

**Tutoriel – Grunt**

**Application sur Finterra & Agrimarket**

Logaviv

30 Juillet 2014

Table des matières

[1. Référence du document 3](#_Toc394481393)

[2. Gestion des versions 4](#_Toc394481394)

[3. Introduction 5](#_Toc394481395)

[4. Installation 6](#_Toc394481396)

[5. Exemples d’utilisations 11](#_Toc394481397)

# Référence du document

**Document**

| Nom du fichier | Grunt-Tutoriel - Application sur Finterra & Agrimarket.docx |
| --- | --- |
| Version | 1.1 |
| Titre | Grunt – Tutoriel – Application sur Finterra & Agrimarket |
| Auteur | Logaviv |
| Date du document | 04/09/14 |
| Date d’impression | 04/09/14 |
| Nombre de pages | 12 |

# Gestion des versions

|  |  |  |  |
| --- | --- | --- | --- |
| **Version de base** | **Date de modification** | **Auteurs** | **Modifications effectuées** |
| 1.0 | 29/07/2014 | P.BEROUCRY | Création du document |
| 1.1 | 03/09/2014 | P.BEROUCRY | Ajout d’exemples d’utilisations |

# Introduction

* 1. GruntJS, kesako au juste ?

Dans un contexte où Logaviv accentue ses projets web en architecture full REST avec notamment les parties clientes codés en JavaScript, nous allons être de plus en plus confrontés à réaliser un certain nombre de tâches dites « répétitifs » comme **corriger les erreurs de syntaxes** ou encore **concaténer nos codes**.

GruntJS va donc être la solution qui va nous sauver la mise ! ☺ . Cet outil d’exécution de commandes en JavaScript regroupe tout un écosystème de composants faisant ce que je viens de citer un peu plus haut.

Pour plus d’information, je vous conseille de consulter les sites suivants :

* [GRUNT – the Javascript Task Runner](http://gruntjs.com/)
* [Disko – gruntjs pour les nuls](http://www.disko.fr/reflexions/technique/gruntjs-pour-les-nuls/)

# Installation

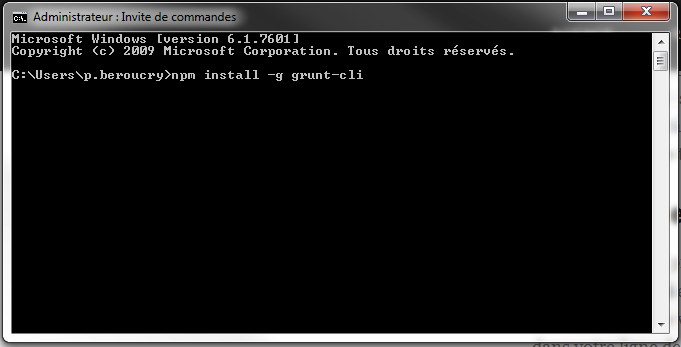
* 1. Prérequis

Afin de pouvoir installer GruntJS, **il est impératif que NodeJS soit installé sur votre ordinateur**. Si tel n’est pas le cas, veuillez cliquer sur le lien suivant :

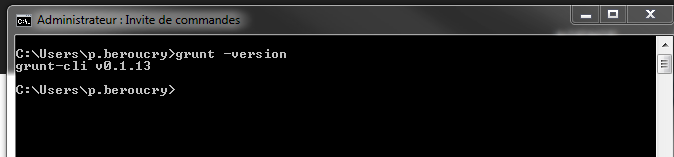
* [NodeJS](http://nodejs.org/)
  1. Installation

L’installation de grunt se fait par l’intermédiaire du gestionnaire de paquet de Node ([**npmjs**](https://www.npmjs.org/)).

Tout d’abord, aller sur votre invite de commande et taper la commande suivante :



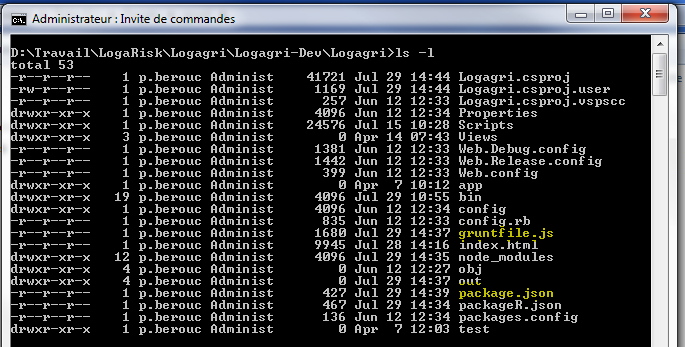
Pour vérifier si l’installation s’est bien déroulée, taper la commande suivante :



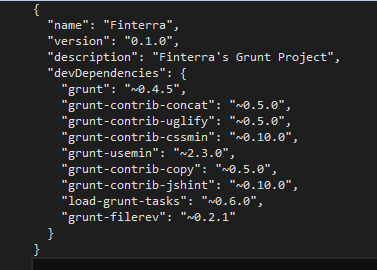
* 1. Package.json & Gruntfile.js

A ce stade du tutoriel, nous partons du principe que l’installation s’est déroulé parfaitement.

