▾ 🔍 □ ✕	
[-] Scope	
Scoping Method	Geometry Selection
Contact	1 Face
Target	1 Face
Contact Bodies	Left elbow
Target Bodies	Connector
Protected	No
[-] Definition	
Type	No Separation
Scope Mode	Manual
Behavior	Program Controlled
Trim Contact	Program Controlled
Suppressed	No
[-] Advanced	

# Mesh and mesh data

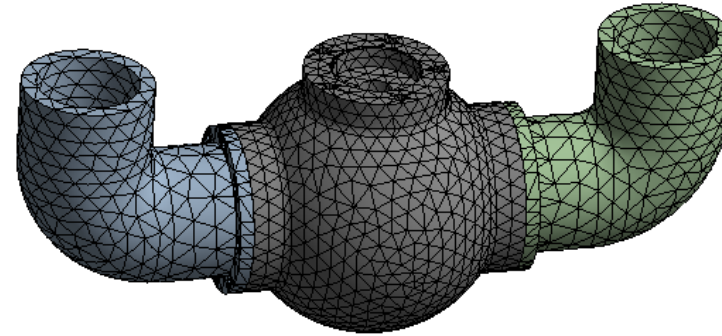
`ExtAPI.DataModel.Project.Model.Mesh`



```
>>> mesh = ExtAPI.DataModel.Project.Model.Mesh
>>> sizing = mesh.AddSizing()
>>> sizing.ElementSize
'0 [mm]'
>>> sizing.ElementSize = Quantity('20 [mm]')

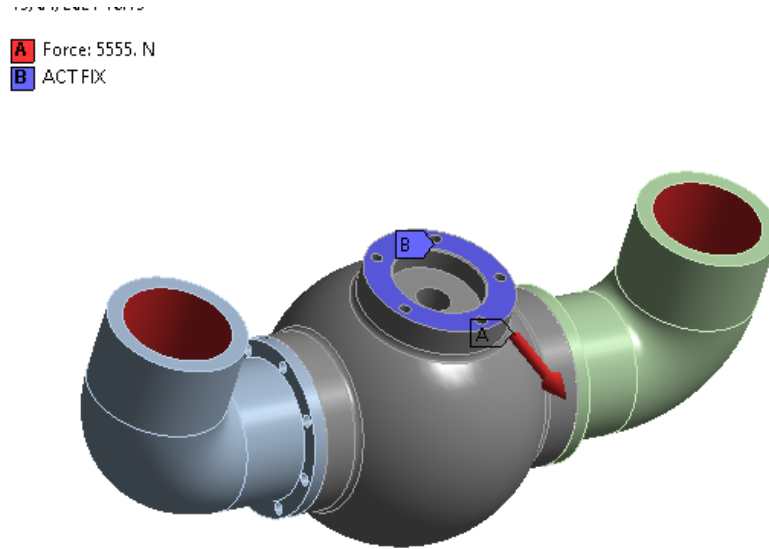
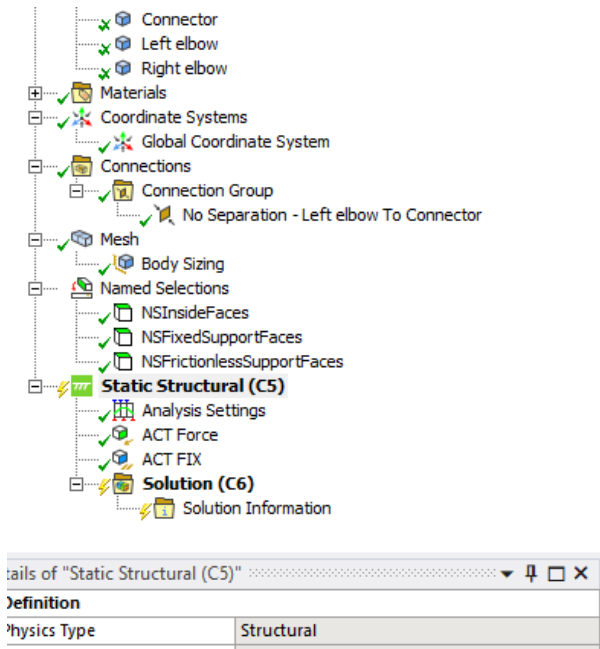
>>> body = ExtAPI.DataModel.GetObjectsByName("Connector")[0].GetGeoBody()
>>> tempSel = ExtAPI.SelectionManager.CreateSelectionInfo(SelectionTypeEnum.GeometryEntities)
>>> tempSel.Ids = [body.Id]
>>> sizing.Location = tempSel
```

