node-epics-ca: An EPICS Channel Access client for Node.js

Lin Wang

wanglin@ihep.ac.cn

Controls Group, Accelerator System Division, CSNS

Dongguan Campus of IHEP

EPICS Collaboration Meeting 2023

Apr 24 – 28, 2023, hosted by the Fermilab

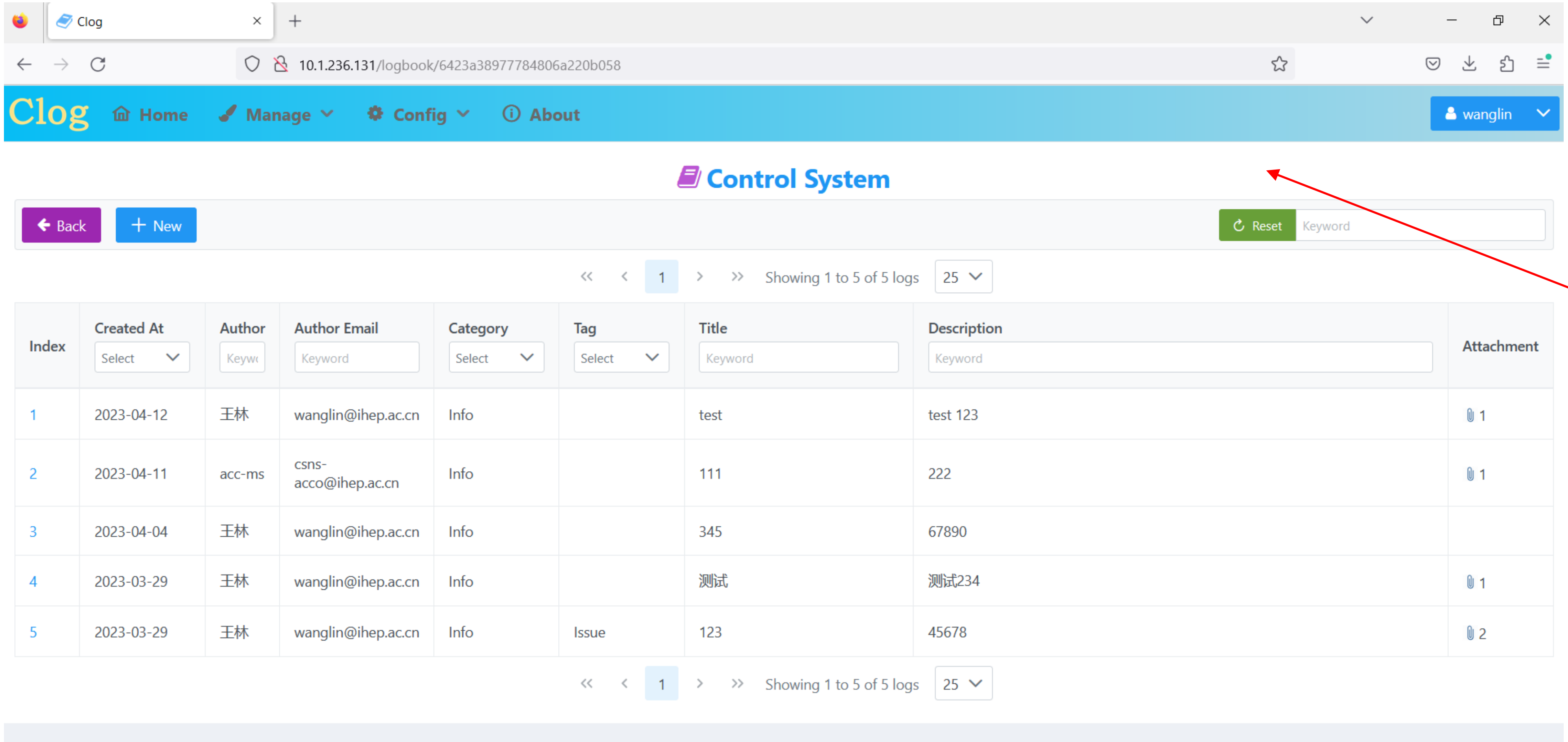
CSNS CHINESE ACADEMY OF SCIENCES

Content

- （1）Scenario
- （2）Design
- （3）Implementation
- （4）Usage example
- （5）Potential problems

Scenario

- **Node.js web applications or services may want to access PV values in an IOC.**
- **Take Clog as an example, maybe someday it is required to display CSNS beam power PV value on the web page.**



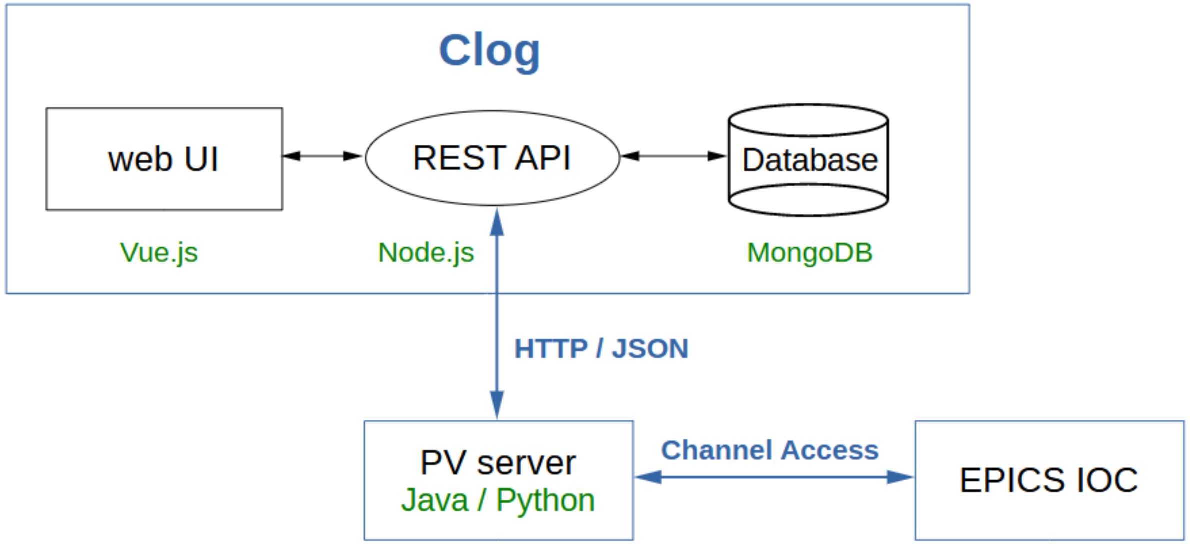
Control System

Index	Created At	Author	Author Email	Category	Tag	Title	Description	Attachment
	Select	Keyw	Keyword	Select	Select	Keyword	Keyword	
1	2023-04-12	王林	wanglin@ihep.ac.cn	Info		test	test 123	📎 1
2	2023-04-11	acc-ms	csns-acco@ihep.ac.cn	Info		111	222	📎 1
3	2023-04-04	王林	wanglin@ihep.ac.cn	Info		345	67890	
4	2023-03-29	王林	wanglin@ihep.ac.cn	Info		测试	测试234	📎 1
5	2023-03-29	王林	wanglin@ihep.ac.cn	Info	Issue	123	45678	📎 2

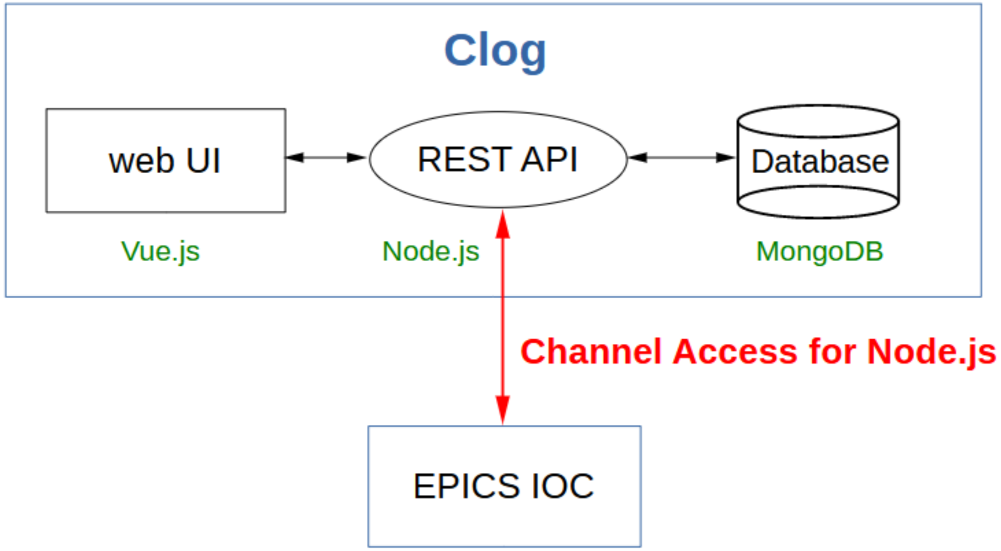
Power: 140kW

Scenario

- **Access PV values from IOC in Node.js?**
  - **Option #1: From a PV server**
  - **Option #2: Directly from the IOC**
- **A Channel Access client for Node.js is needed in Option #2.**



Access PV values from a PV server



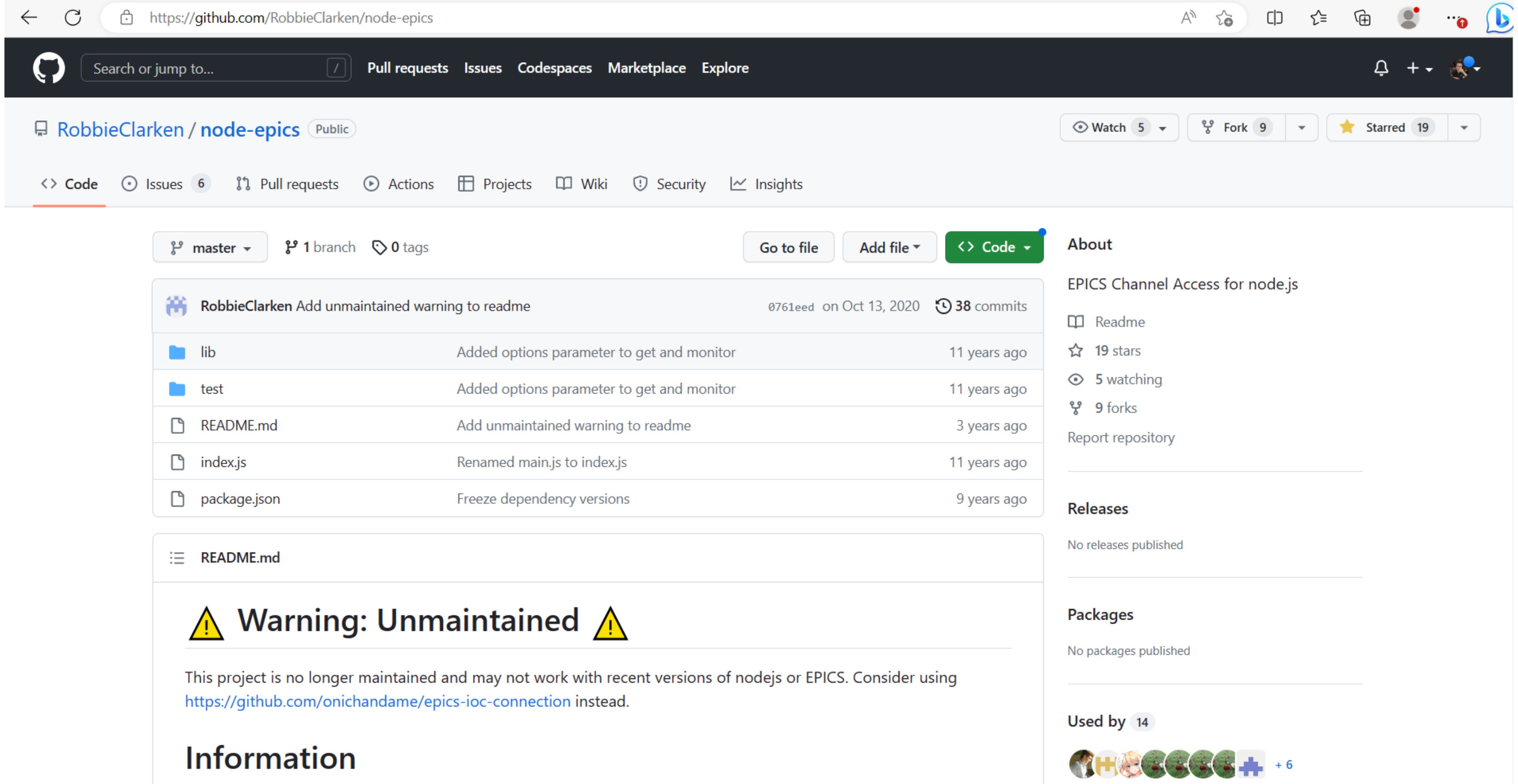
Access PV values directly from IOC

Search for existing Node.js CA client

- node-epics
  - https://github.com/RobbieClarken/node-epics
  - https://www.npmjs.com/package/epics
  - FFI implementation
- epics-ioc-connection
  - https://github.com/onichandame/epics-ioc-connection
  - https://www.npmjs.com/package/epics-ioc-connection
  - FFI implementation
- epics-tca
  - https://www.npmjs.com/package/epics-tca
  - Pure TypeScript/JavaScript implementation

Search for existing Node.js CA client

- node-epics
  - Not updated for almost 10 years
  - Has been announced unmaintained



https://github.com/RobbieClarken/node-epics

RobbieClarken / node-epics Public

Watch 5 · Fork 9 · Starred 19

Code · Issues 6 · Pull requests · Actions · Projects · Wiki · Security · Insights

master · 1 branch · 0 tags · Go to file · Add file · Code

RobbieClarken Add unmaintained warning to readme · 0761eed on Oct 13, 2020 · 38 commits

lib	Added options parameter to get and monitor	11 years ago
test	Added options parameter to get and monitor	11 years ago
README.md	Add unmaintained warning to readme	3 years ago
index.js	Renamed main.js to index.js	11 years ago
package.json	Freeze dependency versions	9 years ago

README.md

⚠ Warning: Unmaintained ⚠

This project is no longer maintained and may not work with recent versions of nodejs or EPICS. Consider using https://github.com/onichandame/epics-ioc-connection instead.

Information

About

EPICS Channel Access for node.js

Readme
19 stars
5 watching
9 forks
Report repository

Releases

No releases published

Packages

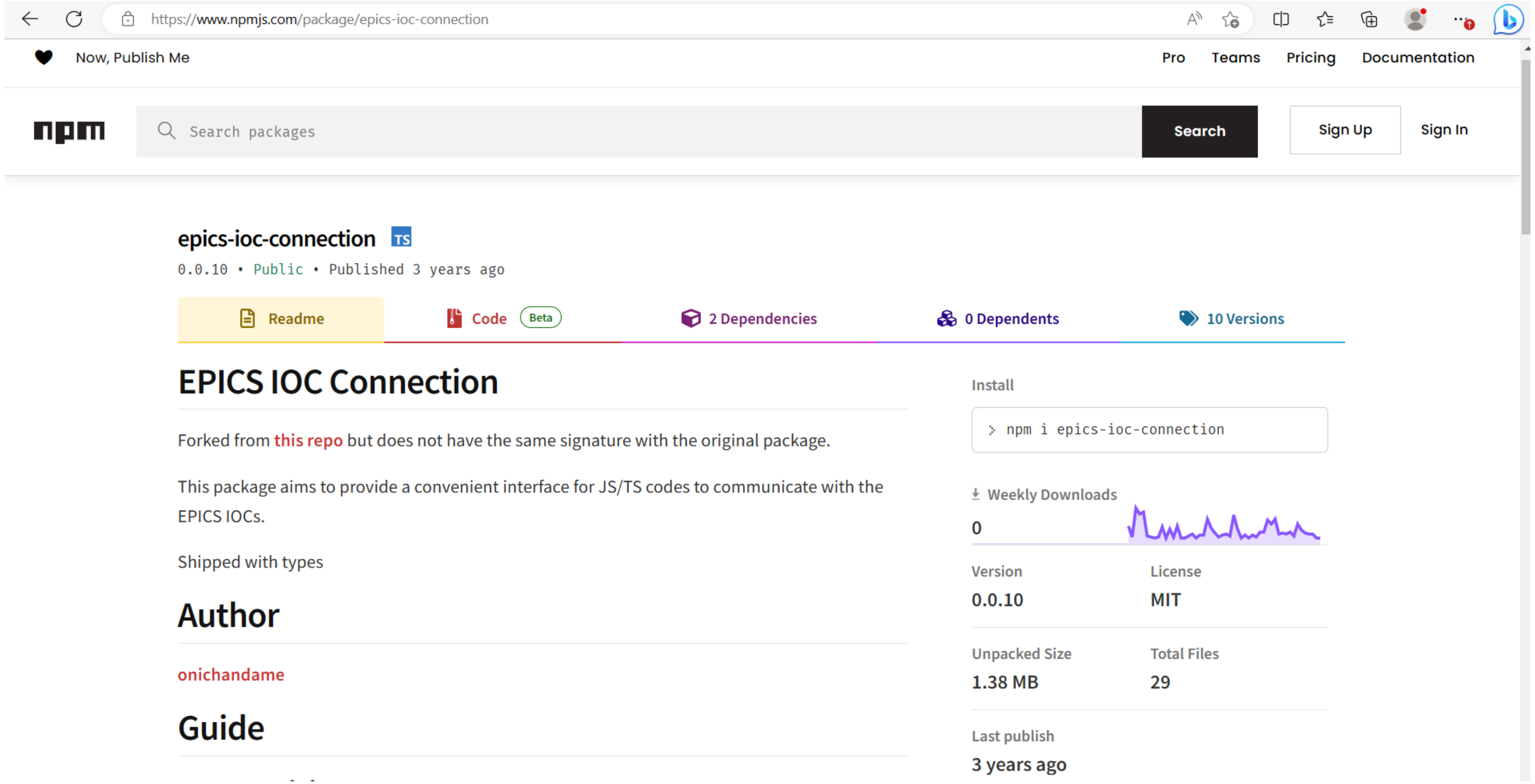
No packages published

Used by 14

+ 6

Search for existing Node.js CA client

- epics-ioc-connection
  - Not updated for almost 3 years
  - May not work for recent Node.js versions



epics-ioc-connection TS

0.0.10 • Public • Published 3 years ago

EPICS IOC Connection

Forked from this repo but does not have the same signature with the original package.

This package aims to provide a convenient interface for JS/TS codes to communicate with the EPICS IOCs.