Pour commencer à utiliser grunt, il faut impérativement créer un fichier **package.json** et un fichier **Gruntfile.js** rassemblant la description ainsi que les dépendances d’un projet grunt. Dans le cas de Finterra, ce fichier doit se situer à la racine du projet Logagri :



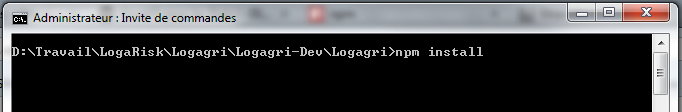
Description du fichier package.json :



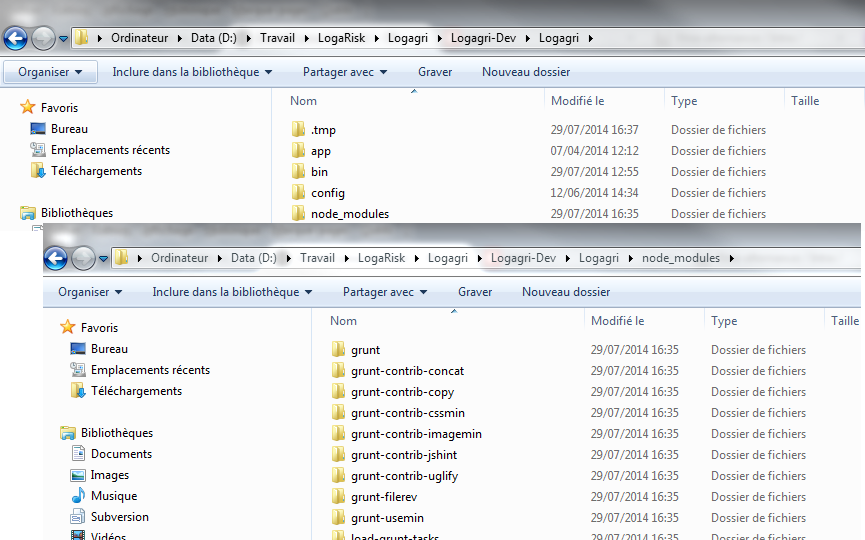
Comme énoncé précédemment, ce fichier va permettre de d’installer l’ensemble des plugins que nous pourrions avoir besoin dans notre projet Finterra pour l’utilisation de grunt. A titre d’information, ce fichier est présent sur le projet, vous n’aurez pas à le créer. Pour plus d’information sur le principe des packages :

* [Nodejitsu – package dependencies done right](http://blog.nodejitsu.com/package-dependencies-done-right/)

Une fois le fichier présent sur le projet, placez-vous dans le dossier Logagri du projet finterra. Veuillez ensuite taper la commande suivante :



Si la commande s’est bien déroulé, un dossier « nodes\_modules » est désormais présent dans le dossier du projet avec l’ensemble des packages grunt de notre projet :



Description du fichier gruntfile.js :

module.exports = function (grunt) {

    require('load-grunt-tasks')(grunt);

    grunt.initConfig({

        jshint: {

            options: {

                sub: true,

            },

            all: ['app/js/\*\*/\*.js']

        },

    });

    grunt.registerTask('default', ['concat']);

}

Ce fichier va contenir l’ensemble des tâches à exécuter par grunt (correction de code, concaténation du code, mise en production automatique). Nous allons détailler ce fichier au fur et à mesure des exemples d’utilisations.

