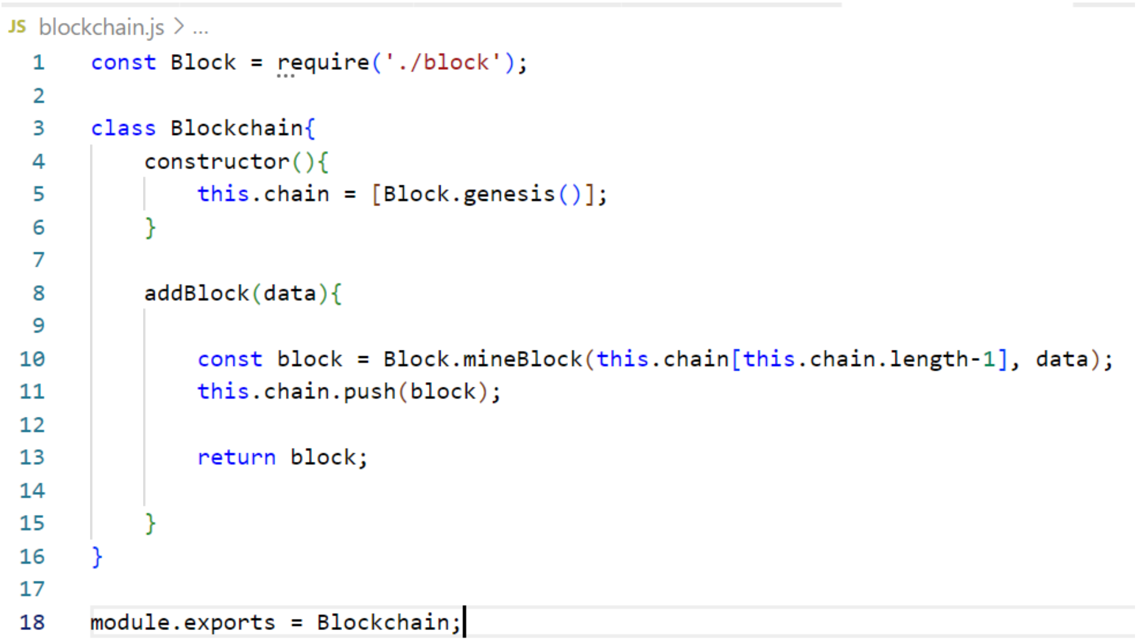
BlockChain

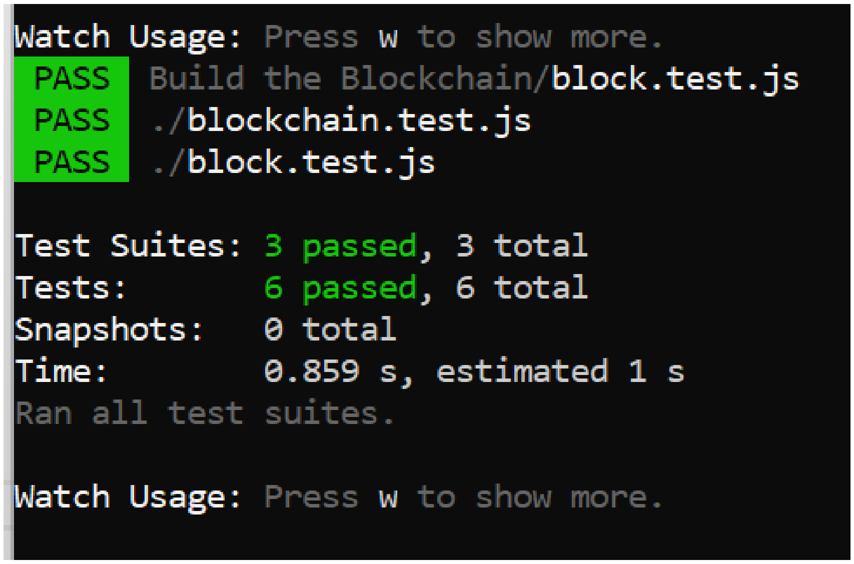
Build the Blockchain - the Chain:

Build the Blockchain Class:



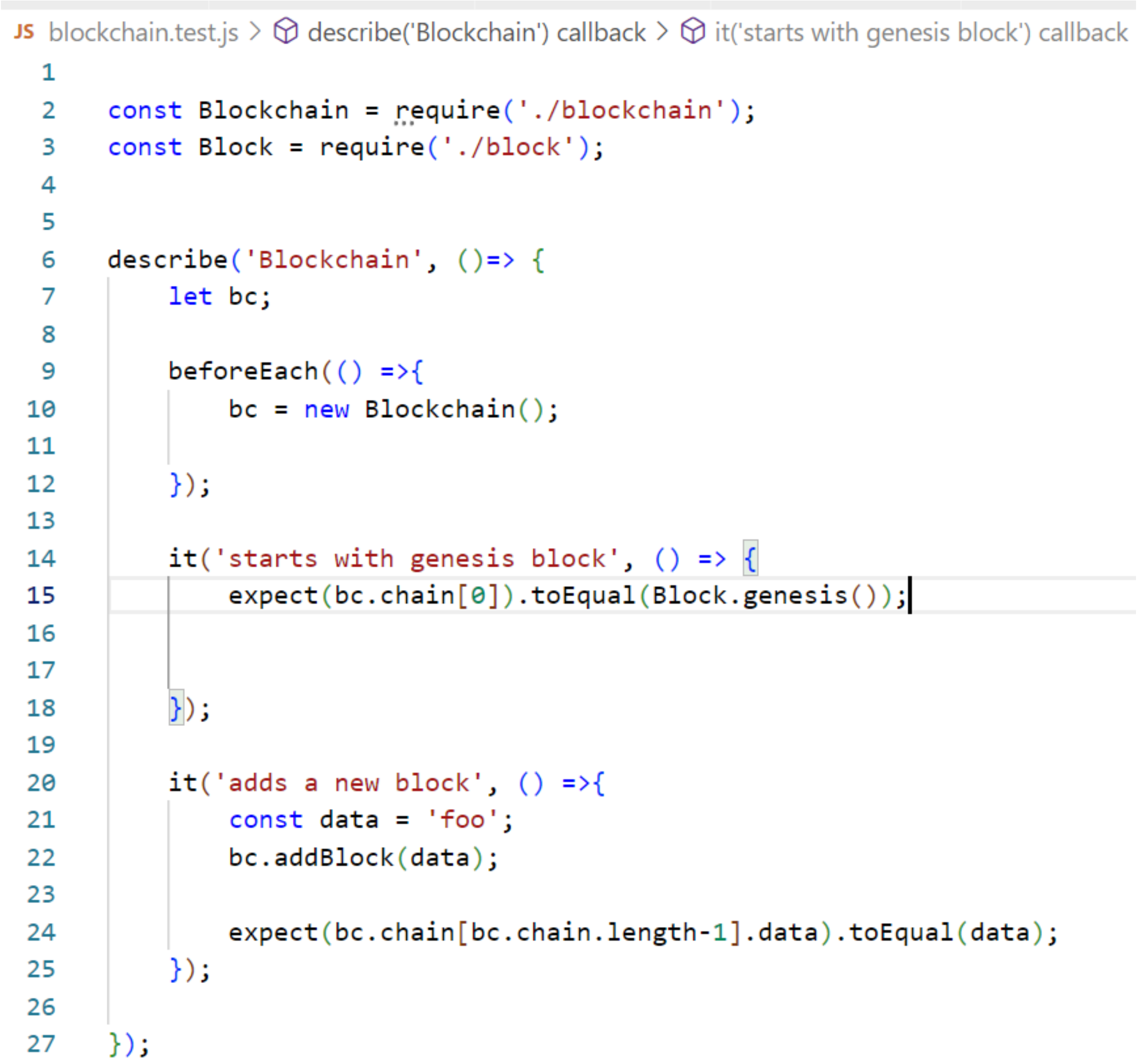
```javascript
const Block = require('./block');

class Blockchain{
    constructor(){
        this.chain = [Block.genesis()];
    }

    addBlock(data){

        const block = Block.mineBlock(this.chain[this.chain.length-1], data);
        this.chain.push(block);

        return block;

    }
}

module.exports = Blockchain;
```

Test the Blockchain:



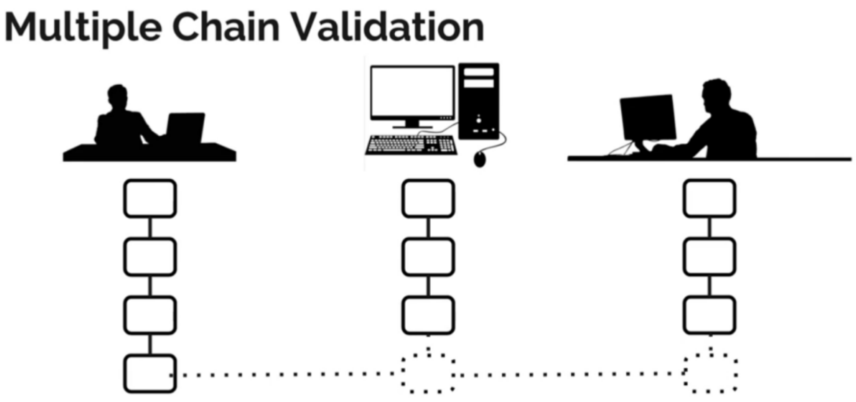
```
Watch Usage: Press w to show more.
 PASS  Build the Blockchain/block.test.js
 PASS  ./blockchain.test.js
 PASS  ./block.test.js

Test Suites: 3 passed, 3 total
Tests:       6 passed, 6 total
Snapshots:   0 total
Time:        0.859 s, estimated 1 s
Ran all test suites.

Watch Usage: Press w to show more.
```

BlockChain

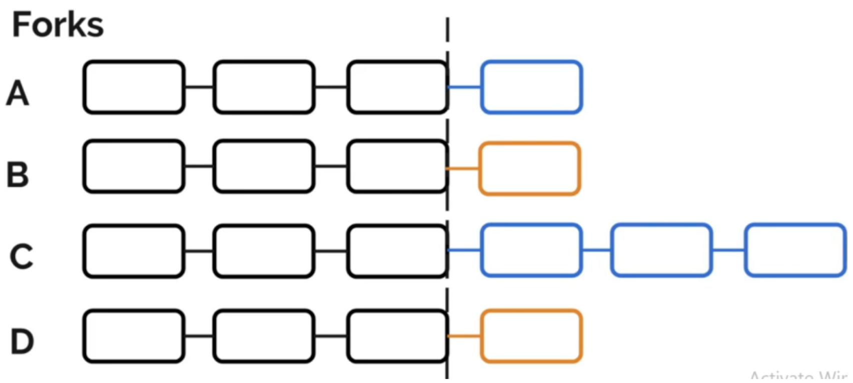

JS blockchain.test.js > describe('Blockchain') callback > it('starts with genesis block') callback

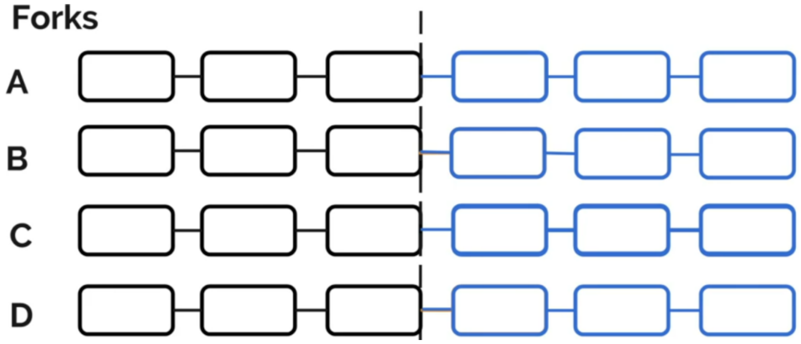
```

const Blockchain = require('./blockchain');
const Block = require('./block');


describe('Blockchain', ()=> {
    let bc;

    beforeEach(() =>{
        bc = new Blockchain();

    });

    it('starts with genesis block', () => {
        expect(bc.chain[0]).toEqual(Block.genesis());


    });

    it('adds a new block', () =>{
        const data = 'foo';
        bc.addBlock(data);

        expect(bc.chain[bc.chain.length-1].data).toEqual(data);
    });

});
```

Multiple Chain Validation:

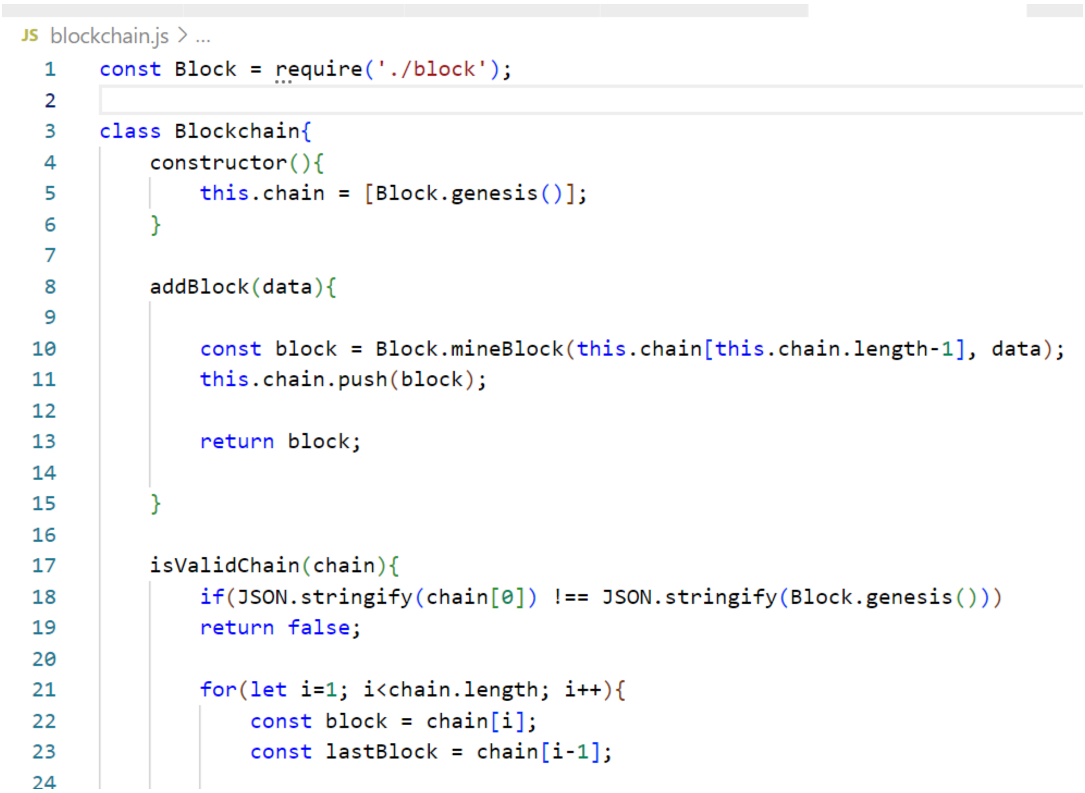


BlockChain

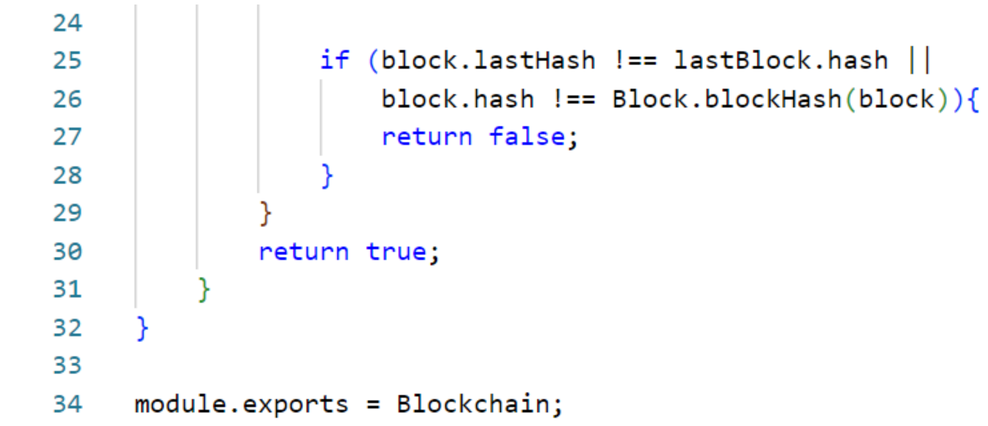




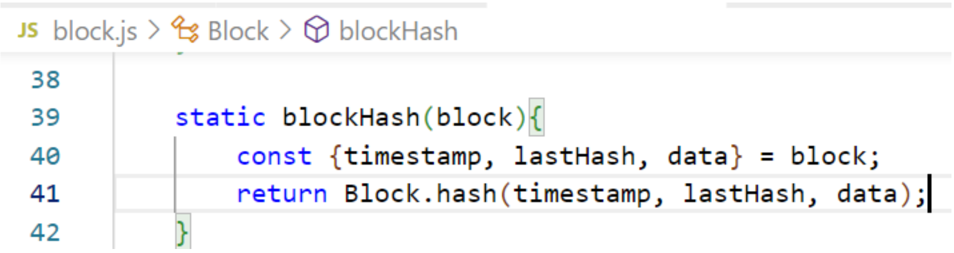
Chain Validation:



```
const Block = require('./block');

class Blockchain{
    constructor(){
        this.chain = [Block.genesis()];
    }

    addBlock(data){

        const block = Block.mineBlock(this.chain[this.chain.length-1], data);
        this.chain.push(block);

        return block;

    }

    isValidChain(chain){
        if(JSON.stringify(chain[0]) !== JSON.stringify(Block.genesis()))
        return false;

        for(let i=1; i<chain.length; i++){
            const block = chain[i];
            const lastBlock = chain[i-1];

```

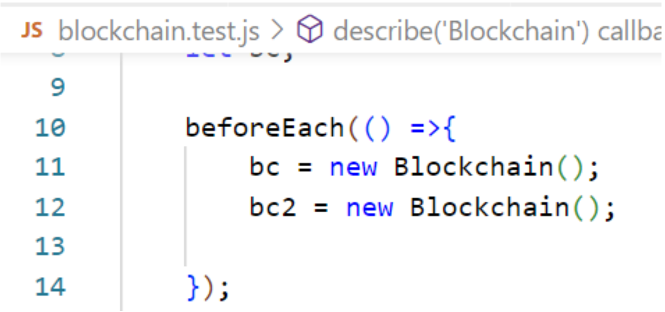


```
            if (block.lastHash !== lastBlock.hash ||