Shipped with types

Author

onichandame

Guide

Install

```
> npm i epics-ioc-connection
```

Weekly Downloads
0

Version
0.0.10

License
MIT

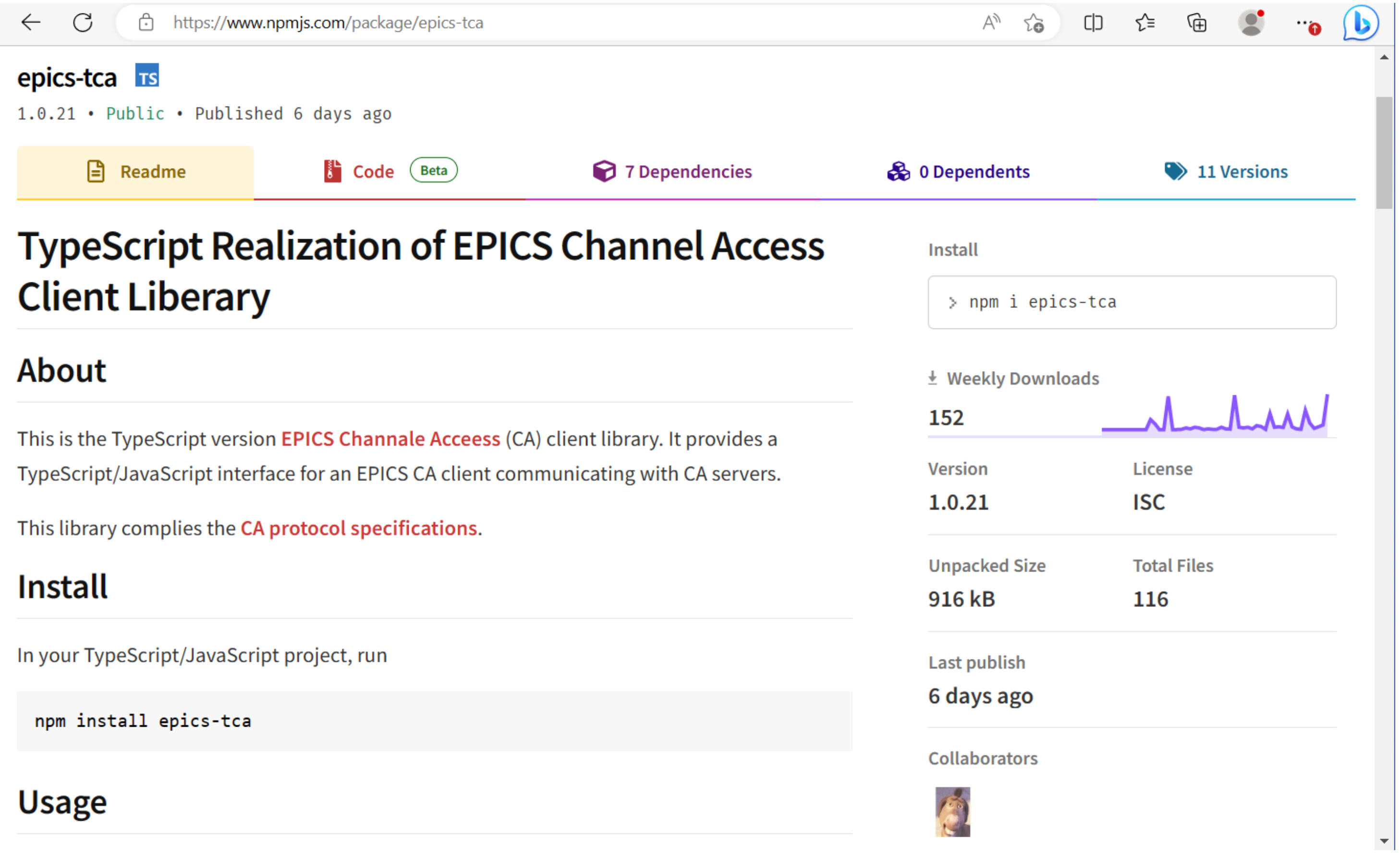
Unpacked Size
1.38 MB

Total Files
29

Last publish
3 years ago

Search for existing Node.js CA client

- epics-tca
  - A pure TypeScript / JavaScript implementation of EPICS CA
  - Developed by ORNL and is actively maintained



https://www.npmjs.com/package/epics-tca

epics-tca TS

1.0.21 • Public • Published 6 days ago

TypeScript Realization of EPICS Channel Access Client Liberary

About

This is the TypeScript version EPICS Channale Acceess (CA) client library. It provides a TypeScript/JavaScript interface for an EPICS CA client communicating with CA servers.

This library complies the CA protocol specifications.

Install

In your TypeScript/JavaScript project, run

```
npm install epics-tca
```

Usage

Install

```
> npm i epics-tca
```

Weekly Downloads

152

Version

1.0.21

License

ISC

Unpacked Size

916 kB

Total Files

116

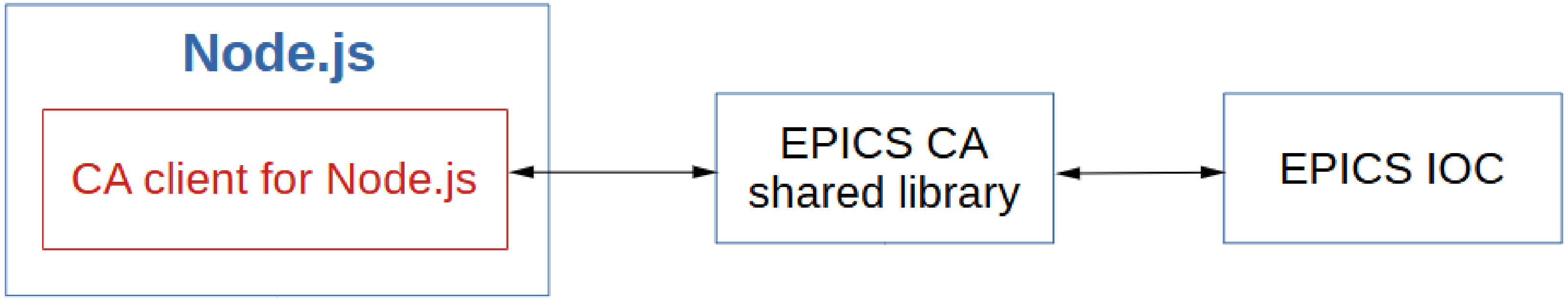
Last publish

6 days ago

Collaborators

Scenario

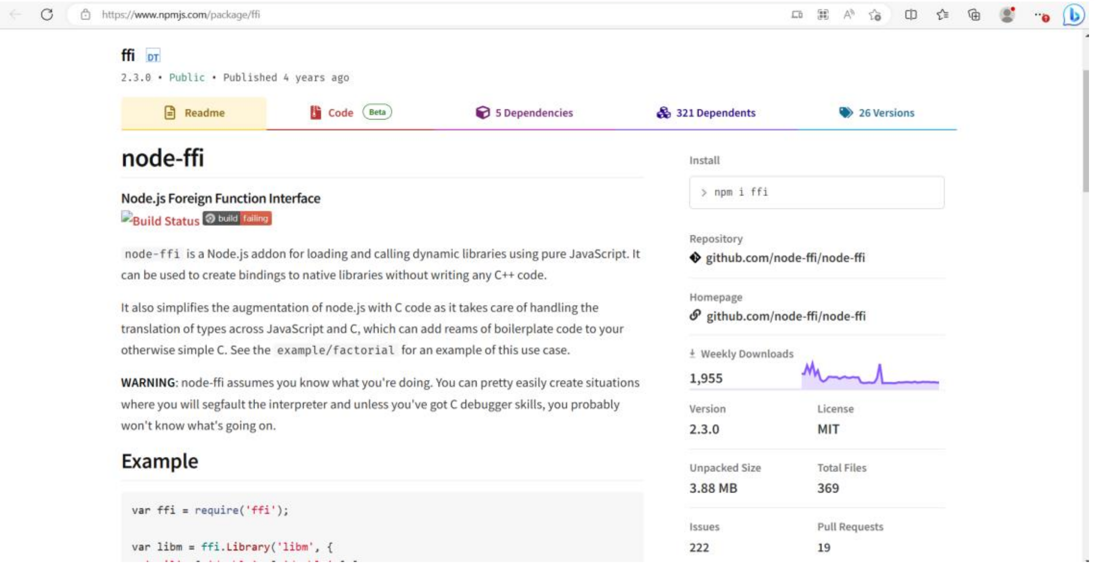
- CA client for Node.js is useful in Node.js applications.
- node-epics and epics-ioc-connection are very useful CA clients.
- However, node-epics and epics-ioc-connection are almost obsolete.
- We decided to develop a CA client for Node.js with FFI implementation based on node-epics and epics-ioc-connection. In other words, re-implement node-epics and epics-ioc-connection.
- Note: I am not aware of the existence epics-tca until I finished the development of node-epics-ca, so the work described in the presentation is done based on an assumption that there is no actively-maintained CA client for Node.js.



CA client for Node.js with FFI implementation

Design

- FFI libraries / packages for Node.js
  - ffi
    - https://www.npmjs.com/package/ffi
    - https://github.com/node-ffi/node-ffi
    - Not updated for nearly 4 years
    - Used by node-epics
  - ffi-napi
    - https://www.npmjs.com/package/ffi-napi
    - https://github.com/node-ffi-napi/node-ffi-napi
    - Not updated for nearly 2 years
    - Used by epics-ioc-connection
  - koffi
    - https://www.npmjs.com/package/koffi
    - https://github.com/Koromix/rygel/tree/master/src/koffi
    - https://koffi.dev/
    - Actively maintained by a software developer from France
    - I finally chose koffi as FFI library to access EPICS CA shared library



https://www.npmjs.com/package/ffi

ffi DT

2.3.0 • Public • Published 4 years ago

Readme | Code Beta | 5 Dependencies | 321 Dependents | 26 Versions

node-ffi

Node.js Foreign Function Interface

Build Status build failing

`node-ffi` is a Node.js addon for loading and calling dynamic libraries using pure JavaScript. It can be used to create bindings to native libraries without writing any C++ code.

It also simplifies the augmentation of node.js with C code as it takes care of handling the translation of types across JavaScript and C, which can add reams of boilerplate code to your otherwise simple C. See the `example/factorial` for an example of this use case.

WARNING: node-ffi assumes you know what you're doing. You can pretty easily create situations where you will segfault the interpreter and unless you've got C debugger skills, you probably won't know what's going on.

Example

```
var ffi = require('ffi');

var libm = ffi.Library('libm', {
```

Install

```
> npm i ffi
```

Repository
github.com/node-ffi/node-ffi

Homepage
github.com/node-ffi/node-ffi

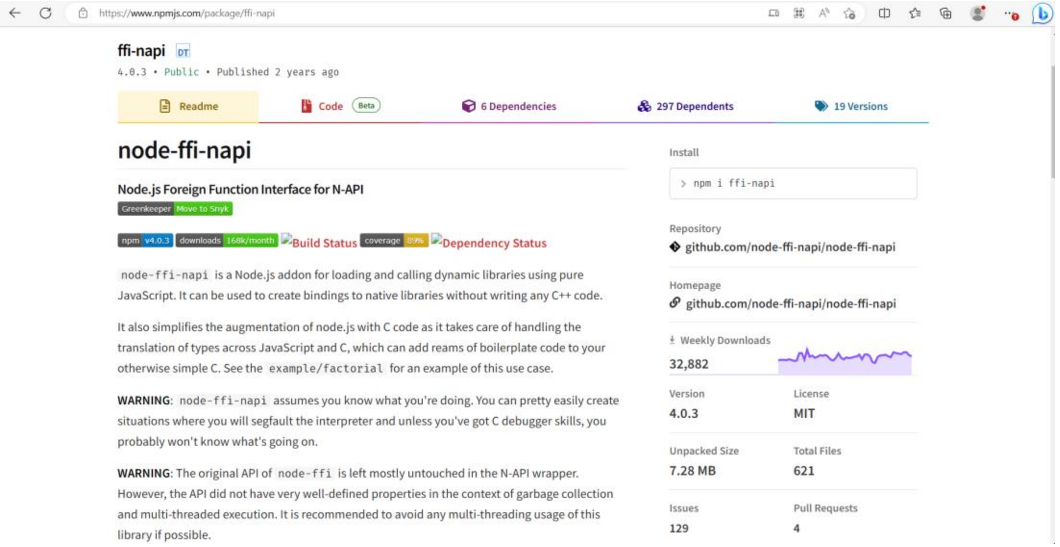
Weekly Downloads
1,955

Version	License
2.3.0	MIT

Unpacked Size	Total Files
3.88 MB	369

Issues	Pull Requests
222	19

ffi package



https://www.npmjs.com/package/ffi-napi

ffi-napi DT

4.0.3 • Public • Published 2 years ago

Readme | Code Beta | 6 Dependencies | 297 Dependents | 19 Versions

node-ffi-napi

Node.js Foreign Function Interface for N-API

Greenkeeper Move to Snyk

npm v4.0.3 downloads 168k/month Build Status coverage 89% Dependency Status

`node-ffi-napi` is a Node.js addon for loading and calling dynamic libraries using pure JavaScript. It can be used to create bindings to native libraries without writing any C++ code.

It also simplifies the augmentation of node.js with C code as it takes care of handling the translation of types across JavaScript and C, which can add reams of boilerplate code to your otherwise simple C. See the `example/factorial` for an example of this use case.

WARNING: `node-ffi-napi` assumes you know what you're doing. You can pretty easily create situations where you will segfault the interpreter and unless you've got C debugger skills, you probably won't know what's going on.

WARNING: The original API of `node-ffi` is left mostly untouched in the N-API wrapper. However, the API did not have very well-defined properties in the context of garbage collection and multi-threaded execution. It is recommended to avoid any multi-threading usage of this library if possible.

Install

```
> npm i ffi-napi
```

Repository
github.com/node-ffi-napi/node-ffi-napi

Homepage
github.com/node-ffi-napi/node-ffi-napi

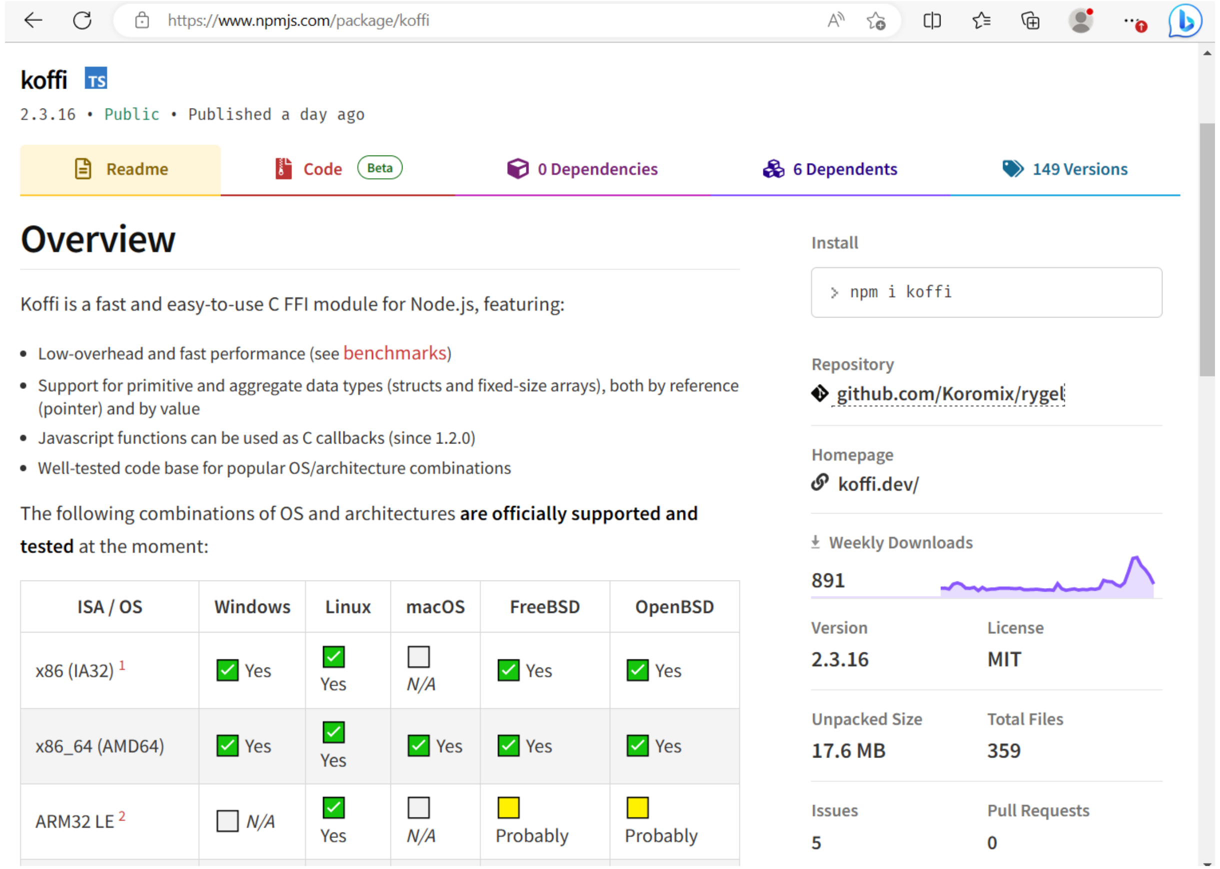
Weekly Downloads
32,882

Version	License
4.0.3	MIT

Unpacked Size	Total Files
7.28 MB	621

Issues	Pull Requests
129	4

ffi-napi package



https://www.npmjs.com/package/koffi

koffi TS

2.3.16 • Public • Published a day ago

Overview

Koffi is a fast and easy-to-use C FFI module for Node.js, featuring:

- Low-overhead and fast performance (see benchmarks)
- Support for primitive and aggregate data types (structs and fixed-size arrays), both by reference (pointer) and by value
- Javascript functions can be used as C callbacks (since 1.2.0)
- Well-tested code base for popular OS/architecture combinations

The following combinations of OS and architectures **are officially supported and tested** at the moment:

ISA / OS	Windows	Linux	macOS	FreeBSD	OpenBSD
x86 (IA32) [1]	✓ Yes	✓ Yes	□ *N/A*	✓ Yes	✓ Yes
x86_64 (AMD64)	✓ Yes	✓ Yes	✓ Yes	✓ Yes	✓ Yes
ARM32 LE [2]	□ *N/A*	✓ Yes	□ *N/A*	🟨 Probably	🟨 Probably

