**记录-record**

**2021.5.16**

1. 验证二叉搜索树

2.恢复二叉搜索树

3.相同的树

核心： 使用中序树来进行判断root.left>>root >>root.right

4.

React 视频学习todoMvc demo

React.fragement setState({}) e.preventDefault() e=> this,get(e)

解构赋值 findIndex 返回第一个符合条件否则就是underfined

Let {} = {};

数组对象去重

const arr = [{id: 1, name:’123’},{id: 1, name:’1234}]

let map = new Map(); arr.filter() !map.has(id) && map.set(id, 1);

hashChange

dva基于react和reduce

window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_ && window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_())是为了能在谷歌浏览器中用redux devtools调试工具。

**2021.5.17**

1. 对称二叉树
2. 层序遍历
3. 锯齿形遍历
4. 最大深度

IOS橡皮效果去除

1. window.addEventListener('touchmove', function(event) {
2. event.preventDefault();
3. }, false);
4. window.addEventListener('touchstart', function(event) {
5. event.preventDefault();
6. }, false);

字体抗锯齿 font-smoothing: antialiased;

Dva框架使用，reducer react-saga react-router结合起来的

**2021.5.18**

1. 前序和中序还原二叉树

2. 后序和中序还原二叉树

3．二叉树的遍历2 reverse

4. 数组还原为二叉树

5. 链表还原二叉树

DVA学习

    const app = dva();

    app.model({

      namespace: 'index',

      state: 0,

      reducers: {

        add(state) {

          return state + 1;

        }

      },

      effects: {

        \*adds(action, {call, put}) {

          yield call(delay, 1000);

          yield put({type: 'add'});

        }

      }，

Subscription: {

Keys({dispatch}) {

Dispatch({type:’add’});

}

}

    });

    app.router(() => <Home/>);

    app.start('#root');

Effect 是一个 Generator 函数，内部使用 yield 关键字，标识每一步的操作（不管是异步或同步）。

闭包

为一个页面添加几个用于调整字体大小的按钮。一种方法是以像素为单位指定 body 元素的 font-size，然后通过相对的 em 单位设置页面中其它元素（例如页眉）的字号

      function makeSizes(size) {

        return function() {

          document.body.style.fontSize = size + 'px';

        }

      }

      document.getElementById('ids').onclick = makeSizes(12);

document.getElementById('ids').onclick = makeSizes(142);

document.getElementById('ids').onclick = makeSizes(16);

**用闭包模拟私有方法**

      var counter = (function () {

        var privateCounter = 0;

        function changeBy(val) {

          privateCounter += val;

        }

        return {

          increment: function () {

            changeBy(1);

          },

          decrement: function () {

            changeBy(-1);

          },

          value: function () {

            return privateCounter;

          },

        };

      })();

      console.log(counter.value()); // logs 0

      counter.increment();

      counter.increment();

      console.log(counter.value()); // logs 2

      counter.decrement();

      console.log(counter.value()); // logs 1