



N-API: Next Generation Node API for native modules

Arunesh Chandra, Microsoft

Michael Dawson, IBM

Arunesh Chandra, Microsoft

Senior Program Manager @ Microsoft

Node-ChakraCore Project

Time-Travel Debugging

Contact me:

Arunesh.Chandra@Microsoft.com

Twitter: @AruneshC

Github: @aruneshchandra



Michael Dawson, IBM

Senior Software Developer @ IBM

IBM Node.js community lead/IBM Runtime Technologies

Node.js collaborator, and TSC member

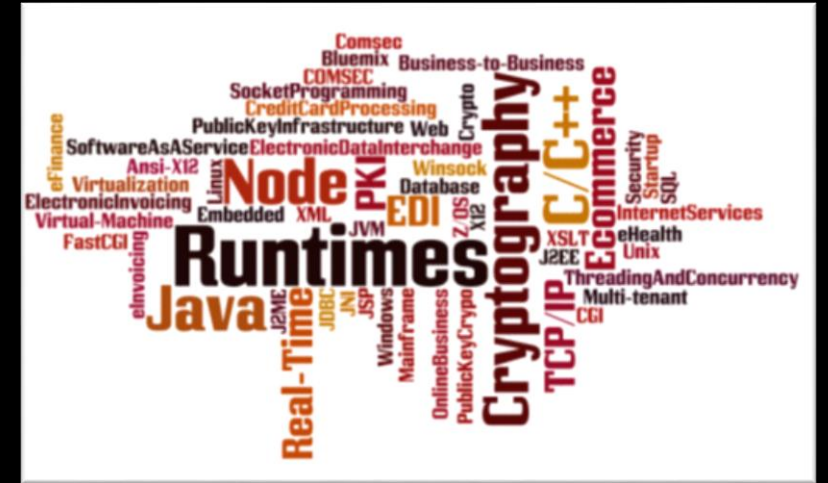
Active in LTS, build, benchmarking, api and post-mortem working groups

Contact me:

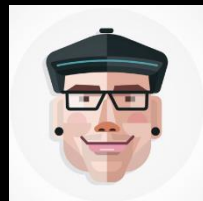
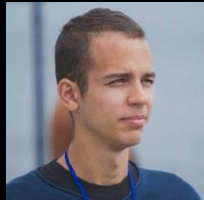
Twitter: @mhdawson1

Github: @mhdawson

Linkedin: <https://www.linkedin.com/in/michael-dawson-6051282>



Contributors





What is N-API ?

N-API is a **stable Node API layer** for native modules, that **provides ABI compatibility** guarantees across different Node versions & flavors.

N-API enables native modules to just work across different versions and flavors of Node.js **without recompilations!**



Why do I care about N-API ?

Reduces friction from upgrading to newer Node.js versions in production deployments