Install

```
> npm i koffi
```

Repository

github.com/Koromix/rygel

Homepage

koffi.dev/

Weekly Downloads

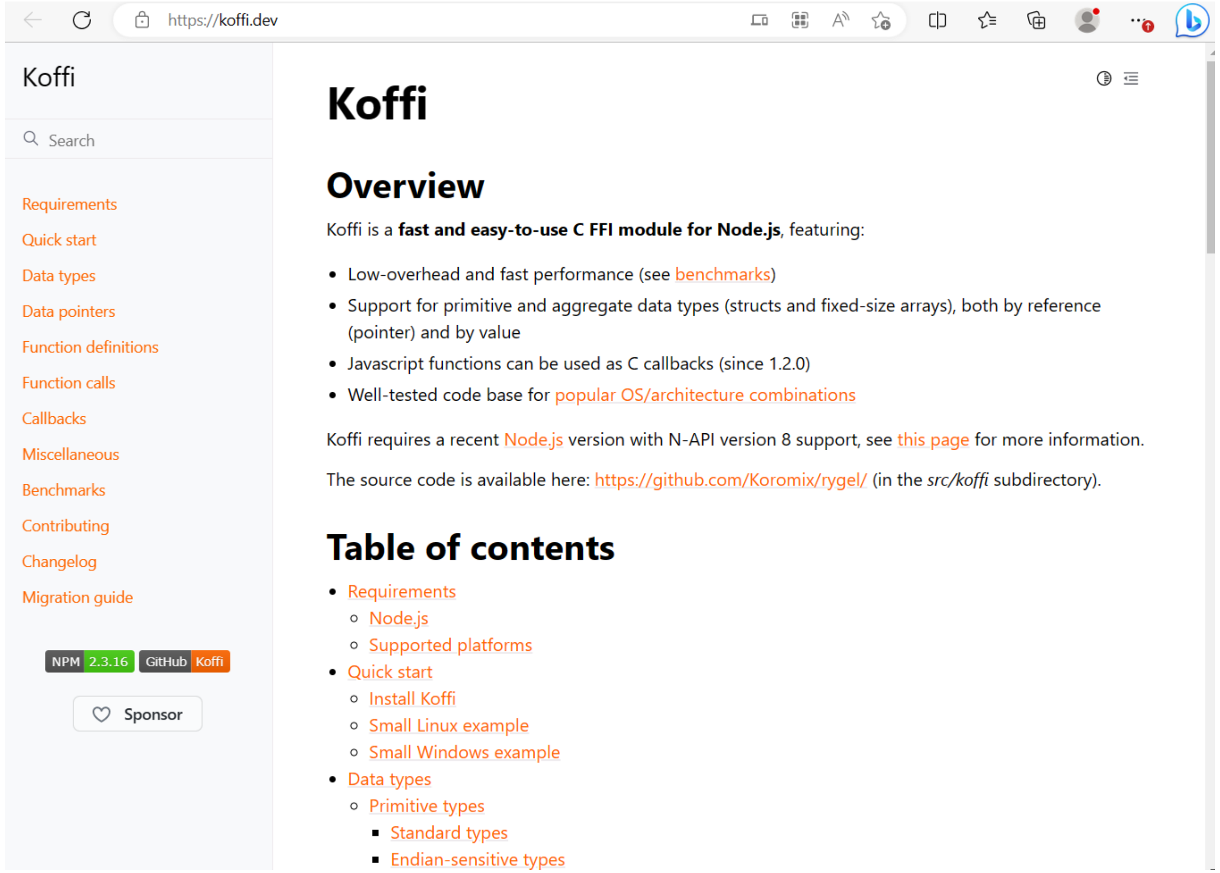
891

Version	License
2.3.16	MIT

Unpacked Size	Total Files
17.6 MB	359

Issues	Pull Requests
5	0

koffi package page



https://koffi.dev

Koffi

Search

Koffi

Overview

Koffi is a **fast and easy-to-use C FFI module for Node.js**, featuring:

- Low-overhead and fast performance (see benchmarks)
- Support for primitive and aggregate data types (structs and fixed-size arrays), both by reference (pointer) and by value
- Javascript functions can be used as C callbacks (since 1.2.0)
- Well-tested code base for popular OS/architecture combinations

Koffi requires a recent Node.js version with N-API version 8 support, see this page for more information.

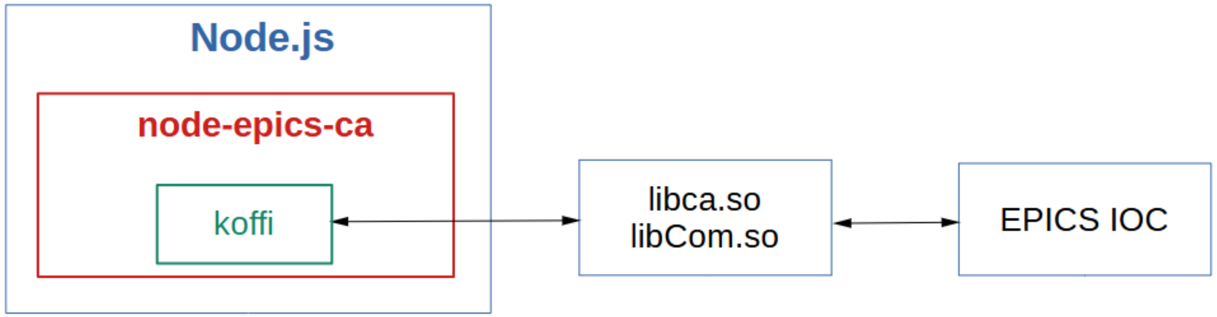
The source code is available here: https://github.com/Koromix/rygel/ (in the *src/koffi* subdirectory).

Table of contents

koffi documentation page

Design

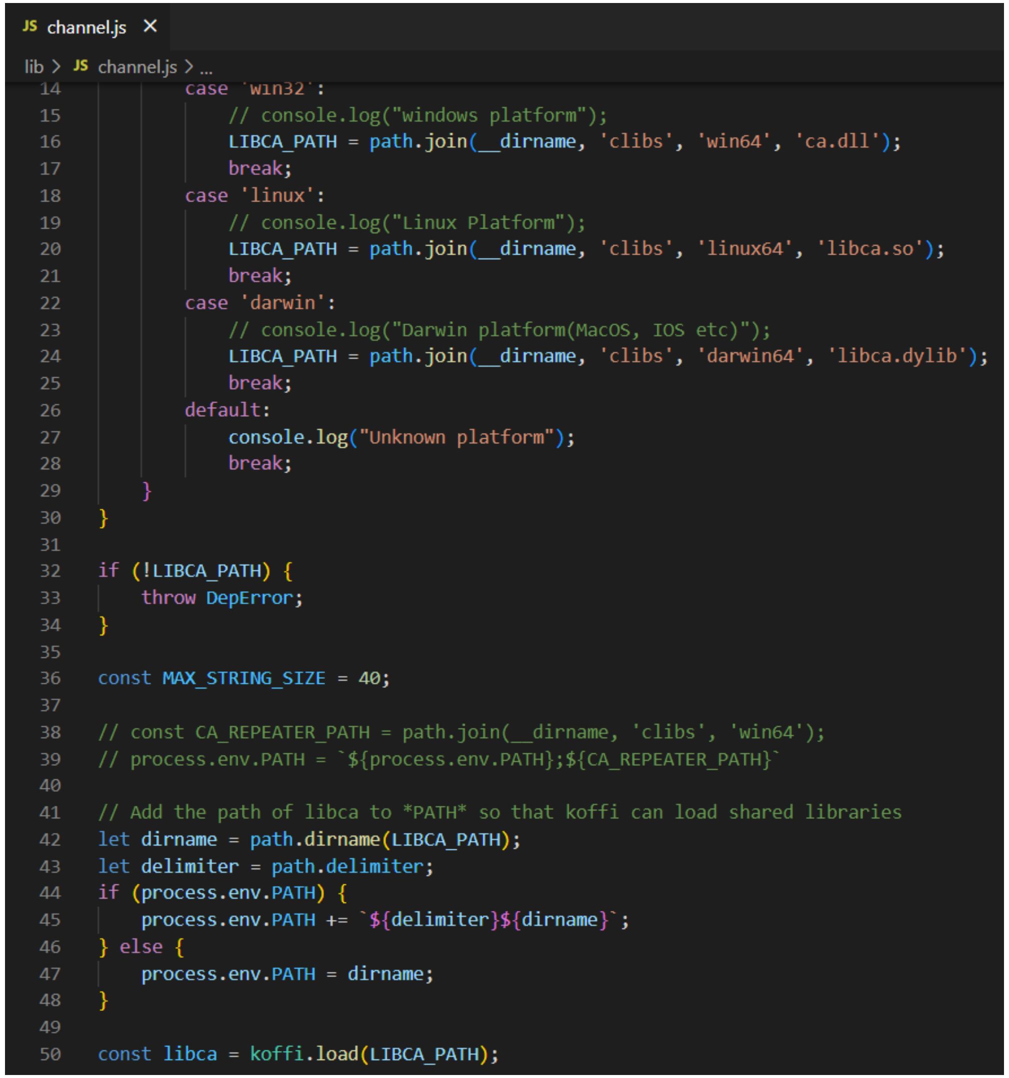
- node-epics-ca is developed based on node-epics and epics-ioc-connection.
- Use koffi as FFI library to access EPICS CA shared libraries
  - FFI libraries (node-ffi and node-ffi-napi) that node-epics and epics-ioc-connection use are obsolete
- Optimize the monitor() method
  - A PV can be monitored when its initial state is disconnected
- Take EPICS CA shared libraries directly from pyepics 3.5.1
  - https://github.com/pyepics/pyepics/tree/master/epics/clibs
- References
  - epics/ca.py in pyepics
  - EPICS CA tools: caget, caput, camonitor



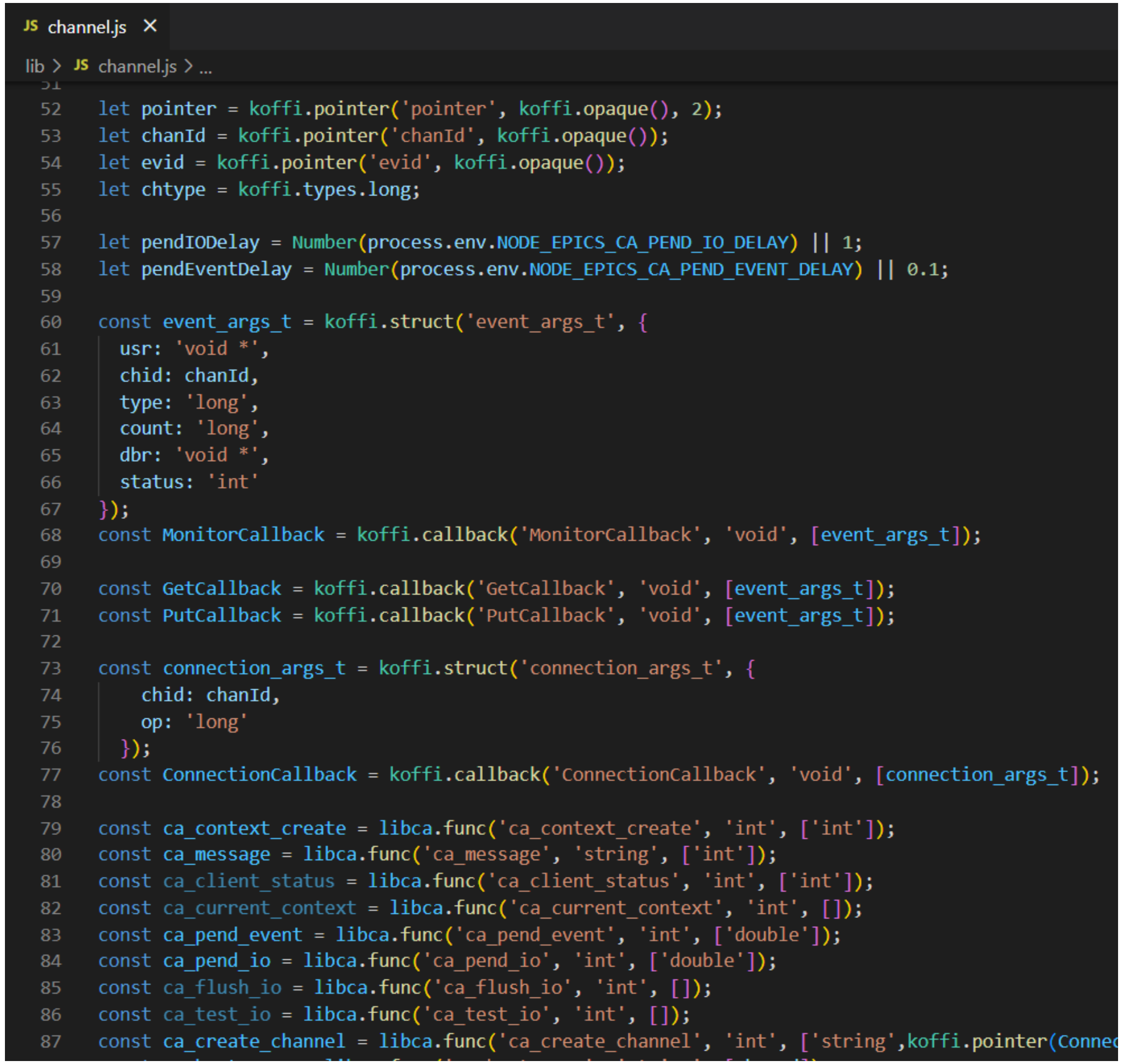
Interface of node-epics-ca

Implementation

- **Load the shared library and register functions to koffi**



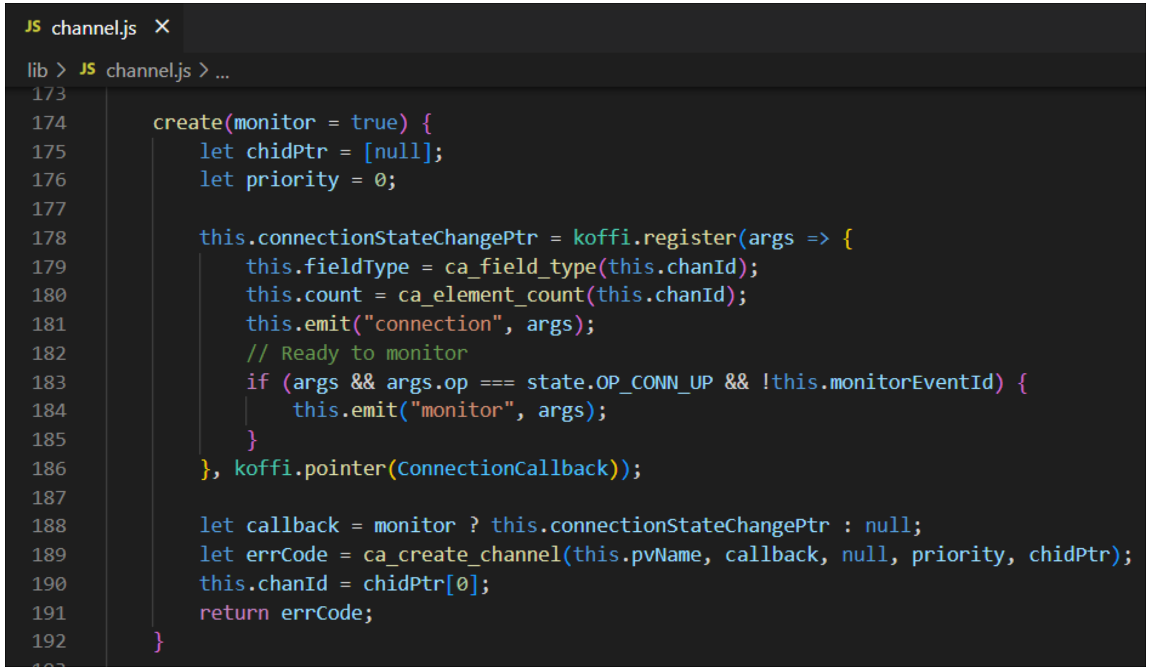
```javascript
        case 'win32':
            // console.log("windows platform");
            LIBCA_PATH = path.join(__dirname, 'clibs', 'win64', 'ca.dll');
            break;
        case 'linux':
            // console.log("Linux Platform");
            LIBCA_PATH = path.join(__dirname, 'clibs', 'linux64', 'libca.so');
            break;
        case 'darwin':
            // console.log("Darwin platform(MacOS, IOS etc)");
            LIBCA_PATH = path.join(__dirname, 'clibs', 'darwin64', 'libca.dylib');
            break;
        default:
            console.log("Unknown platform");
            break;
    }
}

if (!LIBCA_PATH) {
    throw DepError;
}

const MAX_STRING_SIZE = 40;

// const CA_REPEATER_PATH = path.join(__dirname, 'clibs', 'win64');
// process.env.PATH = `${process.env.PATH};${CA_REPEATER_PATH}`

// Add the path of libca to *PATH* so that koffi can load shared libraries
let dirname = path.dirname(LIBCA_PATH);
let delimiter = path.delimiter;
if (process.env.PATH) {
    process.env.PATH += `${delimiter}${dirname}`;
} else {
    process.env.PATH = dirname;
}

const libca = koffi.load(LIBCA_PATH);
```



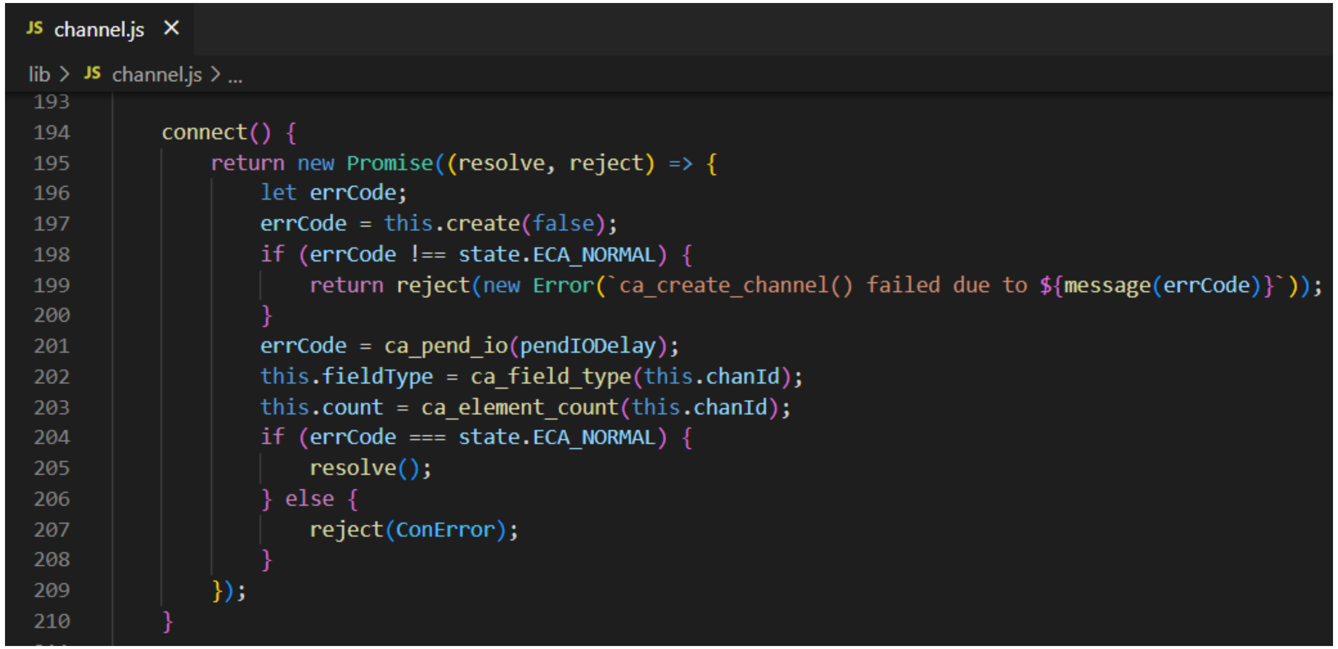
```javascript
let pointer = koffi.pointer('pointer', koffi.opaque(), 2);
let chanId = koffi.pointer('chanId', koffi.opaque());
let evid = koffi.pointer('evid', koffi.opaque());
let chtype = koffi.types.long;

let pendIODelay = Number(process.env.NODE_EPICS_CA_PEND_IO_DELAY) || 1;
let pendEventDelay = Number(process.env.NODE_EPICS_CA_PEND_EVENT_DELAY) || 0.1;

const event_args_t = koffi.struct('event_args_t', {
  usr: 'void *',
  chid: chanId,
  type: 'long',
  count: 'long',
  dbr: 'void *',
  status: 'int'
});
const MonitorCallback = koffi.callback('MonitorCallback', 'void', [event_args_t]);

const GetCallback = koffi.callback('GetCallback', 'void', [event_args_t]);
const PutCallback = koffi.callback('PutCallback', 'void', [event_args_t]);

const connection_args_t = koffi.struct('connection_args_t', {
    chid: chanId,
    op: 'long'
  });
const ConnectionCallback = koffi.callback('ConnectionCallback', 'void', [connection_args_t]);

const ca_context_create = libca.func('ca_context_create', 'int', ['int']);
const ca_message = libca.func('ca_message', 'string', ['int']);
const ca_client_status = libca.func('ca_client_status', 'int', ['int']);
const ca_current_context = libca.func('ca_current_context', 'int', []);
const ca_pend_event = libca.func('ca_pend_event', 'int', ['double']);
const ca_pend_io = libca.func('ca_pend_io', 'int', ['double']);
const ca_flush_io = libca.func('ca_flush_io', 'int', []);
const ca_test_io = libca.func('ca_test_io', 'int', []);
const ca_create_channel = libca.func('ca_create_channel', 'int', ['string',koffi.pointer(Conne
```