`ExtAPI.DataModel.Project.Model.Analyses[0].MeshData`



```
>>> meshData = ExtAPI.DataModel.Project.Model.Analyses[0].MeshData
>>> meshData.ElementCount
21833
>>> meshData.NodeCount
36615
>>> meshData.Elements[0].Centroid
[-0.16265137107062, -0.208395264572271, -0.342956119942903]
>>> meshData.Elements[0].CornerNodeIds
[51, 815, 933, 604]
```

# Boundary conditions



```
FY = -5555
analysis = ExtAPI.DataModel.Project.Model.Analyses[0]
F = analysis.AddForce()
F.DefineBy = LoadDefineBy.Components
F.YComponent.Output.SetDiscreteValue(0, Quantity(FY, "N"))
F.Location = ExtAPI.DataModel.GetObjectsByName("NSInsideFaces")[0]
F.Name = "ACT Force"
G = analysis.AddFixedSupport()
G.Location = ExtAPI.DataModel.GetObjectsByName("NSFixedSupportFaces")[0]
G.Name = "ACT FIX"
```

Details of "ACT Force"	
Scope	
Scoping Method	Named Selection
Named Selection	NSInsideFaces
Definition	
ID (Beta)	119
Type	Force
Define By	Components
Applied By	Surface Effect
Coordinate System	Global Coordinate System
<input type="checkbox"/> X Component	0. N (ramped)
<input type="checkbox"/> Y Component	-5555. N (ramped)
<input type="checkbox"/> Z Component	0. N (ramped)
Suppressed	No

Details of "ACT FIX"	
Scope	
Scoping Method	Named Selection
Named Selection	NSFixedSupportFaces
Definition	
ID (Beta)	121
Type	Fixed Support
Suppressed	No



# Automate postprocessing

# Inserting results

- Create standard results:

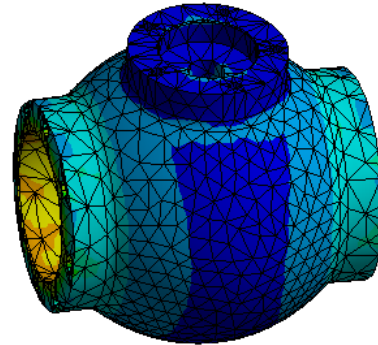
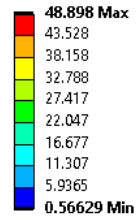
**Project\***

- Model (D4)
  - Geometry
    - Connector
    - Left elbow
    - Right elbow
  - Materials
  - Coordinate Systems
  - Connections
  - Mesh
  - Named Selections
  - Static Structural (D5)
    - Analysis Settings
    - Fixed Support
    - Frictionless Support
    - Pressure
  - Solution (D6)
    - Solution Information
    - Total Deformation
    - Equivalent Stress

**Details of "Equivalent Stress"**

<b>Scope</b>	
Scoping Method	Geometry Selection
Geometry	1 Body
<b>Definition</b>	
Type	Equivalent (von-Mises) Stress
By	Time
<input type="checkbox"/> Display Time	Last
Calculate Time History	Yes
Identifier	
Suppressed	No
<b>Integration Point Results</b>	
Display Option	Averaged
Average Across Bodies	No
<b>Results</b>	
<input type="checkbox"/> Minimum	0.5662877948 MPa
<input type="checkbox"/> Maximum	48.89832556 MPa
<input type="checkbox"/> Average	15.41198206 MPa

**D: Copy of Static Structural \_ After script**  
Equivalent Stress  
Type: Equivalent (von-Mises) Stress  
Unit: MPa  
Time: 1  
13/04/2021 18:10



```
>>> solution = ExtAPI.DataModel.Project.Model.Analyses[0].Solution
>>> solution.AddTotalDeformation()
>>> stressConnector = solution.AddEquivalentStress()
>>> body=ExtAPI.DataModel.GetObjectsByName("Connector")[0].GetGeoBody()
>>> tempSel = ExtAPI.SelectionManager.CreateSelectionInfo(SelectionTypeEnum.GeometryEntities)
>>> tempSel.Ids = [body.Id]
>>> stressConnector.Location = tempSel
>>> solution.EvaluateAllResults()
>>> |
```



# Inserting results

- Create user-defined results:

File Edit View Insert Results Tools Help

**Solution (D6)**

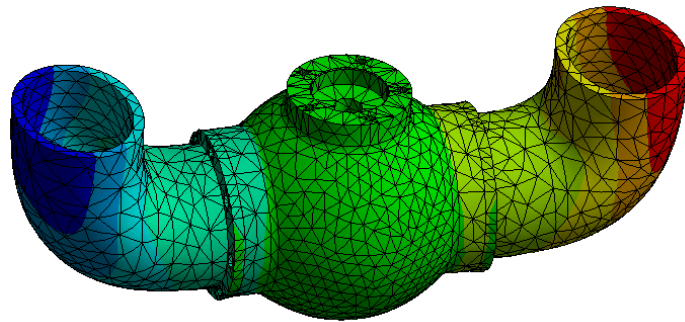
- ✓ Solution Information
- ✓ Total Deformation
- ✓ Equivalent Stress
- ✓ **User Defined Result**

Details of "User Defined Result"

<b>Scope</b>	
Scoping Method	Geometry Selection
Geometry	All Bodies
<b>Definition</b>	
Type	User Defined Result
Expression	= UX
Input Unit System	Metric (mm, t, N, s, mV, mA)
Output Unit	
By	Time
<input type="checkbox"/> Display Time	Last
Coordinate System	Global Coordinate System
Calculate Time History	Yes
Identifier	
Suppressed	No
<b>Results</b>	
<input type="checkbox"/> Minimum	-3.397449851e-002
<input type="checkbox"/> Maximum	3.381570056e-002
<input type="checkbox"/> Average	6.015824653e-006
Minimum Occurs On	Left elbow
Maximum Occurs On	Right elbow
<b>Information</b>	

D: Copy of Static Structural \_ After script  
User Defined Result  
Expression: UX  
Time: 1  
13/04/2021 18:12

0.033816 Max  
0.026283  
0.018751  
0.011219  
0.0036867  
-0.0038455  
-0.011378  
-0.01891  
-0.026442  
-0.033974 Min



0.00 300.00 (mm)  
150.00

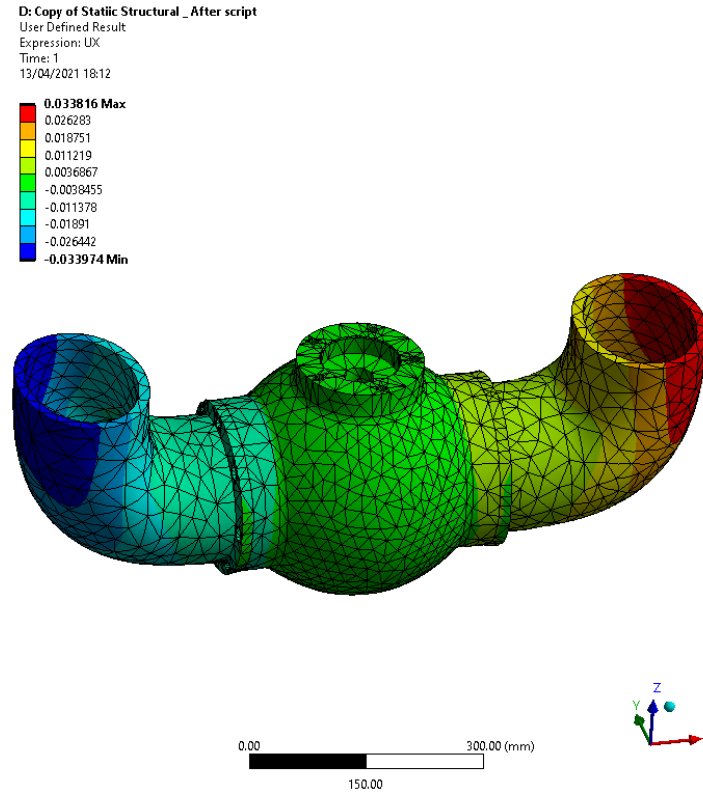
```
>>> UDR = solution.AddUserDefinedResult()  
>>> UDR.Expression="UX"  
>>> solution.EvaluateAllResults()
```

## Accessing plot data

The screenshot displays the ANSYS Workbench environment. On the left, the 'Solution (D6)' tree is visible, showing a hierarchy of solution steps: 'Solution Information', 'Total Deformation', 'Equivalent Stress', and 'User Defined Result'. The 'User Defined Result' step is highlighted with a blue box. On the right, the 'Details of User Defined Result' dialog box is open, showing the configuration for the 'User Defined Result'.

