Last updated: 2015.03.17

CSCI 4140 – Tutorial 8

Deploying Node.js Applications on OpenShift

Matt YIU, Man Tung (mtyiu@cse)

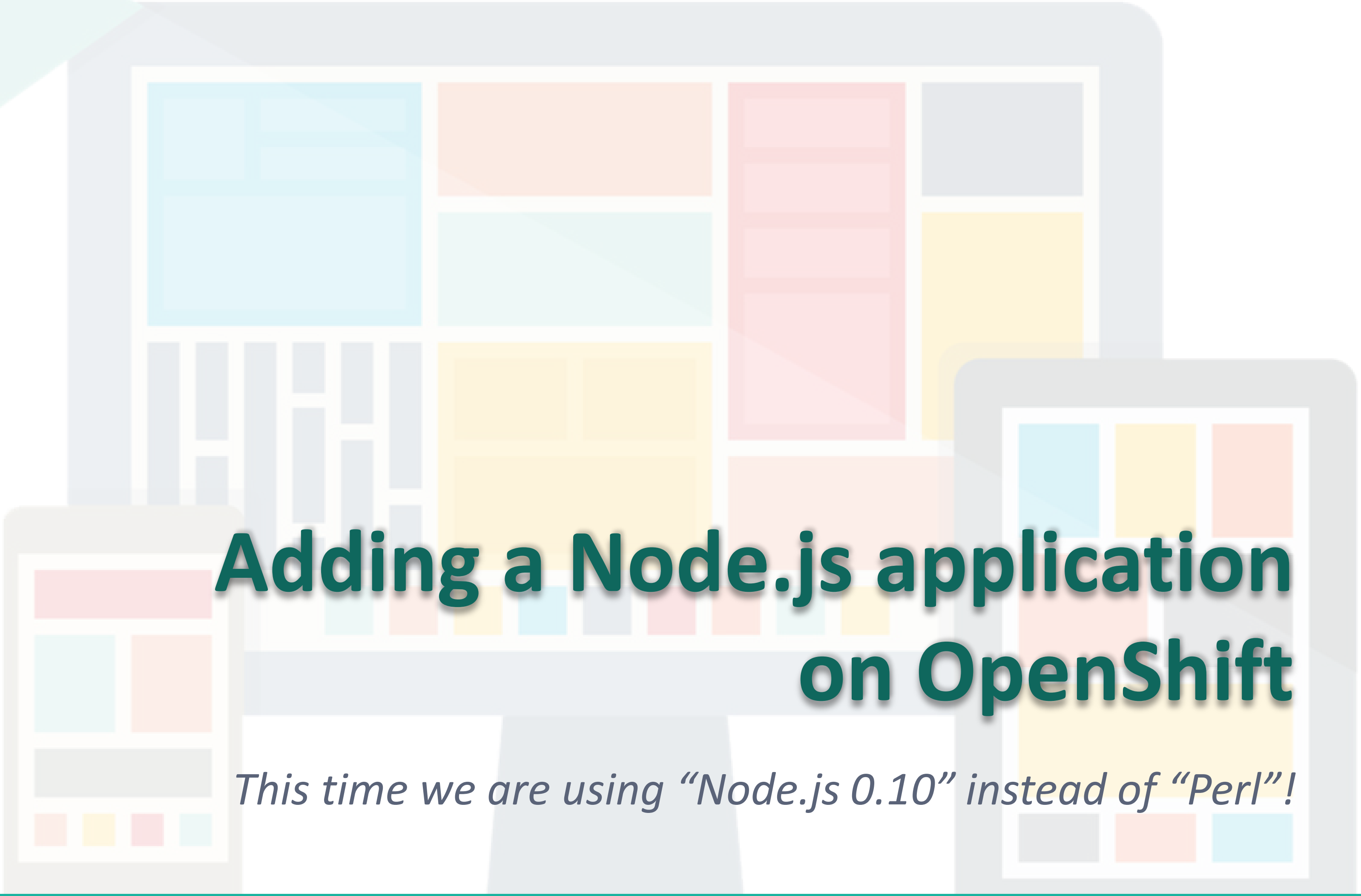
SHB 118

Office Hour: Tuesday, 3-5 pm

2015.03.12

Prerequisite

- We will start with an **Express application**
  - Please follow the instructions on the tutorial slides "**Installing Node.js and Express on [Windows|Linux or Mac]**", pp. 19-21 for creating an application skeleton
  - If you don't use an Express application, you need to figure out where to configure the server's listen IP address and port number
- We will deploy the application on **OpenShift**
  - You should have an account already ☺
  - I assume that you finished all configurations for using OpenShift (e.g., adding SSH keys)

