



Community Experience Distilled

# Learning Yeoman

Design, implement, and deliver a successful modern web application project using three powerful tools in the Yeoman workflow

Jonathan Spratley

[PACKT] open source\*  
PUBLISHING community experience distilled

# Learning Yeoman

Design, implement, and deliver a successful modern web application project using three powerful tools in the Yeoman workflow

**Jonathan Spratley**



BIRMINGHAM - MUMBAI

# Learning Yeoman

Copyright © 2014 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: August 2014

Production reference: 1120814

Published by Packt Publishing Ltd.  
Livery Place  
35 Livery Street  
Birmingham B3 2PB, UK.

ISBN 978-1-78398-138-0

[www.packtpub.com](http://www.packtpub.com)

Cover image by Tony Shi ([shihe99@hotmail.com](mailto:shihe99@hotmail.com))

# Credits

**Author**

Jonathan Spratley

**Reviewers**

Lauren Bridge

Dejan Markovic

Matt Momont

**Commissioning Editor**

Edward Gordon

**Acquisition Editors**

Sam Birch

Ellen Bishop

**Content Development Editor**

Rikshith Shetty

**Technical Editor**

Neha Mankare

**Copy Editors**

Janbal Dharmaraj

Karuna Narayanan

Alfida Paiva

**Project Coordinators**

Melita Lobo

Kartik Vedam

**Proofreaders**

Simran Bhogal

Lauren Harkins

Lawrence Herman

Joanna McMahon

**Indexers**

Hemangini Bari

Tejal Soni

**Graphics**

Ronak Dhruv

**Production Coordinator**

Arvindkumar Gupta

**Cover Work**

Arvindkumar Gupta

# About the Author

**Jonathan Spratley**, or as his friends call him, Jonnie, is currently working at GE Software and actively uses technologies such as Yo, Bower, Grunt, Node, AngularJS, and Backbone. He leveraged his knowledge of creating Yeoman web applications to write this book. He started developing with HTML, CSS, and JavaScript around 2004 and has spent several years designing and implementing web applications for both mobile and desktop browsers. He is a full-stack developer with experience in both server- and client-side technologies. His passion for tools and technologies that streamline a developer's productivity has driven him to become an autodidact. He has written articles for Safari Books Online Blog, and Flash & Flex Developer's Magazine. This is his first time as an author.

---

I would like to dedicate this book to all those who believed in me.

---

# About the Reviewers

**Lauren Bridge** is a software engineer on the Predix platform team at GE Software. She earned her BS in Computer Science from the University of Michigan and is working on her MCS at the University of Illinois at Urbana-Champaign. She mostly develops in JavaScript and Java, and she enjoys learning more frontend web technologies every day.

---

I would like to thank Jonnie for teaching me about Yeoman.

---

**Dejan Markovic** is an accomplished web developer who enjoys working on both frontend and backend technologies. Since 2003, he has added great value to numerous projects for small- and medium-sized businesses as well as major corporations such as Rogers Media and Softchoice. He is the co-owner of NYTO Group (New York City/Toronto Group), a premium web development company in Toronto, Canada.

NYTO Group's portfolio is available at <http://nytogroup.com/portfolio/>. NYTO Group is always looking for new projects and partnerships.

**Matt Momont** is a full-stack developer at GE Software working on the Industrial Internet – the Internet of (really big) Things. He holds a Computer Science degree from the University of Notre Dame and is currently pursuing his Masters in Computer Science at the University of Illinois at Urbana-Champaign. He likes Yeoman because it helps backend developers quickly scaffold out frontend web applications.

# www.PacktPub.com

## Support files, eBooks, discount offers, and more

You might want to visit [www.PacktPub.com](http://www.PacktPub.com) for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at [www.PacktPub.com](http://www.PacktPub.com) and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at [service@packtpub.com](mailto:service@packtpub.com) for more details.

