**SENAI – SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL**

|  |
| --- |
| **BRUNO PIRES**  **DAVI AMBROZIO**  **MARIA EDUARDA** |

**SELETORES, CASCATA E HERANÇA**

|  |
| --- |
| **CURITIBA**  **2017** |

**RESUMO:**

No nível mais alto, o efeito cascata é que controla toda a precedência em CSS e funciona . A ideia por trás da herança é relativamente fácil de se compreender. Elementos herdam estilos dos seus contêineres pai. Se você definir que a tag body use a cor vermelha, então todo os textos dentro dos elementos de body também serão vermelhos, caso não haja alguma regra específica.   
CSS é baseado em seletores de montagem e declarar as características que cada um desses elementos será exibido no navegador. Você pode usar html <p> <body> <html> <etc> ou incluir classes que são nomeados com pontos.

Palavras Chaves:Cascata, Herança, Seletores, Html, Css.

**LISTA DE FIGURAS**

[**Figura 1 – Exemplo Herança 10**](#_Toc483053228)

[**Figura 2 – Exemplo de Atributos 11**](#_Toc483053229)

**SUMÁRIO**

[**1.** **INTRODUÇÃO:** 4](#_Toc483066719)

[**2.** **SELETORES** 5](#_Toc483066720)

[**2.1.** **Seletores - Simples e combinados** 6](#_Toc483066721)

[**2.2.** **O seletor universal** 7](#_Toc483066722)

[**2.3.** **Seletores - tipo** 7](#_Toc483066723)

[**2.4.** **Seletor - classe** 7](#_Toc483066724)

[**2.5.** **Seletor - ID** 8](#_Toc483066725)

[**3.** **HERANÇA** 9](#_Toc483066726)

[**3.1.** **ATRIBUTOS** 10](#_Toc483066727)

[**3.2.** **HERANÇA DENTRO DO HTML** 11](#_Toc483066728)

[**3.3.** **HERANÇA DENTRO DO CSS** 12](#_Toc483066729)

[**3.4.** **VANTAGENS DE SE UTILIZAR HERANÇA** 13](#_Toc483066730)

[**3.5.** **DESVANTAGENS DE SE UTILIZAR A HERANÇA** 14](#_Toc483066731)

[**4.** **EFEITO CASCATA** 15](#_Toc483066732)

[**5.** **Conclusão** 19](#_Toc483066733)

[**REFERENCIAS** 20](#_Toc483066734)

# **INTRODUÇÃO:**

Neste trabalho vera sobre 3 estilos de css.

Os seletores são modelos que casam com elementos de uma arvore do documento e portanto podem ser usados para selecionar outros documentos no HTML.

A herança, que como os filhos herdam algo do pai, no css os elementos herdam algo de seu anteriores.

E no efeito cascata, é um estabelecimento de uma prioridade para aplicação da regra de estilo a um elemento.

# **SELETORES**

São as tags que colocamos no html e as mudamos no css podemos deixá-las da forma que acharmos melhor e mais bonitas com o css podemos fazer algumas alterações como por exemplo mudar o tamanho, a fonte, cor, posição se ela vai ter hiper link ou não e etc. E as demonstramos de jeitos diferentes para cada uma como mostra a tabela abaixo:

|  |  |  |
| --- | --- | --- |
| **Seletor** | **Formato** | **Descrição** |
| Universal | \* | Casa com qualquer elemento. |
| Tipo | E | Casa com qualquer elemento E. |
| Classe | .info | Casa com qualquer elemento cujo atributo **class** tem o valor **info**. |
| ID | #footer | Casa com qualquer elemento cuja **id** seja **footer**. |
| Descendente | E F | Casa com qualquer elemento F que seja descendente do elemento E. |
| Filho | E > F | Casa com qualquer elemento F que seja filho do elemento E. |
| Adjacente | E + F | Casa com qualquer elemento F que se segue imediatamente a E. |
| Atributo | E[att] | Casa com qualquer elemento E que tenha um atributo **att** independentemente do valor deste atributo. |
| Atributo | E[att=val] | Casa com qualquer elemento E que tenha um atributo **att** de valor igual a **val**. |
| Attribute | E[att~=val] | Casa com qualquer elemento E que tenha um atributo **att** de valor igual a um valor qualquer separado por um espaço de um valor igual a **val**. |
| Atributo | E[att|=val] | Casa com qualquer elemento E que tenha um atributo **att** de valor igual a um valor qualquer separado por um hífen de um valor começando com **val**. |
| Pseudo classe :first-child | E:first-child | Casa com o elemento E que for o primeiro filho do seu elemento pai. |
| Pseudo classe link | E:link E:visited | Casa com link ainda não vistado (:link) ou com link já visitado (:visited) links. |
| Pseudo classes dinâmicas | E:active E:hover E:focus | Casa com E quando de uma ação do usuário. |
| Pseudo classe :language | E:lang(c) | Casa com elementos E escritos no idioma c. |
| Pseudo-elemento :first-line | E:first-line | Casa com o conteúdo da primeira linha do elemento E. |
| O pseudo-elemento  :first-letter | E:first-letter | Casa com a primeira letra do elemento E. |
| Pseudo-elementos  :before e :after | E:before E:after | Usado para gerar conteúdo antes ou depois do conteúdo do elemento E. |

# **Seletores - Simples e combinados**

Existem duas categorias básicas de seletores: os simples e os combinados.

Um seletor simples consiste em um tipo qualquer de seletor ou o seletor universal seguido por nenhum ou algum seletor de atributo, seletor tipo ID, seletor de classe ou pseudo-classe. A seguir uma regra contendo um exemplo de seletor simples:

p.info { background:#ff0; }

Um seletor combinado (algumas vezes cham

ado de seletor contextual) consiste de dois ou mais seletores simples separados por um elemento de combinação. A seguir um exemplo de seletor combinado.

div p { font-weight:bold; }

A regra acima aplica-se a todo elemento p que seja descendente do elemento div.

Um pseudo-elemento pode ser colocado como apêndice a um seletor. Em seletores combinados, o pseudo-elemento somente poderá ser adicionado como apêndice ao último seletor simples.

Nas Partes 2 e 3 deste artigo serão detalhados com mais profundidade os seletores combinados, os elementos de combinação e os pseudo-elementos.

# **O seletor universal**

O seletor universal é representado por um asterisco, “\*”, e casa com todos os elementos do documento. É raro ver-se empregado em uma folha de estilos, mas o seletor universal é muito usado com seletores tipo ID e seletores de classe. Se o seletor universal não for o único componente de um seletor simples, o “\*” não deve ser usado :

.myclass é equivalente a \*.myclass

#myid é equivalente a \*#myid

Um uso bastante popular para o seletor universal é o uso para zerar margens e paddings de todos os elementos do documento:

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

Este procedimento é também conhecido como [Global White Space Reset](http://leftjustified.net/journal/2004/10/19/global-ws-reset/). (NT: link para matéria em ingles)

# **Seletores - tipo**

Um seletor tipo, casa com qualquer instância de um determinado tipo de elemento. A regra a seguir casa com qualquer elemento (do tipo) parágrafo no documento e configura seu tamanho de fonte para 2em:

p { font-size:2em; }

# **Seletor - classe**

O seletor de classe é representado por um ponto, “.”, e tem como alvo elementos com um determinado valor para seu atributo class. A regra a seguir aplica-se a todo elemento parágrafo cuja classe tenha o nome “info”:

p.info { background:#ff0; }

Você pode atribuir vários nomes para a classe de um elemento – o atributo class pode conter uma lista de vários nomes separados por espaço em branco. Assim, os seletores de classe podem ser usados para casar com elementos cuja classe contenha vários nomes. A regra a seguir casa com elementos p que tenham os nomes “info” e “error” declarados em seu atributo class:

p.info.error { color:#900; font-weight:bold; }

Nota: Seletores com múltiplos nomes de classes não são suportados pelas versões atuais do Internet Explorer, mas serão pelo IE7.

