



TÉCNICAS DE

# ACESSIBILIDADE

CRIANDO UMA **web** PARA TODOS

Jalves Mendonça Nicácio



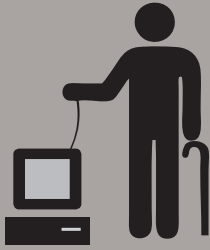
Programa  
SENAI de  
Ações Inclusivas



**SENAI**



edUFAL



# **Técnicas de Acessibilidade**

**Criando uma web para todos**

**Jalves Mendonça Nicácio**



## UNIVERSIDADE FEDERAL DE ALAGOAS

### Reitora

Ana Dayse Rezende Dorea

### Vice-reitor

Eurico de Barros Lôbo Filho

### Diretora da Edufal

Sheila Diab Maluf

### Conselho Editorial

Sheila Diab Maluf (Presidente)  
Cícero Péricles de Oliveira Carvalho  
Elton Casado Fireman  
Roberto Sarmento Lima  
Iracilda Maria de Moura Lima  
Lindemberg Medeiros de Araújo  
Leonardo Bittencourt  
Eurico Eduardo Pinto de Lemos  
Antonio de Pádua Cavalcante  
Cristiane Cyrino Estevão Oliveira

### Capa:

Carlos Eduardo do Nascimento Araújo

### Revisão:

Marcus de Melo Braga

### Diagramação:

Jalves Mendonça Nicácio

### Supervisão Gráfica:

Márcio Roberto Vieira de Melo

Catálogo na fonte  
Universidade Federal de Alagoas  
Biblioteca Central - Divisão de Tratamento Técnico  
Bibliotecária Responsável: Helena Cristina Pimentel do Vale

---

N582t Nicácio, Jalves Mendonça.

Técnicas de acessibilidade : criando uma web para todos / Jalves Mendonça  
Nicácio. - Maceió : EDUFAL, 2010.  
100 p. : il.

Inclui bibliografia.

O projeto da 1ª edição (2007) foi financiado pelo Banco do Nordeste do Brasil.

1. Internet (Redes de computadores). 2. World Wide Web (Sistema de recuperação da informação). 3. Interação homem-máquina. 4. Informática.

I. Título.

CDU: 004.5

---

### Direitos desta edição reservados à

Edufal - Editora da Universidade Federal de Alagoas  
Campus A. C. Simões, BR 104, Km 97,6 - Fone/Fax: (82) 3214.1111  
Tabuleiro do Martins - CEP: 57.072-970 - Maceió - Alagoas  
E-mail: edufal@edufal.ufal.br - Site: www.edufal.ufal.br

### Editora afiliada:



Associação Brasileira  
das Editoras Universitárias

## **A Web acessível como um caminho para auxiliar na Educação Inclusiva**

Desde 1999 o SENAI Nacional desenvolve com sucesso, através da Unidade de Educação Profissional - UNIEP, Programa SENAI de Ações Inclusivas - PSAI, que estabeleceu o amplo atendimento a pessoas com deficiência física, intelectual, auditiva, visual ou múltipla, bem como, superdotados (altas habilidades) nas unidades operacionais dos seus 27 Departamentos Regionais. O PSAI, além de atender ao contingente de pessoas com necessidades especiais, promove o acesso aos cursos do SENAI para mulheres, negros e índios e a re-qualificação profissional de pessoas idosas.

O grande foco do PSAI é oportunizar a educação profissional para todos os cidadãos que por algum motivo sócio-cultural, econômico ou por preconceito, são tolhidos de exercer este direito constitucional.

No que se refere à inclusão digital de pessoas com necessidades especiais, o Brasil tem adotado uma política favorável a ações que procuram promover a educação inclusiva, amplamente defendida pela legislação brasileira. Desta forma, o SENAI Nacional, através do PSAI, vem capacitando seus docentes para utilização dos “softwares leitores de tela” mais usados no Brasil. Tais softwares são utilizados pelas pessoas cegas em sua interação com o computador e a informática. Assim, os docentes do SENAI foram habilitados para receber, nos cursos de informática, alunos deficientes visuais, tornando possível para essas pessoas o acesso a mídias digitais como a Web.

Em Alagoas, o SENAI vem desenvolvendo com sucesso várias ações dentro do PSAI desde 2002. Além de oferecer cursos de capacitação profissional para pessoas com deficiência física ou sensorial, o SENAI/AL tem ampliado suas ações no sentido de aumentar a participação dessas pessoas no mercado de trabalho. Um dos maiores exemplos desta atuação é a publicação do Guia Alagoas Inclusiva, uma cartilha informativa com perguntas e respostas para orientação de empregadores na contratação de pessoas com deficiência.

No âmbito da informação digital, o SENAI/AL, numa promoção da Rede de Inclusão Social da Pessoa com Deficiência em Alagoas, criou o site Alagoas Inclusiva, que traz informações e serviços para empresas e comunidade. O Site conta com um banco de profissionais composto por pessoas com

deficiência que foram qualificadas e certificadas pelo SENAI e outras instituições de educação que compõem a Rede de Inclusão Social.

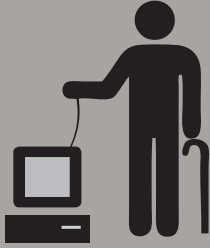
No entanto, mesmo diante de todas as iniciativas já mencionadas, fazem-se necessários a difusão e o ensino de técnicas que viabilizem a construção e publicação na Web de páginas com conteúdo acessível, preparando desta forma os futuros construtores de páginas da Internet na perspectiva da inclusão.

A edição especial desse livro, patrocinada pelo SENAI em parceria com a EDUFAL e o Banco do Nordeste do Brasil, com a devida anuência do autor e também colaborador do SENAI em Alagoas, Jalves Mendonça Nicácio, é mais uma etapa dessa estratégia educativa. É imprescindível pensar numa Web que possibilite o acesso a qualquer pessoa, independente de suas limitações. Mais ainda: contribuir para que este ideal se realize deve ser considerado uma obrigação de todos que queiram compartilhar informações através de uma página na Internet.

Entendemos que com a publicação desta edição especial, o SENAI ratifica seu compromisso com a construção da cidadania, por intermédio da educação profissional, e oportuniza a qualificação não apenas dos seus educandos, mas também do seu quadro docente, que faz a diferença no processo de inclusão de pessoas com necessidades especiais.

**Marben Montenegro Loureiro**

*Diretor Regional do SENAI/AL*



# Sumário

**Apresentação** **x**

## Introdução

**Do que trata este livro?** **xii**

**1. O problema da inacessibilidade à páginas de Web** ..... **xiii**

Caso 1: .....xiii

Caso 2: .....xiii

Caso 3: .....xiv

**2. Quem deveria ler este livro?**..... **xv**

## Capítulo 1

**O mínimo que você deve saber...** **18**

**1. Afinal, o que é “Web Acessível”?**..... **19**

O que quer dizer Web?..... 19

O que quer dizer acessível? ..... 21

O que quer dizer Web acessível? ..... 22

**2. Sobre a Web e seus componentes.** ..... **23**

---

## Capítulo 2

### **Diretrizes para construção de conteúdo acessível 28**

1. **Diretrizes W3C ..... 29**
2. **Diretrizes Brasileiras: E-MAG ..... 31**
3. **Diretrizes WCAG Samurai..... 32**

## Capítulo 3

### **Regras simples, ótimos resultados! 34**

1. **Identificando a linguagem utilizada ..... 37**
2. **Trabalhando com texto alternativo ..... 38**
3. **Trabalhando com CSS ..... 40**  
Um exemplo prático:..... 42
4. **Trabalhando com medidas relativas ..... 49**
5. **Pensando na estrutura da página ..... 52**  
Descrição de página..... 54
6. **Tabelas ..... 56**  
Como um leitor de tela deve ler uma tabela:..... 57
7. **Formulários ..... 61**
8. **Acrônimos e Abreviações..... 65**

## Capítulo 4

### **Técnicas para aplicação de acessibilidade 68**

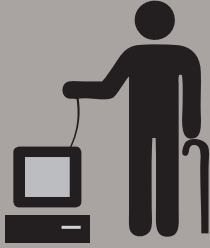
1. Teclas de Atalho..... 70
2. Skip links ..... 72
3. Controle do tamanho da letra ..... 79
4. Problemas de acessibilidade com CAPTCHA..... 85
5. Breadcrumbs Trail..... 89
6. Menu de Acessibilidade ..... 91

## Capítulo 5

### **Cinto de Utilidades! 94**

1. Ferramentas de avaliação/validação de acessibilidade . 95
2. Softwares para avaliação de acessibilidade ..... 95
3. Ferramentas validadoras de HTML/CSS ..... 96
  - Validadores de sintaxe HTML/XHTML ..... 96
  - Validadores de CSS..... 96
4. Softwares leitores de tela ..... 97
5. Complementos de acessibilidade para Firefox ..... 97
6. Ferramentas que úteis para teste e desenvolvimento .... 98
7. Ítens de leitura Obrigatória..... 98
8. Ítens de leitura recomendada ..... 98





# Apresentação

Alguns sonhos que conseguimos realizar em nossas vidas são, muitas vezes, vistos como meras fantasias nossas, diante de todos os obstáculos que teremos que enfrentar para torná-los algo real, concreto. O presente livro que me cabe agora apresentar é um bom exemplo disso. Nasceu de uma proposta de trabalho de conclusão de curso que gerou uma iniciativa de um projeto de acessibilidade para o site da editora da Universidade Federal de Alagoas e que, no seu decorrer, criou as condições necessárias à realização desse projeto.

Acessibilidade é um tema ainda pouco explorado e de bibliografia escassa, apesar da sua inegável relevância. Está intimamente ligado à questão da inclusão digital e da atenção às pessoas, ao ser humano de uma forma geral. Sob esse ponto de vista, o presente trabalho reveste-se de uma natureza altruísta em querer buscar difundir o uso de técnicas e ferramentas que promovam a acessibilidade para esse instrumento fantástico e entusiástico que hoje é a Web.

Trata-se de um texto que, apesar de abordar questões técnicas, destinadas a um público especializado, cumpre sua missão com uma linguagem simples e acessível, como deve ser o tom de uma obra que pretende abordar tal tema. Nele, o leitor é conduzido desde aos conceitos básicos da acessibilidade, sua motivação e importância, até às técnicas e ferramentas avançadas para a sua utilização plena por especialistas, ou até mesmo por pessoas que não tenham tanta experiência em desenvolvimento de aplicações para a Web, mas que pretendam publicar algo, desde um simples blog até um site mais complexo.

Com a disponibilização de ferramentas extremamente fáceis para publicação de conteúdo na Web, pessoas sem muita experiência de programação de computadores podem marcar presença, cada vez mais, no contexto

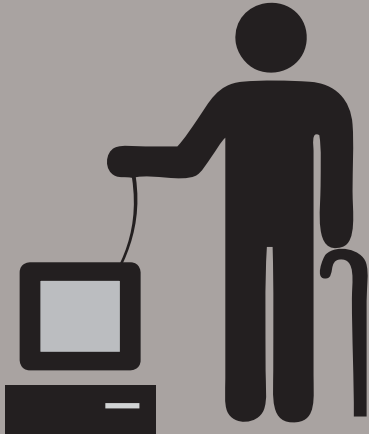
global que a Internet criou, divulgando seus produtos, serviços ou até mesmo pensamentos, nessa grande rede. Para todo esse público, a presente obra pode contribuir com valiosas informações, com o intuito de possibilitar a criação de espaços na Web que possam ser acessados com a maior abrangência possível.

Ampliar a acessibilidade das informações disponíveis na rede implica o aumento de visitas e, logicamente, de possíveis usuários ou clientes adicionais, gerando resultados e contribuindo para uma maior inclusão social. A adesão ao tema aqui abordado é um dos raros exemplos em que todos saem ganhando. Mas, apesar de ser uma ferramenta útil para todos, pouco se faz em termos de acessibilidade no cenário global. Há outro sonho incubado nessa obra: de que um dia a Web seja amplamente acessível, incluindo jovens e idosos, pessoas carentes de atenção e aquelas mais autoconfiantes. O livro é a contribuição do seu autor, jovem idealista e visionário, para a transformação de uma realidade, para mudar o mundo. É a sua decisão pessoal de interferir nessa realidade, optando por tentar fazer alguma diferença nesse universo midiático, em que a Web se transformou.

Por todas essas questões, esta obra tem um mérito louvável. É o primeiro livro no cenário nacional sobre esse assunto e, como toda iniciativa pioneira, abre espaço para novas pesquisas e trabalhos no tema abordado. Pode ser adotado como livro texto para cursos de desenvolvimento de aplicações para a Web e, até mesmo, para cursos de graduação em Sistemas de Informação e Ciências da Computação.

Espera-se que o leitor não se detenha apenas a conhecer os recursos de acessibilidade aqui tratados e que se una a um grupo cada vez maior de profissionais que trabalham para melhorar as condições de acesso ao conteúdo da Web. Se cada um de nós der um simples passo nessa caminhada, a sinergia gerada pelos pequenos esforços individuais poderá contribuir para um considerável aumento do número de pessoas que podem tirar proveito de todas as tecnologias inclusivas que já existem e que estão à nossa disposição, para construir um mundo cada vez mais acessível a todos nós.

**MARCUS DE MELO BRAGA**



## INTRODUÇÃO

### **Do que trata este livro?**

Internet, e-mail, site, home page, servidor, navegador, links por toda parte! Quem nunca ficou confuso com toda esta tecnologia? Mesmo quem já nasceu na geração “on-line”, algumas vezes pode se sentir perdido, até porque esta é uma tecnologia em constante desenvolvimento desde sua criação.

Na verdade, desde a primeira vez que o mundo ouviu falar de Web, nada tem sido como antes. A Internet, e a Web por sua vez, vem a cada ano provocando mudanças radicais na maneira como os homens se relacionam, trocam informações, resolvem problemas, estudam ou simplesmente se divertem. Muitas vezes, a Web substituiu antigos serviços conhecidos pelo homem, como os correios e o telefone. Desta forma, não há como negar a importância da Web nos dias de hoje.

## **1. O problema da inacessibilidade à páginas de Web**

Este é um problema que não passa despercebido para um grupo de pessoas que tentam se beneficiar das vantagens dos serviços oferecidos pela Web. Por diversas razões, muitas vezes um determinado serviço ou página da Web poderá estar inacessível a uma pessoa ou grupos de pessoas.

Por exemplo, imagine os seguintes casos:

### **Caso 1:**

João quer fazer umas compras num site de vendas on-line. Nada complicado: ele quer adquirir uns CD's e alguns livros. No site, ele encontra instruções explicando que preços com desconto estão em vermelho. Tudo estaria muito bem, a não ser por um detalhe: João é daltônico e não consegue distinguir a cor vermelha. A situação se complica quando João se depara com um formulário onde os campos obrigatórios também são destacados dos demais pela cor vermelha.