**Details of "User Defined Result"**

<b>Scope</b>	
Scoping Method	Geometry Selection
Geometry	All Bodies
<b>Definition</b>	
Type	User Defined Result
Expression	= UX
Input Unit System	Metric (mm, t, N, s, mV, mA)
Output Unit	
By	Time
<input type="checkbox"/> Display Time	Last
Coordinate System	Global Coordinate System
Calculate Time History	Yes
Identifier	
Suppressed	No
<b>Results</b>	
<input type="checkbox"/> Minimum	-3.397449851e-002
<input type="checkbox"/> Maximum	3.381570056e-002
<input type="checkbox"/> Average	6.015824653e-006
Minimum Occurs On	Left elbow
Maximum Occurs On	Right elbow
<b>Information</b>	



```
>>> UDR.PlotData
```

	Node	Values
0	11136	-2.3445E-05
1	11137	6.6036E-06
2	11138	4.6504E-05
...		
26170	11133	0.0077966
26171	11134	0.0077826
26172	11135	0.0081679

```
>>> |
```

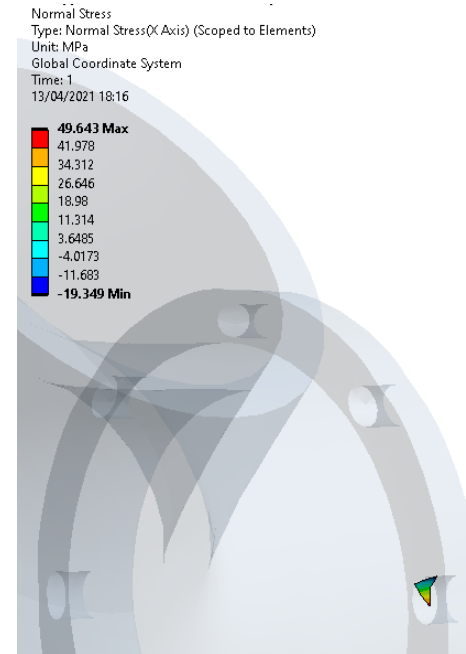
# Access data from the result file

- ❑ Retrieve all components of stress tensor on one element:

```
model=ExtAPI.DataModel.Project.Model # refer to model
reader = model.Analyses[0].GetResultsData() # get results data of first analysis in the tree
StressResults = reader.GetResult('S') # obtain stress results
StressElement1 = StressResults.GetElementValues(1) # retrieve results on element n°1
```

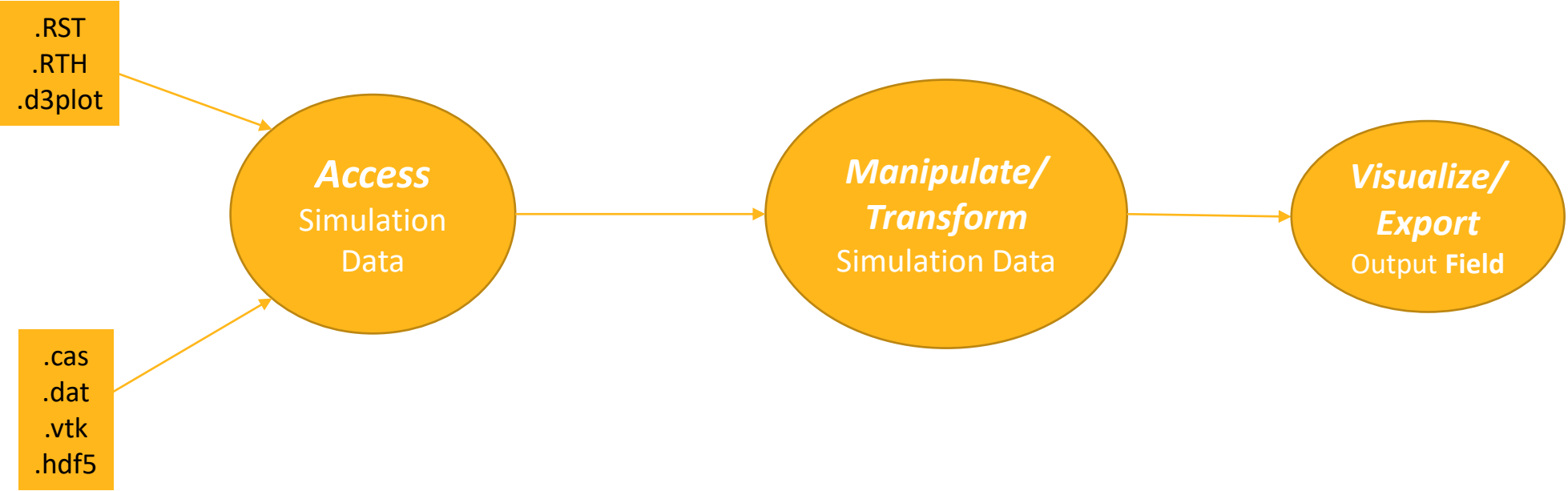
- ❑ Retrieve stress values in one direction:

```
>>> model=ExtAPI.DataModel.Project.Model # refer to model
>>> reader = model.Analyses[0].GetResultsData() # get results data of first analysis in the tree
>>> StressResults = reader.GetResult('S') # obtain stress results
>>> StressResults.SelectComponents(["X"]) # select direction
>>> StressXElement1=StressResults.GetElementValues(1)
>>> StressXElement1
[-0.292112916707993, 27.8284664154053, -19.3488845825195, 49.6434173583984]
>>> |
```

