Graphity Documentation

Instructions

Graph Creation

Node Tool - Left click on any empty space to create a disconnected node
Edge Tool - Left click on any two pre-existing nodes to create a directed edge
Eraser Tool - Left click on any node to delete that node and any corresponding edges

Code Generation

Press the **Generate** button to generate the JSON text corresponding to the graph created using the interface

Graph Visualization

User input in the text editor is transformed into a TypeScript function that is called, which can be used to visualize the algorithm performed on the graph in real-time. The user does not need to include a class header, only the body of the following function:

```
function traverse(graph: Graph): void {
    /* User input goes here */
}
```

At each step the user would like to visualize, **visit()** should be called on the node they would like highlighted. Additional documentation can be found below.

Documentation

Graph class

```
class Graph {
    /* Creates a new Graph object */
    constructor();

    /* Returns the array of all Node objects in the graph */
    getNodes(): GraphNode[];

    /* Returns the array of all Edge object in the graph */
    getEdges(): DirectedEdge[];

    /* Adds a new GraphNode object into the Graph */
    addNode(node: GraphNode): void;

    /* Adds a new Edge object between two pre-existing nodes
*/
    addEdge(from: GraphNode, to: GraphNode): void;
}
```

GraphNode class

```
class GraphNode {
    /* Creates a new GraphNode objects with the given id and
val */
    constructor(id: string, val: string);

    /** Returns an array of GraphNodes the current node
      * has a directed edge to */
    getEdges(): GraphNode[];

    /* Adds an Edge from the given GraphNode to a target
GraphNode */
    addEdge(target: GraphNode): void;

    /* Marks a GraphNode as visited
     * and displays it on the visualizer */
    visit(): void;

    /* Returns whether a GraphNode has been visited already
*/
    isVisited(): boolean;
}
```

DirectedEdge class

```
class DirectedEdge {
    /* Creates a DirectedEdge between the two given
GraphNodes */
    constructor(from: GraphNode, to: GraphNode);
}
```