At [www.PacktPub.com](http://www.PacktPub.com), you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.



<http://PacktLib.PacktPub.com>

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

### Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print and bookmark content
- On demand and accessible via web browser

### Free access for Packt account holders

If you have an account with Packt at [www.PacktPub.com](http://www.PacktPub.com), you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

# Table of Contents

<b>Preface</b>	<b>1</b>
<b>Chapter 1: Modern Workflows for Modern Webapps</b>	<b>7</b>
<b>An overview of Yeoman</b>	<b>7</b>
Yeoman's architecture	8
Node's package manager	8
Features of Yeoman	8
Quick installation	9
Installing Yeoman and friends	9
Installing a generator	10
Scaffolding with Yo	10
Creating the project	10
Invoking the generator	10
Directory structure	11
The build process	12
The Connect LiveReload server	13
Previewing the server	13
Package management with Bower	14
Code linting with JSHint	16
Automation	18
Testing with PhantomJS	22
Running tests	23
Optimizing for production	24
<b>Self-test questions</b>	<b>26</b>
<b>Summary</b>	<b>26</b>
<b>Chapter 2: Getting Started</b>	<b>27</b>
<b>Yo – generators</b>	<b>27</b>
The Yeoman workflow	28
Official generators	28

<b>The generator-webapp</b>	<b>29</b>
Features	29
Installing the generator-webapp	29
Using the generator-webapp	29
Options	30
Example usage	30
Previewing	30
Conclusion	31
<b>The generator-angular</b>	<b>31</b>
Features	32
Installing the generator-angular	32
Using the generator-angular	32
Options	32
Example usage	32
Angular subgenerators	33
Previewing	35
Conclusion	36
<b>The generator-backbone</b>	<b>36</b>
Features	36
Installing the generator-backbone	36
Using the generator-backbone	36
Options	37
Example usage	37
Backbone subgenerators	37
Previewing	38
Conclusion	39
<b>The generator-ember</b>	<b>39</b>
Features	40
Installing the generator-ember	40
Using the generator-ember	40
Options	40
Example usage	40
Ember subgenerators	41
Previewing	42
Conclusion	42
<b>Self-test questions</b>	<b>43</b>
<b>Summary</b>	<b>43</b>
<b>Chapter 3: My Angular Project</b>	<b>45</b>
<b>Anatomy of an Angular project</b>	<b>45</b>
Why Angular?	46
Creating a new Angular project	47
Installing the generator-angular	47
Scaffolding the application	47
Configuring the application	49
Creating the application definition	51
Creating the application controller	52
Creating the application views	53

---

Customizing the main view	56
Previewing the application	57
<b>Testing an Angular application</b>	<b>58</b>
Angular unit tests	58
End-to-end tests with Protractor	59
<b>Angular controllers</b>	<b>61</b>
Creating controllers	62
Using controllers	62
Testing controllers	63
<b>Angular services</b>	<b>64</b>
Creating services	64
Using services	65
Testing services	65
<b>Angular filters</b>	<b>67</b>
Creating filters	67
Using filters	68
Testing filters	68
<b>Angular directives</b>	<b>69</b>
Creating directives	69
Using directives	70
Testing directives	71
<b>Angular views</b>	<b>72</b>
Creating the Angular views	72
<b>Self-test questions</b>	<b>74</b>
<b>Summary</b>	<b>75</b>
<b>Chapter 4: My Backbone Project</b>	<b>77</b>
<b>Anatomy of the Backbone project</b>	<b>77</b>
<b>The new Backbone project</b>	<b>78</b>
Installing the generator-backbone	78
Scaffolding a Backbone application	79
Understanding the directory structure	80
Configuring the application	81
Scaffolding the app view	82
The Backbone app view	82
The Handlebars app template	83
Scaffolding the main view	84
The Backbone main view	84
Scaffolding the app router	86
Bootstrapping the app	87
Previewing the app	88
<b>Testing</b>	<b>89</b>
Configuration	89
Unit testing	91
End-to-end tests	92

