NODE.JS GLOBAL

DEPLOY AND TOOLS

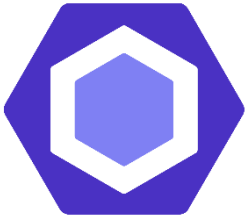
JULY, 2018

AGENDA

- Development tools you may not know, but should:
  - ESLint
  - Nodemon
  - Postman
  - Fiddler
- Installing Node.js on server:
  - PPA
  - NVM
- Application deploy:
  - PaaS (Heroku)
  - Docker
- Demonizing the application:
  - Forever
  - PM2
- Q&A

DEVELOPMENT TOOLS. ESLINT

The primary reason ESLint was created was to allow developers to create their own linting rules.
Code linting is a type of static analysis that is frequently used to find problematic patterns or code that doesn't adhere to certain style guidelines.



```
$ npm install eslint-g
```

```
$ eslint [options] [file|dir|glob]*
```

Can also be integrated to almost all of popular IDEs/Code editors, build tools, Extended via shared configs from well-known companies (Airbnb, Google) and for popular technologies/frameworks (Angular, React, Jasmine)

https://github.com/airbnb/javascript

https://eslint.org/docs/user-guide/getting-started

https://github.com/eslint/eslint

DEVELOPMENT TOOLS. NODEMON

Nodemon is a utility that will monitor for any changes in your source and automatically restart your server. Perfect for development.

```
$ npm install nodemon -g
```

And then instead of using

```
$ node server.js
```

we using

```
$ nodemon server.js
```

https://github.com/remy/nodemon

DEVELOPMENT TOOLS. POSTMAN

A tool that provides efficient way to work with APIs.

Features:

- Composing HTTP requests (headers, auth, data)
- Automated testing with collections of requests
- Mock specific server data
- History logging
- API monitoring

https://www.getpostman.com

DEVELOPMENT TOOLS. FIDDLER

Web debugging proxy tool

Features:

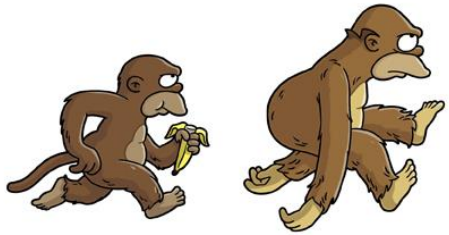
- Request proxying
- Traffic recording (save log with whole requests/responses data)
- Composing HTTP requests (headers, auth, data)
- Performance, security testing
- Customizable free tool

INSTALL NODE.JS ON SERVER

Simple installation

```
$ sudo apt-get install nodejs
```

```
$ sudo apt-get install npm
```



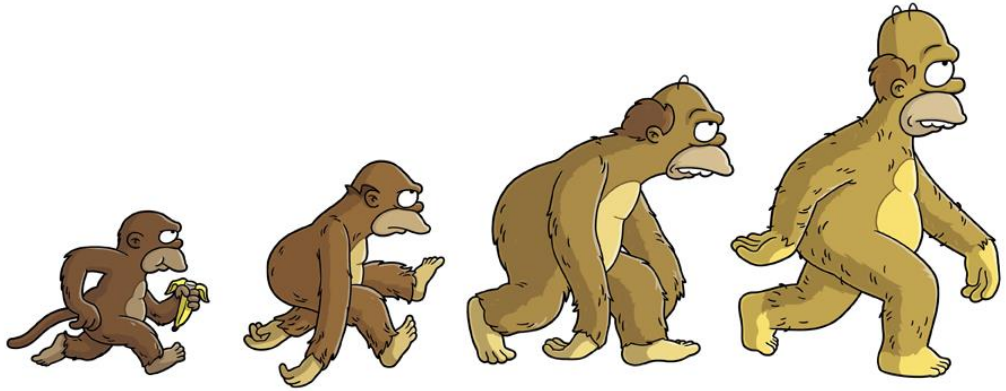
simple installation

INSTALL NODE.JS ON SERVER

PPA installation

```
$ curl https://deb.nodesource.com/setup_8.x | sudo bash
```

```
$ sudo apt-get install nodejs
```