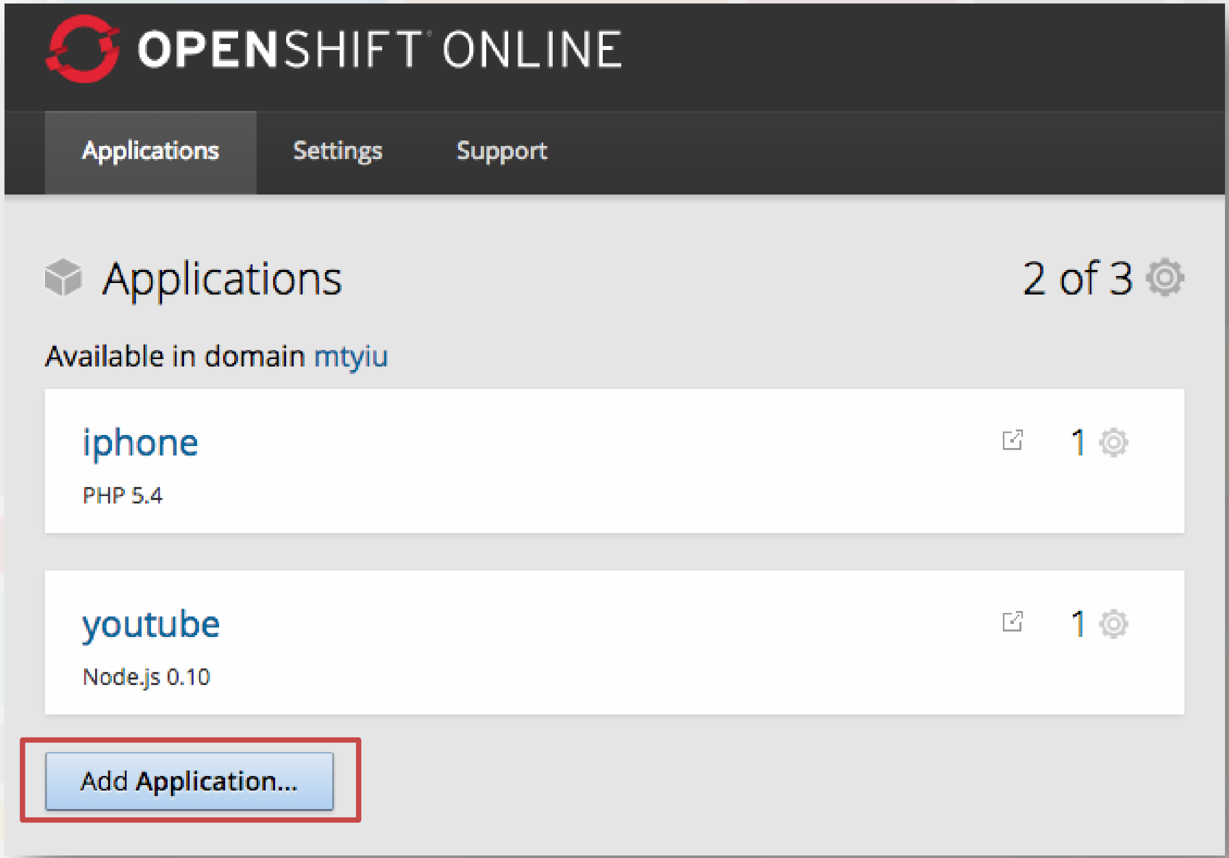


Adding a Node.js application on OpenShift

This time we are using “Node.js 0.10” instead of “Perl”!

