PADRÕES WEB (NORMAS) DA W3C PARA USO DO XHTML/CSS

1. Introdução a padrões Web (Web Standards)

Tim Berners-Lee's sonha para sua invenção, a World Wide Web, um espaço de uso comunitário onde compartilha-se informações de trabalho, lazer e socialização (The World Wide Web, A very short personal history). Como desenvolvedores Web criando sites corporativos, sociais e educacionais, nós transformamos este sonho em realidade.

Mas, nesta época de crescimento avassalador, a Web necessita de orientação para desenvolver seu pleno potencial. As Web standards são estas orientações. As standards asseguram que todos tenham acesso às informações e também tornam o processo de desenvolvimento Web mais rápido e mais agradável.

Conformidade com as standards facilitam o acesso a Web aos usuários portadores de necessidades especiais. Pessoas cegas podem se valer de leitores de páginas Web. Pessoas com reduzido grau de visão podem rearranjar as páginas web e aumentar o tamanho para facilitar a leitura. E, pessoas usando dispositivos portáteis podem navegar tão facilmente quanto aquelas em grandes workstations.

Como explicaremos mais adiante existem muitas razões práticas para que os desenvolvedores Web estejam engajados com as standards. Por exemplo: motores de busca encontrarão maior facilidade na indexação do site. O uso de código específico para determinado browser frequentemente duplica ou triplica o trabalho de criação das páginas Web e ainda não contempla futuros novos tipos de mídia. Esta situação só assume este aspecto caótico, pelo não uso das standards.

Alguns temem as standards por acreditar que elas impõem limites. Na verdade, elas eliminam muito do trabalho tedioso de desenvolvimento Web, proporcionando ao desenvolvedor mais tempo e maior flexibilidade para livre criação. As standards compatibilizam documentos Web tanto para aplicações futuras quando para tecnologias passadas.

Muitos usos da Web, inclusive aqueles que hoje em dia não passam de sonhos, não serão possíveis ou serão bem mais difíceis sem uma total compatibilidade com as standards. Atualmente os sistemas e software amplamente usados, são universalmente aceitos, mas, quem sabe o que está por vir no futuro? Amarrar-nos à tecnologias proprietárias significa

entregar nosso futuro ao sabor do sucesso ou insucesso de um fornecedor. Confiar em standards de abrangência universal, proporcionará a sobrevivência da Web, motivando a continuação de inovações ao ritmo atual.

As standards têm muito mais a oferecer,e, nós do The Web Standards Project (WaSP), consideramos necessário ajudar você a aprender mais sobre elas. Este documento é meramente um ponto de partida; ele explica concretamente o que são as standards, para que elas servem e qual sua importância. Cada documento que publicamos na Web é uma informação a mais para uso comum neste universo que é a Web. Podemos fazer isto pensando no crescimento futuro, ou ao contrário, sem visão futurística adicionando peso e carga aos documentos o que certamente causará uma rutura futura. A escolha é nossa; as conseqüências serão para todos.

2. O que são as standards?

2.1 W3C Standards - O que é o W3C?

O World Wide Web Consortium (W3C) é um consórcio de âmbito internacional dedicado a "conduzir a Web ao seu pleno potencial". É dirigida por Tim Berners-Lee, o inventor da Web. Fundado em 1994, o W3C tem mais de 400 organizações membro incluindo aí a Microsoft, America Online (proprietária da Netscape Communications), Apple Computer, Adobe, Macromedia, Sun Microsystems, e uma vasta gama de fabricantes de hardware e software, fornecedores de conteúdos, instituições acadêmicas e companhias de telecomunicações. O Consórcio está hospedado em três instituições de pesquisas - - MIT nos EUA, INRIA na Europa e Keio University no Japão .

2.2 O que faz o W3C?

O W3C desenvolve especificações abertas (de facto standards) para aumentar a interoperabilidade dos produtos para Web. As Recomendações do W3C são desenvolvidas por grupos de trabalho formados por menbros do Consórcio e experts convidados. Os grupos de trabalho criam esboços e propostas de recomendações , baseados em um consenso comum de companhias e organizações interessadas na criação de aplicações Web. Estas são então submetidas à apreciação dos membros do W3C e seu diretor para aprovação formal como uma Recomendação. Maiores detalhes sobre este processo e seus vários estágios, podem ser obtidos no website doW3C.

2.3 O que são as standards do W3C?

HTML 4.0 - Linguagem de marcação para Hypertexto

A linguagem de marcação para hypertexto (HTML) é universalmente usada para estruturar textos nos documentos Web. Os browsers interpretam estes documentos e apresentam sua estrutura conforme a mídia do usuário. Por exemplo: browsers visuais apresentam o elemento **strong** (**** ... ****) em texto negrito, ao passo que um leitor de tela imprimirá ênfase ao texto quando o ler.

Com o uso das Folhas de estilos em cascata (CSS) os autores podem definir como os elementos estruturais serão apresentados, sobrescrevendo as apresentações padrão (defaut) dos browsers.

XML 1.0 - Linguagem extensível de marcação

Linguagem extensível de marcação (XML) é uma linguagem parecida com o HTML, mas em lugar de se utilizar de um conjundo fixo e definido de elementos, permite que você defina seus próprios elementos - ou use um conjunto definido por outra pessoa. Permite até o uso de vários conjuntos de elementos em um mesmo documento - através do uso do XML namespaces.

Algumas aplicações XML, tais como XHTML e MathML, já se tornaram Recomendações do W3C Outras estão em fase de esboço no W3C.

As standards para folhas de estilo, tais como CSS e XSL, comtemplam uma grande variedade de opções para especificar como os documentos XML devem ser renderizados. Os browsers ainda oferecem suporte esporádico à renderização direta de documentos XML, por isto o uso de HTML (ou XHTML) com as CSS para estilização, é a solução a ser adotada. Atualmente 0 XML é mais usando para comunicação entre aplicações.

XML é mais flexível do que HTML, sobretudo por permitir a criação de elementos e sistemas de estruturação próprios. Isto torna o XML um formato ideal para a organização de grandes quantidades de dados, sendo, hoje, usado em muitos bancos de dados e motores de busca.

Trecho de um documento XML

```
<addressbook>
  <entry>
    <name>Bill Gates</name>

<email>bgates@microsoft.com</email>
  </entry>
  <entry>
    <name>Marc Andreesen</name>

<email>marca@netscape.com</email>

  </entry>
  <entry>
    <name>Jon S. von
Tetzchner</name>

    <email>jon@opera.com</email>
  </entry>

</addressbook>
```

XHTML 1.0, 1.1, Modularização XHTML

XHTML

XHTML 1.0 é uma reformulação do HTML como aplicação XML. XHTML 1.0 pode ser vista como uma linguagem ideal nascida do HTML 4.01, e tecnicamente bem mais precisa devido a influência do XML.

XHTML será apresentado no seu browser de maneira idêntica ao seu HTML equivalente. Você pode querer optar pelo uso do XHTML quando houver interesse em servir seu conteúdo de diferentes maneiras, por exemplo, para um PDA; A sintaxe precisa das regras do XML torna muito mais fácil e menos dispendioso o processamento do XHTML quando comparado com o do HTML comum.

Ideologicamente, XHTML 1.0 herda os seguintes conceitos do HTML 4.01:

- A apresentação dos documentos deve ser separada da formatação via folhas de estilos
- Os documentos devem ser acessíveis
- Os documentos devem ter caráter internacional

XHTML 1.0 também usa o modelo das três DTDs:

Strict, Transitional, e Frameset. Este modelo surgiu com o HTML 4.0 e continuou com o HTML 4.01.

Algumas práticas do XML aplicáveis ao XHTML incluem:

- Todos os tipos de documentos são declarados por um correta declaração DOCTYPE
- A estrutura do documento é constituida por: uma declaração DOCTYPE, o elemento html com o respectivo namespace XHTML declarado, o elemento head incluindo o elemento title e o elemento body
- Todos os nomes de elementos e atributos são escritos em letras minúsculas e o nome de atributos escritos entre aspas.
- Elementos não vazios (p ex: p, li) devem ter tags de fechamento
- Elementos vazios (p.ex: br, hr, img) devem ter suas tags terminadas por uma barra (<br />)
- Documentos devem validar contra as DTD neles declaradas

XHTML 1.1 está constituída em três partes básicas:

- DTD Strict da XHTML 1.0 (com pequenas modificações)
- Modularização XHTML
- Anotação Ruby

A redação de documentos em XHTML 1.1, pode ser feita de duas maneiras. A primeira delas com o uso da DTD pública para XHTML 1.1, que não permitindo qualquer atributo de apresentação, resulta em documentos extremamente estruturados. A separação entre a

estrutura e a apresentação é total, sendo toda a apresentação servida através de folha de estilo.

A outra maneira de redigir documentos em XHTML 1.1 é com o uso da modularização XHTML. A modularização consiste na compartimentação dos conhecidos componentes do HTML e do XHTML (tais como os textos, as tabelas, os frames, os formulários) em módulos distintos. Você pode escrever sua própria DTD e usar somente os componentes que necessite. É a característica extensível da linguagem, possibilitando ao autor Web a oportunidade de criar uma marcação "customizada".

A anotação Ruby é uma maneira especial de tratar as anotações para determinados caracteres asiáticos. Ruby faz parte do trabalho desenvolvido com vistas a internacionalização da Web.

CSS - Folhas de Estilos em Cascata

Folha de Estilos em Cascata (CSS) é um mecanismo para alterar a aparência dos elementos do HTML ou do XML , através da definição de estilos aplicáveis ao diferentes elementos, à classes de elementos ou à instâncias individuais.

Folhas de estilos podem ser usadas para definir a aparência de todo o site. Seguindo a introdução das CSS, o W3C recomenda que definições específicas para layout sejam retiradas do HTML e entregues às folhas de estilos, ensejando assim, a criação de uma World Wide Web mais simples e estrutural.

Document Object Model - Nível 1

A inserção de uma linguagem de scripts (tais como ECMAScript, a versão standard do JavaScript) em uma página Web e o máximo aproveitamento de todo o poder e interatividade do script, é possivel graças ao DOM. (Em termos de programação o Document Object Model (DOM) Nível 1 é um "Application Programming Interface" (API) para interação com páginas web.) Ele possibilita ao script, acesso fácil à estrutura, ao conteúdo e à apresentação de um documento escrito em linguagens tais como o HTML e as CSS.

O DOM é compatível com futuras melhorias tecnológicas; ele permitirá a todas as linguagens de script interagir com qualquer linguagem usada no documento. Esta standard não só tornará mais fácil programar HTML dinâmico, como também simplificará a tarefa de adaptações para futuras tecnologias da Internet.

O que é ECMA?

O European Computer Manufacturers Association (ECMA) - Associação Européia de Fabricantes de Computadores - é uma organização oficialmente fundada em 1961 com a finalidade de levantar necessidades para elaboração de standards para os formatos operacionais de computadores, incluindo linguagem de programação e códigos input/output.

A ECMA tem sua sede em Genebra, Suiça, próxima ao quartel general da International Organization for Standardization (ISO) e da International Electrotechnical Commission (IEC). Em 1994, com a finalidade de representar melhor a diversificação das suas atividades, o nome

da organização foi mudado para ECMA - European Association for Standardizing Information and Communication Systems.

O que faz a ECMA?

O principal objetivo da ECMA é a eleboração de Standards e produção de Relatórios Técnicos na área das tecnologias de comunicações e informação. ECMA não é um instituto oficial para normatização e sim uma associação de companhias que frequentemente colabora com instituições oficiais nacionais e internacionais.

As Standards da ECMA têm sido aceitas como base para standards européias e internacionais. Já foram publicadas mais de 270 ECMA Standards e 70 Relatórios Técnicos.

Das standards, 85 foram tornadas standards internacionais pela International Organization for Standardization (ISO) e 25, tornadas standards européias pelo European Telecommunications Standards Institute (ETSI).

O que são as standards ECMA?

ECMAScript (standard para JavaScript)

ECMAScrip é uma linguagem standard de script, baseada nas largamente enpregadas JavaScript da Netscape e JScript da Microsoft. A standard para ECMAScript é definida pelo ECMA's Technical Committee 39 (TC-39).

O principal uso de ECMAScript, que é uma linguagem baseada em objetos, é a manipulação dos objetos definidos pelo Document Object Model (DOM) em uma página Web. Estes objetos (na prática, os elementos que compõem uma página Web ou a própria página como um todo) podem ser adicionados, suprimidos, movidos ou ter suas propriedades alteradas. Isto possibilita ao desenvolvedor web implementar efeitos tais como animação de textos, rollovers de imagens e mudança de páginas baseada em inputs do usuário.

A atual especificação para ECMAScript é a ECMA Standard ECMA-262, *ECMAScript Language Specification, 2nd edition*.

3. Quais são as vantagens de se adotar as web standards?

Acessibilidade

Para software/máquinas

Páginas em conformidade com as standards terão grande visibilidade em resultados de buscas na Web. Documentos conformes, graças a sua estruturação, facilitam e fornecem informações detalhadas aos mecanismos de busca, resultando em uma indexação mais apurada.

Os softwares, quer instalados do lado do cliente quer do lado do servidor terão maior facilidade em entender a estrutura de documentos em conformidade com as standards e a

tarefa de instalar um mecanismo de busca dentro do próprio site torna-se mais fácil e propicia melhores resultados.

As standards são escritas de modo a que os browsers antigos entendam a estrutura básica dos documentos. Mesmo que eles, browsers antigos, sejam incapazes de entender as mais recentes e sofisticadas implementações das Standards, estarão aptos a renderizar os conteúdos do site. Certamente, o mesmo se aplica para sistemas robotizados que coletam informações do site para motores de busca ou outros indexadores.

Códigos em conformidade com as standards são passíveis de serem validados em serviços de validação. Os validadores processam os documentos e reportam uma lista de erros, tornando a tarefa de busca e correção de erros bem mais fácil e rápida.

Documentos conformes são mais facilmente convertidos para outros formatos, tais como bancos de dados ou documentos Word. Isto possibilita maior versatilidade no uso das informações contidas em um documento da World Wide Web, e simplifica a migração para novos sistemas - hardware bem como software - incluindo dispositivos como TVs e PDAs.

Para as pessoas

Acessibilidade é um conceito atrelado a variadas web standards, especialmente ao HTML.

Significa não somente acesso à web para pessoas portadoras de necessidades especiais, como também à usuários com browsers não convencionais, incluindo-se aí os browsers de voz, que lêem os documentos para pessoas com restrições visuais, os browsers Braille que convertem textos para a escrita Braille, os browsers portáteis com monitores minúsculos, as tela para tele-textos e demais dispositivos não usuais.

Com a crescente diversidade de meios para acesso a Web, os ajustes ou duplicações dos websites para atender aos novos dispositivos torna-se uma tarefa cada vez mais difícil (na verdade, pode-se dizer, uma tarefa impossível nos dias atuais). As standards são um grande passo para a solução deste problema. Sites em conformidade com as standards serão consistentemente renderizados não só em browsers convencionais novos e antigos como também em browsers não usuais e sofisticados tipos de mídia.

Algumas das consequências de se ignorar as standards são óbvias: a mais evidente é a restrição de acesso ao site. Para os seus negócios, qual é o sentido em limitar o acesso a uma fraçãodo total de pessoas interessadas em visitar seu site? Para um site comercial, negar acesso mesmo que seja para uma fração mínima de seu público alvo, pode fazer uma diferença muito grande para o faturamento. Para um site educacional, é evidente assegurar acesso não só aos estudantes com sofisticados browsers gráficos, como também àqueles menos favorecidos, usando browsers de textos ou estudantes portadores de necessidades especiais usando browsers especiais.

O mesmo princípio aplica-se a qualquer tipo de website— ainda que seja grande a tentação de se afastar das standards e tirar o máximo proveito de soluções proprietárias os ganhos em acessibilidade com as standards, são bem mais compensadores a longo prazo.

Estabilidade

A grande maioria das web standards são em geral projetadas com vistas a compatibilidade com o passado e com o futuro — assim documentos projetados em conformidade com versões antigas das standards continuarão válidos para novos browsers, e aqueles projetados em conformidade com as standards atuais "degradarão graciosamente" produzindo um resultado aceitável em browsers antigos.

Durante sua existência, um website pode vir a ser gerenciado por várias equipes de designers e é importante que qualquer um compreenda e edite o código com facilidade. As web standards baseiam-se em um conjunto de regras que qualquer desenvolvedor Web é capaz de seguir, entender e estar familiarizado. Quando um desenvolvedor projeta segundo as standards, outro desenvolvedor é capaz de assumir sem qualquer problema ou solução de continuidade.

4. Conclusão

Como desenvolvedor web, nós estamos constantemente tendo que administrar e solucionar inconsistências na maneira de renderização de páginas web por variados tipos e versões de browsers. Isto demanda um maior consumo de tempo na criação de múltiplas versões de código ou codificação para um só browser, o que torna o site de díficil, quando não impossível, acesso para uma parcela do público que o visita. Esta situação se agravará com o advento de novos hardware e software capacitandos a acessar a Web, tais como telefones, pagers, e PDAs.

As web standards não são leis cabalísticas, impostas por organizações enclausuradas em torres de marfim. Como já foi descrito, as standards são para pessoas que as criaram e optaram por adotá-las — fabricantes de browsers, desenvolvedores web, fornecedores de conteúdos, e outras organizações.

Projetar páginas web em conformidade com as standards reduz o tempo de desenvolvimento e de manutenção. Correções e soluções de problemas são mais fáceis, porque o código segue as standards. Não há preocupações com codificação e manutenção de várias versões do mesmo código. *Uma* só versão do site é suficiente.

A adoção universal das standards está se tornando de primordial importância. A missão do The Web Standards Project é a de fazer da Web um lugar melhor para desenvolvedores e usuários finais, incentivando fabricantes de browsers e editores de páginas web a aderir às standards em suas aplicações. Este esforço receberá uma grandiosa ajuda quando desenvolvedores *usarem* as standards e insistirem em que seus geradores de códigos e dispositivos de renderização estejam em conformidade com as standards.

Nosso desejo de que as razões que apresentamos propiciem a você, desenvolvedor web, todo o incentivo para aderir as standards, e também forneçam a argumentação necessária para que você motive no seu trabalho seus colegas desenvolvedores a também aderir às standards.

