Telerik Academy

LESS

The Dynamic Stylesheet Language

CSS Styling

Telerik Software Academy

telerikacademy.com

Follow us

Telerik Academy

Table of Contents

Telerik Academy

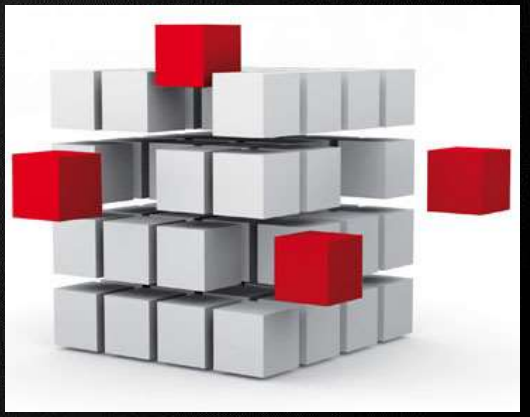
LESS Overview

What is LESS?

Telerik Academy

LESS Overview

- Extension to CSS - http://lesscss.org/
  - Write "less" code – compile to normal CSS
  - Can be parsed both Client and Server side
    - Using a LESS parser written in JavaScript
- LESS Features include
  - Variables
  - Mixins
  - Color editing functions
  - Selector nesting and more



Telerik Academy

Using LESS on the Client

- LESS can be parsed on the client (browser)
  - Include a JavaScript file, that does the parsing

```
<link rel="stylesheet/less" type="text/css" href="styles.less"/ >
<script src="less.js"></script> //after the less link
```

- Steps:
  - Write your LESS
  - Using Visual Studio, you should add a mime type for the LESS in web.config

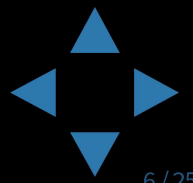
```
<configuration> <system.webServer> <staticContent>
  <mimeMap fileExtension=".less" mimeType="text/css" ></mimeMap>
</staticContent> </system.webServer> </configuration>
```

- You are ready to go

Telerik Academy

Using LESS on the Client

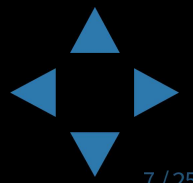
- How client-side LESS works?
  - The JavaScript performs a AJAX GET request to LESS file
  - Then it parses all the LESS code into pure CSS
  - The CSS is appended to the HEAD of the page
- Using client-side LESS is slow
  - All the parsing is done by the client
  - i.e. spend some of the browser resources for pointless parsing - Imagine a 2000-lines-long LESS file…



Telerik Academy

Parsing LESS on the Server

- LESS can be parsed on the server in many ways
  - Using the client approach and copy the parsed CSS
    - Not good enough, lots of copy-pastying
  - Using node.js to do the parsing
    - Better solution - the parsing is automated
  - Using plugins
    - IDEs - VS Web Essentials, Webstorm plugins
    - Text editors - VS Code, Atom, Sublime Text and other also have plugins



Telerik Academy

How To Use Node.js

- Install Node from https://nodejs.org/
- Open Command Prompt and write

```
npm install less -g
```

- Navigate to the folder with the .less file
- Write down the following

```
$ lessc {less file} {output CSS file}
```

- You can also minify the result

```
$ lessc –x {less file} {output CSS file}
```

Telerik Academy

LESS Features

Selector Nesting, Mixins, Variables, etc…

Telerik Academy

Selector Nesting

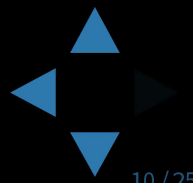
- LESS introduces selector nesting

```
body {
    font: normal 16px arial;
    color: #fff;
    background-color: #011b63;
    h1 {
        font-size: 2.3em;
        font-weight: bold;
    }
}
```

- Parses to

```
body {
    font: normal 16px arial;
    color: #fff;
    background-color: #011b63;
}

body h1 {
    font-size: 2.3em;
    font-weight: bold;
}
```

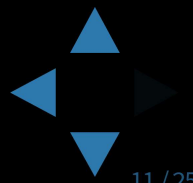
Telerik Academy

Selector Nesting

- All selectors inside a selector are translated to nested selectors
- Selectors can also reference themselves inside their selector using the symbol &

```
a {
  color: black;
  &:hover { color: red; }
}
```

```
a { color: black; }
a:hover { color: red; }
```



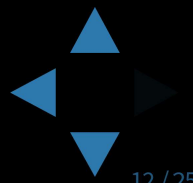
Telerik Academy

LESS Variables

- LESS also has variables
  - Using the @ (at) symbol
  - Can be used to store colors, size, etc…
- Usable to set default background-color, font-color, font-size, etc…

```
@link-color: #ffffff;
@v-link-color: #646363;
a {
  color: @link-color;
  &:visited {
    color: @v-link-color;
}
```

```
body a {
    color: white;
}
body a:visited {
    color: #646363;
}
```

