HTML マニュアル 開発環境構築からサイト構築実践まで HTML の基礎と応用

1.HTML の構成詳細

- ・ヘッダ内に記述可能な要素
- ヘッダ内で複数記述できる要素
- ・文章の更新情報を示す要素
- ・ブロックレベル要素
- ・見出し
- ・リスト
- ・表を構成する要素
- ・インライン要素
- ・特別な要素

2.ブラウザとレンタリングエンジン

3.開発環境

- ・サクラエディタのダウンロード
- ・Firefox、Firebug のインストール
- ・Web Developer のインストール

4.Firebug 操作方法

- ・コンソール画面
- ・HTML タブ
- ・接続タブ

5.Firebugを使用したデバッグ

6.Web Developer を使用したデバッグ

7.Color Zilla を使用した開発補助

1.HTML の構成詳細

HyperText Markup Language(ハイパーテキスト・マークアップ・ランゲージ、略称:HTML)は、ウェブ上のドキュメントを記述するためのマークアップ言語。マークアップ言語とは、文章の構造、段落などや見栄え、フォントサイズなどに関する指定を文章とともにテキストファイルに記述するための言語である。このため、データ記述言語に分類される。文章に対するそれらの指定をマークアップ(markup)と呼び、マークアップを記述するための文字列をタグ(tag)と呼ぶ。

HTML では、ウェブの基幹的役割を持つ技術の一つで、HTML でマークアップされたドキュメントはほかのドキュ メントへのハイパーリンクを設定できるハイパーテキストであり、画像・リスト・表などの高度な表現力を持たせる ことができる。

HTML では、まず文書型宣言が必要となる。HTML 4.01 Strict の文書型宣言は以下のようなものである。 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

次に <html> タグを明記する。ここから </html> が Web ページ全体になる。このタグのなかにページの情報を記述する。

<head>タグ

ページが読み込まれるときに一番最初に読み込まれるデータを記述する。ここには html ページの文字コード。スタイルシート(.css)と呼ばれるページの外観を共有できるファイルのパスの記述、また、Java Script と呼ばれるブラウザ上で実行されるインタプリタ型のプログラミング言語が記述される。Web ページを実際に開いたとき、ブラウザはまずこのヘッダから読み込み、次に<body>タグ以降を読み込む。

<body>タグ

ここが Web ページでの本文になる。Html 文章は大きく分けて<head>タグのくくりと<body>タグのくくりに分け られる。ここでは文字や画像、映像(Flash など)、音楽(midi,mp3 など)といったマルチメディアコンテンツの 表示や Java Script の処理対象となるデータを記述する。Html 文章はフォルダの階層構造のような入れ子 の構造になっているので他のプログラミング言語と比べ読みやすく記述しやすくなっている。

```
例) サンプルの Web ページ
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html lang="ja">
 <head>
 <meta http-equiv="content-type" content="text/html; charset=UTF-8">
 k rev="stylesheet" href="スタイルシートの相対パス" type="text/css" />
 <script type="text/javascript" src="Java Script ファイルの相対パス"></script>
 <script type="text/javascript"><!--
   alert("テスト");
 //--></script>
 <title> タイトル </title>
 </head>
 <body>
    本文
 </bodv>
</html>
```

> ヘッダ内に記述可能な要素

title

タイトル(document title)を指定する。主なウェブブラウザ実装系では、タイトルバーに表示される。HTML 文書 で唯一必須かつ省略のできない要素。

base

相対パスの基準 URL (document base URI) を href 属性で指定する。

ヘッダ内で複数記述できる要素 (repeatable head elements)

link

自分自身と href 属性で指定したファイルとの関係を rel 属性で定義する。rel 属性は HTML4.01 で有効なもの には alternate, stylesheet, start, next, prev, contents, index, glossary, copyright, chapter, section, subsection, appendix, help, bookmark があり、他に使われるものとして、shortcut icon がある。