Dan Shaw

@dshaw

Following



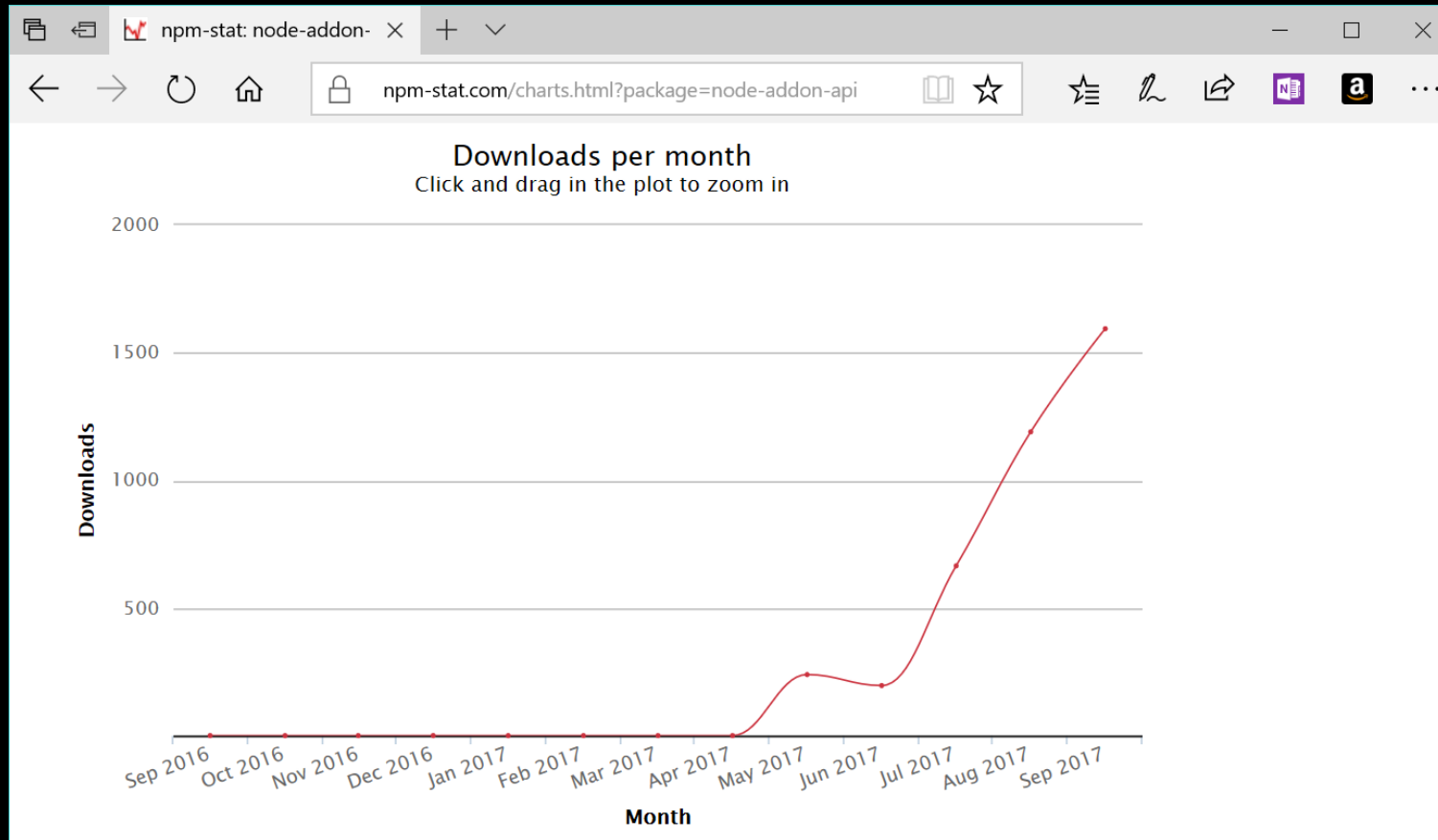
Replying to @dshaw @AruneshC and 4 others


Mission critical native code dependencies is the #1 reason I've been hearing for folks who haven't upgraded to node v4+.

12:05 PM - 3 Apr 2017


Reduces maintenance cost for Native module maintainers

The response so far ...



 **Mikeal Rogers**
@mikeal Following

Pretty big deal, new native layer for Node.js has landed

 **Ingvar Stepanyan**
@RReverser Following

@nodejs N-API is the kind of API that I wish JS engines would have had for a long time



Is it the new NaN ?

- N-API has more complete isolation from V8
- Compile once/run multiple versions
- Both C and C++ usage supported
- N-API expected to replace NaN usage