Referencia Bibliográficas dos Autores:

Web Standards Project Developer Education Committee: *Stephan Nedregaard (coordenação), Kynn Bartlett, Gail T. Cohen, Jens Edlund, Nick Finck, Tomas Fjetland, Peter Fleck, Markus Gut, Holger Maier, Julian Missig, Laura Mollett, Randy Piatt, Lewis A. Shadoff, Juergen Steinwender, Bart Szyszka, Matthew Thomas, Dane Weber*

Fonte: http://www.w3.org

Apostila XHTML/CSS

Nota do Autor:

Olá pessoal,

Nosso Mini-curso sobre XHTML/CSS visa levar os alunos a obter um conhecimento mais específico do desenvolvimento WEB, par a isso após essa longa discussão sobre Standards Web podemos iniciar o processo de produção com XHTML usando o CSS. O nosso Mini-curso seguirá os seguintes passos:

Lista de conteúdos do Mini-Curso e Divisão dos Assuntos Individuais dos ministrantes:

Grade Geral do Mini-Curso	Introdução ao Mini-Curso
1 -Introdução - Página Básica (Alessandro)	1 - Standard Web (Samuel Santos)
2 - Folha de Estilos (Aleksandro)	2 - Introdução ao CSS (Aleksandro)
3 - Estilizando texto (Samuel Santos)	3 - Revisão do HTML puro (Alessandro)
4 - As lista (Aleksandro)	
5 - Os links (Alessandro)	
6 - Box CSS (Samuel Santos)	
7 - Menus (Aleksandro)	
8 - DIV's (alessandro)	
9 - Layouts (Samuel Santos)	
10 – Hospedagem (Alessandro e Aleksandro)	

Cronograma Específico:
(Alessandro)

- 1- Introdução - Página Básica (Alessandro)
- Como criar um HTML atentando par a o padrão W3C, com DTD transitional, DOCTYPE, e XHTML.

- 2 - Revisão HTML puro (Alessandro)
- 1 – elementos , <h1, ..., h6 ></h1, ... , h6>
- 2 – elementos
- 3 – elementos
- 4 – elementos
- 5 – elementos
- 6 – elementos <dd></dd>
- 7 – elementos <img></img>
- 8 – elementos <bgcolor />
- 9 – elementos
- 10 – elementos <a >
- 11– "..." <...n></...n>

* E Demais elementos (veja outros)!

- 3 - Os links (Alessandro)
- estilos em Links;

**atenção para prioridades na declaração;

- 4 - DIV's (alessandro)

Cronograma Específico:
(Aleksandro)

- 2 - Introdução ao CSS (Aleksandro)
- O que é css?
- Surgimento?
- Porque usar css?
- Vantagens ?

- 2 - Folha de Estilos (Aleksandro)
- O que é folha de estilo?
- folhas externa ?
- estilo interno?
- estilos "inline"?

- 3 - As lista (Aleksandro)
- tipos de lista;
- estilos em lista(imagens, fundo);
- propriedade CSS para listas;

- 4 - Menus (Aleksandro)
- Tipos de Menus;
- técnicas de construção;
- Criar um menu CSS ajustável ao site que do mini-curso;

Cronograma Específico: (Samuel Santos)	
1 - Standard Web (Samuel Santos) Standards W3C (normas) O que é XHTML, DTD, DOCTYPE O que é XML, DOM , EMAC e Ruby. Vantagens e Acessibilidade 2 - Estilizando texto (Samuel Santos) Sintaxe CSS Tipos de CSS A propriedade CSS font A propriedade CSS text	A propriedade CSS margin A propriedade CCS border A propriedade CSS padding A propriedade CSS background A propriedade CSS list Pseudos-Elements Entrelinhas Medidas CSS 3 - Box CSS (Samuel Santos) criar boxes e posicionar 4 - Layouts (Samuel Santos) Construir o layout para o site.

* Estudem mais que o indicado, (Todo o conteúdo!)

** Procurem outros materiais mais específicos para seus tópicos;

**** Pratiquem bastante XHTML/CSS é programação (são apenas código), se não praticar não vai saber usar, a melhor forma de aprender é ir vendo o que cada propriedade faz (Façam vários teste de suas página XHTML)

***** Estudem pois, estamos na reta final (Tá VALENDO !).

*** Bons Estudos e dedicação Sempre!

Essa primeira parte é importantíssima pois, Abrange de forma Geral o Conteúdo

(Leitura Obrigatória)

• 1 - O que significa a sigla CSS?

CSS é a sigla para Cascading Style Sheets que em português foi traduzido para **folhas de estilo em cascata** e nada mais é, que um documento onde são definidas regras de fomatação ou de estilos, a serem aplicadas aos elementos estruturais de marcação.

• 2 - Qual é finalidade das CSS?

A finalidade das CSS é a de retirar do HTML toda e qualquer declaração que vise a formatação, a apresentação do documento. Isto significa dizer que tags do tipo `<font>`, ~~<bold>~~ `<b>`, ~~~~ <i>....etc, bem como uso de colunas e linhas de tabelas para obtenção de espaçamentos não são admitidos ou admitidos com restrições em um projeto Web com CSS. Vale dizer simplificando: **HTML para estruturação e CSS para apresentação.**

Atualização em 2005-08-26: As tags `<b>` e `<i>` cumprem finalidade unicamente de apresentação. Elas não têm qualquer efeito sobre tecnologias assistivas. Foram criadas à época antiga de marcar documentos web e basicamente calcados em estruturação para mídia tipográfica, não imputando qualquer relevância ao texto com elas marcado e nada significam para leitores de tela ou mecanismos de busca por exemplo. Não estão em desuso, são previstas no XHTML 1.1, podem e devem ser usadas se a sua intenção é a de *contemplar somente mídia visual*.

As tags `<strong>` e `<em>`, estas sim, são interpretadas por todas as tecnologias assistivas e devem ser usadas em substituição a `<b>` e `<i>` se a sua intenção é a de contemplar estas tecnologias.

Contudo, embora o suporte para elas deva continuar ainda por muito tempo, nada aponta para uma revitalização de `<b>` e `<i>`. Pelo contrário, as evidências a esta época, indicam que devam ser banidas a partir do XHTML 2.0.

• 3 - Porque devo usar CSS?

CSS permite que você retire da marcação HTML toda a formatação (apresentação) do documento WEB. Quem vai determinar cores, formas, tipos e tamanhos, posicionamento e em fim todo o "visual" da página são as CSS. As vantagens estão relatadas nas respostas a seguir.

• 4 - Quais as vantagens de se usar CSS ?

Inúmeras são as vantagens do uso das CSS nos documentos Web. Eis uma relação das principais:

- controle total sôbre a apresentação do site a partir de um arquivo central;
- agilização da manutenção e redesign do site;
- saida para diferentes tipos de mídia a partir de uma versão única de HTML;
- redução do tempo de carga dos documentos Web;
- adequação simplificada aos critérios de acessibilidade e usabilidade;
- elaboração de documentos consistentes com as aplicações de usuários futuras;
- aumento considerável na portabilidade dos documentos Web.

• 5 - As CSS estão de acordo com as Web Standards?

Sim estão. As CSS estão entre as práticas vivamente recomendadas pelo W3C para projetos Web. Os esforços dos órgãos normatizadores apontam no momento atual para elaboração de documentos Web acessíveis, usáveis e portáteis com grande ênfase. E, CSS facilita e simplifica a obtenção destas três variáveis.

• 6 - O que é regra CSS?

Regra CSS e uma unidade mínima de programação de estilos, que segue uma sintaxe própria e destina-se a estilizar uma ou mais propriedades CSS. (ver resposta seguinte)

• 7 - Como escrevo uma regra CSS?

A sintaxe de uma regra CSS compreende um **seletor** uma **propriedade** e um **valor** escritos como mostrado abaixo:

seletor {propriedade: valor;}

Nota: ao conjunto propriedade:valor denominamos declaração CSS

Por exemplo:

```
p {text-indent:10pt;} é uma regra CSS

p é o seletor.
{text-indent: 10pt} é a declaração CSS.
text-indent - é a propriedade CSS
10pt - é o valor CSS
```

• 8 - Quais são os caracteres permitidos em regras CSS?

São as letras de a-z, A-Z, os números de 0-9, underscore, hífen e caracteres de escape. Caracteres Unicode 161-255, bem como caracteres Unicode como códigos numéricos. Não é permitido iniciar-se um nome com um traço ou número.

• 9 - As sintaxe CSS é sensível a maiúsculas/minúsculas?

Não. Regras CSS não são case sensitivas, isto é, pode-se usar maiúsculas ou minúsculas nas folhas de estilo indeferentemente. Isto é válido sómente para as declarações específicas de CSS. Por exemplo: mesmo numa declaração CSS, figura.gif é diferente de FIGURA.gif

• 10 - Posso incluir comentários nas regras CSS?

Sim. Comentários podem e devem ser usados nas folhas de estilo. É recomendável que se faça amplo uso de comentários para fornecer informações sobre os seletores, propriedades e valores declarados, com o intuito de facilitar futuras modificações ou mesmo entendimento do código gerado. Os comentários devem estar entre as marcas **/***

e */ e podem ser inseridos em qualquer espaço em branco dentro da folha de estilos. Não são permitidos comentários aninhados.

```
/* Isto é um comentário CSS */
```

• 11 - O que é efeito cascata das CSS?

Efeito cascata é o método pelo qual é aplicada uma regra CSS em função do pêso (importância, prioridade) da regra CSS. Quando a um elemento HTML aplica-se mais de uma regra CSS, diz-se que há um conflito de regras e será aplicada aquela que tiver maior pêso (importância, prioridade) determinada pela ordem do efeito cascata.

• 12 - Como é determinada a ordem para o efeito cascata?

A ordem para o efeito cascata é a ordenação das regras CSS de acordo com critérios pré-estabelecidos, com a finalidade de resolver eventuais conflitos entre regras. E estes critérios são os abaixo enumerados:

1. Localizadas todas as regras CSS aplicáveis ao seletor/propriedade, determina-se a especificidade de cada uma delas. A regra mais especifica entre as conflitantes será aplicada.

   Se não for encontra uma (ou mais) regra aplicável ao elemento HTML, este herdará as propriedades de estilo de seu elemento pai. Não havendo elemento pai será aplicado o valor inicial default de estilo para aquele elemento

2. Regras com declaração importante (! important) tem prioridade sobre aquelas sem a declaração. Se o autor e o usuário declaram regras conflitantes com !important, as do usuário têm prioridade sobre as do autor.
3. Regras com mesmo peso (sem !important) e conflitantes, declaradas pelo autor têm prioridade sobre aquelas declaradas pelo usuário. Regras do usuário com maior especificidade que aquelas do autor têm a prioridade. Regras com igual especificidade declaradas pelo autor têm prioridade sobre as do usuário. Regras do autor e do usuário têm prioridade sobre as regras default das aplicações do usuário (p.ex.: o browser)
4. Regras mais específicas têm a prioridade sobre as menos específicas.
5. Entre regras de mesmo pêso, têm prioridade aquelas declaradas por último na sequência das regras na folha de estilos.

   Folhas de estilo inline, incorporadas, lincadas e importadas (@import) nesta ordem, têm do maior para o menor pêso (estilos inline têm o maior pêso). Entre várias folhas de estilo lincadas têm maior prioridade aquela importada por último (mais próxima da tag </head>. O mesmo ocorre entre as folhas importadas (@import)

•12 - O que é declaração CSS?

Declaração CSS e o fragmento de uma regra CSS dentro dos colchetes { }. A declaração CSS compõe-se de duas partes: a propriedade e o valor e uma regra CSS pode conter várias declarações separadas por um ponto-e-vírgula.

Por exemplo:

```
h1 {color: #000000; /* esta linha contém a declaração CSS */ }
```

• 13 - Posso atribuir mais de uma declaração a um seletor?

Sim, as declarações devem ser separadas por ponto-e-vírgula.

Por exemplo:

```
p {
background: #FFFFFF;
color: #000000;
}
```

• 14 - O que é declaração com uso de '!important' ?

Usa-se !important em uma declaração CSS para incrementar o pêso (ou prioridade) da declaração no efeito cascata das regras de estilo. Convém ressaltar que CSS2 mudou o conceito para declarações com !important do autor e do usuário. A recomendação atual determina que !important do usuário prevaleça sobre o autor.

```
Sintaxe de uso:
h3 {
font-size:120% !important;
color: #000000
}
Esta regra aumenta o peso para o
tamanho de letra do elemento h3
```

• 15 - O que é seletor?

Seletor é uma entidade que identifica um elemento HTML ou define uma classe ou pseudo classe na qual a regra de estilo será aplicada.

Por exemplo:

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

O seletor é **p** (elemento HTML parágrafo) e a regra CSS escrita determina que os parágrafos terão uma fonte de tamanho 12px.

```
p, ul {text-indent: 10pt;}
```

Os seletores são **p** e **ul**

• 16 - O que é agrupamento de seletores?

Agrupamento de seletores é uma forma compacta de reunir seletores que compartilham as mesmas regras de estilo.

Por exemplo:

```
p { color:#000000;}
.classeb {color: #000000;}
span {color: #000000;}
```


Os seletores **p .classeb** e **span** so
em letras na cor preta.
então podemos agrupá-los assim:

```
p, .classeb, span { color:#000000;}
```

Notar que seletores agrupados devem ser
separados por vírgula

•17- O que é seletor tipo classe?

Seletor tipo classe tem uma abrangência mais ampla. É um seletor cujo nome você "inventa" e pode ser aplicado a qualquer elemento HTML. A sintaxe para este tipo de seletor é um nome (nome da classe) precedido de um **.** (ponto) O nome pode conter letras de a-z, A-Z, números de 0-9, hífen, ou caracter de escape. Caracteres Unicode 161-255, bem como qualquer caracter Unicode de código númerico, contudo não podem começar com um (traço) ou um número.

• 18 - Como defino e escrevo uma classe ?

Classe é um seletor CSS aplicável a um ou mais elementos HTML ou a instâncias de um ou mais elementos HTML os quais serão estilizados segundo as regras CSS declaradas na classe.

Por exemplo:

```
.minhaclasse {color: #FF0000;}
O nome da classe você atribui ("inventa")
seguindo umas regrinhas que veremos adiante.
```

A classe denominada **minhaclasse** define a cor vermelha (#FF0000) para as letras do elemento onde ela for aplicada. Notar que um seletor do tipo classe tem em sua sintaxe um **.** (ponto) antes do nome da classe.

```
<p class="minhaclasse">
Aqui o texto do parágrafo..bla...bla...
</p>

<h3 class="minhaclasse">
Aqui o texto do cabeçalho...
</h3>
```

```
Acima tanto o parágrafo como o cabeçalho
serão na cor vermelha.
```

E aqui umas regrinhas para escolha do nome da classe e sua sintaxe:

Repito e enfatizo que você deve colocar um **.** (ponto) imediatamente antes do nome da classe. O nome da classe poderá ser composto por letras de a-z, ou A-Z, números de 0-9, underscore, hífen, caracter de escape, caracter Unicode (161-255), bem como qualquer caracter Unicode numérico.

Nomes de classe no podem começar com traço ou número.

É de boa norma (não obrigatório) escolher-se o nome da classe pela sua função e não pela sua aparência. Por exemplo: **.cor_rodape** é preferivel a **.rodape_verde** Se no futuro você resolve mudar a cor no rodapé de verde para vermelho vai ficar esquisito não é?

Segundo CSS1 apenas uma classe pode ser atribuída a um seletor. CSS2 permite que você atribua mais de uma classe a um seletor.

• 19 - O que são seletores tipo pseudo-elementos?

Pseudo-elementos são usados em CSS para adicionar efeitos especiais a um seletor ou a parte de um seletor. Eles não são elementos HTML.

```
p:fist-line { font-size:160%;}
h3:first-letter { color:#000000;}
pseudo-elementos so separados dos
seletores por : dois-pontos como mostrado
acima
```

Os pseudo-elementos em CSS1 são os dois mostrados no exemplo acima:

:first-line que estiliza a primeira linha de uma frase.

:first-letter que estiliza a primeira letra de uma palavra.

Em CSS2 temos ainda os pseudo-elementos **:before** e **: after**

Pseudos elementos podem ser combinados com classes. Observe abaixo:

```
p .generica:first-letter {
font-style: italic;
}
Nesta regra CSS a primeira letra dos
parágrafos que pertencem a classe generica,
será em itálico.
```

• 20 - O que são seletores tipo pseudo classes?

Pseudo classes são usadas em CSS para adicionar efeitos especiais a um seletor. Elas não são elementos HTML. As pseudo classes em CSS1 são definidas apenas para o elemento HTML **a** (<a>) conforme abaixo:

:link que estiliza links não visitados.

:visited que estiliza links visitados.

:hover que estiliza links com o ponteiro do mouse em cima.

:active que estiliza links ativos (clicados).

Nota: Em CSS2 temos ainda as pseudo classes **:first-child** e **:lang**

```
a:link {
background: #000000;
color: #FFFFFF;
}
a:visited {
background: #FFFF00;
color: #000000;
}
a:hover {
background: #FFFFCC;
color: #003366;
}
a:active {
background: #000000;
color: #00FF00;
}
```

Pseudos classes podem ser combinadas com classes. Observe abaixo:

```
a.topo:link {
background: #000000;
color: #FFFFFF;
}
a.topo:hover {
background: #FFFFFF;
color: #000000;
}
```

• 21- O que é seletor tipo ID?

Seletor tipo ID é um identificador individual associado a UM E SOMENTE UM elemento HTML no documento. Não se pode ter mais de um elemento HTML com a mesma ID, ao contrário das classes, que podem ser aplicadas a vários elementos HTML em um mesmo documento.

A sintaxe para o seletor ID é o caracter # , seguido de um nome que você "inventa". Este nome pode conter letras de a-z, A-Z, dígitos 0-9, espaço, hífen, ou caracter de escape. Caracteres Unicode 161-255, bem como qualquer caracter Unicode de código númerico, contudo não podem começar com um (traço) ou um número.

Por exemplo:

```
#menu {
color: #000000;
background:#FFFFFF;
}
<div id="menu">
Esta é uma div estilizada pela regra CSS
menu
</div>
```


• 22- O que é seletor contextual?

Seletor contextual é aquele que se refere a uma ocorrência especifica de um elemento HTML. A sintaxe para definir um seletor contextual é uma string formada por seletores individuais separados por um espaço em branco. A regra é aplicável ao último elemento da string (que satisfaz a sequência da string). Os exemplos abaixo esclarecem os seletores contextuais.

Por exemplo:

```
p span {color:#FF0000;} - aplica-se
                a span dentro de um parágrafo.
Não vale para span dentro de um h2 por
exemplo.

ul li a {font-size:12px;} - aplica-se a
um link dentro de um
item de lista de uma lista não ordenada.
```

• 23 - O que é seletor de atributo?

Seletor de atributo é aquele que se refere a um atributo do elemento HTML, podendo ser definido de três maneiras distintas:

1-) referência ao atributo do elemento;

```
img[hspace] {
border: 2px solid #FF0000;
}
Coloca bordas de 2px de cor vermelha nas
imagens que tenham o atributo "hspace"
```

2-) referência ao atributo e seu valor;

```
img[hspace="20px"] {
border: 2px solid #FF0000;
}
Coloca bordas de 2px de cor vermelha nas
imagens que tenham o atributo "hspace"
igual a 20px
```

3-) referência ao atributo e a um de seus valores.

```
p[title bla="nononono"] {
border: 2px solid #FF0000;
}
Coloca bordas de 2px de cor vermelha nos
parágrafos que tenham a palavra bla
no atributo "title=nononono".
```

Nota: Este tipo de seletor no é suportado por alguns browsers atuais! Faz parte das especificações CSS2.

• 24 - O que é seletor "filho"?

Seletor "filho" é aquele contido dentro de outro elemento HTML (chamado de seletor "pai").

```
#esquerda p {
font-size:14px;
}
Esta regra CSS estabelece tamanho de letra
de 14px para palavras nos parágrafos que
estão dentro da #esquerda. Parágrafos "filhos"
 de #esquerda!
```

índice

• 25 - O que é propriedade CSS?

