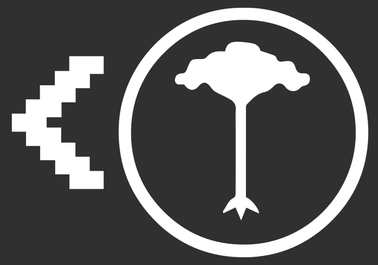
TRILHA FRONTEND

UX HTML CSS JS



Dia 3

- CSS
- Pré-processadores
- Boas práticas
- Frameworks
- Ferramentas

Hands On
- Aplicar CSS na estrutura HTML do dia 2

Desafio
- Vai parar por aqui?

CSS - O que é?

CSS (Cascading Style Sheets) é uma linguagem de estilo usada para descrever a apresentação de um documento escrito em HTML ou em XML.

O CSS descreve como elementos são mostrados na tela.

Foi desenvolvido para habilitar a separação do conteúdo e formato de um documento de sua apresentação, incluindo elementos como cores, formatos de fontes, layout, entre outros.

CSS - Criadores

Håkon Wium Lie, convidou Bert Bos e juntos resolveram criar um jeito mais fácil para formatar a informação cada vez mais complexa no HTML.

Em 1995 eles apresentaram sua proposta, o W3C se interessou pelo projeto e resolveu criar uma equipe, liderada por Håkon e Bert Bos.

O resultado apareceu logo. Em 1996, eles lançaram a recomendação oficial pelo W3C do CSS Level 1 (CSS 1).

CSS - Sintaxe

O objetivo básico do CSS é permitir que um motor de navegador renderize os elementos na página com características específicas.

Para isso precisamos escrever de acordo com a seguinte sintaxe:

```
seletor {
    propriedade: valor;
}
```

```
seletor1,
seletor2 {
    propriedade1: valor1;
    propriedade2: valor2;
}
```

CSS - Seletores

Seletores são "padrões" ou "modelos" que casam com os elementos do DOM, portanto podem ser usados para selecionar os nós de um documento HTML.

Tipos de seletores:

- Universal
- Tipo
- Atributo
- Classe
- Id
- Pseudo-classe
- Pseudo-elemento
- Filhos
- Adjacentes
- Descendentes

CSS - Seletor Universal

Estiliza todos os elementos da marcação.

```
* {
    margin: 0;
    padding: 0;
}
```

CSS - Seletor de tipo

Estiliza os elementos da marcação que são de um mesmo tipo.

```
p {
    font-size: 12px;
}
```

No HTML:

```
<p>O tamanho da minha fonte é 12px</p>
<p>A minha também!</p>
```

CSS - Seletor de atributo

Estiliza elementos que tenham um determinado atributo.

```
input[type=text] {
    background-color: yellow;
}
```

No HTML:

```
<input type="text" name="name" value="OpenSanca">
```

CSS - Seletor de classe

Estiliza um elemento com um determinado valor para o atributo class.

```
.description {
    font-size: 12px;
}
```

No HTML:

```
<p class=”description”>O tamanho da minha fonte é 12px</p>
<p>A minha não!</p>
```

CSS - Seletor de ID

Estiliza um elemento com um determinado valor para o atributo id.

```
#title {
    font-weight: bold;
}
```

No HTML:

```
<h1 id=”title”>Estou em negrito!</h1>
```

CSS - Seletor de pseudo-classe

Estiliza um elemento de acordo com com um estado especial em que se encontra.

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

No HTML:

Se passar o mouse em mim, fico vermelho!

CSS - Seletor de pseudo-elemento

Estiliza certas partes do documento associadas ao elemento.

```
p::after {
    content: "Pseudo-elemento!";
    color: red;
}
```

No HTML:

Vai aparecer um

Renderizado:

Vai aparecer um Pseudo-elemento!

CSS - Seletor filho

Estiliza o elemento filho de determinado elemento.

```
p > span {
    color: red;
}
```

No HTML:

```
<p>
    <span>Eu sou vermelho!</span>
    <i>Eu não!</i>
</p>
```

CSS - Seletor Adjacente (siblings)

Estiliza elementos adjacentes ou “irmãos” de determinado elemento.

```
p + span {
    color: red;
}
```

No HTML:

```
<p>Olá mundo!</p>
<span>Eu sou vermelho!</span>
<span>Eu não!</span>
```

```
p ~ span {
    color: red;
}
```

No HTML:

```
<p>Olá mundo!</p>
<span>Eu sou vermelho!</span>
<span>Eu também</span>
```

