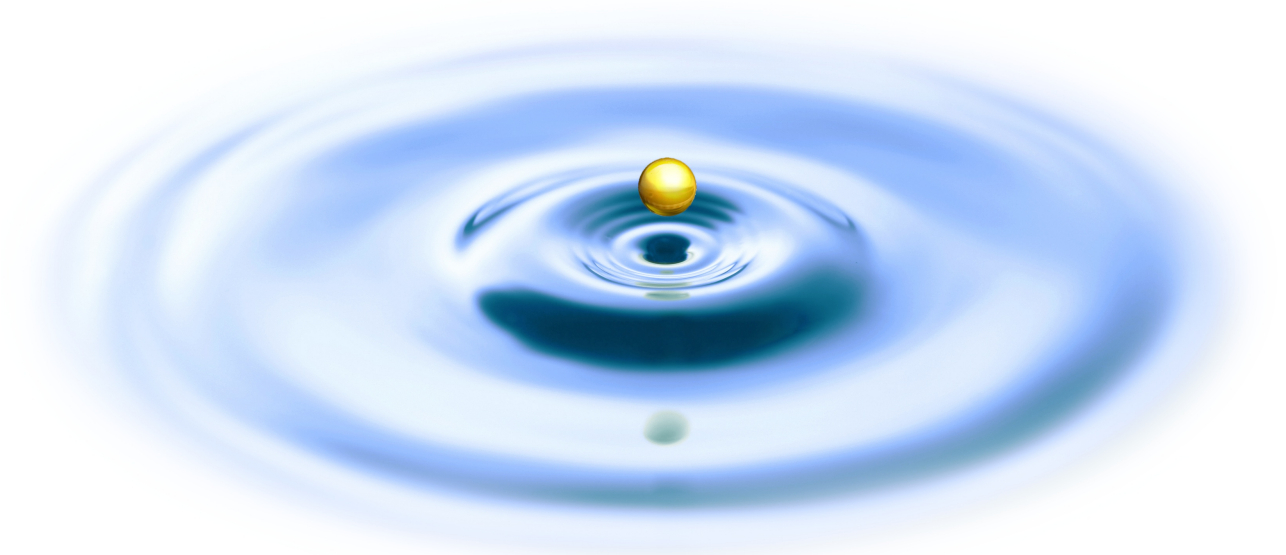
CSS Coding Style Conventions

Standards Guidelines & Rules

Michel Bagnol
www.nodewave.com

Version 1.3
27th May 2009



About this document

Introduction

There are a some thoughts on CSS Coding Conventions spread over the web, but none was complete and straightforward, this document solves this issue by giving you a ready to use, complete and professional CSS Coding Convention.

Audience

Developers using CSS (Cascading Style Sheets).
Project Manager having a several CSS Developers.

Purpose

This document give guidelines to improve development productivity with CSS, this by organizing code and making it very readable, and exchangeable among developers without a loss of productivity while “diving-in”.

Philosophy

The main focus of these guidelines is to ensure:

- Maintainability,
- Readability,
- Size.

Versions

27th May 2009	1.3	Improved tip for reseting of html elements
10th May 2009	1.2	Added Tips
9th May 2009	1.1	Better naming explanation with more examples, Changed multiple selectors to multi-line, Added alphabetical order on multiple properties
16th April 2009	1.0	First release

Author

Michel Bagnol

Terms of use

Feedback

We would really appreciate your feedback & comments, so feel free to send a message to: **m.***no_spam*@**nodewave.com**
(replace « *no_spam* » by « **bagnol** »)

Contents

CSS Coding Style Conventions....1
About this document....2
Introduction....2
Audience....2
Purpose....2
Philosophy....2
Versions....2
Author....2
Terms of use....2
Feedback....2
Contents....3
How to use CSS....4
External....4
Inline....4
Embedded....4
Files....5
Naming....5
Using multiple CSS files....5
Separating concerns....5
Syntax....6
CSS Version....6
Reference....6
Validation....6
Vocabulary....6
Basics....7
Indentation....7
Selectors....7
Multiple Selectors....7
Properties....7
Multiples properties....8
Multi-values properties....9
Shorthand properties....9
Conclusion....10
Styling....11
Commenting....11
File header....11
Capitalization....11
Naming convention....12
Values....14
Colors....14
Units....14
Urls....14
Fonts....14
Organization....16
Grouping....16
Tips....17
Reseting html elements....17
Debugging by showing borders....18
Force vertical scrollbar....18
Complete example....19