### **Caso 2:**

Toda vez que Sandra precisa ir sozinha ao supermercado ela fica aflita. Sandra tem síndrome de Down e tem dificuldade com leitura, cálculos matemáticos e conceitos abstratos. Na hora de escolher um produto, ela fica confusa com as diversas opções e acaba perdendo o controle de quanto ela está gastando.

Uma vez um amigo dela a apresentou a um site onde ela poderia fazer as mesmas compras que ela fazia no supermercado pela internet. Ela tentou usar o site algumas vezes e logo descobriu uma série de dificuldades: o site nunca deixava claro em que seção ela estava, nem como encontrar um determinado produto. Em algumas seções o layout da página muda completamente e a Sandra fica na dúvida se ainda está no mesmo site ou não.

### **Caso 3:**

A primeira vez que o Henrique usou um programa leitor de tela foi uma comoção só! Pela primeira vez em muito tempo ele sentia que poderia fazer alguma coisa sem ajuda de ninguém. Henrique é cego e acabou de ser apresentado a um programa capaz de ler todo texto que se encontra na tela.

Imaginem quanta coisa agora ele será capaz de fazer! Através da tecla TAB do teclado ele pode navegar de link em link e encontrar o que deseja. Com este programa, agora ele poderá acessar os serviços da Web, ele pode pagar suas contas, comprar produtos, fazer pesquisas, mandar e-mails, etc.

Mas sua fascinação não durou muito tempo. Logo Marcos descobriu que nem todos os sites permitiam que o leitor de telas acessasse adequadamente seu conteúdo.

Todos estes exemplos servem para evidenciar uma

realidade vivida por milhões de pessoas em todo mundo. Quando privamos estas pessoas do acesso aos serviços e informações que circulam na Web, estamos criando um sério problema de exclusão social e digital. Encare da seguinte forma: Para pessoas sem deficiência, sistemas da Web facilitam a vida, mas para quem não pode ver ou ouvir ou andar, estes mesmos sistemas são mais do que facilitadores. Para essas pessoas, ter acesso a esses sistemas, muitas vezes significa tornar possível a realização de tarefas que, de outra forma, exigiriam um esforço imenso ou até mesmo seriam impossíveis de serem realizadas.

Então, o que podemos fazer para que o conteúdo das páginas e sistemas da Web seja acessível a pessoas portadoras de necessidades especiais? É exatamente disso que trata este livro. Abordaremos aqui os principais conceitos que precisamos saber para tornar ou construir sites acessíveis.

## **2. Quem deveria ler este livro?**

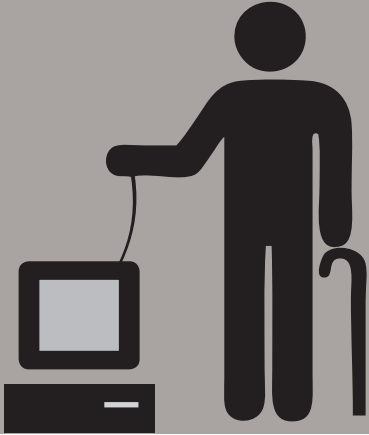
Nem todas as pessoas que publicam algum conteúdo na Web estão ligadas à área de Informática e, por isso, não necessariamente tem conhecimento das tecnologias utilizadas para construir sites. Contudo, todas as pessoas que publicam conteúdos na Web, seja texto, áudio ou vídeo, deveriam se preocupar com a questão da acessibilidade.

Então, mesmo que você não entenda nada de HTML, CSS ou qualquer outra tecnologia da Web, ainda assim você pode compreender quais são os pontos críticos na acessibi-

lidade de um site, pode aprender a realizar testes de usabilidade e acessibilidade ou pode realizar a nobre missão de informar as outras pessoas que ainda não despertaram para este problema que atinge a maioria esmagadora dos sites da Web.

Gerentes de Tecnologia da Informação, gerentes de marketing e desenvolvedores de negócios também deveriam ler este livro. Cada um dentro de sua área e por razões diferentes deveriam se preocupar em garantir acesso aos produtos que desenvolvem. Tal atitude traria diversos benefícios, além de criar novas oportunidades de negócios.

Programadores, web masters, web designers, consultores de usabilidade, pessoas que possuem ou administram sites e estudantes de informática devem se preocupar em aprender como garantir acesso a pessoas com variadas necessidades especiais. Mais fácil do que consertar um site que não está acessível é já construí-lo levando-se em consideração que pessoas com deficiências, sejam elas temporárias ou permanentes, vão querer acessar este site.



# 1

## **O mínimo que você deve saber...**

Para que você entenda claramente do que estamos tratando aqui, é necessário que primeiro compreenda alguns conceitos básicos. De nada vai adiantar começarmos a explicar a aplicação de algumas técnicas em códigos de páginas da Web sem antes definirmos alguns termos que geralmente causam bastante confusão.

Tais conceitos são muito importantes para que você possa ter uma visão ampla sobre porque alguns sites são inacessíveis para algumas pessoas e o que já tem sido feito para resolver este problema.



## **1. Afinal, o que é “Web Acessível”?**

### **O que quer dizer Web?**

Para começar, não confunda mais Internet com Web. Apesar de serem extremamente ligadas, são coisas bem distintas e tem propósitos diferentes. Quem veio primeiro na história foi a Internet, criada inicialmente para fins militares. Sua função era, e ainda é, conectar computadores. A Internet fez isso tão bem que hoje temos uma rede mundial de computadores. Alias, “rede mundial de computadores” define muito bem o que é Internet.

Uma vez que já temos uma rede de comunicação bem estruturada entre computadores, agora precisamos de programas que permitam a troca de mensagens entre esses computadores. Assim, surgiram os servidores de e-mail e os programas clientes de e-mail. Basicamente funcionam como o sistema de correio convencional, só que ao invés de termos agência de correios, temos servidores de e-mail. No lugar do carteiro, temos a internet com todo seu cabeamento e protocolos, e os remetentes e destinatários são sempre computadores.

Mas trocar informações por e-mails não era suficiente. Até então, em termos práticos, a Internet era pouco mais do que uma ferramenta para troca de correspondências e disponibilização de arquivos. Além disso, não havia compatibilidade entre os diversos sistemas de informação utilizados. O problema era: dado às circunstâncias, como compartilhar

publicações e tecnologias entre pessoas geograficamente distantes, facilitando o acesso a qualquer tipo de informação?

Nos anos 80, O Centro Europeu de Pesquisa Nuclear (CERN) estava particularmente interessado em solucionar esta questão. Para tanto contou com a colaboração de Tim Berners-Lee, um engenheiro de sistemas com larga experiência em telecomunicações. Tim primeiramente criou um tipo padrão de documento eletrônico e, com isso, resolveu o problema de incompatibilidade entre arquivos disponibilizados na internet.

Segundo a solução de Tim, os arquivos seriam disponibilizados na internet seguindo uma idéia parecida com a de troca de e-mails: através de um programa servidor (responsável por disponibilizar o documento) e programas clientes (que fariam a solicitação deste documento ao servidor).

Mas o que foi realmente genial na solução de Tim Berners-Lee foi a utilização de URL's (endereços de páginas) para acesso aos diversos documentos e a possibilidade de vincular esses documentos. Funciona assim: cada documento, seja texto, imagem, som ou vídeo, teria uma URL própria. Para abrir qualquer um destes documentos, bastava solicitar isto no programa cliente, que hoje conhecemos pelo nome de navegador. Dentro do próprio documento de texto existem links para outros documentos.

A possibilidade de vincular um documento a outro (e este segundo documento a um terceiro, e assim por diante)

nos dá uma sensação de que estamos lidando com algo que pode fugir ao controle. A gente vai seguindo os links e, quando menos esperamos, voltamos para a primeira página, de onde saímos inicialmente. Se formos pensar em um gráfico ilustrando esse emaranhado de links, esse gráfico seria algo parecido com uma teia de aranha (em inglês, web). Daí o nome dado à invenção de Tim Berners-Lee.

Enfim, a Web nada mais é do que este mar de links, sites e sistemas on-line onde milhões de pessoas diariamente navegam e procuram encontrar tudo aquilo que desejam. É uma tecnologia criada para tornar qualquer tipo de informação ou serviço acessível a qualquer hora do dia ou da noite.

## **O que quer dizer acessível?**

Geralmente, quando ouvimos falar que algo é “Acessível”, nós primeiramente pensamos em deficientes físicos. Você provavelmente já ouviu alguém dizer: “aquela passarela nova é acessível” ou “aquele ônibus é acessível”. Mas com uma olhadinha rápida no Aurélio, logo descobrimos que algo é acessível quando é fácil de aproximar, tratar ou obter. Isto sem dúvida melhora a compreensão do que é “ser acessível”, ou seja, é bem mais do que pensávamos inicialmente.

Por exemplo: nem tudo que é fácil para um adulto normal é fácil para uma criança, ou um idoso, ou um deficiente físico e assim por diante. Nem tudo que é acessível para um cego é também acessível para um paraplégico ou simplesmente para alguém com a perna engessada. Cada pessoa,

independente de ser deficiente física ou não, possui um grau diferente de necessidade para poder acessar alguma coisa.

Quando dizemos, portanto, que algo é acessível, isto deveria significar que qualquer pessoa, independente de sua necessidade, terá facilidade em entrar, aproximar, subir, utilizar, etc.

## **O que quer dizer Web acessível?**

Web acessível é a representação de uma Web ideal, onde todas as pessoas teriam acesso ao seu conteúdo. Não só pessoas, mas também sistemas, uma vez que sistemas também acessam conteúdos de páginas na internet para algum propósito, dependendo do sistema.

Um exemplo: o sistema de buscas mais utilizado no mundo e que vale mais de dois bilhões de dólares, o Google. Ele garimpa toda a Web procurando os resultados mais relevantes para uma busca. Alguém aí pode estar se perguntando “como o Google faz isso?”. Bom, uma forma de se fazer isso é entrar em cada página e “ler” seu conteúdo, da mesma forma que os leitores de telas utilizados por cegos. Quando o sistema não consegue acessar alguma página ou site, este ou é indexado de maneira equivocada ou é simplesmente deixado de lado.

Acessibilidade Web também está ligada a diversos dispositivos. Hoje em dia, páginas da web também podem ser exibidas em celulares, palmtops e até em televisão. Se não

houver uma preocupação em como as informações do site serão exibidas nestes dispositivos, há uma forte possibilidade de que seu conteúdo não seja disponibilizado adequadamente.

Existem vários fatores que podem impossibilitar ou dificultar o acesso ao conteúdo de uma página. Na introdução deste livro vimos alguns casos que relatam a dificuldade que algumas pessoas encontram para acessar informações devido a determinadas características de um site. O que veremos mais adiante é como podemos detectar pontos críticos de acessibilidade e como poderemos corrigir o problema.

Contudo, ainda precisaremos explicar mais alguns detalhes sobre o funcionamento da Web, o que faremos no próximo tópico.

## **2. Sobre a Web e seus componentes.**

Para que a Web funcione, ela depende de certos componentes; o mais óbvio deles é a própria Internet, ou seja, toda esta estrutura de cabos, placas de redes, protocolos, fibra óptica e computadores. A Internet é a plataforma onde a Web funciona.

A Web é composta por documentos e sistemas. Ou seja, páginas da Web, sistemas de webmails, sistemas de busca e tudo mais que roda na Web são outro tipo de componente da Web, e nós chamaremos de Conteúdo.

Nenhum conteúdo existiria se não houvesse quem o

criasse. Portanto, aqui entram os Desenvolvedores de websites e sistemas Web como o terceiro componente essencial da Web.

Mas todo desenvolvedor utiliza programas que o ajudam a criar conteúdo para a Web: são os softwares de criação de conteúdo. Além desses, os desenvolvedores também utilizam softwares de validação, que validam o código criado pelo desenvolvedor e avaliam a acessibilidade do conteúdo. Muitas vezes essas duas ferramentas são integradas em uma só ou trabalham em parceria. Tais softwares são nosso quarto grupo de componentes essenciais da Web. Nós poderemos chamá-los de ferramentas de criação e validação.

Se tivermos quem cria conteúdo como um componente da Web, então teremos também o usuário da Web como outro componente. Usuário é toda e qualquer pessoa que navega, acessa e lê conteúdos publicados na Web.

Mas para navegar pela Web, é preciso que o usuário possua também um navegador, como o Internet Explorer, Opera ou Firefox. Algumas vezes é possível que um usuário não esteja acessando a Web através de um navegador convencional como estes anteriormente citados. Um usuário pode acessar conteúdos da Web através de uma TV digital, por exemplo. Então, chamaremos de Agentes de Usuários todo sistema ou dispositivo capaz de requisitar acesso a qualquer conteúdo da Web e este será o sexto componente.

Não podemos nos esquecer que muitos usuários não

podem acessar conteúdos da Web diretamente pelo agente de usuário. Usuários cegos, por exemplo, se utilizam de leitores de telas ou de dispositivos especiais que reproduzem o conteúdo de uma página da Web em braille. Essas ferramentas que auxiliam usuários que possuem algum tipo de necessidade especial são chamados de ferramentas assistivas, nosso último grupo de componentes da Web.

A figura abaixo foi retirada do site Iniciativa de Acessibilidade Web<sup>1</sup> (sigla em inglês: WAI), que está vinculado ao W3C, uma organização presidida pelo inventor da Web e que se preocupa em criar padrões para as tecnologias utilizadas na Web. A figura representa todos estes componentes e como eles se relacionam entre si.

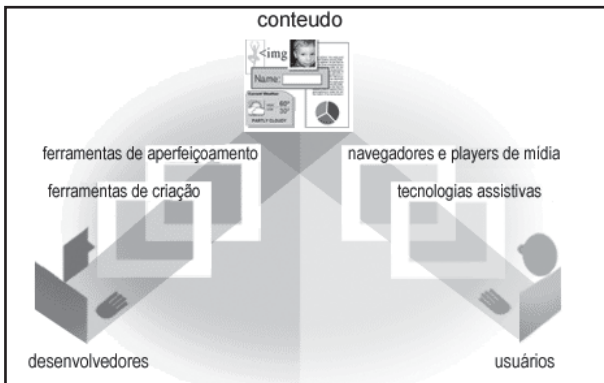


Figura 1. Componentes da Web e seus relacionamentos

É exatamente aí, entre o usuário e o conteúdo, que

1 <http://www.w3c.org/WAI>

o problema da falta de acessibilidade aparece. E se o usuário for cego? E se for surdo? E se não tiver uma das mãos, ou estiver temporariamente incapacitado?

