Cheatography

GraphSpace Python Client Cheat Sheet

by annaritz via cheatography.com/67574/cs/16994/

Import Statements

```
from graphspace_python.api.client import GraphSpace
from graphspace_python.graphs.classes.gsgraph import GSGraph
from graphspace_python.graphs.classes.gsgroup import GSGroup
from graphspace_python.graphs.classes.gslayout import GSLayout
```

Create a Graph

`G = GSGraph()`	Create empty graph
`G.add_node('a')`	Add node to graph
`..., label='A', ...)`	Label to display
`..., popup='html text', ...)`	Popup text
`G.add_edge('a', 'b')`	Add undirected edge to graph
`..., directed=True, ...)`	Directed edge
`..., popup='html text', ...)`	Popup text
`G.set_name('My Sample Graph')`	Add name/title to graph
`G.set_tags(['sample'])`	Add tags to graph

Add Node Style

`G.add_node_style('a', shape='ellipse')`	Node shape
`..., color='blue', ...)`	Node color
`..., color='#A152B8', ...)`	HTML color
`..., bubble='blue', ...)`	Bubble style
`..., height=40, width=20, ...)`	Custom size
`..., style='dashed', ...)`	Border style
`..., border_color='red', ...)`	Border color
`..., border_width=2, ...)`	Border width

Arguments should be combined into one `add_node_style` command.

Add Edge Style

`G.add_edge_style('a', 'b', edge_style='dotted')`	Edge style
`..., directed=True, ...)`	Directed edge
`..., color='#FF5733', ...)`	HTML color
`..., arrow_shape='tee', ...)`	Arrow shape

Add Edge Style (cont)

`..., width=4, ...)`	Edge width
`..., color='magenta', ...)`	Edge color
`..., arrow_fill='hollow', ...)`	Arrow fill
`..., width=4, ...)`	Edge width

Arguments should be combined into one `add_edge_style` command.

Post Graph to GraphSpace

`gs = GraphSpace('uname', 'psswd')`	Connect to GraphSpace
`graph = gs.post_graph(G)`	Post (new) graph to GraphSpace
`graph = gs.update_graph(G)`	Update (existing) graph to GraphSpace
`gs.delete_graph(graph=graph)`	Delete graph from GraphSpace

Update Graph if it Exists, Otherwise Post it

```
def post(G, gs):
  try:
    graph = gs.update_graph(G)
  except:
    graph = gs.post_graph(G)
  return graph
```

Fetch Posted Graph

`graph = gs.get_graph(graph_name='My Sample Graph')`	Fetch by graph name
`graph = gs.get_graph(graph_id=12345)`	Fetch by graph id
`graph.name`	Name of graph
`graph.id`	ID of graph

GraphSpace Python Client Cheat Sheet

by annaritz via cheatography.com/67574/cs/16994/

Make Graph Public

`gs.publish_graph(graph=graph)`	Make graph public
`gs.unpublish_graph(graph=graph)`	Make graph private

Groups

`mygroup = GSGroup(name='My group', description='sample group')`	Create group
`group = gs.post_group(mygroup)`	Add group
`group = gs.get_group(group_name='My group')`	Fetch group
`gs.share_graph(graph=graph, group_name='My group')`	Share graph with group
`gs.unshare_graph(graph=graph, group_name='My group')`	Remove graph from group

Layouts

`L = GSLayout()`	New empty layout
`L.set_node_position('a', y=38.5, x=67.3)`	Set x and y coords
`L.add_node_style(),L.add_edge_style()`	Add styles in layout
`L.set_name('My Sample Layout')`	Layout name
`L.set_is_shared(1)`	Visible to members who can see graph
`layout = graphspace.post_graph_layout(L, graph=graph)`	Post (new) layout
`layout = graphspace.get_graph_layout(layout_name='My Sample Layout', graph=graph)`	Fetch layout
`layout1 = graphspace.update_graph_layout(layout)`	Update (existing) layout

Graphs usually have manually-determined layouts from GraphSpace, which are saved through the user interface.