meta

文書の情報 (generic metainformation)を定義する。

object

埋め込みオブジェクト (generic embedded object) であることを示す。大抵の場合はインライン要素として使用 する。data 属性に URI、type 属性に MIME タイプ(image/gif 等)を指定することで FLASH や音楽など様々な種 類のオブジェクトを出力する事ができる。内容には代替(オブジェクト要素も可)を記述する。例えば一番外から オブジェクト要素(FLASH)、オブジェクト要素(画像)、説明文、のように入れ子にすると FLASH の利用できない 環境では画像が、FLASH と画像の利用できない環境では説明文が出力される。これは img 要素等の alt 属性 と比べると高性能な代替システムであるが、これに対応しないウェブブラウザも少なくない。

param

プロパティ値の設定(named property value)を行う。object 要素で使用するメディアに対しての初期値を設定するために使用する。内容は空で終了タグは存在しない。

style

スタイル情報(style info)を記述する。type 属性の記述が必要(text/css 等)。

script

スクリプト(script statements)を記述する。

type 属性(text/javascript など)とmeta 要素でスクリプトタイプの宣言が必要。

> 文章の更新情報を示す要素

ins

追加された文書(inserted text)であることを示す。 datetime 属性に追加日時を記載できる。

del

削除された文書(deleted text)であることを示す。 datetime 属性に削除日時を記載できる。

ブロックレベル要素

ブロックレベル要素は見出し、段落など文書を構成する基本要素となるものです。ブロックレベル要素の内容モ デルには、別のブロック要素やインライン要素を含むことができますが、逆にインライン要素の中にブロックレベ ル要素を置くことはできません。

р

段落(paragraph)を構成する。

blockquote

引用文(long quotation)であることを示す。

ウェブブラウザによってはインデントして表示することがあるが、この要素はあくまで引用目的に用いるべきで あり、単にインデントするために用いるのは好ましくない。また、内容にクォーテーションマーク(「″」や「'」)を 用いるのは避けるほうが良い。cite 属性で引用元・出典を明示することができる。

pre

整形済みテキスト(preformatted text)の節であることを示す。

内容にはインライン要素を含める事もできるが img、object、big、small、sub、sup 要素は使用できない。 ウェブブラウザのサイズによる折り返しがされなくなるため、リキッドレイアウトを行うには不向き。

div

ブロックレベルでのスタイルコンテナ。

この要素自体に意味は無い。文書の構造を示すときや、既存の要素には含められない物の代用要素として使用する。

noscript

スクリプトが動作しない環境での代替コンテンツ(alternate content container for non script-based rendering) であることを示す。スクリプトが動作する環境では無視される。代替となりうるコンテンツを示す要素である。

hr

水平線(horizontal rule)を示す。内容は空で終了タグは存在しない。

address

著者の情報・連絡方法(information on author)を示す。メールアドレスを記述するのが一般的。ただし、address 要素内のメールアドレスがメールアドレス検索ロボットの標的になりスパムメールなどの被害に遭う可能性も ある。

form

コントロール要素を纏めた入れ物であることを示す(interactive form)。action 属性が必須で、入力されたデータの処理を行う URI を指定する。

fieldset

フォームコントロールのグループ(form control group)を示す。 関連したフォームコントロールを記述することで利便性を高めることが出来る。

legend

fieldset 要素の表題(fieldset legend)を示す。

▶ 見出し

h1, h2, h3, h4, h5, h6

文章の見出し(heading)を示す。h1 が最上位の見出しで、h6 が最下位の見出し。

> リスト

ul

順序なしリスト(unordered list)を示す。一つ以上の li 要素を含む必要がある。

ol

順序付きリスト(ordered list)を示す。一つ以上の li 要素を含む必要がある。

li

リストの項目(list items)を示す。ブロック要素を含められるため、入れ子のリストを作成することもできる。

dl

定義のリスト(definition list)であることを示す。dt 要素か dd 要素を1つ以上含む必要がある。

dt

定義する用語(definition term)であることを示す。

dd

定義の説明(definition description)であることを示す。

> 表を構成する要素

表形式のデータを表す要素群。その特性からレイアウトに使用されることがあるが間違いである。また正しくマー クアップする為には(他の要素に比べ)多少の知識を要する。これは音声ブラウザの挙動など、アクセシビリティを 考慮した場合に制約が多いため。

