



Démo – Node.js

Autoscaling App Node.js sur Openshift

Sylvain Lavoie
Version 1.0 - juin 2017



Desjardins

Coopérer pour créer l'avenir

Agenda

1

Présentation de la plateforme « Node.js »

2

Notre site web « Ma filmothèque »

3

Préparation de l'environnement OpenShift

4

Outil de charge « httpperf »

1 Présentation de la plateforme « Node.js »

➤ Historique

- Écrit par Ryan Dahl en 2009.
- Il a été inspiré par une barre de progression dans un navigateur qui est déconnecté du serveur.
- Projet « open source » et multiplateforme.

➤ Caractéristiques

- C'est un « JavaScript runtime » s'appuyant sur l'engin « Chrome's V8 JavaScript ».
- Modèle « I/O non-blocking ».
- Architecture événementielle (Event-driven).
- Un des plus gros écosystèmes de librairie « open source » avec « NPM ».
 - Plusieurs frameworks dont Express, React, Angular, Meteor.

➤ Applications écrites en JavaScript sous Node.js (<https://electron.atom.io/apps/>)

- Microsoft Visual Studio Code.
- Atom A hackable text editor.
- GitHub Desktop.

➤ Plus de 1400 contributeurs au projet, dont plusieurs compagnies.

➤ Stack MEAN (MongoDB Express Angular Node.js)

1 Les utilisateurs de Node.js

- Voir la stack des autres compagnies (<https://stackshare.io/>)

Node.js 

Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications

Votes	Fans	Stacks	Integrations	Jobs New
6.37K	7.24K	6.57K	66	1.79K

COMPANIES USING NODE.JS

The screenshot displays a grid of 60 company logos, arranged in 6 rows and 10 columns. The logos represent various companies that use Node.js in their technology stacks. Some recognizable logos include Twitter, Uber, Netflix, Airbnb, and many others.

2 Notre site web « Ma filmothèque »

- Site Web sous Node.js.
 - Sources sur Github : <https://github.com/sylvoie/testoc.git>
- Déploiement et recréation de l'environnement :
 - Selon le fichier configuration « package.json »
 - Exécute commande « npm install »
 - Installation « dependencies »
 - Création répertoire « node_modules »
 - Exécute « start »: « node app.js »

- Framework :
 - Bootstrap
 - Express
 - Embedded JavaScript templates (EJS)

- Résultat de l'application
 - Un page Web
 - 28 requêtes HTTP
 - Dont 19 images
 - Poids 496.8 KB

```
{
  "name": "testoc",
  "version": "1.0.0",
  "description": "Site Web pour test de charge dans OpenShift",
  "main": "app.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1",
    "start": "node app.js"
  },
  "author": "Sylvain Lavoie",
  "license": "ISC",
  "dependencies": {
    "ejs": "^2.4.1",
    "express": "^4.13.4",
    "request": "^2.72.0"
  }
}
```

package.json

Ma filmothèque Login Sign Up

La télésérie CYBER CSI saison 2

<p>Why-Fi D.B. Russell joins the team as the new Director of Next Generation Forensics while they investigate a case of burglary and homicide committed by someone who hacked the home's security system. Date de diffusion : 6 Oct. 2015</p>	<p>Heart Me Raven sets out to prove a friend's innocence in the murder of a man she met through a dating app. Date de diffusion : 13 Oct. 2015</p>	<p>Brown Eyes, Blue Eyes The Cyber team tries to find the person who hacked a police officer's body camera and made the incendiary video go viral. Date de diffusion : 20 Oct. 2015</p>
<p>Red Crone The Cyber team investigates a child abduction case inspired by a myth where an abductor lures children through a cell phone app. Date de diffusion : 27 Oct. 2015</p>	<p>Hack E.R. When a hacker takes control of all networked medical devices at a hospital in Dallas and threatens to kill one patient every hour if his demands are not met, the Cyber team must find the source and figure out how they accessed an airtight security system. Date de diffusion : 3 Nov. 2015</p>	<p>Gone in 6 Seconds Avery and the team track down a hacker who's cyber-jacking vehicles and using them as remote-controlled cars to cause crashes. Date de diffusion : 10 Nov. 2015</p>

Hands-on OpenShift



3 Création de notre projet

OPENSIFT CONTAINER PLATFORM

New Project

* Name

syltest3

A unique name for the project.

Display Name

syltest3

Description

3ième test

Create

Cancel

3 Choix d'un langage dans le catalogue

syltest3 » Add to Project

[Browse Catalog](#) Deploy Image Import YAML / JSON

Choose from web frameworks, databases, and other components to add content to your project.

