Adding Vertices

7 min

Now that we have set up our

Preview: Docs Systems for organizing data that dictate how items relate to one another, are accessed, and modified.

data structures

, we can provide an easier way to manage the graph's list of vertices. This gives us an opportunity to abstract out the places that use the Vertex class.

Currently, we would have to manually create a new Vertex instance and add it into the Graph's list of vertices to populate the graph. If we create an .addVertex()

Preview: Docs Loading link description

method

in the Graph class, it simplifies the process of adding a vertex to the graph. This alleviates the burden of knowing how the Vertex class should interact with the Graph class for whoever is using our Graph. They only need to interact with the Graph class.

Instructions

1. Checkpoint 1 Passed

1.

Inside the Graph class, add an .addVertex() method that expects a single parameter, data, which contains the data of vertex to create. Using this argument, create a Vertex instance and add it to the Graph's list of vertices.

We will want to return the newly created Vertex to signal to the method caller that a vertex was successfully created and added to the list.

Hint

Graphs are made of a collection of vertices and edges (that connect the vertices). By creating a new vertex in the .addVertex() method, we can push it directly into the list of vertices, which adds the vertex to the graph.

Don't forget to return the new vertex for later use!

2. Checkpoint 2 Passed

2.

Great! Now we can set up our train network graph and see what it looks like with our train station vertices.

Underneath the Graph class, create a Graph instance and assign it to the trainNetwork variable. Then use the .addVertex() method to add two train stations with the names, 'Atlanta' and 'New York'. Assign the newly created vertices to the variables atlantaStation and newYorkStation, respectively. We will use these variables later on.

Call the .print() method on the trainNetwork. We should see our graph with two vertices inside it. They should be labeled Atlanta and New York, respectively.

Graph.js

```
const Edge = require('./Edge.js');

const Vertex = require('./Vertex.js');


class Graph {
  constructor() {
    this.vertices = [];
  }


  print() {
    this.vertices.forEach(vertex => vertex.print());
  }


  addVertex(data) {
    const vertex = new Vertex(data)
    this.vertices.push(vertex);
    return vertex;
  }
}


const trainNetwork = new Graph();

const atlantaStation = trainNetwork.addVertex('Atlanta');

const newYorkStation = trainNetwork.addVertex('New York');
```

```
trainNetwork.print();


module.exports = Graph;
```

Vertex.js

```
const Edge = require('./Edge.js');

class Vertex {
  constructor(data) {
    this.data = data;
    this.edges = [];
  }

  print() {
    const edgeList = this.edges.map(edge =>
        edge.weight !== null ? `${edge.end.data} (${edge.weight})` : edge.end.data);

    const output = `${this.data} --> ${edgeList.join(', ')}`;
    console.log(output);
  }
}

module.exports = Vertex;
```

Edge.js

```
class Edge {
  constructor(start, end, weight = null) {
    this.start = start;
    this.end = end;
```

```
    this.weight = weight;
  }
}

module.exports = Edge;
```