Foi por causa das necessidades especiais destas pessoas que as ferramentas assistivas foram criadas. A tendência, nesse ponto, é pensar: “Então, se existem as ferramentas assistivas, logo está tudo resolvido. Todas as pessoas podem navegar pela internet.” Mas não é bem assim.

O fato é que só se pode garantir uma boa acessibilidade a um determinado site quando todos esses componentes estão bem relacionados entre eles. Não pode haver uma boa acessibilidade se um determinado browser (navegador) não der suporte a nenhum leitor de tela, por exemplo. Também não haveria motivação para um desenvolvedor programar algo que facilitasse o acesso a uma página Web se as ferramentas de criação ou os agentes de usuário não derem suporte para tal facilidade que o desenvolvedor deseja programar.

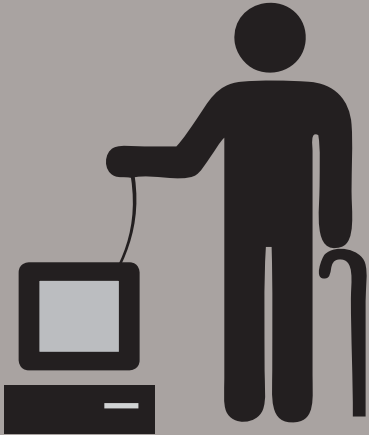
É muito comum encontramos desenvolvedores de conteúdo que não estão preocupados se algum deficiente físico vai tentar acessar a página que ele criou. O que geralmente acontece é que muitas páginas da Web não podem ser acessadas pelas ferramentas assistivas, causando transtorno para aqueles que dependem de tais ferramentas.

Um exemplo claro de inacessibilidade é o simples fato de permitir a navegação num site somente pelo mouse. Isto já é suficiente para deixar de lado todas as pessoas que, por



algum motivo, não podem utilizar este dispositivo e navegam através do teclado.

Se você ainda não se deu conta, pessoas que possuem somente deficiência na visão, apesar de geralmente não apresentarem nenhum problema nos braços e nas mãos, também estão incapacitadas de utilizar o mouse por um motivo simples: mouse é um dispositivo que, para ser utilizado, precisa da mão e dos olhos do usuário. Faça um teste e tente usar o mouse com os olhos fechados.



# 2

## **Diretrizes para construção de conteúdo acessível**

Até agora, pudemos perceber que o problema da falta de acesso a páginas Web ocorre justamente entre os componentes Usuário e Conteúdo. Tal dificuldade é um reflexo de um relacionamento fraco entre outros dois ou mais componentes da Web. Por exemplo, um fraco vínculo entre o desenvolvedor e a ferramenta de criação pode acarretar num conteúdo mal formado.

O quadro piora se as ferramentas de validação de conteúdo não se basearem em regras claras para poder avaliar o conteúdo adequadamente. Por este motivo, a melhor solução para garantir que todos os componentes funcionem harmoniosamente bem é estipular conjuntos de regras que mostrem claramente os pontos críticos de acessibilidade.

Existem atualmente vários documentos internacionais que propõem regras de acessibilidade para Web com o propósito de orientar desenvolvedores de ferramentas de criação, ferramentas de avaliação e desenvolvedores de conteúdo. Todos, no entanto, baseiam-se em diretrizes do W3C. Por esse motivo, iniciaremos este capítulo entendendo como as diretrizes do W3C estão organizadas e depois seguiremos analisando outras diretrizes que possuem relevância dentro da realidade brasileira.

## 1. Diretrizes W3C

O World Wide Web Consortium (W3C) é uma organização mundialmente conhecida por elaborar documentos de especificação de tecnologias especialmente criadas para a Web, tais como o HTML, XHTML, CSS, dentre muitas outras.

O W3C procura desenvolver padrões de tecnologias para Web de forma que possibilite a criação e interpretação dos conteúdos para Web. A idéia é que sites desenvolvidos segundo tais padrões possam ser acessados por qualquer pessoa ou tecnologia, independente de qual hardware ou software seja utilizado.

Por intermédio da iniciativa WAI (*Web Accessibility Initiative*), o W3C desenvolve diretrizes para acessibilidade Web. Existem diretrizes específicas para diferentes grupos de componentes:

- ATAG (*Authoring Tool Accessibility Guidelines*) – é

destinada para fabricantes de ferramentas de criação de conteúdo. Ela orienta os fabricantes a implementarem funcionalidades em seu produto que ajude ao desenvolvedor de conteúdo a obedecer as recomendações descritas na WCAG.

- WCAG (*Web Content Accessibility Guidelines*) – é destinada para conteúdo das páginas Web, e é utilizada por desenvolvedores de páginas Web. Fabricantes de ferramentas de criação e ferramentas de avaliação também consultam este documento com o objetivo de refinar melhor o funcionamento de seus produtos e criar uma aderência melhor ao trabalho realizado pelo desenvolvedor de conteúdo.

- UAAG (*User Agent Accessibility Guidelines*) – é destinada para desenvolvedores de web browsers e players de mídia, incluindo também alguns aspectos das tecnologias assistivas.

Todas essas diretrizes foram fundamentadas nas especificações técnicas da Web (HTML, XML, CSS, SVG, SMIL, etc.), que também são desenvolvidas pela W3C.

A primeira versão da WCAG ficou pronta em 1999 e desde então tem sido largamente utilizada, reproduzida e referenciada por diversos documentos sobre o assunto.

A WCAG possui 14 diretrizes e cada diretriz se desdobra em alguns pontos de verificação. Como é um documento um pouco extenso, para facilitar o entendimento dessas diretrizes, a WAI organizou a WCAG em níveis de prioridade.

Dessa forma, cada ponto de verificação está associado a um nível de prioridade, que pode ser 1, 2 ou 3.

Os pontos de verificação de prioridade UM devem ser satisfeitos para que todos os grupos de usuários tenham a possibilidade de acessar as informações contidas no documento.

Os pontos de verificação de prioridade DOIS deveriam ser satisfeitos. Caso contrário, um ou mais grupos de usuários terão dificuldades em acessar as informações contidas no documento.

Os pontos de verificação de prioridade TRÊS podem ser satisfeitos e existem para evitar que alguns grupos de usuários sintam alguma dificuldade em acessar as informações contidas no documento.

Existe uma versão mais recente da WCAG (versão 2.0) que recomenda acessibilidade num sentido mais amplo que a versão anterior. Enquanto a WCAG está mais focada em acessibilidade para deficientes físicos, a segunda versão deste documento foi desenvolvida para aplicação à diferentes tecnologias da Web.

## **2. Diretrizes Brasileiras: E-MAG**

A partir de uma iniciativa do Ministério do Planejamento, o Brasil, assim como em outros países, também criou o seu modelo de acessibilidade. O Modelo de Acessibilidade

do Governo Brasileiro (E-MAG) está organizado em dois documentos: A Cartilha técnica (ou visão técnica) contém as diretrizes de adequação de conteúdo da Web, sendo direcionada para profissionais de informática, e o Modelo de Acessibilidade (ou visão do cidadão) contém orientações que auxiliam na implementação das diretrizes e procura facilitar o entendimento do modelo.

O modelo brasileiro foi elaborado com base nas normas adotadas em outros países e, foi principalmente baseada na WCAG, porém, segundo o próprio modelo, tudo foi feito de forma que ficasse coerente com as necessidades brasileiras.

O e-MAG adotou os mesmos níveis de prioridade das recomendações estabelecidas pelo WAI e definiu também três níveis de acessibilidade.

Para quem nunca chegou a ler um documento de orientação de acessibilidade, o e-MAG é um bom começo. A organização da cartilha técnica facilita bastante a compreensão das diretrizes e todas as recomendações de acessibilidade do documento se assemelham muito aos pontos de verificação adotados pela WCAG 1.0.

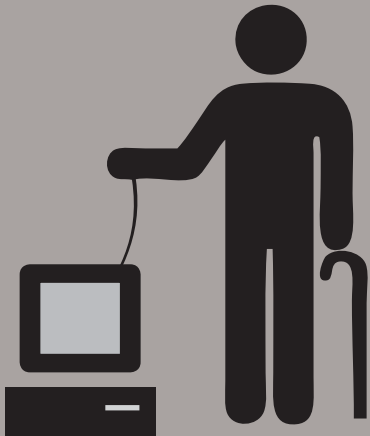
### **3. Diretrizes WCAG Samurai**

WCAG Samurai é um grupo de desenvolvedores que não possuem nenhuma ligação com o W3C e que estavam insatisfeitos com algumas características da WCAG 2.0. Por isso propuseram uma lista de correções para a WCAG 1.0

e disponibilizaram este documento como uma alternativa à WCAG 2.0. O documento WCAG Samurai errata está disponível desde fevereiro de 2008.

Afinal o que há de errado com WCAG 2.0? O grande problema deste documento é seu tamanho. A WCAG 2.0 é grande demais para transmitir, de uma maneira fácil de compreender, as idéias sobre acessibilidade na Web. Procurando reduzir sua complexidade, muitos desenvolvedores sugeriram a criação de uma versão compacta. Mas a dificuldade para a criação desta versão mais compacta está na escolha de qual seria seu conjunto de regras apropriado.

O documento WCAG Samurai Errata, na verdade procura fazer uma correção de algumas recomendações da versão 1.0 da WCAG. O motivo pra isso é simples: muitas recomendações escritas na WCAG 1.0 são consideradas desatualizadas por que este documento foi publicado em 1999 e muita coisa mudou nesses 9 anos, principalmente por causa da absorção de novos conceitos por parte dos desenvolvedores, tais como padrões Web, desenvolvimento em camadas, Web 2.0 e Web semântica.



# 3

## **Regras simples, ótimos resultados!**

Nunca se falou tanto em acessibilidade quanto nos últimos anos, e isto não é uma realidade exclusivamente brasileira. Muitos países reformaram sua legislação para que ela englobasse os últimos conceitos sobre o assunto, incluindo aí os mais atualizados padrões de acessibilidade para a tecnologia da informação.

Toda esta mobilização dos países é empolgante e serviu para que muitas pessoas que estão de alguma forma envolvidas com projetos Web obtivessem conhecimento sobre acessibilidade, ou na pior das hipóteses, pelo menos ouviram falar.

Ainda assim, o número de sites que se preocuparam em tornar seu conteúdo acessível é quase inexpressivo perto



da quantidade de sites que não se preocuparam com isso. Mas se nós já temos uma Lei<sup>1</sup> que trata sobre o assunto e se nós já temos diretrizes que orientam como devemos proceder então o que está faltando?

Lêda Spelta procurou explicar isso em seu artigo “Sete Mitos e Um Equívoco”<sup>2</sup> . A base de todo problema se encontra na total falta de conhecimento sobre o que é mito e o que é realidade quando o assunto é sites acessíveis. Muitos projetistas de sites, sem uma base concreta, acreditam que fazer um site acessível é um trabalho muito demorado e custoso. Outros acreditam que é uma tarefa difícil de ser realizada.

Mas em muitos casos, não é nem uma coisa nem outra. Se a acessibilidade de um site for planejada junto com o desenvolvimento do próprio site, além de não atrasar o andamento do projeto, a acessibilidade poderia ter um custo baixo ou até nulo.

Uma dificuldade maior poderia aparecer quando o site já existe e precise ser adequado, mas ainda assim o resultado final deste trabalho compensaria o esforço.

Em todo caso, sempre podemos tomar alguns cuidados que já melhorariam a acessibilidade do site. São coisas

1 Decreto-lei 5296 de 2 de dezembro de 2004

2 Publicado no site Acesso Digital. Di

sponível em:

[http://acessodigital.net/art\\_acessibilidade-web-7-mitos-e-um-equivoco.html](http://acessodigital.net/art_acessibilidade-web-7-mitos-e-um-equivoco.html)

simples de serem feitas, mas que fazem uma grande diferença, principalmente se o site for totalmente inacessível. Então, o mínimo que você pode fazer pela acessibilidade de uma página é isto:

1) Mantenha o HTML de sua página semanticamente correto e válido – por exemplo, use sempre “h1”, “h2” para títulos e “p” para parágrafos. Sempre valide seu HTML.

2) Organize o conteúdo de sua página de forma que ele faça sentido quando for lido – uma boa dica aqui é desabilitar o css de sua página para verificar como o navegador renderiza seu código HTML.

3) Sempre disponibilize texto alternativo para qualquer conteúdo visual – nós veremos como fazer isso ainda neste capítulo.

4) Verifique se todos os seus títulos e textos de links fazem sentido, mesmo quando são lidos fora do contexto.

Você pode encarar estas quatro orientações como um ponto de partida para acessibilidade do conteúdo. No entanto, seguir somente estas quatro premissas não garante uma validação do conteúdo, nem humana nem por sistema avaliador.

Passar pelas validações de acessibilidade vai exigir do projetista mais atenção às diretrizes adotadas, seja WCAG, e-

MAG, WCAG-Samurai ou qualquer outra. Mas nem por isso o trabalho se torna de difícil compreensão e execução.

A seguir discutiremos uma lista de orientações criadas a partir das diretrizes WCAG 1.0 e que pode ser bastante útil caso você ainda esteja dando seus primeiros passos. Perceba que nenhuma dessas orientações exige um nível de conhecimento muito aprofundado sobre HTML e CSS.

## **1. Identificando a linguagem utilizada**

Essa é uma das primeiras coisas que todos deveriam checar quando criam uma página. Aliás, identificar idioma principal utilizado num documento faz todo sentido e é uma coisa muito simples de fazer.

A razão pela qual esta identificação é importante é que, em primeiro lugar, isso ajuda os sites de busca a indexar melhor a página quanto à língua em que ela foi escrita. Além disso, esta identificação auxilia os softwares leitores de tela a decidir em que língua o conteúdo desta página será falado.

Como foi dito antes, identificar o idioma de um documento é muito simples e vai depender apenas do tipo documento que você está criando e a alteração vai ocorrer justamente na linha que identifica o tipo de documento.

Por exemplo, na tabela abaixo nós podemos ver como fica a declaração de tipo de documento quando o idioma é português do Brasil:

Tipo de Documento	Declaração para idioma pt-br
HTML	<code>&lt;html lang="pt-br"&gt;</code>
XHTML (transitional)	<code>&lt;html xmlns="http://www.w3.org/1999/xhtml" xml:lang="pt-br" lang="pt-br"&gt;</code>
XML e XHTML (strict)	<code>&lt;html xmlns="http://www.w3.org/1999/xhtml" xml:lang="pt-br" lang="pt-br"&gt;</code>

Uma observação importante aqui é os leitores de tela mais avançados conseguem fazer uma identificação automática do idioma. Contudo ainda existem no mercado muitos softwares leitores de tela que não possuem esta facilidade.