Implementation

- **Create and connect channel**
  - Get or Put: create() and then connect()
  - Monitor: register callback and create(), then emit “monitor” event in the callback



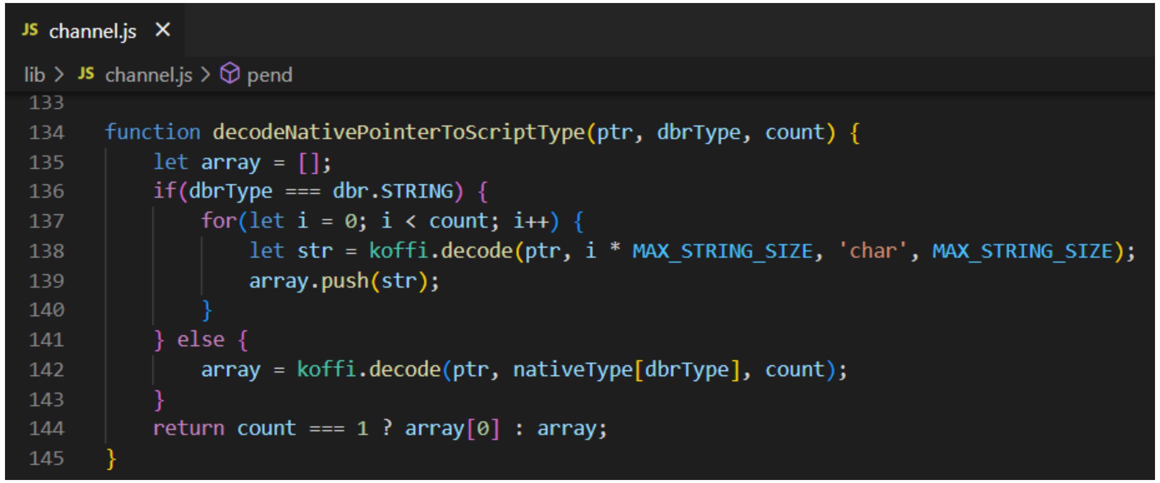
```javascript
    create(monitor = true) {
        let chidPtr = [null];
        let priority = 0;

        this.connectionStateChangePtr = koffi.register(args => {
            this.fieldType = ca_field_type(this.chanId);
            this.count = ca_element_count(this.chanId);
            this.emit("connection", args);
            // Ready to monitor
            if (args && args.op === state.OP_CONN_UP && !this.monitorEventId) {
                this.emit("monitor", args);
            }
        }, koffi.pointer(ConnectionCallback));

        let callback = monitor ? this.connectionStateChangePtr : null;
        let errCode = ca_create_channel(this.pvName, callback, null, priority, chidPtr);
        this.chanId = chidPtr[0];
        return errCode;
    }
```



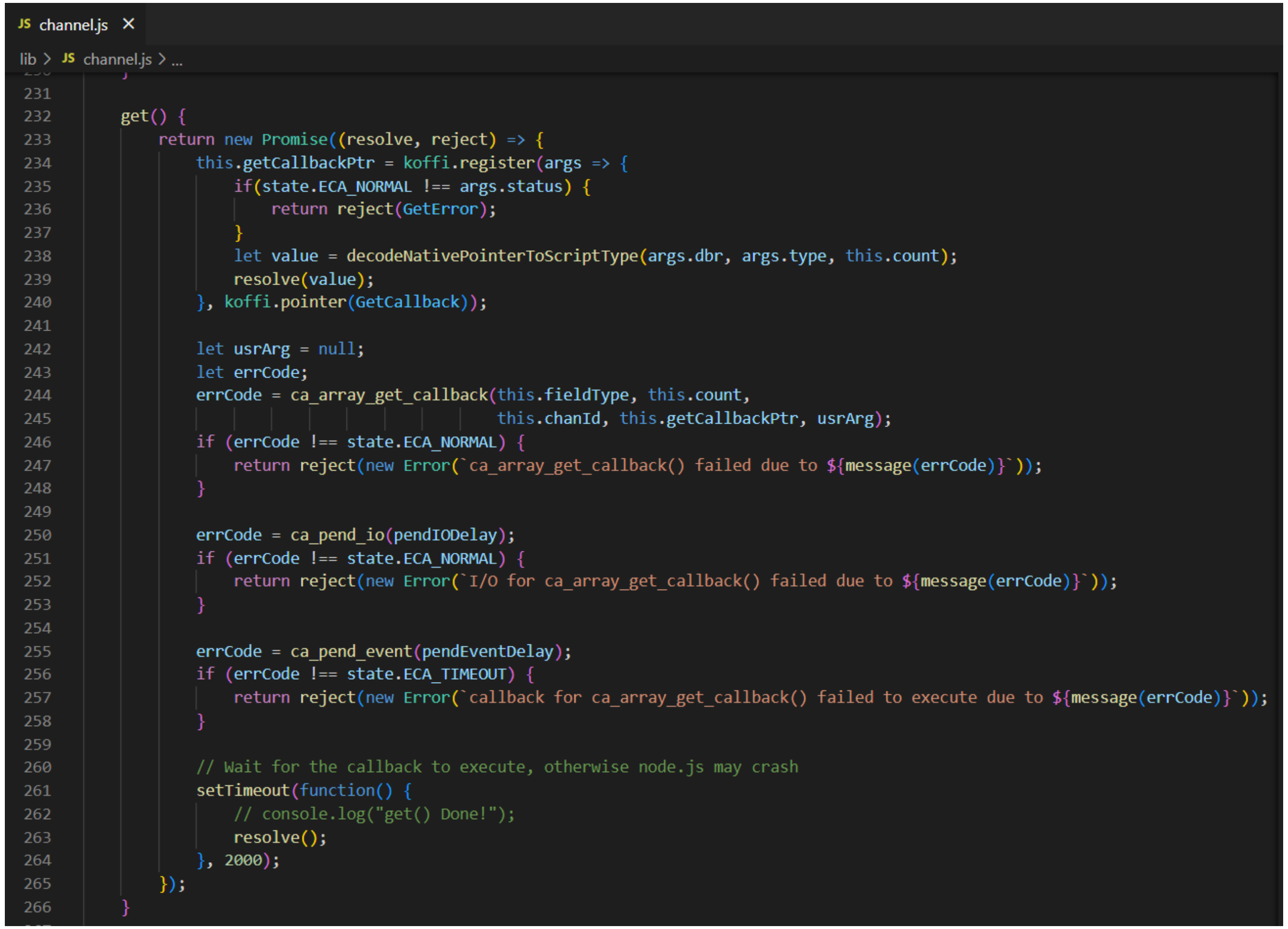
```javascript
    connect() {
        return new Promise((resolve, reject) => {
            let errCode;
            errCode = this.create(false);
            if (errCode !== state.ECA_NORMAL) {
                return reject(new Error(`ca_create_channel() failed due to ${message(errCode)}`));
            }
            errCode = ca_pend_io(pendIODelay);
            this.fieldType = ca_field_type(this.chanId);
            this.count = ca_element_count(this.chanId);
            if (errCode === state.ECA_NORMAL) {
                resolve();
            } else {
                reject(ConError);
            }
        });
    }
```

Implementation

- **Get**
  - Request PV value and decode native pointer to JavaScript type in callback
  - Wait 2 seconds for callback to resolve first, otherwise node.js may crash
  - Node.js is asynchronous event-driven, the waiting does not affect JavaScript execution



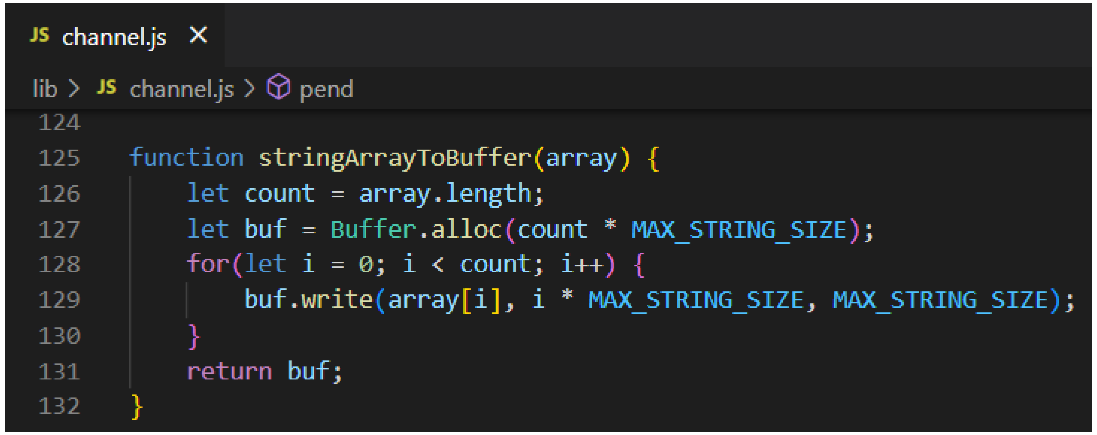
```
function decodeNativePointerToScriptType(ptr, dbrType, count) {
    let array = [];
    if(dbrType === dbr.STRING) {
        for(let i = 0; i < count; i++) {
            let str = koffi.decode(ptr, i * MAX_STRING_SIZE, 'char', MAX_STRING_SIZE);
            array.push(str);
        }
    } else {
        array = koffi.decode(ptr, nativeType[dbrType], count);
    }
    return count === 1 ? array[0] : array;
}
```



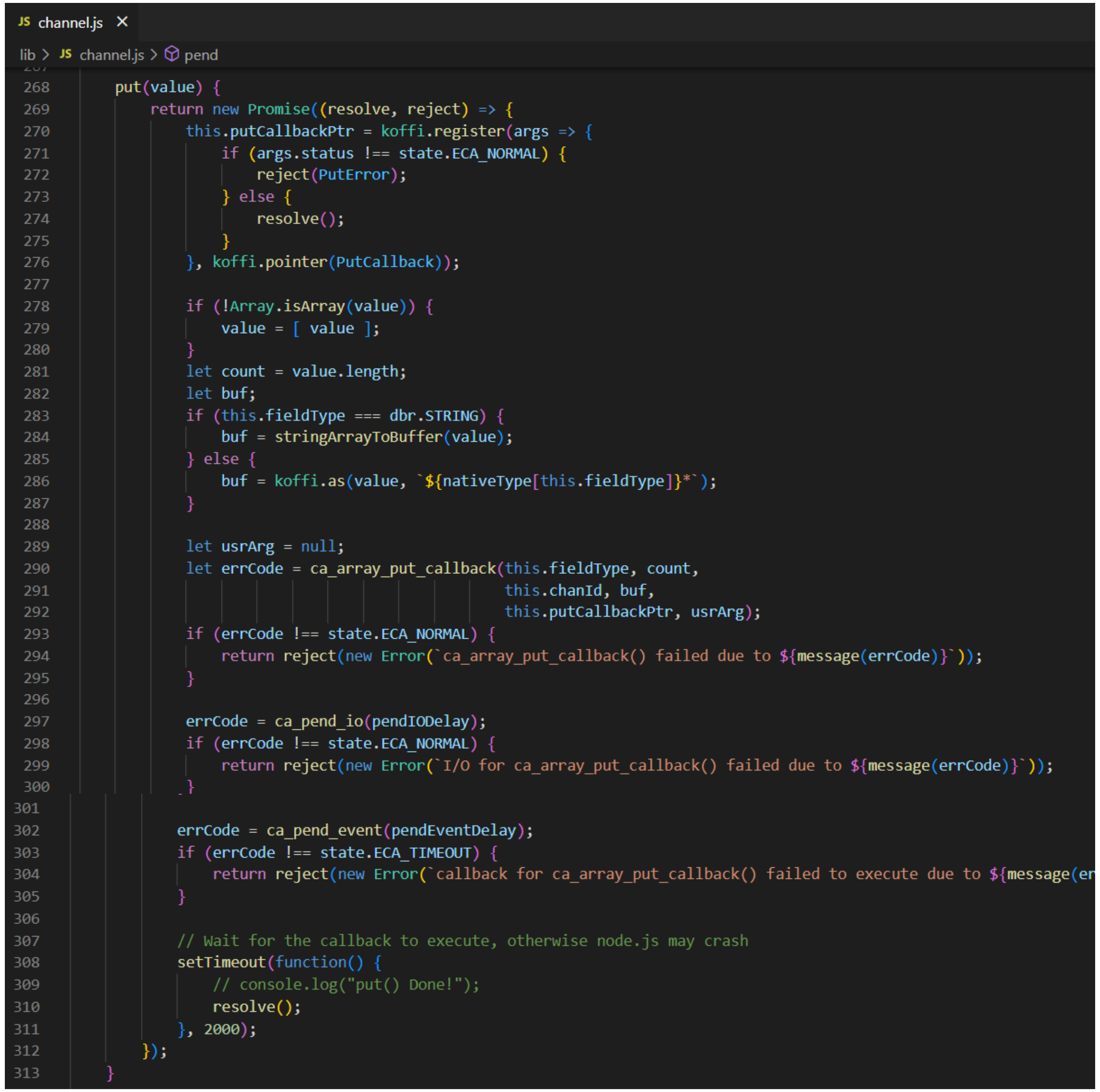
```
    get() {
        return new Promise((resolve, reject) => {
            this.getCallbackPtr = koffi.register(args => {
                if(state.ECA_NORMAL !== args.status) {
                    return reject(GetError);
                }
                let value = decodeNativePointerToScriptType(args.dbr, args.type, this.count);
                resolve(value);
            }, koffi.pointer(GetCallback));

            let usrArg = null;
            let errCode;
            errCode = ca_array_get_callback(this.fieldType, this.count,
                                            this.chanId, this.getCallbackPtr, usrArg);
            if (errCode !== state.ECA_NORMAL) {
                return reject(new Error(`ca_array_get_callback() failed due to ${message(errCode)}`));
            }

            errCode = ca_pend_io(pendIODelay);
            if (errCode !== state.ECA_NORMAL) {
                return reject(new Error(`I/O for ca_array_get_callback() failed due to ${message(errCode)}`));
            }

            errCode = ca_pend_event(pendEventDelay);
            if (errCode !== state.ECA_TIMEOUT) {
                return reject(new Error(`callback for ca_array_get_callback() failed to execute due to ${message(errCode)}`));
            }

            // Wait for the callback to execute, otherwise node.js may crash
            setTimeout(function() {
                // console.log("get() Done!");
                resolve();
            }, 2000);
        });
    }
```

Implementation

- **Put**
  - Convert JavaScript type to Buffer and put the Buffer value to PV
  - Wait 2 seconds for callback to resolve first, otherwise node.js may crash
  - Node.js is asynchronous event-driven, the waiting does not affect JavaScript execution



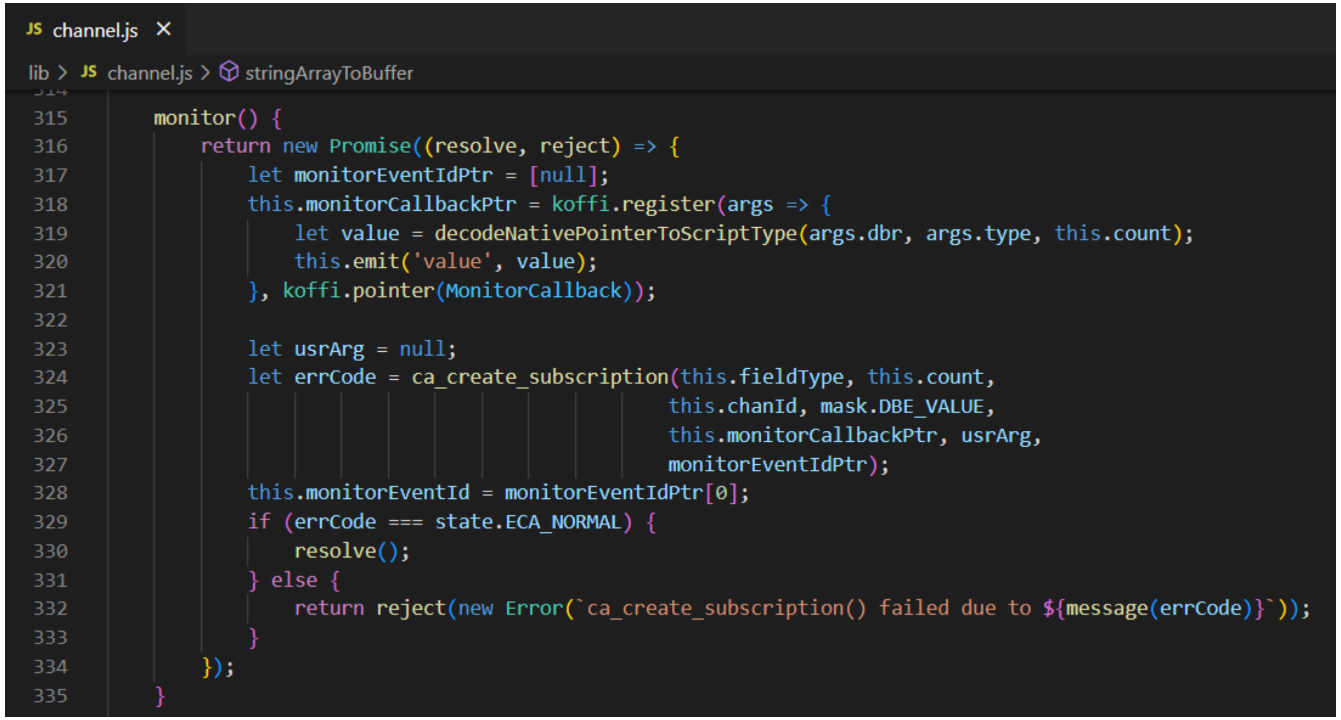
```javascript
function stringArrayToBuffer(array) {
    let count = array.length;
    let buf = Buffer.alloc(count * MAX_STRING_SIZE);
    for(let i = 0; i < count; i++) {
        buf.write(array[i], i * MAX_STRING_SIZE, MAX_STRING_SIZE);
    }
    return buf;
}
```