CSS - Seletor Descendente

Estiliza o elemento descendente ao elemento.

```
p span {
    color: red;
}
```

No HTML:

```
<p>Olá mundo!<br>
    <span>Eu sou vermelho!</span>
</p>
<span>Eu não!</span>
```

CSS - Especificidade

Ordem crescente de especificidade:

- Seletores Universais
- Seletores de tipo (tag)
- Seletores de classe
- Seletores de atributo
- Seletores de Pseudo-classes
- Seletores ID
- Estilo Inline

CSS - Exceções

A exceção !important

Quando a regra !important é utilizada na declaração do estilo substitui qualquer outra declaração feita no CSS, onde quer que esteja na lista de declaração.

```
#title{
    font-size: 12px;
}

h1{
  font-size: 18px !important;
}
```

No HTML:

<h1 id=”title”>Minha fonte tem tamanho 12... não não, espera, tem 18px</h1>

CSS - Exceções

A exceção :not

A pseudo-classe de negação :not não é considerada uma pseudo-classe no cálculo de especificidade. Contudo, seletores colocados na pseudo-class de negação são entendidos como seletores normais.

```
div.outer p {
   color: red;
}
div:not(.outer) p {
  color: blue;
}
```

No HTML

```
<div class="outer">
      <p>Eu sou vermelho</p>
      <div class="inner">
            <p>Eu sou azul!</p>
       </div>
</div>
```

CSS - Cálculo de especificidade

Para o cálculo de especificidade, cada tipo seletor tem uma pontuação. A pontuação é somada de acordo com os valores de cada seletor:

CSS inline: 1000 pontos;
ID: 100 pontos;
Classes, pseudo-classe e atributos: 10 pontos;
Elementos: 1 ponto.

Quanto maior o número, mais específico e mais força ele tem sobre outros seletores.

CSS - Cálculo de especificidade

```
header nav ul li a {
    color:blue;
  }
```

Como tem 5 elementos descritos, a pontuação é 0005.

```
.itemlink {
  color:red;
}
```

Cada classe equivale a um ponto na segunda casa: 0010.

CSS - Herança

No CSS, algumas propriedades podem ser herdadas de elementos ancestrais.

```
p {
    color: blue;
}
```

No HTML

```
<p>Tudo aqui é azul, <span>até eu!</span></p>
```

CSS - Herança

Nem todas as propriedades serão herdadas por elementos filho. Geralmente as propriedades que se referem ao box-model (height, width, margin, padding) não aceitam herança.

Propriedades não herdadas:
background, border (exceto: border-collapse e border-spacing), clip, content, counter, clue, display, float, height, left, margin, outline, overflow, padding, page-break, pause, play-during, position, right, table-layout, text--decoration, top, unicode-bidi, vertical-align, width, z-index.

CSS - Propriedades

A propriedade define uma característica visual para o elemento HTML “selecionado” pelo seletor.

Podemos dividir as propriedades nos seguintes grupos:

- Fontes, Texto & Escrita
- Posição e visualização
- Interface de usuário
- Miscelânea
- Espaçamento e margens
- Tabelas
- Alinhamento
- Backgrounds e bordas
- Cores
- Listas e contadores
- Colunas e layouts
- Transição
- Animação
- Entre outros

CSS - Fontes, Texto & Escrita

```
p {
    font-size: 12px;
    font-family: Arial, Helvetica, sans-serif;
}

a:hover {
    text-decoration: underline;
}
```

CSS - Posição e visualização

```
.header {
    position: fixed;
    top: 0;
    left: 0;
    right: 0;
}

.error {
    display: none;
}
```

```
.image{
    float: left;
}
```

CSS - Interface de usuário

```
.btn {
    cursor: pointer;
}


.btn:before {
   content: "Texto ";
}
```

CSS - Miscelânea

```
.class {
   all: initial;
}
```

```
.class {
   rotation: 90deg;
}
```

CSS - Espaçamento e margens

```
h1 {
    margin-top: 20px;
}


.container {
    padding: 20px;
}
```

CSS - Tabelas

```
.class {
  border-collapse: separate;
  border-spacing: 10pt 5pt;
}

.class {
   table-layout: fixed;
}
```

CSS - Alinhamento

```
p {
   line-height: 10px;
   font-size: 10px;
}

.class {
   vertical-align: top;
}
```

CSS - Backgrounds e bordas

```
.home {
    background-image: url('images/nome.jpg');
    background-repeat: no-repeat;
}

input {
    border: 1px solid #333333;
}
```

CSS - Cores

```
h1 {
    color: white;
}

.box {
    -ms-filter:"progid:DXImageTransform.Microsoft.Alpha(Opacity=50)"; // IE8
    filter: alpha(opacity=50); // IE 5-7
    opacity: 0.5;
}
```