<b>Backbone.Events</b>	<b>92</b>
Creating events	92
Using events	92
Testing events	93
<b>Backbone.Model</b>	<b>93</b>
Scaffolding models	94
Using the Backbone models	95
Creating a model	95
Updating a model	95
Saving a model	96
Destroying a model	96
Validating a model	96
Testing a model	97
<b>Backbone.Collection</b>	<b>97</b>
Creating collections	98
Using collections	99
Testing collections	99
<b>The Backbone view</b>	<b>100</b>
Creating views	100
Using views	100
Testing views	102
<b>Backbone.Router</b>	<b>103</b>
Creating routers	103
Using routers	103
Testing routers	104
<b>Self-test questions</b>	<b>106</b>
<b>Summary</b>	<b>106</b>
<b>Chapter 5: My Ember Project</b>	<b>107</b>
<b>Anatomy of the Ember project</b>	<b>107</b>
<b>The new Ember project</b>	<b>108</b>
Installing the generator-ember	108
Scaffolding the application	108
Understanding the directory structure	109
Application configuration	110
Application definition	111
The application template	112
The index template	113
The feature component	114
Previewing the application	115
<b>Testing</b>	<b>116</b>
The test helpers	117

---

Setup	117
End-to-end integration tests	118
Unit tests	120
<b>Ember Data</b>	<b>121</b>
Ember Data concepts	122
<b>Models</b>	<b>122</b>
Creating a model	122
Methods	122
Attributes	123
Fixtures	124
<b>Records</b>	<b>125</b>
Finding all records	125
Finding a single record	125
Creating a record	125
Deleting a record	126
<b>Routes</b>	<b>126</b>
Creating the routes	126
Using routes	127
Posts route	128
Post route	128
Posts edit route	129
<b>Templates</b>	<b>129</b>
Handlebar helpers	129
Posts template	130
Post template	131
Posts edit template	133
<b>Controllers</b>	<b>134</b>
Post edit controller	135
<b>Self-test questions</b>	<b>136</b>
<b>Summary</b>	<b>136</b>
<b>Chapter 6: Custom Generators</b>	<b>137</b>
<b>Anatomy of a generator</b>	<b>137</b>
Types of generators	138
<b>The new custom generator</b>	<b>138</b>
Installing the generator-generator	138
Using generator-generator	138
Understanding the directory structure	139
Adding logic to the generator	140
Initializing the generator	141
Asking questions to the user	141
Copying the project files	143

Copying the application files and folders	144
Installing dependencies with Bower	145
<b>Creating custom templates</b>	<b>146</b>
Creating the Gruntfile.js file	146
Creating the package.json file for npm	149
Creating the .editorconfig file for IDEs	149
Creating the .jshintrc file for JSHint	150
Creating the .travis.yml file for Travis CI	151
The .gitattributes file for Git	151
The .gitignore file for Git	151
Creating the .bowerrc file for Bower	152
Creating the bower.json file for Bower	152
Creating the application templates	153
<b>Testing a custom generator</b>	<b>156</b>
Setup	156
Testing the generator output	158
Test generator loading	160
<b>The new custom subgenerator</b>	<b>162</b>
Understanding the subgenerator's directory structure	162
Creating subgenerator templates	163
Adding logic to the subgenerator	163
Using your custom generator	164
Link your generator	164
Scaffolding a new webapp	165
<b>Self-test questions</b>	<b>167</b>
<b>Summary</b>	<b>167</b>
<b>Chapter 7: Custom Libraries</b>	<b>169</b>
<b>The new CommonJS project</b>	<b>170</b>
Installing the generator-commonjs	170
Scaffolding a CommonJS project	170
The CommonJS logic	171
Module properties	172
Connecting to MongoDB	173
Finding all models	174
Finding a model	174
Creating a model	175
Updating a model	176
Testing a CommonJS project	177
Test for no model	179
Test finding all models	179
Test finding one model	180
Test creating a model	180
Test updating a model	181
Test destroying a model	182
Deploying to npm	183
Conclusion	183

