set_pipeline_parameter_from_expression /parameter_name=t_mass /expression=metric::processed::height * 4.6 /as_integer=true **;** the pipeline parameter contains the value of the expression.

getting the syntax correct can be challenging because there can be a conflict between the & as a concatenation parameter in a pipeline parameter and the & as a boolean “and” operator.

for many pipeline commands visual3d recognizes the conflict and handles the situations in a context specific way.

visual3d cannot make that distinction and “bad” things happen.

consider the following which uses string comparisons and a pipeline parameter in the expression:

set_pipeline_parameter_from_expression /parameter_name= mval /expression=(&::tag_names& == utility) * 4.6 /as_integer=false **;** the goal was to set mval to 4.6 when the pipeline parameter tag_names was equal to the string “utility”;

one solution is to separate the pipeline parameter from the expression.

set_pipeline_parameter /parameter_name=temp /parameter_value=(&::tag_names& == utility)*4.6 **;**

set_pipeline_parameter_from_expression /parameter_name=t_mass /expression=::temp /as_integer=false **;**
another solution, which is even trickier in terms of syntax is:

set_pipeline_parameter_from_expression /parameter_name=scott /expression=(“&::tag_names&”==“ utility”) ! /as_integer=true **;** this forces visual3d to recognize the strings as strings and the & is treated correctly.

get global metrics

global metrics can be set to a pipeline parameter by setting “global” in front of the metric name. more information about expressions can be found here.

set_pipeline_parameter_from_expression /parameter_name=katie /expression=global::metric::processed::mass /as_integer=false **;**