CSS Reset

CSS Reset

- *CSS reset*
  - Aplicação de um conjunto de regras CSS que inviabilize a aplicação de estilos por defeito (browsers) de forma a tornar o ponto de partida tão neutro quanto possível.
    - Existem disponíveis vários tipos de CSS *reset*

http://cssreset.com/

CSS Reset

- O *CSS reset* deve ser copiado para o inicio da CSS de forma a garantir que é efetuado de forma correta.
  - Forma simples de evitar inconsistências de formatação dos diversos browsers

```
html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary,
time, mark, audio, video {
        margin: 0;
        padding: 0;
        border: 0;
        font-size: 100%;
        font: inherit;
        vertical-align: baseline;
}
```

```
/* HTML5 display-role reset for older browsers */
article, aside, details, figcaption, figure,
footer, header, hgroup, menu, nav, section {
        display: block;
}
body {
        line-height: 1;
}
ol, ul {
        list-style: none;
}
blockquote, q {
        quotes: none;
}
blockquote:before, blockquote:after,
q:before, q:after {
        content: '';
        content: none;
}
table {
        border-collapse: collapse;
        border-spacing: 0;
}
```

http://meyerweb.com/eric/tools/css/reset/

Normalize.css

Normalize.css

A modern, HTML5-ready alternative to CSS resets

"Normalize.css makes browsers render all elements more consistently and in line with modern standards. ***It precisely targets only the styles that need normalizing.*** *"*

```
/*! normalize.css v8.0.1 | MIT License | github.com/necolas/normalize.css */

/* Document
   ========================================================================== */

/**
 * 1. Correct the line height in all browsers.
 * 2. Prevent adjustments of font size after orientation changes in iOS.
 */

html {
  line-height: 1.15; /* 1 */
  -webkit-text-size-adjust: 100%; /* 2 */
}

/* Sections
   ========================================================================== */

/**
 * Remove the margin in all browsers.
 */

body {
  margin: 0;
}
```

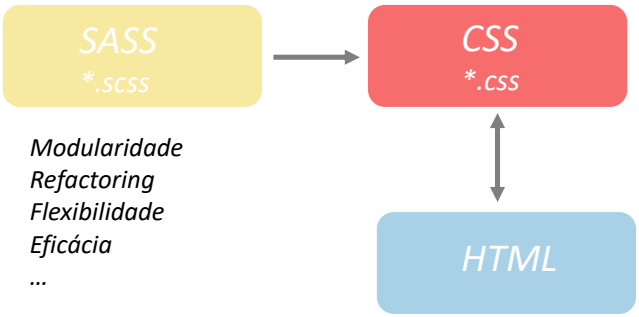
https://necolas.github.io/normalize.css/8.0.1/normalize.css

Sass

Syntactically Awesome Style Sheets

Sass

- Pré-processador código CSS
  - facilita a manutenção de código CSS, melhora a organização do código, mais escalável,…
  - permite a utilização de *features* não previstas na especificação CSS:
    - variáveis, reutilização, ….
  - ficheiros de extensão *.scss a partir dos quais são criados os ficheiros *.css a ligar ao HTML (HTML **não reconhece** *.scss)



Modularidade
Refactoring
Flexibilidade
Eficácia
…

Sass

- Instalação do Sass

```
C:\>gem install sass
```

RubyInstaller for Windows

http://rubyinstaller.org/downloads/

- Existem duas formas distintas de compilar código *.scss

http://sass-lang.com/install

  - Diretamente através da linha de comando

```
C:\>sass input.scss output.css
```

  - Tendo por base uma aplicação
    - Existem várias aplicações que podem ser utilizadas para a compilação de Sass
      - *Sublime Text 3 (SASS extension); Brackets (Brackets SASS), Visual Studio Code, …*