## 2. Trabalhando com texto alternativo

“Alt” é um parâmetro encontrado em algumas marcações de HTML e XHTML, principalmente naquelas que representam algum elemento não textual. Por exemplo, a marcação de imagem, como é mostrado no fragmento de código abaixo:

```

```

Esta marcação seria passada para o navegador, que entenderia que precisa carregar a imagem 'imagem1.jpg' nesta página. O problema é que a informação contida na imagem só estaria disponível para aqueles que conseguem enxergar. Como poderemos saber o que a imagem está retratando sem, contudo, visualizar a imagem? Bom, nós podemos vincular a imagem a um texto alternativo, da seguinte forma:

```

```

Dessa forma, agora já temos uma idéia do que se trata a imagem que está na página. O programa leitor de tela, ao se deparar com esta marcação irá procurar se existe algum "alt" declarado para poder ler. Sistemas que fazem busca por imagem, também se utilizam do texto alternativo para poder indexar as imagens contidas em websites.

Porém nem todas as imagens que estão numa página precisam ser descritas. Algumas imagens só têm propósitos decorativos e não possuem nenhuma relevância para o conteúdo da página. Nestes casos, declare um "alt" com espaço em branco:

```

```

A maioria dos usuários da Web tem pressa de encontrar o que desejam em uma determinada página, por isso, dificilmente eles lêem uma página da Web linha por linha até que encontrem o que estão procurando. Usuários que utilizam programas leitores de tela não são diferentes neste ponto. Eles também têm pressa em encontrar o que desejam e não querem ler todas as palavras de uma página.

A diferença é que usuários cegos não lêem o conteúdo, eles escutam. Como eles querem encontrar a informação procurada o mais rápido possível, então eles não param para ouvir todas as palavras. Geralmente eles vão pulando para o próximo link, próximo parágrafo até chegarem ao ponto que desejam. Muitos usuários de leitores de tela têm o costume de configurar a voz do sintetizador em uma velocidade alta para ganhar tempo enquanto ouvem o conteúdo.

Alguns desenvolvedores de conteúdo para a Web acham equivocadamente que declarar um texto alternativo para todas as imagens de uma página ajuda àqueles que não podem ou tem dificuldade de enxergar e colocam textos alternativos inclusive em imagens decorativas: “aqui é uma bola”, “imagem de arredondamento de canto”, “uma linha pra separar o rodapé”, etc. Perda de tempo! Isso só dificulta mais a leitura do documento.

### **3. Trabalhando com CSS**

Quando comecei a criar páginas para a Web, ninguém falava ainda em CSS. Tudo era feito usando o HTML e Ja-

vascript, para criar algum dinamismo na página. Uma página com muitos detalhes de layout era sinônimo de uma página com muitas tabelas aninhadas, ou seja, tabela dentro de tabela. Quanto mais tabelas nós tínhamos no documento, mais marcações <TR><TD> teríamos para gerenciar.

Também toda informação de cor, tamanho, fonte, tudo enfim deveria estar no HTML. O resultado disto: um documento altamente confuso e difícil de manusear. Era impossível abrir mão de um bom programa de edição de código HTML tipo WYSIWYG (What You See Is What You Get), como o Macromedia Dreamweaver.

Neste cenário caótico, o CSS aparece como um forte aliado, tornando possível separar de uma vez por todas o conteúdo de uma página de sua estrutura e apresentação. Dessa forma, toda estrutura e conteúdo de uma página devem ficar no HTML, enquanto o CSS se encarrega de sua apresentação.

Entenda por apresentação toda informação de tamanho, largura, altura, tipo de fonte, altura de linha, cores, posicionamento na página, tudo que faça referência a como o conteúdo deve ser apresentado na tela. Esta técnica ou metodologia de criar uma página Web separando-a em camadas é conhecida como tableless. A metodologia Tableless recebeu este nome por não permitir, no HTML, a utilização de tabelas para montagem de apresentação de conteúdo.

Existem muitas vantagens na utilização de CSS para

apresentação de uma determinada página da Web. Uma das mais fortes é que o HTML agora fica livre de tabelas aninhadas e de elementos que só são utilizados para apresentação de conteúdo, como o elemento `<font>`. Dessa forma, fica muito mais simples fazer qualquer manutenção no conteúdo desta página.

Do ponto de vista da acessibilidade, uma vez que todos os dados da apresentação do conteúdo estão separados em um arquivo CSS, é muito simples fazer alterações na apresentação deste conteúdo e até mesmo criar apresentações diferentes para tipos de mídia diferentes. Assim, você pode criar, por exemplo, um conjunto de regras CSS para apresentação na tela do computador e outro conjunto para impressão do conteúdo, eliminando algumas informações desnecessárias na impressão.

### **Um exemplo prático:**

Perdido no meio de tantos conceitos? Então vamos melhorar um pouco as coisas com um exemplo bastante prático. Quem trabalha com desenvolvimento para a Web, ao menos uma vez já precisou montar um layout de página com três colunas. Quase instantaneamente pensamos: “Simples! Basta criar uma tabela, inserir as três colunas e está feito o layout.” Realmente parece simples, não? Vamos ver como fica o código HTML para a solução apresentada:



```
<table width="95%" border="0">
  <tr>
    <td colspan="5">
      <h1 align="center"><font
face="Verdana, Arial, Helvetica, sansserif"> Layout
com 3 Colunas</font></h1>
    </td>
  </tr>
  <tr>
    <td valign="top"><div align="justify">
      <font face="Arial, Helvetica,
sansserif">Conte&uacute;do da esquerda Lorem ipsum
dolor sitamet, consectetuer adipiscing elit. Ut vo-
lutpat. Nulla pellentesque. In sed neque.</font></
div>
    </td>
    <td></td>
    <td valign="top">
      <p align="justify"><font
face="Arial, Helvetica, sans-serif">Escreve alguma
coisa...</font></p>
```

O código acima é somente um fragmento de todo trabalho que teremos que realizar para criar e manter um site utilizando a técnica convencional. Perceba que no trecho de código, além de utilizarmos tabelas para criar a disposição do conteúdo, também abusamos no uso da tag `<font>`. O que aconteceria se esta página fizesse parte de um portal com mais de 100 páginas como esta? E se precisássemos alterar o

tipo de fonte ou a cor da fonte em todas as páginas?

Apesar do código estar bastante poluído com os elementos de tabela, a renderização deste HTML é bastante simples. Com este código, o navegador simplesmente vai distribuir a informação em 3 colunas, conforme a imagem abaixo:

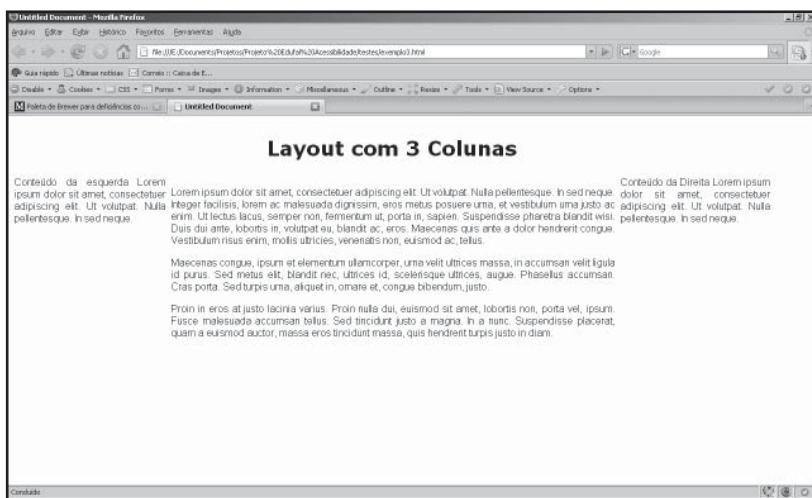


Figura 2. Exemplo de layout com 3 colunas

Quando utilizamos a metodologia Tableless, a primeira coisa que fazemos é eliminar a tabela. No lugar dela, utilizaremos a tag <div>, conforme mostra o trecho de código a seguir:

```
<body>
  <div id="geral">
    <h1>Layout com 3 Colunas</h1>
    <div id="esquerda" class="coluna">Conteúdo da colu-
na da esquerda...</div>
    <div id="conteudo" class="coluna">
      <p>Lorem ipsum dolor sit amet,
consectetuer adipiscing elit. Ut volutpat. Nulla
pellentesque. In sed
neque. Integer facilisis, lorem ac malesuada dig-
nissim, eros metus posuere urna, et vestibulum urna
justo ac enim. Ut
      </p>
      <p>Maecenas congue, ipsum et ele-
mentum ullamcorper, urna velit ultrices massa, in
accumsan velit
ligula id purus. Sed metus elit, blandit nec, ul-
trices id,scelerisque ultrices, augue. Phasellus
accumsan. Cras porta.
      </p>
      <p>Proin in eros at justo lacinia
varius. Proin nulla dui, euismod sit amet, lobortis
non, porta vel,
ipsum. Fusce malesuada accumsan tellus. Sed tinci-
dunt justo a magna. In a nunc. Suspendisse place-
rat, quam a euismod
      </p>
    </div>
    <div id="direita" class="coluna"> Con-
teúdo da Direita Lorem ipsum dolor sit</div>
  </div>
</body>
```



de conhecimento sobre como manipular folhas de estilo. Neste ponto, a primeira coisa que você deve fazer é criar um link entre a folha de estilo e seu código HTML. Isto é feito dentro da declaração `<head>` do seu documento. Assim:

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

Estilo.css é seu arquivo que define as regras que serão aplicadas ao documento. Mas estas regras só serão aplicadas para um certo tipo de mídia, que é definido no atributo `media`. Neste caso, esta folha de estilo serve somente para monitores (`screen`). Existem outras medias para as quais você também pode criar folhas de estilo. As mais comuns são: `screen`, `print` e `handheld`.

Para o nosso exemplo, que queremos construir um layout de 3 colunas, a folha de estilo poderia ser definida da seguinte forma:

```
h1{
    font-family: Verdana, Arial, Helvetica, sans-
    serif;
    font-size: 2em;
    font-weight: bold;
}
#geral{
    width:710px;
    text-align: center;
    position: absolute;
left: 50%;
    margin-left: -355px;
}
#esquerda, #direita {
    width:150px;
}
.coluna {
    font-family: Verdana, Arial, Helvetica, sans-
    serif;
    font-size: 70%;
    text-align: justify;
    float:left;
    margin:2.5px;
    padding:2.5px;
}
```

Toda regra CSS é formada por um Seletor, propriedades e valores. Propriedades e valores sempre ficam entre as chaves e o seletor é a parte que define em qual elemento esta regra será aplicada. Por exemplo, a primeira regra do código CSS descrito anteriormente será aplicado a qualquer elemento `h1` existente no HTML e as alterações que `h1` irá sofrer se encontram entre as chaves: mudança de tipo, tamanho e lar-

gura de fonte.

E as três colunas do layout? Quem vai definir isso no CSS de exemplo? É exatamente a última regra do nosso exemplo, que será aplicada a qualquer elemento que pertença a classe `coluna`. O ponto (.) antes da palavra “coluna” indica que esta palavra está se referindo a uma classe. Se você olhar novamente o código HTML, vai perceber que existem 3 elementos `<div>` que pertencem a esta classe “coluna”. Esta regra se aplica exatamente a esses três elementos. E, mais precisamente, a regra `float:left` é que faz com que os três elementos `<div>` se posicionem um ao lado do outro, criando assim o layout de 3 colunas.

A Web está repleta de informações que orientam como os desenvolvedores podem utilizar HTML e CSS em conjunto. Um ótimo começo para quem deseja aprender mais sobre o assunto é fazer uma pesquisa na Web por “Tableless”. Contudo, no capítulo 7 você encontrará uma lista de ótimos artigos disponíveis na Web sobre o esta metodologia.

## **4. Trabalhando com medidas relativas**

Quando trabalhamos com CSS, temos à disposição várias unidades de medidas: pixel, pontos, centímetro, milímetro, em, porcentagem, dentre outras. Essas unidades dividem-se em absolutas ou relativas.

Unidades Absolutas	Unidades Relativas
Polegadas (in)	Pixel (px)
Centímetros (cm)	Ems (em)
Milímetros (mm)	Ex (ex)
Pontos (pt)	Porcentagem (%)
Picas (pc)	

Qualquer elemento que possuir alguma de suas dimensões declarada como relativa sempre vai tomar como referência as dimensões de algum outro elemento, dependendo da unidade de medida.

No caso da unidade em, usa-se como referência o tamanho da fonte do seletor em que ela foi declarada. Por exemplo, observe a seguinte regra CSS:

```
P {  
  font-size: 14px;  
  line-height: 1em;  
}
```

Declaramos para a altura da linha do parágrafo (line-height) um valor relativo (1 em). Para calcular quanto é este valor em pixel, basta verificar qual é o tamanho da fonte declarada para este mesmo seletor (14px) e multiplica-lo pelo valor relativo declarado ( $14 * 1 = 14px$ ). Assim, o valor em pixels da altura da linha para a regra acima será 14. Observe



que, se `line-height` fosse declarado com o valor `0.5em`, a altura da linha mediria a metade de `font-size`.

Caso a regra não defina qual é o valor de `font-size`, então o valor que seria tomado como referência seria o `font-size` do elemento `pai`. Caso esta regra se refira ao elemento `HTML` ou `BODY`, o valor tomado por referência é o valor de `font-size` padrão do navegador.

Muitos autores classificam a medida `pixel` como absoluta. Isto acontece porque `pixel` só depende da resolução de tela e do tipo de sistema operacional, ou seja, a não ser que o usuário troque sua resolução de tela ou seu sistema operacional, ele não terá controle sobre os elementos declarados com `px`.

Para ilustrar, considere o elemento parágrafo (`P`) do exemplo anterior que foi declarado com tamanho de fonte `14px`. Se o usuário estiver utilizando o Internet Explorer para navegar nesta página e tentar aumentar o tamanho do texto<sup>3</sup>, ele não vai conseguir.

Quando levamos em consideração a acessibilidade de uma página, trabalhar com medidas relativas é muito mais apropriado. São as medidas relativas que nos permite criar layouts de página totalmente elásticos, podendo se adaptar a tamanhos de tela variados e garantindo ao usuário uma experiência confortável ao visualizar o conteúdo de uma página.

---

3 Podemos fazer isso clicando no menu `exibir > tamanho do texto > maior`

## 5. Pensando na estrutura da página

Se preocupar com a organização e estrutura da página de Web é uma obrigação do bom projetista. Ele deve pensar em como organizar o conteúdo na página de maneira que esta faça sentido, mesmo quando o usuário não possa utilizar CSS.

Lembre-se que:

- 1) Nem todo usuário navega utilizando o mouse.
- 2) Nem todo usuário pode ver sua página.

Pensar na estrutura da página é como criar uma planta-baixa da sua página. Você deve planejar o lugar onde cada elemento vai aparecer e manter essa informação consistente em todo o site.