How to use CSS

External

Use CSS as an external file, linking the CSS file in your HTML files like hereunder:

```
<html><head>
      <!-- CORRECT WAY -->
      <link rel="stylesheet" href="my_theme.css" type="text/css">
</head><body>
      <span class="myclass">Use external CSS</span>
</body></html>
```

Inline

It is **forbidden to use CSS inline of HTML tags,** by using the “style” attribute in HTML tags

```
<html><body>
      <!-- FORBIDDEN !!! -->
      <span style="color:red;">Inline CSS is forbidden!<span>
</body></html>
```

This looses the main purpose of CSS of separating design (CSS) from content (HTML).

Embedded

It is **forbidden to use CSS embedded in HTML**, defining the “style” tag in the head section of an HTML document.

```
<html><head>
      <!-- FORBIDDEN !!! -->
      <style type="text/css">
      <!--
            .myclass
                  {color: red;}
      -->
      </style>
</head><body>
      <span class="myclass">Embedded CSS is also forbidden!</span>
</body></html>
```

The CSS has to be maintained on each HTML, this is not desirable. Also web-browsers can't cache the CSS separatly, and therefore it will display slower and consume more bandwidth.

Files

Naming

Give the CSS files descriptive names, like:

```
default-theme-html.css
```

Using multiple CSS files

The use of multiple CSS files is done within the HTML file using multiple "<link/>" tags in the header:

```
<!-- HTML File -->
<html><head>
	<!-- CORRECT WAY -->
	<link rel="stylesheet" href="reset.css" type="text/css">
	<link rel="stylesheet" href="blue-theme-html.css" type="text/css">
	<link rel="stylesheet" href="blue-theme-classes.css" type="text/css">
	<link rel="stylesheet" href="blue-theme-ids.css" type="text/css">
</head><body>
	<h1>This is the correct way to use multiple CSS files</h1>
</body></html>
```

It is forbidden to use the "@import" command in the CSS files.

```
/* CSS File */
	/* FORBIDDEN !!! */
	@import url('reset.css')
	@import url('default-theme-html.css')
	@import url('default-theme-classes.css')
	@import url('default-theme-ids.css')
```

Separating concerns

Separating concerns by selector type allows better maintainability, easy style switching and code-reuse. Apply these rules if you have a somewhat big project or if you want to reuse your CSS Styles across projects.

First, make files dedicated for HTML tags:

```
reset.css
default-theme-html.css
blue-theme-html.css
```

Then, make files dedicated to classes, categorized by general functionality, allowing you to keep libraries of useful classes:

```
default-theme-classes-page-layout.css
default-theme-classes-nice-forms.css
```

Then, make files dedicated to Identifiers (Ids), they are very specific to your application:

```
default-theme-ids-dynamic-content.css
```

Finally, make files dedicated to browser specific tweaks (I personally prefer not to have any, I skip away anything that isn't 100% cross-browser compatible):

```
default-theme-browser-ie.css
default-theme-browser-opera.css
default-theme-browser-firefox.css
```

Syntax

CSS Version

Use the latest version of CSS that is “Candidate W3C Recommendation”:
As of 2009-04-16, use CSS 2.1 (http://www.w3.org/TR/CSS21/)
It is forbidden to use “under development” versions.

Reference

The reference documentation for CSS (Cascading Style Sheet) is located here:
http://www.w3schools.com/css
Note: In this document, it is assumed that you are familiar with CSS, and at least with selectors.

Validation

Validate your resulting files with the w3c CSS validator
http://jigsaw.w3.org/css-validator/

Vocabulary

The CSS syntax is made up of three parts: a selector, a property and a value:

```
selector
      {property:value;}
```

Basics

Indentation

- Indentation are done using tabulations only.
- Selectors are indented once (one tabulation from the left and side).
- Properties assignment are indented twice (two tabulation from the left and side).

Selectors

- Please make sure you understand clearly the cascading nature of CSS (refer to the w3school website)
- Keep in mind that you can assign multiple classes to a single element.
- Don't forget that selectors are case sensitive.

Multiple Selectors

- Multiple selectors should each be on a single line, with no space after each comma “,”.
- Group of selectors starts grouping HTML elements, then HTML elements with class, then classes, and finally id's. This with each group ordered alphabetically.