                block.hash !== Block.blockHash(block)){
                return false;
            }
        }
        return true;
    }
}

module.exports = Blockchain;
```

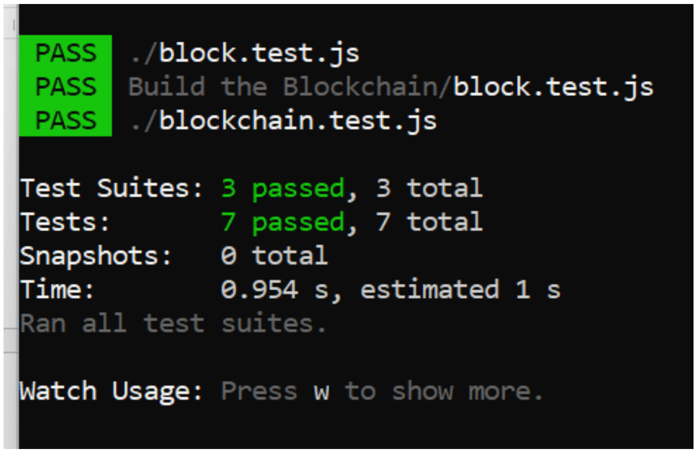


```
    static blockHash(block){
        const {timestamp, lastHash, data} = block;
        return Block.hash(timestamp, lastHash, data);
    }
```

Test Chain Validation:

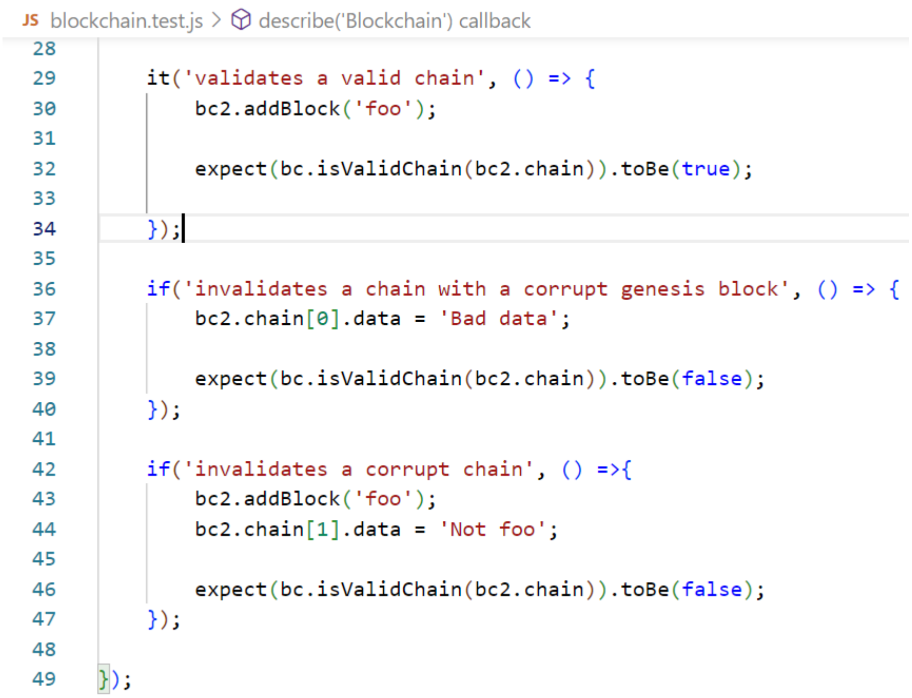


```

    beforeEach(() =>{
        bc = new Blockchain();
        bc2 = new Blockchain();

    });
```



```
 PASS  ./block.test.js
 PASS  Build the Blockchain/block.test.js
 PASS  ./blockchain.test.js

Test Suites: 3 passed, 3 total
Tests:       7 passed, 7 total
Snapshots:   0 total
Time:        0.954 s, estimated 1 s
Ran all test suites.

Watch Usage: Press w to show more.
```