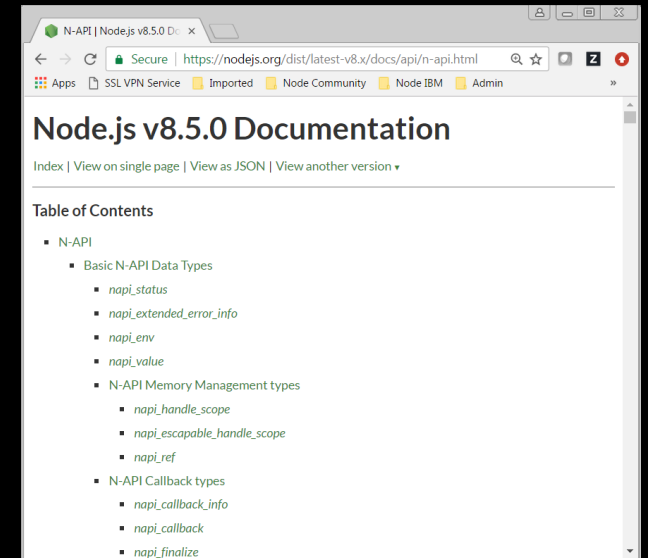


N-API Demo

API Shape



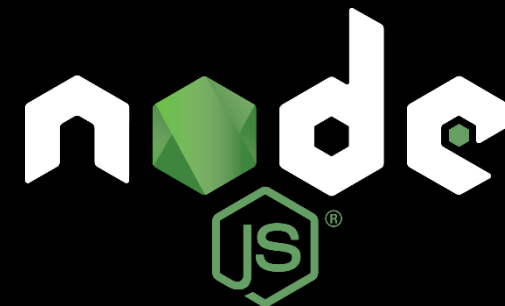
- Collection of C APIs available natively in Node.js 8.0
 - API Docs -
<https://nodejs.org/dist/latest-v8.x/docs/api/n-api.html>
 - ../src/node_api.h



```
napi_status napi_create_array(napi_env env, napi_value* result);  
napi_status napi_get_last_error_info(napi_env e, const napi_extended_error_info** result);  
napi_status napi_is_exception_pending(napi_env e, bool* result);  
napi_status napi_get_and_clear_last_exception(napi_env e, napi_value* result);  
napi_status napi_throw(napi_env e, napi_value error);
```

Examples – C- API

3_callbacks



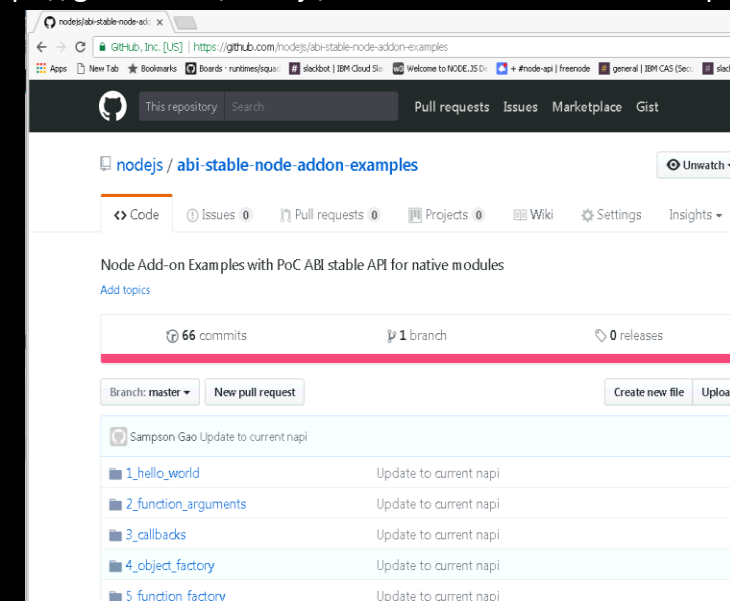
```
#include <node_api.h>
napi_value RunCallback(napi_env env, const napi_callback_info info) {
    ...
}

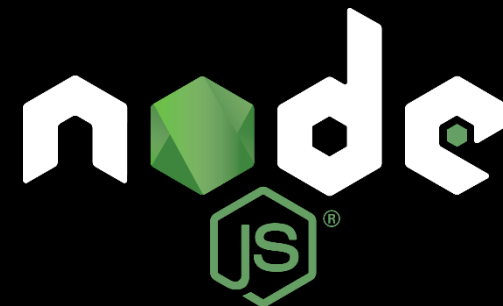
#define DECLARE_NAPI_METHOD(name, func) \
    { name, 0, func, 0, 0, 0, napi_default, 0 }

napi_value Init(napi_env env, napi_value exports) {
    napi_value new_exports;
    napi_status status =
        napi_create_function(env, "", NAPI_AUTO_LENGTH, RunCallback, nullptr, &new_exports);
    assert(status == napi_ok);
    return new_exports;
}

NAPI_MODULE(addon, Init);
```