O tipo de elemento não precisa necessariamente ser declarado. Este procedimento, não declarar o tipo de elemento, equivale a usar o seletor universal como tipo de elemento. A regra a seguir casa com qualquer elemento da classe “info”, independentemente do tipo de elemento:

.info { background:#ff0; }

# **Seletor - ID**

O seletor ID é representado por um sinal de "tralha" (ou "jogo da velha"), “#”, e tem como alvo elementos com um deteminado valor de atributo ID. A regra a seguir aplica-se a todos os elementos cujo nome de ID seja “info”, independentemente do tipo de elemento:

#info { background:#ff0; }

Se você especificar um determinado tipo de elemento a regra será aplicada somente àquele tipo de elemento que tenha o nome da ID especificado:

p#info { background:#ff0; }

É importante lembrar que seletores ID tem uma especificidade maior que seletores de classe e que um valor de ID deve ser único em um mesmo documento. Assim um determinado seletor ID será aplicável a um único elemento no documento.

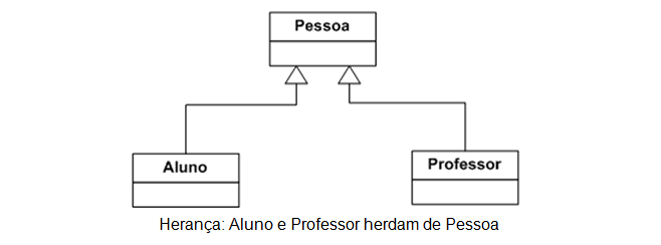
# **HERANÇA**

A principal característica da herança é dela conseguir que suas classes “compartilhem” seu atributos, métodos e outros membros de classe desta mesma, para ser possível a relação entre as classes é adotado um sistema hierarquicamente.

Suas duas classes principais são a classe base e classe derivada, a característica da primeira é que ela sede características a uma outra determinada classe, já a classe derivada herda estas características da classe base ou seja a classe base concede características a classe derivada.

**EX:**

Diga-se que a classe base tenha cedido as seguintes características ela é uma “classe pessoa” com os seguintes campos nome e idade. Então a classe derivada pega estas características e coloca o campo cargo, ou seja, ela herdou os campos nome e idade e acrescentou o campo cargo.

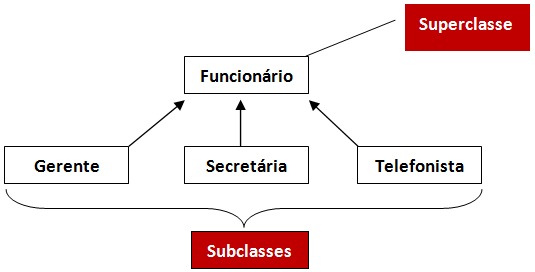


**Figura 1 – Exemplo Herança**

**Fonte:** http://www.usandoaccess.com.br/tutoriais/classe-no-access-orientacao-a-objetos.asp?id=1

# **ATRIBUTOS**

Os atributos são características dentro de uma determinada classe, então tamanho, espaçamento entre outras coisas, trazendo para um exemplo mais básico imagina-se que a classe é um animal e os atributos são suas características, então dentro deste animal existem vários “atributos” como altura, peso entre outros.



**Figura 2 – Exemplo de Atributos**

**Fonte:** <http://www.devmedia.com.br/abstracao-encapsulamento-e-heranca-pilares-da-poo-em-java/26366>

Na figura a cima pode ser explicado bem como se funcionam os atributos, a “superclasse” seria a classe que no caso é funcionário e os atributos seria a “subclasse”, entretanto funcionário é a classe principal e gerente, secretaria e telefonista são as características desta classe.

# **HERANÇA DENTRO DO HTML**

Para se entender como é o funcionamento da herança dentro de um código HTML é necessário saber sobre a árvore do documento, mas o que seria está “árvore do documento” ela é que faz os elementos de marcação se relacionarem entre os mesmo, como se fossem um relacionamento de uma família.