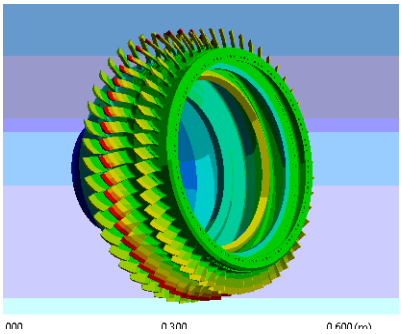


# More more advanced and efficient postprocessing

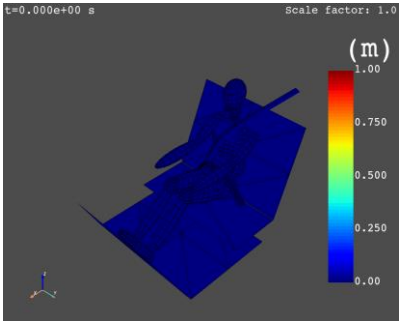
Consider using DPF (Data Processing Framework)



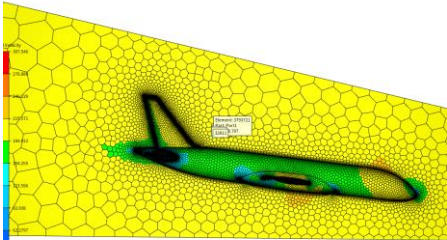
MAPDL



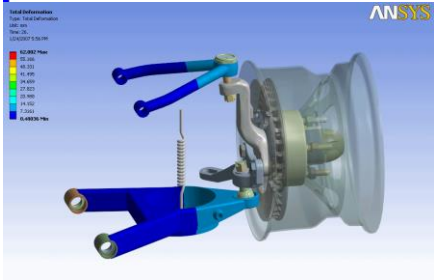
LS Dyna



Fluids



Motion





# Useful resources

# Training material

- On the [Ansys Learning Hub](#)



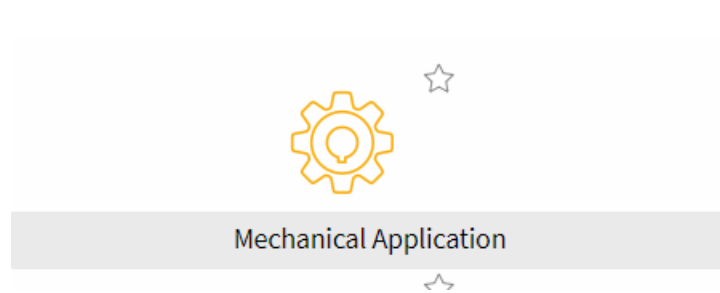
## Ansys Mechanical Scripting

- I want to access the Mechanical API to create my own buttons on the Mechanical user interface, how can I do that?
- Is there a way to modify the solver settings via the Mechanical console?
- How can I create new loads and results objects using the Mechanical API scripting?
- Can I explore the mesh and the geometry programmatically?
- I want to create automatically a Mechanical model from the preprocessing until the postprocessing stages, how can I do that?

Download complete course, Watch previous recordings, Enroll in an upcoming class

# Documentation

- In the [Ansys Help](#)



## ► Documentation

[Mechanical Application Release Notes](#)

[Mechanical User's Guide \(PDF 📄\)](#)

[Mechanical Acoustic Analysis Guide \(PDF 📄\)](#)

[Structural Optimization in Mechanical \(PDF 📄\)](#)

[Explicit Dynamics Analysis Guide \(PDF 📄\)](#)

[Scripting in Mechanical Guide \(PDF 📄\)](#)

[Mechanical Object Reference \(PDF 📄\)](#)