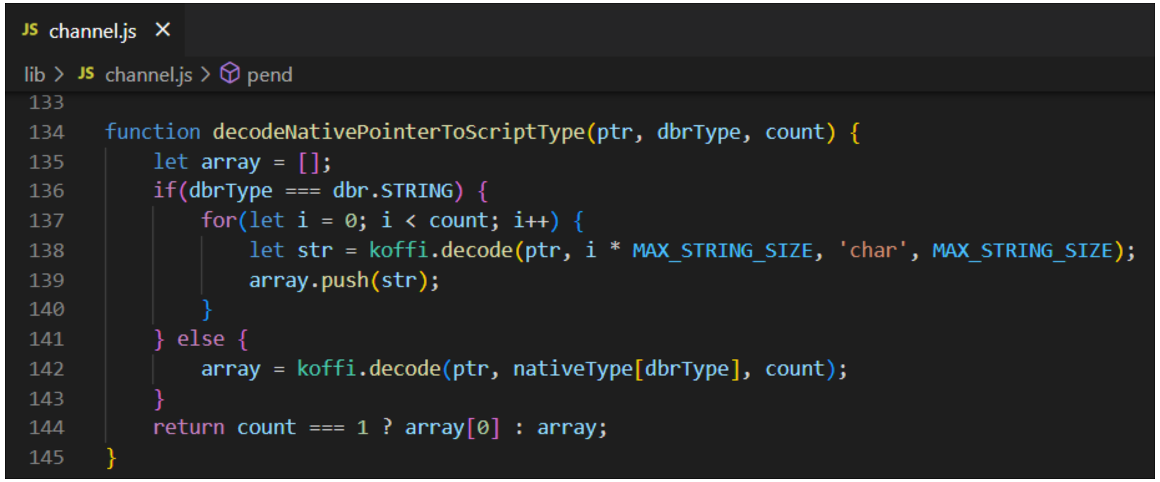
```javascript
    put(value) {
        return new Promise((resolve, reject) => {
            this.putCallbackPtr = koffi.register(args => {
                if (args.status !== state.ECA_NORMAL) {
                    reject(PutError);
                } else {
                    resolve();
                }
            }, koffi.pointer(PutCallback));

            if (!Array.isArray(value)) {
                value = [ value ];
            }
            let count = value.length;
            let buf;
            if (this.fieldType === dbr.STRING) {
                buf = stringArrayToBuffer(value);
            } else {
                buf = koffi.as(value, `${nativeType[this.fieldType]}*`);
            }

            let usrArg = null;
            let errCode = ca_array_put_callback(this.fieldType, count,
                                                this.chanId, buf,
                                                this.putCallbackPtr, usrArg);
            if (errCode !== state.ECA_NORMAL) {
                return reject(new Error(`ca_array_put_callback() failed due to ${message(errCode)}`));
            }

            errCode = ca_pend_io(pendIODelay);
            if (errCode !== state.ECA_NORMAL) {
                return reject(new Error(`I/O for ca_array_put_callback() failed due to ${message(errCode)}`));
            }

            errCode = ca_pend_event(pendEventDelay);
            if (errCode !== state.ECA_TIMEOUT) {
                return reject(new Error(`callback for ca_array_put_callback() failed to execute due to ${message(er
            }

            // Wait for the callback to execute, otherwise node.js may crash
            setTimeout(function() {
                // console.log("put() Done!");
                resolve();
            }, 2000);
        });
    }
```

Implementation

- **Monitor**
  - Subscribe monitor of a PV with callback
  - Decode native pointer to JavaScript type in callback
  - Emit “value” event in callback



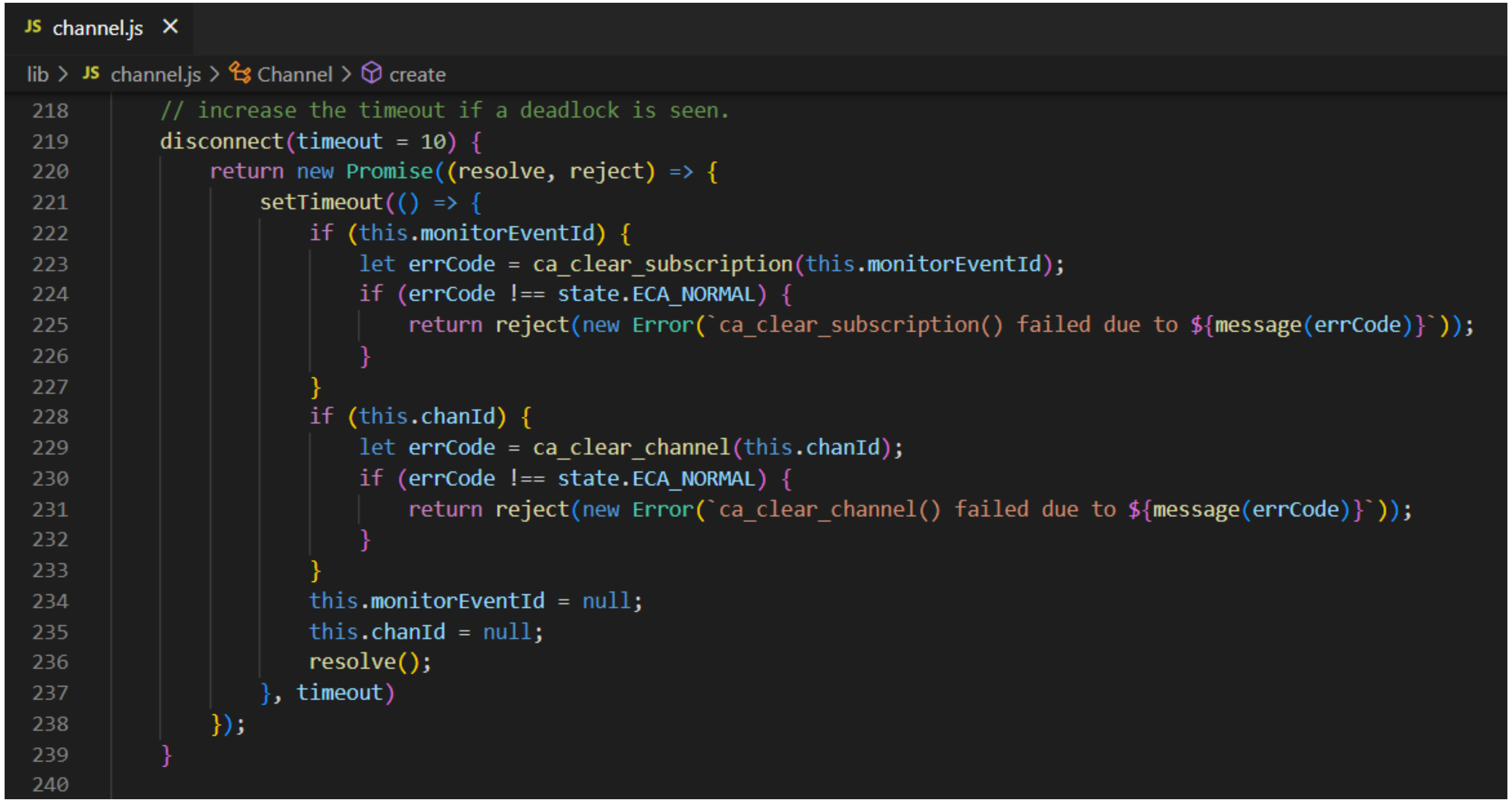
```
    monitor() {
        return new Promise((resolve, reject) => {
            let monitorEventIdPtr = [null];
            this.monitorCallbackPtr = koffi.register(args => {
                let value = decodeNativePointerToScriptType(args.dbr, args.type, this.count);
                this.emit('value', value);
            }, koffi.pointer(MonitorCallback));

            let usrArg = null;
            let errCode = ca_create_subscription(this.fieldType, this.count,
                                                 this.chanId, mask.DBE_VALUE,
                                                 this.monitorCallbackPtr, usrArg,
                                                 monitorEventIdPtr);
            this.monitorEventId = monitorEventIdPtr[0];
            if (errCode === state.ECA_NORMAL) {
                resolve();
            } else {
                return reject(new Error(`ca_create_subscription() failed due to ${message(errCode)}`));
            }
        });
    }
```



```

function decodeNativePointerToScriptType(ptr, dbrType, count) {
    let array = [];
    if(dbrType === dbr.STRING) {
        for(let i = 0; i < count; i++) {
            let str = koffi.decode(ptr, i * MAX_STRING_SIZE, 'char', MAX_STRING_SIZE);
            array.push(str);
        }
    } else {
        array = koffi.decode(ptr, nativeType[dbrType], count);
    }
    return count === 1 ? array[0] : array;
}
```

Implementation

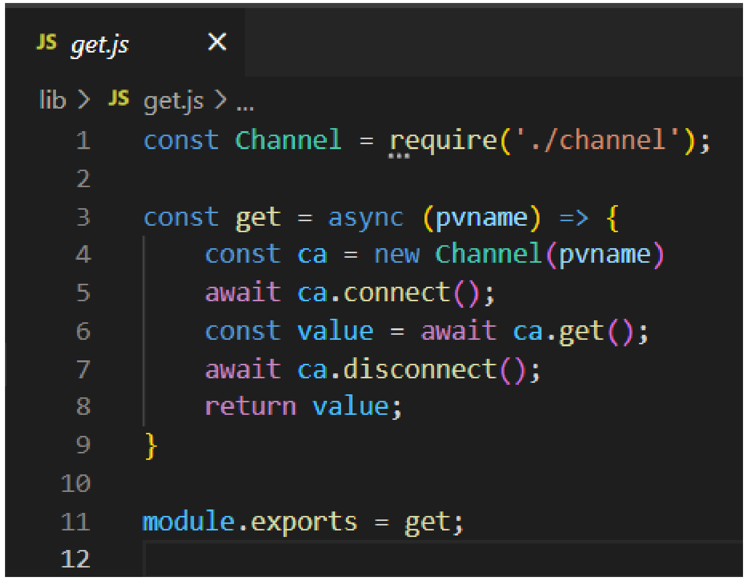
- **Disconnect**
  - Clear subscription
  - Clear channel



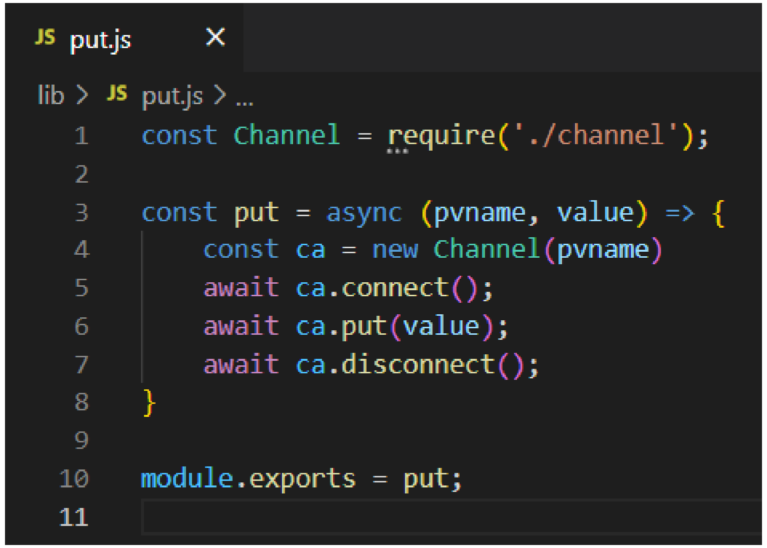
```
    // increase the timeout if a deadlock is seen.
    disconnect(timeout = 10) {
        return new Promise((resolve, reject) => {
            setTimeout(() => {
                if (this.monitorEventId) {
                    let errCode = ca_clear_subscription(this.monitorEventId);
                    if (errCode !== state.ECA_NORMAL) {
                        return reject(new Error(`ca_clear_subscription() failed due to ${message(errCode)}`));
                    }
                }
                if (this.chanId) {
                    let errCode = ca_clear_channel(this.chanId);
                    if (errCode !== state.ECA_NORMAL) {
                        return reject(new Error(`ca_clear_channel() failed due to ${message(errCode)}`));
                    }
                }
                this.monitorEventId = null;
                this.chanId = null;
                resolve();
            }, timeout)
        });
    }

```

Implementation

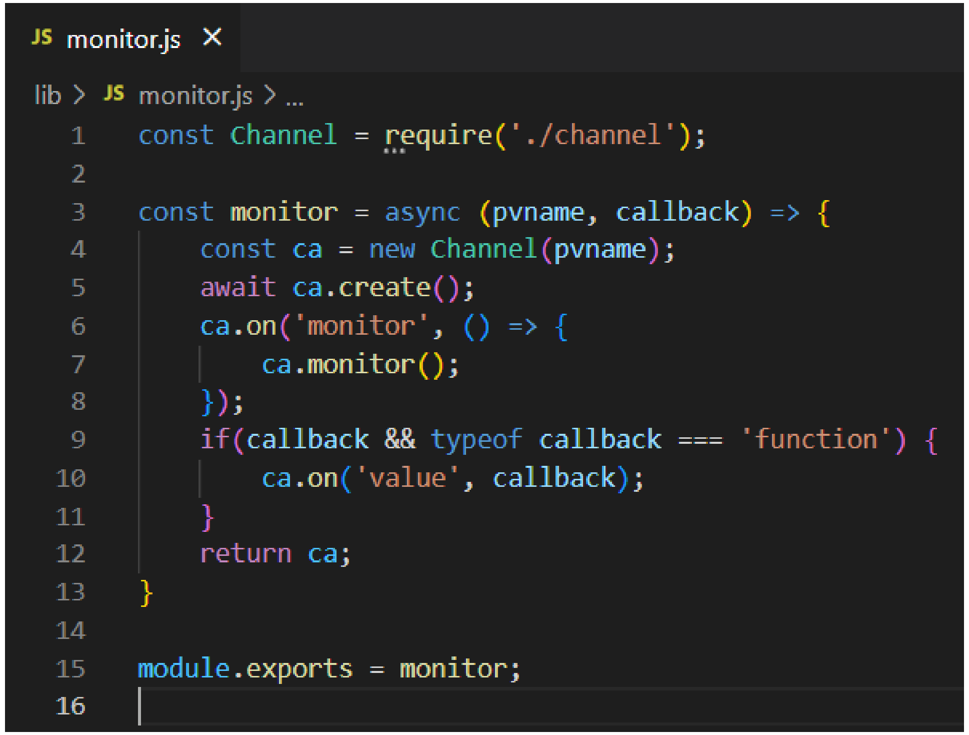
- **Provide a user-friendly API like caget, caput and camonitor.**
- **Wrap get(), put() and monitor() methods of Channel.**



```javascript
const Channel = require('./channel');

const get = async (pvname) => {
    const ca = new Channel(pvname)
    await ca.connect();
    const value = await ca.get();
    await ca.disconnect();
    return value;
}

module.exports = get;
```



```javascript
const Channel = require('./channel');

const put = async (pvname, value) => {
    const ca = new Channel(pvname)
    await ca.connect();
    await ca.put(value);
    await ca.disconnect();
}

module.exports = put;
```

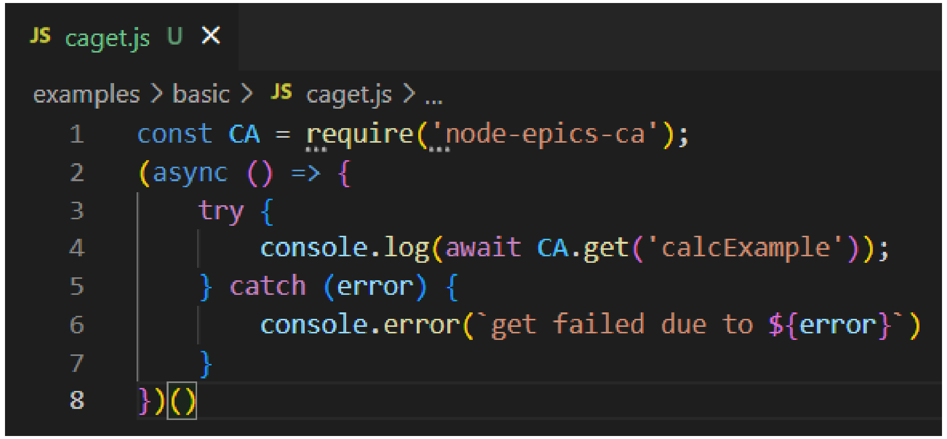


```javascript
const Channel = require('./channel');

const monitor = async (pvname, callback) => {
    const ca = new Channel(pvname);
    await ca.create();
    ca.on('monitor', () => {
        ca.monitor();
    });
    if(callback && typeof callback === 'function') {
        ca.on('value', callback);
    }
    return ca;
}

module.exports = monitor;
```

Usage

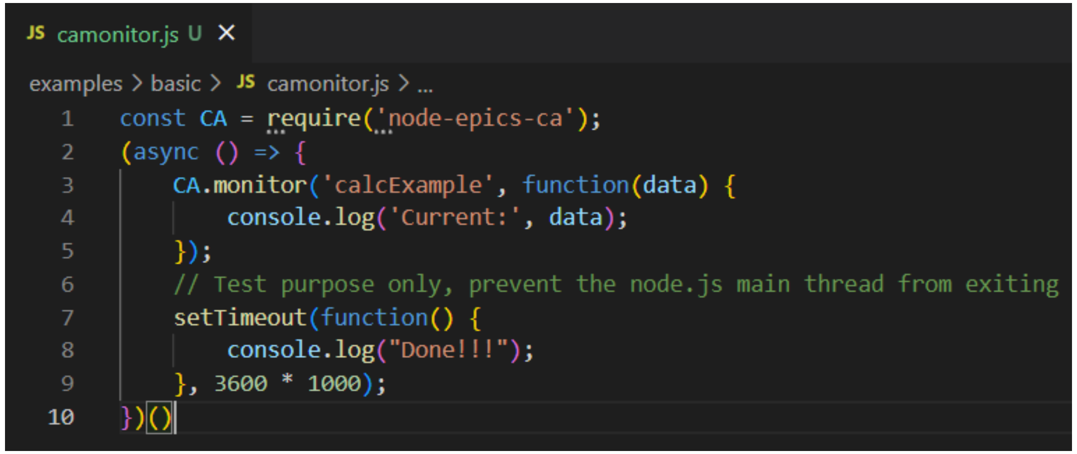
- Installation
  - npm install node-epics-ca
- Environment variable
  - NODE_EPICS_LIBCA
  - NODE_EPICS_CA_PEND_IO_DELAY
  - NODE_EPICS_CA_PEND_EVENT_DELAY
- Usage example directory
  - https://github.com/wanglin86769/node-epics-ca/tree/master/examples