---

<b>The new Node.js module project</b>	<b>184</b>
Installing the generator-node	184
Scaffolding a Node.js module project	184
The NodeJS module logic	185
Testing a Node.js module	186
Deploying	189
Conclusion	189
<b>The new jQuery project</b>	<b>189</b>
Installing the generator-jquery	189
Scaffolding a jQuery project	190
Adding the plugin logic	191
Testing a jQuery plugin	192
Creating the unit test	193
Deploying to Bower	196
Conclusion	196
<b>Self-test questions</b>	<b>197</b>
<b>Summary</b>	<b>197</b>
<b>Chapter 8: Tasks with Grunt</b>	<b>199</b>
<hr/>	
<b>Overview on GruntJS</b>	<b>199</b>
<b>Installing the Grunt CLI</b>	<b>200</b>
Installing Grunt	200
Grunt usage	201
Grunt options	201
Installing the generator-gruntfile	201
<b>Using Grunt</b>	<b>202</b>
The package.json file	202
The Gruntfile.js file	203
Loading tasks	203
Creating the alias tasks	204
Multiple target tasks	204
Registering the basic tasks	205
The new project	208
<b>My custom Grunt plugin</b>	<b>209</b>
Installing the generator-gruntplugin	209
Usage	209
The directory structure	209
The Grunt plugin logic	210
Plugin options	211
Using Grunt to read files	211
Using Grunt to write files	212
Testing a Grunt plugin	213
Creating test fixtures	214
Running the tests	214

Deploying to npm	215
Usage	215
<b>Self-test questions</b>	<b>216</b>
<b>Summary</b>	<b>216</b>
<b>Chapter 9: Yeoman Tips and Tricks</b>	<b>217</b>
<b>WebApp generator solutions</b>	<b>217</b>
Creating a RESTful Node.js server	218
Installing module dependencies	218
Creating the server	218
Configuring the server	218
Configuring the data source	219
Defining server routes	220
Starting the server	223
Running the server	223
Testing the server	223
Setting up the proxy server	224
Conclusion	225
<b>Angular generator solutions</b>	<b>226</b>
Protractor e2e testing	226
Installing Protractor	226
Installing the grunt-protractor-runner	227
Configuring the Protractor task	227
Creating the Protractor configuration	227
Creating an e2e spec	229
Starting the Selenium WebDriver	232
Starting the application	233
Running e2e tests	233
<b>Backbone generator solutions</b>	<b>233</b>
Code coverage with Karma	233
Installing Karma and plugins	234
Karma configuration	234
Configuring test-main.js	236
Running tests	237
Code coverage report	238
<b>Self-test questions</b>	<b>238</b>
<b>Summary</b>	<b>238</b>
<b>Appendix: Yeoman Resources</b>	<b>239</b>
<b>Reference guides</b>	<b>239</b>
Yo – the scaffolding tool	239
Usage	239
Bower – the package tool	240
Usage	240
Commands	240
Options	240
Grunt – the build tool	241

Usage	241
Options	241
<b>Git</b>	<b>242</b>
Usage	242
Commands	242
<b>Jasmine – behavior-driven JavaScript</b>	<b>243</b>
Structure of a suite	243
Matchers	243
Spy matchers	244
Reserved words	244
<b>Installation guides</b>	<b>245</b>
Installing Git	245
Installing Git on Windows	245
Installing Git on Mac	246
Installing Node.js and npm	246
Installing Node on Windows	246
Installing Node on Mac	246
Installing Yo	247
Installing Yo on Mac/Windows	247
Installing Grunt	247
Installing Grunt on Mac/Windows	247
Installing Bower	248
Installing Bower on Mac/Windows	248
<b>Self-test answers</b>	<b>248</b>
<b>Summary</b>	<b>253</b>
<b>Index</b>	<b>255</b>

---



# Preface

Now is the time to start using a workflow that can keep up with the fast pace of the development world. Software changes so fast that keeping your project libraries updated and using the latest code has always been a manual, tedious process. Well, not anymore, thanks to modern tooling that has taken my development productivity to greater levels! I have been using Yeoman since the early versions, where Yeoman was the one tool that could do it all.