**EX:**

<! DOCTYPE html >

<html lang="pt-br">

<head>

<title>Tutorial CSS - Herança CSS</title>

</head>

<body>

<ul>

<li>Item 1</li>

<li>Item 2</li>

<li>Item 3</li>

</ul>

</body>

</html>

Neste caso funciona da seguinte maneira, é como se o “HTML” fosse pai dos elementos “Head”, “Body”.

“Head” é o pai de “Title” e “Body” pai de “UL”.

“UL” é filho de “Body” e pai dos três “LI”

Assim é o funcionamento de uma árvore de documentos, como se fosse uma árvore geológica, numa família o filho herda as características do seu pai ou avó e nos códigos também é assim o CSS vai herdar características dos elementos HTML.

# **HERANÇA DENTRO DO CSS**

Diga-se que temos um parágrafo e uma ênfase da seguinte forma:

<p>Texto do parágrafo com um <em> elemento EM </em> nele contido. </p>

Se quisermos dar uma cor para este parágrafo utilizaremos o seguinte comando:

P {

Color: Red;

}

Então o parágrafo ficara da seguinte maneira:

Texto do parágrafo com um elemento EM nele contido.

Ele ficara todo vermelho pois o programador deu uma cor pra esta mesma.

Agora se você quiser deixar uma cor para o parágrafo e uma para a ênfase o código será o seguinte:

P {

Color: Red;

}

P em {

Color: Blue;

}

Então o parágrafo ficara da seguinte maneira:

Texto do parágrafo com um elemento EM nele contido.

No primeiro caso não definimos uma cor específica para EM porém este elemento herdou a propriedade do “pai” que neste caso é o parágrafo, para conseguirmos colocar uma determinada cor somente no elemento EM teve que se anular o efeito para EM e especificar um somente para ele que no caso foi:

P em {

Color: Blue;

Porém a cor não precisa ser especificamente herdada do seu elemento “pai” a cor pode ser herdada de outro elemento.

**EX:**

<div id=”Cor”>

<p>Texto do parágrafo com um <em> elemento EM </em> nele contido. </p>

</div>

O elemento “EM” também ficara com a cor vermelha pois no CSS foi definido da seguinte maneira:

#Cor {

Color: Red;

}

O parágrafo( P) herdou a cor da “Div” e o “EM” herdou do “P”, ou seja, a Div seria o pai do P e o EM o filho do P, então o EM herdou a cor do seu avó que nesse caso seria a Div, entretendo assim mostrando que o elemento não necessariamente precisa herdar um comando de um elemento “pai”.

# **VANTAGENS DE SE UTILIZAR HERANÇA**

A principal vantagem de se utilizar a herança é que é muito mais prático, imaginasse que será utilizado um estilo para uma página inteira digamos que a cor será vermelha, com a herança você cria somente uma vez o estilo sendo assim não será necessário ficar refazendo os códigos.

EX:

Se você quiser colocar uma cor vermelha para sua página inteira, você usara o seguinte código.

Body {

Color: Red;

}

Sendo assim a página inteira ira herdar este estilo.

# **DESVANTAGENS DE SE UTILIZAR A HERANÇA**

A herança é algo muito prático como já dito, porém ela tem uma desvantagem não são todos os elementos que pegam a herança muitos a herança não funciona são estes:

* 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.

# **EFEITO CASCATA**

O que é:

É um efeito usado para controlar o visual de um documento HTML, não raro resulta em algumas regras CSS distintas, aplicáveis a um mesmo elemento no documento, ou seja, estabelece um conflito entre regras.

Serão explicados três maneiras de pôr estilo em uma tag:

1. Utilizando o atributo style diretamente na tag.
2. Utilizando a tag style, no cabeçalho.
3. Utilizando um arquivo CSS.

A primeira regra da hierarquia apresentada. É quem será escolhido está na ordem 1 a 3.

A prioridade é do atributo maior, no caso style na tag, depois a tag <style> no cabeçalho do HTML, e por fim o arquivo CSS.