- 1) Onde vai ficar o sistema de navegação?
- 2) Que informações vão aparecer no topo da página?
- 3) Onde vou exibir o conteúdo propriamente dito?
- 4) Vou precisar de uma navegação local? Onde vou colocar?

Todas essas indagações e outras adicionais devem ser respondidas no momento em que se planeja como as informações e links vão ser dispostos na página. Qualquer que seja a decisão tomada sobre esta organização, sua página sempre deve permitir uma navegação pelo teclado sem nenhuma restrição.

Deve-se levar em consideração que, quando se navega pelo teclado, o deslocamento é feito através da tecla TAB e das teclas de direção, sempre de cima para baixo e da esquerda para a direita. Não é aconselhável alterar este fluxo de deslocamento por um motivo simples: pessoas com deficiência visual já esperam que o deslocamento ocorra neste sentido. Leve em consideração que pessoas cegas não podem ver como sua página está organizada.

A figura abaixo é uma demonstração de como uma página pode ser estruturada. Para este exemplo foram levados em consideração as áreas: topo, navegação global, navegação local, conteúdo e rodapé.



Figura 4. Estrutura básica de um site

Algumas vezes, por algum motivo, os links ou elementos de formulário em seu HTML são escritos em uma ordem que não segue uma seqüência lógica. Nestes casos, use o atributo "tabindex" para definir a ordem de acesso. O exemplo abaixo demonstra como podemos fazer isto:

```
<a TABINDEX="10" href="página.html" title="Alguma página">ir para página</a>
```

## Descrição de página

Quando você entra num site pela primeira vez, quanto tempo você leva pra aprender a navegar por este site? Em quanto tempo você consegue encontrar o que está procurando? Se o site estiver bem organizado, bastam alguns segundos.

Acontece desta forma por que quando entramos em um site pela primeira vez, a primeira coisa que fazemos é dar uma olhada geral, procurando por alguma coisa que nos dê vontade de clicar: pode ser um link de uma notícia, pode ser uma opção de menu, ou pode ser um formulário de pesquisa no site. Isto depende do perfil de cada usuário.

O fato é que sempre estamos procurando por algo e, quando fazemos isso, contamos com um aliado fortíssimo: a nossa visão. Através da nossa visão temos acesso à página

inteira, mesmo antes clicar em qualquer coisa. Se pudéssemos perceber como as informações que são transmitidas pela nossa visão são processadas pelo nosso cérebro, poderia ser algo assim:

“Bom, deixe-me ver... Ali está a logomarca, legal. Mas onde será que eles colocaram a cessão de CD's? Será que é aqui? Não deve ser... Este site não tem busca? Pode ser ali. Será que dá pra clicar ali? Ah, olha ali o menu...aqui! Pronto! Achei!”

Não dá pra negar, a visão realmente ajuda muito em algumas tarefas. Mas nem todas as pessoas podem contar com ela. E o que acontece com pessoas que não podem enxergar? Vamos imaginar o mesmo cenário descrito anteriormente, só que agora o usuário é cego.

Quer ter uma idéia de como uma pessoa se sente numa situação destas? Então faça o seguinte: faça uma busca na internet, pode ser qualquer coisa. Depois clique em qualquer um dos links que veio no resultado, mas tem que ser um site que você nunca tenha visitado antes. Desligue o monitor antes que a página carregue e tente navegar. A sensação é de que caímos de pára-quedas em algum lugar completamente desconhecido, totalmente no escuro.

Nestes casos, existem várias formas de trazer alguma orientação. Dependendo do site, se ele for simples, com poucas opções, um mapa do site já seria suficiente. Mas quando o site é de uma complexidade considerável, então uma solução

seria criar uma página com uma descrição textual de como o site está organizado. Você pode aproveitar e colocar também nesta página uma lista de atalhos para diversos links no seu site.

## 6. Tabelas

Em 1994, o elemento TABLE entrava definitivamente na especificação do HTML, cuja versão na época era a 2.0. O TABLE foi introduzido para atender uma demanda por uma representação específica para dados tabulados ou cruzamento de dados, tipo: tabela de preços, tabela de medidas, etc.

Ninguém imaginava, até então, que os desenvolvedores iriam descobrir uma outra função para este elemento. Eles descobriram que as tabelas também poderiam controlar o layout e a apresentação das informações em uma página da Web.

A Web então, entra numa nova fase, repleta de sites com layout de alta complexidade e recheadas de tabelas (e tabelas dentro de tabelas) que não passavam nenhuma informação, só estavam lá para ajustar a apresentação da página. O código HTML passou então a cumprir duas missões: disponibilizar conteúdo e ajustar a apresentação do documento.

Contudo, o uso de tabelas para construir o layout pode causar sérios problemas à acessibilidade. Com toda essa mistura entre apresentação e conteúdo, muitas vezes um programa leitor de tela fica impossibilitado de acessar o

conteúdo, comprometendo o amplo entendimento da informação daquela página.

Na verdade as tabelas, quer sejam usadas para dados tabulares ou para construção de layout, se não forem bem estruturadas, os leitores de tela acabam fazendo uma grande confusão ao ler os dados para os seus usuários, tornando a tabela impossível de ser entendida.

Não é fácil criar tabelas totalmente acessíveis por que depende de uma série de coisas. Primeiro, depende da própria destreza do desenvolvedor para utilizar o máximo de recursos disponíveis para torná-la acessível. Depende também da própria aplicação que lê a tela, que deve possuir a capacidade de acessar os dados da tabela de diversas formas. Quanto mais complexa for a tabela, mais cuidado o desenvolvedor deverá ter ao construir a tabela.

Um bom ponto de partida é compreender como um leitor de tela lê a tabela. Na maioria dos casos, os leitores possuem uma série de atalhos pelo teclado que permitem o acesso a uma linha ou a uma célula. Uma observação importante é que o leitor de tela deve ser capaz de associar o cabeçalho de uma linha ou coluna à célula que está sendo lida. Sem isso, é impossível para um deficiente visual entender de que se trata a informação que está sendo lida.

### **Como um leitor de tela deve ler uma tabela:**

Para começar, um leitor de tela não lê (literalmente)

a tela, como o nome sugere. Os leitores de tela vão direto para o código HTML dos documentos, por que, para eles o que interessa realmente é extrair o conteúdo que está lá. Pouco importa para o software leitor como aquelas informações estão sendo diagramadas na tela.

No leitor de tela Jaws, que é um dos mais famosos, a primeira coisa que é lida numa tabela é seu título, depois é lido o sumário, se estiver disponível. O terceiro passo é posicionar o cursor de leitura na primeira célula da tabela. Após isso, o leitor de tela aguarda os comandos de navegação na tabela para continuar a leitura, que vai depender do desejo do usuário: lê a linha, lê somente a célula atual, lê a célula com seu cabeçalho correspondente, etc.

Para ficar mais claro, observe a tabela da figura abaixo. É uma tabela simples de ser entendida e consegue transmitir sem nenhum problema as informações que nela estão contidas. Não há nenhuma dificuldade, por exemplo, em descobrir que o preço do prato individual do yakisoba custa R\$ 11,00.

Tabela de preços dos pratos do restaurante chinês

<b>Nome do Prato</b>	<b>Valor Individual</b>	<b>Valor duplo</b>
Yakisoba	11,00	20,00
Yakimeshi	10,00	18,00
Yakibifum	18,00	35,00

Figura 5. Exemplo de tabela



O código HTML para esta tabela seria assim:

```
<table border="1" summary="Tabela com um nível de
cabeçalho horizontal">
  <caption>Tabela de preços dos pratos do res-
taurante chinês</caption>
  <tr>
    <th id="nomeprato">Nome do Prato</th>
    <th id="individual">Valor Individual</th>
    <th id="duplo">Valor duplo</th>
  </tr>
  <tr>
    <td headers="nomeprato">Yakisoba</td>
    <td headers="individual">11,00</td>
    <td headers="duplo">20,00</td>
  </tr>
  <tr>
    <td headers="nomeprato">Yakimeshi</td>
    <td headers="individual">10,00</td>
    <td headers="duplo">18,00</td>
  </tr>
  <tr>
    <td headers="nomeprato">Yakibifum</td>
    <td headers="individual">18,00</td>
    <td headers="duplo">35,00</td>
  </tr>
</table>
```

Observe que no código desta tabela foram utilizados algumas tags e atributos de tabela do HTML que são pouco conhecidos: `summary`, `caption`, `th` e `headers`. Vamos descreve-las agora.

- **Summary** – é um atributo de `table` que não aparece na tela, mas é lido pelo software leitor. Ele serve para que o desenvolvedor possa descrever alguma particularidade da tabela.
- **Caption** – Deve ser declarado imediatamente abaixo da tag `<table>`. Fornece um título para a tabela e é exibido na tela logo acima da tabela. Também é lido pelo leitor de tela.
- **Th** – Marca uma célula de cabeçalho. É muito importante que o cabeçalho de uma tabela seja corretamente marcado no HTML, para que o leitor de tela reconheça quais células são dados e quais são cabeçalho.
- **Headers** – este atributo de `<td>` (célula de dado) vincula uma determinada célula a um cabeçalho existente. Em `headers` é declarado o `id` do cabeçalho correspondente.

Existem outras maneiras de montar esta mesma tabela e chegar a resultados parecidos ao que chegamos aqui. De qualquer forma, o mais importante é que a tabela sempre seja construída de maneira que permita ao software leitor de tela o máximo de acesso aos dados da tabela.

Assim, para que o desenvolvedor crie tabelas bem estruturadas é necessário:

- 1) Fornecer o título da tabela através do elemento `CAPTION`
- 2) Para melhor compreensão de tabelas, fornecer um resumo informando o propósito da tabela através do

atributo SUMMARY no elemento TABLE. Vale salientar que o resumo não fica visível na página, mas o leitor de tela o descreve.

3) Utilizar TH para identificar os cabeçalhos de linhas e colunas e TD para identificar as células com dados.

4) Associar, de alguma forma, os dados TD aos seus respectivos cabeçalhos TH.

## 7. Formulários

Os formulários HTML são estruturas que permitem que usuários submetam dados a uma página. É uma das formas mais simples e eficientes de interagir com o usuário. Por este motivo, o formulário acabou se transformando numa matéria de extrema importância no HTML.

O Formulário é composto basicamente por três partes:

- O Container
- A entrada de dados
- Os botões

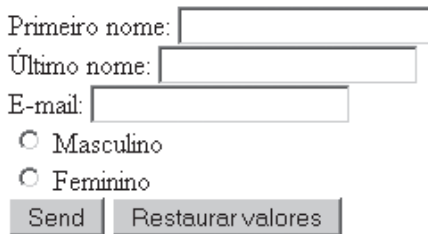
Os controles de entrada de dados e botões ficam dentro do container do formulário, que é definido no HTML através da tag <form>. O elemento <form> possui atributos específicos que informam para o navegador para onde e de que forma devem ser enviados os dados digitados no formulário.

Os controles que permitem a entrada de dados va-

riam, de acordo com o tipo de dado. Os elementos básicos que permitem a entrada de dados são:

- INPUT: que especifica vários tipos de campos como text, submit, button e radio.
- SELECT: Utilizado para listas seleccionáveis (como um menu).
- TEXTAREA: campo de texto multi-linha, como observações e comentários;

A figura 6 ilustra um formulário com controles de entrada de texto, radio (com as opções "masculino" e "feminino") e um botão para envio das informações.



Primeiro nome:

Último nome:

E-mail:

Masculino

Feminino

Figura 6. Exemplo de Formulário

Uma coisa que a maioria dos desenvolvedores desconhece é que existe uma ordem certa nos navegadores e leitores de tela para exibição e leitura dos elementos de um formulário. Por exemplo, quando um navegador encontra um controle `<input type= "text">`, ele espera que o rótulo para este controle esteja antes do controle. Já para os campos do tipo `<input type= "radio">`, o navegador espera encontrar

o rótulo depois do campo.

A tabela abaixo mostra uma relação entre o controle e seu rótulo.

Controle	Posição do Rótulo	Exemplo
<code>&lt;input type="text"&gt;</code>	Antes	Nome:  <code>&lt;input type="text" name="nome" /&gt;</code>
<code>&lt;input type="password"&gt;</code>	Antes	Senha:  <code>&lt;input type="password" name="senha" /&gt;</code>
<code>&lt;input type="radio"&gt;</code>	Depois	<code>&lt;input type="radio" name="estcivil" value="1" /&gt;</code> casado 
<code>&lt;input type="checkbox"&gt;</code>	Depois	<code>&lt;input type="checkbox" name="chkassinar" value="Assinar" /&gt;</code> Assinar a revista
Select	Antes	Tratamento  <code>&lt;select name="tratamento"&gt;</code> <code>&lt;option&gt;Sr.&lt;/option&gt;</code> ... <code>&lt;/select&gt;</code>

Controle	Posição do Rótulo	Exemplo
Textarea	antes	Comentários  <textarea name="txtComments"></ textarea>

Rótulos são sempre importantes, tanto para navegadores comuns quanto para leitores de tela. São os rótulos que identificam o que deve ser preenchido em cada campo do formulário e, em muitos casos, informam ao usuário se o preenchimento de determinado campo é obrigatório.

Por conta disso, a melhor forma de promover acessibilidade em formulários HTML é fortalecendo este vínculo entre rótulo e campo de preenchimento de dados. Para isto, utilizamos o elemento <label> do HTML. Para entendermos seu uso, vamos tomar como exemplo o formulário da figura X. O código HTML para o primeiro campo seria assim:

```
<LABEL for="primeironome">Primeiro nome:
  <INPUT type="text" id="primeironome">
</LABEL>
```

O atributo FOR do elemento LABEL associa explicitamente um rótulo a um controle. O valor atribuído a FOR deve ser o mesmo valor do atributo ID do controle associado. Neste exemplo, o elemento LABEL está associado ao elemento INPUT através do atributo FOR de LABEL que possui o mesmo

valor do atributo ID do elemento INPUT.

Existem outros elementos do HTML que auxiliam na construção de um formulário bem organizado e acessível. Por exemplo, os elementos:

- `<fieldset>` - pode agrupar um conjunto de controles de formulário de acordo com a natureza dos dados. Por exemplo: dados pessoais e dados profissionais.
- `<legend>` - Fornece um título para um determinado `fieldset`.

O desenvolvedor deve utilizá-los sempre que achar necessário e conveniente, de forma a facilitar o entendimento das solicitações do formulário.

## 8. Acrônimos e Abreviações

Muita gente desconhece, mas existem dois elementos do HTML criados para serem usados em acrônimos e abreviações. Estes elementos executam dois papéis importantes:

- 1) Provê uma definição dos termos através do atributo `title`, como por exemplo:

```
<acronym title="Estados Unidos da América">EUA</acronym>
<abbr title="Cascading style sheets">CSS</abbr>
```

Esta função, além de ajudar os usuários leitores de

tela que podem desconhecer o termo, também pode ser bastante útil para sistemas de buscas.