Telerik Academy

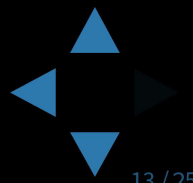
Interpolation

- LESS variables can be inserted as CSS properties
  - Using @{...}

```
@border-side: top;
@border-color: blue;
@border-style: ridge;
@border-width: 15px;

border-@{border-side} :
  @border-width @border-style @border-color;
```

```
border-top : 15px ridge blue
```



Telerik Academy

Extend

- The :extend pseudo selector can be used to extend a selector
- More here: http://lesscss.org/features/#extend-feature

```
nav ul {
  &:extend(.inline);
  background: blue;
}
.inline {
  color: red;
}
```

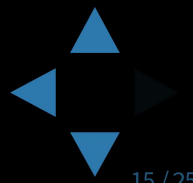
```
nav ul {
  background: blue;
}
.inline, nav ul {
  color: red;
}
```


Telerik Academy

Functions

- LESS support predefined functions
  - For stuff with colors
  - For sizes (percentages)
  - Find all the functions at http://lesscss.org/functions/

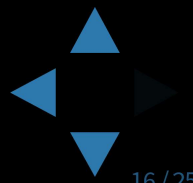
```
@color: #ffffff;
background-color: lighten(@color, 10%);
color: darken(@color, 15%);
percentage(0.5); //returns 50%
```



Telerik Academy

Mixins

- Mixins are kind of developer defined functions
  - The developer can make them for clear LESS
- Two kind of mixins
  - Parameterless
    - Get a default styles every time
  - With parameters
    - Get style based on some parameters
    - Gradient, borders, etc…



Telerik Academy

Mixins

- How to define mixins?
  - Use .mixin-name
  - Then the styles are normal LESS
  - How to use the mixin?
    - Place use .mixin-name

```
.clearfix {
    zoom: 1;
    &:after {
        display: block;
        content: "";
        height: 0;
        clear: both;
    }
}
```

```
ul#main-nav{
  .clearfix;
}
```

Telerik Academy

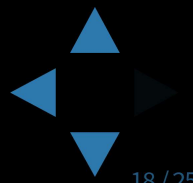
Mixins with Arguments

- Mixins can also be defined with parameters
  - i.e. for gradient-background
- Use the arguments like a C# method

```
.opacity(@value){
    ...
}

//arguments can take default values
.box(@border: none, @bg: rgba(0,0,0,0.7), @size: 200px) {
    ...
}

div.box-div {
    .box; //using the mixin with default parameter values
    .opacity(0.5); //using the mixin with 0.5 value
}
```

Telerik Academy

Pattern-matching Mixins

- Mixins can define different behavior, depending on its parameters

```
.color(dark; @color){
  color: @arken(@color, 25%);
}
.color(light; @color){
  color: lighten(@color, 25%);
}
//called no matter which of the above is called
@color(@_; @color){
  border: 1px solid @color;
}
```

Telerik Academy

Mixin "Return" Values

- Mixins can have return values because of the variable scope

```
.average(@x, @y) {
  @average: ((@x + @y) / 2);
}

div {
  .average(16px, 50px); // "call" the mixin
  padding: @average;    // use its "return" value
}

/* Compiles to */
div {
  padding: 33px;
}
```

Telerik Academy

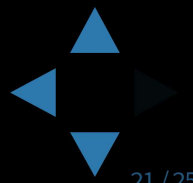
Importing Other Files

- You can import other files wherever you want
  - In our main file
  - In our "other-styles.less" file
  - Results in

```
.foo { background: #900; }
@import "other-styles.less";
```

```
/* other-styles.less */
.foobar { background: #900; }
```

```
.foo { background: #900; }
.foobar { background: #900; }
```



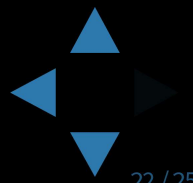
Telerik Academy

Conditional Mixins

- Also called Mixin Guards
  - Using when keyword can specify conditions on which the mixin will be called
  - Operators are >, >=,=, =<, <

```
.mixin (@a) when (lightness(@a) >= 50%) {
    background-color: white;
}

.mixin (@a) when (lightness(@a) < 50%) {
    background-color: black;
}
```



Telerik Academy

Loops

- Because of Mixin Guards you can create loop/iterations structure

```
.loop(@counter) when (@counter > 0) {
    .loop((@counter - 1)); /* next iteration */
    width: (10px * @counter); /* code for each iteration */
}

div {
    .loop(5); /* launch the loop */
}
```