Since the Yeoman project grew, it has evolved into something I have always wanted in the web development community, such as code generators that can quickly scaffold out working applications that are in alignment with the best practices of that specific framework or language. Now, the time has come and Yeoman is going to take the development world by storm and grow into something that will become a standard in creating modern web applications.

This book is a compilation of using the most popular Yeoman generators on npm. We explore the options that each tool has to offer and use them to create various types of projects, ranging from AngularJS applications to Node.js modules. This book provides examples and information regarding the tools in Yeoman.

## What this book covers

*Chapter 1, Modern Workflows for Modern Webapps*, is an overview of the three core tools used in the Yeoman workflow – Yo, Bower, and Grunt. We cover how to use these tools in development and how to incorporate the workflow into new or existing projects, followed by an example of each of the features in Yeoman.

*Chapter 2, Getting Started*, begins with installing Yeoman for development and an overview on the AngularJS, Backbone.js, Ember.js, and webapp generators, the options and subgenerators it uses, and examples of using each generator to start the project.

*Chapter 3, My Angular Project*, starts out with covering the concepts of Angular and the anatomy of an AngularJS application. We will use the generator-angular to scaffold an extendable AngularJS application that uses directives, services, and factories. We will cover setting up a CRUD application with unit tests that use the Karma runner.

*Chapter 4, My Backbone Project*, covers the anatomy of a Backbone.js project and the concepts behind the library. We create a Backbone application to perform CRUD operations on a data source that is unit tested using Jasmine. The project uses CoffeeScript, Require.js, and AMD to create a well-structured app ready for extending.

*Chapter 5, My Ember Project*, starts out by creating a new Ember.js project. We then cover how an Ember application is structured and the concepts around the framework, configuring a test environment that is used to run both unit and integration tests.

*Chapter 6, Custom Generators*, covers the Yeoman generator API and the common methods used when developing generators. We also cover installing and invoking the generator-generator to create a custom Yeoman generator with option prompts that scaffold a custom application based on users' feedback. We cover how to handle testing the generators using nodeunit and then we publish the generator to npm.

*Chapter 7, Custom Libraries*, covers using Yeoman to create custom libraries that are deployed to either Bower or npm. We learn how to use the Node.js generator and the CommonJS generator to create a Node module, followed by a client-side jQuery plugin that handles sending CRUD operations to a Node REST API server.

*Chapter 8, Tasks with Grunt*, starts out by covering all the available options when using the Grunt command. We install two Yeoman Grunt generators: the Gruntfile generator that enables adding Grunt to existing projects, and the Grunt plugin generator. We cover creating a custom Grunt task that is then deployed to npm along with unit tests using the nodeunit framework.

*Chapter 9, Yeoman Tips and Tricks*, aims to cover the holes from the Yeoman generators and specific projects. We cover adding code coverage to a Backbone.js project, as well as setting up Protractor to run end-2-end testing for our Angular project.

## What you need for this book

You will need to have the following installed software on your development box in order to properly run the examples and tutorials in the chapters:

- Node 0.10.24
- NPM 1.4.7
- Git 1.8.5.2
- Ruby 1.9.2
- Text editor of some sort
- Google Chrome

## Who this book is for

This book is for newbies and intermediate web developers looking to speed up the process when it comes to creating web applications of various frameworks. You should have basic knowledge of HTML, CSS, and JavaScript. The examples in this book will use jQuery style selectors and methods, so some jQuery experience is needed. The tools in this book involve the command line, so having basic knowledge about using the shell to invoke commands on a system is required. As long as you understand basic principles about structuring HTML and OO JavaScript applications, you should have no problem following the step-by-step examples in the chapters.

## Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows:  
"The `bower.json` file is how Bower manages the project's dependencies."

A block of code is set as follows:

```
<div class="header">
  <ul class="nav nav-pills pull-right">
    <li ng-repeat="item in App.menu"
      ng-class="{ 'active': App.location.path() === item.href}">
      <a ng-href = "#{item.href}"> {{item.title}} </a>
    </li>
  </ul>
  <h3 class="text-muted"> {{ App.sitetitle }} </h3>
</div>
```


When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:


```
require 'scripts/config'
LearningYeomanCh5 = window.LearningYeomanCh5 =
  Ember.Application.create(
    LOG_VIEW_LOOKUPS: true
    LOG_ACTIVE_GENERATION: true
    LOG_BINDINGS: true
    config: window.Config
  )
```

Any command-line input or output is written as follows:

```
$ npm install -g generator-webapp
```

**New terms** and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "See for yourself; open Chrome Developer Tools and click on the **Network** tab."

 [ Warnings or important notes appear in a box like this. ]

 [ Tips and tricks appear like this. ]

## Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book – what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to [feedback@packtpub.com](mailto:feedback@packtpub.com), and mention the book title via the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on [www.packtpub.com/authors](http://www.packtpub.com/authors).

## Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

## Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at <http://www.packtpub.com>. If you purchased this book elsewhere, you can visit <http://www.packtpub.com/support> and register to have the files e-mailed directly to you.

## Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books – maybe a mistake in the text or the code – we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting <http://www.packtpub.com/submit-errata>, selecting your book, clicking on the **errata submission form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from <http://www.packtpub.com/support>.

## **Piracy**

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at [copyright@packtpub.com](mailto:copyright@packtpub.com) with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

## **Questions**

You can contact us at [questions@packtpub.com](mailto:questions@packtpub.com) if you are having a problem with any aspect of the book, and we will do our best to address it.

# 1

## Modern Workflows for Modern Webapps

This chapter will cover the three core tools that make up the Yeoman workflow, how to use these tools in development, and how to incorporate this workflow into new or existing projects.

In this chapter, you will learn the following topics:

- The Yeoman tools and architecture
- Downloading and installing Yeoman
- Features of Yeoman
- Using the Yeoman tools

### An overview of Yeoman

The term **modern webapps** is a relatively new thing, as the Web is still in its infancy stage. As the Web matures, so does the need for developer tools and workflows, thanks to some modern-day Web pioneers over at Google. Paul Irish and Addy Osmani have developed a modern workflow that goes by the name of **Yeoman**.

The Yeoman workflow is a collection of three tools to improve developers' productivity when building web applications: **Yo** is the scaffolding tool, **Grunt** is the build tool, and **Bower** is the package tool.

- Yo is used to scaffold things such as projects and files from templates
- Grunt is used for task management, testing, code linting, and optimization
- Bower is used for package management and to manage client-side dependencies

## Yeoman's architecture

The Yeoman toolset runs in the Node.js environment and is invoked from the command line. Each tool is installed using **Node's package manager (npm)** and uses the npm repository to manage all plugins.

## Node's package manager

Node.js is a platform that is built on Chrome's JavaScript runtime engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight, efficient, and perfect for real-time applications that run across distributed devices.

The official package manager for Node.js is npm. From Node versions 0.6.3 and up, npm is bundled and installed automatically with the environment. The npm package manager runs through the command line and manages the application dependencies that are available on the npm registry.

 The current Node.js version used in this book is v0.10.28. 