<https://github.com/nodejs/abi-stable-node-addon-examples>





Examples – C- API

```
#include <node_api.h>

napi_value RunCallback(napi_env env, const napi_callback_info info) {
    napi_status status;

    size_t argc = 1;
    napi_value args[1];
    status = napi_get_cb_info(env, info, &argc, args, nullptr, nullptr);

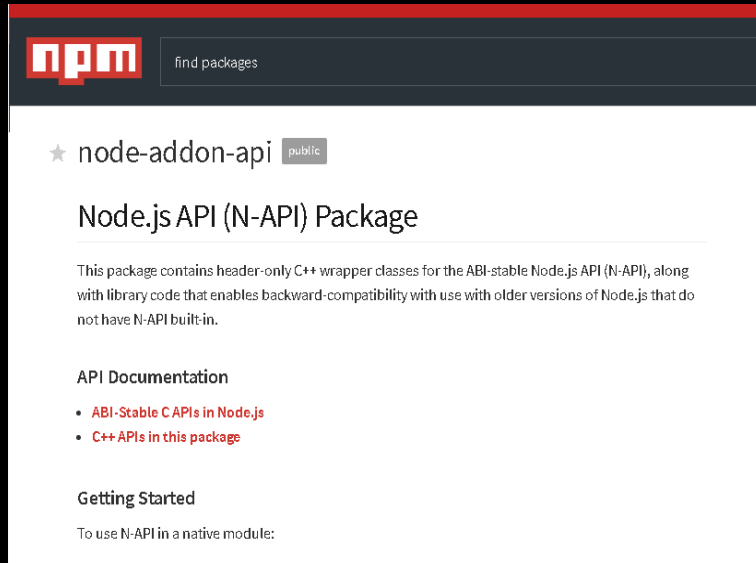
    napi_value cb = args[0];

    napi_value argv[1];
    status = napi_create_string_utf8(env, "hello world", NAPI_AUTO_LENGTH, argv);

    napi_value global;
    status = napi_get_global(env, &global);

    napi_value result;
    status = napi_call_function(env, global, cb, 1, argv, &result);
    return nullptr;
}
```

Example – C++ Wrapper Module

A screenshot of the npm website showing the package page for "node-addon-api". The page includes the npm logo, a search bar, and the package name "node-addon-api" with a "public" tag. Below this is the title "Node.js API (N-API) Package" and a description: "This package contains header-only C++ wrapper classes for the ABI-stable Node.js API (N-API), along with library code that enables backward-compatibility with use with older versions of Node.js that do not have N-API built-in." There are sections for "API Documentation" with links to "ABI-Stable C APIs in Node.js" and "C++ APIs in this package", and "Getting Started" with the instruction "To use N-API in a native module:".

npm find packages

★ node-addon-api public

Node.js API (N-API) Package

This package contains header-only C++ wrapper classes for the ABI-stable Node.js API (N-API), along with library code that enables backward-compatibility with use with older versions of Node.js that do not have N-API built-in.

