Community Experience Distilled

Less Web Development Essentials

Second Edition

Leverage the features of Less to write better, reusable, and maintainable CSS code

Bass Jobsen

[PACKT] PUBLISHING open source* community experience distilled

Less Web Development Essentials

Second Edition

Leverage the features of Less to write better, reusable, and maintainable CSS code

Bass Jobsen

[PACKT] PUBLISHING open source community experience distilled

BIRMINGHAM - MUMBAI

Less Web Development Essentials

Second Edition

First published: April 2014

Second edition: April 2015

Production reference: 1270415

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

ISBN 978-1-78355-407-2

www.packtpub.com

Credits

Author

Bass Jobsen

Reviewers

Ben Burwell

Shahin Katebi

Maulik Suchak

Xun (Brian) Wu

Acquisition Editor

Tushar Gupta

Content Development Editor

Sriram Neelakantan

Technical Editors

Novina Kewalramani

Shruti Rawool

Copy Editors

Jasmine Nadar

Alpha Singh

Trishla Singh

Laxmi Subramanian

Neha Vyas

Project Coordinator

Judie Jose

Proofreaders

Safis Editing

Paul Hindle

Indexer

Tejal Soni

Production Coordinator

Alwin Roy

Cover Work

Alwin Roy

About the Author

Bass Jobsen has been programming for the Web since 1995, covering everything from C to PHP, and is always on the hunt for the most accessible interfaces. Based in Orthen, the Netherlands, he was the coauthor of *Less Web Development Cookbook, Packt Publishing*, which contains over 110 practical recipes to help you write leaner, more efficient CSS code using Less.

He uses Less in his daily job for web design tasks, WordPress theme development, the awesome StreetArt.nl project, and other Twitter Bootstrap apps.

He is always happy to help you at `http://stackoverflow.com/users/1596547/bass-jobsen`, and he blogs at `http://bassjobsen.weblogs.fm/`.

Also, check out his Bootstrap WordPress Starter Theme (JBST) and other projects on GitHub at `https://github.com/bassjobsen`.

Writing this book wouldn't have been possible without the support of my family. The entire team at Packt was helpful, specifically Tushar Gupta who was a patient and excellent motivator and critical reader. Sriram Neelakantan helped me to dot the *i*s and cross the *t*s. I would also like to thank the reviewers of this book, Ben, Shahin, Maulik, and Xen (Brian), for their critical and valuable suggestions, which make this book even better.

Last but not least, I should not forget to thank the Less core team—Alexis Sellier (`@cloudhead`), Jon Schlinkert (`@jonschlinkert`), Luke Page (`@lukeapage`), Marcus Bointon (`@Synchro`), Mária Jurčovičová (`@sommeri`), Matthew Dean (`@matthew-dean`), Max Mikhailov (`@seven-phases-max`), and all the other contributors who have made coding Less possible in the first place.

About the Reviewers

Ben Burwell is a software engineer and web developer from Hopewell, New Jersey. He enjoys expression-oriented languages, security engineering, user interface design, and coffee. Ben also likes to spend time hiking, cooking, and enjoying fine typography. This is the first time he has reviewed a book for Packt Publishing. He has previously written about web and software development on his website (`https://www.benburwell.com/`).

Shahin Katebi is a software architect and developer with over 10 years of experience in creating apps for various platforms (Mac, iOS, Windows, and the Web). He works as a mobile solutions consultant with different companies in the business, marketing, and technical fields, and also works with start-up teams worldwide. He teaches iOS / Mac OS development and is a mentor at Startup Weekend events, where he helps start-up teams develop their own businesses. He is the founder and business manager of Seeb Co. (`http://seeb.co/`), a creative mobile app development organization that creates apps for customers around the world. He is also running and accelerating multiple start-up businesses in various fields worldwide.

Shahin has also helped with the fact checking and technical reviewing *iOS and OS X Network Programming Cookbook* and *PHPStorm Cookbook,* both by Packt Publishing.

I want to give my special thanks and love to my adorable Nafiseh for her treasured presence and support.

Maulik Suchak is a professional frontend developer and designer with over 6 years of experience. He is the cofounder of CVsIntellect.com, a résumé builder service.