Sass

- Diferenças de Sintaxe
  - *.sass ***VS.*** *.scss
    - *.sass é relativo à primeira versão do Sass, cuja sintaxe era significativamente diferente das CSS
      - Indentação em vez de chavetas e não são necessários "**;**" a separar as propriedades…
      - Esta sintaxe diferente não contribuiu para a generalização do uso deste pré-processador CSS
    - *.scss foi introduzido na última versão das CSS (CSS3).
      - Baseado na mesma sintaxe que as CSS
      - Disponibiliza um conjunto de novas *features* não disponíveis nas CSS
    - As duas sintaxes são reconhecidas, ainda que a *.scss seja atualmente a mais utilizada tanto mais que segue os mesmos princípios de sintaxe do que a CSS (uso de **{ }** e **;**)
      - A sintaxe utilizada nestes slides é *.scss

Sass: Características Principais

- Variáveis
  - permite a declaração de variáveis, as quais são precedidas de **$**
    - Tipos de dados:
      - numbers
      - strings
      - colors
      - null
      - lists
      - maps

```
$margin-values: 1px 2px 3px 4px;
```

```
$map: (
    key: value,
    nextkey: nextvalue
);
```

  - As variáveis devem:
    - possuir um significado semântico
    - seguir um padrão/convenção (ex: prefixos “header-” “footer-” “section-”, ou sufixos “-color”)

Sass: Características Principais

- Variáveis
  - Importantes quando o mesmo valor é utilizado em vários sítios
  - Muito importantes para a estruturação e manutenção do código, principalmente em projetos de grande dimensão

variaveis.scss

```
$font-stack:    Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```

```
SASS - Compressed
SASS
```

variaveis.css

```
body {
  font: 100% Helvetica, sans-serif;
  color: #333; }
```

Sass: Características Principais

- *Scope* das variáveis
  - Uma variável declarada num seletor é visível apenas nesse seletor.

variaveis.scss

variaveis.scss

```scss
$primaryColor: darkorange;

body {
  $primaryColor: lightgray;
  background-color: $primaryColor;
}

p {
  width:200px;
  color:white;
  background-color: $primaryColor;
}
```

```
<body>
    <p>VARIABLES SCOPE</p>
</body>
```

variaveis.css

variaveis.css

```css
body {
  background-color: lightgray; }

p {
  width: 200px;
  color: white;
  background-color: darkorange; }
```

VARIABLES SCOPE

Sass: Características Principais

- Scope das variáveis
  - **!global** permite declarar uma variável global num seletor mas que é visível em todo o código

variaveis.scss

variaveis.scss

```scss
$primaryColor: darkorange;

body {
  $primaryColor: lightgray !global;
  background-color: $primaryColor;
}

p {
  width:200px;
  color:white;
  background-color: $primaryColor;
}
```

```
<body>
    <p>VARIABLES SCOPE</p>
</body>
```

variaveis.css

variaveis.css

```css
body {
  background-color: lightgray; }

p {
  width: 200px;
  color: white;
  background-color: lightgray; }
```

VARIABLES SCOPE

Sass: Características Principais

- *Nesting*
  - Permite a criação de uma clara hierarquia visual, declarando diretamente os seletores enquadrados nos seletores que definem o contexto
    - Seletores contextuais em que o 1º nível de contexto é dado pelo ***nest*** que enquadra os outros seletores
    - A extensão desta dependência de seletores resulta em seletores cada vez mais específicos

sass_1.scss

sass_1.scss

```scss
$primary-color: lightgray;
$width-element: 300px;

nav {
  ul {
    margin: 0;
    padding: 0;
    list-style: none;
  }

  li { display: inline-block; }

  a {
    display: block;
    padding: 6px 12px;
    text-decoration: none;
  }
}
```

sass_1.css

sass_1.css

```css
nav ul {
  margin: 0;
  padding: 0;
  list-style: none; }
nav li {
  display: inline-block; }
nav a {
  display: block;
  padding: 6px 12px;
  text-decoration: none; }
```


Sass: Características Principais

- *@import / Partials*
  - Sass permite importar pequenos blocos de código **.scss (partials)*
    - *Partial* é um ficheiro *.scss cujo nome inicia com *underscore*.
      - O *underscore (_)* indica que se trata de um *partial* (a ser importado por outro ficheiro *.scss e não convertido diretamente num *.css)
    - Os *Sass partials* são usados com a diretiva @import (só é especificado o nome do *partial*)
    - Torna o código mais modular (reset, variáveis, funções, …) e consequentemente mais fácil de manter.

_reset.scss

_reset.scss