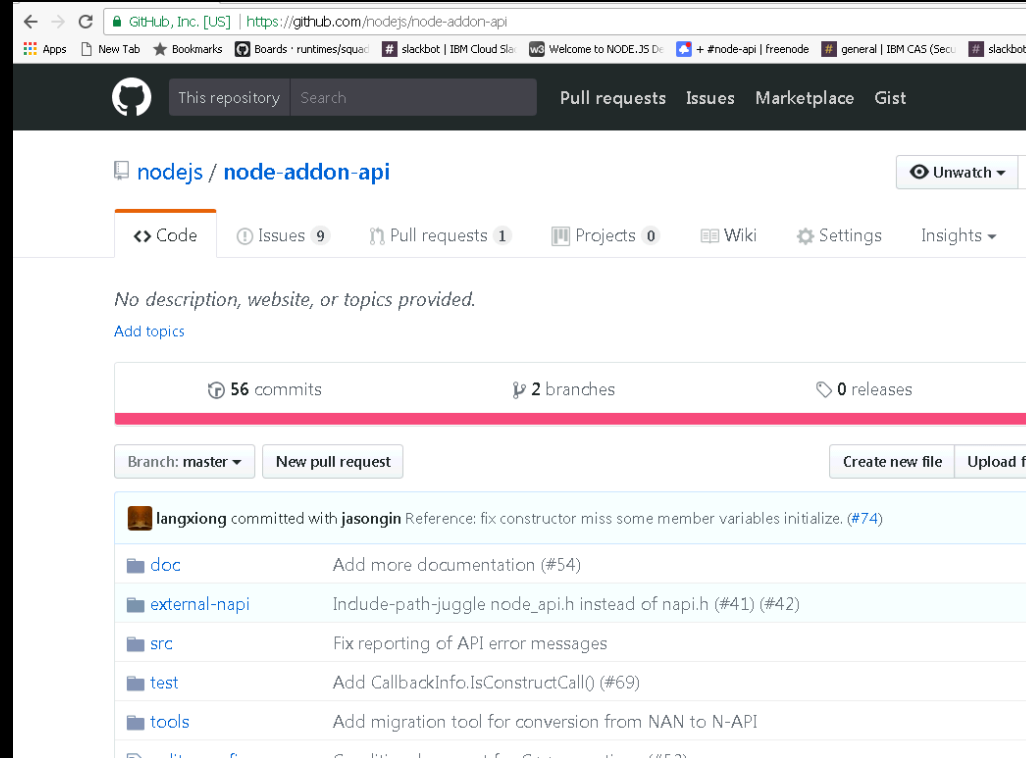
API Documentation

- [ABI-Stable C APIs in Node.js](#)
- [C++ APIs in this package](#)

Getting Started

To use N-API in a native module:

<https://www.npmjs.com/package/node-addon-api>

A screenshot of the GitHub repository page for "nodejs/node-addon-api". The page shows the repository name, a search bar, and navigation links for "Pull requests", "Issues", "Marketplace", and "Gist". Below this is a summary of the repository: "No description, website, or topics provided." and "Add topics". It also shows statistics: "56 commits", "2 branches", and "0 releases". There are buttons for "Branch: master", "New pull request", "Create new file", and "Upload file". A list of recent commits is visible, including one by "langxiong" titled "Reference: fix constructor miss some member variables initialize. (#74)".

GitHub, Inc. [US] | <https://github.com/nodejs/node-addon-api>

This repository Search Pull requests Issues Marketplace Gist

nodejs / node-addon-api Unwatch

Code Issues 9 Pull requests 1 Projects 0 Wiki Settings Insights

No description, website, or topics provided.

Add topics

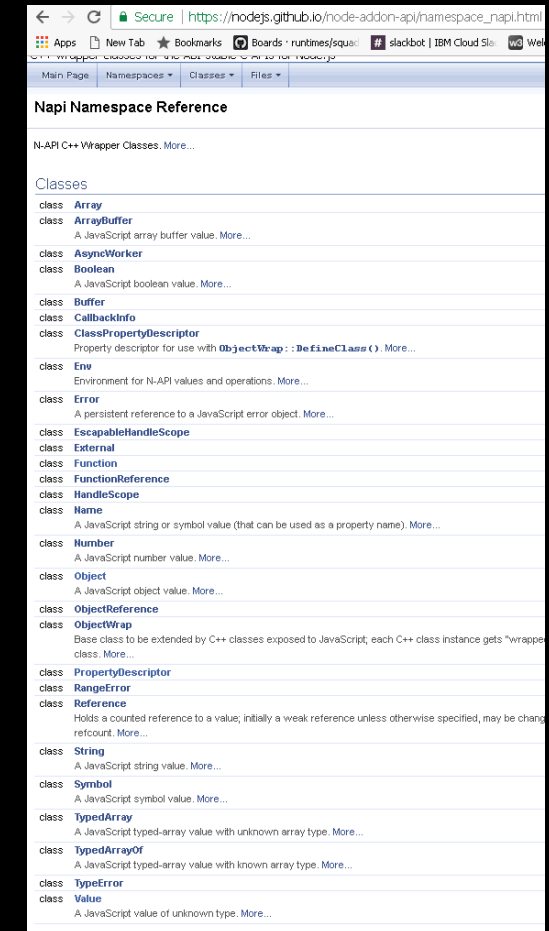
56 commits 2 branches 0 releases

Branch: master New pull request Create new file Upload file

langxiong committed with jasongin Reference: fix constructor miss some member variables initialize. (#74)

