# http://www.talkingdotnet.com/grunt-js-interview-questions/

# Grunt js interview questions for beginners

[December 16, 2015](http://www.talkingdotnet.com/grunt-js-interview-questions/)  [Talking Dotnet](http://www.talkingdotnet.com/author/talkingdotnet/)  [Grunt](http://www.talkingdotnet.com/grunt/), [Interview Questions](http://www.talkingdotnet.com/interview-questions/)

[Grunt](http://gruntjs.com/) is gaining popularity and it has become part of every web application these days. In this post, find list of latest and updated grunt js interview questions and their answers for freshers as well as experienced users. These interview questions will help you to prepare for the interviews, for quick revision and provide strength to your technical skills.

**Grunt js interview questions**

**Recommended Reading**

[Gulp js Interview questions](http://www.talkingdotnet.com/gulp-js-interview-questions/)  
[TypeScript Interview questions](http://www.talkingdotnet.com/typescript-interview-questions/)

**Q1. What is Grunt?**

**Ans.** [Grunt](http://gruntjs.com/) is a JavaScript task runner, which can automate tasks like minification, compilation, unit testing, checking js errors. Once configured, one doesn’t have to worry about these tasks.

**Q2. Why to use Grunt?**

**Ans.** [Grunt](http://gruntjs.com/) has become very popular and has tons of plugins to choose from. These plugins are great assets for any app to automate various things with minimum efforts.

**Q3. How do you install Grunt?**

**Ans.** Grunt and Grunt plugins are installed and managed via npm, the Node.js package manager. To install grunt, first ensure the npm is installed properly. And then run following command.

|  |  |
| --- | --- |
| 1 | npm install grunt --save-dev |

Please note, --save-dev is optional. You will learn more about this in coming questions.

**Q4. How do you setup/configure Grunt?**

**Ans.** Once installed, you need to add 2 files to setup Grunt.  
1. **[package.json](http://www.talkingdotnet.com/various-configuration-json-files-in-asp-net-5/):** This file is used by npm to store metadata for projects published as npm modules. So basically, there will be list of all Grunt plugins, along with grunt which your project is using.  
2. **Gruntfile:** This file is named Gruntfile.js and is used to configure or define tasks and load Grunt plugins.

**Q5. What is --save-dev option while installing the grunt?**

**Ans.** As mentioned in previous answer that “[package.json](http://www.talkingdotnet.com/various-configuration-json-files-in-asp-net-5/)” file holds the metadata for grunt plugins. So when grunt is installed using --save-dev option, the metadata is added to package.json. So you don’t have to add it manually. And this is how your package.json will look like,

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | {    "name": "my-project-name",    "version": "0.1.0",    "devDependencies": {      "grunt": "~0.4.5",      "grunt-contrib-jshint": "~0.10.0",      "grunt-contrib-nodeunit": "~0.4.1",      "grunt-contrib-uglify": "~0.5.0"    }  } |

**Q6. What is the difference between --save and --save-dev?**

**Ans.** Before will look at the difference, it is important to understand the difference between dependencies and devDependencies.  
devDependencies are for the development-related scripts, e.g. unit testing, packaging scripts, documentation generation, etc. where dependencies are required for production use, and assumed required for dev as well. As for example, you can include some plugin which you require during development like (for debugging or unit testing) but you don’t need them on production.  
So --save adds packages under dependencies and --save-dev adds under devdependencies section.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | {    "name": "my-project-name",    "version": "0.1.0",    "dependencies": {      "grunt": "~0.4.5",      "grunt-contrib-nodeunit": "~0.4.1",      "grunt-contrib-uglify": "~0.5.0"    }    "devDependencies": {      "grunt": "~0.4.5",      "grunt-contrib-jshint": "~0.10.0",    }  } |

**Q7. What does ~ (tilde) sign means in package.json?**

**Ans.** In the simplest terms, the tilde matches the most recent minor version (the middle number). ~1.2.3 will match all 1.2.x versions but will miss 1.3.0. The caret, on the other hand, is more relaxed. It will update you to the most recent major version (the first number). ^1.2.3 will match any 1.x.x release including 1.3.0, but will hold off on 2.0.0.  
You can also define the exact version number that you wish to use like “1.3.5” or to always use latest version, simply use latest or \*.

