

NAME

gv_ocaml - graph manipulation in ocaml

SYNOPSIS**USAGE****INTRODUCTION**

gv_ocaml is a dynamically loaded extension for **ocaml** that provides access to the graph facilities of **graphviz**.

COMMANDS**New graphs**

New empty graph

```
graph_handle gv.graph (name);
graph_handle gv.digraph (name);
graph_handle gv.strictgraph (name);
graph_handle gv.strictdigraph (name);
```

New graph from a dot-syntax string or file

```
graph_handle gv.readstring (string);
graph_handle gv.read (string filename);
graph_handle gv.read (channel);
```

Add new subgraph to existing graph

```
graph_handle gv.graph (graph_handle, name);
```

New nodes

Add new node to existing graph

```
node_handle gv.node (graph_handle, name);
```

New edges

Add new edge between existing nodes

```
edge_handle gv.edge (tail_node_handle, head_node_handle);
```

Add a new edge between an existing tail node, and a named head node which will be induced in the graph if it doesn't already exist

```
edge_handle gv.edge (tail_node_handle, head_name);
```

Add a new edge between an existing head node, and a named tail node which will be induced in the graph if it doesn't already exist

```
edge_handle gv.edge (tail_name, head_node_handle);
```

Add a new edge between named tail and head nodes which will be induced in the graph if they don't already exist

```
edge_handle gv.edge (graph_handle, tail_name, head_name);
```

Setting attribute values

Set value of named attribute of graph/node/edge - creating attribute if necessary

```
string gv.setv (graph_handle, attr_name, attr_value);
string gv.setv (node_handle, attr_name, attr_value);
string gv.setv (edge_handle, attr_name, attr_value);
```

Set value of existing attribute of graph/node/edge (using attribute handle)

```
string gv.setv (graph_handle, attr_handle, attr_value);
string gv.setv (node_handle, attr_handle, attr_value);
string gv.setv (edge_handle, attr_handle, attr_value);
```

Getting attribute values

Get value of named attribute of graph/node/edge

```
string gv.getv (graph_handle, attr_name);  
string gv.getv (node_handle, attr_name);  
string gv.getv (edge_handle, attr_name);
```

Get value of attribute of graph/node/edge (using attribute handle)

```
string gv.getv (graph_handle, attr_handle);  
string gv.getv (node_handle, attr_handle);  
string gv.getv (edge_handle, attr_handle);
```

Obtain names from handles

```
string gv.nameof (graph_handle);  
string gv.nameof (node_handle);  
string gv.nameof (attr_handle);
```

Find handles from names

```
graph_handle gv.findsubg (graph_handle, name);  
node_handle gv.findnode (graph_handle, name);  
edge_handle gv.findedge (tail_node_handle, head_node_handle);  
attribute_handle gv.findattr (graph_handle, name);  
attribute_handle gv.findattr (node_handle, name);  
attribute_handle gv.findattr (edge_handle, name);
```

Misc graph navigators returning handles

```
node_handle gv.headof (edge_handle);  
node_handle gv.tailof (edge_handle);  
graph_handle gv.graphof (graph_handle);  
graph_handle gv.graphof (edge_handle);  
graph_handle gv.graphof (node_handle);  
graph_handle gv.rootof (graph_handle);
```