## Features of Yeoman

Before we dig deep into using each tool of the workflow, let's take a look at some of the Yeoman tooling features that will help you in your next project:

- **Quick install:** Easily installs all three tools from the npm repository using one command
- **Scaffolding:** Fast and easy-to-use command-line tool to create new projects or files from templates that individual generators provide
- **Build process:** Tasks for concatenation, minification, optimization, and testing
- **Preview server:** Connect LiveReload server to preview your application in the browser
- **Package management:** Search, install, and manage project dependencies via the command line
- **Code linting:** Scripts are run against JSHint to ensure language best practices
- **Automation:** A simple watch task to compile CoffeeScript, LESS, or SASS, and reload the browser upon changes

- **Testing:** Executes JavaScript code in multiple real browsers with the Karma runner
- **Optimization:** Images are optimized using OptiPNG and JPEGtran, and HTML is optimized using the HTML minifier

The preceding features are dependent on what the individual generators provide via Grunt tasks. By default, the Angular, Backbone, Ember, and other webapp generators provide tasks to perform all the features listed.

 Grunt tasks are individual plugins that perform specific operations on files or folders.

## Quick installation

Modern tools usually mean more tools to learn, but learning the tools of the Yeoman workflow is easier than you think. To demonstrate by example, here is how easy it is to get a modern web application up and running, all from the command line.


## Installing Yeoman and friends

To install all three tools in the Yeoman workflow, just execute the following command in the terminal:

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

The command will install Yo, Grunt, and Bower into your systems path as follows:

- The `-g` option flag specifies the installation to be globally available in your path, allowing the `yo` command to be invoked from anywhere
- If using the latest versions of Node and Git, Yeoman will automatically install Bower and Grunt while installing Yo

 The `-g` flag installs globally and requires an administrator user.

If you run into any errors during the initial installation process, you can install the `envcheck` module to ensure that your system is ready for all of Yeoman's features; just execute the following command:

```
$ npm install -g envcheck
```

## Installing a generator

To install generators for Yo, use npm. Let's install the generic webapp generator; open a terminal and execute the following command:

```
$ npm install -g generator-webapp
```

The preceding command will install the webapp generator globally on your system, easily letting you create new web projects within any directory of your choice.

## Scaffolding with Yo

Yeoman includes a powerful command-line utility that can scaffold files based on individual generator templates, allowing you to save time creating files from scratch. There are over 700 community generators on npm.

- To search for generators, add the `generator-` prefix before the name, as follows:  

```
$ npm search generator-[name]
```
- To install generators, use `npm install` passing in the name of the package, as follows:

```
$ npm install generator-[name]
```



The `npm` attribute is the package manager for Node.js and comes bundled with it.

## Creating the project

All Yeoman commands work off the current directory, so create a new folder named `learning-yeoman-ch1`, open the terminal, and `cd` to that location into the newly created directory.

## Invoking the generator

Yo, the scaffold tool, will easily create and set up project configuration, files, and folders needed for a modern web application. Execute the following command:

```
$ yo webapp
```



```
| | |─ bootstrap
| | └─ jquery
| └─ favicon.ico
| └─ images
| └─ index.html
| └─ robots.txt
| └─ scripts
| | └─ main.js
| └─ styles
|   └─ main.css
└─ bower.json
└─ node_modules
└─ package.json
└─ test
    └─ bower.json
    └─ bower_components
    └─ index.html
    └─ spec
```

Just think of Yeoman as a helpful robot that does all the hard work for you, creating all the necessary files and folders to get started with development.

## The build process

Yeoman includes Grunt, a task-based command-line tool for JavaScript projects. It is used to perform various build tasks on projects and exposes several useful tasks that you will want to use in your workflow. Yeoman automatically creates and configures `Gruntfile.js`, which specifies the configuration of the tasks and targets.

The following order of commands is used for a seamless development workflow:

```
$ yo webapp           #scaffold application
$ bower install jquery #install dependency
$ grunt serve        #start preview server
$ grunt test         #run unit tests
$ grunt              #create optimized build
```