He has also started an open source project called DearWeb.org (`https://github.com/ermauliks/dearweb.org`), a theme library that will help any non-frontend developer to write HTML/CSS. More information about him is available at `www.mauliksuchak.com`.

Xun (Brian) Wu is a software developer and a software architect with over 10 years of experience in a variety of industries, including financial services, information technology, entertainment, and sports.

His background is in Java, UI, and big data development.

He is always enthusiastic about exploring new ideas, technologies, and opportunities that arise. He always keeps himself up to date by reading books, attending trainings, and researching.

He has previously reviewed *Learning Express Web Application Development, Rapid LESS, Rapid Lo-Dash, Mastering D3.js, Mastering CSS, Mastering MEAN Web Development, Rapid Meteor, Mastering Kendo UI, Rapid Grunt*, and *Learning Flask*, all for Packt Publishing.

I would like to thank my parents, my wife, and my child for their patience and support throughout this endeavor.

---

www.PacktPub.com

Support files, eBooks, discount offers, and more

For support files and downloads related to your book, please visit `www.PacktPub.com`.

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` 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` for more details.

At `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.

`https://www2.packtpub.com/books/subscription/packtlib`

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can search, access, and read 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 a web browser

Free access for Packt account holders

If you have an account with Packt at `www.PacktPub.com`, you can use this to access PacktLib today and view 9 entirely free books. Simply use your login credentials for immediate access.

This book is dedicated to Colinda, Kiki, Dries, Wolf, and Leny.

Table of Contents

Preface **vii**

Chapter 1: Improving Web Development with Less **1**

Using CSS3 for styling your HTML **2**
Using the CSS selectors to style your HTML 2
Specificity, inheritance, and cascade in CSS 3
How CSS specificity works 4
Building your layouts with flexible boxes 6
Compiling Less **7**
Getting started with Less 7
Using the watch function for automatic reloading 10
Debugging your code 11
Plugins 13
Your first layout in Less **15**
Vendor-specific rules 16
Autoprefixing 16
The -prefix-free library 18
Build rounded corners with border-radius 18
Preventing cross-browser issues with CSS resets **22**
Creating background gradients 23
CSS transitions, transformations, and animations **25**
Box-sizing **29**
Server-side compiling **33**
Using CSS source maps for debugging 34
Plugins 35
Compressing and minimizing your CSS 36
Compiling your Less code into CSS automatically 37
Graphical user interfaces 39
The OOCSS, SMACSS, and BEM methodologies **40**
Summary **42**

Chapter 2: Using Variables and Mixins 43
Using comments in Less 43
Nested comments 44
Special comments 44
Using variables 45
Organizing your files 46
Naming your variables 48
Using a variable 49
Organizing variables 49
The last declaration wins! 50
Variable declaration is not static! 52
Lazy loading 53
Variable interpolation 53
Escaping values 54
Using mixins 57
Basic mixins 58
Parametric mixins 58
Default values 59
Naming and calling your mixins 60
Using mixins with multiple parameters 61
More complex mixins for linear gradient backgrounds 63
The @arguments and @rest special variables 67
Passing rulesets as arguments 69
Return values 70
Changing the behavior of a mixin 71
Switches 71
Argument matching 72
Guarded mixins 72
CSS guards 74
Using guards and argument matching to construct loops 75
Using mixins to loop through a set of values 77
The !important keyword 79
Summary 80
Chapter 3: Nested Rules, Operations, and Built-in Functions 81
The navigation structure 81
Working with nested rules 82
Using mixins and classes 87
Variables 89
Classes and namespaces 90
Referencing the parent selector with the & symbol 92
Guards nesting and the & usage 97