```scss
html,
body,
ul,
ol {
   margin: 0;
  padding: 0;
}
```

sass_1.scss

```scss
@import 'reset';

body {
  font: 100% Helvetica, sans-serif;
  background-color: #efefef;
}
```

sass_1.css

sass_1.css

```css
html,
body,
ul,
ol {
  margin: 0;
  padding: 0; }

body {
  font: 100% Helvetica, sans-serif;
  background-color: #efefef; }
```

Sass: Características Principais

- *@mixin*
  - Conjunto de declarações CSS que se pretende reutilizar
  - A criação do grupo de declarações CSS é baseado na diretiva **@mixin**
    - É obrigatório atribuir um nome ao @mixin
    - É possível a passagem de parâmetros ao @mixin
      - O parâmetro deve ser definido com ($nome_parâmetro)
    - @mixin é usado com base na diretiva **@include**
    - Os @mixin podem ser agrupados num *partial* de forma a libertar a *.scss principal

sass_1.scss

sass_1.scss

```
@mixin paragraph($radius) {
    background-color: orange;
    color:white;
    width: 200px;
    height:100px;
    text-align:center;
    padding:10px;
    border-radius: $radius;
}

p {@include paragraph(10px); }
```

sass_1.css

sass_1.css

```
p {
  background-color: orange;
  color: white;
  width: 200px;
  height: 100px;
  text-align: center;
  padding: 10px;
  border-radius: 10px; }
```


Sass: Características Principais

- *Inheritance (@extend)*
  - A diretiva **@extend** permite a partilha de declarações CSS entre diferentes seletores
    - Minimiza a repetição de código em diferentes seletores

sass_1.scss

```
.global {
  border: 4px solid darkorange;
  width:50px;
  height:30px;
  padding: 10px;
  text-align:center;
  background-color:darkorange;
  color: white;
}

.first {
  @extend .global;
  border-color: lightgray;
}

.second {
  @extend .global;
  border-color: orange;
}

.third {
  @extend .global;
  border-color: darkgray;
}
```

sass_1.css

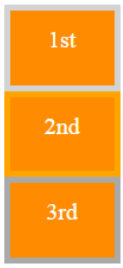
```
.global, .first, .second, .third {
  border: 4px solid darkorange;
  width: 50px;
  height: 30px;
  padding: 10px;
  text-align: center;
  background-color: darkorange;
  color: white; }

.first {
  border-color: lightgray; }

.second {
  border-color: orange; }

.third {
  border-color: darkgray; }
```

```
<body>
    <div class="first">1st</div>
    <div class="second">2nd</div>
    <div class="third">3rd</div>
</body>
```



Sass: Características Principais

- @extend
  - cada @extend pode ser aplicado a apenas um seletor

*.scss

```
.error{
    color:red;
}

.critical-error{
    @extend .error;
    font-weight:bold;
    border:2px solid red;
}
```

```
.minor-error{   background-color:gray;}

.critical-error{
    @extend .error  .minor-error;
    font-weight:bold;
    border:2px solid red;}
```


  - múltiplos @extend no mesmo seletor

```
.critical-error{
    @extend .error;
    @extend .minor-error;
    font-weight:bold;
    border:2px solid red;}
```

  - encadeamento de @extend

```
.main-error{
    @extend .critical-error;
    font-size: 125%;}
```


Sass: Características Principais

- @extend
  - É possível definir um seletor exclusivamente para ser utilizado com *@extend*
    - Iniciam-se por %
    - Não podem ser compilados diretamente para CSS, só são compilados se forem alvo de um @extend

*.scss

```
%error{color:red;}

.critical-error{
    @extend %error;
    font-weight:bold;
    border:2px solid red;}
```

*.css

```
.critical-error {
      color: red; }

.critical-error {
    font-weight: bold;
    border: 2px solid red; }
```

    - O @extend de um seletor inexistente, provoca um erro e inibe a geração do respetivo *.ccs
      - **!optional** permite contornar esta limitação