Usage example #1: basic



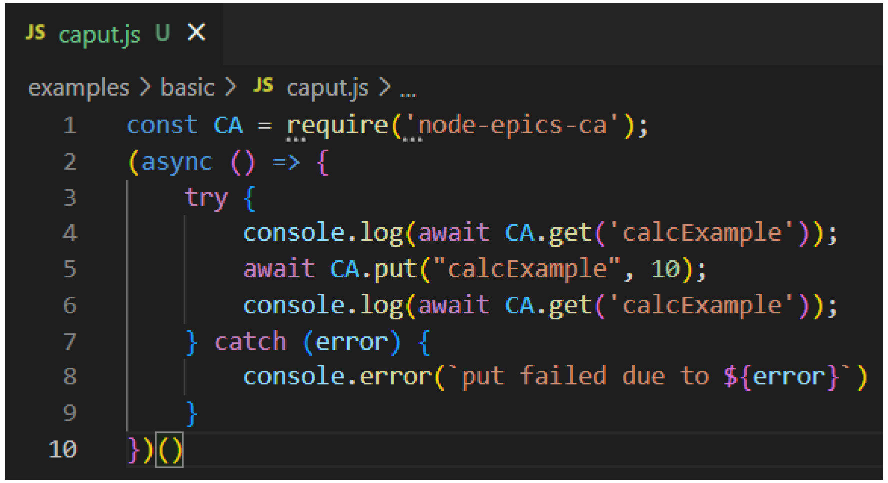
```javascript
const CA = require('node-epics-ca');
(async () => {
    try {
        console.log(await CA.get('calcExample'));
    } catch (error) {
        console.error(`get failed due to ${error}`)
    }
})()
```

node-epics-ca/examples/basic/caget.js



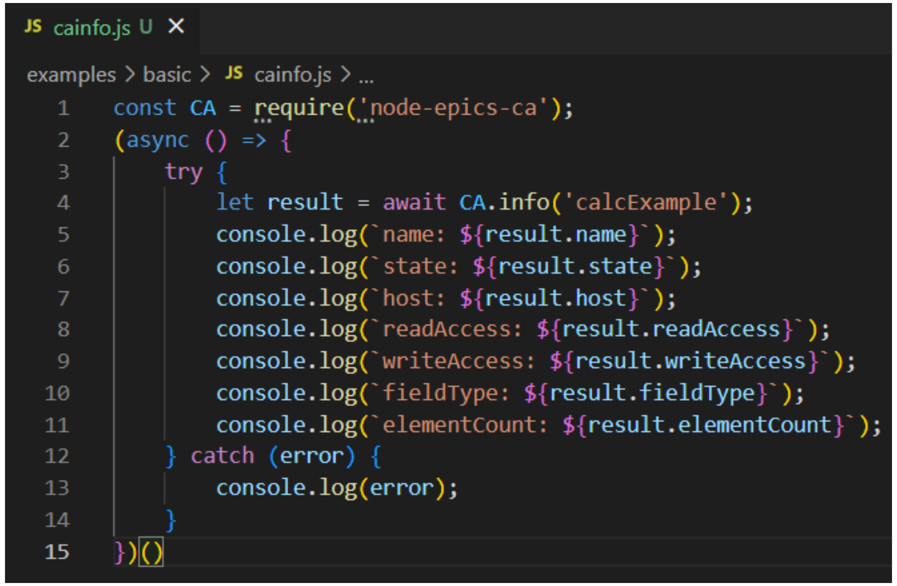
```javascript
const CA = require('node-epics-ca');
(async () => {
    CA.monitor('calcExample', function(data) {
        console.log('Current:', data);
    });
    // Test purpose only, prevent the node.js main thread from exiting
    setTimeout(function() {
        console.log("Done!!!");
    }, 3600 * 1000);
})()
```

node-epics-ca/examples/basic/camonitor.js



```javascript
const CA = require('node-epics-ca');
(async () => {
    try {
        console.log(await CA.get('calcExample'));
        await CA.put("calcExample", 10);
        console.log(await CA.get('calcExample'));
    } catch (error) {
        console.error(`put failed due to ${error}`)
    }
})()
```

node-epics-ca/examples/basic/caput.js

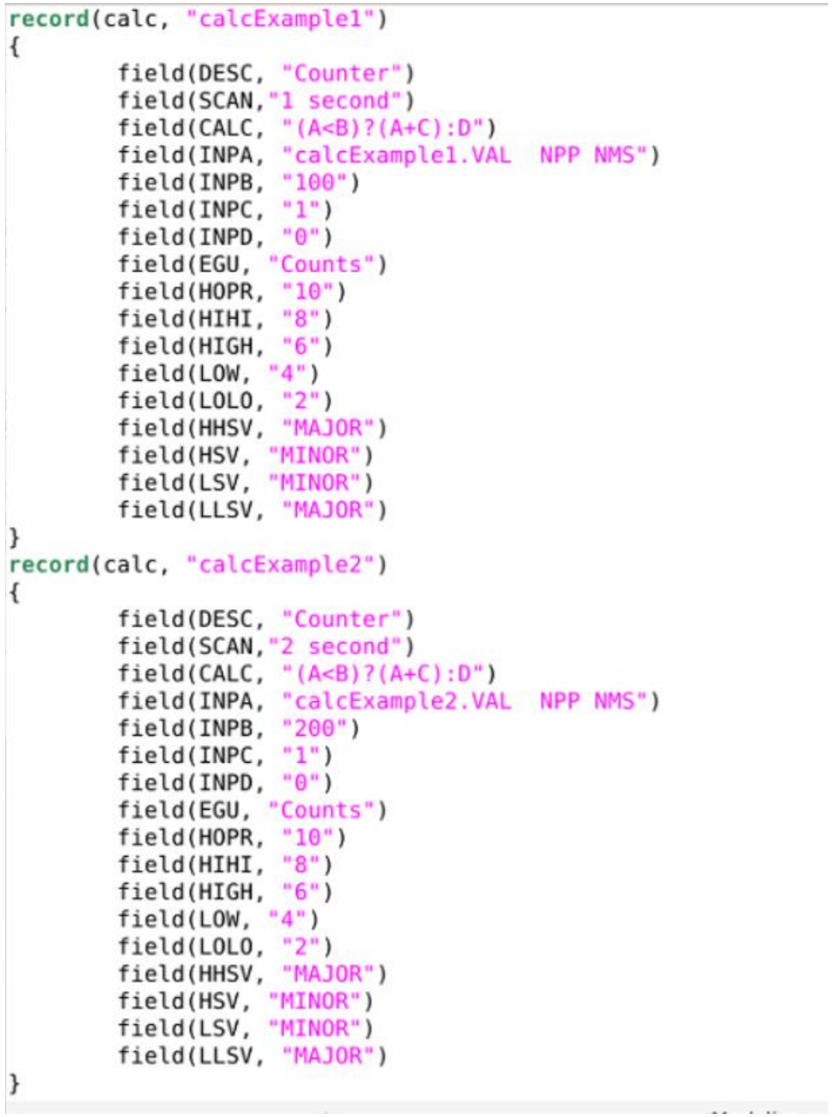


```javascript
const CA = require('node-epics-ca');
(async () => {
    try {
        let result = await CA.info('calcExample');
        console.log(`name: ${result.name}`);
        console.log(`state: ${result.state}`);
        console.log(`host: ${result.host}`);
        console.log(`readAccess: ${result.readAccess}`);
        console.log(`writeAccess: ${result.writeAccess}`);
        console.log(`fieldType: ${result.fieldType}`);
        console.log(`elementCount: ${result.elementCount}`);
    } catch (error) {
        console.log(error);
    }
})()
```

node-epics-ca/examples/basic/cainfo.js

Usage example #2 : service

- **Implement a web service in Node.js**
  - Get PV value
  - Put PV value
  - Monitor PV value
  - node-epics-ca/examples/service/server.js
- **Packages**
  - express: web framework for Node.js
  - ws: a Node.js WebSocket library
  - node-epics-ca: EPICS Channel Access client for Node.js
- **API example**
  - `GET     http://localhost:3000?pv=calcExample1`
  - `PUT     http://localhost:3000?pv=calcExample1&value=5`
  - `WebSocket    ws://localhost:3001`

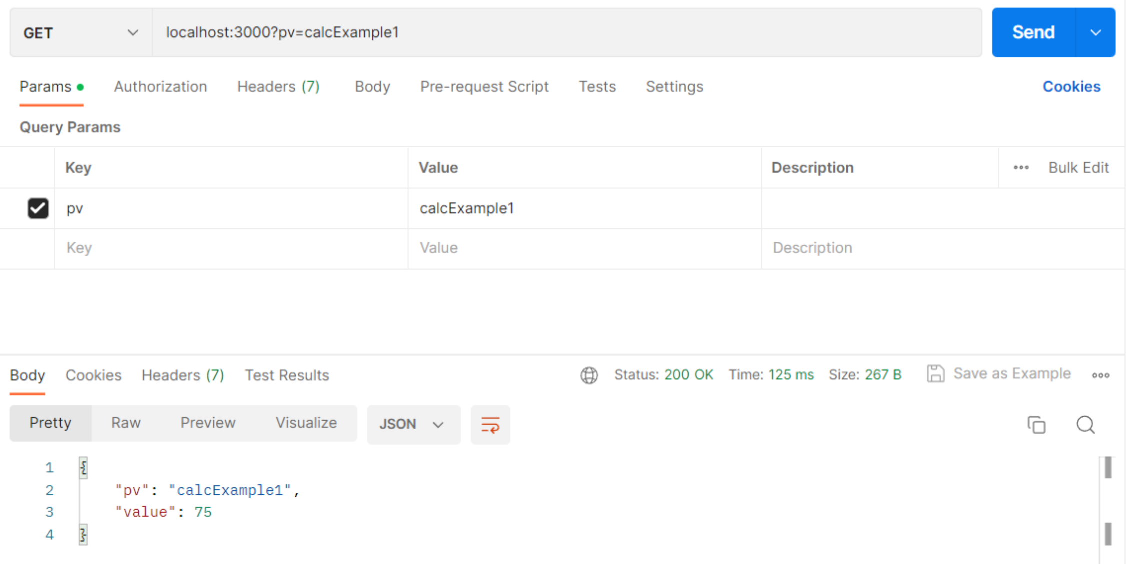


```
record(calc, "calcExample1")
{
        field(DESC, "Counter")
        field(SCAN,"1 second")
        field(CALC, "(A<B)?(A+C):D")
        field(INPA, "calcExample1.VAL  NPP NMS")
        field(INPB, "100")
        field(INPC, "1")
        field(INPD, "0")
        field(EGU, "Counts")
        field(HOPR, "10")
        field(HIHI, "8")
        field(HIGH, "6")
        field(LOW, "4")
        field(LOLO, "2")
        field(HHSV, "MAJOR")
        field(HSV, "MINOR")
        field(LSV, "MINOR")
        field(LLSV, "MAJOR")
}
record(calc, "calcExample2")
{
        field(DESC, "Counter")
        field(SCAN,"2 second")
        field(CALC, "(A<B)?(A+C):D")
        field(INPA, "calcExample2.VAL  NPP NMS")
        field(INPB, "200")
        field(INPC, "1")
        field(INPD, "0")
        field(EGU, "Counts")
        field(HOPR, "10")
        field(HIHI, "8")
        field(HIGH, "6")
        field(LOW, "4")
        field(LOLO, "2")
        field(HHSV, "MAJOR")
        field(HSV, "MINOR")
        field(LSV, "MINOR")
        field(LLSV, "MAJOR")
}
```

PVs for test

Usage example #2: service

- **Get PV value**



GET localhost:3000?pv=calcExample1

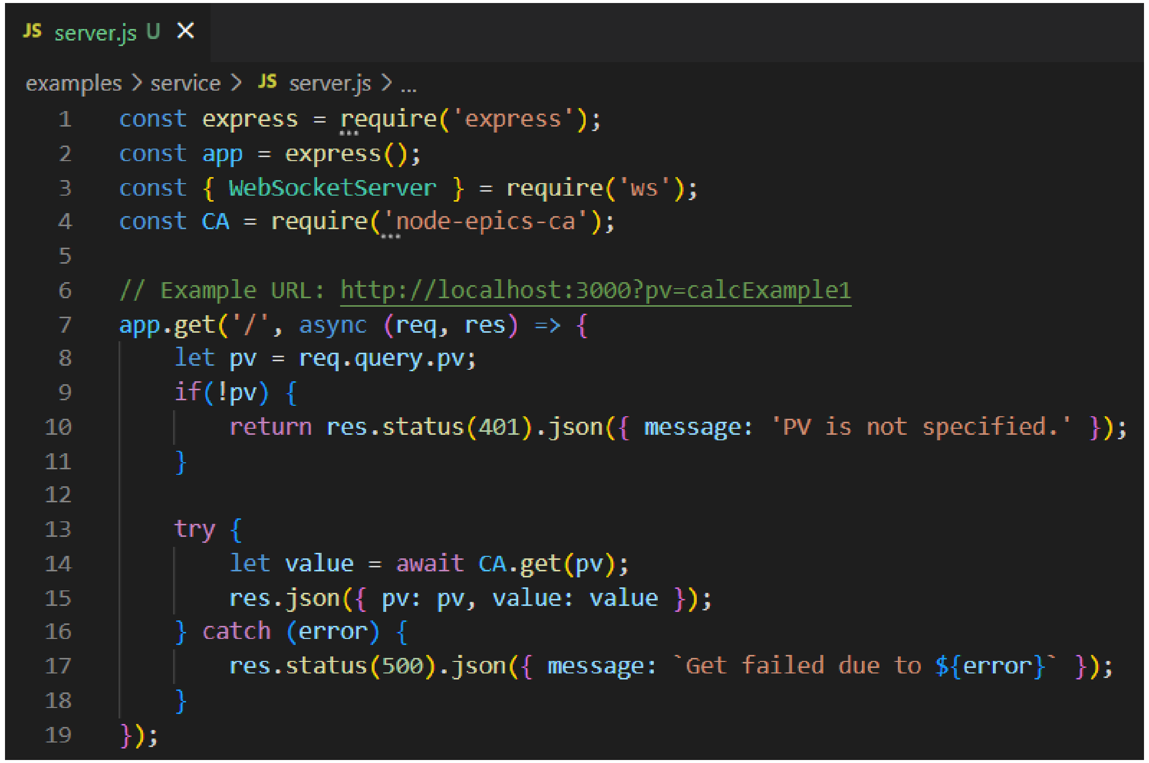
Query Params

	Key	Value	Description
☑	pv	calcExample1	
	Key	Value	Description

Status: 200 OK Time: 125 ms Size: 267 B

```
{
    "pv": "calcExample1",
    "value": 75
}
```

Get PV value using Postman

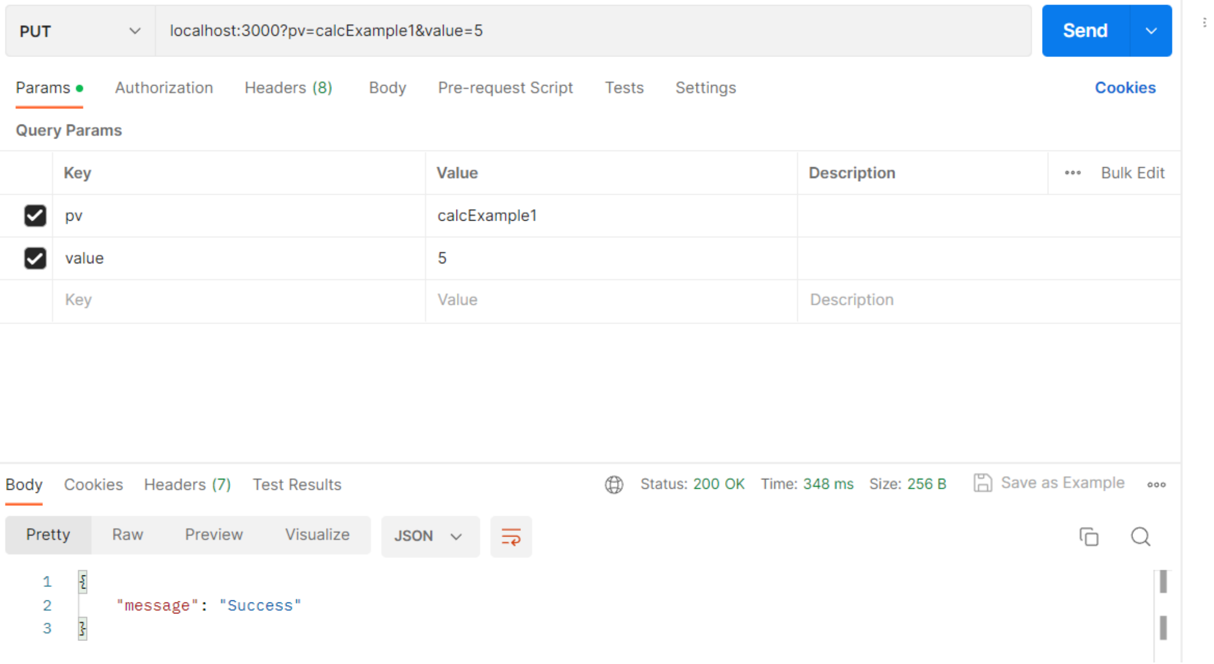


```
const express = require('express');
const app = express();
const { WebSocketServer } = require('ws');
const CA = require('node-epics-ca');

// Example URL: http://localhost:3000?pv=calcExample1
app.get('/', async (req, res) => {
    let pv = req.query.pv;
    if(!pv) {
        return res.status(401).json({ message: 'PV is not specified.' });
    }

    try {
        let value = await CA.get(pv);
        res.json({ pv: pv, value: value });
    } catch (error) {
        res.status(500).json({ message: `Get failed due to ${error}` });
    }
});
```

HTTP interface for get

Usage example #2 : service

- **Put PV value**



PUT localhost:3000?pv=calcExample1&value=5 Send

Params ● | Authorization | Headers (8) | Body | Pre-request Script | Tests | Settings | Cookies