Example:

```
h1,
h2,
h3,
table th,
p.myclass,
.yourclass,
#thisone
      {color: #00FFA0;}
```

Properties

- Properties assignment start the line after the selector.
- The property and it's value are separated by a colon followed by a space “: ”.
- All property assignment end with a semicolon “;”.
- The first property assignment is preceded by an opening curly brace “{”, with no spaces or new line to separate if from the property name.
- The last property assignment is followed by an closing curly brace “}”,

with no spaces or new line to separate if from the semicolon that follows the last properties value.

- Keep properties to the minimum. If there isn't a point to adding a property, don't add it.

Example:

```
h1
	{padding-top: 5px;}
```

Multiples properties

Only one property assignment is written on a line, except for the following cases:

- Independent side properties referring to the same general property (like "margin", "padding", "border-*xxx*-color":

```
h1
	{margin-top: 5px; margin-left 10px;}
```

Note: Side properties are always written in the clockwise order: top, right, bottom, left.

- Dimensions: Height and Width properties (except line-height)

```
h1
	{width: 80%; min-width: 100px; height: 30px;}
```

Note: Max-Width, Width, Min-Width, Max-Height, Height, Min-Height are always written in this order.

In all other cases, only one property assignment is allowed per line (and always indented twice):

```
h1
	{color: black;
	border-bottom-color: black;}
```

Multiple properties should be listed in alphabetical order, like :

```
/* CORRECT */
h1
	{background-color: white;
	border-bottom-color: black;
	color: black;}
```

rather than:

```
/* FORBIDDEN !!! */
h1
	{color: black;
	background-color: white;
	border-bottom-color: black;}
```

The different lines of properties are ordered depending on the content:

- Paddings, dimensions & Margins are ordered from the outer to the inner (padding, dimension, margin):

```
h1
	{margin-top: 5px; margin-left 10px;
	padding-top: 10px;
	width: 580px; height: 48px;}
```

- Alignments & Positioning are ordered from the outer to the inner

(position, float, overflow, vertical-align, text-align, direction, text-indent, white-space):

Multi-values properties

For properties which can have multiple values, each value should be seperated with a comma followed by a space ", ".

```
body
	{font-family: helvetica, sans-serif;}
```

Shorthand properties

Usage of shorthand properties is forbidden except in the following conditions:

- Setting all four sides of a property to the same value:

```
h1
	{border-width: 1px;
	margin: 5px;
	padding: 0px;}
```

Note: It is forbidden to set four sides of a property to different values:

```
h1	/* FORBIDDEN !!! */
	{margin: 0px 5px 10px 5px;}

h1	/* CORRECT */
	{margin-top: 0px; margin-right: 5px; margin-bottom: 10px; margin-
left: 5px;}
```

- Setting border width and style, always width followed by style:

```
h1
	{border: 1px dotted;
	border-bottom: 2px solid;}
```

Note: You may not use the "border" shorthand to set the border-color property.

- Setting background image

```
h1
	{background: url('img/picture.png') no-repeat top left;}
```

Note: You may not use the "background" shorthand to set the background-color property.

- Setting the list-type

```
ul li
	{list-style: url('img/list-item.png') outside none;}
ol li
	{list-style: outside decimal;}
```

It is always forbidden to have several properties set on a single line if a shorthand property is used.

```
h1	/* FORBIDDEN !!! */
	{margin: 0px; margin-top: 5px; margin-left: 10px;}

h1	/* CORRECT */
	{margin: 0px;
	margin-top: 5px; margin-left: 10px;}
```

It is forbidden to use the other shorthand properties (like: “font”, “list-style”, “outline”)

Conclusion

Good syntax

```
/* GOOD SYNTAX */
h1,
h2,
table th,
.yourclass,
#thisone
      {font-family: arial;
      font-weight: bold;}
```

Bad syntax

```
/* BAD SYNTAX */
#thisone,h2,table th,h1 ,.yourclass{
font:Arial
      BOLD
}
```

Styling

Commenting

- Avoid commenting CSS file more than described in this guidelines. CSS isn't a programming language and is straightforward to read when organized properly.
- Avoiding comment will save bandwidth and will also help readability. Only tricky parts should be commented.
- When commenting a trick, comment after the semicolon, separating the comment by a single space:

```
html
      {height: 100%;
      margin-bottom: 1px; /* force vertical scrollbar */}
```

- Avoid multi-line comments.