2) Providencia uma forma de softwares leitores de tela determinarem se o termo deve ser pronunciado ou soletrado. Isso pode ser feito de duas maneiras: ou você deixa que o software leitor de tela decida se deve ler ou soletrar, dependendo de qual elemento foi utilizado, ou então você define isso usando CSS, como nas regras abaixo:

```
abbr {speak:spell-out;}  
acronym {speak:normal;}
```

Como a finalidade aqui é assegurar que um documento seja acessível da melhor maneira possível e para um maior número de pessoas, então é muito importante que você possa compreender quando deve usar `abbr` e quando deve usar `acronym`.

Uma abreviação é uma forma encurtada de uma palavra ou frase utilizada para representar a forma completa. Por exemplo:

- Sr. É a abreviação de senhor
- Av. é abreviação de avenida.

A forma mais comum de abreviação é geralmente um conjunto de letras seguidas de um ponto. Porém existem outros tipos diferentes de abreviação:



- Inicialismo – são aquelas formadas pela primeira letra de cada palavra do termo representado. Quem trabalha com informática já deve estar bastante familiarizado com este tipo de abreviação. Por exemplo: HTML, CSS, EUA

- Contração – é muito comum no idioma inglês. Tanto pode ser uma forma encurtada de uma palavra terminando com a última letra desta palavra, como também pode ser duas palavras encurtadas juntas, mas separadas por um apóstrofo. Exemplos:

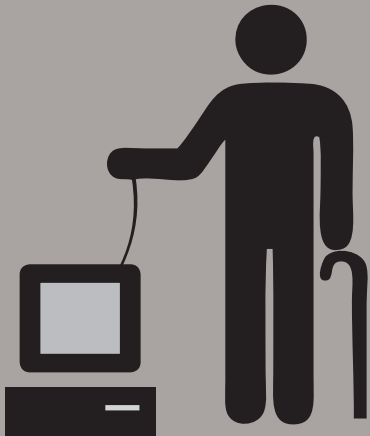
- Pagto – Abreviação de pagamento
- Can't – Abreviação de can not em inglês
- D'água – abreviação de 'de água'.

- Acronismo – é um tipo de abreviação cujo conjunto de letras que formam o acrônimo pode ser pronunciado, como se fosse uma palavra. Por exemplo:

- NASA – National Aeronautics and Space Administration
- OTAN – Organização do Tratado do Atlântico Norte.

Não existe no HTML uma forma correta de se marcar uma contração. De qualquer forma, como seu uso geralmente é muito comum, dificilmente uma contração irá merecer uma marcação especial.

No entanto deve-se marcar os demais tipos de abreviações. Somente utilize a marcação <acronym> para acrônimos.



# 4

## Técnicas para aplicação de acessibilidade

Já algum tempo nota-se uma preocupação crescente por parte da sociedade em geral em tornar o acesso a lugares e a informação ao menos um pouco melhor do que eram antes. Na fila de um banco pode-se perceber no chão, uma trilha de um material emborrachado formado por linhas e por pontos. Depois descobre-se que era um tipo de assoalho especial para ajudar na orientação de deficientes visuais. No mesmo dia, no caixa automático, havia uma entrada para fones de ouvido e o teclado numérico tinha botões em alto relevo.

As propagandas políticas agora sempre vêm acompanhadas por legenda simultânea das falas e narrações. Em vários pontos da cidade percebi rampas para acesso às calçadas.

Todos estes recursos que são instalados pela cidade e facilitam o acesso de pessoas deficientes são chamados de dispositivos de acessibilidade.

Na verdade, o conceito de dispositivos de acessibilidade é ainda mais abrangente. Tudo que é construído ou fabricado com a intenção de melhorar, facilitar ou promover acesso a alguma coisa ou algum lugar é um dispositivo de acessibilidade.

Desta forma a lista de dispositivos de acessibilidade aumenta consideravelmente, e pode ir de uma simples bengala a uma passarela construída sobre uma avenida.

Na Web também podemos construir dispositivos de acessibilidade que facilitam o acesso de pessoas deficientes. Os recursos que vamos apresentar neste capítulo são técnicas de acessibilidade que se forem bem aplicadas podem fazer diferença quando um dado usuário tenta acessar o conteúdo de algum site.

Uma coisa importante sobre este capítulo é que ele vai exigir de você um pouco de conhecimento sobre CSS, HTML, Javascript e Tableless.

Uma vez que nosso foco aqui é estudarmos as técnicas de acessibilidade existentes e que permitem promover acessibilidade através da Web, não iremos aqui nos prender a detalhes sobre estas linguagens e metodologia. Contudo, você encontrará no capítulo 5 uma lista de artigos, sites e

material para estudo.

## 1. Teclas de Atalho

Utilizar tecla de atalho é a forma mais rápida de acessar algumas partes de uma página. De um modo geral os usuários conhecem teclas de atalho, e muitas vezes as utiliza, mesmo sem perceber que estão fazendo isso.

Duas das combinações de teclas mais comuns são as famosas combinações CONTROL + C (comando para copiar) e CONTROL + V (comando para colar). Não importa em que contexto elas são aplicadas, mas teclas de atalho sempre têm um objetivo em comum: permitir que o usuário acesse pelo teclado os comandos mais importantes de um determinado sistema.

Muita gente não sabe, mas é possível colocar em páginas da Web comandos que permitam acesso rápido a um determinado recurso. Utilizando-se o atributo `accesskey`, você consegue associar uma determinada letra ou número a um outro elemento da página, como no exemplo abaixo:

```
<a href= "pagina.html" accesskey= "1">Página</a>
```

Neste exemplo nós configuramos a tecla 1 para ser o atalho do link. Para o usuário acessar este link através da tecla de atalho, e se ele estiver utilizando o Internet Explorer, por exemplo, ele terá que digitar a combinação ALT + 1.

Os elementos HTML que dão suporte ao atributo `accesskey` são todos os elementos do HTML que geralmente são associados a um evento, como o click do mouse ou o enter do teclado. Ou seja, você pode declarar um `accesskey` para os seguintes elementos: `a`, `area`, `textarea`, `legend`, `label`, `input` e `button`.

Os elementos `Label` e `Legend`, apesar de normalmente não serem associados diretamente a eventos do mouse ou teclado, também fazem parte desta lista. Isto acontece porque eles são elementos intimamente ligados a controles de entrada de dados, como o `input`. Semanticamente, não haveria sentido em utilizar esses elementos em um documento HTML se eles não estiverem rotulando uma entrada de dados ou um conjunto de entradas.

A utilização de teclas de atalho é um recurso bastante interessante e amplia grandemente as chances de sucesso de um usuário que utiliza o teclado para navegar. Contudo, de nada adianta criar teclas de atalho se o usuário não souber da existência delas. Portanto, divulgar suas teclas de atalho é tão importante quanto criá-las.

Uma maneira de divulgar suas teclas de atalho é em um documento de descrição do site, se ele tiver um, ou então no mapa do site. Também é possível divulgá-las diretamente nas páginas onde essas teclas de atalho aparecem. Cada desenvolvedor pode encontrar uma forma criativa de fazer esta divulgação, procurando preservar o layout da página.

Uma desvantagem do uso das teclas de atalho é que cada navegador da web utiliza uma combinação de tecla diferente para o mesmo accesskey.

Desta forma, supondo que você criou uma tecla de atalho para um link utilizando a tecla C (accesskey = "c"). Dependendo do navegador utilizado pelo usuário ele teria que digitar uma combinação específica de teclas. Assim:

Navegador	Combinação de Teclas
Internet Explorer (Windows)	ALT + C
Internet Explorer (MAC)	CTRL + C
Firefox, Mozilla, Netscape (Windows)	ALT + SHIFT + C
Firefox, Mozilla, Netscape (Mac)	CTRL + SHIFT + C
Safari, Omniweb (Mac)	CTRL + C
Konqueror (Linux)	CTRL + C

## 2. Skip links

Os Skip Links, também conhecido como Jump Links, são links para outra parte da mesma página. A sua função é possibilitar que os usuários possam navegar pela página saltando algumas partes da mesma.

Por exemplo, vamos supor que você quer chegar a uma determinada parte de uma página onde se encontra um

formulário. Mas para chegar ao formulário, usando somente o teclado, você terá que passar primeiro por alguns links do topo da página e depois pela navegação para só então chegar ao formulário. A técnica do skip link é justamente para evitar este tipo de problema. Utilizando um skip link, você poderia saltar direto para o formulário.

Hoje em dia, é comum em sites que se preocupam com acessibilidade encontrarmos no topo das páginas um link com o texto “Ir para o conteúdo”, “Pular navegação” ou algo parecido. Estes são skip links em ação!

É muito simples colocar skip links em uma página, principalmente se esta página foi construída utilizando-se separação de camadas. A única coisa que vamos precisar para construir estes links é saber para que parte da página o skip link vai apontar.

Por exemplo, se no código HTML se está utilizando a seguinte marcação para indicar a entrada de conteúdo:

```
<div id="conteudo">...</div>
```

Então pode-se criar no topo da página um skip link para o conteúdo através do seguinte código:

```
<a href="#conteudo" title="Skip Link para o Conteúdo"> Ir para o Conteúdo </a>
```

O mais importante nesta técnica é compreender que o skip link serve para pular alguma coisa. De nada adianta colocar um link para pular a navegação depois da navegação. É por isso que os skip links são geralmente os primeiros links da página. Mas nada impede que um skip link seja colocado no meio do conteúdo para pular, por exemplo, uma lista.

No site [acessodigital.net](http://acessodigital.net) podemos encontrar um exemplo bastante claro do uso de skip links. Neste site, este dispositivo está localizado no topo à direita, conforme a figura abaixo:

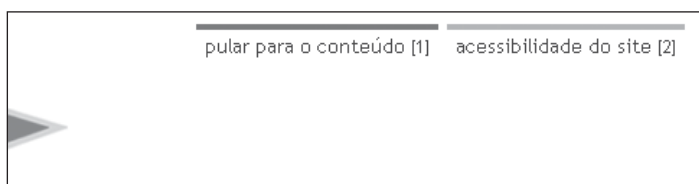


Figura 7. Exemplo de skip links

Notou o número ao lado do texto? Isto é uma indicação visual de que estes links podem ser acessados diretamente através de combinações de teclas (accesskey). Isto mostra que é perfeitamente possível combinar skip links com o atributo accesskey. Para o caso desta página em particular, no primeiro skip link seria definido da seguinte forma:

```
<a href="#conteudo" accesskey="1">pular para o  
conteúdo [1]</a>
```



A princípio, skip links eram sempre usados de forma visível, ou seja, os links sempre faziam parte da apresentação da página. Contudo, como os skip links acabam ocupando espaço na página, alguns desenvolvedores não gostavam de exibi-los. Foi assim que surgiu a técnica de esconder os skips links através de CSS, utilizando a regra `display:none`.

A idéia era, já que quem utiliza skip links são usuários de leitores de tela, então omitimos o skip link visualmente, mas ele permanece no código HTML para que o leitor de telas possa acessá-lo.

Contudo esta técnica não se mostrou muito boa por dois motivos. Primeiro porque um dos leitores de tela mais utilizado, o Jaws, estava omitindo a leitura de qualquer elemento do código que tivesse a declaração `display:none` em sua regra de apresentação, o que fazia com que o skip link não fosse lido.

Além disso, não são só usuários de programas leitores de tela que utilizam skip links. Na verdade, qualquer usuário que navega pelo teclado pode se beneficiar deste dispositivo.

Uma outra técnica desenvolvida para skip links é conhecida como método `off-left` e consiste em utilizar a seguinte regra CSS para o skip link:

```
.off-left{
  Position:absolute;
  Left:2000px;
}
```

Esta regra não impede que os navegadores e leitores de tela carreguem o link, porém ele também não vai aparecer na tela por que o link acaba sendo jogado para um ponto fora da tela. É como se a tela do computador fosse um palco e o skip link tivesse sido colocado nos bastidores.

O método Off-Left resolve o primeiro problema (o do Jaws não acessando o link), mas continua prejudicando a acessibilidade dos demais usuários de teclado.

Por fim, uma outra metodologia foi desenvolvida com a intenção de solucionar os dois problemas apontados. Este método se parece bastante com o Off-Left, mas com uma diferença: a medida que o usuário vai apertando TAB no teclado, as opções de skip links vão aparecendo na tela. Dessa forma, ora os links estão escondidos, ora eles estão na tela. Este método foi batizado de skip link off-screen.

Implementar este método é mais fácil do que parece. A primeira coisa a se fazer aqui é criar uma lista com todos os skip links. Algo como:

```
<ul class="off-left">
  <li><a href="#conteudo">Ir para conteúdo</a></li>
  ...
</ul>
```

O restante será controlado através de regras CSS:

1. Primeiro, a tag <ul> vai receber a classe off-left da técnica anterior para sair de cena;

2. Segundo, a tag <a> receberá uma regra sobre seu posicionamento que será absoluto. Esta regra é importante para que possamos ter um maior controle no posicionamento dos links.

```
a {
    position:absolute;
}
```

3. Depois, vamos criar para as tags <a>, uma pseudo-classe que irá descrever o comportamento de cada link quando receberem foco pelo teclado. Esta regra varia, de acordo com o navegador utilizado pelo usuário. Por exemplo, para funcionar com Internet Explorer, esta regra deve ser:

```
a:active{
    left: 20100px;
}
```

Para a maioria dos outros navegadores, a regra para fazer a mesma coisa é:

```
a:focus{
    left: 20100px;
}
```

Você também pode criar uma regra mista, combinan-

do a duas opções:

```
a:active, a:focus{  
    left: 20100px;  
}
```

Você pode encontrar esta técnica aplicada a vários sites na internet. Na imagem abaixo, por exemplo, esta técnica é aplicada ao site da Unilever. Observe que, através do CSS, você tem o controle total de onde e como os skip links vão aparecer.



Figura 8. Método Off-screen

Existe uma preocupação quanto a usabilidade do site que utiliza o método off-screen para controle de apresentação de skip links. Segundo alguns especialistas em usabilidade, não existe neste método clareza suficiente para que um visitante perceba visualmente que o site possui skip links e, por conta disso, o visitante acaba não utilizando os links.

Apesar disto, este método é bastante interessante e, sendo bem implementado e até combinado com outros dispositivos de acessibilidade, pode dar excelentes resultados.

### 3. Controle do tamanho da letra

Nem sempre o tamanho da letra utilizado no site é adequado para a visão de quem o visita. Por isso é muito interessante para a acessibilidade do site que este tamanho de letra seja configurável pelo visitante. Claro que sempre existe a possibilidade do visitante utilizar o controle do próprio navegador para realizar esta tarefa, mas se o site não tiver um layout adequado, esses controles podem acabar distorcendo a apresentação do site.