Passing rulesets to mixins **98**
Operating on numbers, colors, and variables **99**
Property merging **101**
Built-in functions **102**
JavaScript 102
List functions 103
Using the color functions 105
The darken() and lighten() functions 107
Manipulating colors 108
Functions for color operations 108
Color blending with Less 111
Evaluating the type of an input value 113
Extending Less with custom functions 114
The box-shadow mixin **115**
Summary **116**
Chapter 4: Testing Your Code and Using Prebuilt Mixins Libraries **117**
Revisiting the CSS background gradients **118**
Unused code 120
Chrome's developer tools 120
Firebug CSS usage add-on 122
Testing your code **122**
Understanding TDD 122
All about style guides 123
Building a style guide with StyleDocco 123
Testing your code with tdcss.js 125
Prebuilt mixins **127**
Creating gradients and layouts with Less Elements 128
Using the comprehensive Less Hat library 131
The lesshat mixin for the less.js plugin 133
Using the 3L library of prebuilt mixins 133
SEO and HTML debugging 134
Using the Clearless library of prebuilt mixins 134
Using Preboot's prebuilt mixins for your project 137
Using the more-or-less library 139
The Less-bidi library 141
Integrating other techniques into your projects using Less **142**
Creating animations with Less 142
Using iconic fonts 144
Retina.js 149
Summary **150**

Chapter 5: Integrating Less in Your Own Projects 151
Importing CSS into Less 152
Using the @import rule 152
Migrating your project to Less 155
Organizing your files 155
Converting the CSS code to the Less code 156
Media queries and responsive design 157
Elastic measurement units in media queries 158
Making your layout fluid 158
Testing your layouts on a mobile phone 161
Coding first for mobile 162
Using grids in your designs and workflow 162
The role of CSS float in grids 163
Making your grid responsive 164
The role of the clearfix 166
Using a more semantic strategy 166
Building your layouts with grid classes 167
Building nested grids 168
Alternative grids 170
Using the display inline-box 170
Flexbox grids 172
Building your project with a responsive grid 174
Using Preboot's grid system 174
Using the grid mixins to build a semantic layout 179
Extending your grids 182
Adding the grid classes for the small grid 182
Applying the small grid on your semantic code 185
Summary 186
Chapter 6: Using the Bootstrap 3 Frontend Framework 187
Introduction to Bootstrap 187
The Bootstrap grid 189
Grid variables and mixins 191
Working with Bootstrap's Less files 194
Building a Bootstrap project with Grunt 194
Autoprefixing your code 197
Compiling your Less files 198
Bootstrap Less plugin 198
Diving into Bootstrap's Less files 199
Creating a custom button with Less 200
Customizing Bootstrap's navbar with Less 202
Bootstrap classes and mixins 205

The a11y theme for Bootstrap 211
Color schemes with 1pxdeep 211
Summary 212
Chapter 7: Less with External Applications and Frameworks 213
Cardinal CSS 214
The Less Cardinal plugin 215
Using Semantic UI with Less 215
Deploying Ionic with Less 217
Using Ionic's build chain with Less 218
Frameworks for building your grids with Less 220
The Semantic Grid System 220
Skeleton's responsive boilerplate 221
WordPress and Less 222
Using the Sage theme by Roots with Less 223
JBST with a built-in Less compiler 223
The Semantic UI WordPress theme 225
WordPress plugins and Less 226
The WP Less to CSS plugin 226
Using Less with the Play framework 226
Using Bootstrap with the Play framework 228
AngularJS and Less 228
The ngBoilerplate system 229
Meteor and Less 229
Ruby on rails and Less 230
Alternative compilers for compiling your Less code 232
The Less.php compiler 232
The .less compiler for .NET apps 233
Summary 234
Index 235

Preface

Less (Leaner CSS) is a preprocessor, which means Less code compiles into static CSS code. Less changes the way you write and maintain your CSS code; the final CSS code will be the same, but better organized in many situations.

In your hands is the *Less Web Development Essentials, Second Edition* book. Since the successful first edition of this book, Less has been improved still further. This book has been updated for the all new features that come with the release of Less Version 2. The Less Version 2 release not only has many improvements, but has also introduced the possibility to use and write Less plugins. Less plugins will enable you to customize Less for your own unique needs.

In the semantic sense, valid CSS code is also valid Less code. Less tries to stay as close as possible to the declarative nature of CSS. Coding Less is intuitive and easy to learn for those who are already familiar with CSS.

Since the introduction of CSS3 in 2012, the role of CSS in modern web development has become more and more important. Nowadays, CSS3 is not only used to style your HTML documents, but also plays an important role in the responsibility of your designs. Last but not least, CSS3 extends CSS with features such as animations and transitions.