Filter by name or description

Languages



Java

JS

JavaScript

.NET

.NET



Perl



PHP



Python



Ruby

Technologies

3 Précision du langage

syltest3 » Add to Project » Catalog » JavaScript

JavaScript

Filter by name or description



Node.js

BUILDS SOURCE CODE

Build and run Node.js 4 applications on RHEL 7. For more information about using this builder image, including OpenShift considerations, see <https://github.com/sclorg/s2i-nodejs-container/blob/master/...>

Version

4 — latest

Select



Node.js + MongoDB (Persistent)

nodejs-mongo-persistent

Select



Node.js + MongoDB (Ephemeral)

nodejs-mongodb-example

Select

3 Identifier notre projet et la source de notre app

sytest3 » Add to Project » Catalog » Node.js

Loading...



Node.js

Version: 4

* Name

Identifies the resources created for this application.

* Git Repository URL

Sample repository for nodejs: <https://github.com/openshift/nodejs-ex.git> [Try It ↑](#)

Show [advanced options](#) for source, routes, builds, and deployments.

Create

Cancel

➤ <https://github.com/sylvoie/testoc.git>

3 Paramètre de route pour rejoindre notre site web

Routing

[? About Routing](#)

Create a route to the application

Hostname

Public hostname for the route. If not specified, a hostname is generated.

The hostname can't be changed after the route is created.

Path

Path that the router watches to route traffic to the service.

Target Port

Target port for traffic.

Security

Secure route

Routes can be secured using several TLS termination types for serving certificates.

3 Paramètre de déploiement et construction

Build Configuration

[? About Build Configuration](#)

- Configure a webhook build trigger [?](#)
- Automatically build a new image when the builder image changes [?](#)
- Launch the first build when the build configuration is created

Environment Variables (Build and Runtime) [?](#)

<input type="text" value="name"/>	<input type="text" value="value"/>	×
-----------------------------------	------------------------------------	---

[Add Environment Variable](#)

Deployment Configuration

[? About Deployment Configuration](#)

Autodeploy when

- New image is available
- Deployment configuration changes

Environment Variables (Runtime only) [?](#)

[Show Image Environment Variables](#)

<input type="text" value="name"/>	<input type="text" value="value"/>	×
-----------------------------------	------------------------------------	---

[Add Environment Variable](#)

3 Paramètres d'élasticité

Scaling

[About Scaling](#)

Strategy

Automatic

Scale replicas manually or automatically based on CPU usage.

[Learn More](#)

Min Pods

1

The lower limit for the number of pods that can be set by the autoscaler. If not specified, defaults to 1.

* Max Pods

4

The upper limit for the number of pods that can be set by the autoscaler.

CPU Request Target

80

%

The percentage of the CPU request that each pod should ideally be using. Pods will be added or removed periodically when CPU usage exceeds or drops below this target value. Defaults to 80%.

[Learn More](#)

You should configure resource limits below for autoscaling. Autoscaling will not work without a CPU request.

3 Ajout d'une limitation des ressources CPU et RAM

Resource Limits

[? About Resource Limits](#)

CPU

Request

250

millicores

The minimum amount of CPU the container is guaranteed.

Limit

500

millicores

The maximum amount of CPU the container is allowed to use when running.

[What are millicores?](#)

Memory

Request

256

MiB

The minimum amount of memory the container is guaranteed.

Limit

512

MiB

The maximum amount of memory the container is allowed to use when running.

[What are MiB?](#)

3 Ajout d'étiquette

Labels

[? About Labels](#)

The following labels are being added automatically. If you want to override them, you can do so below.

Each label is applied to each created resource.

X

[Add Label](#)

Hide [advanced options](#) for source, routes, builds, and deployments.

Create

Cancel

3 Votre projet est complété

[syltest3](#) » [Add to Project](#) » [Node.js](#) » [Next Steps](#)

Application created. [Continue to overview.](#)

Manage your app

The web console is convenient, but if you need deeper control you may want to try our command line tools.

Command line tools

[Download and install](#) the `oc` command line tool. After that, you can start by logging in, switching to this particular project, and displaying an overview of it, by doing:

```
oc login https://openshift-master.ose3sandbox.com
oc project syltest3
oc status
```

For more information about the command line tools, check the [CLI Reference](#) and [Basic CLI Operations](#).

Making code changes

A GitHub [webhook trigger](#) has been created for the **testoc** build config.