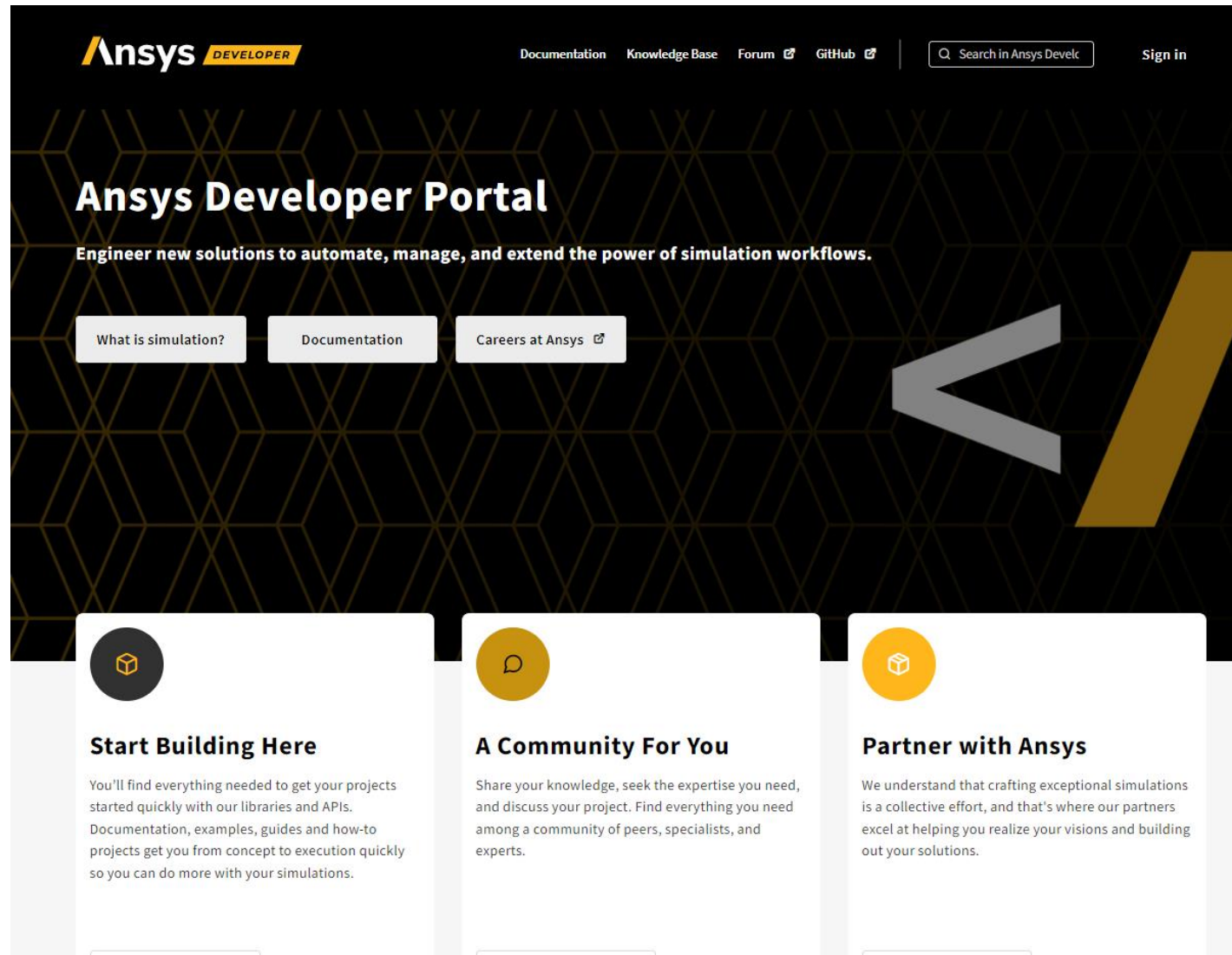
[What's New in Mechanical](#)

[Mechanical Add-ons Guide \(PDF 📄\)](#)

[BETA: Mechanical Beta Features \(PDF 📄\)](#)

- [Scripting Quick Start](#)
- [Mechanical APIs](#)
- [Scripting Examples](#)