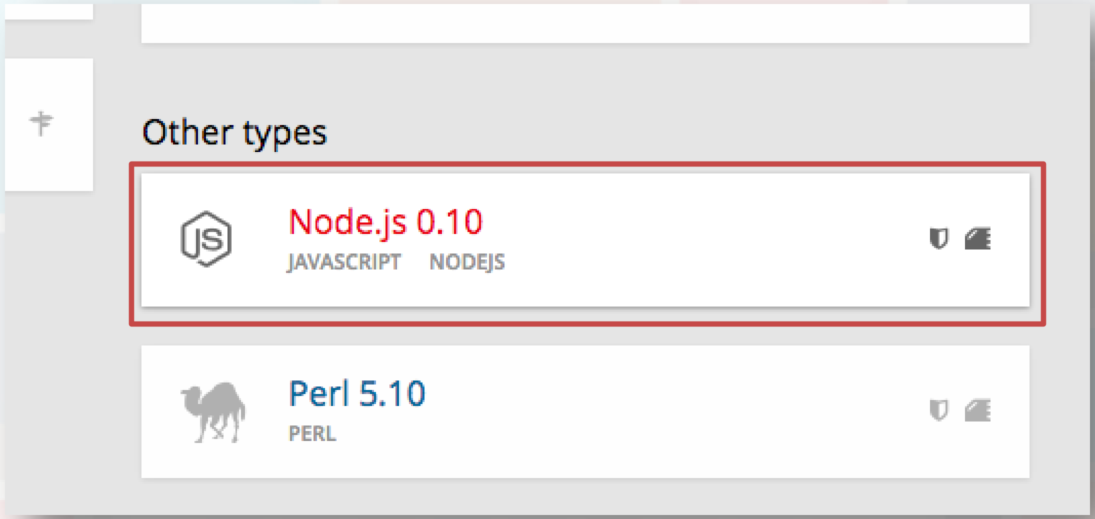
Step 1. Add Application

- Login to your OpenShift console and click "**Add Application…**"



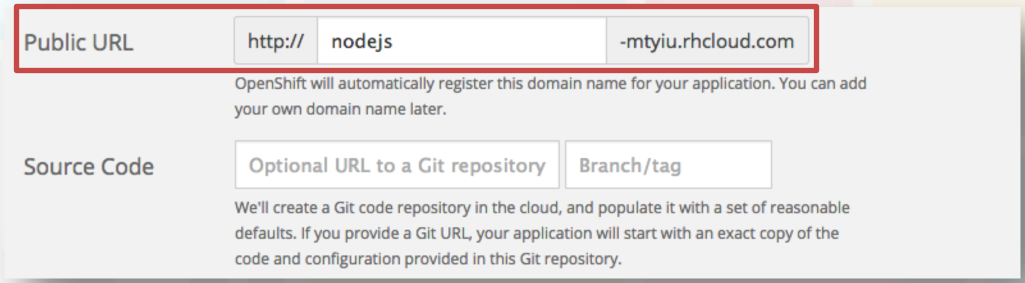
Step 2. Choose a type of application

- Select “**Node.js 0.10**”



Step 3. Configure the application

- Remember to change the **public URL** of your application
- Keep default settings for other configurations
- Click "**Create Application**"

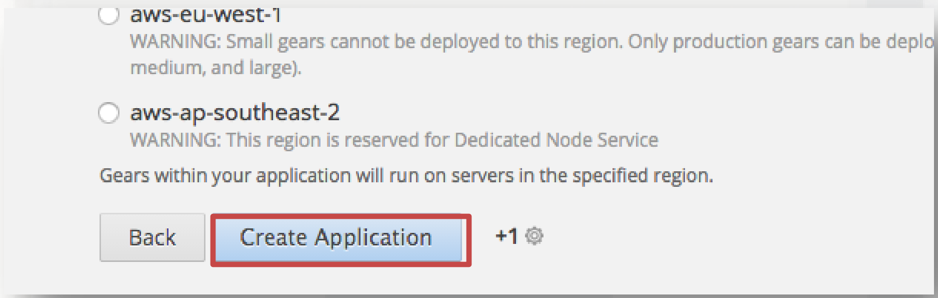


Public URL http:// nodejs -mtyiu.rhcloud.com

OpenShift will automatically register this domain name for your application. You can add your own domain name later.

Source Code Optional URL to a Git repository Branch/tag

We'll create a Git code repository in the cloud, and populate it with a set of reasonable defaults. If you provide a Git URL, your application will start with an exact copy of the code and configuration provided in this Git repository.

