Telerik Academy

Tools for JavaScript Development

Unleash the Power of JavaScript Tooling

End-to-end JavaScript Applications

Telerik Software Academy
http://academy.telerik.com

Telerik Academy

Table of Contents

- Project tools
  - Package Management: NPM & Bower
  - Scaffolding: Yeoman
  - Task Automation: Grunt

Telerik Academy

Project Tools

No matter the Editor

Project Tools

- NPM & Bower
  - Install Node.js packages or client libraries
- Grunt
  - Tasks runner
  - Create different tasks for build/development/test cases
- Yeoman
  - Scaffolding of applications
  - One-line-of-code to create a project template with views/routes/modules/etc…

Telerik Academy

Package Management

Bower & NPM

Telerik Academy

Package Management: NPM

- Node.js Package Management (NPM)
  - Package manager for Node.js modules
  - `$ npm init` in CMD (Win) or Terminal (MAC/Linux)
  - Initializes an empty Node.js project with package.json file

```
$ npm init
//enter package details
name: "NPM demos"
version: 0.0.1
description: "Demos for the NPM package management"
entry point: main.js
test command: test
git repository: http://github.com/user/repository-name
keywords: npm, package management
author: doncho.minkov@telerik.com
license: BSD-2-Clause
```

Telerik Academy

Package Management: NPM (2)

- Installing modules
  - `$ npm install package-name [--save-dev]`
    - Installs a package to the Node.js project
    - `--save-dev` suffix adds dependency in package.json

```
$ npm install express --save-dev
$ npm install jade --save-dev
```

- Before running the project
  - `$ npm install`
    - Installs all missing packages from package.json

```
$ npm install
```

Telerik Academy

Package Management: NPM

Live Demo

Telerik Academy

Package Management: Bower

- Bower is a package management tool for installing client-side JavaScript libraries
  - Like jQuery, KendoUI, AngularJS, etc…
  - It is a Node.js package and should be installed first

```
$ npm install -g bower
```

  - `$ bower init` in CMD (Win) or Terminal (Mac/Linux)

```
$ bower init
```

  - Asks for pretty much the same details as `$ npm init`
  - Creates bower.json file to manage libraries

Telerik Academy

Package Management: Bower (2)

- Searching for libraries

```
$ bower search kendo
```

```
Search results:

    kendo-ui git://github.co
    angular-kendo-ui git://g
    kendo-backbone git://git
    knockout-kendo git://git
    kendo-global git://githu
```

```
bower cached          git://gith
bower validate        2014.1.318
i.git#*
bower cached          git://gith
bower validate        2.0.3 agai
.0.3
bower resolution      Removed un
bower install         kendo-ui#2
bower install         jquery#2.0

kendo-ui#2014.1.318 bower_comp
└── jquery#2.0.3

jquery#2.0.3 bower_components\
```

- Installing libraries

```
$ bower install kendo-ui
```

Telerik Academy

Package Management: Bower

Live Demo

Telerik Academy

Grunt

Tasks Runner

Grunt

- Grunt is a Node.js task runner
  - It can runs different tasks, based on configuration
  - Tasks can be:
    - Concat and minify JavaScript/CSS files
    - Compile SASS/LESS/Stylus
    - Run jshint, csshint
    - Run Unit Tests
    - Deploy to Git, Cloud, etc…
    - And many many more

Grunt (2)

- Why use a task runner?
  - Task runners gives us automation, even for different profiles:

DEVELOPMENT	TEST	BUILD
jshint	jshint	jshint
stylus	stylus	stylus
csshint	csshint	csshint
connect	mocha	concat
watch		uglify
		copy
		usemin

Configuring Grunt

- To configure grunt, create a Gruntfile.js file in the root directory of your application
  - It is plain-old Node.js
  - Grunt is configured programmatically
  - Create an module that exports a single function with one parameter – the grunt object

```
module.exports = function (grunt) {
  //configure grunt
};
```

Configuring Grunt (2)

- All the configuration is done inside the module
  - First execute the `grunt.initConfig()` method and pass it the configuration

```
module.exports = function (grunt) {
  grunt.initConfig({
    ...
  });
};
```

Telerik Academy

Configuring Grunt Plugins

- To use a plugin in grunt:
  - Install the plugin

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

  - Load the plugin

```
//inside the grunt module
grunt.loadNpmTasks('grunt-contrib-jshint');
```