File header

Use the following file header

```
/*    Filename:         stylesheet.css
 *    Description:      default CSS theme for HTML pages
 *
 *    Last revision:
 *    $Revision: 3 $
 *    $Date: 2009-05-27 21:42 $
 *    $LastChangedBy: michel.bagnol $
 *    $HeadURL: http://svn.nodewave.com/repos/trunk/stylesheet.css $
 */
```

Using the dollars “$” and appropriate keywords allows automatic update by subversion source code versionning system in the section under “Last revision”.

Capitalization

Everything is written lowercase except:

- Name of block in comment, written entirely uppercase:
  `/* ===== COLORS ===== */`
- Name of blocks sections in comment, written with first letter uppercase:
  `/* --- White --- */`
- Hexadecimal values, written entirely uppercase:
  `#00A0FF`

In other cases, lowercase is required for XHTML compatibility.

Naming convention

- Compose names using hyphen “-” separator. It is the same separator than the CSS language.
  For example, use composed names like:
  ```
  /* CORRECT */
  sales-order
  ```
  and not composed names like:
  ```
  /* FORBIDDEN !!! */
  salesOrder,
  SalesOrder,
  sales_order
  ```

- Use full English descriptions for names. Avoid using abbreviations.
  For example, use names like:
  ```
  /* CORRECT */
  .first-name,
  .last-name,
  .middle-initial
  ```
  rather than the shorter versions:
  ```
  /* FORBIDDEN !!! */
  .f-name,
  .last-n,
  .m-i
  ```

- Choose semantic names based on functionality, not on apperance or position.
  In other words, choose meaningful names that describe what the class is being used for, and not expressing it's representation or some relative position to other elements.
  For example, use functional names like:
  ```
  /* CORRECT */
  .budget-debit
     {color: #FF0000;}
  .budget-credit
     {color: #00FF00;}
  .main-content
     {color: black;}
  .main-navigation
     {color: white;
     background-color: black;}
  .copyright
     {color: #AAAAAA;}
  ```
  and do not use appearance names like:
  ```
  /* FORBIDDEN !!! */
  .red
     {color: #FF0000;}
  .green
     {color: #00FF00;}
  .left-content
     {color: black;}
  .right-sidebar
     {color: white;
     background-color: black;}
  .footer
     {color: #AAAAAA;}
  ```

Functionality based names are more change friendly, makes it easier to understand what you're changing from the CSS point of view, and as the design should not be taken into account from the HTML point of view it confirms this architecture.

- Avoid unnecessarily long names.
  For example, use shortened names like:
  ```
  /* CORRECT */
  .article-description
  ```
  and do not use long names like:
  ```
  /* FORBIDDEN !!! */
  .description-of-the-article
  ```
  If you feel that you cannot clearly express the meaning of a class or identifier within a reasonable name (up to about 20 characters), you should think about restructuring your code.

- Avoid names that are very similar.
  For example, if you name a class perishable-products, it may be better to refer to a particular product with naming the identifier selected-perishable-product (or this-product or product-1) like:
  ```
  /* CORRECT */
  .perishable-product
     {font-weight: normal;}
  #selected-perishable-product
     {font-weight: bold;}
  ```
  rather than perishable-product to avoid confusion:
  ```
  /* FORBIDDEN !!! */
  .perishable-product
     {font-weight: normal;}
  #perishable-product
     {font-weight: bold;}
  ```

Values

Colors

All color values are written in the Hexadecimal format and using uppercase:

```
#00FFA0
```

Except for Black, White and Transparent which are written plain text in lower case:

```
white
```

Note: The usage of all others color-names in plain text is forbidden (aqua, blue, fuchsia, green, lime, maroon, navy, olive, purple, red, silver, teal, and yellow), the only exception to the Hexadecimal format is “black”, “white” and “transparent”.

Units

Measures are used mostly for font-sizes and positioning.

- Prefer measurement units which are re-sizeable by browsers: “%” or “em”.
  See the CSS Techniques for Web Content Accessibility Guidelines 1.0 for more details.
- For precision positioning (relative image positions) use: “px”.
  It is forbidden to use “pt”, use “px” instead.

Urls

All url are written surrounded by quotes “'”

```
table th
      {background: url('img/table-header.png') repeat-x;}
```

Fonts

To choose fonts, refer to statistical study of the most common fonts on Windows, Mac and Linux in the following site:

http://www.codestyle.org/css/

Build font stacks with Windows fonts first, then Macintosh fonts, then Linux fonts and finaly always generic font family.