Query Params

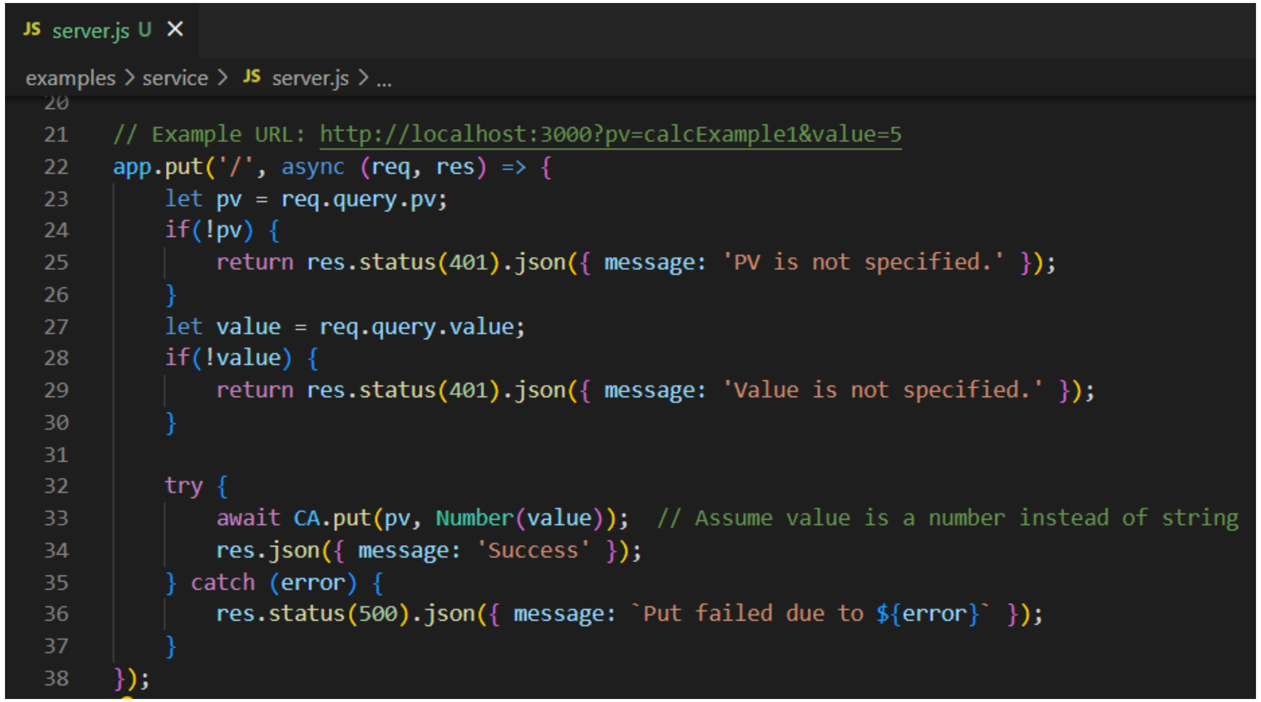
	Key	Value	Description
☑	pv	calcExample1	
☑	value	5	
	Key	Value	Description

Bulk Edit
Body | Cookies | Headers (7) | Test Results
Save as Example
Pretty | Raw | Preview | Visualize | JSON

Status: 200 OK Time: 348 ms Size: 256 B

```
{
    "message": "Success"
}
```

Put PV value using Postman

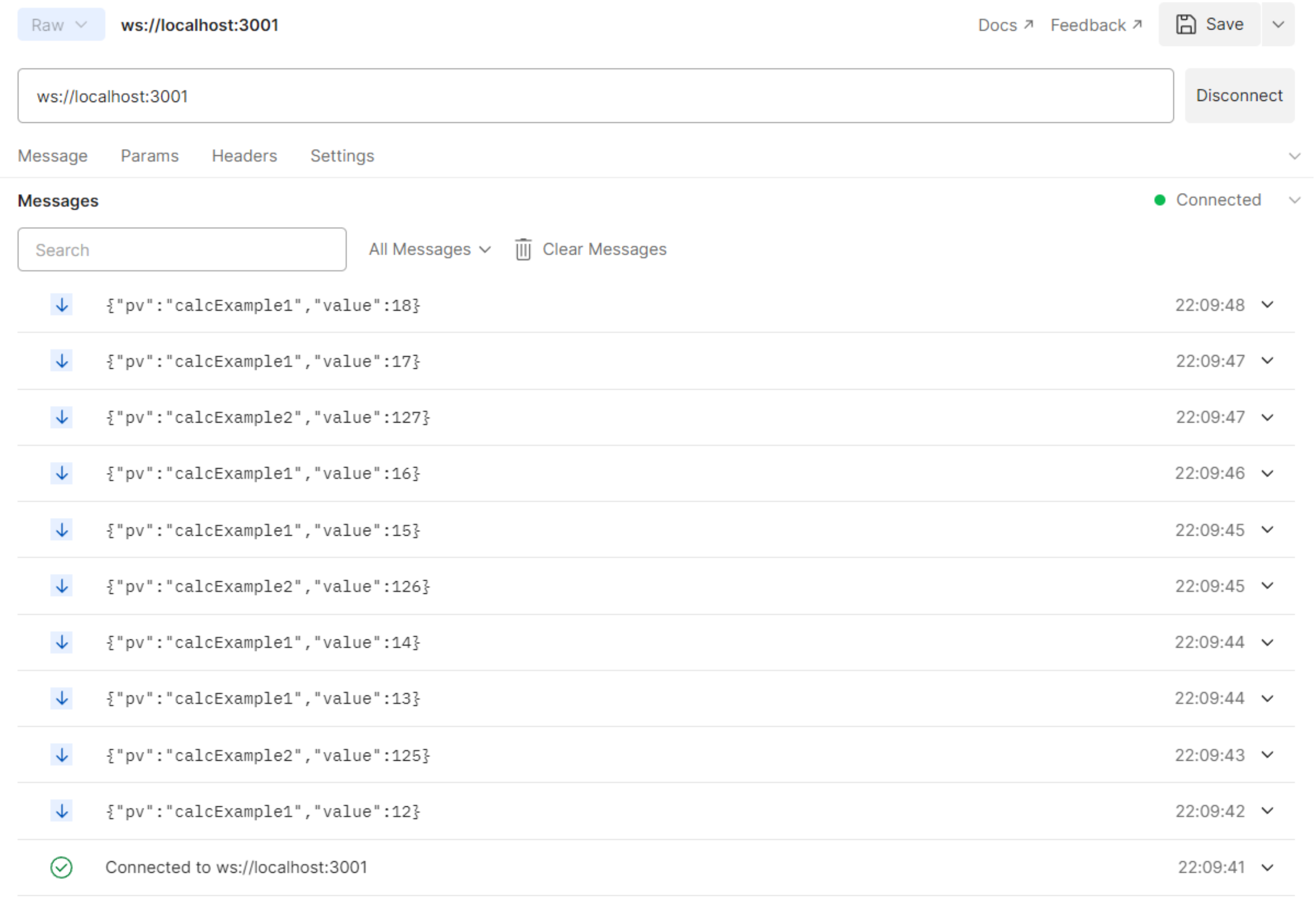


```javascript
// Example URL: http://localhost:3000?pv=calcExample1&value=5
app.put('/', async (req, res) => {
    let pv = req.query.pv;
    if(!pv) {
        return res.status(401).json({ message: 'PV is not specified.' });
    }
    let value = req.query.value;
    if(!value) {
        return res.status(401).json({ message: 'Value is not specified.' });
    }

    try {
        await CA.put(pv, Number(value));  // Assume value is a number instead of string
        res.json({ message: 'Success' });
    } catch (error) {
        res.status(500).json({ message: `Put failed due to ${error}` });
    }
});
```

HTTP interface for put

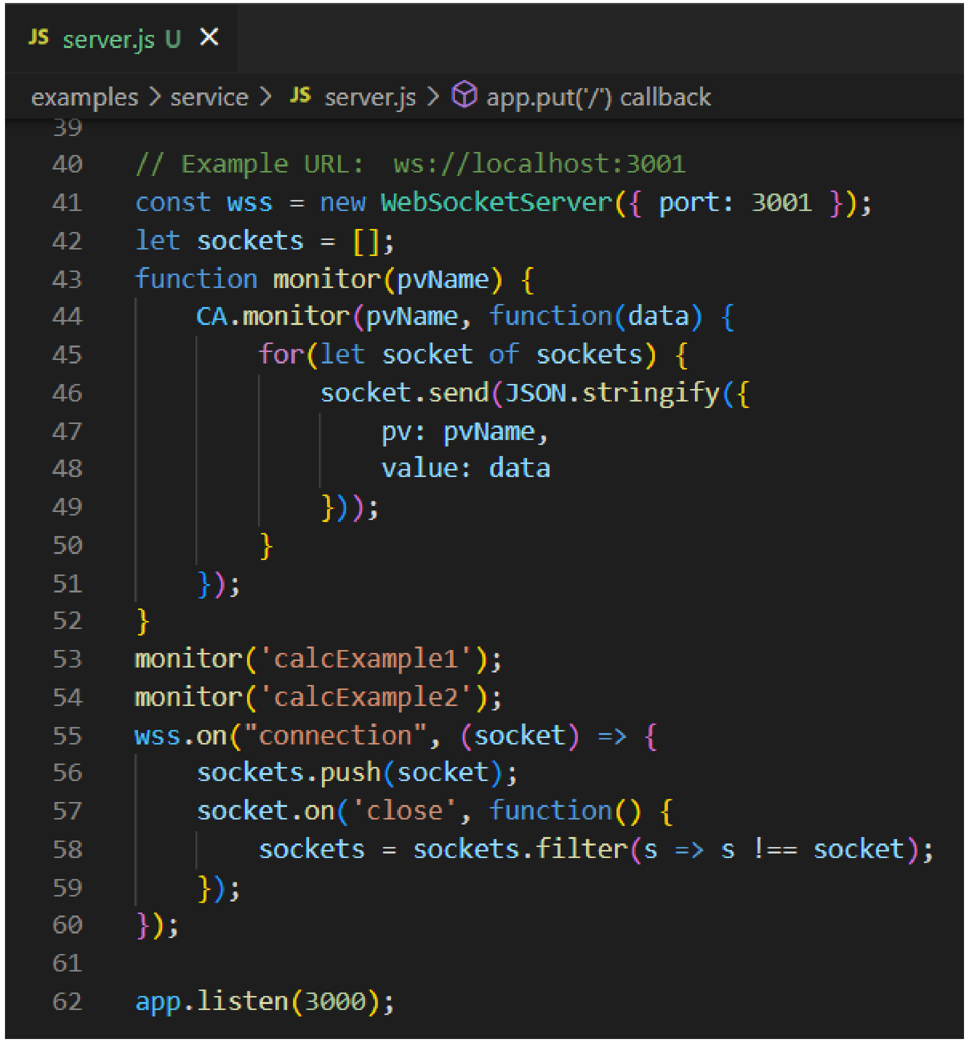
Usage example #2 : service

- **Monitor PV value**



```
{"pv":"calcExample1","value":18}    22:09:48
{"pv":"calcExample1","value":17}    22:09:47
{"pv":"calcExample2","value":127}   22:09:47
{"pv":"calcExample1","value":16}    22:09:46
{"pv":"calcExample1","value":15}    22:09:45
{"pv":"calcExample2","value":126}   22:09:45
{"pv":"calcExample1","value":14}    22:09:44
{"pv":"calcExample1","value":13}    22:09:44
{"pv":"calcExample2","value":125}   22:09:43
{"pv":"calcExample1","value":12}    22:09:42
Connected to ws://localhost:3001    22:09:41
```

Monitor PV value using Postman

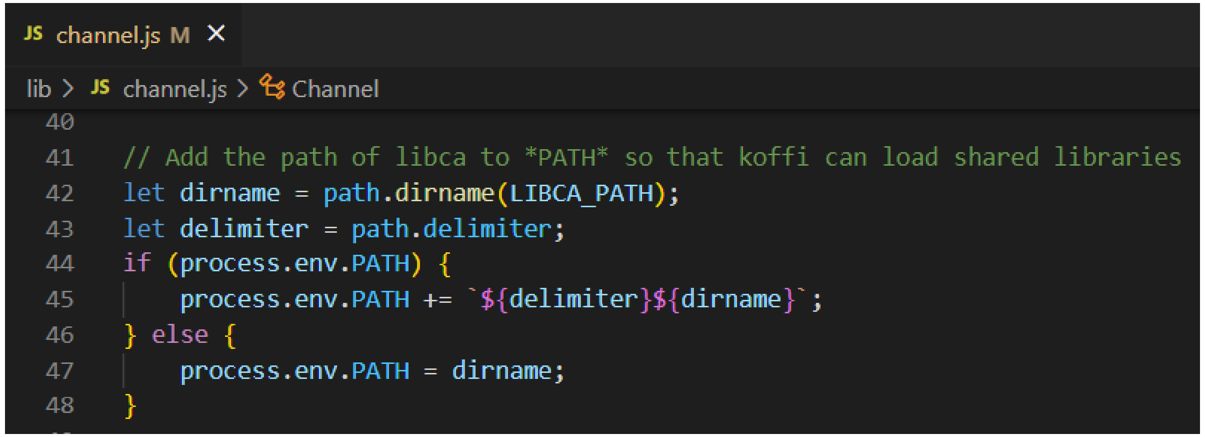


```javascript
// Example URL:  ws://localhost:3001
const wss = new WebSocketServer({ port: 3001 });
let sockets = [];
function monitor(pvName) {
    CA.monitor(pvName, function(data) {
        for(let socket of sockets) {
            socket.send(JSON.stringify({
                pv: pvName,
                value: data
            }));
        }
    });
}
monitor('calcExample1');
monitor('calcExample2');
wss.on("connection", (socket) => {
    sockets.push(socket);
    socket.on('close', function() {
        sockets = sockets.filter(s => s !== socket);
    });
});

app.listen(3000);
```

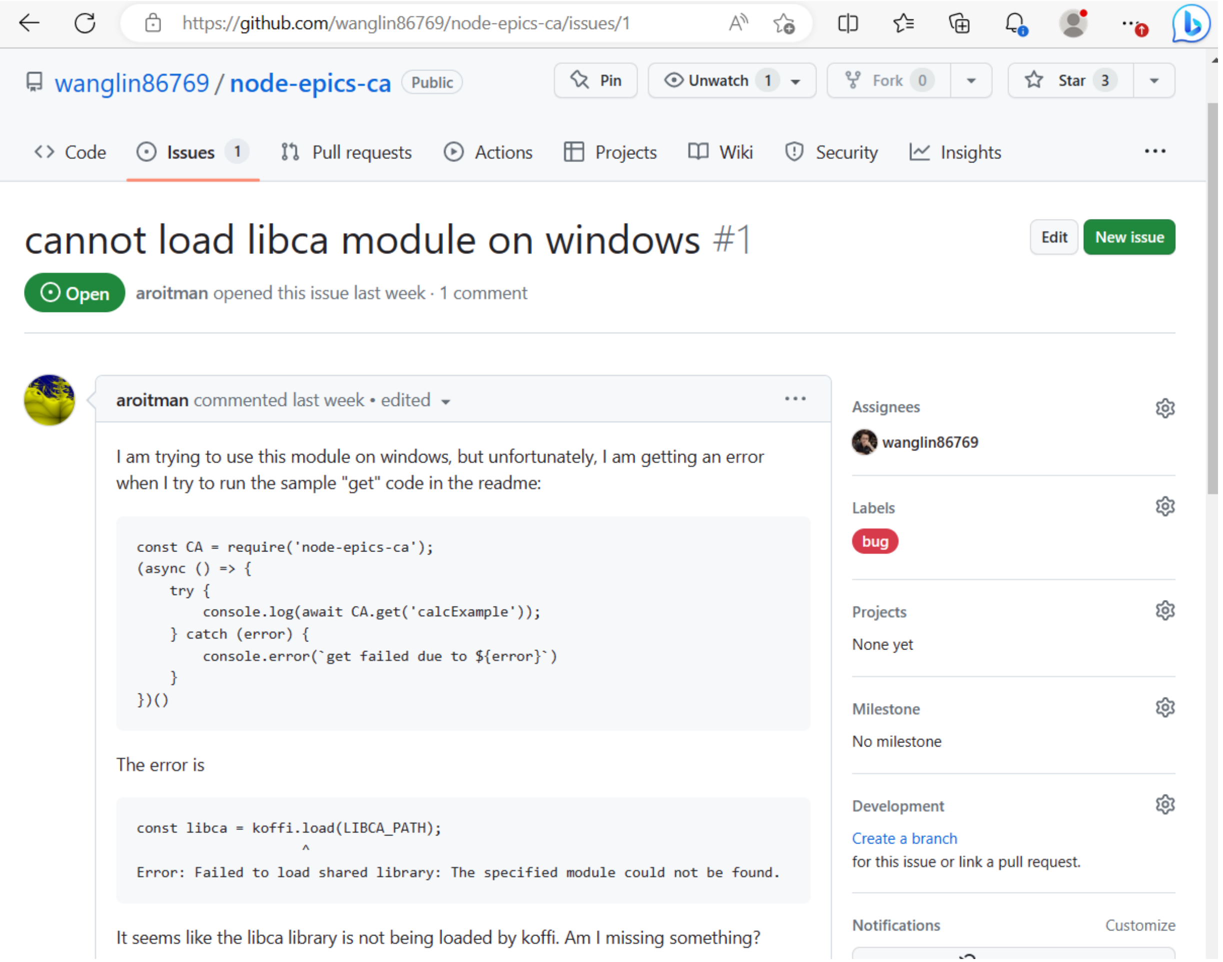
HTTP interface for monitor

Potential problem #1: Cannot find shared library

- Received a report that libCom cannot be loaded on Windows, since libca depends on libCom.
- Added path of libCom to %PATH% environment variable to fix the issue.
- Did not add the path to LD_LIBRARY_PATH or DYLD_LIBRARY_PATH, not sure whether the issue will happen on Linux or MacOS.



```
// Add the path of libca to *PATH* so that koffi can load shared libraries
let dirname = path.dirname(LIBCA_PATH);
let delimiter = path.delimiter;
if (process.env.PATH) {
    process.env.PATH += `${delimiter}${dirname}`;
} else {
    process.env.PATH = dirname;
}
```



cannot load libca module on windows #1

Open aroitman opened this issue last week · 1 comment

aroitman commented last week • edited

I am trying to use this module on windows, but unfortunately, I am getting an error when I try to run the sample "get" code in the readme:

```
const CA = require('node-epics-ca');
(async () => {
    try {
        console.log(await CA.get('calcExample'));
    } catch (error) {
        console.error(`get failed due to ${error}`)
    }
})()
```

The error is

```
const libca = koffi.load(LIBCA_PATH);
                   ^
Error: Failed to load shared library: The specified module could not be found.
```

It seems like the libca library is not being loaded by koffi. Am I missing something?