Propriedade é a característica a ser estilizada pela regra CSS. Por exemplo: cor do fundo (background-color), tamanho das letras (font-size), etc...A cada propriedade deve corresponder um ou mais valores e ao conjunto propriedade mais valor(es) chamamos, declaração CSS

• 26 - O que é propriedade CSS em declaração única?

Algumas das propriedades CSS (por exemplo: background, font, padding, border,...) permitem que se declare mais de um valor na mesma declaração CSS. Regras que se valem desta permissão são ditas em declaração única.

Por exemplo:

```
p {
font-weight: bold;
font-style: italic;
font-variant: small-caps;
font-size: 16px;
font-family: sans-serif;
}
```

A regra acima que estiliza a propriedade font de um parágrafo pode ser escrita em declaração única conforme mostrado abaixo:

```
p {font: bold italic small-caps 16px
sans-serif;}
```

Se estiver especificada a propriedade line-height, esta poderá também ser inserida na declaração única logo após o tamanho de fonte, como mostrado abaixo para uma line-height de 20px

```
p {
font:bold italic small-caps 16px/20px
sans-serif;
}
```


• 27 - O que é valor CSS?

Valor é o aspecto a ser assumido pela propriedade CSS declarada. A propriedade determina **o que** vai ser estilizado e o valor determina **o quanto** ou **como** vai ser estilizado.

Por exemplo:

```
h1 {background: #0000FF;}
Nesta regra CSS é declarado o valor
#0000FF (azul)para o fundo do elemento h1.
```

• 28 - O que é valor default de uma propriedade CSS?

Valor default é o valor de cada uma das propriedades CSS atribuido por default pela aplicação de usuário (p.ex: o browser do usuário). Todas as propriedades CSS têm um valor inicial (default). Se nenhuma regra CSS estabelece o valor de uma propriedade este valor será tomado como o valor default da aplicação do usuário.

•29 - O que é herança?

Documentos HTML são hierarquicamente estruturados. Há um ancestral, o elemento de nível mais alto (pai) do qual os demais elementos (filhos) são descendentes. Os elementos filhos herdam as características do elemento pai. Por exemplo: cores e tamanhos de letras.

Assim numa folha de estilos as regras de estilo CSS para o elemento pai serão herdadas pelos elementos filhos. **Nem todas as propriedades são herdadas.** A herança é passada do elemento pai para os filhos e estes se tiverem filhos tambem herdarão e assim por diante, seguindo a herança até o elemento de menor nível.

Os estilos herdados podem ser sobrescritos por declarações específicas para os elementos filho.

•30 - Onde devo salvar minhas CSS ?

As CSS podem ser salvas em três lugares diferentes no seu site.

- dentro da própria tag HTML a estilizar - Diz-se que a aplicação do estilo é **inline**.
- dentro da tag <style> colocada na tag <head> do documento HTML a estilizar - Diz-se que a aplicação do estilo é **incorporada**.
- em um arquivo independente salvo com a extensão .css - Diz-se que a aplicação do estilo é **por folha de estilos externa**.

•31 - O que é estilo inline?

Diz-se que o estilo é inline quando as regras de estilo são declaradas dentro da própria tag HTML, com o uso do atributo **style**. As regras de estilo são aplicadas exclusivamente naquele elemento HTML. Este tipo de declaração retira toda a flexibilidade das CSS uma vez que você terá que agir diretamente na tag HTML se desejar fazer uma mudança de estilos, seu uso **deve ser evitado** ou no máximo restrito a um ou poucos documentos.

Por exemplo:

```
<p style="text-indent: 10px;">
Parágrafo com recuo de 10px</p>
```

•32 - O que é folha de estilo incorporada?

Folha de estilo incorporada é uma folha de estilo anexada a um só documento HTML. As regras de estilo são válidas tão somente para aquele documento específico. A folha de estilo é especificada dentro do elemento <head> do documento e declarada como conteúdo da tag <style>.

Por exemplo:

```
<head>
<style type="text/css">
<!--
p {
text-indent: 12px;
-->
</style>
</head>
```

Nota: As regras de estilos em folhas incorporadas, são escritas como se fossem um comentário HTML, (entre as marcações <!-- e --> com a finalidade ocultar as informações de estilo para aqueles navegadores que não dão suporte para folhas de estilo. A não colocação da marcação de comentário fará com os navegadores antigos exibam as regras de estilo na tela (ou outra mídia de exibição usada) como se fora um texto.

•33 - O que é folha de estilo externa?

Folha de estilo externa é um arquivo que pode ser gerado em qualquer editor de texto, e salvo com extensão **.css**, contendo regras de estilo e que pode ser lincado a um ou vários documentos HTML. Esta é sem dúvida a maneira mais eficaz para se formatar todo um site, bem como mudar sua aparência parcial ou globalmente, pois a simples edição de um só arquivo fará o efeito.

O arquivo é lincado ao documento HTML pela declaração link, colocada dentro do elemento <head>. Os arquivos que contém as regras de estilo ou seja a folha de estilos devem ter a extensão **.css** . Por exemplo: **meu_estilo.css**

Por exemplo:

```
<head>
<link rel="stylesheet" href="meu_estilo.css"
type="text/css">
</head>
```

índice

• 34 - Posso usar estilos inline, incorporados e externos em mesmo documento?

Sim, a existência dos três tipos de folhas de estilos em um mesmo documento não sofre qualquer restrição, porém você deve estar ciente do "efeito cascata" das folhas de estilos, para que os efeitos que você pretende com suas CSS estejam declarados na folha certa.

índice

• 35 - O que é folha de estilo importada?

É uma folha de estilos para ser importada por um ou mais documentos HTML com a vantagem de poder ser combinada com outras folhas de estilo. Isto permite que se crie uma folha de estilos contendo regras gerais para todos os documentos e outras tantas folhas mais específicas contendo regras válidas para alguns documentos apenas, ou mesmo para alguns elementos específicos os quais requerem declarações de estilo adicionais.

A sintaxe para importar folhas de estilos é através da declaração **@import** dentro do elemento <style>. A declaração @import deve ser colocada **antes** de qualquer outra declaração contida dentro do elemento <style>. Se mais de uma folha de estilo for importada elas respeitarão o "efeito cascata", isto é: a prioridade_1 é para a declaração escrita em último lugar no elemento <style>, a prioridade_2, para a penúltima e assim sucessivamente.

Por exemplo:

```
type="text/css">
<style type="text=css">
```

```
<!-- @import url(../folhaparcial_1.css);
     @import url(../folhaparcial_2.css);
<link rel="stylesheet"
href="regras_gerais.css"
-->
</style>
```

• 36 - Como fazer layouts alternativos para meu site ?

Para possibilitar aos seus visitantes a escolha de uma layout ou mesmo cores diferentes para navegar no seu site, você deverá primeiramente criar as CSS para os diferentes layouts.

Em seguida disponibilizar na página botões ou links para acionar os layouts. O clique nestes botões ou links acionará um código javascript que chamará uma das CSS criadas. Ver respostas a seguir.

Nota:O código Java Script que faz funcionar as CSS alternativas foi desenvolvido por um inglês chamado Paul Sowden e é mundialmente utilizado por desenvolvedores e projetistas CSS. Você encontra o código e uma explicação completa das folhas alternativas neste TUTORIAL.

• 37 - O que é folha de estilo alternativa?

Uma folha de estilo alternativa é aquela projetada para aplicar um estilo alternativo, a ser usado no lugar do estilo principal declarado como persistente e/ou preferido e que está aplicado ao documento.

Estas folhas de estilo permitem ao usuário escolher um estilo alternativo para ser aplicado ao documento - uma maneira bastante conveniente de se especificar estilos diferenciados para diferentes tipos de mídia.

Nota: Cada grupo de folhas alternativas deve ser especificado com um atributo "title" único. Por exemplo: a sintaxe para link a uma folha de estilo alternativa é conforme abaixo:

```
<link rel="alternate stylesheet"
href="meu_estilo3.css"
type="text/css" title="alternativo3"
media=screen>
<link rel="alternate stylesheet"
href="meu_estilo4.css"
type="text/css" title="aternativo4"
media=print>
```

• 38 - O que é folha de estilo persistente?

Uma folha de estilo diz-se persistente quando é uma folha de estilo default do documento. Ela pode ser desabilitada em favor da folha de estilo alternativa. A sintaxe para link a uma folha de estilo persistente é conforme abaixo:

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

índice

• 39 - O que é folha de estilo preferida?

Uma folha de estilo diz-se preferida quando é uma folha de estilo default do documento e é identificada como tal quando declarado o atributo "title" no elemento link. Ela pode ser desabilitada em favor da folha de estilo alternativa. Poderá existir uma só folha de estilo deste tipo no documento. A sintaxe para link a uma folha de estilo preferida é conforme abaixo:


```
<link rel="stylesheet"
href="meu_estilo2.css"
type="text/css" title="preferido">
```

• 40 - Como combinar múltiplas folhas de estilo em uma só?

Para aplicar várias folhas de estilo (cada uma contendo um conjunto diferente de regras de estilo) em um documento basta dar o mesmo nome ao valor do atributo "title" do elemento link. As folhas de estilo assim agrupadas serão aplicadas ao documento como folhas do tipo preferida. A sintaxe para link a um conjunto de folhas de estilo assim combinadas é conforme abaixo:

```
<link rel="stylesheet"
href="principal.css"
title="combinada">
<link rel="stylesheet"
href="estilo_letras.css"
title="combinada">
<link rel="stylesheet"
href="estilo_tabelas.css"
title="combinada">
```

• 41 - O que são elementos inline?

São os elementos HTML que não causam uma quebra de linha no fluxo da apresentação. Podem estar contidos em elementos de bloco ou mesmo em outros elementos inline. Não podem conter elementos de bloco.

São elementos inline do HTML 4.0:

em, strong, dfn, code, samp, kbd, var, cite, abbr, acronym, tt, i, b, big, small, sub, sup, br, script, map,

São elementos inline do HTML 4.0 Transitional:
em, strong, dfn, code, samp, kbd, var, cite, abbr, acronym, tt, i, b, u, s, strike, big, small, sub, sup, a, img, object, applet, font, basefont, br, script, map, q, span, bdo, iframe, input, select, textarea, label, button, (ins, del).

•42 - O que so elementos nível de bloco?

São os elementos HTML que causam uma quebra de linha no fluxo da apresentação. Podem estar contidos em outros elementos de bloco. Não podem estar contidos em elementos inline. Podem conter tanto elementos inline como elementos de bloco.

São elementos nível de bloco do HTML 4.0:
p, h1, h2, h3, h4, h5, h6,ul, ol, pre, dl, div, noscript, blockquote, form, hr, table, fieldset, address, (ins, del).

São elementos de bloco do HTML 4.0 Transitional:
p, h1, h2, h3, h4, h5, h6, ul, ol, dir, menu, pre, dl, div, center, noscript, noframes, blockquote, form, isindex, hr, table, fieldset, address, (ins, del).

• 42 - O que so elementos pai e filho?

Elemento pai é aquele que contém outro elemento (este chamado de elemento filho).

```
<p>Aqui o texto de <span>um</span>
parágrafo!
No exemplo acima:
<p> é o elemento pai e <span> o elemento
 filho.
```

Caso não seja especificado o elemento filho herda as propriedades do elemento pai. Ver herança.

• 43 - Posso usar folhas de estilo e declarações de estilo no HTML em um mesmo documento?

Sim. As folhas de estilo e os estilos declarados no HTML, serão ignorados naqueles navegadores que NÃO dão suporte para CSS. As declarações de estilo no HTML devem ser evitadas ou reduzidas ao máximo.

• 44 - Quem tem precedência: atributos HTML ou propriedades CSS?

Propriedades CSS têm precedência sôbre atributos HTML. Se ambos (CSS e atributos HTML) estão especificados, os atributos HTML serão exibidos naqueles navegadores que NÃO dão suporte para CSS, no entanto não terão efeito algum em navegadores mais modernos com suporte para CSS.

• 45 - Pode CSS ser aplicada a um documento não HTML?

Sim, CSS pode ser aplicado a qualquer documento de formato estruturado como por exemplo um documento XML.

• 46 - Como posso centrar um elemento nível de bloco?

Declare sua largura e ajuste suas margens esquerda e direita para um valor **auto** (automático).

```
#tudo {
width:760px;
margin-left:auto;
margin-right:auto;
}
```

A regra CSS acima centra na tela, qualquer que seja a resolução do monitor, um bloco com 760px de largura.

Nota: O Internet Explorer não reconhece esta regra. Para contornar isto usamos um artifício (hack) que consiste em declarar mais uma regra só para o IE como mostrado abaixo:

```
#tudo {
width:760px;
margin-left:auto;
margin-right:auto;
text-align:center; /* Para o IE */
}
```

• 47 - Devo usar " " (aspas) ao declarar URL?

O uso de aspas duplas ou simples ao declarar URL nas folhas de estilo é facultativo.

No exemplo abaixo todas as três formas de declaração estão corretas.

```
#tudo {
background-image:url(imagem.gif)
}
#tudo {
background-image:url('imagem.gif')
}
#tudo {
background-image:url("imagem.gif")
}
```

• 48 - Como retirar o sublinhado dos links?

Retira-se o sublinhado do link com a propriedade **text-decoration** ajustada para none

```
a:link {
text-decoration:none;
}
a:visited {
text-decoration:none;
}
a:hover {
text-decoration:none;
}
a:active {
text-decoration:none;
}
```

```
ou ainda

a:link, a:visited, a:hover, a:active {
text-decoration:none;
}
```

Mas, cuidado ao retirar o sublinhado dos links. Assegure-se de que esta prática não irá tornar seus links pouco identificáveis na página. Considere neste caso declarar um fundo colorido para o link.


```
a:link {
text-decoration:none;
background-color:#FFFFCC;
}

esta regra retira o sublinhado e coloca
um fundo colorido no link.
```

• 49 - Como estilizar links de maneira diferente na mesma página?

Criando-se classes para cada uma das maneiras que se queira estilizar

```
a.maneira1:link {
text-decoration:none;
background-color:#FFFF00;
}
a.maneira2:visited {
text-decoration:underline;
background-color:#0000FF;
}
a.maneira3:hover {
text-decoration:overline;
background-color:#FF0000;
}
e no HTML
<a class="maneira1">Meu link_1</a>
<a class="maneira2">Meu link_2</a>
<a class="maneira3">Meu link_3</a>
```

• 50 - Como declarar fontes cujo nome é composto por mais de uma palavra?

Nomes de fontes cujo nome é composto por mais de uma palavra devem ser declarados entre ' ' (aspas simples) ou " " (aspas duplas)

```
p {
font-family: 'Times New Roman', Times,
Serif;
}
ou
p {
font-family: "Times New Roman", Times,
Serif;
}
```

- 51 - **Qual a maneira mais rápida de aprender CSS?**

Se você souber ou vier a descobrir no futuro, por favor informe para eu colocar aqui! **:-)**

Obrigado por ter vindo conhecer este FAQ, espero que ele seja útil para agilizar seu aprendizado de CSS e lhe sirva de uma referência para futuras consultas.

TODO aprendizado requer ESTUDO e praticar, praticar, praticar, praticar e praticar outra vez!

Fonte: Maurício Samy Silva

http://www.maujor.com/maujor.com/tutorial/faq.html

Segunda Parte do Material para o Mini-Curso - **Introdução as CSS**

O nascimento do HTML

A linguagem de marcação HTML (Hyper Text Markup Language) foi desenvolvida e aperfeiçoada até tornar-se tal como a conhecemos nos dias atuais a partir de uma "invenção" devida a um pesquisador físico, para tráfego de seus textos e informações de natureza científica.

Assim, o embrião do HTML surgiu para servir a uma comunidade bastante restrita, a comunidade de cientistas. Com a introdução gradativa de novas tags, atributos e aplicações específicas, o HTML tornou-se padrão mundial de apresentação de conteúdo na Web.

O HTML atual

E, já com várias inovações o HTML era usado para construção de páginas Web, que no início limitavam-se a exibir informações contidas nos documentos.

A evolução vinha atropelando tudo com uma avalanche de novos aplicativos, facilidades, softwares, hardware etc. E o HTML não passou ao largo, pelo contrário, a simples linguagem de marcação destinada a apresentar conteúdos carecia de uma maior flexibilidade no sentido de manipular visualmente os conteúdos.

Novas tags e atributos foram inventados, tais como a tag `font` e o atributo `color` que permitiam alterar a aparência de textos. Assim nasceu a **estilização** dos conteúdos.

E a evolução trazendo novas descobertas, corre célere neste dinamismo alucinante que estamos testemunhando até os dias de hoje. Novas tags e novos atributos de estilo foram introduzidos no HTML. Com isso, a velha linguagem de marcação passou a exercer uma dupla função. A função de estruturar o conteúdo através da marcação e a função de apresentá-lo ou seja de dar a aparência final.

Os problemas criados

Mas, esta dupla função do HTML, se por um lado resolveu uma necessidade dos designers e projetistas por outro acabou trazendo sérios problemas aos projetos criados. Os documentos Web publicados na Internet, cada vez mais sofisticados e extensos, estavam fugindo do controle de seus criadores.

Para ilustrar suponha o seguinte exemplo:

Seu melhor cliente telefona às 17:00h da tarde de uma sexta-feira (sempre ligam nesta hora para solicitar alguma coisa não é mesmo?) e diz o seguinte;

teremos uma reunião aqui na empresa, na segunda-feira às 0800h com um potencial comprador e é nossa intenção fazer uma apresentação dos nossos produtos através do site que você criou e mantém. Seguindo uma sugestão do nosso departamento de marketing precisamos mudar a cor de todos os títulos no site de verde para vermelho, pois que esta é a cor principal da marca do nosso comprador e com isso pretendemos fixar uma cumplicidade subliminar. Isto é bem simples de fazer, não é? Afinal é só mudar a cor! Dá para você 'botar no ar' até às 19:30h ? Quero dar uma olhadinha antes de encerrar o expediente. OK?

Claro que você concorda e responde que vai providenciar rapidinho, afinal *é só para mudar a cor*. Mas, são 180 páginas no site! E os títulos são tags de cabeçalho deste tipo:

```
<h1><font color="#00FF00">Título</font></h1>
```

Supondo uma média de 3 títulos por página, você tem um total de 540 tags `font` para editar e mudar o atributo `color`. E se o seu cliente tivesse pedido para mudar a cor dos textos, e do fundo? Bem, este exemplo simples dá uma dimensão de um dos problemas criados com a mistura de marcação com apresentação - estilização!

A solução proposta

Cada vez mais ficava evidente que esta mistura que maravilhou os projetistas Web no início, tornara-se uma grande dor de cabeça. E é claro, a solução passava por dissociar linguagem de marcação da estilização.

Desta necessidade, eu diria mesmo uma imposição, nasceu as CSS, sigla em inglês para *Cascading Style Sheet* que em português foi traduzido para Folha de Estilo em Cascata.

A introdução deste conceito preconiza o uso dos elementos (tags) HTML, exclusivamente para marcar e estruturar o conteúdo do documento. Nenhum elemento HTML será usado para alterar a apresentação, ou seja estilizar o conteúdo. A tarefa de estilização ficará a cargo das CSS que nada mais é do que um arquivo independente do arquivo HTML no qual são declaradas propriedades e valores de estilização para os elementos do HTML.