Font names with spaces must surrounded by double-quotes “"”.

Here are some proposal of font stacks, one for each generic font family:

- Serif font stack proposal:

```
body
      { font-family: "Palatino Linotype", "New Century Schoolbook",
```

```
Georgia, "Century Schoolbook", "Times New Roman", Garamond, "DejaVu Serif",
times, serif; }
```

- Sans-Serif font stack proposal:

```
body
      {font-family: "Trebuchet MS", Verdana, Helvetica, Arial, "Bitstream
Vera Sans", "DejaVu Sans", sans-serif;}
```

- Monospace font stack proposal:

```
body
      {font-family: "Lucida Console", "Bitstream Vera Sans Mono",
"Consolas", "Lucida Sans Typewriter", "DejaVu Sans Mono", "courier new",
courier, monospace;}
```

- Fantasy font stack proposal (be aware that this font stack renders very differently depending on os and browser):

```
body
      { font-family: "Comic Sans MS", Purisa, Papyrus, "URW Chancery L",
Impact, fantasy; }
```

- Cursive font stack proposal (be aware that this font stack renders very differently depending on os and browser):

```
body
      {font-family: "Lucida Handwriting", "Brush Script MT", "Apple
Chancery", "Monotype Corsiva", "URW Chancery L", chancery, script, handwriting,
comic, cursive;}
```

Organization

Grouping

Group kinds of definitions in blocks separated by comments describing the group.

This grouping allows easy update of styling, for example when changing colors or fonts, adjusting sizes or alignments,... You will also find it much easier to concentrate on what you're focusing on.

- A block for all color definitions, grouped by color and then by attribute type:
```
/* ===== COLORS ===== */
      /* --- White --- */
      /* --- Gray --- */
      /* --- Black --- */
      /* --- Mycolor #004966 --- */
```
- A block for all images and list-style definitions:
```
/* ===== IMAGES ===== */
```
- A block for all fonts (except colors) definitions, with sections for different kinds of text transformations:
```
/* ===== FONTS ===== */
      /* --- Font-family and Font-stretch --- */

      /* --- Font-size --- */
      /* ordered from the biggest to smallest */
      /* Note: usage of font-size-adjust is forbidden */

      /* --- Line-height & Word-spacing & Letter-spacing --- */
      /* ordered from the biggest to smallest */

      /* --- Font-variant & Text-transform --- */

      /* --- Font-weight & Font-style & Text-decoration --- */
```
- A block for all alignments and positioning definition:
```
/* ===== ALIGNMENTS & POSITIONING ===== */
```
- A block for all border styling (except colors) definitions:
```
/* ===== BORDERS ===== */
```
- A block for all sizes & spacing definitions:
```
/* ===== PADDINGS & DIMENSIONS & MARGINS ===== */
```

Tips

Reseting html elements

Browsers have different default CSS files for the HTML tags, in order to keep things looking the same across browsers you can use a css file to reset all values which are browser dependent, so that all browsers will react the same with your css.

Make sure to declare this css before your other css in the HTML file.

The following proposal was inspired from http://developer.yahoo.com/yui/reset/

```
/*    Filename:         reset.css
 *    Description:      reset HTML CSS styles which have different default
across browsers
 *
 *    Last revision:
 *    $Revision: 1 $
 *    $Date: 2009-05-09 21:42 $
 *    $LastChangedBy: michel.bagnol $
 *    $HeadURL: http://svn.nodewave.com/repos/trunk/reset.css $
 */

/* ===== IMAGES ===== */
      ol,
      ul
            {list-style:none;}

/* ===== FONTS ===== */
      /* --- Font-size --- */
      address,
      caption,
      cite,
      code,
      dfn,
      em,
      strong,
      th,
      var
            {font-style:normal;
            font-weight:normal;}

      h1,
      h2,
      h3,
      h4,
      h5,
      h6
            {font-size:100%;
            font-weight:normal;}

/* ===== ALIGNMENTS & POSITIONING ===== */
      caption,
      th
            {text-align:left;}

/* ===== BORDERS ===== */
      table
```

```
            {border-collapse:collapse;
            border-spacing:0;}

      abbr,
      acronym,
      fieldset,
      img
            {border:0;}

/* ===== PADDINGS & DIMENSIONS & MARGINS ===== */
      body,
      blockquote,
      dd,
      div,
      dl,
      dt,
      form,
      fieldset,
      h1,
      h2,
      h3,
      h4,
      h5,
      h6,
      input,
      li,
      ol,
      p,
      pre,
      td,
      textarea,
      th,
      ul
            {margin:0;
            padding:0;}

/* ===== MISCELLANEOUS ===== */
      q:before,
      q:after
            {content:'';}
```

