

Developing Node.js Services

Structure of a Node.js Service Repository

A Node.js service repository has to be a ZIP file containing at least two elements:

- A file **package.json** containing some service information the Bridge needs for managing the service.
- All JavaScript source code and resources.

If your Node.js service needs additional modules (libraries), you need to install all modules locally and include the sub-folder **node_modules**. You cannot install modules globally to the Bridge Node.js. You can use 'npm install' to install those additional modules (see [Deploying a Node.js Service Repository via the E2E Bridge](#)).

Structure of package.json

The following json elements are required in the file package.json:

```
{
  "name": "HelloWorld",
  "version": "0.0.1",
  "description": "Hello world HTTP server",
  "scripts": {
    "start": "node server.js"
  },
}
```

Element	Description	Mandatory
name	Contains the name of the Node.js service as displayed on the Bridge. This is the name taken into account when applying the option Overwrite Service .	mandatory
version	Contains the version of the Node.js service.	optional
description	Contains a description of the Node.js service.	optional
scripts /start	Contains the call to start the Node.js service: node <name of the start JavaScript file>.js	mandatory

Additional json elements can be added to package.json as needed (e.g. for adding packages/libraries), but will be ignored by the Bridge.

Environment Variables

The Bridge provides some environment variables that are strongly recommended to be used by developers in their code:

Variable	Description	Default
PORT	Node.js service port	-
CONFIG_DIR	Path to the configuration directory of the service (within the working directory of the service). • All configuration files (e.g. service settings) that are stored in this location will survive a Node.js service update. Files that have been saved to the working directory only, will be deleted on update. • This directory will be created by the Bridge on service startup, if not existent.	repository/config/local
E2E_CONF_ROOT	If you use the e2e-conf Node.js module , you can initialize e2e-conf with E2E_CONF_ROOT . This is especially useful if your e2e-conf is not a top level dependency but a sub-module dependency.	repository

On this Page:

- [Structure of a Node.js Service Repository](#)
 - [Structure of package.json](#)
- [Environment Variables](#)
- [Writing to Bridge Log Files](#)
 - [Trigger Bridge Monitoring From the Error Log](#)
- [Implementing a Service Shutdown Activity](#)

Related Pages:

- [e2e-conf - Easy Configuration for Bridge Node.js Services](#)
- [Setting Environment Variables for Node.js Services](#)
- [Node.js documentation on signal events](#)

Related Documentation:

- [Deployment of Node.js Services](#)
- [Logging of Node.js Services](#)
- [Bridge Monitoring](#)

E2E_LOGS	If the Node.js service writes log files in this directory, you can view them in the Bridge. - If the log file keeps growing, you should rotate the log file per day and use this naming pattern: <prefix>_YYYY-MM-DD.log. - If the log file has a fixed size, you can use the naming pattern <prefix>.log. Do not use the following prefixes: start, stdout, stderr and transaction. They are reserved prefixes by the Bridge.	logs
-----------------	--	------

These special environment variables are displayed on the details tab of any Node.js service. Add and configure them on tab **Environment Variables** of the Node.js service (see [Setting Environment Variables for Node.js Services](#)).

Writing to Bridge Log Files

If you are writing messages to log files that will be displayed on the Bridge (npm-install, start, stdout, stderr, custom logfiles), you must use **UTF-8 encoding**. If you do not, special characters may be displayed wrongly in the [log view](#).

Trigger Bridge Monitoring From the Error Log

Bridge 7.3.0 The Bridge comes with a built-in monitoring functionality, which can catch failure events for any service running on the Bridge (see [Bridge Monitoring](#)). If such an error event occurs, the Bridge calls the registered monitoring service, that executes whatever has to be done in this case, for instance sending an e-mail to the system administrator. The monitoring service is just another SOAP service that is running on the Bridge and is registered as monitoring service.

The monitoring service has a standardized interface (see [Monitoring](#)) to which you need to provide the necessary data with your Node.js log. Generally, all lines from **stderr** are written to the Bridge error log as they are. As the Bridge will read this information from your Node.js logs and provide it to the monitoring service, your log file should have the following structure:

All values (except the actual error message) are to be provided in square brackets.

```
[Timestamp][Log Level][Service/Domain][Code][Unused 1][Unused 2] - Error
Message
[09:09:43][Error][SAP-Interface/SAP][815A][[]] - File not found.
```

Log Field Name	Description	Log Examples	Monitoring Interface	Monitoring Examples
Timestamp	Timestamp in local system time, time part only. The Bridge will add the date part and convert the date /time to UTC. The timestamp is the marker for the beginning of a log entry.	[09:09:43]	timestamp	2019-03-13T08:09:43Z
Log Level	The Bridge will only forward log messages with log level Error to the monitoring service. The given log level value will then be converted to lowercase (first letter uppercase).	[Error] [ERROR]	level	Error
Service/Domain	Service identifier of the Node.js service. Provide here two values separated by a '/'. The Bridge will split this value into service and type .	[SAP_Interface/SAP]	service	SAP_Interface
			type	SAP
Code	The error code of the error.	[815A] [#156]	code	815A #156
Unused 1	Two dummy fields which are needed for internal compatibility reasons. If you log some information here, this will be ignored by the Bridge.	[]		
Unused 2		[]		

Error Message	The actual error message, separated by ' - '. The error message may contain multiple lines. The timestamp is the marker for the beginning of a new log entry.	- File not found.	description	File not found.
	Automatically provided by the Bridge.		host	e2ebridge.e2e.ch
	Automatically provided by the Bridge.		processID	4711
	Automatically provided by the Bridge. The Bridge will provide this value as unknown to the monitoring service.		category	unknown
	Automatically provided by the Bridge. The Bridge will provide the path to the corresponding Node.js service log file.		detailLocation	/opt /E2E_BRIDGE_DATA /nodejs_SAP_Interface /logs /stderr_2019-03-19.log

Implementing a Service Shutdown Activity

Bridge 7.2.0 Upon stopping a Node.js service, the Bridge will send an operating system signal (SIGINT) to the service to stop it. If you want to do some clean-up actions before stopping, you have to implement a signal handler for SIGINT in your Node.js service.

For more information, refer to the [Node.js documentation on signal events](#).