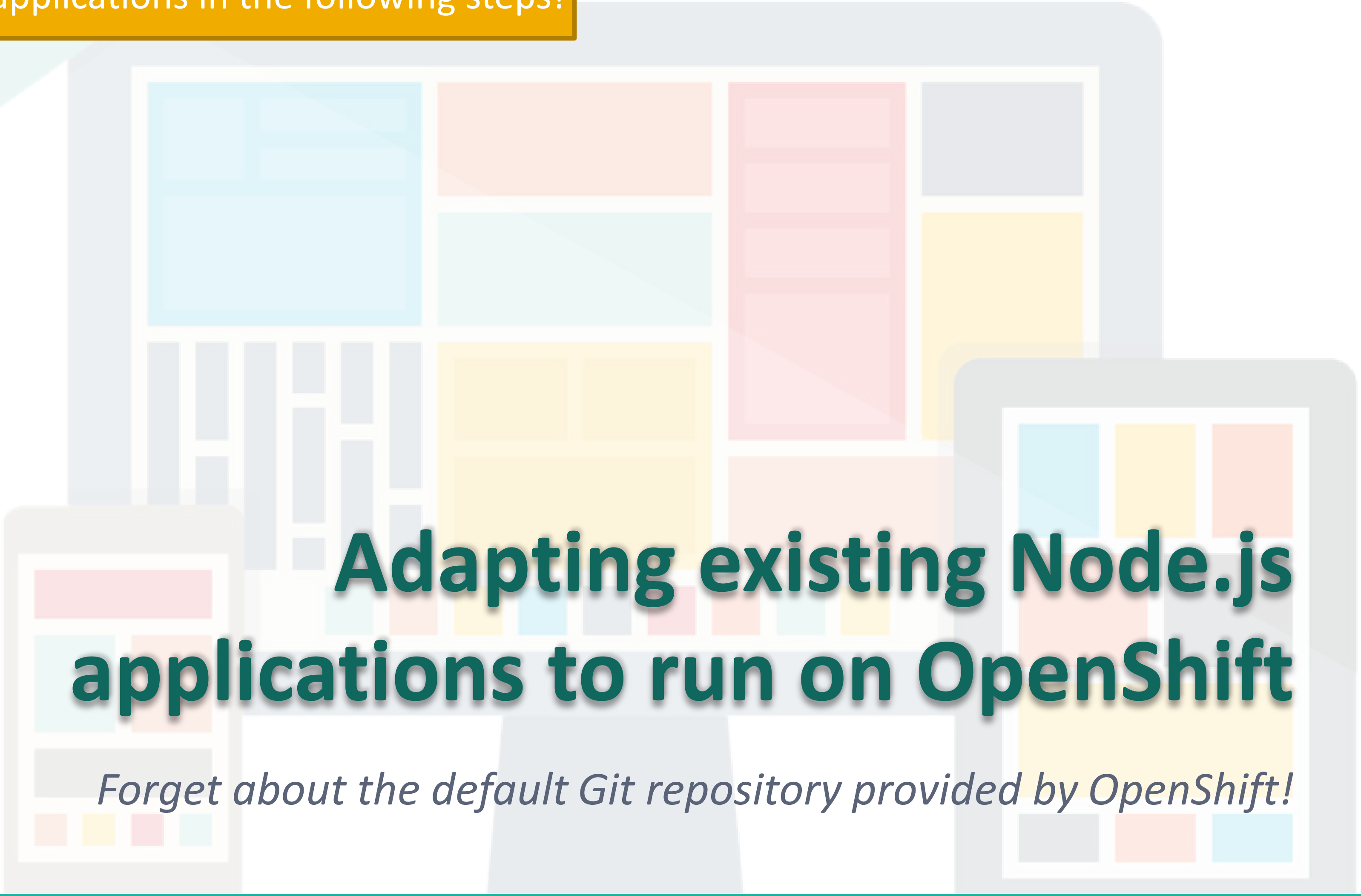


○ aws-eu-west-1
WARNING: Small gears cannot be deployed to this region. Only production gears can be deplo
medium, and large).

○ aws-ap-southeast-2
WARNING: This region is reserved for Dedicated Node Service

Gears within your application will run on servers in the specified region.

Warning: We focus on adapting existing Express applications in the following steps!



Adapting existing Node.js applications to run on OpenShift

Forget about the default Git repository provided by OpenShift!