- doc Add more documentation (#54)
- external-napi Include-path-juggle node_api.h instead of napi.h (#41) (#42)
- src Fix reporting of API error messages
- test Add CallbackInfo.IsConstructCall() (#69)
- tools Add migration tool for conversion from NAN to N-API
- editorconfig Conditional support for C++ exceptions (#52)

<https://github.com/nodejs/node-addon-api>

A screenshot of the "Napi Namespace Reference" page from the Node.js GitHub repository. The page lists various N-API C++ wrapper classes and their descriptions. The classes listed include: Array, ArrayBuffer, AsyncWorker, Boolean, Buffer, CallbackInfo, ClassPropertyDescriptor, Env, Error, EscapableHandleScope, External, Function, FunctionReference, HandleScope, Name, Number, Object, ObjectReference, ObjectWrap, PropertyDescriptor, RangeError, Reference, String, Symbol, TypedArray, TypedArrayOf, TypeError, and Value. Each class has a brief description and a link to "More...".

Napi Namespace Reference

N-API C++ Wrapper Classes. More...

Classes

- class **Array**
- class **ArrayBuffer**
A JavaScript array buffer value. More...
- class **AsyncWorker**
- class **Boolean**
A JavaScript boolean value. More...
- class **Buffer**
- class **CallbackInfo**
- class **ClassPropertyDescriptor**
Property descriptor for use with `ObjectWrap::DefineClass()`. More...
- class **Env**
Environment for N-API values and operations. More...
- class **Error**
A persistent reference to a JavaScript error object. More...
- class **EscapableHandleScope**
- class **External**
- class **Function**
- class **FunctionReference**
- class **HandleScope**
- class **Name**
A JavaScript string or symbol value (that can be used as a property name). More...
- class **Number**
A JavaScript number value. More...
- class **Object**
A JavaScript object value. More...
- class **ObjectReference**
- class **ObjectWrap**
Base class to be extended by C++ classes exposed to JavaScript; each C++ class instance gets "wrapped" class. More...
- class **PropertyDescriptor**
- class **RangeError**
- class **Reference**
Holds a counted reference to a value; initially a weak reference unless otherwise specified; may be changed to a strong reference. More...
- class **String**
A JavaScript string value. More...
- class **Symbol**
A JavaScript symbol value. More...
- class **TypedArray**
A JavaScript typed-array value with unknown array type. More...
- class **TypedArrayOf**
A JavaScript typed-array value with known array type. More...
- class **TypeError**
- class **Value**
A JavaScript value of unknown type. More...

https://nodejs.github.io/node-addon-api/namespace_napi.html



Example – C++ Wrapper Example

NaN

node-addon-api

```
#include <nan.h>
```

```
#include <napi.h>
```

```
void RunCallback(const Nan::FunctionCallbackInfo<v8::Value>& info) {  
    v8::Local<v8::Function> cb = info[0].As<v8::Function>();  
    const unsigned argc = 1;  
    v8::Local<v8::Value> argv[argc] =  
        { Nan::New("hello world").ToLocalChecked() };  
    Nan::MakeCallback(Nan::GetCurrentContext()->Global(), cb, argc, argv);  
}
```

```
void RunCallback(const Napi::CallbackInfo& info) {  
    Napi::Env env = info.Env();  
    Napi::Function cb = info[0].As<Napi::Function>();  
    cb.MakeCallback(env.Global(), {  
        Napi::String::New(env, "hello world")  
    });  
}
```



```
void Init(v8::Local<v8::Object> exports,  
          v8::Local<v8::Object> module) {  
    Nan::SetMethod(module, "exports", RunCallback);  
}
```