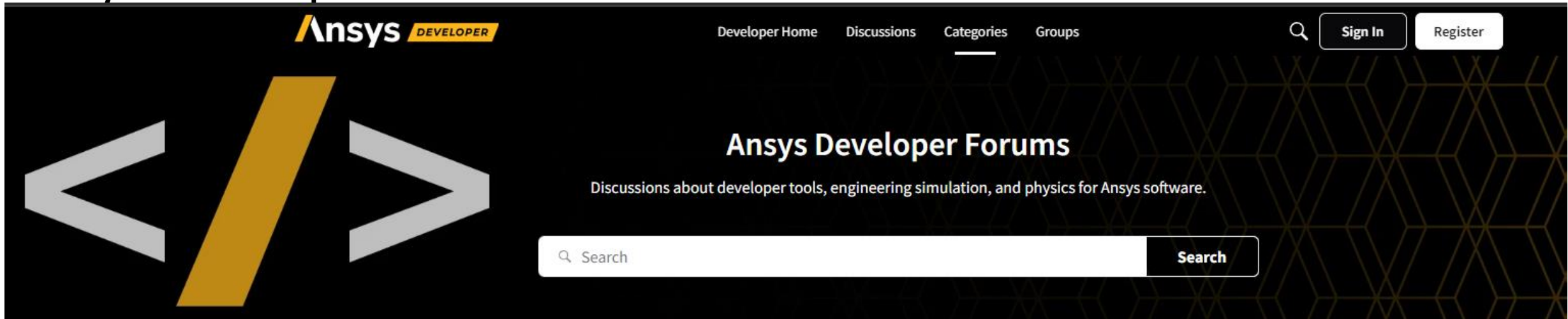
# Ansys Developer Portal







<https://developer.ansys.com>



# Ansys Developer Forum



-  Answered ✓ PyDPF 17 views 1 comments 0 reactions Started by Pavel Most recent by Ayush Kumar Jan 22, 2023 Engineering Simulation
- 
- Retrieve messages from Mechanical**  
Through ACT scripting, how can I retrieve the info, warning and error messages in Mechanical?
-  Answered ✓ AnsysACT 16 views 3 comments 0 reactions Started by Pennelle Marone-Hiltz Most recent by Pennelle Marone-Hiltz Jan 20, 2023 Engineering Simulation
- 
- How to get results with DPF for a time/frequency not available in my rst file?**  
For instance, let's imagine that our harmonic analysis contains results for frequency 12Hz and 15Hz. If I want to get a result for 14Hz and 14.5Hz, how can I do it?
-  DPF 17 views 1 comments 0 reactions Started by Javier Vique Most recent by Javier Vique Jan 17, 2023 Engineering Simulation
- 
- How can I read a file using Python?**  
How can I read a file using Python?
-  Answered ✓ IronPython 53 views 6 comments 2 reactions Started by Pennelle Marone-Hiltz Most recent by Pennelle Marone-Hiltz Jan 12, 2023 Engineering Simulation

<https://discuss.ansys.com/>

# Automation cheat slides

### Selection Manager

ExtAPI.SelectionManager

- Clear selection:
 

```
ExtAPI.SelectionManager.ClearSelection()
```
- Get IDs of entities currently selected in the UI:
 

```
ExtAPI.SelectionManager.GetCurrentSelections()
```
- Create a new selection and use it:
 

```
# create a new empty selection
CompObj = ExtAPI.SelectionManager.CreateSelectionInDB([SelectionTypeFrom, NameKey, Entity])
# provide list of IDs of entities to select
CompObj.IDs = [1]
# create new selection method
selection = ExtAPI.DataModel.Project.Model.AddCustomSelection()
# assign selection
selection.AssignToCompObj(CompObj)
```

### Selecting objects by name

ExtAPI.DataModel.GetObjectByName()

Using GetObjectByName:

This method will return a list of all the entities with the named passed in argument:

```
ExtAPI.DataModel.GetObjectByName("Test") > selection = list of all entities named "Test"
```

Selecting a named selection by its name and scope this selection to a focus:

```
analysis = ExtAPI.DataModel.Project.Model.Analysis[0] > selection analysis
selection.AssignToCompObj() > add CompObj
myObj = ExtAPI.DataModel.GetObjectByName("TestSelection") [0] > selection list of IDs in the tree with
name "TestSelection"
selection.AssignToCompObj() > scope named selection
```

### Geometry

ExtAPI.DataModel.Project.Model.Geometry

- Loop through children inside geometry branches:
 

```
for part in ExtAPI.DataModel.Project.Model.Children:
    for body in part.Children:
        print(body.Name)
```
- If the model uses model assembly, this structure is lost as parts and bodies are grouped in folders. It is then better to select bodies through their type (as below):
 

```
# Get list of all bodies:
geometry = ExtAPI.DataModel.Project.Model.Geometry
listOfBodies = geometry.GetChildren([DataModel.Category.Body, Type])
```

### Geo Data

ExtAPI.DataModel.GeoData

- Loop through assemblies, parts and bodies of GeoData:
 

```
for Assembly in ExtAPI.DataModel.Project.Model.Assemblies:
    for Part in Assembly.Parts:
        for Body in Part.Bodies:
            for Face in Body.Faces:
                print(Face.Area)
```

### Materials

materials module

- Material properties defined in Engineering Data can be accessed thanks to the material module:
 

```
# Import materials module
import materials
# get the list of materials in the Mechanical tree
mat = ExtAPI.DataModel.Project.Model.Materials.Children[0]
print(mat.Name)
# get engineering data material properties for this material
matDef = mat.GetEngineeringDataMaterial()
# get and print list of material properties
listOfProp = materials.GetMaterialProperties(matDef)
print(listOfProp)
# get and print elasticity properties (if it exists)
elasticity = materials.GetMaterialPropertiesByType(matDef, "Elasticity")
print(elasticity)
```