table

表(table)を構成する要素の大枠を示す。summary 属性に表の要約・解説を記述する。 内容の要素には細かく順番が規定されていて以下の通りである。 0個か1個の caption 要素、0個以上の colgroup 要素と col 要素、0個以上の thead 要素、0個以上の tfoot 要素、1個以上の tbody 要素。

caption

表題(table caption)を示す。table 要素の直下に記述する。

thead, tfood, tbody

それぞれヘッダ(table header)、フッタ(table footer)、本体(table body)で、表内での節(table section)を示す。内容は tr 要素のみで、1 つ以上の記述が必要である。また、各要素内においての列の数は等しくなくてはならない。

colgroup

構造的な列のグループ(table column group)を示す。 グループ化させるには 2 つの方法があり、span 属性で列数を指定する方法か col 属性を列数分記述する方 法である。

col

列のグループ(table column)を示す。

colgroup 要素と違い、構造的なグループを示さず、スタイルの指定を主としたグループ化を行う。span 属性で グループ化する列数を指定する。内容は空で終了タグは存在しない。

tr

表内の行(table row)を示す。内容に1つ以上のth要素かtd要素が必要。

th, td

それぞれ、ヘッダのセル(table header cell)、データのセル(table data cell)を示す。視覚系でないユーザーエー ジェントに対応する為にいくつかの整形方法がある。

▶ インライン要素

インライン要素は、主としてブロックレベル要素の内容として用いられるもので、文書の構造を構成するというより、 ブロックレベル要素内の特定の部分になんらかの役割や機能を持たせる要素です。たとえば、ある語句に対して ハイパーリンク機能を与えるアンカー要素、特定の語句を強調する要素などです。インライン要素は、その内容 に文字データあるいは他のインライン要素を持つことができますが、ブロックレベル要素を含むことはできません。

✓ 物理要素(fontstyle)

見た目を定義する要素。スタイルシートで代替が可能。

i 文字を斜体(italic)にする。

b

文字を太字(bold)にする。

u

文字に下線(underline)を引く。

s (strike)

文字に中央線を引く。打ち消し線または、訂正線。

big

文字を大きく(large)する。

small

文字を小さく(small)する。

✓ 論理要素(phrase)

論理的な意味を表現する要素。

em

強調(Indicates emphasis)する。

strong

強く強調(Indicates stronger emphasis)する。

dfn

定義された用語(defining)を示す。

code

プログラムなどのソースコード(computer code)を示す。

samp

プログラムなどによる出力のサンプル(sample output from programs, scripts, etc.)を示す。

kbd

ユーザが入力する文字(text to be entered by the user)を示す。 アプリケーションのチュートリアルなどでユーザが自由に入力できる箇所であることを示す際などに用いる。

var

プログラムなどに用いる変数(variable or program argument)を示す。

cite

出典(citation or a reference)を示す。

addr

略語(abbreviated form)を示す。 title 属性に略さない状態の語を明示することができる。

acronym

頭字語(acronym)を示す。 abbr 要素と同様に、title 属性に略さない状態の語を明示することができる。

▶ 特別な要素

а

アンカー(anchor)であることを示す。href 属性にリンク先 URI を指定しハイパーリンクを作成する。内容にはリ ンク先の概要を表記する。内容だけを見てリンク先が判断できることが望ましく、「ここをクリック」等は使うべき でないとされる。内容が冗長になる場合は title 属性で説明を付加することができる。accesskey 属性でショー トカットキーを設定することができ、ユーザビリティの向上が期待できる。