Writing correct functional CSS code will be the first thing—and keeping this code readable—working on all major browsers will be the second. CSS files grow and become untidy during the development and maintenance processes. CSS also doesn't have the ability to modify existing values or reuse common styles. Also, doing math or defining variables is not possible in CSS. This is where Less comes into the frame.

Less is a dynamic style sheet language designed by Alexis Sellier. Started in 2010 and now maintained and extended by the Less core team, it helps you make your CSS code maintainable, reusable, and also prevents code duplications. It also enables you to write your CSS code DRY (Do not Repeat Yourself), extending CSS syntax with variables, mixins, functions, and many other techniques. Less ensures that the CSS language is more maintainable, themable, and extendable.

In this book, you will learn to write, compile, and understand Less. We will help you to perform faster and more cost effective web development. You will get practical tips to integrate Less into your current and new projects. After reading this book, you will write clear and readable CSS3 with Less. Instead of spending your time debugging your complex CSS code for a specific device or browser, you can pay more attention to your real-design tasks.

Your clients will be happy with your advanced and stable designs. Development and maintenance time, along with expenditure, will decrease.

Also, other projects know the power of Less. Projects such as Twitter Bootstrap and the WordPress Roots theme rely on Less. These projects build clear and extendable frameworks with Less.

Less is open source and licensed under Apache License. The source code of Less is maintained on GitHub. Everybody will be allowed to contribute to it. You can use Less free of charge.

What this book covers

Chapter 1, *Improving Web Development with Less*, talks about how CSS3 brought web designers advanced functions such as gradients, transitions, and animations; however, it also stresses how CSS code can become more complex and difficult to maintain. Less helps you to make your CSS maintainable, reusable, and also prevents code duplications.

Chapter 2, *Using Variables and Mixins*, explains why variables allow you to specify widely used values in a single place and then reuse them throughout the style sheet, making global changes as easy as changing one line of code. Mixins allow you to embed all the properties of a class into another class by simply including the class name as one of its properties. It also explains what parametric mixins are and how to use them.

Chapter 3, *Nested Rules, Operations, and Built-in Functions*, explains how to use nested rules for making inheritance clear and creating shorter style sheets. It also shows you how to create complex relationships between properties and use the built-in functions of Less.

Chapter 4, *Testing Your Code and Using Prebuilt Mixins Libraries*, explains how to use well-written and tested Less code of third parties for your projects. Pre-built mixins and other sources help you to (re)use them.

Chapter 5, *Integrating Less in Your Own Projects*, explains how to organize your files for new projects or get the projects you maintain ready for using Less.

Chapter 6, *Using the Bootstrap 3 Frontend Framework*, explains how to use, customize, and extend Bootstrap with Less. Bootstrap is a popular CSS, HTML, and JavaScript framework to build mobile-first responsive designs.

Chapter 7, *Less with External Applications and Frameworks*, explains how to use other frameworks and grid systems to build your HTML designs with Less, and with greater ease, including the integration of Less into WordPress.

What you need for this book

To understand and fully profit from the contents of this book, we first expect you to build a website with CSS. A basic understanding of CSS will also be required. Understanding CSS selectors and CSS precedence will help you to get the most out of it. We will introduce these CSS aspects in short, in the first chapter. Understanding the basics of using functions and parameters in functional languages such as JavaScript is valuable but not required. Don't panic if you know nothing about functions and parameters, as this book contains clear examples. Even without any (functional) programming knowledge, you can learn Less and we will help you do this. The most important skill will be the willingness to learn.

All the chapters of this book contain examples and example code. Running and testing these examples will help you to develop your Less skills. You will need a modern web browser such as Google Chrome or Mozilla Firefox to run these examples. Use any preferred text or CSS editor to write your Less code.

Who this book is for

Every web designer who works with CSS and wants to spend more time on real design tasks should read this book. It doesn't matter whether you are a novice web designer or have used CSS for years, both will profit reading this book and learn Less. We also recommend this book for teachers and students in modern web design and computer science. Less does not depend on a platform, language, or CMS. If you use CSS, you can and should profit from Less.