Uma forma melhor de controlar isto é fornecendo no próprio site a opção de aumentar ou diminuir o tamanho da letra. Este recurso é oferecido usando dois links ou duas imagens representando botões que, quando acionados, fazem uma chamada a uma função Javascript. Esta função percorre todo o documento e aumenta (ou diminui) o tamanho da letra dos campos informados.

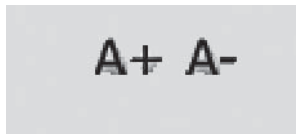


Figura 9. Exemplo de controle do tamanho de letra

A vantagem em utilizar esta técnica é que ela permite que desenvolvedor do site mantenha o controle sobre a apresentação. O desenvolvedor controla quais partes do layout

podem ter o texto ajustado e o quanto a letra pode aumentar ou diminuir.

A primeira coisa que fazemos no código Javascript é definir todos os tamanhos que poderão ser utilizados no site em um vetor. Desta forma, você pode controlar exatamente quais tamanhos de texto são permitidos.

```
var tamanhos = new Array( '9px', '10px', '11px',  
    '12px', '13px', '14px', '15px' );
```

Outra definição importante neste momento é o tamanho inicial da letra, o primeiro tamanho exibido na tela.

```
var tamanhoInicial = 2;
```

Note que a variável `tamanhoInicial` não é um valor pertencente ao vetor `tamanhos`, e sim um índice desse vetor. Isso quer dizer que, quando definimos `tamanhoInicial = 2`, na verdade queremos dizer que o primeiro tamanho de letra é 11px, já que no Javascript a primeira posição de um vetor é zero.

Iniciadas essas duas variáveis, o restante do código é uma função que recebe como parâmetros o id do elemento que terá o tamanho das letras alterado. Esta função também receberá um valor, que sempre será +1 ou -1, dependendo da escolha do visitante.

```
function mudaTamanho(idName, valor){  
    tamanho = tamanhoInicial + valor;  
    if (tamanho < 0){  
        tamanho = 0;  
    }  
    if (tamanho > 6){  
        tamanho = 6;  
    }  
    document.getElementById(idName).style.fontSize = tamanhos[tamanho];  
}
```

Nesta função, a variável responsável pela mudança do tamanho é tamanho, que inicialmente recebe a soma do tamanho inicial com o valor passado por parâmetro. Logo depois é feita uma verificação, para evitar que tamanho ultrapasse os valores permitidos. Em seguida, a variável tamanho é aplicada ao tamanho da letra do elemento passado por parâmetro (idName).

Para usar esta função, basta criar uma chamada onde os links de alteração de tamanho aparecem. Por exemplo, os links poderiam ser escritos desta forma:

```
<a href="javascript:mudaTamanho("texto",  
+1);">[+]</a>  
<a href="javascript:mudaTamanho("texto",  
-1);">[-]</a>
```

Contudo, existe um grave problema quanto ao uso de chamadas a funções Javascript dentro de links. O que acontece quando temos no código HTML algo do tipo `<a href="javascript:mudaTamanho("texto", +1);">[+]</a>`, e o browser não tiver suporte a javascript ativado?

Na verdade, não acontece nada! Nem o que se esperava que acontecesse, que neste caso, era aumentar a letra. O que ocorre nesta situação é que a falta de suporte a Javascript está impedindo o bom funcionamento da página da Web e bloqueando o acesso a algum recurso. Tal problema é conhecido como Javascript obstrutivo.

Nós podemos fazer algumas alterações na chamada da função javascript mudaTamanho() para permitir uso não obstrutivo desta função.

Uma forma de solucionar o problema é criar uma função que manipule eventos relacionados aos links que aumentam ou diminuem o tamanho da letra<sup>1</sup>. Desta forma, utilizamos a seguinte função Javascript:

```
function manipulaEvento(){
    var linkAumenta = document.
    getElementById("aumenta");
```

---

1 Bruno Torres chama esta manipulação de elementos HTML via Javascript de Camada de Comportamento. Tal camada é responsável por controlar eventos depois que o código HTML (ou XHTML ou XML) já tiver sido renderizado. Leia mais em: <http://www.obasicodaweb.com/padroes-web-desenvolvimento-em-camadas-camada-de-comportamento>



```
var linkDiminui = document.  
getElementById("diminui");  
  
linkAumenta.onclick = function(){  
    mudaTamanho("texto", +1);  
    return false;  
}  
  
linkDiminui.onclick = function(){  
    mudaTamanho("texto", -1);  
    return false;  
}
```

Esta função é capaz de detectar quando algum click de mouse é dado sobre um dos dois links, [+] e [-], que estão na página da Web. A função `manipulaEvento( )` dispensa a necessidade de uma chamada direta a função que controla a alteração do tamanho de letra. Desta forma, os links da página serão reescritos sem o uso de javascript:

```
<a href="erroJavascript.html" id="aumenta">[+]</  
a>  
<a href="erroJavascript.html" id="diminui">[-]</  
a>
```

Note que, se o browser não tiver suporte a Javascript, uma página chamada `erroJavascript.html` é carregada, procurando dar um tratamento adequado para o problema.

O próximo passo é garantir que o acesso aos links também possa ser feito via teclado. Para que você possa ter

uma idéia geral da solução adotada, a função `manipulaEvento()` foi escrita na íntegra, logo abaixo:

```
function manipulaEvento(){
    var linkAumenta = document.
    getElementById("aumenta");
    var linkDiminui = document.
    getElementById("diminui");

    linkAumenta.onclick = function(){
        mudaTamanho("texto", +1);
        return false;
    }
    linkDiminui.onclick = function(){
        mudaTamanho("texto", -1);
        return false;
    }
    linkAumenta.onkeypress = function(e){
        var keynum;
        if(window.event){ // para o IE
            keynum = window.event.keyCode;
        }

        else if(e.keyCode){ // Netscape/Firefox/Opera
            keynum = e.keyCode;
        }
        if (keynum == 13) {
            mudaTamanho2(`texto`, +1);
            return false;
        }
    }
    linkDiminui.onkeypress = function(e){
        var keynum;
        if(window.event){ // para o IE
```

```
        keynum = window.event.keyCode;
    }
    else if(e.keyCode){ // Netscape/Firefox/
Opera
        keynum = e.keyCode;
    }
    if (keynum == 13) {
        mudaTamanho2('texto', -1);
        return false;
    }
}
}
```

Para que esta técnica funcione, ainda falta ajustar um último detalhe: devemos executar a função de manipulação de evento assim que a página seja carregada. Desta forma, estaremos associando os links da página aos eventos implementados na função javascript.

```
<script type="text/javascript">
window.onload = manipulaEvento;
</script>
```

## 4. Problemas de acessibilidade com CAPTCHA

A palavra CAPTCHA é um acônimo da expressão "*Completely Automated Public Turing test to tell Computers and Humans Apart*" (Teste de Turing público completamente automatizado para diferenciar entre computadores e humanos).

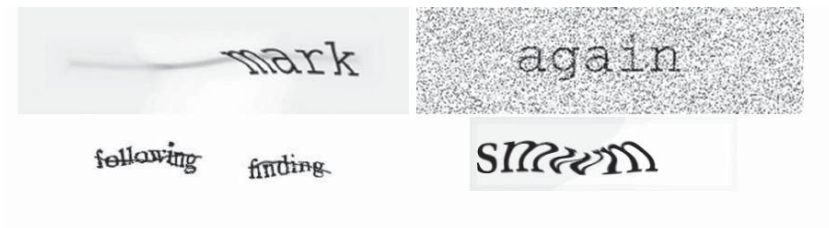


Figura 10. Exemplos de CAPTCHA

A motivação para o uso de CAPTCHA` s está na necessidade de segurança contra invasões de robôs. Por exemplo, empresas que oferecem e-mail gratuito na Web como o Google ou Yahoo, precisam garantir que quem está criando uma conta de e-mail é realmente um ser humano, e não um programa de computador.

Esta distinção entre seres humanos e robôs é feita através do teste de Turing, um teste capaz de distinguir entre homens e máquinas, baseado na capacidade cognitiva do ser humano.

A figura 10 mostra exemplos de um tipo comum de CAPTCHA, onde o usuário deve identificar as letras que são distorcidas em uma imagem.

Apesar de todos os esforços para se criar algoritmos de CAPTCHA cada vez mais difíceis de quebrar, já existem muitos casos de robôs que conseguiram responder o teste com sucesso. Este é o caso do algoritmo do CAPTCHA utilizado pelo Yahoo, o Gimpy, que foi quebrado por um grupo

de cientistas através de um método denominado por eles de EZ-Gimpy<sup>2</sup>.

Do ponto de vista da acessibilidade, a utilização do CAPTCHA causa uma série de problemas. Como a técnica é baseada em reconhecimento visual e, pela sua própria natureza, não permite que softwares leitores de tela interpretem seu conteúdo, o uso de CAPTCHA impossibilita o acesso de pessoas com deficiência visual ou visibilidade reduzida. O CAPTCHA cria também barreiras para outros grupos de pessoas que podem possuir inabilidades na aprendizagem que envolve interpretação de texto.

Dependendo do projeto do CAPTCHA aplicado, as falhas podem ser tão catastróficas que podem impedir o acesso mesmo de pessoas que não possuem nenhum tipo de deficiência. Por exemplo, o uso de '0' (zero) e 'o' (letra o) podem confundir o usuário.

Atualmente existe muita discussão sobre este assunto e há uma preocupação por parte de órgãos como o W3C, que procuram solucionar este problema. Alguns dispositivos de acessibilidade foram implementados como um recurso auxiliar. O próprio W3C publicou um documento<sup>3</sup> com propostas de solução, porém, até agora nenhuma delas completamente satisfatórias:

◆ **Uso de testes lógicos do tipo “Quanto é 1 + 1?”**

---

2. Veja em: <http://www.cs.sfu.ca/~mori/research/gimpy>

3 <http://www.w3.org/TR/turingtest/#solutions>

ou **“Qual é a cor do céu?”** - esta solução exige a criação manual destes testes, o que viola um dos princípios do CAPTCHA, a automatização. Além disso, algumas pessoas com deficiência cognitiva poderiam ter dificuldades em responder estes testes.

- ◆ **Uso de dados próprios do usuário, como número de cartão de crédito** - A maioria das pessoas não se sentiriam a vontade em fornecer dados pessoais para poder ingressar em um site desconhecido.
- ◆ **Trabalhar com operadores humanos** - eficiente, mas tem um custo muito alto. Além disso, nem todos os sites podem, por diversas razões, dispor deste tipo de serviço.
- ◆ **Confirmação através de senha enviada por SMS para o celular do usuário** - Esta é uma idéia interessante utilizada pelo Google em alguns casos. Contudo, entram aqui outras complicações relacionadas a acessibilidade de dispositivos móveis.
- ◆ **Saída sonora** - esta é a alternativa mais bem aceita atualmente. São arquivos de som gerados automaticamente e geralmente possuem distorções para dificultar a compreensão da mensagem por robôs. Mas existem algumas situações onde o uso deste dispositivo é ineficaz. Por exemplo, em ambientes de muito barulho, em computadores sem caixas de som ou sem plugins para seu correto funcionamento, ou utilização por pessoas surdas. Uma maneira de reduzir as deficiências deste método é disponibilizar o CAPTCHA por áudio junto com o CAPTCHA visual.

## 5. Breadcrumbs Trail

E lá vão eles, João e sua irmã Maria, sendo levados pela madrasta malvada para serem abandonados dentro da floresta. Mas João tinha um plano: para não perder o caminho de volta ele foi deixando pra trás migalhas de pão.

Se você conhece esta estória, então provavelmente você já sabe o que são *breadcrumbs trail*. Aliás, este dispositivo recebeu este nome justamente por causa deste conto de fadas.

*Breadcrumbs trail* (trilha de migalhas de pão) é o nome em inglês de uma técnica que permite ao usuário se localizar dentro de um site, mesmo se ele tenha ido parar em uma página interna deste site através de um sistema de busca, por exemplo.

O formato convencional de um *breadcrumb* é uma linha contendo links separados pelo símbolo >. Cada link indica um local por onde o usuário já passou ou também pode indicar uma página de um nível acima, como podemos ver na figura 11.



Figura 11. Exemplo de breadcrumbs

Todo *breadcrumb* é montado de forma a fornecer ao

usuário um caminho pelo qual ele pode voltar sem problemas, além de informar onde o usuário está.

Existem três tipos de *breadcrumbs*:

- *Breadcrumb* de localização
- *Breadcrumb* de caminho
- *Breadcrumb* de atributo

Nos *breadcrumbs* de localização, as migalhas são organizadas por hierarquia. Algo do tipo:

Página inicial > Seção > Subseção > **página atual**

A figura 12 é um exemplo de *breadcrumb* de localização aplicado em um site.

*Breadcrumbs* de caminho organizam as migalhas de acordo com o caminho tomado pelo usuário para chegar onde está. Como *breadcrumbs* de caminho depende da escolha de rota do usuário, essas migalhas são geradas dinamicamente através de scripts de programação.



Figura 12. Breadcrumb de localização

*Breadcrumbs* de atributo são muito comuns em sites de e-commerce, onde a maioria das páginas são geradas dinamicamente. Sua principal característica é que eles não representam toda a página, mas atributos dela. Eles mos-



tram informações de caminhos que podem ser seguidos para alcançar um determinado recurso do site.

categorias	Localização	Faixas de preços	Ou refinar por
1 GB (366)	São Paulo (400)	Até R\$40.00 (201)	Usado   Novo
2 GB (118)	Rio De Janeiro (77)	De R\$40.00 a R\$60.00 (132)	Melhores Vendedores
Até 512 MB (106)	Paraná (25)	De R\$60.00 a R\$100.00 (138)	Com MercadoPago
A partir de 4 GB (33)	Minas Gerais (25)	De R\$100.00 a R\$300.00... (153)	Atremate   Compre Já
	<a href="#">Ver todas</a>	Mais de R\$300.00 (20)	<a href="#">Ver todas</a>

Figura 13. Breadcrumb de atributo

## 6. Menu de Acessibilidade

O menu de acessibilidade é uma forma de aglomerar os demais dispositivos de acessibilidade em um único dispositivo. Funciona como um painel ou barra de acessibilidade.

Pensando somente na acessibilidade do site, este não é um ítem essencial, mas pode ajudar, dependendo do usuário que visita o site. Sua função é simplesmente organizar esteticamente recursos de acessibilidade, tais como: skip links, controle de tamanhos de letra, breadcrumbs, esquemas de cores, etc.