Estas declarações de estilo, quer sejam estruturadas em um arquivo externo com extensão `.css` quer sejam declaradas de outros modos (importadas, lincadas, incorporadas ou inline), contém todas as regras de estilo para os elementos do documento HTML.

Voltando àquela situação criada no item anterior, agora você mudaria a cor de TODOS os cabeçalhos `h1` em TODO o site em CINCO SEGUNDOS. Às 19:20h você retorna a ligação do cliente e pede para a secretária avisá-lo de que "já está no ar", sem maiores traumas, correrias e estresses. Ah e mais, mesmo que o site tivesse 1.800 páginas e não as 180 da situação criada, você gastaria os mesmos cinco segundos.

As restrições

A idéia, a filosofia mesmo, de projeto Web aponta para uso amplo das CSS, ainda não explorada em toda sua potencialidade por razões de incompatibilidades de certas propriedades CSS com navegadores mais antigos e com as interpretações diferentes das CSS por parte das aplicações de usuários criadas por fabricantes distintos.

Contudo, há uma tendência - e torcemos para que se concretize rapidamente - de que as novas tecnologias voltadas para o desenvolvimento, não só das variadas aplicações de usuário como também de softwares e hardwares, atendam e se enquadrem dentro das recomendações e especificações dos órgãos normatizadores, notadamente as standards do W3C.

Quando o projeto Web em todas as suas incontáveis variantes, seguir a normatização e padronização recomendada pelo W3C, teremos uma Web muito mais fácil, dinâmica e agradável.

O efeito cascata

Que estilo será aplicado, quando há conflito de estilos especificados (por exemplo: uma regra de estilo determina que os parágrafos serão na cor preta e outra que serão na cor azul) para um mesmo elemento HTML?

Aqui entra o **efeito cascata,** que nada mais é, do que o estabelecimento de uma *prioridade* para aplicação da regra de estilo ao elemento. Para determinar a prioridade são considerados diversos fatores, entre eles, o tipo de folha de estilo, o local físico da folha de estilo no seu todo, o local físico da regra de estilo na folha de estilo e a especificidade da regra de estilo.

A prioridade para o efeito cascata em ordem crescente:

1. folha de estilo padrão do navegador do usuário;
2. folha de estilo do usuário;
3. folha de estilo do desenvolvedor;
    - estilo externo (importado ou linkado).
    - estilo incorporado (definido na seção `head` do documento);
    - estilo inline (dentro de um elemento HTML);
4. declarações do desenvolvedor com !important;
5. declarações do usuário com !important;

Assim, uma declaração de estilo com !important definido pelo usuário prevalece sobre todas as demais, é a de mais alta prioridade. Entre as folhas de estilo definidas pelo desenvolvedor do site, os estilos inline (dentro de um elemento HTML) tem a prioridade mais elevada, isto é, prevalecerá sobre a folha de estilo definida na seção `head`, e, esta prevalecerá sobre uma folha de estilo externa. A prioridade mais baixa é para estilos padrão do navegador.

A regra CSS e sua sintaxe

Uma regra CSS é uma declaração que segue uma sintaxe própria e que define como será aplicado estilo a um ou mais elementos HTML . Um conjunto de regras CSS formam uma Folha de Estilos. Uma regra CSS, na sua forma mais elementar, compõe-se de três partes: um seletor, uma propriedade e um valor e tem a sintaxe conforme mostrado abaixo:

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

Seletor: genericamente, é o elemento HTML identificado por sua tag, ou por uma classe, ou por uma ID, ou etc., e para o qual a regra será válida (por exemplo: `<p>`, `<h1>`, `<form>`, `.minhaclasse`, etc...);

Propriedade: é o atributo do elemento HTML ao qual será aplicada a regra (por exemplo: font, color, background, etc...).

Valor: é a característica específica a ser assumida pela propriedade (por exemplo: letra tipo arial, cor azul, fundo verde, etc...)

Na sintaxe de uma regra CSS, escreve-se o seletor e a seguir a propriedade e valor separados por dois pontos e entre chaves { }. Quando mais de uma propriedade for definida na regra, deve-se usar ponto-e-vírgula para separá-las. O ponto-e-vírgula é facultativo no caso de propriedade única e também após a declaração da última propriedade no caso de mais de uma.

No entanto é de boa técnica usar-se sempre o ponto-e-vírgula após cada regra para uma propriedade.

Ver os exemplos abaixo:

```
p {
font-size: 12px; /* ponto-e-vírgula é facultativo */
}

body {
color: #000000;
background: #FFFFFF;
font-weight: bold; /*ponto-e-vírgula é facultativo */
}
```


No exemplo abaixo, o **seletor** é o "documento todo" (body - a página web), a **propriedade** é o fundo do documento e o **valor** é a cor branca.

```
body {
background: #FFFFFF;
}
```

Se o valor for uma palavra composta, deverá estar entre aspas duplas " ", ou simples ':

```
h3 {
font-family: "Comic Sans MS"
}
```

Para **maior legibilidade** das folhas de estilo, é de boa prática usar linhas distintas para escrever cada uma das declarações — propriedade e seu valor —, como mostrado abaixo:

```
p {
text-align: right;
color: #FF0000;
}
```

Isto não é obrigatório! A regra abaixo tem o mesmo efeito da regra acima e ambas as sintaxes estão corretas:

```
p {text-align: right;color: #FF0000;}
```

NOTA: A razão do uso de ponto e vírgula na última declaração ou mesmo quando só há uma declaração é que durante a fase de projeto da Folha CSS quase sempre estaremos acrescentando novas declarações e a última declaração quase nunca é a última na fase de projeto. Assim, esta prática certamente nos poupará revisões por ter esquecido um ponto e vírgula.!!!!

Agrupamento de Seletores

Uma regra CSS quando válida para vários seletores, estes podem ser agrupados. Separe cada seletor com uma vírgula. No exemplo abaixo agrupamos todos os elementos cabeçalho. A cor de todos os cabeçalhos será verde.

```
h1, h2, h3, h4, h5, h6 {
color: #00FF00;
}
```

O seletor classe

Mas você não está restrito somente aos elementos HTML (tags) para aplicar regras de estilo!

Você pode "inventar" um nome e com ele criar uma **classe** a qual definirá as regras CSS. E o mais interessante das classes, é que elas podem ser aplicadas a **qualquer elemento** HTML. E mais ainda, você pode aplicar estilos diferentes para o mesmo tipo de elemento do HTML, usando classes diferentes para cada um deles.

A sintaxe para o seletor classe é mostrada abaixo. Elemento HTML mais um nome qualquer que você "inventa" precedido de . (ponto):

```
 elemento HTML.minhaclasse {
propriedade: valor;
}
```

Nota: Para o nome que você "inventa" evite usar números e caracteres especiais. Tanto quanto possível use só letras de a-z e de A-Z. Há restrições quanto ao uso de números e caracteres. Minha experiência e conselho: Use só letras!

Por exemplo: suponha que você queira ter dois tipos de parágrafos em seu documento: um parágrafo com letras na cor preta e um parágrafo com letras na cor azul.

```
p.corum {
color:#000000;
}

p.cordois {
color:#0000FF;
}
```

No seu documento HTML as regras seriam aplicadas conforme abaixo:

```
<p class ="corum"> este parágrafo terá cor preta.</p>

<p class ="cordois">
este parágrafo terá cor azul.
</p>
```

Este item foi atualizado em 2007/07/03

Em CSS 1 não é válido atribuir mais de uma classe para um elemento HTML. O exemplo abaixo está errado:

```
<p class ="corum" class ="cordois">
Aqui há um erro.
```

```
</p>
```

Nota: CSS 2 mudou este conceito, permitindo declarar mais de uma classe, desde que o nome das classes sejam separados por um espaço.

```
<p class ="corum cordois">
Aqui não há erro.
</p>
```


Ao criar uma classe você talvez queira que ela seja aplicável a qualquer elemento HTML. Neste caso basta que se omita o nome do elemento antes da classe. Por exemplo: a regra CSS a seguir pode ser aplicada a qualquer elemento HTML ao qual você deseja atribuir cor azul:

```
.cortres {
    color: #0000FF;
    }
```

No exemplo a seguir tanto o cabeçalho `<h2>` como o parágrafo `<p>` terão cor azul:

```
<h2 class="cortres">
Este cabeçalho é azul.
</h2>

<p class="cortres">
Este parágrafo é azul.
</p >
```

O seletor ID

O seletor ID difere do seletor de classe, por ser ÚNICO. Um seletor ID só pode ser aplicado a UM e somente UM elemento HTML dentro do documento.

Você pode "inventar" um nome e com ele criar uma **ID** a qual definirá as regras CSS. Uma **ID só pode ser aplicada a UM elemento** HTML.

A sintaxe para o seletor ID é mostrada abaixo. Um nome qualquer que você "inventa" precedido de # ("tralha", "jogo-da-velha" :-)):

```
#minhaID {
propriedade: valor;
        }
```

Nota: Para o nome que você "inventa" evite usar números e caracteres especiais. Tanto quanto possível use só letras de a-z e de A-Z. Há restrições quanto ao uso de números e caracteres. Minha experiência e conselho: Use só letras!

Inserindo comentários nas CSS

Você pode inserir comentários nas CSS para explicar seu código, e principalmente ajudá-lo a relembrar de como você estruturou e qual a finalidade de partes importantes do código. Daqui há alguns meses a menos que você seja um privilegiado, terá esquecido a maior parte daquilo que você levou horas para "bolar". O comentário introduzido no código, será ignorado pelo browser. Um comentário nas CSS começa com o "/*", e termina com " */". Veja o exemplo abaixo:

```
/* este é um comentário*/
p {
font-size: 14px;        /* este é outro comentário*/
color: #000000;
font-family: Arial, Serif;
}
```


Vinculando folhas de estilo a documentos

Os três tipos de vinculação de folhas de estilo

As folhas de estilo podem ser vinculadas a um documento de três maneiras distintas:

1. Importadas ou lincadas;
2. Incorporadas;
3. Inline.

Folha de estilo externa

Uma folha de estilo é dita externa, quando as regras CSS estão declaradas em um documento a parte do documento HTML. A folha de estilo é um arquivo separado do arquivo html e que tem a extensão .css

Uma folha de estilo externa é ideal para ser aplicada a várias páginas. Com uma folha de estilo externa , você pode mudar a aparência de um site inteiro mudando um arquivo apenas (o arquivo da folha de estilo).

O arquivo css da folha de estilo externa deverá ser lincado ou importado ao documento HTML, dentro da tag `<head>` do documento. A sintaxe geral para lincar uma folha de estilo chamada estilo.css é mostrada abaixo.

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

A sintaxe geral para importar uma folha de estilo chamada estilo.css é mostrada abaixo:

```
<head>
...........
<style type="text/css">
@import url("estilo.css");
</style>
..........
</head>
```


O browser lerá as regras de estilo do arquivo estilo.css, e formatará o documento de acordo com elas.

Uma folha de estilo externa pode ser escrita em qualquer editor de texto. O arquivo não deve conter nenhuma tag HTML. As folhas de estilo devem ser "salvas" com uma extensão .css

Folha de estilo incorporada ou interna

Uma folha de estilo é dita incorporada ou interna, quando as regras CSS estão declaradas no próprio documento HTML.

Uma folha de estilo incorporada ou interna, é ideal para ser aplicada a uma única página. Com uma folha de estilo incorporada ou interna, você pode mudar a aparência de somente um documento, aquele onde a folha de estilo esta incorporada.

As regras de estilo, válidas para o documento, são declaradas na seção `<head>` do documento com a tag de estilo `<style>`, conforme sintaxe mostrada abaixo:

```
<head>
...........
<style type="text/css">
<!--
body {
background: #000000;
url("imagens/minhaimagem.gif");
}
h3 {
color: #FF0000;
}
p {
margin-left: 15px;
padding:1.5em;
}
```

```
-->
</style>
..........
</head>
```

O browser lerá agora as regras de estilo na própria página, e formatará o documento de acordo com elas.

Nota: Um browser ignora normalmente as tags desconhecidas. Isto significa que um browser velho que não suporte estilos, ignorará a tag `<style>`, mas o conteúdo da tag será mostrado na tela. É possível impedir que um browser velho mostre o conteúdo da tag, "escondendo-o" através do uso da marcação de comentário do HTML.

Observe a inclusão dos símbolos `<!--` (abre comentário) `-->` (fecha comentário) no código acima.

Folha de estilo inline

Uma folha de estilo é dita inline, quando as regras CSS estão declaradas dentro da tag do elemento HTML.

Um estilo inline só se aplica a um elemento HTML. Ele perde muitas das vantagens de folhas de estilo pois mistura o conteúdo com a apresentação. Use este método excepcionalmente, como quando quiser aplicar um estilo a uma única ocorrência de um elemento.

A sintaxe para aplicar estilo inline é mostrada a seguir:

```
<p style="color:#000000; margin: 5px;">
Aqui um parágrafo de cor preta e com 5px nas 4  margens.
</p>
```

Folhas múltiplas de estilo

Se alguma propriedade for definida para um mesmo elemento em folhas de estilo diferentes, entrará em ação, o EFEITO CASCATA e prevalecerão os valores da folha de estilo mais específica.

Suponha, uma folha de estilo externa com as seguintes propriedades para o seletor `h2`:

```
h2 {
color: #FFCC00;
text-align: center;
font: italic 9pt Verdana, Sans-serif;
}
```

e, uma folha de estilo interna com as seguintes propriedades para o seletor `h2`:

```
h2 {
```

```
color: #FFCC00;
text-align: center;
font: italic 10pt Verdana, Sans-serif;
}
```

Se ambas as páginas estiverem vinculadas ao documento, como há um conflito no tamanho das letras para `<h2>`, prevalecerá a folha interna e a letra de `<h2>` terá o tamanho igual a 10 pt.

A propriedade font

As fontes nos elementos HTML

As propriedades para as fontes, definem as características (os valores na regra CSS) das letras que constituem os textos dentro dos elementos HTML.

As propriedades básicas para fontes e que abordaremos neste tutorial são as listadas abaixo:

- color:.....cor da fonte
- font-family:.....tipo de fonte
- font-size:.....tamanho de fonte
- font-style:.....estilo de fonte
- font-variant:.....fontes maiúsculas de menor altura
- font-weight:.....quanto mais escura a fonte é (negrito)
- font:.....maneira abreviada para todas as propriedades

Valores válidos para as propriedades da fonte

- **color:**
  1. código hexadecimal: #FFFFFF
  2. código rgb: rgb(255,235,0)
  3. nome da cor: red, blue, green...etc
- **font-family:**
  1. nome da família de fontes : define-se pelo nome da fonte, p. ex:"verdana", "helvetica", "arial", etc.
  2. nome da família genérica: define-se pelo nome genérico, p. ex:"serif", "sans-serif", "cursive", etc.
- **font-size:**
  1. xx-small
  2. x-small
  3. small
  4. medium
  5. large
  6. x-large
  7. xx-large
  8. smaller

9. larger
10. length: uma medida reconhecida pelas CSS (px, pt, em, cm, ...)
11. %

- **font-style:**
  1. normal: fonte normal na vertical
  2. italic: fonte inclinada
  3. oblique:fonte obliqua
- **font-variant:**
  1. normal: fonte normal
  2. small-caps: transforma em maiúsculas de menor altura
- **font-weight:**
  1. normal
  2. bold
  3. bolder
  4. lighter
  5. 100
  6. 200
  7. 300
  8. 400
  9. 500
  10. 600
  11. 700
  12. 800
  13. 900

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para um ou mais elementos HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Esta é a melhor e mais rápida maneira de você aprender CSS. Bons estudos! E faça ótimo proveito dos tutoriais.

color ... A cor da fonte

```
<html>
<head>
<style type="text/css">
<!--
h1 {color: #FF0000;}
h2 {color: #00FF00;}
p {color: rgb(0,0,255);}
-->
</style>
</head>
```

```
<body>
<h1>Cabeçalho com letras vermelhas</h1>
<h2>Cabeçalho com letras verdes</h2>
<p>Parágrafo com letras azuis</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Cabeçalho com letras vermelhas

Cabeçalho com letras verdes

Parágrafo com letras azuis

font-family...O tipo de fonte

```
<html>
<head>
<style type="text/css">
<!--
h2 {font-family: arial, helvetica, serif;}
p {font-family: sans-serif;}
-->
</style>
</head>
<body>
<h2>Família por nome: arial, helvetica, serif;</h2>
<p>Família genérica: sans-serif;<p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Família por nome: arial, helvetica, serif;

Família genérica: sans-serif;

Notas: A propriedade font-family é usada para definir uma lista de tipos de fontes.

O browser assume o primeiro nome que ele reconhece na lista.

Separar cada nome por uma vírgula e sempre prever um nome genérico.

Se o nome da fonte for composto (mais de uma palavra, p. ex: Comic Sans MS), usar aspas duplas no nome. Se estiver definindo um atributo de "style" em HTML, onde as aspas duplas já estão presentes usar no nome de fonte composto, aspas simples.

font-size...O tamanho de fonte

```
<html>
<head>
<style type="text/css">
<!--
h1 {font-size: 14px;}
h2 {font-size: smaller;}
p {font-size: 100%;}
-->
</style>
</head>
<body>
<h1>Letras com tamanho: 14px</h1>
<h2>Letras com tamanho: smaller</h2>
<p>Letras com tamanho:100%</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Letras com tamanho: 14px

Letras com tamanho: smaller

Letras com tamanho:100%

font-style...O estilo de fonte

```
<html>
<head>
<style type="text/css">
<!--
h1 {font-style: italic;}
h2 {font-style: normal;}
p {font-style: oblique;}
-->
</style>
</head>
<body>
<h1>Este é o estilo italic</h1>
<h2>Este é o estilo normal</h2>
<p>Este é o estilo oblique</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Este é o estilo italic

Este é o estilo normal

Este é o estilo oblique

font-variant...fontes maiúsculas "menores"

```
<html>
<head>
<style type="text/css">
<!--
h3 {font-variant: normal;}
p{font-variant: small-caps;}
-->
p</style>
</head>
<body>
<h3>Este cabeçalho com letras normais</h3>
<p>Este parágrafo com letras em "small-caps"</p>
</body>
</html>
```


Este é o efeito da folha de estilo acima:

Este cabeçalho com letras normais

ESTE PARÁGRAFO COM LETRAS EM "SMALL-CAPS"

font-weight...Peso das fontes - negrito (intensidade da cor)

```
<html>
<head>
<style type="text/css">
<!--
p {font-weight: bold;}
-->
</style>
</head>
<body>
<p>
Este é um parágrafo em negrito</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Este é um parágrafo em negrito

font...Todas as propriedades das fontes em uma declaração única

Esta é a maneira abreviada de você escrever uma regra para as propriedades das fontes.

A sintaxe geral é esta: `font: [style] [variant] [weight] [size] [/line-height] [family] | caption | icon | menu | message-box | small-caption | status-bar | inherit`

Você pode declarar todas ou algumas das propriedades.

Os valores `size` e `family` são obrigatórios. Os demais são facultativos (se você os omitir será adotado o valor default ou herdado do elemento pai).

Os valores `style`, `variant`, `weight` e `size`, podem ser declarados em qualquer ordem.