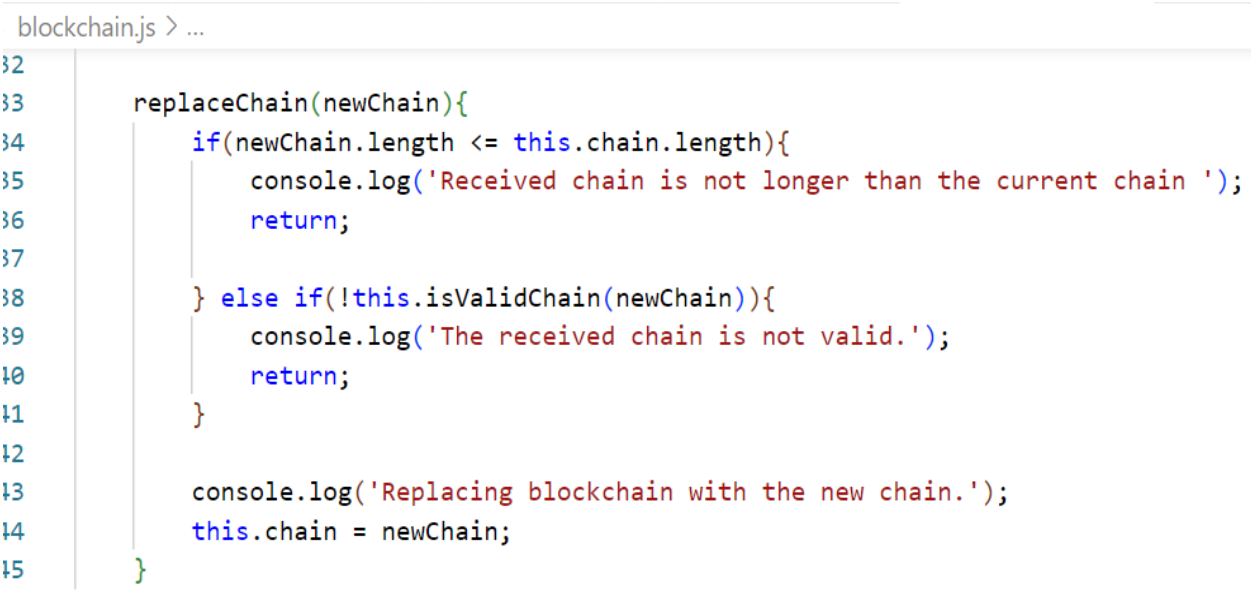
BlockChain



JS blockchain.test.js > describe('Blockchain') callback

```javascript

    it('validates a valid chain', () => {
        bc2.addBlock('foo');

        expect(bc.isValidChain(bc2.chain)).toBe(true);

    });

    if('invalidates a chain with a corrupt genesis block', () => {
        bc2.chain[0].data = 'Bad data';

        expect(bc.isValidChain(bc2.chain)).toBe(false);
    });

    if('invalidates a corrupt chain', () =>{
        bc2.addBlock('foo');
        bc2.chain[1].data = 'Not foo';

        expect(bc.isValidChain(bc2.chain)).toBe(false);
    });

});
```

Replace the Chain:

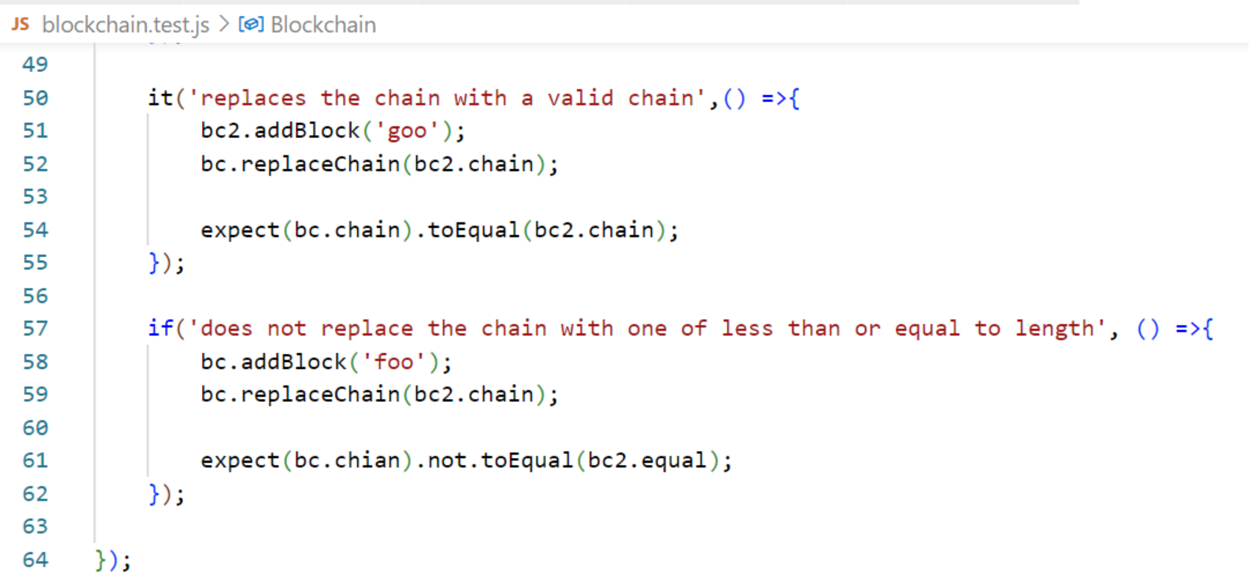


blockchain.js > ...

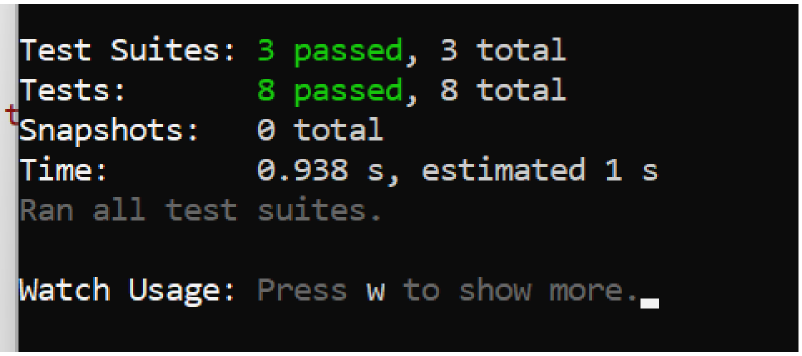
```javascript

    replaceChain(newChain){
        if(newChain.length <= this.chain.length){
            console.log('Received chain is not longer than the current chain ');
            return;

        } else if(!this.isValidChain(newChain)){
            console.log('The received chain is not valid.');
            return;
        }

        console.log('Replacing blockchain with the new chain.');
        this.chain = newChain;
    }
```

BlockChain

Test Chain Replacement:



```
    it('replaces the chain with a valid chain',() =>{
        bc2.addBlock('goo');
        bc.replaceChain(bc2.chain);

        expect(bc.chain).toEqual(bc2.chain);
    });

    if('does not replace the chain with one of less than or equal to length', () =>{
        bc.addBlock('foo');
        bc.replaceChain(bc2.chain);

        expect(bc.chian).not.toEqual(bc2.equal);
    });

});
```



```
Test Suites: 3 passed, 3 total
Tests:       8 passed, 8 total
Snapshots:   0 total
Time:        0.938 s, estimated 1 s
Ran all test suites.

Watch Usage: Press w to show more.
```