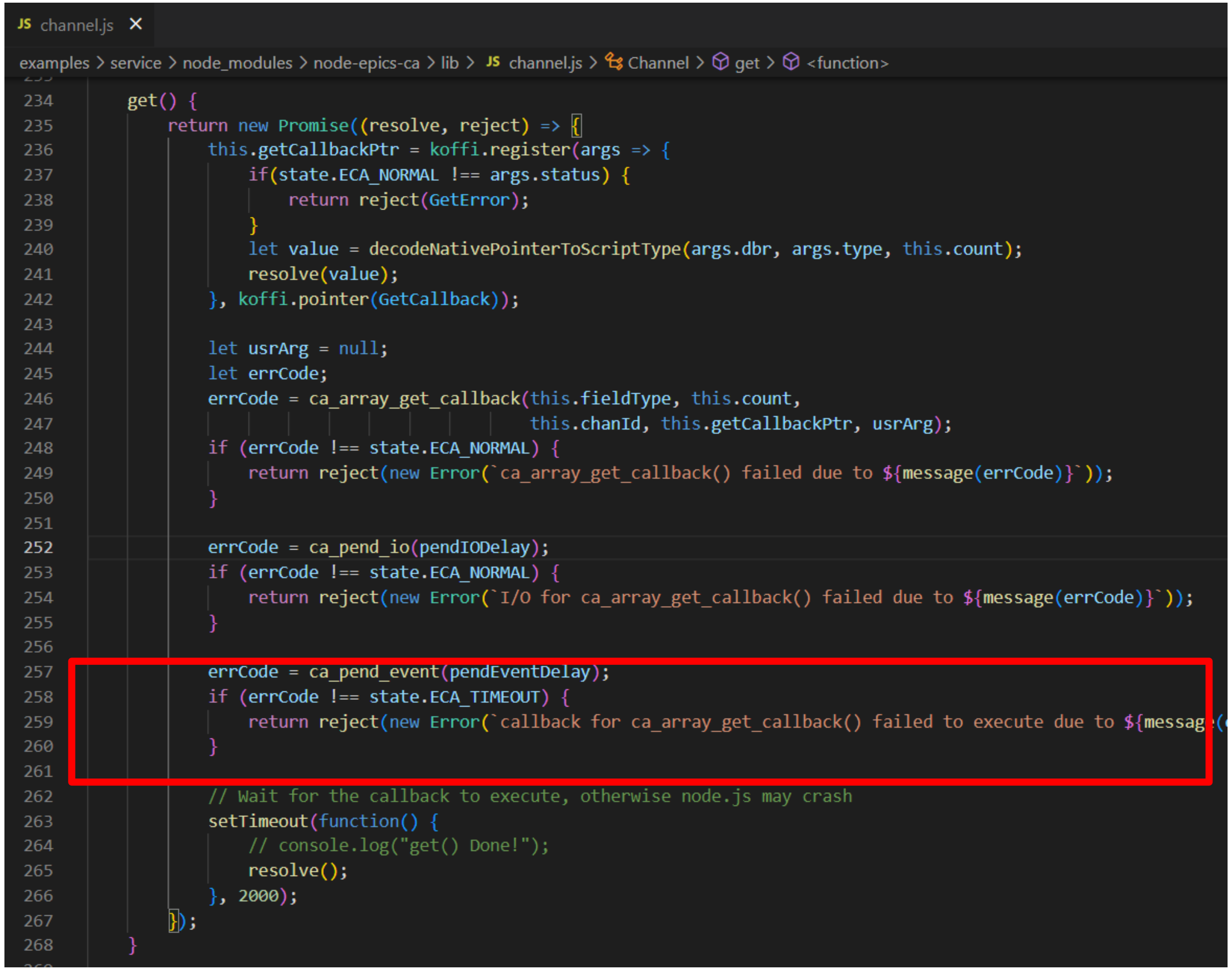
Potential problem #2: Performance issue due to ca_pend_event()

- ca_pend_event is used to wait the callback.

- However, ca_pend_event() behaves as a constant delay rather than event waiting inside get() and put() in node-epics-ca, which affects performance.

- Node.js is event-driven and single-thread, probably ca_pend_event() is not necessary, not sure so far.

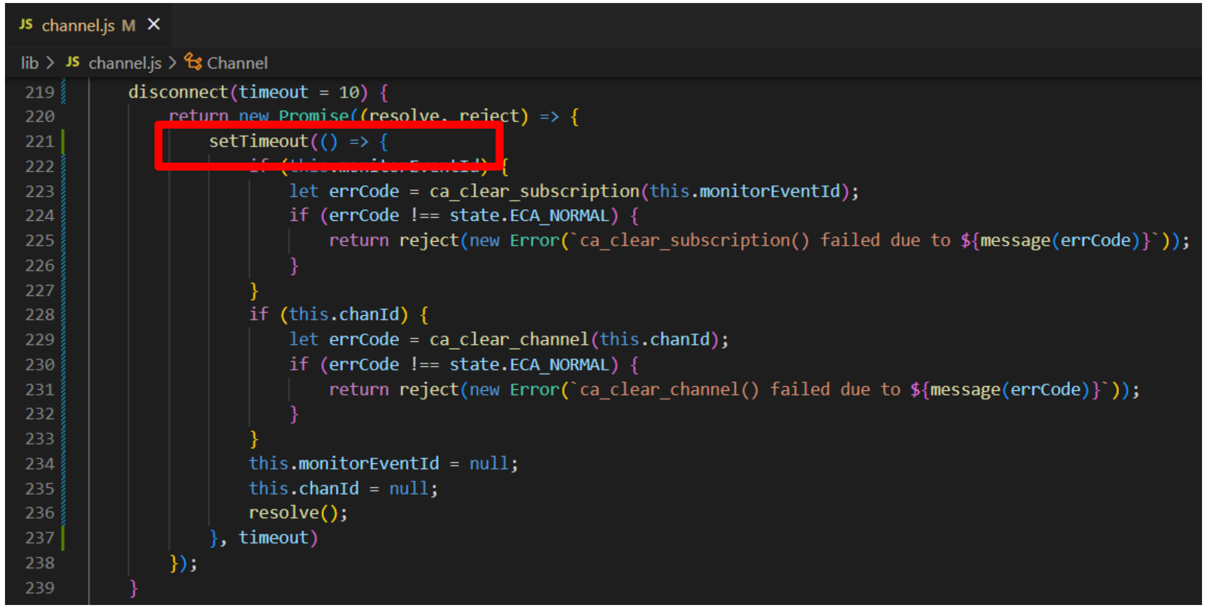
- Therefore, the question is whether ca_pend_event() can be get rid of in Node.js environment.


JS channel.js ✕
examples > service > node_modules > node-epics-ca > lib > JS channel.js > Channel > get > <function>

```
    get() {
        return new Promise((resolve, reject) => {
            this.getCallbackPtr = koffi.register(args => {
                if(state.ECA_NORMAL !== args.status) {
                    return reject(GetError);
                }
                let value = decodeNativePointerToScriptType(args.dbr, args.type, this.count);
                resolve(value);
            }, koffi.pointer(GetCallback));

            let usrArg = null;
            let errCode;
            errCode = ca_array_get_callback(this.fieldType, this.count,
                                            this.chanId, this.getCallbackPtr, usrArg);
            if (errCode !== state.ECA_NORMAL) {
                return reject(new Error(`ca_array_get_callback() failed due to ${message(errCode)}`));
            }

            errCode = ca_pend_io(pendIODelay);
            if (errCode !== state.ECA_NORMAL) {
                return reject(new Error(`I/O for ca_array_get_callback() failed due to ${message(errCode)}`));
            }

            errCode = ca_pend_event(pendEventDelay);
            if (errCode !== state.ECA_TIMEOUT) {
                return reject(new Error(`callback for ca_array_get_callback() failed to execute due to ${message(
            }

            // Wait for the callback to execute, otherwise node.js may crash
            setTimeout(function() {
                // console.log("get() Done!");
                resolve();
            }, 2000);
        });
    }
```

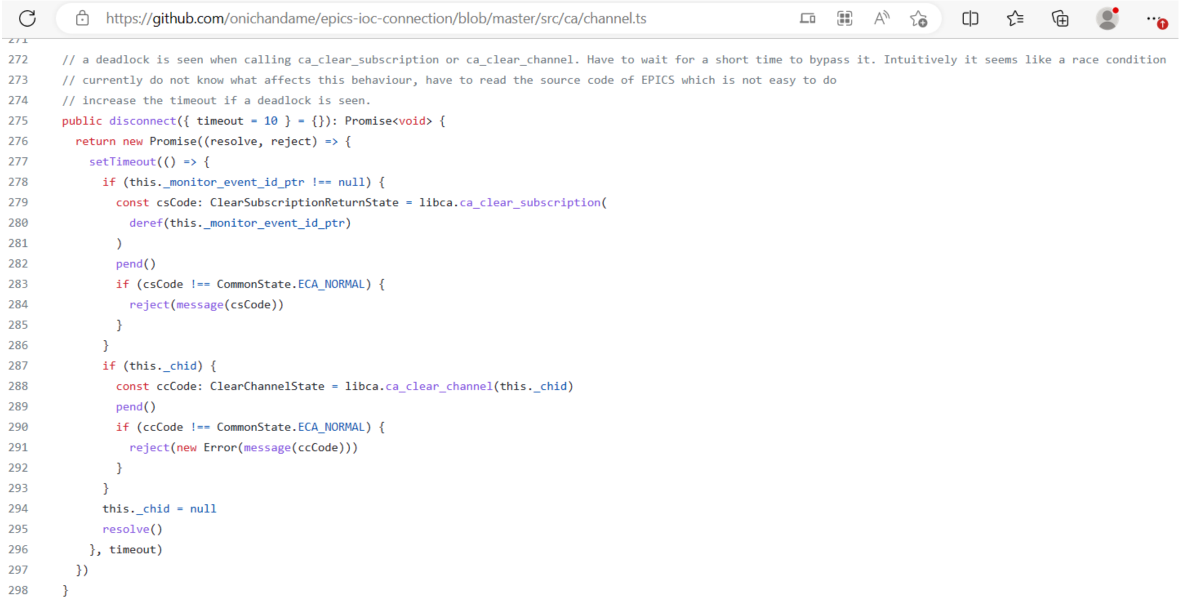
Potential problem #3: Deadlock issue when calling ca_clear_channel()

- **Sometimes deadlock occurs when ca_clear_channel() is called multiple times.**
- **Also reported in epics-ioc-connection package**
  - https://github.com/onichandame/epics-ioc-connection/blob/master/src/ca/channel.ts
- **Workaround is to add setTimeout() delay as suggested in epics-ioc-connection package to reduce the probability of occurrence.**



```javascript
    disconnect(timeout = 10) {
        return new Promise((resolve, reject) => {
            setTimeout(() => {
                if (this.monitorEventId) {
                    let errCode = ca_clear_subscription(this.monitorEventId);
                    if (errCode !== state.ECA_NORMAL) {
                        return reject(new Error(`ca_clear_subscription() failed due to ${message(errCode)}`));
                    }
                }
                if (this.chanId) {
                    let errCode = ca_clear_channel(this.chanId);
                    if (errCode !== state.ECA_NORMAL) {
                        return reject(new Error(`ca_clear_channel() failed due to ${message(errCode)}`));
                    }
                }
                this.monitorEventId = null;
                this.chanId = null;
                resolve();
            }, timeout)
        });
    }
```

Deadlock in node-epics-ca

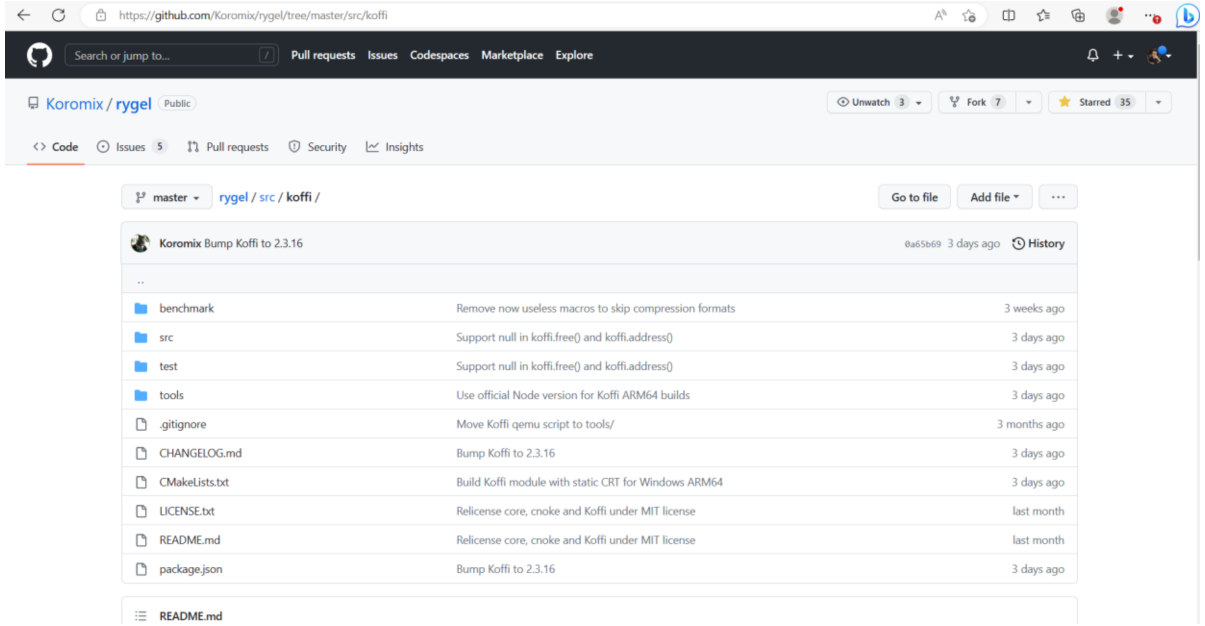


```typescript
  // a deadlock is seen when calling ca_clear_subscription or ca_clear_channel. Have to wait for a short time to bypass it. Intuitively it seems like a race condition
  // currently do not know what affects this behaviour, have to read the source code of EPICS which is not easy to do
  // increase the timeout if a deadlock is seen.
  public disconnect({ timeout = 10 } = {}): Promise<void> {
    return new Promise((resolve, reject) => {
      setTimeout(() => {
        if (this._monitor_event_id_ptr !== null) {
          const csCode: ClearSubscriptionReturnState = libca.ca_clear_subscription(
            deref(this._monitor_event_id_ptr)
          )
          pend()
          if (csCode !== CommonState.ECA_NORMAL) {
            reject(message(csCode))
          }
        }
        if (this._chid) {
          const ccCode: ClearChannelState = libca.ca_clear_channel(this._chid)
          pend()
          if (ccCode !== CommonState.ECA_NORMAL) {
            reject(new Error(message(ccCode)))
          }
        }
        this._chid = null
        resolve()
      }, timeout)
    })
  }
```

Deadlock in epics-ioc-connection

Potential problem #4: Package koffi lifetime issue

- Implementation of node-epics-ca heavily relies on koffi package
- koffi is a Node.js FFI library maintained by a software developer from France
- Not sure how long the lifetime of koffi will be, hope it be supported as long as possible



https://github.com/Koromix/rygel/tree/master/src/koffi

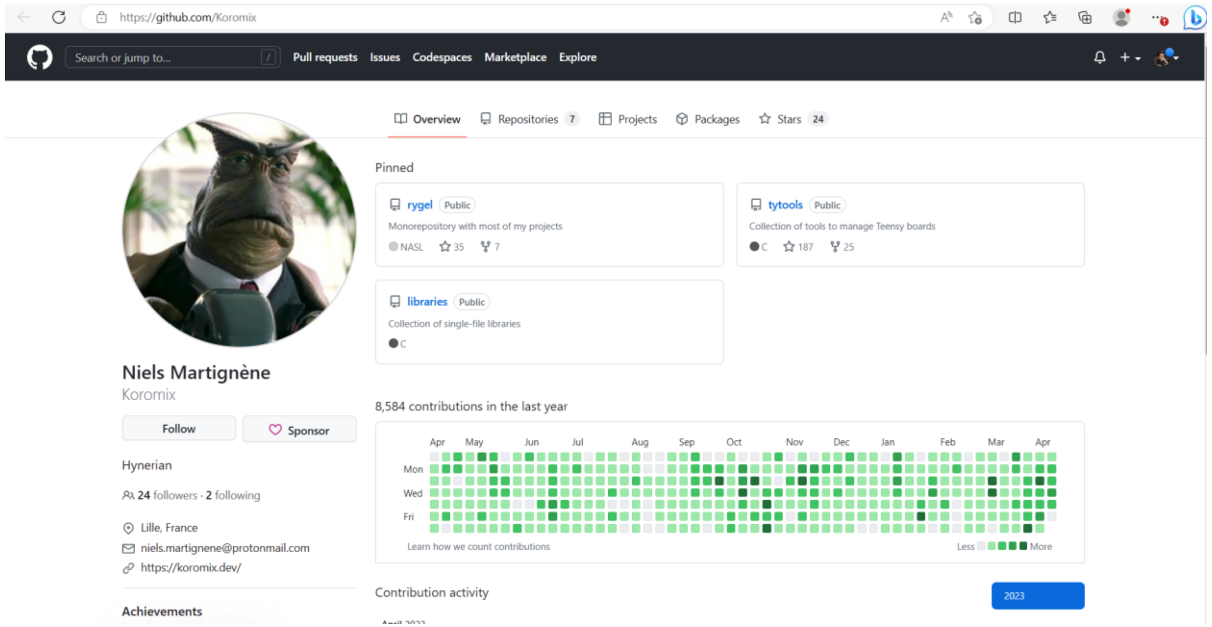
Koromix / rygel Public

rygel / src / koffi /

Koromix Bump Koffi to 2.3.16 0a65b69 3 days ago History

..		
benchmark	Remove now useless macros to skip compression formats	3 weeks ago
src	Support null in koffi.free() and koffi.address()	3 days ago
test	Support null in koffi.free() and koffi.address()	3 days ago
tools	Use official Node version for Koffi ARM64 builds	3 days ago
.gitignore	Move Koffi qemu script to tools/	3 months ago
CHANGELOG.md	Bump Koffi to 2.3.16	3 days ago
CMakeLists.txt	Build Koffi module with static CRT for Windows ARM64	3 days ago
LICENSE.txt	Relicense core, cnoke and Koffi under MIT license	last month
README.md	Relicense core, cnoke and Koffi under MIT license	last month
package.json	Bump Koffi to 2.3.16	3 days ago

README.md



https://github.com/Koromix

Pinned

rygel Public
Monorepository with most of my projects
NASL ☆ 35 7

tytools Public
Collection of tools to manage Teensy boards
C ☆ 187 25

libraries Public
Collection of single-file libraries
C

Niels Martignène
Koromix

Hynerian

24 followers · 2 following

Lille, France
niels.martignene@protonmail.com
https://koromix.dev/

Achievements

8,584 contributions in the last year

Contribution activity

Summary

- node-epics-ca is an EPICS Channel Access client for Node.js with FFI implementation
- It is developed based on existing node-epics and epics-ioc-connection packages.
- The shared libraries in clibs directory are directly taken from pyepics.
- The FFI package in use is koffi.
- Github and npm package links
  - https://github.com/wanglin86769/node-epics-ca
  - https://www.npmjs.com/package/node-epics-ca
- Development team
  - Design & Implementation: Lin Wang wanglin@ihep.ac.cn
  - Line Manager: Yuliang Zhang zhangyl@ihep.ac.cn

Thanks for your attention!