You can now set up the webhook in the GitHub repository settings if you own it, in <https://github.com/sylvoie/testoc/settings/hooks>, using the following payload URL:

```
https://openshift-master.ose3sandbox.com
```

3 Ajout d'un moniteur de santé

Timed Screenshot

Build `testoc`, #1 ✔ Complete. a minute ago [View Log](#)

 `testoc` has containers without health checks, which ensure your application is running correctly. [Add Health Checks](#)

 `testoc`

Deployment Config `testoc` - a few seconds ago

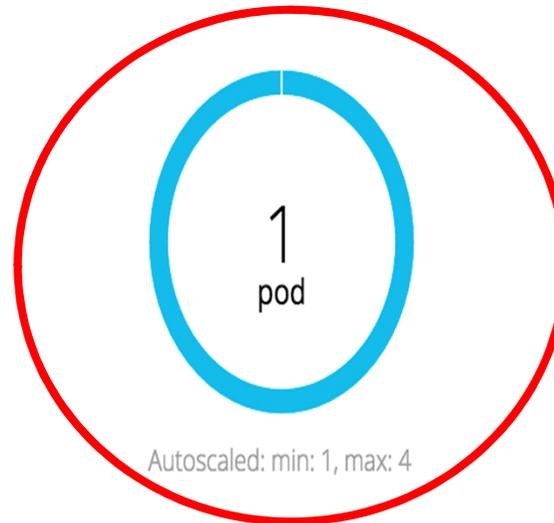
`sy/test3/testoc beb65e4`

#1

MiB Memory

Cores CPU

KiB/s Network



3 Ajout d'un moniteur de santé

syltest3 » Deployments » testoc » Edit Health Checks

Health Checks: testoc

Container health is periodically checked using readiness and liveness probes.

[Learn More](#)

Readiness Probe

A readiness probe checks if the container is ready to handle requests. A failed readiness probe means that a container should not receive any traffic from a proxy, even if it's running.

*** Type**

HTTP GET

Use HTTPS

Path

/

*** Port**

8080

Initial Delay

25

seconds

How long to wait after the container starts before checking its health.

Timeout

2

seconds

How long to wait for the probe to finish. If the time is exceeded, the probe is considered failed.

[Remove Readiness Probe](#)

Liveness Probe

A liveness probe checks if the container is still running. If the liveness probe fails, the container is killed.

[Add Liveness Probe](#)

Pause rollouts for this deployment config

Pausing lets you make changes without triggering a rollout. You can resume rollouts at any time. If unchecked, a new rollout will start on save.

Save

Cancel

4 Utilisation de HTTPERF (Invocation)

- Projet « open source » à l'origine développé par HP.
- Outil CLI pour mesurer la performance d'un serveur Web en générant de la charge.
- Sources sur Github : <https://github.com/httpperf/httpperf>
- Pour l'invoquer :
 - `httpperf --server lesite --port 80 --num-conns 1000 --num-cal 100 --timeout 1 --hog --rate 100`

Options	Description
<code>--num-conn 3000</code>	instructs httpperf to make 3000 connections.
<code>--num-cal 100</code>	instructs httpperf to issue 100 requests per connection.
<code>--rate 200</code>	specifies how many new connections are made every second, 200 in our case.
<code>--timeout 1</code>	instructs httpperf to report as errors any requests that aren't answered within X seconds.
<code>--client=x/2</code>	Specifies that the machine httpperf is running on is client I out of a total of N clients.

- Pour une description des autres options, voir le man httpperf.
- Autre outil « open source » : AB (Apache HTTP server benchmarking tool)
 - `ab -k -n 1000 -c 10 -s 1 testoc-syltest2.apps.ose3sandbox.com/`
 - `man ab`

4 Utilisation de HTTPERF (Résultat)

```
[sylvioie@centaure ~]$ httpperf --server testoc-syltest2.apps.ose3sandbox.com --port 80 --num-conns 1000
--num-cal 100 --timeout 5 --hog --rate 100
httpperf --hog --timeout=5 --client=0/1 --server=testoc-syltest2.apps.ose3sandbox.com --port=80 --uri=/
--rate=100 --send-buffer=4096 --recv-buffer=16384 --num-conns=1000 --num-calls=100
httpperf: warning: open file limit > FD_SETSIZE; limiting max. # of open files to FD_SETSIZE
Maximum connect burst length: 1
```