Na cascata é como se o HTML chamasse primeiro a tag do atributo style. Caso não tenha, ele irá chamar a tag <style> no cabeçalho. Caso não tenha novamente ele irá chamar o CSS.

Exemplo do Efeito Cascata.

Ao se inserir uma tag <style> no cabeçalho do site, dizendo que a tag H1 terá a cor Verde.

Claramente, o resultado será a cor verde, afinal este é o único estilo definido:

<!DOCTYPE html>

<html>

<head>

<title> Efeito cascata em CSS </title>

<style type="text/css">

h1 { color: green; }

</style>

</head>

<body>

<h1> Que cor é essa ?</h1>

</body>

</html>

Agora alterando apenas um pouco, vamos acrescentar um atributo style dentro desta tag H1, dizendo que ela deve ser vermelha:

<h1 style="color: red;"> Que cor é essa ?</h1>

Agora temos um pequeno “confronto“ de ordens, pois a tag diz que a cor é verde, mas o atributo diz que deve ser vermelho.

Porem segundo a nossa hierarquia, o atributo tem mais poder que a tag, portanto a cor que prevalecera é a vermelha.

Agora iremos utilizar os três casos.

Iremos colocar um atributo style, então esta tag irá obedecer o que estará na <style>, e será a azul.

Colocaremos abaixo outra tag H1, mas com o atributo style dizendo que ela é vermelha.

Usando a hierarquia o atributo é maior que a tag, então este texto será vermelho.

estilo.css: h1 { color: yellow; }

<DOCTYPE html>

<head>

<title> Efeito cascata em CSS </title>

<link rel="stylesheet" type="text/css" href="Cascata 2.css"/>

<style type="text/css">

h1 { color: green; }

</style>

</head>

<body>

<h1 style="color: red;"> Que cor é essa ?</h1>

</body>

</html>

Agora vamos retira a declaração do atributo:

<!DOCTYPE html>

<html>

<head>

<title> Efeito cascata</title>

<link rel="stylesheet" type="text/css" href="estilo.css"/>

<style type="text/css">

h1 { color: green; }

</style>

</head>

<body>

<h1> Que cor é essa ?</h1>

</body>

</html>

Deixando apenas a tag <style> e o arquivo **estilo.css.**

Segundo a nossa hierarquia, ela dirá que nosso texto ficara verde.

E por último eliminaremos a tag <style>.

Assim deixando nosso código assim:

<!DOCTYPE html>

<html>

<head>

<title> Efeito cascata em CSS </title>

<link rel="stylesheet" type="text/css" href="estilo.css"/>

</head>

<body>

<h1> Que cor é essa ?</h1>

</body>

</html>

Assim sobrando apenas o estilo.css. E deixando o nossa tag com a cor amarela.

# **Conclusão**

Essa herança é responsável por uma parte da cascata. É aí que está uma das primeiras vantagens de se utilizar o CSS no desenvolvimento com padrões. A vantagem da herança é exatamente podermos modificar poucas linhas do CSS para fazer alterações no site inteiro, sem ter que caçar elemento por elemento e modificando suas propriedades. Os seletores e a herança trabalham juntos e este estilo final de um elemento pode ser citado em vários locais diferentes, que interagem entre si de uma forma complexa. Está forma de interação torna o CSS poderoso, mas pode torna-lo confuso e difícil de depurar.

# **REFERENCIAS**

Guia de referencia CSS(Pagina 8 e 9):

<http://www.w3c.br/divulgacao/pdf/guia-css-w3cbr.pdf>

<http://guilhermemuller.com.br/pt/elearning/html_css_basico/licao/2/sintaxe-seletores-css>

<http://guilhermemuller.com.br/pt/elearning/html_css_basico/licao/1/introducao-css>

<https://github.com/fswceulp/livro-desenvolvimento-software-web/tree/master/css>

<http://www.linhadecodigo.com.br/artigo/2146/jquery-sharp8212-seletores-e-estilizacao.aspx>

<http://www.maujor.com/tutorial/heranca-css.php>