Updated

Step 1. Include a `package.json` file

- All Node.js applications should include a **`package.json`** file in the **root of their project**
  - Since we are using **Express application generator** to create our application skeleton, this file is automatically generated
- Edit the startup script in **`scripts.start`** and **`main`**
  - I don't like to use the default script (**`bin/www`**)
  - Let's change it to "**`server.js`**"

```
{
  "name": "nodejs-openshift",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node server.js"
  },
  "main": "server.js"
  "dependencies": {
    "express": "~4.10.6",
    "body-parser": "~1.10.1",
    "cookie-parser": "~1.3.3",
    "morgan": "~1.5.1",
    "serve-favicon": "~2.2.0",
    "debug": "~2.1.1",
    "jade": "~1.8.2"
  }
}
```

Sample `package.json` file

Updated

Step 1. Include a package.json file

- Remove all **unused dependencies** to save **deployment time**
  - OpenShift will install ***all*** dependencies listed here during deployment
- Include a **.gitignore** file to exclude the "**node_modules**" directory from the Git repository
  - This saves time for **git push**

```
node_modules
node_modules/*
```

Sample .gitignore file

```
{
  "name": "nodejs-openshift",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node server.js"
  },
  "main": "server.js"
  "dependencies": {
    "express": "~4.10.6"~~,~~
    ~~"body-parser": "~1.10.1",~~
    ~~"cookie-parser": "~1.3.3",~~
    ~~"morgan": "~1.5.1",~~
    ~~"serve-favicon": "~2.2.0",~~
    ~~"debug": "~2.1.1",~~
    ~~"jade": "~1.8.2"~~
  }
}
```

Sample package.json file

Step 2. Edit the startup script (`server.js`)

- OpenShift's Node.js cartridge automatically publishes **server connection information** to your application's environment via the following environment variables:
  - **`OPENSHIFT_NODEJS_PORT`**
  - **`OPENSHIFT_NODEJS_IP`**
- The startup script should read configuration details from the **system environment**
- Now edit the startup script (**`server.js`**)

Step 2. Edit the startup script (server.js)

```
var express = require( 'express' );
var app = express();

var server_port = process.env.OPENSHIFT_NODEJS_PORT || 8000;
var server_ip_address = process.env.OPENSHIFT_NODEJS_IP || '127.0.0.1';

app.get( '/', function ( req, res ) {
        res.send( 'Hello World!' );
} );

var server = app.listen( server_port, server_ip_address, function () {
        var host = server.address().address;
        var port = server.address().port;
        console.log( 'Listening at http://%s:%s', host, port );
} );
```

openshift/server.js

Step 2. Edit the startup script (server.js)

```
var express = require( 'express' );
var app = express();

var server_port = process.env.OPENSHIFT_NODEJS_PORT || 8000;
var server_ip_address = process.env.OPENSHIFT_NODEJS_IP || '127.0.0.1';

app.get(
        r
} );

var server
        v
        v
        console.log( 'Listening at http://%s:%s', host, port );
} );
```

openshift/server.js

Store the IP address and port number in variables.
It first tries to read the environment variables. If they do not exist, use "8000" as the port number and "127.0.0.1" as the IP address.
This is important to ensure your project's portability!
Since your local machine does not have these two environment variables, "127.0.0.1:8000" will be used in your development environment.

Step 2. Edit the startup script (server.js)

```
var express = require( 'express' );
var app = express();

var server_port = process.env.OPENSHIFT_NODEJS_PORT || 8000;
var server_ip_address = process.env.OPENSHIFT_NODEJS_IP || '127.0.0.1';

app.get( '/', function ( req, res ) {
        res.send( 'Hello World!' );
} );

var server = app.listen( server_port, server_ip_address, function () {
        var host = server.address().address;
        var port = server.address().port;
        console.log( 'Listening at http://%s:%s', host, port );
} );
```

Configure the server to listen at "<server_ip_address>:<server_port>".

openshift/server.js

Step 2. Edit the startup script (server.js)

```
var express = require( 'express' );
var app = express();

var server_port = process.env.OPENSHIFT_NODEJS_PORT || 8000;
var server_ip_address = process.env.OPENSHIFT_NODEJS_IP || '127.0.0.1';

app.get( '/', function ( req, res ) {
        res.send( 'Hello World!' );
} );

var server = app.listen( server_port, server_ip_address, function () {
        var host = server.address().address;
        var port = server.address().port;
        console.log( 'Listening at http://%s:%s', host, port );
} );
```

openshift/server.js

Reminder: This application does not involve Socket.IO configuration for simplicity! Follow the instructions on the tutorial notes for using Socket.IO.