# Exemples d’utilisations

* 1. Jshint – La correction de syntaxe

Dans le fichier gruntfile.js du projet Finterra, vous devez avoir ceci :

module.exports = function (grunt) {

    require('load-grunt-tasks')(grunt);

    grunt.initConfig({

        jshint: {

            options: {

                sub: true,

            },

            all: ['app/js/\*\*/\*.js']

        },

        uglify: {

            options: {

                mangle: false

            }

        },

        useminPrepare: {

            html: 'index.html',

            options: {

                dest: 'out'

            }

        },

        copy: {

            html: {

                files: [{

                    src: ["index.html"],

                    dest: "out/"

                }

                ]

            }

        },

        filerev: {

            options: {

                encoding: 'utf8',

                algorithm: 'md5',

                length: 8

            },

            images: {

                src: 'out/app/js/min.js'

            }

        },

        usemin: {

            html: 'out/index.html',

        },

        imagemin: {                          // Task

            dynamic: {                         // Another target

                files: [{

                    expand: true,                  // Enable dynamic expansion

                    cwd: 'app/data',                   // Src matches are relative to this path

                    src: ['\*\*/\*.{png,jpg,gif}'],   // Actual patterns to match

                    dest: 'app/min'                  // Destination path prefix

                }]

            }

        }

    });

    grunt.registerTask('default', ['useminPrepare', 'copy', 'concat', 'uglify', 'cssmin','filerev', 'usemin']);

}

Nous allons nous intéresser ici au plugin [jshint](http://www.jshint.com/) :

  jshint: {

            options: {

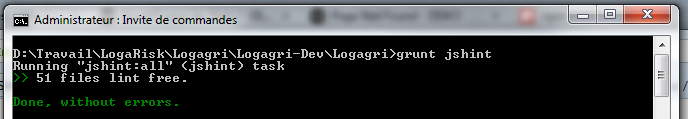
                sub: true,

            },

            all: ['app/js/\*\*/\*.js']

        }

Ce composant va permettre de vérifier la qualité de l’ensemble des fichiers JavaScript de notre fichier :



Afin de comprendre l’ensemble des options et des configurations possibles jshint, je vous invite à consulter le githut suivant => [grunt-contribu-jshint](https://github.com/gruntjs/grunt-contrib-jshint).

* 1. copy – Le déplacement de fichier

 copy: {

            html: {

                files: [{

                    src: ["index.html"],

                    dest: "out/"

                    }, {

                        src: ["app/partials/\*.html"],

                        dest: "out/"

                    },

                    {

                        src: ["app/partials/\*\*/\*.html"],

                        dest: "out/"

                    },

                    {

                        src: ["app/data/\*\*/\*"],

                        dest: "out/"

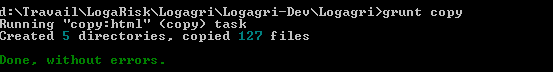
                    }

                ]

            }

        },

Le composant [copy](https://github.com/gruntjs/grunt-contrib-copy) , comme son l’indique, permet de copier des fichiers. Généralement il s’agit de fichier qui n’est pas concernés par quelconques optimisations. Ainsi, lorsque nous tapons la commande « grunt copy », nous obtenons le résultat suivant :



* 1. uglify, usemin & css – concaténation, compressions et minification de fichier.

Le fichier gruntfile.js a été configuré de la façon suivante :

* uglify
  + l’option de mangle permet d’activer la minification des noms de variables en un caractère. Dans le finterra & agrimarket, cette option est désactivée : seule la concaténation et la compression des fichiers d’un projet angular est possible.

        uglify: {

            options: {

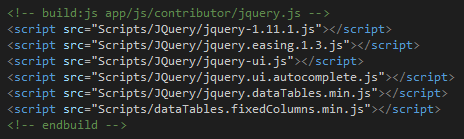
                mangle: false

            }

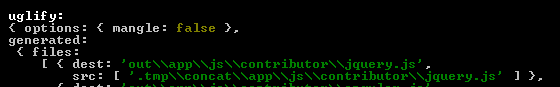
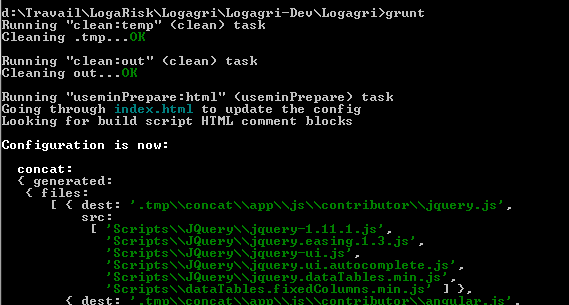
        },

* usemin
  + Va prendre en compte des commentaires des fichiers cibles à concaténer et à ministifier. Concrètement cela donne :

Fichier index.html



Exécution de la commande grunt



Ainsi, lorsque l’ensemble des tâches est finalisé, l’ensemble des scripts de JQuery dans un seul et unique fichier :

