"use strict";

const SHOW = "SHOW\_PRICE";

const UPDATE = "UPDATE\_USD\_PRICE";

let fs = require('fs');

let EventEmitter = require('events');

function readJsonFromFile(fileName) {

try{

const data = fs.readFileSync(fileName, 'utf8');

return JSON.parse(data);

} catch(error){

console.error(`Error reading JSON file: ${error.message}`);

return {};

}

}

class CurrencyConverter extends EventEmitter {

static calculateRates(usdPrices) {

let rates = {};

let usdMap = {};

// Calculate USD conversion rates and store them for cross conversion

for (let i in usdPrices) {

let o = usdPrices[i];

let sym = o['asset\_id\_quote'];

let usdRate = o['rate'];

rates[`USD-${sym}`] = usdRate;

rates[`${sym}-USD`] = 1 / usdRate;

usdMap[sym] = usdRate;

}

// Calculate direct crypto-to-crypto conversion rates

let symbols = Object.keys(usdMap);

for (let from of symbols) {

for (let to of symbols) {

if (from !== to) {

let tag = `${from}-${to}`;

rates[tag] = usdMap[to] / usdMap[from]

}

}

}

return rates;

}

constructor(coin2USD) {

super();

this.rates = this.constructor.calculateRates(coin2USD.rates);

this.on(SHOW, (o) => {

console.log("SHOW event received.");

console.log(o);

const { from, to } = o;

try {

let rate = this.convert(1, from, to);

console.log(`1 ${from} is worth ${rate} ${to}`);

} catch (e) {

console.error(e.message);

}

});

this.on(UPDATE, (o) => {

const { sym, usdPrice } = o;

if (!sym || !usdPrice || usdPrice <= 0) {

console.error("Invalid update parameters.");

return;

}

console.log(`Updating ${sym} price to ${usdPrice} USD.`);

// Update USD rates

// complete the equality

this.rates[`USD-${sym}`] = usdPrice;

this.rates[`${sym}-USD`] = 1/ usdPrice;

// Recalculate all crypto-to-crypto rates

const symbols = Object.keys(this.rates)

.filter(key => key.startsWith('USD-'))

.map(key => key.split('-')[1]);

console.log("symbols" , symbols);

for (let from of symbols) {

for (let to of symbols) {

if (from !== to) {

this.rates[`${from}-${to}`] = this.rates[`USD-${to}`] / this.rates[`USD-${from}`];

}

}

}

console.log("Rates updated successfully.");

});

}

convert(amount, fromUnits, toUnits) {

let tag = `${fromUnits}-${toUnits}`;

let rate = this.rates[tag];

if (rate === undefined) {

throw new Error(`Rate for ${tag} not found`);

}

return rate \* amount;

}

}

// All prices listed are in USD

// write here your JSON File Path (rates.json)

const PATH = './rates.json'

let cnv = new CurrencyConverter(readJsonFromFile(PATH));

console.log(cnv.rates);

console.log("====================================================================");

function test(amt, from, to) {

console.log(`${amt} ${from} is worth ${cnv.convert(amt, from, to)} ${to}.`);

}

test(4000, 'ETH', 'BTC');

test(200, 'BTC', 'EOS');

console.log("====================================================================");

// Test event handling

cnv.emit(SHOW, { from: "EOS", to: "BTC" });

console.log("====================================================================");

cnv.emit(SHOW, { from: "EOS", to: "ETH" });

console.log("====================================================================");

cnv.emit(SHOW, { from: "ETC", to: "ETH" });

console.log("====================================================================");

cnv.emit(SHOW, { from: "LTC", to: "BTC" });

console.log("====================================================================");

cnv.emit(UPDATE, { sym: "BTC", usdPrice: 50000 });

console.log("====================================================================");

cnv.emit(SHOW, { from: "LTC", to: "BTC" });