CSS - Lista e contadores

```
ul {
   list-style: none
}

h1:before {
   content: "Chapter " counter(chapter) ":"
   counter-increment: chapter;
   counter-reset: section;
}
```

CSS - Colunas e layouts

```
div {
    -webkit-column-count: 3; /* Chrome, Safari, Opera */
    -moz-column-count: 3; /* Firefox */
    column-count: 3;
}
```

CSS - Transição

```
div {
    width: 100px;
    height: 100px;
    background: red;
    -webkit-transition: width 2s; /* For Safari 3.1 to 6.0 */
    transition: width 2s;
}

div:hover {
    width: 200px;
}
```

CSS - Animação

```
p {
  animation-duration: 3s;
  animation-name: slidein;
}
```

```
@keyframes slidein {
    from {
        margin-left: 100%;
        width: 300%
    }
    to {
        margin-left: 0%;
        width: 100%;
    }
}
```

Pré-processadores

É uma outra forma de escrever CSS.

Basicamento, você escreve em uma sintaxe definida (Less/Sass/Stylus) e o pré-processador compila para CSS.

O uso dessa forma possibilita algumas vantagens como uso de variáveis, operadores e funções.

Pré-processadores

Vantagens

- Produtividade
- Organização
- Componentização
- Uso de variáveis possibilita grandes mudanças com pouco trabalho
- Aninhamento de seletores facilita o entendimento do código
- Funções/mixins possibilitam reaproveitamento

Pré-processadores

Desvantagens

- Dependência de outras ferramentas para compilar
- Se usado de forma errada, pode gerar código duplicado
- Aninhamento de seletores em muitos níveis pode diminuir e legibilidade do código

Pré-processadores - Variáveis

```
$primaryBrandColor: #1abc9c;

h1 {
    color: $primaryBrandColor;
}

button {
    background-color: $primaryBrandColor;
}
```

Pré-processadores - Mixins

```
@mixin border-radius($radius) {
    -webkit-border-radius: $radius;
    -moz-border-radius: $radius;
    -ms-border-radius: $radius;
    border-radius: $radius;
}

.box {
    @include border-radius(10px);
 }
```

Pré-processadores - Aninhamento

SASS

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

    li {
        display: inline-block;
    }

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

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;
}
```

Pré-processadores - Modularização

```
// _reset.scss

* {
    margin: 0;
    padding: 0;
}
```

```
// base.scss

@import 'reset';

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

Pré-processadores - Herança (Extend)

```
.message {
  border: 1px solid #ccc;
  padding: 10px;
  color: #333;
}
```

```
.success {
  @extend .message;
  border-color: green;
}

.error {
  @extend .message;
  border-color: red;
}
```

Pré-processadores - Operadores

Torna possível a realização de operações matemáticas:

SASS

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

article[role="main"] {
  float: left;
  width: 600px / 960px * 100%;
}

aside[role="complementary"] {
  float: right;
  width: 300px / 960px * 100%;
}
```

CSS

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

article[role="main"] {
  float: left;
  width: 62.5%;
}

aside[role="complementary"] {
  float: right;
  width: 31.25%;
}
```

Boas práticas

Não sobrescreva suas próprias regras CSS.

```
/* desktop */
.botao-magico { float: right; }

/* mobile */
@media (max-width: 600px) {
  .botao-magico { float: none; }
}
```

Boas práticas

Use classes pra reaproveitar código CSS e evite IDs.

```
<div class="box menu">...</div>
<div class="box noticias"></div>
```

```
.box {
   background: #ccc;
   border-radius: 5px;
   box-shadow: 2px 2px 2px black;
}
```

Boas práticas

Não brigue com a especificidade do CSS.

Você já aprendeu a calcular a especificidade nos slides passados, use classes para seus seletores e evite IDs, pois como eles tem maior especificidade, fica dífícil ser utilizado de forma genérica como é possível com as classes.

Evite ao máximo ou nunca use !important, se seu código tiver muito disso, talvez seja melhor repensá-lo.

Boas práticas

Evite números mágicos

```
p{
  margin-top: 37px;
}
```

```
header {
  top: -3px;
}
```

Boas práticas

Use unidades flexíveis.

```
.coluna {
  width: 20%;
}
```

```
.description {
    font-size: 1.25em;
}
```

Fórmula para cálculo do EM.

target ÷ context = result

target = 20px; // Seu objetivo em px
context = 16px; // Valor padrão do elemento pai

result = 1.25 em

Boas práticas

Evite estilizar elementos usando o seletor de tipo

```
<header class="topo">
  <h1 class="chamada-principal">Compre já!</h1>
</header>
```

```
// Faça!
.topo {
    font-size: 1.25em;
}
```

```
// Evite!
header {
    font-size: 1.25em;
}
```

Boas práticas

Dê bons nomes pra suas classes

Os nomes das classes são a ponte entre seu HTML e seu CSS. Eles serão escritos no meio do conteúdo HTML da página e, portanto, devem ter semântica de conteúdo e não visual.

