Welcome to Node.js

As the primary "client-side" development team for our project, we found the need to have a layer between the server-side APIs and the client-side code. Otherwise known as the "translation layer." The translation layer provides the opportunity for client-side developers to create UI-specific services, render template languages server-side that are compatible with JavaScript and share code between the client and server. This is where "Node.js" has many advantages to offer.

Node.js is a server-side JavaScript runtime built upon Google's V8 with a thin core API (networking, file system, OS, etc). The runtime implements a single-threaded, reactive event loop, which allows easy construction of realtime applications. Due to the nature of the event loop, Node.js is able to handle large numbers of active concurrent connections. This makes developing network servers (TCP/UDP/HTTP) and highly dynamic and realtime applications extremely easy. Node is platform agnostic and runs on all major operating systems and hardware.

Node offers several key advantages over other server-side languages, such as:

* Familiar language and toolset to client-side developers. Libraries are often shared between client-side and server-side.
* Integration between client-side JavaScript applications and server-side web services.
* Large open-source community with a single central package repository (NPM).
* Quickly develop stateless, non-blocking applications.
* KISS principals first.

# [Building Your First App](https://code.bestbuy.com/wiki/display/FED/Building+Your+First+App)

# Getting Started

Begin by creating a new Node.js application using NPM:

|  |
| --- |
| mkdir my-node-project  cd my-node-project  npm init |

**name: (my-node-project)** – Use the default value.  
**version: (1.0.0)** – Use the default value.  
**description:**– Enter a description of your project. This is for display purposes only.  
**entry point: (index.js)** – Entry point is only used if other modules will be requiring your project as a dependency. Use the default value.  
**test command:** – Use the value (without the period) "app-node-test-helper."  
**git repository:** – Enter the URL for your project on Stash. This is not a required property, but NPM will warn if its missing.  
**keywords:** – No value is needed as you won't be publishing this project to the public NPM registry.  
**author:** – No value is needed as you won't be publishing this project to the public NPM registry.  
**license:**  – No value is needed as you won't be publishing this project to the public NPM registry.  
**Is this ok? (yes)** – Use default value.

Create your projects initial structure:

|  |
| --- |
| mkdir lib test views logs pm2  touch README.md server.js .gitignore pm2/processes.json pm2/processes-dev.json logs/.gitkeep |

# Dependencies

Add the base set of dependencies to your project:

|  |
| --- |
| npm install --save express body-parser app-express-helper app-node-logger object-mapper  npm install --save-dev pm2 app-node-test-helper |

# Ignored Files

Ignore temporary files generated by tools and build systems:

**.gitignore**

|  |
| --- |
| /coverage  /frontend  /node\_modules  /views  /deploy  \*.tgz  \*.log |

# Entry Point

Create your application's entry point file:

**server.js**

|  |
| --- |
| var express = require("express");  var log = require("app-node-logger");  var appHelper = require("app-express-helper");  var path = require("path");  var pkg = require("./package.json");    log.SetFormat(process.env.LOG\_FORMAT || "Splunk");  log.SetLevel(process.env.LOG\_LEVEL || "info");  log.SetUtilOptions({      depth: null  });    var app = express();    app.use(appHelper.requestLogger(log));    app.use("/heartbeat", appHelper.heartbeat({      applicationVersion: pkg.version  }));    app.listen(process.env.PORT || 3000); |

# PM2 – Production

Configure your application to run in production mode:

**pm2/processes.json**

|  |
| --- |
| {      "apps": [{          "name": "my-node-project",          "script": "server.js",          "env": {              "NODE\_ENV": "production",              "LOG\_FORMAT": "Splunk",              "LOG\_LEVEL": "info",          },          "exec\_mode": "cluster\_mode",          "instances": "max",          "error\_file": "/opt/pm2/log/my-node-project.log",          "out\_file": "/opt/pm2/log/my-node-project.log",          "merge\_logs": true      }]  } |

# PM2 – Development

Configure your application to run in development mode:

**pm2/processes-dev.json**

|  |
| --- |
| {      "apps": [{          "name": "my-node-project",          "script": "server.js",          "env": {              "NODE\_ENV": "dev",              "LOG\_FORMAT": "PreTTY",              "LOG\_LEVEL": "debug"          },          "exec\_mode": "fork\_mode",          "instances": 1,          "watch": [              "lib",              "server.js"          ],          "ignoreWatch": [              "node\_modules"          ],          "error\_file": "logs/my-node-project-error.log",          "out\_file": "logs/my-node-project-out.log",          "merge\_logs": true      }]  } |

# Project Control

Configure NPM to start/stop and control PM2:

**package.json**

|  |
| --- |
| {    "scripts": {      "test": "app-node-test-helper",      "start": "pm2 start pm2/processes-dev.json",      "stop": "pm2 delete pm2/processes-dev.json",      "pm2": "pm2"    }  } |

# Try it Out

Start your application by using NPM:

|  |
| --- |
| npm start  curl http://localhost:3000/heartbeat  #> {"applicationVersion": "1.0.0"} |