img

埋め込み画像(Embedded image)であることを示す。src 属性に URI を指定し画像を表示させる。内容は空で終 了タグは存在しない。alt 属性で画像の説明を記述することが必要である。これは画像に対応できないユーザ ーのため。単にレイアウト用の画像(スペーサー画像等)の場合には alt は空にする。また「画像が表示できま せん」や「画像」と言った代替テキストはユーザにとっては有用でなく、「犬の写真」など簡素な説明や「飼い犬 のポチが…」など情景が把握できる説明が有用である。説明が長文になる場合 longdesc 属性に URI を指定し、 説明の文書を示す事もできる。他のドメインの画像を指定することも可能であるため、著作権の観点から問題 視されることもある。

canvas

解像度に依存したビットマップキャンバスを表示。グラフやゲーム用の表示画像を描画することができる。 HTML 5 で導入予定であり、Internet Explorer 以外の主要ブラウザではすでに実装済み。Internet Explorer 向 けに、VMLを使い、canvas タグを実現するライブラリがある。

2.ブラウザとレンタリングエンジン

ウェブ開発にあたり、特に注意しなければいけないことはブラウザの種類です。2009 年 4 月現在、主なブラウザ のレンタリングエンジンは、Trident(トライデント)と呼ばれる、Windows 版 Internet Explorer に搭載されている HTML レンダリングエンジンと、Gecko(ゲッコー)と呼ばれる、Netscape シリーズ 6 以降および Mozilla ソフトウェア のために開発されたオープンソースの HTML レンダリングエンジン、WebKit(ウェブキット)と呼ばれる、アップルが 中心となって開発されているオープンソースの HTML レンダリングエンジン、Presto(プレスト)と呼ばれる、オペラ・ ソフトウェア (Opera Software ASA) 制作の HTML レンダリングエンジンの4種類に分けられます。実は、ブラウザ のレンタリングエンジンには多少の違いがあり、HTML ファイルを開くと CSS のデザインや JavaScript の動作で若 干の違いが現れてしまいます。

対処方法としては、JavaScript でブラウザを判別して、それぞれのブラウザ別に処理を分岐させて処理を割り当 てて対応させるやりかたと、クライアントから見れるブラウザを限定するように仕様をつくるなどの対処方法がありま す。クライアントで現在一番使われているブラウザが Microsoft の Internet Explorer なのでそれで見れれば他は動 作保障はしないという手もありますが、Mac や Linux では Internet Explorer がないので見れなくなってしまいます。 大抵の場合は目的にあった仕様をしっかりとつくり、お客様と交渉しながらターゲットとなるブラウザの種類などを決 めていきます。

レンタリングエンジン	Trident	Gecko	WebKit	Presto
プラットフォーム	Windows 系 OS	Windows 系 OS UNIX 互換 OS Mac OS X	Windows 系 OS Mac OS X	Windows 系 OS UNIX 互換 OS Mac OS X
			I Phone	防市电品、クーム機
ブラウザ	Internet Explorer	Fire Fox	Safari,Google Chrome	Opera
シェア(2008 年 12 月)	68.15%	21.34%	8.97%	0.71%

それぞれのレンタリングエンジンに対する統一性は現在のところ、相違がありますが、将来的には W3C (World Wide Web Consortium) と呼ばれる各種技術の標準化を推進する為に設立された標準化団体、非営利団体により標準化が進められています。またその他では、ウェブスタンダードプロジェクト (WaSP) という、ウェブサイトをウェブ標準に則って制作することを推進する団体により、標準化テストがつくられています。スタイルシート標準化テストを Acid2 とし、JavaScript による標準化テストを Acid3 として現在使っているブラウザがどの程度まで、標準化にそっているかを確認することができます。

