GLIMPSE Event API

The GLIMPSE platform offers an asynchronous event API designed to simulate real-world events within visualizations. This API facilitates dynamic updates to the visualization, including modifications to nodes and edges, such as changes in size, color, shape, among other stylistic attributes. Additionally, the event API provides limited support for animations, enhancing the interactive experience. The event API operates through a WebSocket server, enabling real-time communication and updates to the visualization. This server-based approach ensures efficient handling of events and seamless integration into existing visualizations. For access to the complete set of options available through the event API, or for assistance with specific functionalities, we encourage you to contact the GLIMPSE team.

Connecting to GLIMPSE local WebSocket Server

Make sure to take a "glimpse" at the python-socketio api docs

```
import socketio

def main():
        # Basic code to connect to the local websocket server
        sio = socketio.Client()
        sio.connect("http://127.0.0.1:5051")

        # Send an python dict object to the glimpse local websocket
        for update_obj in updates_list:
          sio.emit("glimpse", update_obj) # emit update data to the "glimpse" event
          time.sleep(1) # wait a second - can be less than a second if needed

   sio.disconnect()
```

Current Update-Object example (more update variables to come)

```
# node example
update_obj = {
        "elementType": "node",
        "id": "load_32",
        "updates": { # any changes must be put in the "updates" value
                "color": {
                        "border": "#4F2FA6",
                        "background": "#F1EE86"
                },
                "opacity": 0.7,
                "hidden": true,
                "shape": "box",
                "size": 28
        }
}

# edge example
update_obj = {
        "elementType": "edge",
        "id": "OH_line_23-25",
```

```
        "updates": { # any changes must be put in the "updates" value
                "color": {
                   "color": "#D0F665", # can be rgb, rgba, hex, or named color like "blue"
                   "inherit": "to",
                   "opacity": 0.1
                },
                "animation": True, # animate the existing edge
                "dashes": true,
                "hidden": false,
                "width": 7.55
        }
}
```

For more node and edge style changes and what they do check out

- vis.js-node-options-doc
- vis.js-edge-options-doc

The addNode, addEdge, deleteNode, and deleteEdge Events

These events allow GLIMPSE api users to create and delete nodes and edges to a blank or existing visualization in GLIMPSE tool.

Example of nodes to be created

```
newNodes = [
   {
      "objectType": "Switch", # Like a group or category the node belongs under
      "elementType": "node",
      "attributes": { # The attributes object must have an ID
         "id": "switch1",
         "ipaddress": "192.168.0.2"
      },
      "styles": {
         "label": "192.168.0.2",
         "color": "orange", # Can be hex, rgb, rgba, or a named color
         "shape": "triangle",
         "size": 12
      }
   },
   {
      "objectType": "Switch",
      "elementType": "node",
      "attributes": {
         "id": "switch2",
         "ipaddress": "192.168.0.1"
      },
      "styles": {
         "label": "192.168.0.1",
         "color": "orange",
         "shape": "triangle",
         "size": 12
      }
   }
]
```

Example of Edges to be created

```
newEdges = [
   {
      "objectType": "connection",
      "elementType": "edge",
      "attributes": {
         "id": "switch1-switch2", # must have id
         "strength": "strong",
         "speed": "0.71 mbps",
         "from": "switch1", # must have a from node ID
         "to": "switch2" # must have a to node ID
      },
      "styles": { # Must have styles for the edge
         "color": "grey",
         "width": 4,
         "label": "0.71 mbps"
      }
   },
   {
      "objectType": "connection",
      "elementType": "edge",
      "attributes": {
         "id": "switch2-switch3",
         "strength": "weak",
         "speed": "0.55 mbps",
         "from": "switch2",
         "to": "switch3"
      },
      "styles": {
         "color": "grey",
         "width": 3,
         "label": "0.55 mbps"
      }
   }
]
```

Deleting nodes and edges

```
# only need the IDs
nodes_to_delete = [
   "phone2",
   "computer3"
]

edges_to_delete = [
   "switch2-internet1",
   "phone1-switch3"
]
```

Example code for each event

```
def main():
   sio = socketio.Client()
   sio.connect(f"{URL}:{PORT}")
```

```
    # create new nodes in GLIMPSE with the addNode socket event
    for new_node_obj in newNodes:
        sio.emit("addNode", new_node_obj)
        time.sleep(1.5)

    # create new edges between the added nodes in GLIMPSE
    for new_edge_obj in newEdges:
        sio.emit("addEdge", new_edge_obj)
        time.sleep(0.75)

    # delete some nodes via the deleteNode socket event
    for nodeID in nodes_to_delete:
        sio.emit("deleteNode", nodeID)
        time.sleep(3)

    # delete some edge via the deleteEdge socket event
    for edgeID in edges_to_delete:
        sio.emit("deleteEdge", edgeID)
        time.sleep(2)

    for style in styleChanges:
        sio.emit("glimpse", style)
        time.sleep(1.5)

    # disconect from GLIMPSE WebSocket API
    sio.disconnect()


if __name__ == "__main__":
    main()
```