**Q8. What are Grunt modules/plugins?**

**Ans.** Grunt modules are distributed through Node’s NPM directory. Normally, they are prefixed with grunt- and official grunt plugins are prefixed with grunt-contrib. Example: grunt-contrib-uglify. You can get list of all grunt plugins [here](http://gruntjs.com/plugins).

**Q9. How do you install a Grunt plugin?**

**Ans.** The syntax remains name but the only thing which changes is module/plugin name.

|  |  |
| --- | --- |
| 1 | npm install <module> --save-dev |

For example, to install uglify plugin,

|  |  |
| --- | --- |
| 1 | npm install grunt-contrib-uglify --save-dev |

By default, it always installs the latest version available. But if you wish to install specific version then same you can include in the command.

|  |  |
| --- | --- |
| 1 | npm install <module>@version --save-dev |

**Q10. How do you uninstall Grunt?**

**Ans.** Run following command to uninstall grunt.

|  |  |
| --- | --- |
| 1 | npm uninstall grunt |

And if you wish to remove it from package.json, then use --save-dev option.

|  |  |
| --- | --- |
| 1 | npm uninstall --save-dev grunt |

**Q11. What is Grunt-cli?**

**Ans.** Grunt cli is command line interface to run grunt commands. In other words, it’s a tool to access Grunt from command line anywhere in the system. To install, grunt –cli execute following command

|  |  |
| --- | --- |
| 1 | npm install -g grunt-cli |

This will put the grunt command in your system path, allowing it to be run from any directory. **Please note, that installing grunt-cli, doesn’t install grunt on your system.**

-g means global which means it adds to PATH variables of the system so that you can run grunt from anywhere (without going to specific folder on command prompt).

The reason for having two components is to ensure that we can run different grunt versions side-by-side (i.e. legacy versions in older projects). Hence it is recommended to install grunt-cli globally while grunt should be installed on a per-project basis.

**Q12. How does Gruntfile.js uses package.json?**

**Ans.** Task configuration is specified in your Gruntfile via the grunt.initConfig method. Inside of grunt.initConfig(), we read the information from package.json and saved it to a pkg property. With this, we can use the attributes from our package.json file. We can call the name of our project using pkg.name and the version with pkg.version.

|  |  |
| --- | --- |
| 1  2  3  4  5 | module.exports = function(grunt) {    grunt.initConfig({      pkg: grunt.file.readJSON('package.json')    });  }; |

**Q13. Where do you define configuration of grunt plugin?**

**Ans.** Grunt plugin configuration needs to be defined within grunt.initConfig method. See below sample code.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | module.exports = function(grunt) {    grunt.initConfig({      pkg: grunt.file.readJSON('package.json'),      concat: {      // concat task configuration goes here.      },     uglify: {      // uglify task configuration goes here.     }    });  }; |

Above code configures concat and uglify tasks.

**Q14. Can you override default settings of a plugin? If yes, then how?**

**Ans.** Yes, we can override. Inside a task configuration, an options property may be specified to override built-in defaults. In addition, each target may have an options property which is specific to that target.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | module.exports = function(grunt) {    grunt.initConfig({      pkg: grunt.file.readJSON('package.json'),      concat: {       options: {          separator: ';'        }      }    });  }; |

**Q15. What is a task in Grunt?**

**Ans.** Tasks are grunt’s bread and butter. Every time Grunt is run, you specify one or more tasks to run, which tells Grunt what you’d like it to do. See below code.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | grunt.initConfig({    concat: {      development: {        // concat task "development" target options and files go here.      },      production: {        // concat task "production" target options and files go here.      },    },    uglify: {      development: {        // uglify task "development" target options and files go here.      },    },  }); |

Here, there are 2 tasks defined concat and uglify. And for each task, we defined targets. For concat, there are 2 targets “development” and “production” and only “development” for uglify. Creating target allows us to define separate settings for different objectives. Here, we can have different option for development and production version. It’s not compulsory to define a target.