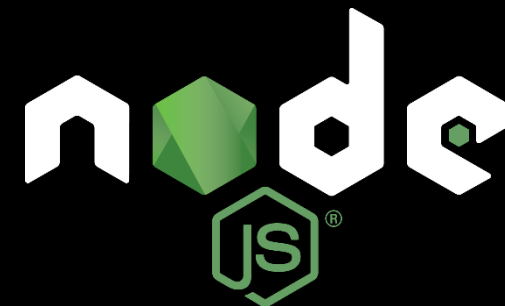
```
Napi::Object Init(Napi::Env env, Napi::Object exports) {  
    return Napi::Function::New(env, RunCallback);  
}
```

```
NODE_MODULE(addon, Init)
```

```
NODE_API_MODULE(addon, Init)
```

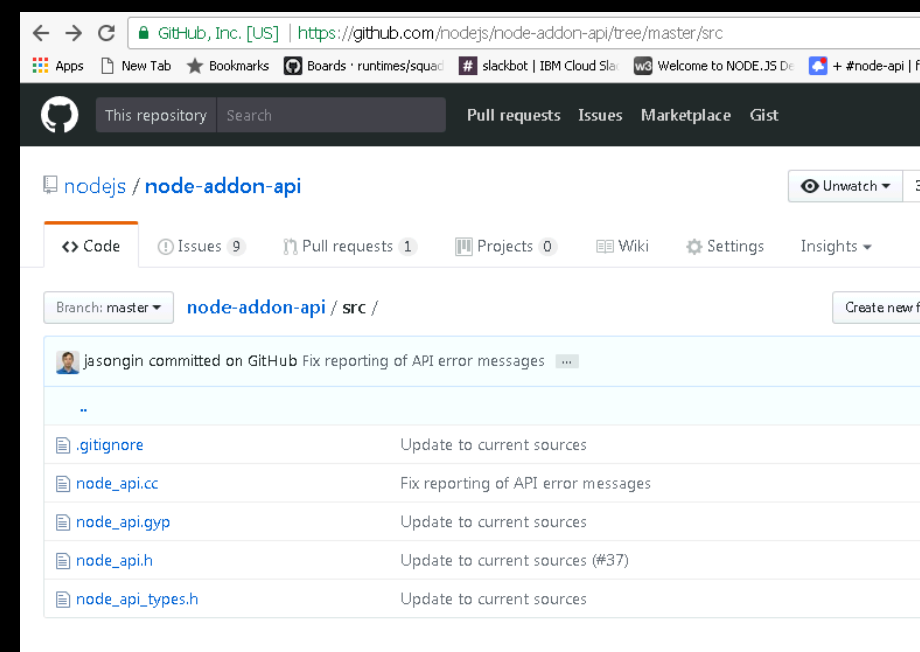
https://github.com/nodejs/abi-stable-node-addon-examples/tree/master/3_callbacks/node-addon-api

<https://github.com/nodejs/node-addon-api/blob/master/tools/conversion.js>



Backward Compatibility

- Available in 8.x
- Plan to backport to 6.x
- For 4.x
 - Copy of API built into node-addon-api module
 - Do **NOT** get build once/run any version
 - Can maintain module with single code base



Calling all native module maintainers and Node.js enthusiasts



- Help wanted to port popular native modules
- Try out one of the ported modules and provide feedback
- Improve the documentation
- Improve test coverage
- Join the N-API Working Group

<https://github.com/nodejs/abi-stable-node>



Useful Links

- Conversion tool for migrating existing NaN modules to N-API
<https://github.com/nodejs/node-addon-api/blob/master/tools/conversion.js>
- Yeoman generator for building N-API module from scratch
<https://github.com/digitalinfinity/generator-napi-module> (C++ Wrapper)
- Guidance for publishing N-API versions of your modules
<https://nodejs.org/en/docs/guides/publishing-napi-modules/>

Copyright and Trademarks

© IBM Corporation and Microsoft Corporation, 2017. All Rights Reserved

IBM, the IBM logo, ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies.

A current list of IBM trademarks is available on the Web at

“Copyright and trademark information” at

www.ibm.com/legal/copytrade.shtml

Microsoft is a trademark of Microsoft Corporation in the United States, other countries, or both.

Node.js is an official trademark of Joyent. IBM SDK for Node.js is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

Java, JavaScript and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

npm is a trademark of npm, Inc.