3.開発環境インストール

HTML でのデバッグ環境としては、代表的なものに Firefox の Firebug などがあります。また、Web 開発では動作の軽いテキストエディタをつかって開発することがほとんどですので今回は代表的なテキストエディタでフリーソフトのサクラエディタをつかって開発を行います。



上記のアドレスからサクラエディタをダウンロードしてインストールします。 サクラエディタはさまざまなプログラミング言語に対応しています。主にプログラミング言語の命令の色づけなどが できます。



http://mozilla.jp/firefox/

今回は開発環境として、Firebug を使用してみたいとおもいます。Firebug は JavaScript のデバッグ環境なので Web 開発には役立ちます。ダウンロードしてインストールします。 起動したらツール->アドオンを選択します。



アドオンを入手のアイコンをクリックし、テキストボックスに Firebug と打ち込んでエンターキーを押します。検索結果が表示されたら、Firefox に追加ボタンをクリックします。



Firefoxを再起動ボタンをクリックし、再起動します。これでインストールは完了です。



から WebDeveloper をダウンロードして Firebug と同じようにインストールします。

😻 Mozilla Firefox スタートページ - Mozilla Firefox	
ファイル(E) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T)) ヘルプ(日)
🕜 🕞 - C 🗶 🏠 http://www.google.co.jp/fire	efox?client=f 🏠 🔹 🔽 Google 🔎
	<u> </u>
Firefox Start	
Filelox Start	
Google	
	検索オブション 表示設定
 ● ワェフ全体から検索 ● 日本語のペーシを検索 Google 検索 	
- 🚀 調査 編集 ⊨ body < html	Q. 🛛 🔀
コンソール HTML CSS スクリプト DOM 接続 オプション・	スタイル レイアウト DOM オプション・
	body { firefox?:official (1 行日) bookyround: #EEEEEE ur [(/impros
theady theady theady theady theady theady theady	/firefox/gradsprite.png) repeat-x
	color: #333333;
	margin: 3px 8px 2em; }
	body, td, a, firefox?:official (1 行日)
	p, .h (font-family: arial,sans-serif;
	}
完了	* //

ブラウザにある、右下の虫(bug)のようなアイコンをクリックするとなにやらでてきます。 これが Firebug です。



右クリックで Firebug を新しいウインドウで開くこともできます。

4.Firebug 操作方法

実際のプログラミングで一番大切なのは実際にコードを書いて動かし、エラーが出たらなぜその箇所でエラーがで るのか考え、修正しながらこまめに書いていくことです。テキストを読んでやるよりは、実際にコードを書いてみて、 エラーが出たらテキスト等で調べる方法が覚えがはやく上達もします。

今回はサクラエディタを使用して JavaScript を書いて Firebug でデバッグをやっていきたいと思います。

😻 Mozilla Firef	iox スタートページ	- Firebug			- D ×
ファイル(<u>E</u>) 表	示 ヘルプ				
調査消去	プロファイル			Q,	
	HTML CSS	スクリプト DOM	接枝		オプション・
	コンソール	パネルは停止し	ています		
	このページで各 ると Firefox の	機能を使用すか停止 動作を低下させる事()	するか決める事ができます。こ こ注意してください。	れらの機能を使用す	ţ
	צכב 🗹	ソール コンソールを	E利用できます。	Disabled Always	-
	☑ スク!	リプト JavaScript	:デバッガを利用できます。	Disabled Always	
	☑ 接続	ミニー ネットワーク	モニターを利用できます。	Disabled Always	
	-	www.google.co.jp (*	選択された機能を有功にします	ġ	

コンソール、スクリプト、接続にチェックを入れ、xxxxxx で選択された機能を有効にしますをクリック。これで機能を使えるようになります。

🥴 Mozilla Fire	fox スタート	ページー	- Firebug					
ファイル(<u>E</u>) 🧋	長示 ヘル	プ						
調査 消去	プロファイノ	Ψ				Q,		
コンソールマ	HTML	CSS	スクリプト	DOM	接载		オプション	/*
>>>								