Target’s can also have their own option parameters which will override the settings of options defined for the task.

**Q16. How do you execute grunt task?**

**Ans.** You can execute the grunt task from command line. Remember, we have grunt command line interface. To execute contact module with grunt for all the tasks.

|  |  |
| --- | --- |
| 1 | grunt concat |

To execute task with particular target

|  |  |
| --- | --- |
| 1 | grunt concat:development |

**Q17. How do we load grunt plugins in Gruntfile.js?**

**Ans.** grunt.loadNpmTasks() is used for loading grunt plugins. Before loading, please ensure that these plugins are already installed via npm.

|  |  |
| --- | --- |
| 1 | grunt.loadNpmTasks('grunt-contrib-uglify'); |

**Q18. How to run multiple tasks together in grunt?**

**Ans.** Every grunt module may have defined a task. But running them individually can make things difficult for developers. What is one wants to run grunt uglify and grunt jshint together?

Using grunt.registerTask(taskName, [description, ] taskList) we can group all the tasks under one roof. Task name can be anything of your choice, description is optional and task list is array of module tasks which you wish to execute. For example,

|  |  |
| --- | --- |
| 1 | grunt.registerTask('development', ['jshint:development', 'concat:development', 'uglify:development']); |

Above code creates a task named “development” and is asked to execute development target of “jshint”, “concat” and “uglify” packages. Similarly, you can register another task for production version.

|  |  |
| --- | --- |
| 1 | grunt.registerTask('production', ['jshint:production', 'concat:production', 'uglify:production']); |

Remember, you always need to define a default task.

|  |  |
| --- | --- |
| 1 | grunt.registerTask('default', ['jshint', 'uglify', 'concat']); |

Now, when you enter grunt on command prompt, it will execute the default task.

**Q19. How to get a stack trace when an error occurs?**

**Ans.** Use the --stack option to see stack traces. Such as grunt task –stack.

**Q20. Which are the most used grunt plugins?**

Ans. Though there are tons of plugins which you can use, but below are the most used.

* [watch](https://www.npmjs.com/package/grunt-contrib-watch): Run predefined tasks whenever watched file patterns are added, changed or deleted.
* [jshint](https://www.npmjs.com/package/grunt-contrib-jshint): Validate files with JSHint
* [uglify](https://www.npmjs.com/package/grunt-contrib-uglify): Minify files with UglifyJS
* [concat](https://www.npmjs.com/package/grunt-contrib-concat): Concatenate files.
* [cssmin](https://www.npmjs.com/package/grunt-contrib-cssmin): Minify CSS
* [less](https://www.npmjs.com/package/grunt-contrib-less): Compile LESS files to CSS.

Below is a sample gruntfile.js for your reference.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41 | module.exports = function(grunt) {    grunt.initConfig({      pkg: grunt.file.readJSON('package.json'),      concat: {        options: {          separator: ';'        },        dist: {          src: ['src/\*\*/\*.js'],          dest: 'dist/<%= pkg.name %>.js'        }      },      uglify: {        options: {          banner: '/\*! <%= pkg.name %> <%= grunt.template.today("dd-mm-yyyy") %>  \*/\n'        },        dist: {          files: {            'dist/<%= pkg.name %>.min.js': ['<%= concat.dist.dest %>']          }        }      },      jshint: {        files: ['Gruntfile.js', 'src/\*\*/\*.js', 'test/\*\*/\*.js'],        options: {          // options here to override JSHint defaults          globals: {            jQuery: true,            console: true,            module: true,            document: true          }        }      }    });    grunt.loadNpmTasks('grunt-contrib-uglify');    grunt.loadNpmTasks('grunt-contrib-jshint');    grunt.loadNpmTasks('grunt-contrib-concat');    grunt.registerTask('test', ['jshint']);    grunt.registerTask('default', ['jshint', 'concat', 'uglify']);  }; |
|  |  |

That’s all folk. If you have something to add in this list, please mention in comments section. Meanwhile, share this in your network and spread this to help others.