Browserify parse l’arbre syntaxique abstrait (AST) des appels à require() et va créer un graphe des dépendances du projet. Il est utilisé à la fois côté serveur que côté navigateur. Pas besoin de bower pour le navigateur et de npm pour le serveur. Il fait du deux en un!

browserify looks at a field called ‘scripts’ in package.json to run task automation

npm install -g browserify => create project and create package.json

Declare verbs in the property scripts to build the tasks as:

build, watch, check, etc…

In package.son:

scripts: {

build: browserify app.js -o build.js,

watch: watchify app.js -o build.js

}

In command line interface:

npm run build

nom run watch…

Declare <script scr=build.js></script> in our index page.

build.js is going to regroup all the modules called by require and manage their dependencies.

In the CLI, we launch: node app.js

And in the browser, we listen to the port with localhost:port to launch our project.

browserify is CommonJS like.

The modules:

- in the web today:

(function(){

var $ = this.jQuery;

this.myExample = function(){};

}());

- in AMD:

define([‘jquery’], function($){return function(){};

});

- in CommonJS:

var $ = require(‘jquery’);

module.exports.myExample = function(){}; // ou exports.myExample = function(){};