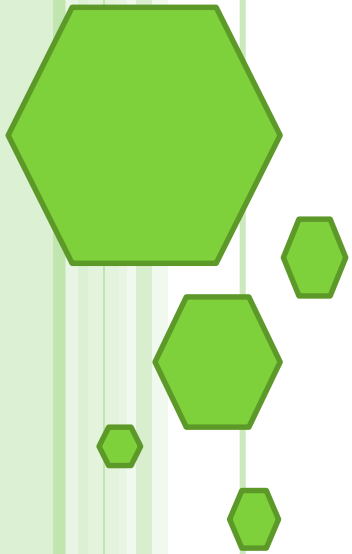
The Server-side JavaScript

Eng. Niveen Nasr El-Den
SD & Gaming CoE.
iTi



Day 3

Universal Module Definition "UMD"

- Build your own module that can work on both client & server side

```
(function umd(modNm, context, definition) {
    if (typeof module !== "undefined" && module.exports) {
        module.exports = definition();
    } else {
        context[modNm] = definition();
    }
})("cart", this, function () {
    class cart {

        constructor() {
            this.items = [];
        }
    }
    cart.prototype.calcTotalCost = function () {

        return this.items.reduce(function (p, c) {
            return p + parseInt(c.itemPrice);
        }, 0);

    }
    cart.prototype.addItem = function (item_nm, item_price) {

        this.items.push({
            itemName: item_nm,
            itemPrice: item_price
        })
    }
    return cart;
})
```

http://git-scm.com/

NPM

- NPM comes bundled with Node.js installation.
- NPM stands for Node Package Manager
- NPM is the online repositories for node.js packages/modules
- It is a package manager for Node.js
- It's a "CLI" command line interface
- Libraries in Node.js are called packages.
- It allow us to install packages/modules, and publish our custom one and share them with other developers

www.npmjs.org

NPM Recommended Commands

- npm install pkg_nm - install a package
- npm install -g pkg_nm - install a package globally
- npm install - install packages listed in package.json
- npm install pkg_nm -- save - install packages and add it to dependencies in package.json file
- npm install pkg_nm --save -dev - install packages and add it to devdependencies in package.json file
- npm uninstall pkg_nm --save –dev
- npm uninstall pkg_nm

NPM Recommended Commands

- npm update pkg_nm - update a package
- npm list – show all packages installed in this application
- npm -g list – show all packages installed globally on your PC
- npm init - initialize a package.json file
- npm adduser
- npm publish
- npm unpublish pkg_nm --force

