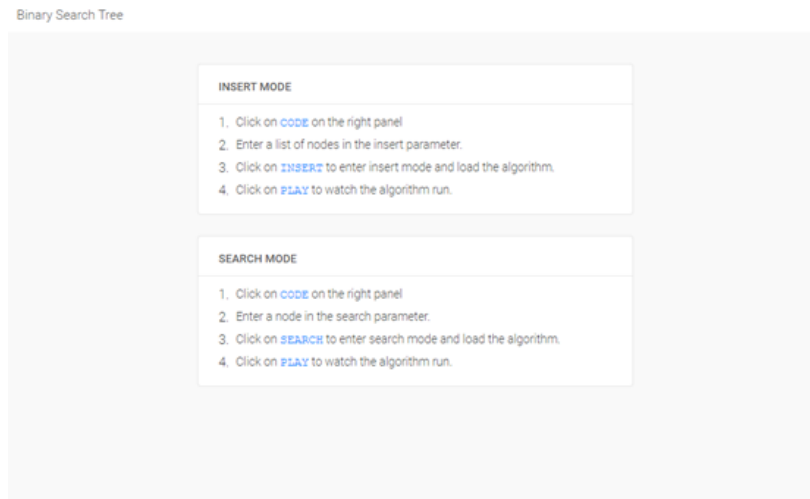# User Guide: Change Content

This guide walks through how to change text for different sections in the UI.
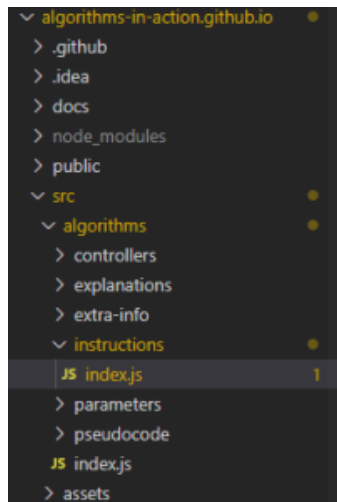
Modify "Instructions"

1. Modify "Instruction"

Binary Search Tree

INSERT MODE

1. Click on CODE on the right panel
2. Enter a list of nodes in the insert parameter.
3. Click on INSERT to enter insert mode and load the algorithm.
4. Click on PLAY to watch the algorithm run.

SEARCH MODE

1. Click on CODE on the right panel
2. Enter a node in the search parameter.
3. Click on SEARCH to enter search mode and load the algorithm.
4. Click on PLAY to watch the algorithm run.

Go to the following file and change the instruction's text.

algorithms-in-action.github.io
- .github
- .idea
- docs
- node_modules
- public
- src
  - algorithms
    - controllers
    - explanations
    - extra-info
    - instructions
      - JS index.js   1
    - parameters
    - pseudocode
    - JS index.js
  - assets

2. Modify "Background" Tab

Binary Search Tree

A binary search tree (bst) is a basic tree data structure that supports a simple searching algorithm. For each node in the binary search tree, with key k, all nodes in its left subtree have keys smaller than k, while all nodes in its right subtree have keys larger than k. Where duplicate keys are allowed in the tree, they usually go into the right subtree by convention.
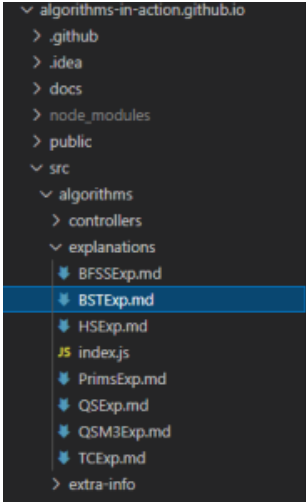
The binary search tree is built up by adding items one at a time. Since the average path length in a tree of n items is $\log n$, the average case complexity of building a bst is $O(n \log n)$. Similarly, the average case for a search for m items in a tree of n items is $O(m \log n)$, that is $O(\log n)$ per item.

The biggest problem with the binary search tree is that its behavior degenerates when there is order in the input data. In the worst case, sorted or reverse sorted data items yield a linear tree, or "stick", the complexity of building the tree is $O(n^2)$, and the complexity of a search for a single item is $O(n)$.

Time Complexity

Algorithm	Average	Worst Case
Space	O(n)	O(n)
Search	O(log n)	O(n)
Insert	O(log n)	O(n)
Delete	O(log n)	O(n)

Go to the following directory and modify the associated markdown file.

```
∨ algorithms-in-action.github.io
  > .github
  > .idea
  > docs
  > node_modules
  > public
  ∨ src
    ∨ algorithms
      > controllers
      ∨ explanations
        BFSSExp.md
        BSTExp.md
        HSExp.md
        JS index.js
        PrimsExp.md
        QSExp.md
        QSM3Exp.md
        TCExp.md
      > extra-info
```

3. Modify "More" Tab

Extra Info

Geeks for Geeks/ Binary Search Tree

Go to the following directory and modify the associated markdown file.

```
∨ algorithms-in-action.github.io
  > .github
  > .idea
  > docs
  > node_modules
  > public
  ∨ src
    ∨ algorithms
      > controllers
      > explanations
      ∨ extra-info
          BFSSInfo.md
          BSTInfo.md
          HSInfo.md
        JS index.js
          PrimsInfo.md
          QSInfo.md
          QSM3Info.md
          TCInfo.md
    ∨ instructions
```