

**NAME**

gv\_sharp - graph manipulation in sharp

**SYNOPSIS****USAGE****INTRODUCTION**

**gv\_sharp** is a dynamically loaded extension for **sharp** that provides access to the graph facilities of **graphviz**.

**COMMANDS****New graphs**

New empty graph

```
SWIGTYPE_p_Agraph_t gv.graph (string name);
SWIGTYPE_p_Agraph_t gv.digraph (string name);
SWIGTYPE_p_Agraph_t gv.strictgraph (string name);
SWIGTYPE_p_Agraph_t gv.strictdigraph (string name);
```

New graph from a dot-syntax string or file

```
SWIGTYPE_p_Agraph_t gv.readstring (string string);
SWIGTYPE_p_Agraph_t gv.read (string filename);
SWIGTYPE_p_Agraph_t gv.read (SWIGTYPE_p_FILE f);
```

Add new subgraph to existing graph

```
SWIGTYPE_p_Agraph_t gv.graph (SWIGTYPE_p_Agraph_t g, string name);
```

**New nodes**

Add new node to existing graph

```
SWIGTYPE_p_Anode_t gv.node (SWIGTYPE_p_Agraph_t g, string name);
```

**New edges**

Add new edge between existing nodes

```
SWIGTYPE_p_Aedge_t gv.edge (SWIGTYPE_p_Anode_t t, SWIGTYPE_p_Anode_t h);
```

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

```
SWIGTYPE_p_Aedge_t gv.edge (SWIGTYPE_p_Anode_t t, string hname);
```

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

```
SWIGTYPE_p_Aedge_t gv.edge (string tname, SWIGTYPE_p_Anode_t h);
```

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

```
SWIGTYPE_p_Aedge_t gv.edge (SWIGTYPE_p_Agraph_t g, string tname, string hname);
```

**Setting attribute values**

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

```
string gv.setv (SWIGTYPE_p_Agraph_t g, string attr, string val);
string gv.setv (SWIGTYPE_p_Anode_t n, string attr, string val);
string gv.setv (SWIGTYPE_p_Aedge_t e, string attr, string val);
```

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

```
string gv.setv (SWIGTYPE_p_Agraph_t g, SWIGTYPE_p_Agsym_t a, string val);
string gv.setv (SWIGTYPE_p_Anode_t n, SWIGTYPE_p_Agsym_t a, string val);
string gv.setv (SWIGTYPE_p_Aedge_t e, SWIGTYPE_p_Agsym_t a, string val);
```

**Getting attribute values**

Get value of named attribute of graph/node/edge

```
string gv.getv (SWIGTYPE_p_Agraph_t g, string attr);
string gv.getv (SWIGTYPE_p_Agnode_t n, string attr);
string gv.getv (SWIGTYPE_p_Agedge_t e, string attr);
```

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

```
string gv.getv (SWIGTYPE_p_Agraph_t g, SWIGTYPE_p_Agsym_t a);
string gv.getv (SWIGTYPE_p_Agnode_t n, SWIGTYPE_p_Agsym_t a);
string gv.getv (SWIGTYPE_p_Agedge_t e, SWIGTYPE_p_Agsym_t a);
```

**Obtain names from handles**

```
string gv.nameof (SWIGTYPE_p_Agraph_t g);
string gv.nameof (SWIGTYPE_p_Agnode_t n);
string gv.nameof (SWIGTYPE_p_Agsym_t a);
```

**Find handles from names**

```
SWIGTYPE_p_Agraph_t gv.findsubg (SWIGTYPE_p_Agraph_t g, string name);
SWIGTYPE_p_Agnode_t gv.findnode (SWIGTYPE_p_Agraph_t g, string name);
SWIGTYPE_p_Agedge_t gv.findedge (SWIGTYPE_p_Agnode_t t, SWIGTYPE_p_Agnode_t h);
SWIGTYPE_p_Agsym_t gv.findattr (SWIGTYPE_p_Agraph_t g, string name);
SWIGTYPE_p_Agsym_t gv.findattr (SWIGTYP
```