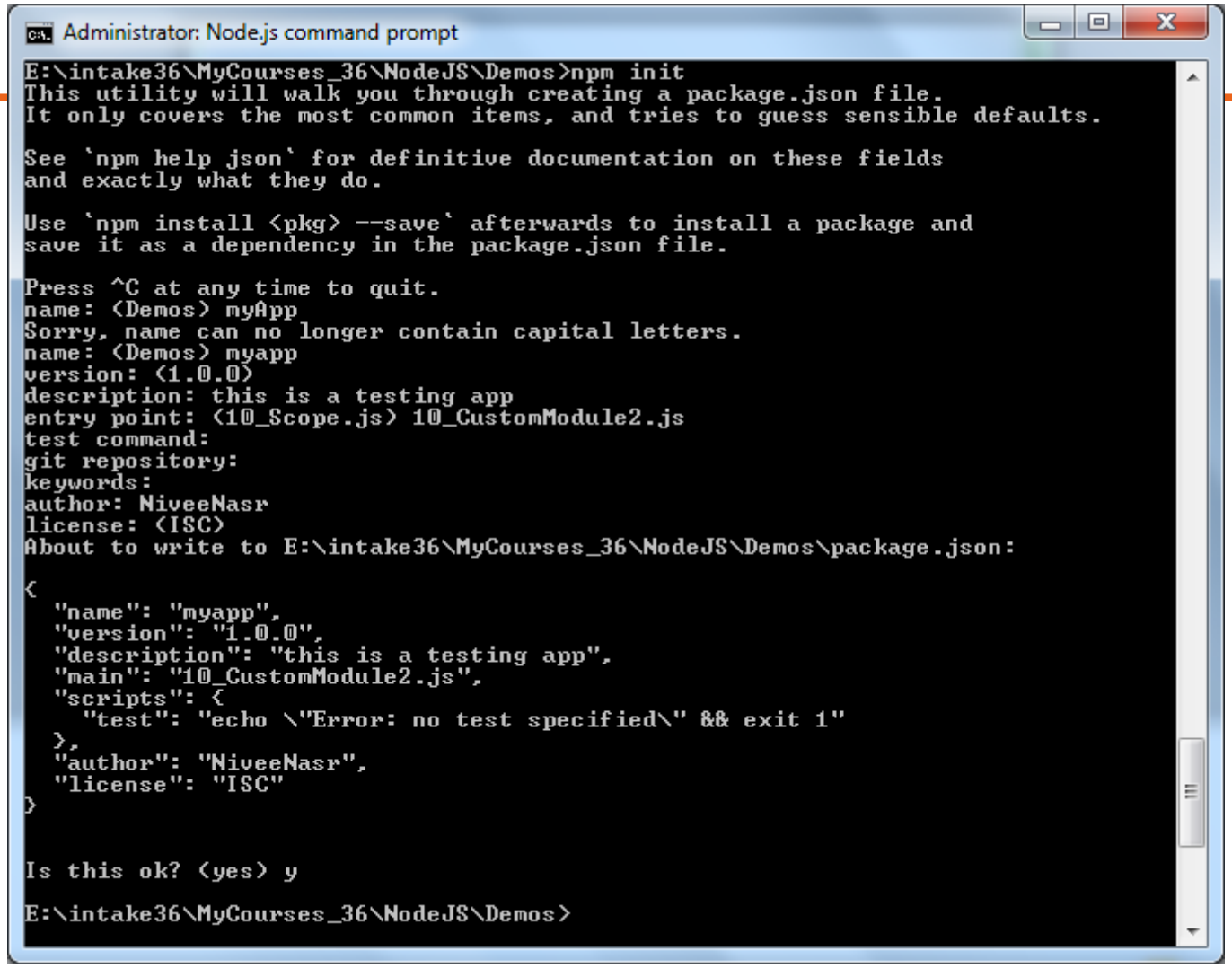
PACKAGE.JSON

- It's a file that contains txt info about project's loaded modules
- It can build or rebuild node-module folder
- Most important fields are name & version.

package.json

```
{
  "name": "myapp",
  "version": "1.0.0",
  "description": "this is a testing app",
  "main": "10_CustomModule2.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "NiveeNasr",
  "license": "ISC"
}
```

http://browsenpm.org/package.json


Administrator: Node.js command prompt
E:\intake36\MyCourses_36\NodeJS\Demos>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.
See `npm help json` for definitive documentation on these fields
and exactly what they do.
Use `npm install <pkg> --save` afterwards to install a package and
save it as a dependency in the package.json file.
Press ^C at any time to quit.
name: (Demos) myApp
Sorry, name can no longer contain capital letters.
name: (Demos) myapp
version: (1.0.0)
description: this is a testing app
entry point: (10_Scope.js) 10_CustomModule2.js
test command:
git repository:
keywords:
author: NiveeNasr
license: (ISC)
About to write to E:\intake36\MyCourses_36\NodeJS\Demos\package.json:
{
"name": "myapp",
"version": "1.0.0",
"description": "this is a testing app",
"main": "10_CustomModule2.js",
"scripts": {
"test": "echo \"Error: no test specified\" && exit 1"
},
"author": "NiveeNasr",
"license": "ISC"
}
Is this ok? (yes) y
E:\intake36\MyCourses_36\NodeJS\Demos>

Semantic Versioning and License

- License – how much you have control over the code & how others can use your code within their applications

spdx.org/licenses

- Version –
  version [Major].[Minor].[Patch]

semver.org

Heavily Used Modules

- Other frameworks that will make it easier using node
  - **Express** – to make things simpler e.g. syntax, DB connections.
  - ~~**Jade**~~, **Ejs** – HTML template system (view engines)
  - **Socket.IO** – to create real-time apps
  - **Nodemon** – to monitor Node.js and push change automatically
  - **Redis** – in memory DB
  - **MongoDB**

Express

- Express is a web application framework for building web apps with node as our backend system.

- express has function names after http verbs
  - We can create routes for http request methods

- Middleware is the essential building blocks of express

Middleware

- Functions show user something goes between the browsers and what we do with data from browsers
- Using of app.use() register a piece of middleware
- Middleware category
  - Third party
  - Custom middleware
  - Built-in
  - Routing function

https://github.com/senchalabs/connect/wiki

- Requests flow move from up to down and move through middleware as they go down

Middleware

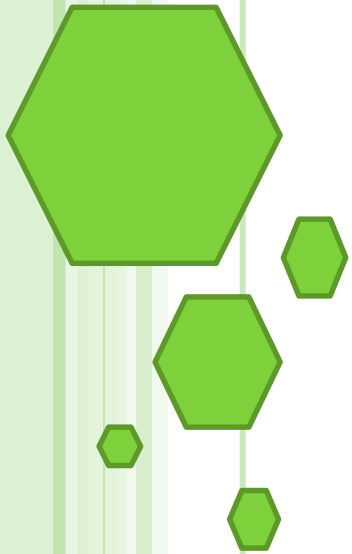
```
//3rd party Middleware
app.use(bodyParser.urlencoded({
  extended: true
}));

//custom Middleware
app.use(function(res,req,next){

    next();
});

//routeFunction Middleware
app.get()

//built-in Middleware
app.use(express.static("./public"));
```



Assignment

Assignments

- Update your module to be UMD and publish it on npm
- Implement task of day 2 using express