Debugging by showing borders

Use this only for debugging purposes:

```
      * {border: 1px solid #ff0000;}
```

Note: Write this on a single line, so it's easier to comment/uncomment. (I know this does not comply to this coding convention, it's the only exception...)

Force vertical scrollbar

It allows you to have the same page width on all pages, even on pages with not enough content to fill the page.

```
      html
            {height: 100%;
            margin-bottom: 1px; /* force vertical scrollbar */}
```

Complete example

```
/*    Filename:         stylesheet.css
 *    Description:      default CSS theme for HTML pages
 *
 *    Last revision:
 *    $Revision: 3 $
 *    $Date: 2009-05-09 21:42 $
 *    $LastChangedBy: michel.bagnol $
 *    $HeadURL: http://svn.nodewave.com/repos/trunk/stylesheet.css $
 */

/* DEBUG
      * {border: 1px solid #ff0000;}
*/

/* ===== COLORS ===== */
/* put all color definitions here, grouped by color and then by attribute type
*/
      /* --- White --- */
      code,
      body,
      table
            {background-color: white;}

      /* --- Gray --- */
      table td
            {border-color: #0F0F0F;}

      /* --- Black --- */
      body
            {color: black;}
      table,
      table th
            {border-color: black;}

      /* --- Mycolor #004966 --- */
      h1,
      h2,
      h3
            {color: #004966;
            border-color: #004966;}

/* ===== IMAGES ===== */
/* put all images and list-style definitions here */
      table th
            {background: url('img/table-header.png') repeat-x;}
      ul li
            {list-style: url('img/list-item.png') outside none;}
      ol li
            {list-style: outside decimal;}

/* ===== FONTS ===== */
/* put all fonts (except colors) definitions here */
      /* --- Font-family and Font-stretch --- */
      body
            {font-family: "Trebuchet MS", Verdana, Helvetica, Arial, "Bitstream
Vera Sans", "DejaVu Sans", sans-serif;}

      /* --- Font-size --- */
```

```
	/* ordered from the biggest to smallest */
	/* Note: usage of font-size-adjust is forbidden */
	h1
		{font-size: 130%;}
	h2
		{font-size: 120%;}
	h3,
	th
		{font-size: 110%;}
	body
		{font-size: 100%;}

	/* --- Line-height & Word-spacing & Letter-spacing --- */
	/* ordered from the biggest to smallest */
	ul li
		{line-height: 90%;}

	/* --- Font-variant & Text-transform --- */
	h1
		{text-transform: uppercase;}
	h2,
	th
		{font-variant: small-caps;}

	/* --- Font-weight & Font-style & Text-decoration --- */
	a
		{text-decoration: underline;}
	a:hover
		{text-decoration: overline underline;}
	h1,
	h2,
	h3,
	#header-title
		{font-weight: bold;}
	th
		{font-style: italic;}

/* ===== ALIGNMENTS & POSITIONING ===== */
/* put all alignments and positioning definitions here */
	table td
		{vertical-align: top;}
	td
		{text-align: left;}

/* ===== BORDERS ===== */
/* put all border styling (except colors) definitions here */
	/* --- No border --- */
	body
		{border: 0px;}

	/* --- Standard border --- */
	table
		{border-collapse: collapse;}
	table,
	table td,
	table th
		{border: 1px solid;}

	/* --- Titles border --- */
	h1,
	h2
		{border-bottom: 1px solid;}
```

```
/* ===== PADDINGS & DIMENSIONS & MARGINS ===== */
/* put all sizes & spacing definitions here */
      html
            {height: 100%;
            margin-bottom: 1px; /* force vertical scrollbar */}
      body
            {height: 100%; width: 100% !important;}
      h1
            {margin-top: 2em;}
      h2
            {margin-top: 1.5em;}
      h3
            {margin-top: 1.2em;}
      h4
            {margin-top: 1em;}
      ul li
            {padding-left: 1em;}
```