O site [Acessibilidade Brasil](http://www.acessobrasil.org.br/)<sup>4</sup> nos dá um exemplo de um menu de acessibilidade bastante simples, porém funcional. Ele aparece no topo de cada página como um painel de links, conforme você pode ver na figura 14. Um recurso interessante deste painel é que ele reconhece automaticamente o navegador utilizado pelo visitante e modifica a orientação de utilização das teclas de acesso.

4 <http://www.acessobrasil.org.br/>

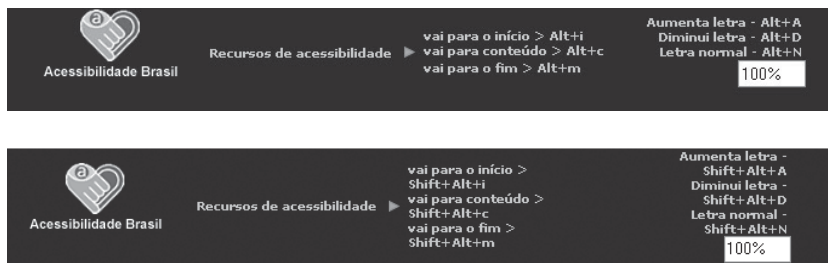


Figura 14. Exemplo de menu de acessibilidade

Já o site do Instituto Benjamin Constant<sup>5</sup> trabalha com um menu de atalhos de navegação estrategicamente localizado logo abaixo da identificação do site. Além do menu, no topo do site, ainda existem outros recursos importantes, como um campo de busca e um link para a página principal. Esses elementos são importantes não só para a acessibilidade, mas para a usabilidade do site em geral. Ainda no site do instituto, um destaque especial para os botões de controle de acessibilidade. Eles servem para controlar o tamanho das letras e o esquema de cor e contraste.

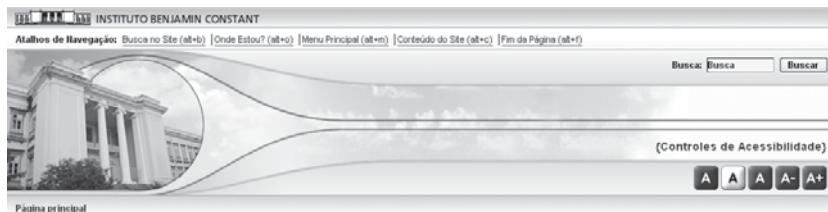


Figura 15. Menu de acessibilidade usando botões

5 <http://www.ibc.gov.br/>

Alguns projetistas preferem criar um menu de acessibilidade com um pouco mais de complexidade, incluindo a opção de exibir ou ocultar o menu, dependendo da escolha do usuário. Um exemplo é o menu de acessibilidade do site do Instituto Ethos<sup>6</sup>. A princípio, o menu fica oculto, mas quando o usuário clica na guia menu de acessibilidade, ele aparece no topo da página, deixando à mostra todos os atalhos e botões de acessibilidade.

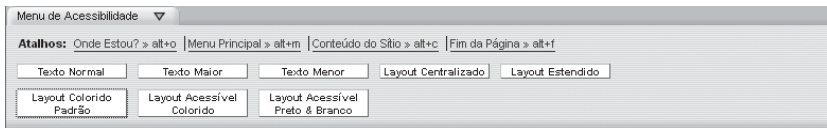
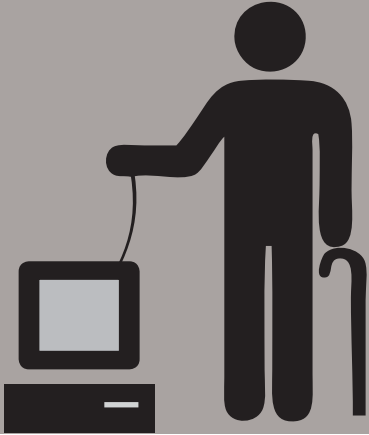


Figura 16. Menu de acessibilidade do Instituto Ethos

---

6 <http://www.ethos.org.br/>



# 5

## **Cinto de Utilidades!**

Este livro está longe de esgotar completamente toda esta discussão sobre acessibilidade a Web. Com certeza, este é um assunto que está amadurecendo e cada vez mais pessoas estão descobrindo o quanto a acessibilidade é importante, não só para deficientes físicos, mas para todo tipo de usuário que acessa a Web a partir dos mais variados dispositivos.

Este capítulo foi escrito especialmente para você, que tem realmente interesse em por em prática tudo que foi exposto até este momento. Aqui você encontrará listas de ferramentas para testar, validar e avaliar seus códigos HTML e CSS. Encontrará também uma lista de artigos importantíssimos, escritos por pessoas que realmente sabem do que estão falando. Aproveite e bom estudo!

## 1. Ferramentas de avaliação/validação de acessibilidade

Avaliadores de acessibilidade são ferramentas semi-automáticas que analisam se o código HTML está de acordo com as recomendações de acessibilidade.

- ◆ **Examinator** - além de avaliar a acessibilidade do site, ele fornece uma nota, de forma que o desenvolvedor possa examinar como a acessibilidade do site se encontra. [[www.acesso.unic.pt/webax/examinator.php](http://www.acesso.unic.pt/webax/examinator.php)]
- ◆ **Cynthia Says** [<http://www.cynthiasays.com/>]
- ◆ **Wave** [<http://wave.webaim.org/>]
- ◆ **A-prompt** [<http://aprompt.snow.utoronto.ca/>]
- ◆ **DaSilva** - Este é um examinador brasileiro, mantido pelo grupo Acessibilidade Brasil. No DaSilva a análise é feita usando as regras de acessibilidade do WCAG e E-MAG [<http://www.dasilva.org.br/>]

## 2. Softwares para avaliação de acessibilidade

- ◆ **TAW3** - Software que analisa a página de acordo com a WCAG. o TAW gera um relatório que mostra os problemas encontrados com sua respectiva descrição, o número da linha onde foi detectado o problema de acessibilidade, link para uma página que contém a explicação do problema e formas de resolvê-lo. [[www.tawdis.net/taw3/cms/en](http://www.tawdis.net/taw3/cms/en)]

- ◆ **ASES** - é um software brasileiro, criado pelo e-Gov para avaliar, simular e corrigir a acessibilidade de sites. Entre outras coisas, avalia acessibilidade, HTML e CSS. [[www.governoeletronico.gov.br/acoes-e-projetos/e-MAG/ases-avaliador-e-simulador-de-acessibilidade-sitios](http://www.governoeletronico.gov.br/acoes-e-projetos/e-MAG/ases-avaliador-e-simulador-de-acessibilidade-sitios)]
- ◆ **Barra de acessibilidade AIS** - mantida pela Vision Australia, é uma barra para ser instalada no Internet Explorer e pode fazer validações de acessibilidade, HTML e CSS e realiza outras tarefas. [[www.visionaustralia.org.au/info.aspx?page=614](http://www.visionaustralia.org.au/info.aspx?page=614)]

### 3. Ferramentas validadoras de HTML/CSS

As ferramentas de validação de HTML e CSS avaliam a sintaxe do código HTML e CSS. Existem várias opções de validadores, fique a vontade para escolher o que mais se adequar as suas necessidades.

#### - Validadores de sintaxe HTML/XHTML

- ◆ **W3C Markup Validation Service** - validador oficial do W3C para HTML e XHTML [<http://validator.w3.org/>]
- ◆ **WDG HTML Validator** [[www.htmlhelp.com/tools/validator/](http://www.htmlhelp.com/tools/validator/)]

#### -Validadores de CSS

- ◆ **W3C CSS Validation Service** - validador oficial do W3C para CSS. [<http://jigsaw.w3.org/css-validator/>]

## 4. Softwares leitores de tela

- ◆ **Jaws** [[www.freedomscientific.com/fs\\_downloads/jaws.asp](http://www.freedomscientific.com/fs_downloads/jaws.asp)]
- ◆ **Virtual Vision** - Nacional, fabricado pela Micropower. [[www.micropower.com.br/v3/pt/acessibilidade/index.php](http://www.micropower.com.br/v3/pt/acessibilidade/index.php)]
- ◆ **DosVox** - Gratuito e em português. Produzido pela UFRJ [[intervox.nce.ufrj.br/dosvox/](http://intervox.nce.ufrj.br/dosvox/)]
- ◆ **NVDA** - Leitor de telas gratuito e de código aberto para Windows. Uma excelente opção para uso e testes. [[www.nvda-project.org](http://www.nvda-project.org)]
- ◆ **Orca** - Leitor de tela para linux [[live.gnome.org/Orca](http://live.gnome.org/Orca)].

## 5. Complementos de acessibilidade para Firefox

- ◆ **Fangs Screen Reader Emulator** - renderiza o conteúdo de uma página simulando um leitor de tela. [[addons.mozilla.org/pt-BR/firefox/addon/402](http://addons.mozilla.org/pt-BR/firefox/addon/402)]
- ◆ **TAW3 with a click** - Permite verificar rapidamente como anda a acessibilidade do site. [[addons.mozilla.org/pt-BR/firefox/addon/1158](http://addons.mozilla.org/pt-BR/firefox/addon/1158)]
- ◆ **Web Developer** - Não é uma ferramenta específica de acessibilidade, mas faz validação pela ferramenta CinthiaSays. É muito para desenvolvimento Web em geral. [[addons.mozilla.org/pt-BR/firefox/addon/60](http://addons.mozilla.org/pt-BR/firefox/addon/60)]

## 6. Ferramentas que úteis para teste e desenvolvimento

- ♦ **Lynx Viewer** - simulador de um navegador em modo texto. [[www.delorie.com/web/lynxview.html](http://www.delorie.com/web/lynxview.html)]
- ♦ **Google Acessible Search** - uma busca do google que prioriza os sites acessíveis no resultado da busca. [<http://labs.google.com/accessible>]

## 7. Ítens de leitura Obrigatória

- ♦ **WCAG 1.0** - Não é uma leitura fácil, mas importantíssima se você quiser conhecer a fundo este assunto. Você pode ler, direto do W3C, a versão original em inglês [[www.w3.org/TR/WCAG10/](http://www.w3.org/TR/WCAG10/)], ou pode ler a tradução de Cláudia Dias [[www.geocities.com/claudiaad/acessibilidade\\_web.html](http://www.geocities.com/claudiaad/acessibilidade_web.html)].
- ♦ **E-MAG** - Acessibilidade do Governo Eletrônico, uma cartilha técnica [[www.inclusaodigital.gov.br/inclusao/arquivos/outros/documentos-gerais-referencias/emag-acessibilidade-de-governo-eletronico-cartilha-tecnica-v20.pdf](http://www.inclusaodigital.gov.br/inclusao/arquivos/outros/documentos-gerais-referencias/emag-acessibilidade-de-governo-eletronico-cartilha-tecnica-v20.pdf)]
- ♦ **Cartilha de codificação E-GOV** - uma cartilha de boas práticas em codificação XHTML, CSS e JavaScript [[www.governoeletronico.gov.br/biblioteca/arquivos/padroes-brasil-e-gov](http://www.governoeletronico.gov.br/biblioteca/arquivos/padroes-brasil-e-gov)]
- ♦ **WCAG Samurai** - Tradução de Maurício Samy Silva [[www.maujor.com/wcagsamurai/](http://www.maujor.com/wcagsamurai/)]

## 8. Ítens de leitura recomendada



- ◆ Acessibilidade web, Usabilidade, Teclado e Leitores de Tela - **Marcos Antônio de Queiroz** [[www.bengalalegal.com/noco.es.php](http://www.bengalalegal.com/noco.es.php)]
- ◆ Como testar a acessibilidade em Websites (parte 1) - **Horácio Soares** [[internativa.com.br/artigo\\_acessibilidade\\_03\\_06.html](http://internativa.com.br/artigo_acessibilidade_03_06.html)]
- ◆ Acessibilidade web: 7 mitos e um equívoco - **Lêda Spelta** [[acessodigital.net/art\\_acessibilidade-web-7-mitos-e-um-equivoco.html](http://acessodigital.net/art_acessibilidade-web-7-mitos-e-um-equivoco.html)]
- ◆ O selo não garante acessibilidade - **Horácio Soares** [[internativa.com.br/artigo\\_acessibilidade\\_06.html](http://internativa.com.br/artigo_acessibilidade_06.html)]
- ◆ Seu menu dropdown funciona com o teclado? - **Bruno Torres** [[brunotorres.net/seu-menu-dropdown-funciona-com-o-teclado](http://brunotorres.net/seu-menu-dropdown-funciona-com-o-teclado)]
- ◆ Formulários acessíveis à prova de spam - **tradução de Maurício Samy Silva** [[www.maujor.com/tutorial/spam-em-formularios.php](http://www.maujor.com/tutorial/spam-em-formularios.php)]
- ◆ Navegação Via Teclado - Teclas de Atalho (Jaws, IE e FF) - **Marcos Antonio de Queiroz** [[www.acessibilidadelegal.com/13-atalhos.php](http://www.acessibilidadelegal.com/13-atalhos.php)]
- ◆ Usabilidade de Interfaces e Arquitetura da Informação: Navegação Estrutural - **Felipe Memória** (o artigo trata sobre breadcrumbs trail) [[www.fmemoria.com.br/artigos/nav\\_estr.pdf](http://www.fmemoria.com.br/artigos/nav_estr.pdf)]
- ◆ Desenvolvendo AJAX acessível aos leitores de tela - **tradução de Maurício Samy Silva** [[www.maujor.com/tutorial/ajax-screen-readers.php](http://www.maujor.com/tutorial/ajax-screen-readers.php)]

# TÉCNICAS DE ACESSIBILIDADE

Acessibilidade é um tema ainda pouco explorado e de bibliografia escassa, apesar da sua inegável relevância. Este livro trata de um texto que, apesar de abordar questões técnicas, destinadas a um público especializado, cumpre sua missão com uma linguagem simples e acessível, como deve ser o tom de uma obra que pretende abordar tal tema. Nele, o leitor é conduzido desde aos conceitos básicos da acessibilidade, sua motivação e importância, até às técnicas e ferramentas avançadas para a sua utilização plena por especialistas, ou até mesmo por pessoas que não tenham tanta experiência em desenvolvimento de aplicações para a Web, mas que pretendam publicar algo, desde um simples blog até um site mais complexo.

Por todas essas questões, esta obra tem um mérito louvável. É o primeiro livro no cenário nacional sobre esse assunto e, como toda iniciativa pioneira, abre espaço para novas pesquisas e trabalhos no tema abordado. Pode ser adotado como livro texto para cursos de desenvolvimento de aplicações para a Web e, até mesmo, para cursos de graduação em Sistemas de Informação e Ciências da Computação.

**Prof. Marcus Braga**

Universidade Federal de Alagoas

Apoio:

**Banco do  
Nordeste**