simple installation

PPA

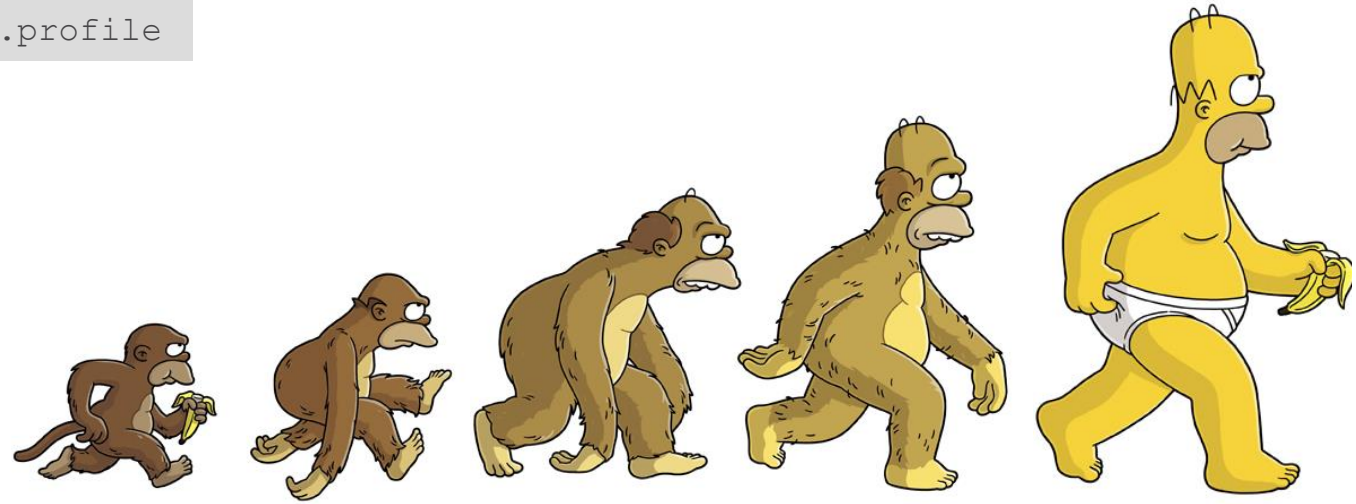
INSTALL NODE.JS ON SERVER

NVM

```
$ sudo apt-get install git build-essential libssl-dev
```

```
$ curl https://raw.githubusercontent.com/creationix/nvm/v0.31.0/install.sh | sudo bash
```

```
$ source ~/.profile
```



simple installation PPA NVM

INSTALL NODE.JS ON SERVER

Using NVM

```
$ nvm ls-remote
```

```
$ nvm install 8.4.0
```

```
$ nvm ls
```

```
$ nvm use 8.4.0
```

```
$ nvm alias default 8.4.0
```

```
$ nvm use default
```

```
$ nvm help
```

NVM

DEPLOY APPLICATION. GIT

The most simpliest way to deploy your application on server is using GIT.

```
$ git clone https://your-repo
```

```
$ cd your-repo-path
```

```
$ npm install
```

```
$ npm run build
```

```
$ npm start
```

DEPLOY APPLICATION. HEROKU

Heroku makes it easy to deploy and scale Node.js applications in the cloud. Run any recent version of Node.js. Deploy apps in seconds utilizing dependency caching. Easily install third-party add-ons like Redis, MongoDB & New Relic.

- Fast start for developer
- Easy to use
- Free

- App starts slow on free account
- All tasties isn't free

DEPLOY APPLICATION. DOCKER

Docker allows you to package an application with all of its dependencies into a standardized unit, called a container, for software development. A container is a stripped-to-basics version of a Linux operating system. An image is software you load into a container.

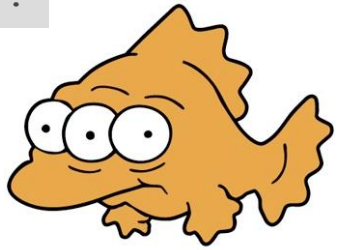
Dockerfile

```
FROM node:carbon
WORKDIR /usr/src/app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 8080
CMD [ "npm", "start" ]
```

```
$ docker build -t [USERNAME]/app-name .
```

```
$ docker login
```

```
$ docker push [USERNAME]/app-name
```



On server

```
$ sudo docker run –p 8080:8080 -d [USERNAME]/app-name
```

DEMONIZING THE APP

Now we install NodeJS and launch our application on server.

It’s time for process managers.

DEMONIZING THE APP: INIT.D

In Unix-based systems, init is the first process started during the boot of the computer system. It is a daemon process that continues running until the system is shut down.

You can implement custom low-level script that will run your node app as a daemon.

- ✚ App will be started/stopped by the system itself
- ▭ You have to restart the app manually or using any side library

https://github.com/chovy/node-startup/blob/master/init.d/node-app

DEMONIZING THE APP: FOREVER

Forever is a simple command-line interface tool for ensuring that a given script runs continuously (forever). Forever's simple interface makes it ideal for running smaller deployments of Node.js apps and scripts.

```
$ npm install forever -g
```

```
$ forever start script.js
```

```
$ forever list
```

```
$ forever stop 1
```

```
$ forever stop script.js
```

```
$ forever stopall
```

https://github.com/foreverjs/forever

DEMONIZING THE APP. FOREVER: DEMO

DEMONIZING THE APP: PM2

PM2 is a production process manager for Node.js applications, that has a built-in load balancer. PM2 allows you to keep applications alive forever and reload them without downtime, and will facilitate common system admin tasks. PM2 also enables you to manage application logging, monitoring, and clustering.

```
$ npm install pm2 -g
```

```
$ pm2 start app.js
```

```
$ pm2 list
```

Name	mode	status	□	cpu	memory
server	fork	online	0	0%	30.4 MB

```
$ pm2 show 0
```

```
$ pm2 stop 0
```

```
$ pm2 restart 0
```

```
$ pm2 delete 0
```

```
$ pm2 startup
```

```
$ pm2 save
```

https://github.com/Unitech/pm2

DEMONIZING THE APP. PM2 : DEMO

DEMONIZING THE APP: PM COMPARISON

Feature	StrongLoop Process Manager	pm2	forever
Run app locally	slc start	pm2 start app.js -name foo	forever start app.js
Restart on failure	Yes	Yes	Yes
OS Startup script support	Yes	Yes	No
Security	HTTP auth and HTTP+SSH	SSH only deploy	No
Set environment vars	Available on install and with slc ctl env-set command	Available as part of ecosystem configuration	No
Log aggregation/rotation	Yes; log file and syslog	Yes; multihost, with rotation. Log file only; no syslog	No

DEMONIZING THE APP: PM COMPARISON

Feature	StrongLoop Process Manager	pm2	forever
Multiple app management	Local/Remote	Local/Remote	Local only
Deploy apps to docker container	Yes	No	No
Remote deploy	Yes	Yes	No
Clustering	Available	Available	No
Profiling	Heap and CPU profiles	No	No
Metrics	CPU, memory, database, NoSQL connectors, many others	CPU, memory, stack traces reported on error output	No

Q&A

NODE.JS GLOBAL

DEPLOY AND TOOLS
BY
SIARHEI MELNIK
MOHAMMADALI GANJI