Conventions

In this book, you will find a number of text styles 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: "Using the Less `autoprefix` plugin or the `-prefix-free` library will be the best practice to add vendor prefixes."

A block of code is set as follows:

```
.box-shadow(@style, @c) when (iscolor(@c)) {  box-shadow: @style
  @c;
}
.box-shadow(@style, @alpha: 50%) when (isnumber(@alpha)) {
  .box-shadow(@style, rgba(0, 0, 0, @alpha));
}
```

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

```
#sidebar{
  h2{
    color: black;
    font-size: 16px;
    .screenreaders-only;
  }
}
```

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

```
lessc --modify-var="mobile=true" source.less
```

New terms and **important words** are shown in bold. Words that you see on the screen, for example, in menus or dialog boxes, appear in the text like this: " In this case, the compiler throws an error: **RuntimeError: No matching definition was found for .mixin(a, b, c, d)**."

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 disliked. Reader feedback is important for us as it helps us develop titles that you will really get the most out of.

To send us general feedback, simply e-mail `feedback@packtpub.com`, and mention the book's title in 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 at `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 from your account at `http://www.packtpub.com` for all the Packt Publishing books you have purchased. 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.

Downloading the color images of this book

We also provide you with a PDF file that has color images of the screenshots/diagrams used in this book. The color images will help you better understand the changes in the output. You can download this file from `http://www.packtpub.com/sites/default/files/downloads/40720S_ColoredImages.pdf`.

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 could 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 to our website or added to any list of existing errata under the Errata section of that title.

To view the previously submitted errata, go to `https://www.packtpub.com/books/content/support` and enter the name of the book in the search field. The required information will appear under the **Errata** section.

Piracy

Piracy of copyrighted 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` 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

If you have a problem with any aspect of this book, you can contact us at `questions@packtpub.com`, and we will do our best to address the problem.

1
Improving Web Development with Less

It is impossible to imagine modern web design without CSS. With CSS3, web designers are able to rely on advanced functions such as gradients, transitions, and animations. On the other hand, with these, the CSS code becomes more complex and difficult to maintain. Less is a CSS preprocessor that extends CSS with modern programming-language concepts. It enables you to use variables, functions, operations, and nesting (rule or selector) while coding your CSS. It also helps you write CSS with the **don't repeat yourself** (**DRY**) principle. The DRY principle prevents you from repeating any kind of information in your code.

This chapter will cover the following topics:

- Introduction to CSS3
- Compiling Less into CSS
- Vendor-specific rules
- CSS3 rounded corners, animations, and gradients
- Using box-sizing border-box
- Server-side compiling, setting up a build process, and using GUIs
- The OOCSS, SMACSS, and BEM methodologies

Using CSS3 for styling your HTML

In web design, you will use HTML to describe the structure of your documents and the CSS language to describe their presentation, including fonts, colors, and layout. The current standard HTML5 and CSS3 versions work on most modern browsers and mobile devices. CSS3 extends the old CSS with new selectors, text effects, background gradients, and animations. The power of CSS3, its new functionalities, and high acceptance on mobile devices using HTML5 and CSS3, makes it the standard for modern web design. The combination of HTML5 and CSS3 is ideal for building responsive websites because of their high acceptance on mobile phones and other devices.

Together, HTML5 and CSS3 introduce many new features. In this book, you will be shown and taught about the concepts of the most significant ones.

Using the CSS selectors to style your HTML

With Less (and CSS), you can style your HTML code using selectors. The CSS selectors are patterns or names that identify which HTML elements of the web page should be styled. The CSS selectors play an important role in writing the Less code.

For `body p.article {color:red}`, the selector here is `body p.article`. Selectors don't refer exclusively to one element. They can point to more than one element and different ones can refer to the same element. For instance, a single `p` selector refers to all the `p` elements, including the `p` elements with a `.article` class. In case of conflicts, cascade and specificity determine the styles that should be applied. When writing the Less code, we should keep the aforementioned rules in mind. Less makes it easier to write complex CSS without changing how your website looks. It doesn't introduce any limitations on your final CSS. With Less, you can edit well-structured code instead of changing the effect of the final CSS.

