**Introduction**

4 min

Trees are wonderful

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

data structures

that can model real life hierarchical information, including organizational charts, genealogical trees, computer file systems, HTML elements on a web page (also known as the Document Object Model, or DOM), state diagrams, and more.

A tree is composed of tree nodes. A tree node is a very simple data structure that contains:

- Data
- A list of children, where each child is itself a tree node

We can add data to and remove data from a tree and traverse it in two different ways:

- Depth-first, or
- Breadth-first

In this lesson, we're going to implement the tree node data structure as a class in JavaScript.

**Instructions**

1. Checkpoint 1 Passed

**1.**

In **TreeNode.js**, you will find an empty TreeNode class. We will maintain the children of TreeNode as a JavaScript array. This will make it easier to add and remove a child.

Define a constructor that takes one parameter, data. Inside the constructor:

    - define a data class property and assign it to the parameter, data
    - define a children class property and assign it to an empty array.

2. Checkpoint 2 Passed

**2.**

Open **script.js**, instantiate a TreeNode class with argument of 1 and assign it to a const variable tree.

Display the contents of tree with console.log.

**TreeNode.js**

```
class TreeNode {
  constructor(data) {
    this.data = data;
    this.children = [];
  }
};

module.exports = TreeNode;
```

**script.js**

```
const TreeNode = require('./TreeNode');

const tree = new TreeNode(1);
console.log(tree);
```

**>>Output**

```
TreeNode { data: 1, children: [] }
```