GRUNT build tool for Nodejs

Wednesday, February 25, 2015

2:14 PM

**Installation:** npm install -g grunt-cli

Task API:

each tasks will have these methods: <http://gruntjs.com/inside-tasks>

this.async : this informs grunt the task is asynchronous

this.requires: if one task depends on other then to force grunt to fail by setting it to true, to inform the task on which it depends did not run, this.requires(tasksList) the inside task can be a list of tasks.

this.requiresConfig: fail if the required config property is missing.

this.name: name of the task

this.nameArgs: name and command line arguments separated by colon

this.args: gives the arguments provided while running grunt command line

Grunt.config: <http://gruntjs.com/api/grunt.config#grunt.config.init>

grunt.initConfig--- grunt.config.init these two functions are used for initializing the configurations in grunt

grunt.config.init(configObject)

grunt.config.init({

jshint: {

all: ['lib/\*.js', 'test/\*.js', 'Gruntfile.js']

}

});

Mostly tasks/goals are written in the

Grunt.config: this function is used for accessing the configuration variables, if two arguments are passed set method is called else get method is called.

Grunt.config.get: gets value of the property from config.

grunt.config.set: has two arguments key and value. grunt.config.set(prop, value)

grunt.config.escape: Escape . dots in the given propString. This should be used for property names that contain dots. grunt.config.escape(propString)

grunt.config.merge: Recursively merges properties of the specified configObject into the current project configuration. grunt.config.merge(configObject)

grunt.config.requires:

Fail the current task if one or more required config properties is missing, null or undefined. One or more string or array config properties may be specified.

grunt.config.requires(prop [, prop [, ...]])

Creating tasks:

grunt.registerTask::: grunt.task.registerTask defining task alias for a list of tasks.

grunt.task.registerTask(taskName, taskList)

task.registerTask('default', ['jshint', 'qunit', 'concat', 'uglify']);

task.registerTask('dist', ['concat:dist', 'uglify:dist']);

Function tack:

grunt.task.registerTask(taskName, description, taskFunction)

grunt.task.registerTask('foo', 'A sample task that logs stuff.', function(arg1, arg2) {

if (arguments.length === 0) {

grunt.log.writeln(this.name + ", no args");

} else {

grunt.log.writeln(this.name + ", " + arg1 + " " + arg2);

}

});

Multitask

grunt.task.registerMultiTask(taskName, description, taskFunction)

grunt.initConfig({

log: {

foo: [1, 2, 3],

bar: 'hello world',

baz: false

}

});

grunt.task.registerMultiTask('log', 'Log stuff.', function() {

grunt.log.writeln(this.target + ': ' + this.data);

});

Create your own plugin

1. **Install**[**grunt-init**](https://github.com/gruntjs/grunt-init)**with npm install -g grunt-init**
2. **Install the gruntplugin template withgit clone git://github.com/gruntjs/grunt-init-gruntplugin.git ~/.grunt-init/gruntplugin(%USERPROFILE%\.grunt-init\gruntplugin on Windows).**
3. **Run grunt-init gruntplugin in an empty directory.**
4. **Run npm install to prepare the development environment.**
5. **Author your plugin.**
6. **Run npm publish to publish the Grunt plugin to npm!**