```
<html>
<head>
<style type="text/css">
<!--
p {
font: italic small-caps bold 14px
 "Comic Sans MS", sans-serif;
}
-->
</style>
</head>
<body>
<p>Parágrafo em declaração única</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

PARÁGRAFO EM DECLARAÇÃO ÚNICA

O valores `caption`, `icon`, `menu`, `message-box`, `small-caption` e `status-bar` referem-se às fontes usadas pelo sistema operacional do visitante do site e devem ser declarados **isolados** na propriedade `font`.

- caption.....fontes usadas em botões;
- icon.....fontes usadas em ícones;
- menu.....fontes usadas em menus;
- message-box...fontes ussadas em caixas de mensagens;
- small-caption...fontes usadas em pequenos controles;
- status-bar.....fontes usadas na barra de status;

O valor `inherit` é usado para herdar a fonte usada pelo elemento pai e também deve ser declarados **isolados** na propriedade `font`.

```
<html>
<head>
<style type="text/css">
<!--
.p1 {
font: status-bar;
```

```
}
.p2 {
font: inherit;
.p3 {
font: small-caption ;
}
}
-->
</style>
</head>
<body>
<p class="p1">Parágrafo com fonte status-bar</p>
<p class="p2">Parágrafo com fonte inherit</p>
<p class="p3">Parágrafo com fonte small-caption</p>
</body>
</html>
```


Este é o efeito da folha de estilo acima:

Parágrafo com fonte status-bar

Parágrafo com fonte inherit

Parágrafo com fonte small-caption

:: A propriedade text ::

Os textos nos elementos HTML

As propriedades para textos, definem as características (os valores na regra CSS) dos textos inseridos dentro dos elementos HTML.

As propriedades para textos são as listadas abaixo:

- color.....cor do texto;
- letter-spacing.....espaçamento entre letras;
- word-spacing.....espaçamento entre palavras;
- text-align.....alinhamento do texto;
- text-decoration.....decoração do texto;
- text-indent.....recuo do texto;
- text-transform.....forma das letras;
- direction.....direção do texto;
- white-space.....como o browser trata os espaços em branco;

Valores válidos para as propriedades do texto

- **color:**
    1. código hexadecimal: #FFFFFF
    2. código rgb: rgb(255,235,0)

3. nome da cor: red, blue, green...etc

- **letter-spacing:**
  1. normal: é o espaçamento default
  2. lenght: uma medida reconhecida pelas CSS (px, pt, em, cm, ...) São válidos valores negativos
- **word-spacing:**
  1. normal: é o espaçamento default
  2. lenght: uma medida reconhecida pelas CSS (px, pt, em, cm, ...) São válidos valores negativos
- **text-align:**
  1. left: alinha o texto a esquerda
  2. right: alinha o texto a direita
  3. center: alinha o texto no centro
  4. justify: força o texto a ocupar toda a extensão da linha da esquerda a direita
- **text-decoration:**
  1. none: nenhuma decoração
  2. underline: coloca sublinhado no texto
  3. overline: coloca um sobrelinhado no texto
  4. line-through: coloca uma linha em cima do texto
  5. blink: faz o texto piscar
- **text-indent:**
  1. lenght: uma medida reconhecida pelas CSS (px, pt, em, cm, ...)
  2. % : porcentagem da largura do elemento pai
- **text-transform:**
  1. none: texto normal
  2. capitalize: todas as primeiras letras do texto em maiúsculas
  3. uppercase: todas as letras do texto em maiúsculas
  4. lowercase: todas as letras do texto em minúsculas
- **direction:**
  1. ltr: texto escrito da esquerda para a direita
  2. rtl: texto escrito da direita para a esquerda
- **white-space:**
  1. normal: os espaços em branco serão ignorados pelo browser
  2. pre: os espaços em branco serão preservados pelo browser
  3. nowrap: o texto será apresentado todo ele numa linha única na tela. Não há quebra de linha até ser encontrada uma tag

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para um ou mais elementos HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Esta é a melhor e mais rápida maneira de você aprender CSS. Bons estudos! E faça ótimo proveito dos tutoriais.

color ... A cor do texto

```
<html>
<head>
<style type="text/css">
<!--
h1 {color: #FF0000;}
h2 {color: #00FF00;}
p {color: rgb(0,0,255);}
-->
</style>
</head>
<body>
<h1>Este  cabeçalho é vermelho</h1>
<h2>Este cabeçalho é verde</h2>
<p>Este parágrafo é azul</p>
</body>
</html>
```


Este é o efeito da folha de estilo acima:

Este cabeçalho é vermelho

Este cabeçalho é verde

Este parágrafo é azul

letter-spacing...O espaço entre letras

```
<html>
<head>
<style type="text/css">
<!--
h2 {letter-spacing: 1.2em;}
p {letter-spacing: 0.4cm;}
-->
</style>
</head>
<body>
<h2> Este é o cabeçalho</h2>
<p> Este é o parágrafo</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

E s t e é o c a b e ç a l h o

E s t e é o p a r a g r á f o

word-spacing...O espaço entre palavras

```
<html>
<head>
<style type="text/css">
<!--
h2 {word-spacing: 1.8em;}
p {word-spacing: 80px;}
-->
</style>
</head>
<body>
<h2> Este é o cabeçalho</h2>
<p> Este é o parágrafo</p>
</body>
</html>
```


Este é o efeito da folha de estilo acima:

Este é o cabeçalho

Este é o paragráfo

text-align...Alinhar o texto

```
<html>
<head>
<style type="text/css">
<!--
h1 {text-align: left;}
h2 {text-align: center;}
h3 {text-align: right;}
p {text-align: justify;}
-->
</style>
</head>
<body>
<h1>Este é o cabeçalho 1</h1>
<h2>Este é o cabeçalho 2</h2>
<h3>Este é o cabeçalho 3</h3>
<p>Este é o parágrafo cujo texto ...</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Este é o cabeçalho 1

Este é o cabeçalho 2

Este é o cabeçalho 3

Este é o parágrafo cujo texto foi alongado para mais de duas linhas para que você possa visualizar o efeito de `text-align: justify` que força o texto a estender-se desde a direita até a esquerda.

text-decoration...Decoração do texto


```
<html>
<head>
<style type="text/css">
<!--
h1 {text-decoration: underline;}
h2 {text-decoration: line-through;}
h3 {text-decoration: overline;}
a {text-decoration: none;}
-->
</style>
</head>
<body>
<h1>Texto com sublinhado</h1>
<h2>Texto com linha em cima</h2>
<h3>Texto com sobrelinhado</h3>
<p>
<a href="http://www.maujor.com">
Este é um link sem sublinhado</a>
</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Texto com sublinhado

~~Texto com linha em cima~~

Texto com sobrelinhado

Este é um link sem sublinhado

text-indent...Recuo do texto

```
<html>
<head>
<style type="text/css">
<!--
h3 {text-indent: 80px;}
p {text-indent: 3em;}
-->
```

```
</style>
</head>
<body>
<h3>Texto com recuo de 80 pixel</h3>
<p>Texto com recuo de 3.0em</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Texto com recuo de 80 pixeis

Texto com recuo de 3.0em

text-transform...Forma das letras do texto

```
<html>
<head>
<style type="text/css">
<!--
h1 {text-transform: none;}
h2 {text-transform: capitalize;}
h3 {text-transform: uppercase;}
h4 {text-transform: lowercase;}
-->
</style>
</head>
<body>
<h1>Texto com letras como digitadas</h1>
<h2>Texto com primeira letra das palavras, maiúsculas</h2>
<h3>Texto com todas letras, maiúsculas</h3>
<h4>Texto com letras minúsculas</h4>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Texto com letras como digitadas

Texto com primeira letra das palavras, maiúsculas

TEXTO COM TODAS LETRAS, MAIÚSCULAS

Texto com letras minúsculas

:: A propriedade `margin` ::

As margens nos elementos HTML

A propriedade para margens, define um valor para espessura das margens dos elementos HTML.

As propriedades para margens são as listadas abaixo:

- margin-top.....define a margem superior;
- margin-right.....define a margem direita;
- margin-bottom.....define a margem inferior;
- margin-left.....define a margem esquerda;
- margin.....**maneira abreviada para todas as margens**

Valores válidos para a propriedade `margin`

1. auto: valor default da margem
2. length: uma medida reconhecida pelas CSS (px, pt, em, cm, ...)
3. %: porcentagem da largura do elemento pai

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS válidas para um elemento HTML, e, definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Bons estudos! E faça ótimo proveito do tutorial.

Nota: Coloquei um fundo cinza mais escuro nos exemplos para facilitar a visualização.

`margin-top`...a margem superior

```
<html>
<head>
<style type="text/css">
<!--
p {margin-top: 2cm;}
-->
</style>
</head>
<body>
<p>Uma margem superior de 2cm</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Uma margem superior de 2cm

`margin-right`...a margem direita

```
<html>
<head>
<style type="text/css">
<!--
p {margin-right: 300px;}
-->
</style>
</head>
<body>
<p>Uma margem direita de 300px nesta
 frase mais longa dentro do parágrafo</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Uma margem direita de 300px nesta
frase mais longa dentro do parágrafo

`margin-bottom`...a margem inferior

```
<html>
<head>
<style type="text/css">
<!--
p {margin-bottom: 2em;}
-->
</style>
</head>
<body>
<p>Uma margem inferior de 2.0em</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Uma margem inferior de 2.0em

`margin-left`...a margem esquerda

```
<html>
<head>
<style type="text/css">
<!--
p {margin-left: 10%;}
-->
</style>
</head>
```

```
<body>
<p>Uma margem esquerda de 10%</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Uma margem esquerda de 10%

`margin`...todas as quatro margens em uma declaração única

A propriedade da margin permitem que você controle o espaçamento em volta dos elementos HTML. São válidos **valores negativos** para margem, com o objetivo de sobrepor elementos.

Em declaração única a ordem das margens é: **superior, direita, inferior** e **esquerda**.

Há quatro modos de se declarar abreviadamente as margens:

1. `margin`: valor1.....as 4 margens terão valor1;
2. `margin`: valor1 valor2.....margem superior e inferior terão valor1 - margem direita e esquerda terão valor2
3. `margin`: valor1 valor2 valor3.....margem superior terá valor1 - margem direita e esquerda terão valor2 - margem inferior terá valor3
4. `margin`: valor1 valor2 valor3 valor4....margens superior, direita, inferior e esquerda nesta ordem.

```
<html>
<head>
<style type="text/css">
<!--
p {margin: 20px 40px 80px 5px;}
-->
</style>
</head>
<body>
<p>Uma margem superior de 20px, uma margem direita de 40px,
uma margem inferior de 80px e uma margem esquerda de 5px</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Uma margem superior de 20px, uma margem direita de 40px, uma margem inferior de 80px e uma margem esquerda de 5px

As bordas nos elementos HTML

As propriedades para as bordas, definem as características (os valores na regra CSS) das quatro bordas de um elemento HTML.

As propriedades para as bordas são as listadas abaixo:

- border-width:.....espessura da borda
- border-style:.....estilo da borda
- border-color:.....cor da borda
- ----------
- border-top-width:.....espessura da borda superior
- border-top-style:.....estilo da borda superior
- border-top-color:.....cor da borda superior
- ----------
- border-right-width:.....espessura da borda direita
- border-right-style:.....estilo da borda direita
- border-right-color:.....cor da borda direita
- ----------
- border-bottom-width:.....espessura da borda inferior
- border-bottom-style:.....estilo da borda inferior
- border-bottom-color:.....cor da borda inferior
- ----------
- border-left-width:.....espessura da borda esquerda
- border-left-style:.....estilo da borda esquerda
- border-left-color:.....cor da borda esquerda
- ----------
- border-top:...**maneira abreviada para todas as propriedades da borda superior**
- border-right:..**maneira abreviada para todas as propriedades da borda direita**
- border-bottom:..**maneira abreviada para todas as propriedades da borda inferior**
- border-left:..**maneira abreviada para todas as propriedades da borda esquerda**
- border:.....**maneira abreviada para todas as quatro bordas**

Valores válidos para as propriedades das bordas

- **color:**
  1. código hexadecimal: #FFFFFF
  2. código rgb: rgb(255,235,0)
  3. nome da cor: red, blue, green...etc
- **style:**
  1. none: nenhuma borda
  2. hidden: equivalente a none
  3. dotted: borda pontilhada
  4. dashed: borda tracejada
  5. solid: borda contínua
  6. double: borda dupla

7. groove: borda entalhada
8. ridge: borda em ressalto
9. inset: borda em baixo relevo
10. outset: borda em alto relevo

- **width:**
  1. thin: borda fina
  2. medium: borda média
  3. thick: borda grossa
  4. length: uma medida reconhecida pelas CSS (px, pt, em, cm, ...)

Vamos a seguir analisar algumas delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para um ou mais elementos HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Bons estudos! E faça ótimo proveito dos tutoriais.

border-width, border-style e border-color

```
<html>
<head>
<style type="text/css">
<!--
h3   {
border-width: medium;
border-style: solid;
border-color: #0000FF;
    }
p   {
border-width: 6px;
border-style: dashed;
border-color: #FF0000;
    }
-->
</style>
</head>
<body>
<h3>Borda média, contínua e azul</h3>
<p>Borda 6px, tracejada e vermelha</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Borda média, contínua e azul

Nota: A propriedade `border-color` não é reconhecida pelo Internet Explorer se for usada isolada. Use a propriedade `border-style` para ser reconhecida pelo Internet Explorer.

Nota: A propriedade `border-color` não é reconhecida pelo Netscape. Use a propriedade `border` para ser reconhecida pelo Netscape.

border-style

Abaixo os estilos de bordas obtidos com a declaração `border-style: valor` (dotted, dashed, etc..)

Borda dotted

Borda dashed

Borda solid

Borda double

Borda groove

Borda ridge

Borda inset

Borda outset

border-width

Estude o código abaixo e tire suas conclusões de como obter outros efeitos com espessuras de bordas

```
<html>
<head>
<style type="text/css">
p    {
border-style: solid;
border-bottom-width: 10px;
border-top-width: 0px;
border-right-width: 0px;
```

```
border-left-width: 0px;
 }
</style>
</head>
<body>
<p>Borda com espessura inferior de 10px</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Borda com espessura inferior de 10px

Nota: A propriedade `border-bottom-width` não é reconhecida pelo Internet Explorer se usada isoladamente. Use `border-style` para ser reconhecida pelo Internet Explorer.

Definir a espessura das bordas superior, esquerda e direita

Proceda de modo semelhante ao mostrado acima.

---

border (declaração única)

Esta é a maneira abreviada de você escrever uma regra para as propriedades das bordas.

Você pode declarar todas as tres propriedadesdas bordas em uma regra única:

A sintaxe geral é esta: `border: size style color;` em qualquer ordem.
Nota: Aconselho a escolher, e adotar, sempre a mesma ordem.

```
<html>
<head>
<style type="text/css">
<!--
p     {
border: thick  groove  rgb(255,0,255)
    }
</style>
</head>
<body>
<p>Bordas em declaração única</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Propriedades CSS das bordas

As propriedades das bordas permitem que você controle o estilo a cor e a espessura das bordas de um elemento HTML.
As propriedades são muitas e como você viu, podem ser declaradas para cada uma das quatro bordas individualmente.

Neste tutorial abordei sumariamente algumas das propriedades, fornecendo as bases para seus estudos mais completos.

:: A propriedade padding ::

Os espaçamentos nos elementos HTML

A propriedade para espaçamentos (alguns traduzem como "enchimento"), define um valor para os espaçamentos entre o conteúdo e as bordas dos elementos HTML.

As propriedades para espaçamentos são as listadas abaixo:

- padding-top.....define a espaçamento superior;
- padding-right.....define a espaçamento direita;
- padding-bottom.....define a espaçamento inferior;
- padding-left.....define a espaçamento esquerda;
- padding.....**maneira abreviada para todas os espaçamentos**

Valores válidos para as propriedades de espaçamento

1. auto: valor default da margem
2. length: uma medida reconhecida pelas CSS (px, pt, em, cm, ...)
3. %: porcentagem da largura do elemento pai

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para um elemento HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Bons estudos! E faça ótimo proveito dos tutorial.

Nota: Coloquei um fundo cinza mais escuro nos exemplos para facilitar a visualização.

`padding-top`...o espaçamento superior

```
<html>
<head>
<style type="text/css">
<!--
p {padding-top: 2cm;}
-->
</style>
</head>
<body>
<p>Um espaçamento superior de 2cm</p>
</body>
</html>
```


Este é o efeito da folha de estilo acima:

Um espaçamento superior de 2cm

`padding-right`...o espaçamento direito

```
<html>
<head>
<style type="text/css">
<!--
p {padding-right: 300px;}
-->
</style>
</head>
<body>
<p>Um espaçamento direito de 300px nesta
 frase mais longa dentro do parágrafo</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Um espaçamento a direita de 300px nesta frase mais longa dentro do parágrafo

`padding-bottom`...o espaçamento inferior

```
<html>
<head>
<style type="text/css">
<!--
p {padding-bottom: 2em;}
-->
</style>
</head>
<body>
<p>Um espaçamento inferior de 2.0em</p>
```

```
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Um espaçamento inferior de 2.0em

`padding-left`...o espaçamento esquerdo


```
<html>
<head>
<style type="text/css">
<!--
p {padding-left: 10%;}
-->
</style>
</head>
<body>
<p>Um espaçamento esquerdo de 10%</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Um espaçamento esquerdo de 10%

`padding`...todos os quatro espaçamentos em uma declaração única

A propriedade padding permite que você controle o espaçamento entre o conteúdo e as bordas dos elementos HTML.

Não são válidos valores negativos para espaçamento.

Em declaração única a ordem das espaçamentos é: **superior, direito, inferior** e **esquerdo**.

Há quatro modos de se declarar abreviadamente os espaçamentos:

1. padding: valor1.....os 4 espaçamentos terão valor1;
2. padding: valor1 valor2.....espaçamento superior e inferior terão valor1 - espaçamento direito e esquerdo terão valor2
3. padding: valor1 valor2 valor3.....espaçamento superior terá valor1 - espaçamento direito e esquerdo terão valor2 - espaçamento inferior terá valor3
4. padding: valor1 valor2 valor3 valor4....os espaçamentos superior, direito, inferior e esquerdo nesta ordem.

```
<html>
<head>
<style type="text/css">
```

```
<!--
p {padding: 20px 40px 80px 5px;}
-->
</style>
</head>
<body>
<p>Um espaçamento superior de 20px,
 um espaçamento direito de 40px,
um espaçamento inferior de 80px
e um espaçamento esquerdo de 5px</p>
</body>
</html>
```


Este é o efeito da folha de estilo acima:

Um espaçamento superior de 20px, um espaçamento direito de 40px, um espaçamento inferior de 80px e um espaçamento esquerdo de 5px

:: A propriedade background ::

O fundo dos elementos HTML

A propriedade background define as características (os valores na regra CSS) do fundo dos elementos HTML.