```
.critical-error{
    @extend %error;
    @extend %critical !optional;
    font-weight:bold;
    border:2px solid red;}
```

Sass: Características Principais

- *@mixin vs. @extend*

@mixin	@extend
reutilização código	reutilização código
permite passagem de parâmetros	não permite parâmetros
Origina menos seletores CSS	Origina mais seletores CSS
repetição de declarações	minimiza repetição de declarações
	limitação no funcionamento com @media
tipicamente **mais rápida** que o @extend	

https://tech.bellycard.com/blog/sass-mixins-vs-extends-the-data/

Sass: Características Principais

- *@media*
  - Permite a definição de CSS media queries
  - Possibilita o *nest* das CSS *media queries*, no entanto no código *.css a definição das *media queries* é sempre efetuada no top level

*.scss

```
#main {
    width:$content-width;
    @media only screen and (max-width: 960px){
        width:auto;
        max-width:960px;
    }
```

*.css

```
@media only screen and (max-width: 960px) {
    #main {
      width: auto;
      max-width: 960px; } }
```

Sass: Características Principais

- *@media*
  - Não é possível efetuar o @extend de seletores definidos fora da media query

```
@media screen{

    .error{color:red;}

    .critical-error{
        @extend .error;
        font-weight:bold;
        border:2px solid red;}
}
```


Sass: Características Principais

- Operadores
  - Conjunto de operadores matemáticos:
    - +, -, *, / , % …

sass_1.scss

```
.container { width: 100%; }

@mixin global{
  color:white;
  text-align:center;
  height:50px;
}

article[role="main"] {
  float: left;
  width: 600px / 960px * 100%;
  background-color:lightgray;
  @include global;
}

aside[role="complementary"] {
  float: right;
  width: 300px / 960px * 100%;
  background-color:darkorange;
  @include global;
}
```

sass_1.css

```
@charset "UTF-8";
.container {
  width: 100%; }

article[role="main"] {
  float: left;
  width: 62.5%;
  background-color: lightgray;
  color: white;
  text-align: center;
  height: 50px; }

aside[role="complementary"] {
  float: right;
  width: 31.25%;
  background-color: darkorange;
  color: white;
  text-align: center;
  height: 50px; }
```

```
<body>
    <article role="main">Main</article>
    <aside role="complementary">Aside</aside>
</body>
```

Sass: Características Principais

- Operadores

operador	*obs.*
+	adição
-	subtração
/	divisão
*	multiplicação
%	resto divisão inteira
==	igual
!=	diferente

https://scotch.io/tutorials/getting-started-with-sass

sass_1.scss

```
$container-width: 100%;

.container {
  width: $container-width;
}

.col-4 {
  width: $container-width / 4;
  height:60px;
  background-color:darkorange;
  color:white;
}
```

sass_1.css

```
@charset "UTF-8";
.container {
  width: 100%; }

.col-4 {
  width: 25%;
  height: 60px;
  background-color: darkorange;
  color: white; }
```

```
    <link href="sass_1.css" rel="stylesheet" type="text/css" />
</head>
<body>
    <div class="col-4">Mathematical Operators</div>
</body>
```

Sass: Características Principais

- *Funções*
  - *Dois tipos de Funções: Built-in functions + Custom Functions*
  - *Built-in Functions:*
    - Conjunto de funções disponibilizadas pelo SASS http://sass-lang.com/documentation/Sass/Script/Functions.html
    - Algumas das funções mais comuns:
      - *darken($color, amount)*
        - permite escurecer uma cor, tem a grande vantagem de já se conhecer a cor a escurecer

```
a    {
color:$link-color;
&:hover{
    color:darken($link-color,15%);}
}
```

      - *lighten($color, amount)*
        - O efeito contrário, torna uma cor mais clara
      - *transparentize($color; amount)*
      - *opacify($color;amount)*

Sass: Características Principais

- *Custom Funtions*
  - @function
    - Permite múltiplos parâmetros (possibilita especificação dos valores por defeito)
    - @return para retornar o valor

```
@function col-width($columns:12, $page-width:100%, $gap:1%){
    @return ($page-width - $gap*($columns - 1))/$columns;
}
```

sass_1.scss

- Chamada à função