Use: “.painel”

Em vez de: “.painel-direita” ou “.painel-esquerda”

Boas práticas

Indentação e divisão lógica

Além de indentar seu código, tente organizar as propriedades em agrupamentos .

```
.seletor {
    [fonte e propriedades de texto]
    [plano de fundo]
    [tamanho]
    [bordas]
    [espaçamentos]
    [posicionamento]
}
```

Boas práticas

Utilize ou aproveite algumas práticas de padrões de arquite-tura CSS

- Object Oriented CSS
- SMACSS
- BEM
- DRY CSS

Boas práticas - Object Oriented CSS

Segundo o OOCSS, um objeto de CSS é todo padrão visual que pode se repetir no projeto e é identificado através de uma classe.

A ideia é separar as características visuais das características estruturais, tornando-os modulares para reutilização em diferentes elementos tendo resultados iguais.

```
.btn-style {
    border-radius:10px;
    background-color: blue;
}
```

```
<a class=”btn-style”></a>

<button class=”btn-style”></button>
```

Boas práticas - SMACSS

Scalable and Modular Architecture for CSS - Divisão em 5 categorias

base: seletores de elementos sem o uso de classes ou IDs

```
body {
    line-height: 1;
}
```

layout: estilos das estruturas que não se repetem na página como header, footer, sidebar, main e todo resto que não é componente ou estrutura reutilizável.

```
.container-main {
  width: 980px;
  margin: 0 auto;
  position: relative;
}
```

Boas práticas - SMACSS

module: responsável por determinar as partes reutilizáveis, os componentes.

```
.button {
  padding: 10px;
  background: #CFCFCF;
  font-size: 14px;
}
```

state: determina o estado do elemento.
.is-active
.is-selected

theme: parte que menos tem aplicação na maioria dos projetos, é usada para separar a parte que dá um novo design para a página.

Boas práticas - BEM (block, element, modifier)

É uma metodologia com várias versões cujo o preceito de esclarecer o desenvolvedor mais sobre o markup através de suas classes.

block - header, footer ou componente (graph, tabs).
element - descendente de um block com certa função.
modifier - propriedade de um block ou element que altera sua aparência.

block e element usam “__” como separador

```
<div class="report-graph">
<div class="report-graph__bar">...</div>
<div class="report-graph__bar report-graph__bar_size_big">...</div>
```

Boas práticas - DRY CSS (Don’t Repeat Your CSS)

O princípio consiste em não repetir propriedades com mesmos valores em seu código. A técnica, assim como outros sistemas, sugere que seu código seja pensando em termos de padrões de aparência.

Segundo o sistema, sempre que a aparência de um elemento mudar, sua classe precisará ser movida para outros agrupamentos nas folhas de estilo.

```
#LIGHT-WHITE-BACKGROUND,
.translation,
.entry .wp-caption,
#full-article .entry img
{
  background-color: #fff;
  border-color: #ccc;
}
```

Frameworks

Bootstrap

Foundation

Material Design Lite

SASS

LESS

Ferramentas

Plugin photoshop - http://css3ps.com/
Gerador de css - http://www.css3generator.com/
Gerador de botão - https://css-tricks.com/examples/ButtonMaker/

HANDS ON!

Referências

http://tableless.com.br/uma-breve-historia-do-css/
https://developer.mozilla.org/pt-BR/docs/Web/CSS
https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Getting_Started/Selectors
http://www.maujor.com/tutorial/guia-completo-seletores-css3.php
https://developer.mozilla.org/en-US/docs/Web/CSS/Reference
http://tableless.com.br/afinal-como-usar-heranca-no-css/
http://www.blooberry.com/indexdot/css/propindex/all.htm
https://css-tricks.com
http://sass-lang.com/
http://blog.caelum.com.br/seu-codigo-css-pode-ser-mais-limpo-flexivel-e-reaproveitavel/
http://usablica.github.io/front-end-frameworks/compare.html
http://css3ps.com/
http://tableless.com.br/oocss-smacss-bem-dry-css-afinal-como-escrever-css/
http://www.devmedia.com.br/cinco-ferramentas-css3-online-para-agilizar-seu-trabalho/27875

Obrigado!

Até semana que vem!