As propriedades background são as listadas abaixo:

- background-color..... cor do fundo;
- background-image..... imagem de fundo;
- background-repeat..... maneira como a imagem de fundo é posicionada;
- background-attachment.....se a imagem de fundo "rola" ou não com a tela;
- background-position.....como e onde a imagem de fundo é posicionada;
- background.....**maneira abreviada para todas as propriedades**;

Valores válidos para as propriedades do fundo

- **background-color:**
  1. código hexadecimal: #FFFFFF
  2. código rgb: rgb(255,235,0)
  3. nome da cor: red, blue, green...etc
  4. transparente: transparent
- **background-image:**
  1. URL: url(caminho/imagem.gif)
- **background-repeat:**
  1. não repete: no-repeat
  2. repete vertical e horizontal: repeat
  3. repete vertical: repeat-y
  4. repete horizontal: repeat-x
- **background-attachment:**

1. imagem fixa na tela: fixed
2. imagem "rola" com a tela: scroll

- **background-position:**
  1. x-pos y-pos
  2. x-% y-%
  3. top left
  4. top center
  5. top right
  6. center left
  7. center center
  8. center right
  9. bottom left
  10. bottom center
  11. bottom right

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para um ou mais elementos HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Esta é a melhor e mais rápida maneira de você aprender CSS. Bons estudos! E faça ótimo proveito dos tutoriais.

A cor do fundo

```
<html>
<head>
<style type="text/css">
<!--
body {background-color: #CCFFFF;} /*azul claro*/
h2 {background-color: #FF0000;}   /* vermelho */
p {background-color: #00FF00;}    /*  verde   */
-->
</style>
</head>
<body>
<h2>Estude CSS</h2>
<p>Com CSS você controla melhor seu layout</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Estude CSS

A imagem de fundo

```
<html>
<head>
<style type="text/css">
<!--
body { background-image: url("/images/css.gif");}
-->
</style>
</head>
<body>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Repetir verticalmente a imagem de fundo

```
<html>
<head>
<style type="text/css">
<!--
body {
background-image: url("/images/css.gif");
background-repeat: repeat-y;
}
-->
</style>
</head>
<body>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Repetir horizontalmente a imagem de fundo

```
<html>
<head>
style type="text/css">
<!--
body {
background-image: url("/images/css.gif");
background-repeat: repeat-x;
}
-->
</style
</head>
<body>
```

```
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Posicionar uma imagem de fundo

```
<html>
<head>
<style type="text/css">
<!--
body {
background-image: url("/images/css.gif");
background-repeat: no-repeat;
background-position: 200px 70px;
}
-->
</style>
</head>
<body>
</body>
</html>
```


Este é o efeito da folha de estilo acima: a imagem esta posicionada a 200 pixel da margem esquerda e 70 pixel da margem superior

Ajustar uma imagem de fundo fixa, que não "rola" com a tela.

```
<html>
<head>
<style type="text/css">
<!--
body {
background-image: url("/images/css.gif");
background-repeat: no-repeat;
background-attachment: fixed;
}
-->
</style>
</head>
<body>
</body>
</html>
```

Este é o efeito da aplicação das regras CSS acima em uma página web.

Todas as propriedades do fundo em uma declaração única

Esta é a maneira abreviada de você escrever uma regra para as propriedades do fundo.

Você pode declarar todas ou algumas das propriedades estudadas em uma regra única:

A sintaxe geral é esta:
`background: color image repeat attachment position;`
em qualquer ordem, podendo ser omitido um mais valores.

Veja o exemplo abaixo:

```
<html>
<head>>
<style type="text/css">/>
<!--
body {
background: #00FF00 url("css.gif")
no-repeat fixed 200px 70px;
}
-->
</style>
</head>
```


:: A propriedade `list` ::

Mudando o estilo das listas HTML

A propriedade list define as características (valores) das listas HTML.

As propriedades list são as listadas abaixo:

- list-style-image..... imagem como marcador da lista;
- list-style-position.....onde o marcador da lista é posicionado;
- list-style-type.....tipo do marcador da lista;
- list-style.....**maneira abreviada para todas as propriedades**;

Valores válidos para as propriedades do lista

- `list-style-image`:
    1. none
    2. URL: url(caminho/marcador.gif)
- `list-style-position`:
    1. outside: marcador fora do alinhamento do texto
    2. inside: marcador alinhado com texto
- `list-style-type`:
    1. none: sem marcador
    2. disc: círculo (bolinha cheia)
    3. circle: circunferência (bolinha vazia)
    4. square: quadrado cheio
    5. decimal: números 1, 2, 3, 4, ...
    6. decimal-leading-zero
    7. lower-roman: romano minúsculo i, ii, iii, iv, ...

8. upper-roman: romano maiúsculo I, II, III, IV, ...
9. lower-alpha: letra minúscula a, b, c, d, ...
10. upper-alpha: letra maiúscula A, B, C, D, ...
11. lower-greek
12. lower-latin
13. upper-latin
14. hebrew
15. armenian
16. georgian
17. cjk-ideographic
18. hiragana
19. katakana
20. hiragana-iroha
21. katakana-iroha

Os tipos de 11 a 20 são de uso específico e sem suporte total pelos navegadores atuais e não serão tratados neste tutorial.

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para o elemento lista HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Bons estudos! E faça ótimo proveito do tutorial.

`list-style-image`...imagem para marcadores de lista

Este exemplo demonstra como definir uma imagem de marcador de listas

```
<html>
<head>
<style type="text/css">
<!--
ul
{
list-style-image: url("seta.gif");
}
-->
</style>
</head>
<body>
<ul>
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ul>
```

```
</body>
</html>
```

A folha de estilo acima resultará nesta lista:

- Item um
- Item dois
- Item tres

list-style-position...posição dos marcadores de lista

Este exemplo demonstra como posicionar os marcadores de listas

```
html>
<head>
<style type="text/css">
<!--
ul.inside
{
list-style-position: inside;
}
ul.outside
{
list-style-position: outside;
}
-->
</style>
</head>
<body>
<ul class="inside">
<li>Este texto destina-se a demonstrar o
valor: "inside" dos marcadores de listas.</li>
<li>E aqui continuamos com mais texto para
 fixar o valor:"inside" dosmarcadores de listas.</li>
</ul>
<ul class="outside">
<li>Este texto destina-se a demonstrar o
 valor: "outside" dos marcadores de listas.</li>
<li>E aqui continuamos com mais texto para
 fixar o valor:"outside" dos marcadores de listas.</li>
</ul>
</body>
</html>
```

A folha de estilo acima resultará nesta lista:

- Este texto destina-se a demonstrar o valor: "inside" dos marcadores
- E aqui continuamos com mais texto para fixar o valor:"inside" dos marcadores de listas.

- Este texto destina-se a demonstrar o valor: "outside" dos marcadores
- E aqui continuamos com mais texto para fixar o valor:"outside" dos marcadores de listas.

Definir os marcadores de listas não ordenadas

Este exemplo demonstra como definir os marcadores de listas não ordenadas.

```
<html>
<head>
<style type="text/css">
<!--
ul.none {
list-style-type: none;
}
ul.disc {
list-style-type: disc;
}
ul.circle {
list-style-type: circle;
}
ul.square {
list-style-type: square;
}
-->
</style>
</head>
<body>
<ul class="none">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ul>
<ul class="disc">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ul>
<ul class="circle">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ul>
<ul class="square">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ul>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

- Item um
- Item dois
- Item tres

- Item um
- Item dois
- Item tres

- Item um
- Item dois
- Item tres

- Item um
- Item dois
- Item tres

Definir os marcadores de listas ordenadas

Este exemplo demonstra como definir os marcadores de listas ordenadas.

```
<html>
<head>
<style type="text/css">
<!--
ol.decimal
{
list-style-type: decimal;
}
ol.lroman
{
list-style-type: lower-roman;
}
ol.uroman
{
list-style-type: upper-roman;
}
ol.lalpha
{
list-style-type: lower-alpha;
}
ol.ualpha
{
list-style-type: upper-alpha;
}
-->
</style>
</head>
<body>
<ol class="decimal">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ol>
<ol class="lroman">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ol>
<ol class="uroman">
```

```
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ol>
<ol class="lalpha">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ol>
<ol class="ualpha">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ol>
</body>
</html>
```


Este é o efeito da folha de estilo acima:

1. Item um
2. Item dois
3. Item tres

1. Item um
2. Item dois
3. Item tres

1. Item um
2. Item dois
3. Item tres

1. Item um
2. Item dois
3. Item tres

1. Item um
2. Item dois
3. Item tres

`list-style`...duas propriedades das listas em uma declaração única

Esta é a maneira abreviada de você escrever uma regra para as propriedades das listas.

Você pode declarar duas das propriedades estudadas em uma regra única:

A sintaxe geral é esta: `list-style: position; imagem` ou `list-style: position; type` podendo inverter a ordem.

Veja o exemplo abaixo:

```
<html>
<head>
<style type="text/css">
<!--
ul
{
list-style: inside url("seta.gif");
}
-->
</style>
</head>
<body>
<ul>
<li>Texto para demonstrar a propriedade
 de declaração única  para listas usando
CSS - Folhas de Estilo em Cascata;</li>
<li>Item dois;</li>
<li>Item tres.</li>
</ul>
</body>
</html>
```


A folha de estilo acima resultará nesta lista:

- Texto para demonstrar a propriedade de declaração única para listas usando CSS - Folhas de Estilo em Cascata;
- Item dois;
- Item tres.

:: Pseudo-elementos CSS ::

Sintaxe

São usados em CSS, para adicionar efeitos a um seletor, ou a parte de um seletor.

A sintaxe dos pseudo-elementos:

```
seletor:pseudo-elemento {propriedade: valor;}
```

As classes em CSS podem também ser usadas com pseudo-elementos.

Esta regra permite que você defina diferentes efeitos para pseudo-elementos localizados em diferentes lugares em uma mesma página.

```
seletor.classe:pseudo-elemento {propriedade: valor;}
```

O pseudo-elemento `first-letter`

O pseudo-elemento `first-letter` é usado para obter um efeito especial na **primeira letra** de um texto.

```
<html>
<head>
<style type="text/css">
p {
font-size: 12pt
}
p:first-letter {
font-size:300%;
}
</style>
</head>
<body>
<p>Este texto destina-se a demonstrar o
 pseudo-elemento first-letter, bla...bla...bla...
bla... bla...bla...bla...bla...bla... bla...bla...
bla... bla...bla...bla...bla...bla... bla...bla...</p>
</body>
</html>
```

O código acima produzirá esse efeito

Este texto destina-se a demosnstrar o pseudo-elemento first-letter, bla...bla...bla... bla... bla...bla...bla...bla...bla... bla...bla... bla... bla...bla...bla...bla...bla... bla...bla... bla... bla...bla...bla...bla...bla... bla...bla... bla... bla...bla...bla...bla...bla... bla...bla...

O pseudo-elemento `first-letter` somente pode ser usado com elementos de bloco.

Propriedades aplicáveis ao pseudo-elemento `first-letter`

- font - propriedades de letras
- color - propriedades de cores
- background - propriedades de fundo
- margin - propriedades de margens
- padding - propriedades de espaçamentos
- border - propriedades de bordas
- text-decoration
- vertical-align (somente para "float: none)
- text-transform
- line-height
- float
- clear

O pseudo-elemento `first-line`

O pseudo-elemento `first-line` é usado para obter um efeito especial na **primeira linha** de um texto.

```
<html>
<head>
<style type="text/css">
p {
```

```
font-size: 12pt
}
p:first-line {
color: #0000FF;
font-variant: small-caps;
}
</style>
</head>
<body>
<p>Um texto qualquer dentro
 de um pseudo-elemento first-line,
 para um  efeito especial na primeira linha</p>
</body>
</html>
```


O código acima produzirá esse efeito

Um texto qualquer dentro de um pseudo-elemento first-line, para um efeito especial na primeira linha. Notar a mudança de cor e o tipo de letra small-caps na primeira linha.

No exemplo acima toda a primeira linha sofre o efeito da definição do pseudo-elemento. A "quebra" da linha depente do tamanho da janela do browser.

O pseudo-elemento `first-line` somente pode ser usado com elementos de bloco.

Propriedades aplicáveis ao pseudo-elemento `first-line`

- font - propriedades de letras
- color - propriedades de cores
- background - propriedades de fundo
- word-spacing - espaçamento entre palavras
- letter-spacing - espaçamento entre letras
- text-decoration
- vertical-align
- text-transform
- line-height
- clear

Pseudo-elementos em classes CSS

Pseudo-elementos podem ser combinados com classes CSS

```
<html>
<head>
<style type="text/css">
p.combinado:first-letter {
color: #FF0000;
font-size:xx-large;
}
</style>
</head>
<body>
```

```
<p class="combinado"> Uma frase com efeito
 especial combinado </p>
</body>
</html>
```

O código acima produzirá esse efeito

Uma frase com efeito especial combinado

As propriedades `line-height` e `margin`

A propriedade CSS de dimensionamento `line-height` permite controlar o espaçamento entre linhas e a propriedade CSS `margin` permite controlar o espaçamento entre elementos HTML.

Observe abaixo o código HTML para um texto composto de dois parágrafos:

```
<html>
<head>
</head>
<body>

<p>
1o. Parágrafo....Lorem ipsum dolor sit
amet, consectetuer adipiscing elit.
 Nulla pharetra egestas neque.
Duis dolor lacus, volutpat ac,
vestibulum nec, suscipit a, felis.
Aenean pharetra orci id elit.
Duis non dui. Suspendisse potenti.
Ut ac risus. Etiam dignissim.
Quisque nec felis.
</p>

<p>
2o.Parágrafo.......Sed blandit est non
ante. Ut imperdiet sagittis mi.
Sed gravida sodales nisl. Ut hendrerit
ipsum eu enim. Duis tempus consequat mauris.
In hac habitasse platea dictumst.
Vivamus lectus justo, commodo in, rutrum non,
eleifend eget, pede. Sed ac lacus. In tortor.
</p>

</body>
</html>
```

O código acima é renderizado pelo navegador conforme mostrado abaixo.

Notar a distância entre as linhas em cada parágrafo, ou seja as entrelinhas (não confunda com distância entre parágrafos):

1o. Parágrafo....Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nulla pharetra egestas neque. Duis dolor lacus, volutpat ac, vestibulum nec, suscipit a, felis. Aenean pharetra orci id elit. Duis non dui. Suspendisse potenti. Ut ac risus. Etiam dignissim. Quisque nec felis.

2o.Parágrafo.....Sed blandit est non ante. Ut imperdiet sagittis mi. Sed gravida sodales nisl. Ut hendrerit ipsum eu enim. Duis tempus consequat mauris. In hac habitasse platea dictumst. Vivamus lectus justo, commodo in, rutrum non, eleifend eget, pede. Sed ac lacus. In tortor.

Alterando o espaçamento entre linhas

No código HTML mostrado acima vamos inserir uma regra CSS para `line-height` que é a propriedade CSS que controla as entrelinhas. Observe abaixo o mesmo código com a regra, definindo uma entrelinha igual a 200%.

Nota: A entrelinha default do bronwser é 100%.

Você pode usar qualquer medida de comprimento, válida em CSS (px, cm, em, %, in...) para o valor da propriedade `line-height`.

```
<html>
<head>
<style type="text/css">
<!--
p {
line-height:200%;
}
-->
</style>

</head>
<body>

<p>
1o. Parágrafo....Lorem ipsum dolor sit
amet, consectetuer adipiscing elit.
 Nulla pharetra egestas neque.
Duis dolor lacus, volutpat ac,
vestibulum nec, suscipit a, felis.
Aenean pharetra orci id elit.
Duis non dui. Suspendisse potenti.
Ut ac risus. Etiam dignissim.
Quisque nec felis.
</p>

<p>
2o.Parágrafo.......Sed blandit est non
ante. Ut imperdiet sagittis mi.
Sed gravida sodales nisl. Ut hendrerit
ipsum eu enim. Duis tempus consequat mauris.
In hac habitasse platea dictumst.
Vivamus lectus justo, commodo in, rutrum non,
eleifend eget, pede. Sed ac lacus. In tortor.
```

```
</p>

</body>
</html>
```

O código acima é renderizado pelo navegador conforme mostrado abaixo.

Notar que a entrelinha que era default 100%, agora está 200% ou seja dobrou:

Nota: Faça algumas experiências com o valor de `line-height`, usando inclusive valores abaixo de 100% e também outras medidas válidas (por exemplo: 12px, 2.3em, 3cm...etc...) e você vai constatar que tem o controle total das entrelinhas.

1o. Parágrafo....Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nulla pharetra egestas neque. Duis dolor lacus, volutpat ac, vestibulum nec, suscipit a, felis. Aenean pharetra orci id elit. Duis non dui. Suspendisse potenti. Ut ac risus. Etiam dignissim. Quisque nec felis.

2o.Parágrafo.....Sed blandit est non ante. Ut imperdiet sagittis mi. Sed gravida sodales nisl. Ut hendrerit ipsum eu enim. Duis tempus consequat mauris. In hac habitasse platea dictumst. Vivamus lectus justo, commodo in, rutrum non, eleifend eget, pede. Sed ac lacus. In tortor.

E o espaçamento (a distância) entre os parágrafos?

Aqui também o controle é todo seu via CSS.

E quem dita as regras para este espaçamento é a propriedade `margin`.

Vamos acresentar mais uma regra CSS no nosso código.

Se voce não lembra da propriedade `margin`, leia este tutorial sobre margens

```
<html>
<head>
<style type="text/css">
<!--
p {
line-height:200%;
margin: 40px 0  40px 0;
-->
</style>
```

```
</head>
<body>

<p>
1o. Parágrafo....Lorem ipsum dolor sit
amet, consectetuer adipiscing elit.
 Nulla pharetra egestas neque.
Duis dolor lacus, volutpat ac,
vestibulum nec, suscipit a, felis.
Aenean pharetra orci id elit.
Duis non dui. Suspendisse potenti.
Ut ac risus. Etiam dignissim.
Quisque nec felis.
</p>

<p>
2o.Parágrafo.......Sed blandit est non
ante. Ut imperdiet sagittis mi.
Sed gravida sodales nisl. Ut hendrerit
ipsum eu enim. Duis tempus consequat mauris.
In hac habitasse platea dictumst.
Vivamus lectus justo, commodo in, rutrum non,
eleifend eget, pede. Sed ac lacus. In tortor.
</p>