Updated

Step 3. Edit the frontend JavaScript

- Since OpenShift's WebSockets support is in preview only, it requires you to specify the **port number**!
- Now in your client page, change the line for establishing the WebSocket connection to the server

```
var socket = io();
```

into

```
var socket = io( 'ws://' + window.location.hostname + ':8000/' );
```

- **Warning:** Do not hard code the URL since we will use the Node.js cartridge prepared by the tutor.
- Let's use **`window.location.hostname`** to get the hostname of the current URL.
- However, the port number needs to be hard-coded as **8000**!

Updated

Step 4. Include `.openshift` directory

- The directory includes all deploy scripts and markers
  - Like what you have done in Assignment 1
- If you need to run a script during deployment, you can follow the instructions on p. 26, Tutorial 1 by Jimmy SINN
  - http://appsrv.cse.cuhk.edu.hk/~ltsinn/csci4140-2015spring/tutorial1_openshift.pdf
- Read Tutorial 4 for more details
  - http://appsrv.cse.cuhk.edu.hk/~ltsinn/csci4140-2015spring/tutorial4.pdf

Step 5. `git commit` and push to OpenShift

- Now back to the root directory of your application
- In case you did not create the Git repository…

```
$ git init
Initialized empty Git repository in
/Users/mtyiu/Development/nodejs-openshift/.git/
$ git add .
```

- Commit your code changes:

```
$ git commit -a -m "<Your commit message>"
```

- Add OpenShift to the remote of the Git repository:

```
$ git remote add origin
ssh://abcdefghijklmnopqrstuvwx@nodejs-
mtyiu.rhcloud.com/~/git/nod
```

Find the remote URL on your OpenShift console.

Step 5. `git commit` and push to OpenShift

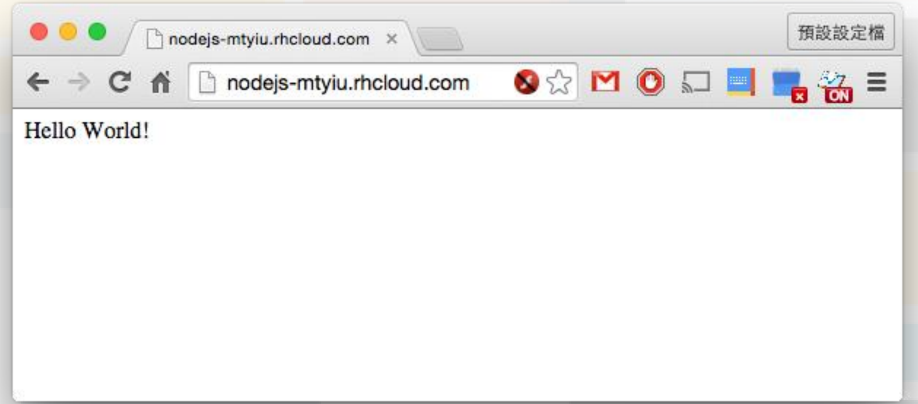
- We are ready to push the code to OpenShift:

```
$ git push -f origin
```

Do you notice the "`-f`" flag? It forces a commit on a remote ref.

Since OpenShift provides a sample Node.js application on the repository, you will not be able to commit your code changes without this flag.

- When it is done, you can visit your website using the public URL you set before:



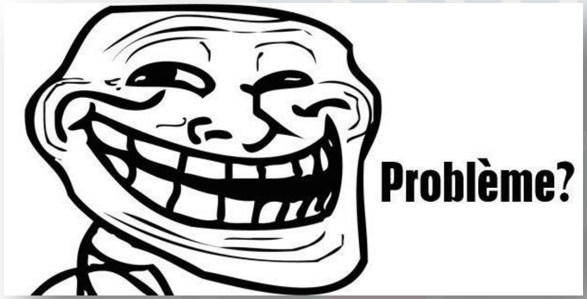
Hello World!

Updated

Working with database?

- **Note:** It is not necessary to use database for Assignment 2!
  - You can save the session IDs inside your Node.js application (e.g., as an array in `server.js`)
- In case you want to use it in your project…
  - Node.js works best with **MongoDB**
    - Read https://blog.openshift.com/run-your-nodejs-projects-on-openshift-in-two-simple-steps/ for more details
  - It may also work with MySQL (though I didn't try)
    - Google yourself ☺
    - Remember to use environment variables to get the MySQL configuration strings

Useful reference:
Node.js Application Hosting – https://developers.openshift.com/en/node-js-overview.html



Problem?

Say "Hi Man Tung" to find me in Facebook group!