Texto

Descrição gerada automaticamente

import yargs from "yargs";

import { hideBin } from "yargs/helpers";

import \* as cfg from "./config.js";

import \* as iso from "./iso\_country\_routines.js";

import got from "got";

// Note: hideBin is a shorthand for process.argv.slice(2)

// - bypass the first two arguments

const argv = yargs(hideBin(process.argv))

  .options({

    refresh: {

      demandOption: false,

      describe: "User wants a fresh download of data",

      boolean: true,

    },

  })

  .help()

  .alias("help", "h")

  .parse();

const baseFunction = async () => {

  try {

    let fileStats = await iso.fileStatsFromFSPromise(cfg.countries);

    let jsonCountries = "";

    let codeQuantity = 0;

    if (fileStats === undefined || argv.refresh) {

      jsonCountries = await iso.getJSONFromWWWPromise(cfg.isocountries);

      let countryList = [];

      await Promise.all(

        jsonCountries.map(async (country) => {

          countryList.push(country);

        })

      );

      codeQuantity = countryList.length;

      await iso.writeFileFromFSPromise(cfg.countries, countryList);

      fileStats = await iso.fileStatsFromFSPromise(cfg.countries);

      console.log("A new " + cfg.countries + " file was written.");

      console.log(cfg.countries + " was created on " + fileStats.birthtime);

      console.log("There are " + codeQuantity + " in " + cfg.countries);

    } else {

      jsonCountries = await iso.getJSONFromWWWPromise(cfg.isocountries);

      let countryList = [];

      await Promise.all(

        jsonCountries.map(async (country) => {

          countryList.push(country);

        })

      );

      codeQuantity = countryList.length;

      console.log(

        "An existing " + cfg.countries + " was read from the file system"

      );

      console.log(cfg.countries + " was created on " + fileStats.birthtime);

      console.log("There are " + codeQuantity + " in " + cfg.countries);

    }

  } catch (err) {

    console.log(err.message);

  }

};

baseFunction();

import got from "got";

import { promises as fsp } from "fs";

const fileStatsFromFSPromise = async (fname) => {

  let stats;

  try {

    stats = await fsp.stat(fname);

  } catch (err) {

    err.code === "ENOENT" //doesnt exist

      ? console.log(`${fname} does not exist`)

      : console.log(err.message);

  }

  return stats;

};

const getJSONFromWWWPromise = (url) => got(url).json();

const writeFileFromFSPromise = async (fname, ...rawdata) => {

  let filehandle;

  try {

    filehandle = await fsp.open(fname, "w");

    let dataToWrite = "";

    rawdata.forEach((element) => (dataToWrite += JSON.stringify(element))); // concatentate

    await fsp.writeFile(fname, dataToWrite); // returns promise

  } catch (err) {

    console.log(err);

  } finally {

    if (filehandle !== undefined) {

      await filehandle.close();

    }

  }

};

const readFileFromFSPPromise = async (fname) => {

  let rawDta;

  try {

    rawDta = fsp.readFile(fname);

  } catch (error) {

    console.log(error);

  } finally {

    if (rawDta != undefined) {import got from "got";

import { promises as fsp } from "fs";

const fileStatsFromFSPromise = async (fname) => {

  let stats;

  try {

    stats = await fsp.stat(fname);

  } catch (err) {

    err.code === "ENOENT" //doesnt exist

      ? console.log(`${fname} does not exist`)

      : console.log(err.message);

  }

  return stats;

};

const getJSONFromWWWPromise = (url) => got(url).json();

const writeFileFromFSPromise = async (fname, ...rawdata) => {

  let filehandle;

  try {

    filehandle = await fsp.open(fname, "w");

    let dataToWrite = "";

    rawdata.forEach((element) => (dataToWrite += JSON.stringify(element))); // concatentate

    await fsp.writeFile(fname, dataToWrite); // returns promise

  } catch (err) {

    console.log(err);

  } finally {

    if (filehandle !== undefined) {

      await filehandle.close();

    }

  }

};

const readFileFromFSPPromise = async (fname) => {

  let rawDta;

  try {

    rawDta = fsp.readFile(fname);

  } catch (error) {

    console.log(error);

  } finally {

    if (rawDta != undefined) {

      return rawDta;

    }

  }

};

export {

  fileStatsFromFSPromise,

  getJSONFromWWWPromise,

  writeFileFromFSPromise,

  readFileFromFSPPromise,

};

      return rawDta;

    }

  }

};

export {

  fileStatsFromFSPromise,

  getJSONFromWWWPromise,

  writeFileFromFSPromise,

  readFileFromFSPPromise,

};