  - Configure the plugin

```
//inside the grunt.initConfig()
grunt.initConfig({
  jshint: {
    app: ['Gruntfile.js', 'path/to/scripts/**/*.js']
  }
});
```

Telerik Academy

Configuring Grunt Plugins

Live Demo

Telerik Academy

Grunt Plugins

Telerik Academy

Grunt Plugins: Build

- jshint (`grunt-contrib-jshint`)
  - Runs jshint for specified files
- csslint(grunt-contrib-csshint)
  - Runs csslint for specified files
- stylus (`grunt-contrib-stylus`)
  - Compiles STYL files into CSS files
- uglify (`grunt-contrib-uglify`)
  - Minifies configured JavaScript files
- concat (`grunt-contrib-concat`)
  - Concats configured JavaScript files

Telerik Academy

Grunt Plugins for Build

Live Demo

Telerik Academy

Grunt Plugins: Development

- connect (`grunt-contrib-connect`)
  - Stars a Web server on a given port and host
- watch (`grunt-contrib-watch`)
  - Watches for changes to configured files
  - Can run other tasks on file changed

Telerik Academy

Grunt Plugins for Development

Live Demo

Telerik Academy

Yeoman

Application Scaffolding

Yeoman

- Yeoman is a Node.js package for application scaffolding
  - Uses bower & NPM to install the js package
  - Has lots of generators for many types of applications:
    - MEAN, AngularJS, Kendo-UI, WebApp, WordPress, Backbone, Express, etc…
    - Each generators install both needed Node.js packages and client-side JavaScript libraries
    - Generated Gruntfile.js for build/test/serve

Yeoman (2)

- Install Yeoman:

```
$ npm install -g yo
```

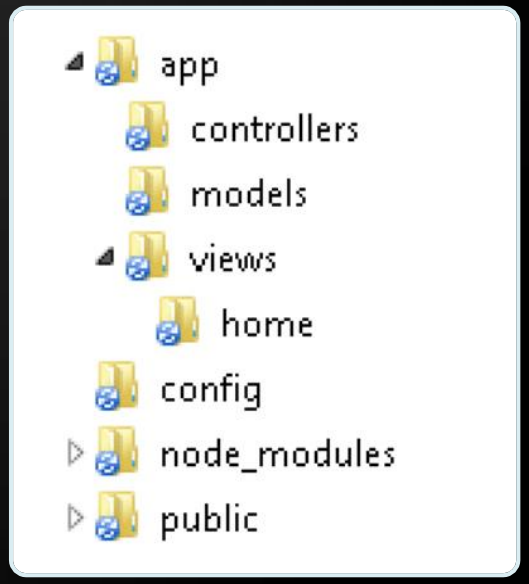
- Install Yeoman generator:

```
$ npm install -g generator-express
```

- Scaffold Express application:

```
$ cd path/to/app/directory
$ yo express
```

  - Generates:



Telerik Academy

Yeoman Scaffolding

Live Demo

Telerik Academy

Tools for JavaScript Development

http://academy.telerik.com

Homework

1. Create a project with a Gruntfile.js and three folders – DEV folder, APP folder and DIST folder
   - Write some CoffeeScript, Jade and Stylus in APP
   - Register the following grunt tasks:
     - Serve:
       - Compiles the CoffeeScript to JS and puts them into DEV/scripts
       - Runs jshint on the compiled JS files
       - Compiles the Jade to HTML and puts them into DEV
       - Compiles the Stylus to CSS and puts them into DEV/styles
       - Copies every image from the APP/images to DEV/images
       - Connect a server on port 9578 and show the contents of DEV
       - Watch for changes to the CoffeeScript, Stylus and Jade files, and if changed – reload the page into the browser

Telerik Academy

Homework (2)

1. *cont. Create a project with a Gruntfile.js and three folders – DEV folder, APP folder and DIST folder
   - Write some CoffeeScript, Jade and Stylus in APP
   - Register the following grunt tasks:
     - Build:
       - Compiles CoffeeScript, Stylus and Jade
       - Runs jshint and csslint
       - Concats all CSS files into one file and minify it into DIST/styles
       - Concats all JS files into one file and uglify it into DIST/scripts
       - Uglifies the HTML files into DIST
       - Copies all images into DIST/images