Web Tools: Task Runners: Grunt

Jogesh K. Muppala

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

香港科技大學
THE HONG KONG UNIVERSITY OF
SCIENCE AND TECHNOLOGY

Grunt

- Grunt: Task runner based on configuration of tasks
- Installing Grunt
  - Grunt Command Line Interface (install globally)
    - Allows for different Grunt versions in different projects

  npm install –g grunt-cli (use sudo on OSX, Linux)

- Install Grunt locally

  npm install grunt –save-dev

    - need to setup *package.json* file

Configuring Grunt

- Configuration in *Gruntfile.js*

```
module.exports = function(grunt) {
        // do requires here
        require('jit-grunt')(grunt);

        // do grunt task configurations here
        grunt.initConfig({

        });

        // register tasks here
        grunt.registerTask('build', [ 'jshint' ]);
        grunt.registerTask('default',['build']);
};
```

File Globbing Patterns

- Grunt uses filename expansion (globbing)
  - * matches any number of characters, but not /
  - ? Matches a single character, but not /
  - ** matches any number of characters, including /, as long as it's the only thing in a path part
  - {} allows for comma-separated list of "or" expressions
  - ! At the beginning of a pattern will negate the match
- We'll see examples of these in the following slides

JSHint

- Install:

  npm install grunt-contrib-jshint --save-dev

  npm install jshint-stylish --save-dev

- Configuration:

```
jshint: {
        options: {
                jshintrc: '.jshintrc',
                reporter: require('jshint-stylish')
         },
        all: {
                src: [ 'Gruntfile.js', 'app/scripts/{,*/}*.js' ]
        }
}
```

JSHint: static JS code analysis tool

.jshintrc: json file contains configuration for JSHint

Creating a Distribution Folder

- Your project contains many files
  - Bower components, node modules, Less/Sass files and configuration files
- You may wish to generate a distribution folder that:
  - contains only files essential to serve up your website
  - compacted and minified files

Copying and Cleaning Up

- Create a distribution folder and clean up the distribution folder:

```
npm install grunt-contrib-copy --save-dev
npm install grunt-contrib-clean --save-dev
```

- Configuration:

```
copy: {
        dist: {
                cwd: 'app',
                src: [ '**','!**/*.css','!**/*.js' ],
                dest: 'dist',
                expand: true
        }
}
```

```
clean: {
        build:{
                src: [ 'dist/']
        }
}
```

```
grunt.registerTask('build',  [ 'clean', 'jshint', 'copy' ]);
```

Preparing the Distribution Folder

- Grunt modules:

  npm install grunt-contrib-concat --save-dev

  npm install grunt-contrib-cssmin --save-dev

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

  npm install grunt-filerev --save-dev

  npm install grunt-usemin --save-dev

grunt-usemin

- Umbrella task that configures and completes most of the CSS and JS minification and uglification tasks
- Flow of tasks:

  useminPrepare → concat

  → cssmin → uglify → filerev → usemin

useminPrepare

- Looks for block configuration in an html file

```
<!-- build:css styles/main.css -->
        . . .
<!-- endbuild -->
```

- Configuration:

```
useminPrepare: {
        html: 'app/menu.html',
        options: { dest: 'dist' }
}
```

- Will automatically generate configuration information for concat, cssmin and uglify tasks

filerev

- Revision your files: adds revision tag to the name of your file:
  - e.g.: main.css → main.23758735.css
- Configuration:

```
filerev: {
        options: {
                encoding: 'utf8', algorithm: 'md5', length: 20
        },
         release: {
                files: [{
                        src: [ 'dist/scripts/*.js', 'dist/styles/*.css', ]
                }]
        }
}
```

usemin

- After concat, cssmin, uglify and filerev are run, this will replace the css and JS links with the single concatenated files from the dest folder
- Configuration:

```
usemin: {
        html: ['dist/*.html'],
        css: ['dist/styles/*.css'],
        options: {
                assetsDirs: ['dist', 'dist/styles']
        }
}
```

Watch

- Keeps a watch on files and reruns tasks whenever changes occur

  npm install grunt-contrib-watch --save-dev

- Configuration:

```
watch: {
        copy: {
                files: [ 'app/**', '!app/**/*.css', '!app/**/*.js'],
                tasks: [ 'build' ]
        },
        livereload: {
                options: { livereload: '<%= connect.options.livereload %>' },
                files: [ 'app/{,*/}*.html', '.tmp/styles/{,*/}*.css',
                        'app/images/{,*/}*.{png,jpg,jpeg,gif,webp,svg}' ]
        }
}
```

Connect

- Start a server to serve the website

  npm install grunt-contrib-connect --save-dev

- Configuration:

```
connect: {
        options: {
                port: 9000, hostname: 'localhost', livereload: 35729
        },
        dist: {
                options: {
                        open: true,
                        base:{ path: 'dist', options: { index: 'menu.html', maxAge: 300000 }}
                }
        }
}
```

Grunt Tasks

- Register Multiple Tasks:

```
grunt.registerTask('build',
        [ 'clean', 'jshint', 'useminPrepare', 'concat', 'cssmin',
                'uglify', 'copy', 'filerev', 'usemin' ]);

grunt.registerTask('serve',['build','connect:dist','watch']);

grunt.registerTask('default',['build']);
```

Exercise: Web Tools: Grunt

- Install Grunt CLI and Grunt packages in your project
- Configure a Grunt file with a set of tasks to build a web project from a source, and serve the built project using a server