Total: connections 1000 requests 86448 replies 86238 test-duration 50.064 s

Connection rate: 20.0 conn/s (50.1 ms/conn, <=997 concurrent connections)
Connection time [ms]: min 5785.6 avg 34490.5 max 46530.2 median 37936.5 stddev 9537.4
Connection time [ms]: connect 285.4
Connection length [replies/conn]: 86.584

Request rate: 1726.7 req/s (0.6 ms/req)
Request size [B]: 89.0

Reply rate [replies/s]: min 1491.5 avg 1724.8 max 1863.5 stddev 108.0 (10 samples)
Reply time [ms]: response 182.5 transfer 200.3
Reply size [B]: header 307.0 content 14082.0 footer 0.0 (total 14389.0)
Reply status: 1xx=0 2xx=86238 3xx=0 4xx=0 5xx=0

CPU time [s]: user 1.11 system 48.95 (user 2.2% system 97.8% total 100.0%)
Net I/O: 24355.0 KB/s (199.5*10⁶ bps)

Errors: total 210 client-timo 210 socket-timo 0 connrefused 0 connreset 0
Errors: fd-unavail 0 addrunavail 0 ftab-full 0 other 0

➤ Pour l'interprétation du résultat, voir : <https://github.com/httpperf/httpperf>

4 Utilisation de HTTPERF

```
Connection rate: 43.2 conn/s (23.2 ms/conn, <=927 concurrent connections)
Connection time [ms]: min 288.2 avg 21263.0 max 62833.9 median 5106.5 stddev 24294.0
Connection time [ms]: connect 2.0
Connection length [replies/conn]: 27.246

Request rate: 844.1 req/s (1.2 ms/req)
Request size [B]: 89.0

Reply rate [replies/s]: min 639.5 avg 798.4 max 893.1 stddev 76.1 (13 samples)
Reply time [ms]: response 756.0 transfer 0.3
Reply size [B]: header 307.0 content 14082.0 footer 0.0 (total 14389.0)
Reply status: 1xx=0 2xx=55663 3xx=0 4xx=0 5xx=0

CPU time [s]: user 0.62 system 67.46 (user 0.9% system 97.1% total 98.0%)
Net I/O: 11333.6 KB/s (92.8*10^6 bps)

Errors: total 3000 client-timo 2295 socket-timo 0 connrefused 0 connreset 705
Errors: fd-unavail 0 addrunavail 0 ftab-full 0 other 0

===== TEST FINI =====

sh-4.2# ./testoc.sh

===== TEST DEBUTE =====

httpperf --hog --timeout=1 --client=0/1 --server=testoc-syltest3.apps.ose3sandbox.com --port=80 --uri=/ --rate=200 --send-buffer=4096 --recv-buffer=16384 --num-conns=3000 --num-calls=100
httpperf: warning: open file limit > FD_SETSIZE; limiting max. # of open files to FD_SETSIZE
Maximum connect burst length: 15

Total: connections 1299 requests 104037 replies 103741 test-duration 42.109 s

Connection rate: 30.8 conn/s (32.4 ms/conn, <=1022 concurrent connections)
Connection time [ms]: min 277.5 avg 28245.7 max 38139.0 median 35690.5 stddev 14168.2
Connection time [ms]: connect 2.6
Connection length [replies/conn]: 80.047

Request rate: 2470.7 req/s (0.4 ms/req)
Request size [B]: 89.0

Reply rate [replies/s]: min 2018.4 avg 2461.9 max 2647.2 stddev 199.8 (8 samples)
Reply time [ms]: response 351.7 transfer 0.6
Reply size [B]: header 307.0 content 14082.0 footer 0.0 (total 14389.0)
Reply status: 1xx=0 2xx=103741 3xx=0 4xx=0 5xx=0

CPU time [s]: user 0.70 system 37.64 (user 1.7% system 89.4% total 91.0%)
Net I/O: 34833.2 KB/s (285.4*10^6 bps)

Errors: total 2000 client-timo 50 socket-timo 0 connrefused 0 connreset 249
Errors: fd-unavail 1701 addrunavail 0 ftab-full 0 other 0

===== TEST FINI =====

sh-4.2# ./testoc.sh

===== TEST DEBUTE =====

httpperf --hog --timeout=1 --client=0/1 --server=testoc-syltest3.apps.ose3sandbox.com --port=80 --uri=/ --rate=200 --send-buffer=4096 --recv-buffer=16384 --num-conns=3000 --num-calls=100
httpperf: warning: open file limit > FD_SETSIZE; limiting max. # of open files to FD_SETSIZE
```

Période de questions



Pour toutes questions veuillez écrire à :
sylvoie@gmail.com