```
#content {
    float:left;
    width:6*col-width(8);
    }

#sidebar {
    float:right;
    width:2*col-width(8);
    }
```

sass_1.scss

Sass: Características Principais

- Condições
  - @if (…) {…} / @else if (…){…} / @else {…}

.scss

```
$contrast:high;

body {
    @if ($contrast==high)
        {   color:black;    }
    @else if ($contrast==low)
        {   color:#DDD; }
    @else{
        color:$text-color;
    }
    font-family: sans-serif;
    ...
```

.css

```
body {  color: black;
        font-family: sans-serif; }
```

```
$contrast:low;
```

.css

```
body {  color: #DDD;
        font-family: sans-serif; }
```

Sass: Características Principais

- @if
  - Podem condicionar de uma forma muito direta toda a formatação (ex.: palete de cores)

```
// Allowed values theme: light; default
$theme:default;

$text-color:#222222;
$theme-color:#170a48;
$secondary-color:#f27731;
$ternary-color:#ccf962;
$menu-item-color:white;
$link-color:$secondary-color;

@if ($theme==light)
{
    $text-color:#d1bec2;
    $theme-color:#2b5e82;
    $secondary-color:#4b65c3;
    $ternary-color:#ccf962;
    $menu-item-color:white;
    $link-color:$secondary-color;
}
```

Welcome to My Site!

Welcome to My Site!

Sass: Características Principais

- Ciclos
  - @for
    - Repetir o processamento um número pré-determinado de vezes
    - É possível utilizar variáveis em nomes de seletores, propriedades e respetivos valores recorrendo à notação **#{}** *(interpolation syntax)*

```
@for $i from 1 through 6 {
    .col-#{$i}{
        width:$i*2em;
    }
}
```

**.scss*

```
.col-1 {
  width: 2em; }

.col-2 {
  width: 4em; }

.col-3 {
  width: 6em; }

.col-4 {
  width: 8em; }

.col-5 {
  width: 10em; }

.col-6 {
  width: 12em; }
```

**.css*

Sass: Características Principais

- Ciclos
  - @each
    - O mais versátil e o mais frequentemente utilizado
      - Percorrer uma lista

*.scss

```
$players: ruben-dias, ljubomir-fejsa, andrija-zivkovic;

@each $player in $players {
    .#{$player}-profile{
        background-image:url('img/#{$player}.png')
    }
}
```

*.css

```
.ruben-dias-profile {
  background-image: url("img/ruben-dias.png"); }

.ljubomir-fejsa-profile {
  background-image: url("img/ljubomir-fejsa.png"); }

.andrija-zivkovic-profile {
  background-image: url("img/andrija-zivkovic.png"); }
```


Sass: Características Principais

- Ciclos
  - @each
    - O mais versátil e o mais frequentemente utilizado
      - Percorrer um map

*.scss

```
$font-sizes: (tiny:8px, small:11px, medium: 13px, large:16px);

@each $name, $size in $font-sizes{
    .#{$name}{
        font-size:$size;
    }
}
```

*.css

```
.tiny {
  font-size: 8px; }

.small {
  font-size: 11px; }

.medium {
  font-size: 13px; }

.large {
  font-size: 16px; }
```

Sass: Características Principais

- Ciclos
  - *@while*
    - Exige que a variável que controla o ciclo seja incrementada, caso contrário origina-se um ciclo infinito

```
$j:2;

@while ($j<=8){
    .picture-#{$j}{
        width:$j * 10%;
    }
    $j:$j+2;
}
```

.scss

```
.picture-2 {
  width: 20%; }

.picture-4 {
  width: 40%; }

.picture-6 {
  width: 60%; }

.picture-8 {
  width: 80%; }
```

.css

Sass

- Apresentadas **apenas** as principais características deste pré-processador
  - Documentação
    - http://sass-lang.com/documentation/file.SASS_REFERENCE.html#syntax
  - Vantagens
    - Organização/Modularidade do código
    - Refactoring
    - CSS mais "limpas"
    - Funcionalidades não disponíveis nas CSS (variáveis; reutilização de código; funções; operações matemáticas; ...)
    - ...
  - Desvantagens
    - Curva de aprendizagem para implementar as novas features
    - ...