PyFluent-Visualization cheat sheet

Version: 0.23.0

Launch a fluent solver session using PyFluent

```
import ansys.fluent.core as pyfluent
solver_session = pyfluent.Solver.from_install()
```

Set up PyFluent session

```
from ansys.fluent.core import examples, SolverEvent
from ansys.fluent.core.solver import (
    ReadCaseData,
    PressureOutlets,
    VelocityInlets,
    WallBoundaries,
    WallBoundary,
    Initialize,
    Iterate,
)
file_path = examples.download_file(
    file_name="exhaust_system.cas.h5",
    directory="pyfluent/exhaust_system",
    )
examples.download_file(
    file_name="exhaust_system.dat.h5",
    directory="pyfluent/exhaust_system",
    )
ReadCaseData(
        settings_source=solver_session
    )(file_name=file_path)
```

Set-up PyFluent-Visualization for post-processing

```
from ansys.units import VariableCatalog
from ansys.fluent.visualization import (
    Contour,
    GraphicsWindow,
    IsoSurface,
    Mesh,
    Monitor,
    Pathline,
    PlaneSurface,
    Vector,
    XYPlot,
    config,
)
config.interactive = False
config.view = "isometric"
```

Display mesh at wall

```
window = GraphicsWindow()
mesh = Mesh(
    solver=solver_session,
    show_edges=True,
    surfaces=WallBoundaries(
        settings_source=solver_session
        )
    )
window.add_graphics(mesh)
window.show()
```

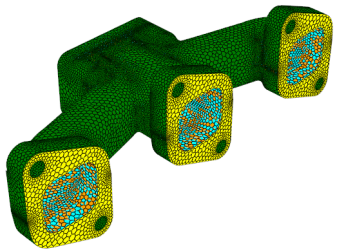

Figure 1: Mesh

Create XY, YZ and ZX plane-surface objects and display

```
window = GraphicsWindow()
xy_plane = PlaneSurface.create_xy_plane(
    solver=solver_session, z=-0.0441921
    )
window.add_graphics(xy_plane, position=(0, 0))

yz_plane = PlaneSurface.create_yz_plane(
    solver=solver_session, x=-0.174628
    )
window.add_graphics(yz_plane, position=(0, 1))

zx_plane = PlaneSurface.create_zx_plane(
    solver=solver_session, y=-0.0627297
    )
window.add_graphics(zx_plane, position=(0, 2))
window.show()
```

Create plane-surface objects from point and normal and display

```
window = GraphicsWindow()
xy_plane =
    PlaneSurface.create_from_point_and_normal(
    solver=solver_session,
    point=[0.0, 0.0, -0.0441921],
    normal=[0.0, 0.0, 1.0],
    )
window.add_graphics(xy_plane)
window.show()
```

Create and display an iso-surface

```
window = GraphicsWindow()
mid_plane_x = IsoSurface(
    solver=solver_session,
    field="x-coordinate",
    iso_value=-0.174,
    )
window.add_graphics(mid_plane_x)
window.show()
```

Display pressure contour at wall

```
window = GraphicsWindow()
pressure_contour = Contour(
    solver=solver_session,
    field=VariableCatalog.ABSOLUTE_PRESSURE,
    surfaces=WallBoundaries(
        settings_source=solver_session
        )
    )
window.add_graphics(pressure_contour)
window.show()
```

Display vector at a wall boundary

```
window = GraphicsWindow()
velocity_vector = Vector(
    solver=solver_session,
    field=VariableCatalog.VELOCITY,
    color_by=VariableCatalog.PRESSURE,
    surfaces=[WallBoundary(
        settings_source=solver_session,
        name="solid_up:1:830")],
    scale=20,
    )
window.add_graphics(velocity_vector)
window.show()
```

Display pathlines

```
window = GraphicsWindow()
pathlines = Pathline(
    solver=solver_session,
    field=VariableCatalog.VELOCITY_MAGNITUDE,
    surfaces=VelocityInlets(
        settings_source=solver_session
        )
    )
window.add_graphics(pathlines)
window.show()
```

Varying opacity

```
window = GraphicsWindow()
window.add_graphics(mesh, opacity=0.05)
window.add_graphics(velocity_vector)
window.show()
```

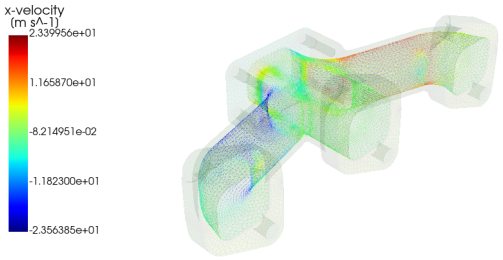

Figure 2: Varying opacity

Create and display XY plot

```
window = GraphicsWindow()
xy_plot_object = XYPlot(
    solver=solver_session,
    surfaces=PressureOutlets(
        settings_source=solver_session
        ),
    y_axis_function=VariableCatalog.TEMPERATURE
    )
window.add_plot(xy_plot_object,
    title="Temperature")
window.show()
```

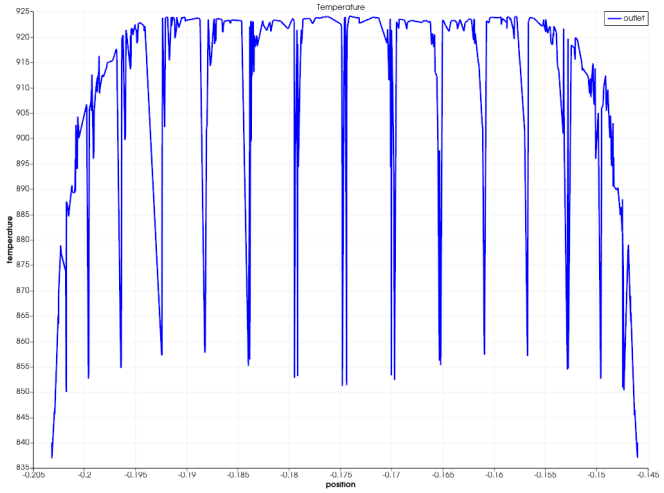

Figure 3: XY plot

Create and display monitors plot

```
window = GraphicsWindow()
residual = Monitor(solver=solver_session,
    monitor_set_name="residual")
window.add_plot(residual)
window.show()
```

Change 'renderer' and save plots

```
window = GraphicsWindow()
residual = Monitor(solver=solver_session,
    monitor_set_name="residual")
window.add_plot(residual)
window.save_graphics("sample_plot.pdf")

window.renderer = "matplotlib"
window.save_graphics("sample_plot.png")
```

Update graphics in real-time

```
plot_window = GraphicsWindow()
residual = Monitor(solver=solver_session,
    monitor_set_name="residual")
plot_window.add_plot(residual)
plot_window.show()

graphics_window = GraphicsWindow()
pressure_contour = Contour(
    solver=solver_session,
    field=VariableCatalog.ABSOLUTE_PRESSURE,
    surfaces=WallBoundaries(
        settings_source=solver_session
        )
    )
graphics_window.add_graphics(pressure_contour)
graphics_window.show()

plot_window.real_time_update(
    events=[SolverEvent.SOLUTION_INITIALIZED,
        SolverEvent.ITERATION_ENDED]
)
graphics_window.real_time_update(
    events=[SolverEvent.SOLUTION_INITIALIZED,
        SolverEvent.ITERATION_ENDED]
)

Initialize(settings_source=solver_session)()
Iterate(settings_source=solver_session)(iter_count=50)
```