</body>
</html>
```


O código acima é renderizado pelo navegador conforme mostrado abaixo.

Notar que a entrelinha continua em 200% e agora o espaçamento entre parágrafos cresceu para 40 pixels, cumprindo a regra CSS, escrita.

1o. Parágrafo....Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nulla pharetra egestas neque. Duis dolor lacus, volutpat ac, vestibulum nec, suscipit a, felis. Aenean pharetra orci id elit. Duis non dui. Suspendisse potenti. Ut ac risus. Etiam dignissim. Quisque nec felis.

2o.Parágrafo.....Sed blandit est non ante. Ut imperdiet sagittis mi. Sed gravida sodales nisl. Ut hendrerit ipsum eu enim. Duis tempus consequat mauris. In hac habitasse platea dictumst. Vivamus lectus justo, commodo in, rutrum non, eleifend eget, pede. Sed ac lacus. In tortor.

Você deve ter notado que o espaçamento do 1o. parágrafo para a borda superior do quadro amarelo e também a do 2o. parágrafo para a borda inferior do quadro amarelo, ambas AUMENTARAM.

Sim, este aumento no espaçamento cumpriu o prescrito na nova regra, ou seja: 40 pixel de margem superior e 40 pixel de margem inferior nos parágrafos.

Mas lembre-se o controle é SEU. Tem como evitar este espaçamento não previsto :-)
Veja o item 1-) abaixo.

Dicas adicionais

1-) Para evitar aquele espaçamento referido acima, crie e aplique uma classe no parágrafo superior com `margin-top: 0;` (ou n pixels) e outra classe ao parágrafo inferior com `margin-bottom: 0;` (ou n pixels);

Você pode também declarar: `margin: 0 0 40px 0;` e suprimir o espaçamento superior, ou ainda `margin: 40px 0 0 0;` e suprimir o espaçamento inferior. E, uma série de outras combinações que ficam a título de exercícios para você.

2-) Se você deseja aplicar regras CSS em alguns elementos do documento e não em todos (por exemplo: alguns parágrafos na página seguirão uma regra `line-height` outros não) crie classes e aplique aos elementos.

:: As medidas CSS de comprimento ::

Introdução

Unidades de medida de comprimento CSS

As unidades de medida de comprimento CSS referem-se a medidas na horizontal ou na vertical (e em sentido mais amplo, em qualquer direção).

O formato para declarar o valor de uma unidade de medida CSS é um número com ou sem ponto decimal **imediatamente precedido** do sinal '+' (mais) ou do sinal '-' (menos), sendo o sinal '+' (mais) o valor "default" e **imediatamente seguido** por uma unidade identificadora (medida CSS válida - p.ex., px, em, deg, etc...). A unidade identificadora é opcional quando se declara um valor '0' (zero).

Algumas das propriedades CSS permitem que sejam declarados valores negativos para unidades de medida. A adoção de valores negativos podem complicar a formatação do elemento e devem ser usados com cautela. Se valores negativos não forem suportados pela aplicação de usuário, eles serão convertidos para o valor mais próximo suportado (e isso pode tornar-se desastroso para um layout).

Unidades de medida de comprimento CSS válidas

São dois os tipos de unidade de medida de comprimento CSS:

UNIDADE RELATIVA

- em
- ex
- px - pixel
- % - percentagem

as unidades relativas são referenciadas a outras unidades como veremos a seguir.

UNIDADE ABSOLUTA

- **pt - point :**1/72 in;
- **pc - pica :**12 points ou 1/6 in;
- **mm - milímetro :**1/10 cm;
- **cm - centímetro :**1/100 m;
- **in - polegada :**2,54 cm;

Unidade relativa- é aquela tomada em relação a uma outra medida. Folhas de Estilo em Cascata que usam unidades de comprimento relativas são mais apropriadas para ajustes de uso em diferentes tipos de mídia. (p. ex., de uma tela de monitor para uma impressora laser).

O valor é tomado em relação:

- **em**: ...ao tamanho da fonte ('font-size') herdada;
- **ex**: ...a altura da letra x (xis) da fonte herdada;
- **px**: ...ao dispositivo (midia) de exibição;
- **%**: ... a uma medida previamente definida.

Unidade absoluta - é aquela que não esta referenciada a qualquer outra unidade e nem é herdada. São unidades de medida de comprimento definidas nos sistemas de medidas pela física e em fim são os conhecidos "centímetros, polegadas etc...). São indicadas para serem usadas quando as mídias de exibição são perfeitamente conhecidas.

Abaixo exemplos ilustrativos do uso destas medidas de comprimento CSS:

```
div { margin: 1.5em; }
h4 { margin: 2ex; }
p { font-size: 14px; }
.classe { padding: 90%; }
hr { width: 14pt; }
h1 { margin: 1pc; }
h2 { font-size: 4mm; }
p.classe { padding: 0.3cm; }
h5.classe { padding: 0.5in; }
```

Nota: Relembro que uma regra CSS tem a seguinte sintaxe

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

Entendendo as unidades de medida CSS

Vamos a seguir definir e analisar cada uma das unidades de medida CSS e apresentar exemplos práticos.

A unidade de medida - pixel

A unidade de medida de comprimento pixel é relativa a resolução do dispositivo de exibição (p.ex: a tela de um monitor).

Sem entrar em maiores considerações teóricas a mais simplista definição de pixel que encontrei é esta:

Pixel é o menor elemento em um dispositivo de exibição, ao qual é possivel atribuir-se uma cor.

Considere um dispositivo de exibição construido com uma densidade de 90 dpi (dpi = dots per inch = pontos por polegada). Por definição, a **referência padrão para pixel** é igual a um ponto no citado dispositivo. Daí pode-se concluir que **1 pixel** naquele dispositivo de exibição é igual a 1/90 inch = 0,28 mm.

Para uma densidade de 300 dpi 1 pixel é igual a 1/300 inch = 0,085mm

Assim, pixel é uma medida relativa a resolução do dispositivo de exibição.

A unidade de medida - em

A unidade de medida de comprimento **em** referencia-se ao tamanho da fonte (letra) do seletor onde for declarada. Quando **em** for declarada para a propriedade `font-size` referencia-se ao tamanho da fonte (letra) do elemento pai. Quando **em** for declarada para o elemento raiz do documento (p. ex: `<html>` em documentos html) referencia-se ao valor inicial (default) do tamanho de fonte (letra).

Os exemplos abaixo esclarecem as definições:

```
h1 { line-height: 1.2em }
```

line-height de `<h1>` será 20% maior do que o tamanho das letras de `<h1>`

```
h1 { font-size: 1.2em }
```

font-size de `<h1>` será 20% maior do que o tamanho das letras herdado por `<h1>` p.ex.: se h1 estiver contido numa div com font-size=10px então font-size de h1 = 12px

A unidade de medida - ex

A unidade de medida de comprimento **ex** é igual a altura da letra **x**(xis) minúscula).

A unidade de medida - percentagem, %

Valores em percentagem são relativos a um outro valor anterior declarado. Este valor anterior há que estar bem definido e em geral esta definição está em uma determinada propriedade do mesmo elemento, na propriedade do elemento "pai" (por exemplo: uma medida CSS de comprimento) ou mesmo no contexto geral da formatação (por exemplo: a largura do bloco de conteúdo).


```
p { font-size: 10px }
p { line-height: 120% }/*120% de'font-size'=12px*/
```

:: Definindo cores em uma regra CSS ::

Objetivo

Detalhar as diferentes maneiras de se escrever a sintaxe para os valores das cores em uma regra CSS

Valores válidos para cores em CSS

Observe as regras de estilo a seguir:

```
1-) div.um {background-color: #FF0000;}

 2-) div.dois {background-color: #F00;}

 3-) div.tres {background-color: rgb(255, 0, 0);}

 4-) div.quatro {background-color: rgb(100%, 0%, 0%);}

 5-) div.cinco {background-color: red;}

 6-) div.seis {background-color: ThreeDShadow;}
```

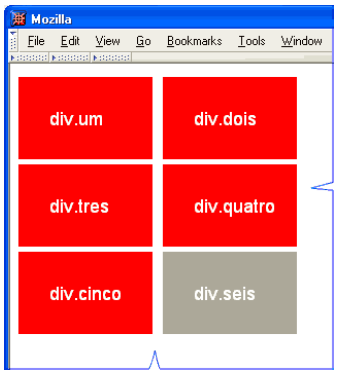
Como você já deve ter concluido apresentei 06 (seis) maneiras diferentes de definir uma cor de fundo para uma `DIV` .

E, se considerarmos que para as duas primeiras regras é válido usar letras minúsculas, existem 08 (oito) maneiras de se definir uma cor em uma regra CSS.

As maneiras mais usadas são as mostradas em 1 e em 2, ou seja, com uso do código hexadecimal de cores.

O efeito das regras no navegador

Observe agora no screenshot a seguir como estas seis DIV's serão renderizadas.



As cinco primeiras estão com a mesma cor de fundo, vermelha o que nos leva a concluir que as cinco primeiras regras mostradas são equivalentes, ou seja são cinco maneiras diferentes de definir um mesmo valor para uma cor.

#FF0000 = #F00 = rgb(255,0,0) = rgb(100%,0%,0%) = red

A sexta cor, `ThreeDShadow` depende do equipamento do usuário.

Vejamos cada uma delas detalhadamente.

Definir uma cor pelo seu código hexadecimal

Esta é a maneira mais conhecida de definir uma cor.

Convém ressaltar que em uma regra CSS é indiferente usar letras maiúsculas ou minúsculas na sintaxe hexadecimal de cores e também que é válido abreviar a notação para três dígitos.

Na notação abreviada cada um dos três dígitos é automaticamente dobrado conforme exemplos a seguir:

```
#FFF = #FFFFFF

#CF9 = #CCFF99

#cde = #ccddee

#49c = #4499cc
```

Não é do escopo deste tutorial detalhar o código hexadecimal, contudo ressalto que os dezesseis dígitos hexadecimais são:
0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F e somente eles são válidos para definir uma cor,

podendo em geral ser usada qualquer combinação deles. Assim: #FFDDHH não define uma cor, pois H não é válido.

Existem várias ferramentas online para determinar o código hexadecimal de uma cor. Uma das que eu costumo usar e indico para vocês é esta:
http://www.colorschemer.com/online.html

Definir uma cor pelo seu código rgb

rgb é abreviatura para:
r = red (vermelha)
g = green (verde)
b = blue (azul)

Assim o código `rgb(xxx, yyy, zzz)` indica uma cor obtida com a mistura de uma quantidade xxx de vermelho com yyy de verde e com zzz de azul.

Duas são as maneiras de se definir a quantidade de cada uma das três cores:

Uma faixa de númeração de 0 (zero) até 255
Em percentagem de 0% até 100%

Não é válido usar em uma definição número e percentagem.

Exemplos:

definições válidas
```
rgb(145, 230, 50) - rgb(20%, 0%, 70%)
```

definição não válida
```
rgb(255, 20%, 120)
```

Não é do escopo deste tutorial detalhar as misturas de cor rgb.

No link indicado no item anterior é possível determinar também, o código rgb de uma cor

Definir cor por palavra-chave

Você pode definir uma cor usando o nome da cor. Os nomes de cor válidos são os listados nas recomendações CSS do W3C.

As Recomendações para CSS 2.1 listam as seguintes 17 cores:

aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, orange, purple, red, silver, teal, white, e yellow

Assim, as regras a seguir são válidas para definir cor

```
p {color: aqua;}
div {background-color: teal;}
```

Definir cor baseado no sistema operacional do usuário

As recomendações para CSS2.1 preconizam a definição da cor baseado nas cores adotadas pelo sistema operacional do usuário.

Este tipo de unidade de definição de cor denominado System Colors está em desuso e não deverá constar das futuras Recomendações CSS3.

Trata-se de uma lista de nomes de cores válidas à semelhança da listagem de cores por palavra-chave e que se refere a áreas do sistema operacional.

As cores previstas são:

ActiveBorder, ActiveCaption, AppWorkspace, Background, ButtonFace, ButtonHighlight, ButtonShadow, ButtonText, CaptionText, GrayText, Highlight, HighlightText, InactiveBorder, InactiveCaption, InactiveCaptionText, InfoBackground, InfoText, Menu, MenuText, Scrollbar, ThreeDDarkShadow, ThreeDFace, ThreeDHighlight, ThreeDLightShadow, ThreeDShadow, Window, WindowFrame, WindowText

Embora os valores CSS sejam "case insensitives" recomenda-se usar a grafia com letras maiúsculas e minúsculas ao se escrever o nome das cores de sistema por razões de legibilidade.

Exemplos:

```
p {color: ThreeDLightShadow;}
div {background: ButtonShadow;}
```

Veja NESTE LINK as cores definidas por regras de estilo para o seu sistema operacional.

:: Abreviando declarações e valores em regras CSS ::

Relembrando a sintaxe e a terminologia de uma regra CSS

É comum encontrar-se em muitos artigos sobre CSS escritos em blogs e sites, em textos de posts em fóruns, em listas de discussão e até mesmo em revistas e jornais, diferentes referências e denominações equivocadas para os componentes de uma regra CSS

Seletores são chamados de elementos ou de tags, *propriedades* são chamadas de seletores ou de atributos, *valores* são chamados de atributos ou de propriedades, *declarações* são chamadas de regras ou funções CSS e por aí vai em uma diversificada combinação dos termos acima citados em uma salada que acaba por confundir iniciantes e as vezes até mesmo outros já com alguma experiência com folhas de estilo em cascata.

Com a finalidade de facilitar o entendimento desta matéria e esclarecer a confusão que vem se formando em torno do assunto, vamos rever a sintaxe e a terminologia de uma regra CSS para que quando eu escrever seletor, declaração, propriedade e valor, não haja dúvidas sobre a porção da regra CSS a que estou me referindo e você não fique se perguntando onde estão os "atributos CSS, as tags CSS, e outros tantos termos equivocados".

Vejamos o que diz as Recomendações do W3C para Folhas de Estilo, nível 1 na seção intitulada Conceitos Básicos

"O projeto, ou desenho do layout, das folhas de estilos é fácil. É preciso apenas conhecer um pouco da linguagem HTML e possuir noções básicas dos termos usados em publicação eletrônica. Como exemplo, para ajustar a cor das letras de um elemento '`H1`' para azul, basta fazer:

```
H1 {color: blue}
```

Este exemplo mostra o que é uma 'regra' simples em CSS. Uma regra é composta de duas partes principais: um selector ('`H1`') e uma declaração ('`color: blue`'). Por sua vez, esta declaração também possui duas partes: uma propriedade ('`color`') e seu valor ('`blue`'). Embora este exemplo especifique apenas uma das várias propriedades necessárias para montar um documento HTML, ela constitui por si só uma 'folha de estilo'. Quando for combinada com outras folhas de estilo ela determinará a apresentação final do documento (uma característica fundamental é que as folhas de estilo podem ser combinadas).

O seletor funciona como a ponte de ligação entre o documento HTML e a folha de estilo, e todos os elementos HTML podem funcionar como possíveis seletores. Os vários elementos HTML estão definidos na Recomendação HTML

etc.”

A Recomendação do W3C define claramente que uma regra CSS é composta de um seletor e uma declaração e que a declaração compreende uma propriedade e um valor.

Na regra CSS a seguir:


```
H1 {color: blue}
```

a terminologia correta é:

- `H1` - seletor;
- `{color: blue}` - declaração;
- `color` - propriedade;
- `blue` - valor.

e a sintaxe correta é:

- Escrever o seletor e a seguir a declaração;
- A declaração deve estar entre { } (chaves);
- Na declaração, separar a propriedade e o valor por : (dois pontos);
- É permitido usar espaços em branco em qualquer quantidade entre cada um dos caracteres da regra;
- É permitido grupar declarações em uma mesma regra e neste caso as declarações deverão ser separadas por ; (ponto-e-vírgula) podendo todas elas estar em uma mesma linha ou em linhas distintas. É facultativo o uso de ; (ponto-e-vírgula) após a última declaração na regra;
- É indiferente o uso de maiúsculas e minúsculas em uma regra CSS, contudo as *classes* e *ID's* devem seguir a mesma grafia constante da marcação.

Estes são os termos normatizados de uma regra CSS e os que usaremos. Portanto, não existe "atributo CSS" ou "tag CSS" ou "elemento CSS" ou "função CSS" ou tantos outros equivocadamente escritos.

Não é do escopo deste tutorial detalhar as boas práticas de escrita das regras em uma folha de estilos.
Sobre este assunto escrevi e recomendo a leitura do tutorial Dicas básicas para projetar folhas de estilos.

Abreviando valores de cores hexadecimais

O formato hexadecimal é uma das opções sintáticas mais usadas para se escrever o valor das cores em regras CSS. A regra a seguir define que os parágrafos serão na cor vermelha (#ff0000).

```
p {color: #ff0000;}
```

e que poderá ser abreviada para:

```
p {color: #f00;}
```

É válido abreviar cores hexadecimais para 3 dígitos. Valores escritos com 3 dígitos são interpretados como se cada um dos dígitos tivesse sido sido declarado duas vezes, isto é:

genericamente, #abc é equivalente #aabbcc

Exemplos:

```
#c30 = #cc3300
#999 = #999999
#ff0 = # ffff00
#d61 = #dd6611
```

É fácil concluir que a abreviação de cores hexadecimais somente é possível para as cores constituidas por 3 pares de dígitos hexadecimais.

Valores para os quatro lados de um elemento nível de bloco

Um elemento nível de bloco ou uma 'caixa' admite estilização em seus quatro lados para algumas propriedades como `border` e `padding` entre outras.
Por exemplo: você pode definir um `padding` superior, um `padding` à direita, um `padding` inferior e um `padding` à esquerda para uma `div`.

A sequência em que você escreve os valores para estilizar os quatro lados de uma 'caixa' é rígida e fixa em uma regra CSS e obedece a seguinte ordem:

em cima , lado direito, embaixo, lado esquerdo

Faça uma analogia com o relógio para não esquecer a sequência.
12 horas (superior), 3 horas (direita), 6 horas (inferior), 9 horas (esquerda).

A regra `div {padding: 2px 3px 8px 7px;}` define para a `div`:
um `padding` inferior igual a 8px:
um `padding` superior igual a 2px;
um `padding` à esquerda igual a 7px:
um `padding` à direita igual a 3px.

Além da mostrada acima é válido abreviar declarações que envolvem os quatros lados de uma 'caixa' de outras 3 maneiras diferentes como mostradas a seguir:

1. `div {padding: 10px;}` `padding` de 10px para os 4 lados;

2. `div {padding: 6px 8px;}` `padding` de 6px para os lados superior e inferior e de 8px para os lados direito e esquedo;
3. `div {padding: 2px 4px 9px;}` `padding` de 2px para o lado superior, de 4px para os lados direito e esquerdo e de 9px para o lado inferior.

Propriedades que admitem abreviação

Veremos ao longo deste tutorial, como abreviar as seguintes propriedades CSS:

1. margin;
2. padding;
3. background;
4. font;
5. list;
6. outline;
7. border.

Abreviando `margin`

As regras a seguir definem valores para as 4 margens para uma `div`:

```
div {
margin-top:10px;
margin-right:8px;
margin-bottom:0;
margin-left:5px;
}
```

E pode ser abreviada para:

```
div {
margin:10px 8px 0 5px;
}
```

Abreviando `padding`

As regras a seguir definem valores para os 4 paddings de um parágrafo:

```
p {
padding-bottom:6px;
padding-top:12px;
padding-left:1px;
padding-right:2px;
}
```

E pode ser abreviada para:

```
div {
padding:12px 2px 6px 1px;
}
```

Abreviando `background`

As regras a seguir definem valores para propriedades background de uma `div`:

```
div {
background-color:#ffffcc;
background-image:url(fundo.gif);
background-repeat:no-repeat;
background-attachment:fixed;
background-position:20px 10px;
}
```


E pode ser abreviada para:

```
div {
background:#ffc url(fundo.gif) no-repeat fixed 20px 10px;
}
```

Abreviando `font`

As regras a seguir definem valores para propriedades de font em um documento:

```
body {
font-style:italic;
font-variant:small-caps;
font-weight:bold;
font-size:11px;
line-height:15px;
font-family:Arial, Helvetica, Sans-serif;
}
```

E pode ser abreviada para:

```
body {
font:italic small-caps bold 11px/15px Arial, Helvetica, Sans-serif;
}
```

Nota: Para abreviar a propriedade `font` é obrigatório definir no mínimo os valores de tamanho e família da `font`. Os demais valores são facultativos. A ordem de declaração dos valores é importante e deve ser assim:

1. começar com `font-style, font-variant e font-weight` sedo que estes três valores são facultativos e podem ser escritos em qualquer ordem;
2. a seguir declarar obrigatoriamente `font-size` e opcionalmente `line-height` (`font-size/line-height`);
3. finalmente declarar obrigatoriamente `font-family`.

Abreviando `list`

As regras a seguir definem valores para propriedades de listas:

```
ul {
list-style-type:square;
list-style-position:inside;
list-style-image:url(image.gif);
}
```

E pode ser abreviada para:

```
ul {list-style:square inside url(image.gif);}
```

A propriedade: `list-style-type` pode ser abreviada para **`list-style`**.
Por exemplo: `list-style-type:none` pode ser abreviada para **`list-style:none;`**

Abreviando `outline`

A propriedade `outline` é pouco conhecida e empregada. Serve para colocar uma margem ao redor de um elemento, com a finalidade de destacá-lo no contexto. Difere da propriedade `border` por não interferir com as dimensões do box model, isto é, não ocupa espaço no box do elemento e em consequência não afeta o posicionamento do box e nem dos boxes adjacentes.

As regras a seguir definem a marcação de um 'destaque' em linha vermelha sólida de 1px ao redor do elemento `h2`:

```
h2 {
outline-color:#f00;
outline-style:solid;
outline-width:1px;
}
```

E pode ser abreviada para:

```
h2 {
outline:#f00 solid 1px;
}
```

Exemplo: Para este parágrafo eu defini um destaque (`outline`) de 5px em linha tracejada na cor azul que será visualizado nos Mozillas e no Ópera, mas não no Internet Explorer que não suporta `outline`.

Abreviando `border`

As regras a seguir definem valores para propriedades de borda:

```
div {
border-width:1px;
border-style:solid;
border-color:#f00;
}
```

E pode ser abreviada para:

```
div {border:1px solid #f00;}
```

As regras a seguir definem valores para espessuras de borda:

```
p {
border-top-width:2px;
border-right-width:1px;
border-bottom-width:3px;
border-left-width:5px;
}
```


E pode ser abreviada para:

```
p {border-width:2px 1px 3px 5px;}
```

As regras a seguir definem valores para cores de borda:

```
h1 {
border-top-color:#f00;
border-right-color:#ccc;
border-bottom-color:#00f;
border-left-color:#999;
}
```

E pode ser abreviada para:

```
p {border-color:#f00 #ccc #00f #999;}
```

As regras a seguir definem valores para estilos de borda:

```
p {
border-top-style:solid;
border-right-style:ridge;
border-bottom-style:double;
border-left-style:dotted;
}
```

E pode ser abreviada para:

```
p {border-style:solid ridge double dotted;}
```

:: CSS Links ::

Introdução

CSS possibilita definir uma variedade infinita de layouts e efeitos para um link ou um conjunto de links.

O limite é a sua imaginação!

Um link visitado, não visitado, ativo ou no estado hover (quando você passa o mouse sobre ele) pode assumir aspectos (cores, fundos, etc...) diferentes através de CSS.

Abordarei neste tutorial as técnicas básicas de manipular os quatro seletores de link.

Pseudo classes

Os efeitos em links são possíveis através de declarações de regras de estilo para as pseudo classes do elemento `<a>` do HTML.

As pseudo classes são usadas em CSS, para adicionar efeitos diferentes a alguns seletores, ou a uma instância de alguns seletores.

Sintaxe

A sintaxe das pseudo classes:

```
seletor:pseudo classe {propriedade: valor}
```

As classes em CSS podem também ser usadas com pseudo classes.

Esta regra permite que você defina diferentes efeitos para links localizados em diferentes lugares em uma mesma página. No último item deste tutorial "Diferentes estilos de links em uma mesma página web" veremos este efeito.

```
seletor.class:pseudo-class {propriedade: valor}
```

Os "seletores:pseudo classe" de links

São quatro as pseudo classes dos links:

1. **a:link**.....define o estilo do link no estado inicial;
2. **a:visited**...define o estilo do link visitado;
3. **a:hover**.....define o estilo do link quando passa-se o mouse sobre ele;
4. **a:active**....define o estilo do link ativo (o que foi "clicado").

Vamos a seguir analisar cada um deles detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada seletor apresento as regras CSS definidas dentro de uma folha de estilos incorporada, e a seguir os efeitos em um link, onde se aplicam as regras.

Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Esta é a melhor e mais rápida maneira de você aprender CSS. Bons estudos! E faça ótimo proveito dos tutoriais.

Grande flexibilidade

É perfeitamente possível com CSS definir-se um estilo diferente para cada um dos quatro seletores, de forma semelhante como é definido para qualquer texto HTML.

Prioridade nas declarações para links

É importante a ordem de definição das regras para os estados dos links. Lembre-se de que pelo "efeito cascata", as regras mais próximas do elemento prevalecem sobre as mais distantes. Assim, por exemplo: se voce define `a: hover` ANTES de `a: visited`, esta prevalecerá sobre `a: hover` e em consequência: O link visitado pela primeira vez assumirá a regra definida em `a: visited` e a partir de então `a: hover` não mais funcionará naquele link pois `a: visited` prevalecerá sobre `a: hover`.

Qual é a ordem normal é lógica das declarações? É simples concluir ! Senão vejamos:

```
1.) a: link é o estado inicial dos links;
      a:link deverá ser a primeira declaração;
2.) a: active deverá acontecer mesmo
      em links já visitados;
     a: active deverá ser declarado depois
     de a:visited;
1.) a: hover não precisa funcionar em a:active;
     a:active pode ser declarado depois de a:hover.


Em consequência a ordem das declarações deve ser:
a:link
a:visited
a:hover
a:active
```

Exemplos de efeitos em links

A seguir exemplos dos efeitos mais comuns e simples aplicando CSS aos seletores de links

1-) Removendo o sublinhado do link

Típicamente, por default os links são sublinhados e na cor azul.

Um efeito muito comum em páginas web é o de se retirar o sublinhado do link normal, mudar a cor e fazer "aparecer" o sublinhado, quando o mouse é passado sobre ele.

Esse efeito obtém-se facilmente com CSS. Abaixo a regra para esse simples efeito:

```
<style type="text/css">
<!--
a:link {text-decoration: none}
a:visited {text-decoration: none}
a:hover {text-decoration: underline;
color: #FF0000;
}
a:active {text-decoration: none}
-->
</style>
```


PASSE O MOUSE AQUI

2-) Adicionando sublinhado e sobrelinhado

Este efeito semelhante ao anterior, consiste em um sobrelinhado adicional no link hover.

Abaixo a regra para esse efeito:

```
<style type="text/css">
<!--
a:link, a:visited, a:active {
text-decoration: underline;
}
a:hover {text-decoration: underline overline;
color:#FF0000;
}
-->
</style>
```

PASSE O MOUSE AQUI

3-) Acrescentando um fundo

Este efeito simulando um "rollover" simples, consiste em acrescentar uma cor de fundo no link hover.

Abaixo a regra para esse efeito:

```
<style type="text/css">
<!--
a:link, a:visited, a:active {
text-decoration: underline;
}
a:hover {text-decoration: underline;
background:#ffc;
color: #FF0000;
}
-->
```

```
</style>
```

PASSE O MOUSE AQUI

4-) Link com um fundo de uma cor que muda no link hover

Este efeito também simula um "rollover" simples, consiste em mudar a cor de fundo no link hover.

Abaixo a regra para esse efeito:


```
<style type="text/css">
<!--
a:link, a:visited, a:active {
text-decoration: underline;
 background:#FFFFFF;
}
a:hover {text-decoration: underline;
 color:#000;
background:#FFFFCC;
}
-->
</style>
```

PASSE O MOUSE AQUI

5-) Link que muda o tamanho da letra no link hover

Neste efeito há uma mudança no tamanho da letra (em geral para maior) no link hover.

Abaixo a regra para esse efeito:

```
<style type="text/css">
<!--
a:link, a:visited, a:active {
text-decoration: none;
}
a:hover {text-decoration: underline;
color:#000000;
font-size:150%; }
-->
</style>
```

PASSE O MOUSE AQUI

Diferentes estilos de links em uma mesma página web

É possível definir diferentes estilos para os 4 seletores de links para serem usados em uma mesma página.

Isso consegue-se utilizando os seletores de contexto que consiste em adicionar uma **classe** aos seletores.

Assim cada conjunto dos 4 seletores de uma classe assumem o comportamento daquela classe e você pode definir quantas classes quiser.

Por exemplo:

```
<html>
 <head>
 <style type="text/css">
 a.classe1:link {text-decoration: none}
 a.classe1:visited {text-decoration: none}
 a.classe1:hover {
 text-decoration: underline;
 color: #FF0000;
 }
 a.classe1:active {text-decoration: none}

 a.classe2:link {
 text-decoration: underline overline
 }
 a.classe2:visited {
 text-decoration: underline overline
 }
 a.classe2:hover {text-decoration: underline;
 color: #00FF00;
 }
 a.classe2:active {
 text-decoration: underline overline
 }
 </style>
 </head>
 <body>
 <a href="#" class="classe1">
 ESTE É O LINK DA classe1
 </a>

 <a href="#" class="classe2">
 ESTE É O LINK DA classe2
 </a>
```

** Digite e veja no navegador o resultado!

:: Efeitos em links com codificação de caracteres ::

Introdução

Mostrar como tirar proveito do uso de folhas de estilo aliadas a codificação de caracteres para obtenção de efeitos em links que tradicionalmente são obtidos com pequenas imagens.

Ao final da leitura deste tutorial você estará capacitado a personalizar links que requerem pequenos efeitos nos seus documentos Web, com uso das CSS e da codificação de caracteres.

Introdução a codificação de caracteres

Se você conhece a codificação de caracteres via ASCII ou através de texto, poderá pular a leitura deste subtítulo.

Em uma linguagem bem simples sem entrar em considerações técnicas detalhadas, pois este não é o foco deste tutorial explico resumidamente a seguir como "funciona" a codificação de caracteres em páginas web.

Vamos acompanhar através de um exemplo:

Este HTML:`<p>A codificação de caracteres </p>`

"diz" ao browser para colocar na tela do usuário o texto:

A codificação de caracteres

Note a existência de um **ç** (c cedilha) e um **ã** (a com til). Estes são dois caracteres, não existentes em outras línguas (inglês por exemplo), assim como também é um caracter especial o **ä** (a com trema-dois pontos sobre a letra) que não existe na língua portuguesa.

Note também que **ç** e **ã** estão aí no seu teclado, mas procure o **ä**.

Eu também não tenho **a com trema** no meu teclado, mas escrevi ele no código HTML deste tutorial.

Aqui mais alguns exemplos de caracteres que não tem no teclado em português: **£ ¤ ¥ § µ ¶ ♥ Ø © ® ♣ † ↑**

Você consegue visualizar e eu consegui escrever no HTML estas pequenas figuras e símbolos graças a codificação de caracteres que nada mais é do que como o próprio nome sugere, escrever o caracter por um código :-)

Mas como é este código?

Básicamente está disponível duas maneiras de se inserir o código para caracteres no documento HTML. Declaração ASCII e declaração por texto.

Sintaxe geral da declaração ASCII: `&#número;`

Sintaxe geral da declaração por texto: `&texto;`

Por exemplo:

no HTML ==> `&#163;` na tela ==> £

no HTML ==> `&clubs;` na tela ==>♣

Continue a leitura do tutorial até o fim e depois vá a esta página http://www.asciitable.com/, conhecer algumas tabelas de codifição de caracteres: depois que entrar nesta página, examine primeiro o link "HTML Code"

Agora que já sabemos como obter na tela pequenas figuras, vamos colocá-las em nossos links personalizados com a vantagem de apresentar uma figura usando o HTML de texto.

Nada de `<img scr="...>` para definir a figura!

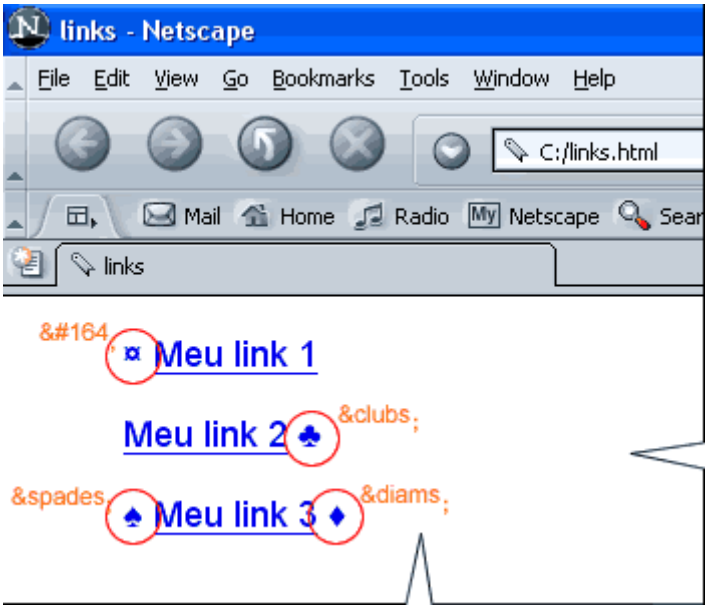
Posicionando o caracter no link

Você poderá posicionar o caracter a esquerda, a direita ou nos dois lados do link.

Observe o código a seguir:

```
<p><a href="caminho.html">&#164; Meu link 1</a></p>
<p><a href="caminho.html">Meu link 2 &clubs;</a></p>
<p><a href="caminho.html">&spades; Meu link 3 &diams;</a></p>
```

E abaixo o resultado no browser:



Revisão dos conceitos

Vimos até aqui que através da codificação de caracteres, podemos inserir pequenas imagens e símbolos nos nossos documentos HTML.

A inserção é feita no fluxo normal do texto como se fora uma letra e desta forma o caracter poderá ser posicionado no início, no final, em ambos e até mesmo dentro de um texto.

Existem disponíveis na Internet as tabelas que fornecem a codificação dos caracteres para você escolher aquele que irá usar.

Vamos nos aproveitar desta facilidade para personalizar links usando as CSS, mas é óbvio que você poderá se valer dos caracteres para outras finalidades decorativas ou não em sua página web.

Efeitos CSS nos caracteres

Vou tomar como base para este tutorial a estilização para um link com um caracter posicionado a esquerda.

As técnicas para posicionamentos a direita ou em ambos os lados é a mesma.

E, a proposta de estilização é a de fazer com que o caracter fique invisível a esquerda do link e torne-se visível quando passa-se o mouse sobre o link.

1º. Passo:

Posicionar o caracter no link. Conforme vimos acima:

```
<p><a href="caminho.html">&#186; MEU LINK </a></p>
```

Resulta no browser:

º MEU LINK

2º. Passo:

Sendo a proposta de estilização a de "esconder e fazer aparecer" o caracter, vamos separar o caracter do texto do link.

A tag HTML própria para esta separação é a tag `<span>`.

Observe agora o código:

```
<p><a href="caminho.html"><span>&#186;</span><span>MEU
LINK</span></a></p>
```

3º. Passo:

Em uma página web é comum a existência links com comportamentos diferentes, daí vamos atribuir uma classe para o parágrafo que contém o link.

E, precisaremos de mais uma classe para cada um dos `<span>` que criamos.

Estas três classes possibilitarão estilizar nosso link.

Criei as classes: `.geral .caracter .linque`

E o novo código com as classes:


```
<p class="geral"><a href="caminho.html">
<span class="caracter" >&#186;</span><
span class="linque">MEU LINK</span></a></p>
```

4º. Passo:

Escrever as regras CSS para "esconder" o caracter quando o link estiver no estado UP (em repouso) e fazê-lo "aparecer" quando o link estiver no estado OVER (com o mouse em cima).

O seletor de contexto para o caracter é: `.geral a span.caracter`

Não sabe o que é seletor contexto? Leia este tutorial.

A propriedade CSS para isto é `visibility` e a regra para esconder é:

```
.geral a span.caracter {visibility:hidden;}
```

e, para aparecer é:

```
.geral a:hover span.caracter {visibility:visible;}
```

O efeito destas duas regras é mostrado abaixo:

Link no estado UP ==> MEU LINK

Link no estado OVER ==> º MEU LINK

Você pode observar que já conseguimos o efeito rollover na bolinha ao lado esquerdo do link (o caracter `&#186;`), mas podemos incrementar este efeito retirando o sublinhado da bolinha!

Retirar o sublinhado do caracter :

A propriedade CSS que retira o sublinhado é: `text-decoration`

Você seria capaz de escrever a regra CSS para retirar o sublinhado somente do caracter ?

Se você respondeu:

```
.geral a:hover span.caracter {text-decoration:none;}
```

raciocinou corretamente, mas escapou-lhe um detalhe e a regra não fará o efeito que você espera.

O sublinhado em links é uma característica default do browser e válida para todo o texto do link. Você não conseguirá retirar parte do sublinhado do texto do link (o caracter), usando uma regra para `<span>`.

Temos que usar um artifício que consiste em **retirar** o sublinhado de todo o link para depois **colocar** o sublinhado só no texto e não no caracter.

Assim:

```
.geral a {text-decoration:none;}
/* retira o sublinhado de todo o link texto mais caracter */
.geral a span.linque {text-decoration:underline;}
/* coloca o sublinhado no texto do link */
```

Bug no Internet Explorer :

As regras CSS como descritas acima fazem o efeito proposto no tutorial em browsers Netscape, Opera, Firefox, Mozilla, mas o Internet Explorer apresenta um bug que consiste em não mostrar o caracter no estado OVER.

Uma das maneiras contornar este bug é declarar uma regra adicional:

```
.geral a:hover {margin: 0;} /* hack para IE */
```

Considerações e dicas finais:

O entendimento da técnica básica mostrada neste tutorial possibilitará a você criar uma variedade cosiderável de efeitos em links. A manipulação dos caracteres quanto ao seu aspecto e posicionamento, conjugada com efeitos adicionais como cores, fundos, e sublinhados somada a sua criatividade, proporcionará mais uma ferramenta CSS para a personalização de seus links.

Alguns exemplos:

Criei uma página com alguns efeitos usando os conhecimentos deste tutorial.

Fique a vontade para copiar e colar o código fonte da página no seu HD e estudar offline.

*** Os temas referentes a menus ficam no site do maujor e outros materiais também dever ser pesquisados antes do minu-curso.

Construir um site é a melhor maneira de iniciar o estude do XHTML/CSS pois, isso lhe dará a oportunidade de sentir as reais necessidades de domínio da linguagem enquanto estiver a desenvolver uma página.

- Bons Estudos! Qualquer duvida: www.maujor.com