CSS3 introduces many new and handy selectors. One of them is `:nth-child(n)`, which makes it possible to style, for example, every fourth paragraph's `p` tag in an HTML document. The CSS code for the preceding description will look as follows:

```
p:nth-child(4n) {
  color:red;
}
```

Downloading the example code

You can download the example code files from your account at `http://www.packtpub.com` for all the Packt Publishing books you have purchased. 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.

Such selectors add powerful functions to CSS3. Nowadays, we are able to perform operations with CSS alone. However, in the past, we needed JavaScript or hardcoded styles (or classes at the very least). Less helps you organize and maintain these new selectors well and this is one of the reasons to learn it. Powerful selectors make CSS more important, but the CSS code also becomes cumbersome and difficult to maintain. Less will prevent this problem in CSS, even making complex code flexible and easy to maintain.

Visit `http://www.w3.org/TR/selectors/#selectors` for a complete list of the CSS selectors.

Specificity, inheritance, and cascade in CSS

In most cases, many CSS styles can be applied on the same HTML element, but only one declaration for each individual property will win. The properties of an element may come from declarations in different rules. The W3C specifications describe the rules for which CSS styles will get the most precedence and will ultimately be applied. You can find these specifications in the next section.

The rules regarding the order of importance have not significantly changed with CSS3. They are briefly mentioned to help you understand some of the common pitfalls with Less/CSS and how to solve them. Sooner or later, you will be in a situation where you're trying to apply a CSS style to an element, but its effect stays invisible. You will reload, pull out your hair, and check for typos again and again, but nothing will help. This is because in most of these cases, your style will be overruled by another style that has a higher precedence.

The global rules for cascade in CSS are as follows:

- Find all the CSS declarations that apply to the element and property in question
- Inline styles have the highest precedence, except for `!important`.

  The `!important` statement in CSS is a keyword used to add weight to a declaration. The `!important` statement is added at the end of a CSS property value. After this, check who set the declaration; styles set by the author get a higher precedence than the styles defined by the user or browser (default). Author styles are defined by CSS in the web page; user styles are set by the user via the settings of his or her web browser; and default styles are set by the web browsers. The importance of the user is higher than the default, and the code with the `!important` statement (refer to *Chapter 2*, *Using Variables and Mixins*, for its meaning in Less) will always get the highest precedence.

Note that browsers such as Firefox have options to disable pages in order to use other alternatives to user-defined fonts. Here, the user settings overrule CSS of the web page. This way of overruling the page settings is not part of the CSS precedence unless they are set using `!important`.

- Calculate the specificity, which is discussed in the next section.
- If two or more rules have the same precedence and specificity, the one declared last wins.

As a Less/CSS designer, you will be making use of the calculated CSS specificity in most cases.

How CSS specificity works

Every CSS declaration gets a specificity, which will be calculated from the type of declaration and the selectors used in its declaration. Inline styles will always get the highest specificity and will always be applied (unless overwritten by the first two cascade rules). In practice, you should not use inline styles in many cases as it will break the DRY principle. It will also disable you from changing your styles only on a centralized location and will prevent you from using Less for styling.

An example of an inline style declaration is shown as follows:

```
<p style="color:#0000ff;">
```

After this, the number of IDs in the selector will be the next indicator to calculate specificity. The `#footer #leftcolumn {}` selector has two IDs, the `#footer {}` selector has one ID, and so on.

Note that in this case, an ID is a unique selector starting with `#`. The `[id=]` selector for the same HTML element counts as an attribute. This means that `div#unique {}` has one ID and `div[id="unique"] {}` has zero IDs and one attribute.

If the number of IDs for two declarations is equal, the number of classes, pseudo classes, and attributes of the selector will be of importance. Classes start with a dot. For example, `.row` is a class. Pseudo classes, such as `:hover` and `:after`, start with a colon, and attributes, of course, are `href`, `alt`, `id`, and so on.

The `#footer a.alert:hover {}` selector scores two (one class and one pseudo class) and the `#footer div.right a.alert:hover {}` selector scores three (two classes and one pseudo class).