

Graph (Graph Server Feature)

The MEP functions on this page are to work with graph (creation, modification, graph algorithms, ...). The used graph libraries are those from <http://graphstream-project.org/>:

- [gs-core](#)
- [gs-algo](#)

See also the related [Graph Data Types](#) in Odysseus. To use the MEP functions, the Graph Server Feature is needed.

For developers: Feel free to use the above mentioned gs libraries and data types to create new MEP functions. The bundle for the MEP functions that you might enhance is `de.uniol.inf.is.odysseus.graph.mep`. And don't forget to keep this page up-to-date.

AddAttributeToGraphElement(Graph/GraphNode/GraphEdge element, String attributeKey, String /Object attributeValue)

Add an attribute to a graph element. The first input is one of the three [Graph Data Types](#), the second key of the attribute, and the third the value of the attribute. The output is the updated graph element.

Example

```
output1 = MAP({EXPRESSIONS = [['AddAttributeToGraphElement(graph, "label", "super graph")', 'graph']]}, input)
output2 = MAP({EXPRESSIONS = [['AddAttributeToGraphElement(node, "label", "super node")', 'node']]}, input)
output3 = MAP({EXPRESSIONS = [['AddAttributeToGraphElement(edge, "label", "super edge")', 'edge']]}, input)
```

AddEdgeToGraph(GraphEdge edge, Graph graph)

Adds an edge to a graph. If there already exists an edge with the same id, its former attributes will be replaced by the attributes of the new edge. The first input is an [Edge Data Type](#), the second a [Graph Data Type](#). The output is the updated graph.

Example

```
output = MAP({EXPRESSIONS = [['AddEdgeToGraph(edge, graph)', 'graph']]}, input)
```

AddNodeToGraph(GraphNode node, Graph graph)

Adds a node to a graph. If there already exists a node with the same id, its former attributes will be replaced by the attributes of the new node. The first input is a [Node Data Type](#), the second a [Graph Data Type](#). The output is the updated graph.

Example

```
output = MAP({EXPRESSIONS = [['AddNodeToGraph(node, graph)', 'graph']]}, input)
```

FindNodesByPattern(Graph graph, String pattern)

Find all nodes in a graph, which id fulfills a given pattern. The first input is a [Graph Data Type](#), the second a pattern for the `nodeId.match(pattern)` call. The output is a [List_GraphNode Data Type](#).

Example

```
output = MAP({EXPRESSIONS = [['FindNodesByPattern(graph, "nodesStartingWithThis.*")', 'nodes_list']]}, input)
```

GetAllEdges(Graph graph)

Gets a list of all edges of a graph. The input is a [Graph Data Types](#). The output is a [List_GraphEdge Data Type](#).

Example

```
output = MAP({EXPRESSIONS = [['GetAllEdges(graph)', 'edges']]}, input)
```

GetAllNodes(Graph graph)

Gets a list of all nodes of a graph. The input is a [Graph Data Types](#). The output is a [List_GraphNode Data Type](#).

Example

```
output = MAP({EXPRESSIONS = [['GetAllNodes(graph)', 'nodes']]}, input)
```

GetGraphElementId(Graph/GraphNode/GraphEdge element)

Gets the id of a graph element (a graph, node or edge). The input is one of the three [Graph Data Types](#). The output is the id (String).

Example

```
output = MAP({EXPRESSIONS = [['GetGraphElementId(mygraph)', 'graphId'], ['GetGraphElementId(mynode)', 'nodeId'], ['GetGraphElementId(myedge)', 'edgeId']]}, input)
```