### Coordinate systems

ExtAPI.DataModel.Project.Model.CoordinateSystems

- Create a coordinate system:
 

```
# create coordinate system
CS = ExtAPI.DataModel.Project.Model.CoordinateSystems.AddCoordinateSystem()
# modify CS name
CS.Name = "New CS"
# change CS type
CS.CoordinateSystemType = CoordinateSystemTypeFrom.Cylindrical
```

### Connections

ExtAPI.DataModel.Project.Model.Connections

- Define a contact region between two faces:
 

```
# create contact
Connections = ExtAPI.DataModel.Project.Model.Connections
newContact = Connections.AddContactRegion()
# modify contact type and define interface method
newContact.ContactType = ContactType.PairContact
newContact.PairContactInterfaceID = 1
ExtAPI.DataModel.Project.Model() > add new contact to new method lines
# select contact and target faces
selected = ExtAPI.SelectionManager.CreateSelectionInDB([SelectionTypeFrom, NameKey, Entity]) > create empty selection
selected.IDs = [127] > define IDs of faces to be included
newContact.ReversedFaces = selected > apply selection to contact side
target = ExtAPI.SelectionManager.CreateSelectionInDB([SelectionTypeFrom, NameKey, Entity])
target.IDs = [128] > define IDs of faces to be included
newContact.TargetFaces = target > apply selection to target side
```

### Mesh

ExtAPI.DataModel.Project.Model.Mesh

- Define a mesh method on a body:
 

```
# create mesh
meshObj = ExtAPI.DataModel.Project.Model.Mesh
# create body selection
CompObj = ExtAPI.SelectionManager.CreateSelectionInDB([SelectionTypeFrom, NameKey, Entity])
CompObj.IDs = [1]
# import and define new meshing method
meshObj = meshObj.AddCustomMethod()
meshObj.AssignToCompObj(CompObj)
meshObj.MethodType = MeshMethodType
```

### Mesh Data

ExtAPI.DataModel.Project.Model.Analysis[0].MeshData

- Access information on an element:
 

```
# create mesh data
meshData = ExtAPI.DataModel.Project.Model.Analysis[0].MeshData
# select any element and display information
elementID = meshData.Elements[0]
elementDef = elementID.ElementDef
element.Type = Type of element
```

Yes, we'll share this slide deck



# Automation challenge

# Automation challenge: Tensile pull test on cylinder

Challenge: The model must be entirely set up, run and post processed without using Mechanical's GUI. Only the scripting console may be used

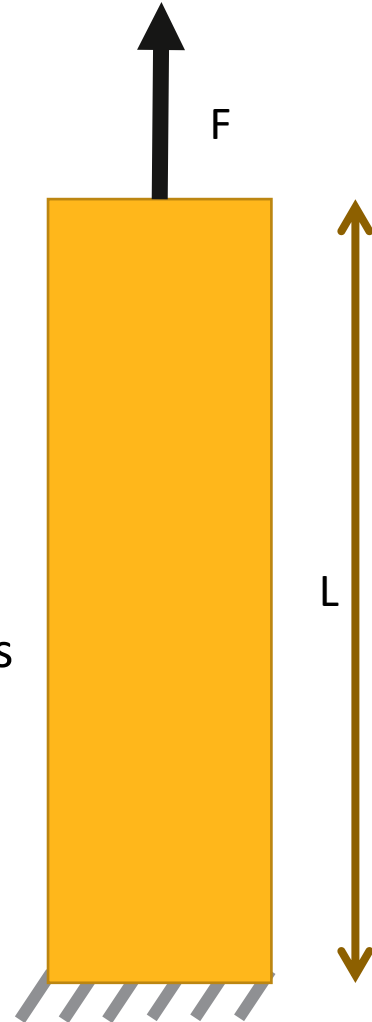
To get started import a cylinder shape into Mechanical

Using the console:

- Apply a pull force of 1N on one of the cylinder face
- Apply a fixed support to the opposite face
- Solve the model
- Extract the maximum displacement in the pull test direction
- From this result calculate the young's modulus and compare it with the material's young's modulus

Reminder

$$\text{Young's Modulus} = \frac{\sigma}{\epsilon} ; \sigma = \frac{F}{A} ; \epsilon = \frac{\Delta L}{L}$$



The Ansys logo, featuring a stylized yellow and black 'A' followed by the word 'nsys' in black.