コンソールタブをクリックするとコンソール画面が表示されます。ここでは現在開いているウェブページに JavaScript

のエラーがあるとその箇所を表示してくれます。また、JavaScript をここから打ち込んで JavaScript を単体でテスト 実行させることもできます。

~

>>> alert("テスト");

ためしに、アラートがでる JavaScript を打ち込んでみます。 コードを入力してエンター alert("テスト");

http://ww	http://www.google.co.jp のページから: 🛛 🔀						
	ታスト						
	ОК						

このように単体で実行できます。

🧐 Mozilla Firefox スタートページ - Firebug	
ファイル(E) 表示 ヘルプ	
調査 編集 I body < html	Q,
コンソール HTML CSS スケリプト DOM 接続 オプション	スタイル レイアウト DOM オプション・
<pre> <html> <pre></pre></html></pre>	<pre>body { firefox?official (1 ff E) background: #FFFFFF url(/images /firefox/gradsprite.png) repeat-x soroll 0 -95px; color: #333333; margin: 3px 8px 2em; } body, td, a, firefox?:official (1 ff E) p, .h { font-family: arial,sans-serif; }</pre>

現在開いているウェブページのソースコードを階層構造で見やすく表示してくれます。また、タグやパスの色付けも行ってくれます。右のスタイルタブは現在のページで使用しているスタイルシート(CSS)を表示してくれます。





また、調査や編集といったタブをクリックすることによりキャッシュ(一時的に保存されたデータ)を書き換えてその場 でウェブページを書き換えることができます。これはキャッシュを書き換えるので実際のファイルには影響はでません

🥴 Mozilla Firefox スタートペー	-ジー Firebug		
ファイル(E) 表示 ヘルプ			
調査 消去 すべて HT	ML CSS JS XHR 画像	Flash 🔍	
コンソール HTML OSS	スクリプト DOM 接続、		オプション・
E GET firefox?c 200 OK	google.co.jp 4 KB	147n	IS
🗄 GET gradspri 200 OK	google.co.jp 376 B		54ms
GET spraycal 200 OK	google.co.jp 3 KB		38ms
GET sprite.pr 304 No	t Modified google.co.jp 10 KB		75ms
GET 0d72W4 304 No	t Modified google.co.jp 6 KB		37ms
ヘッダー レスポンス	Cache		
Last Modified Fr i Apr Last Fetched Fr i Apr Expires Thu Apr Data Size 5464 Fetch Count 73 Device disk	10 2009 14:45:08 GMT+0900 10 2009 14:45:08 GMT+0900 08 2010 08:59:59 GMT+0900		
5 個のリクエスト	21 KB	(15 KB キャッシュから)	273ms

コンテンツの読み込み時間とヘッダー、レスポンスといった情報が得られます。ウェブページの表示を高速させたいときなどに参考になります。

5.Firebugを使用したデバッグ

ここまでは、一般的な Firebug の操作をみてきました。ここからは、Firebug を使用したデバッグのしかたを紹介していきたいと思います。



スクリプトタグを開き、ソースコードを表示します。Firebug では、eclipse と同じようにブレイクポイントを設定し、デ バッグを行うことができます。<script>タグのある JavaScript の範囲内でのデバッグツールなので、あたりまえです が<script>タグで囲まれた範囲内(もしくは、js ファイルのなか)でブレイクポイントを設定してください。なを、外部js ファイルも対象ファイル名を変えるだけでデバッグできます(対象ファイル選択はヘルプの下の段にあります)。まず、 ブレイクポイントを設定したい行をクリックし true または、true になるような条件式を設定します。

😻 Fire	:bug())使い方 - Firebug				
ファイル	L(E)	表示 ヘルプ				
調査	al	左右に並べて表示			2	
עכב	- Jb	更新(<u>R</u>) Ctrl+R	▼ DOM 接枝	監視 スタック	ブレークポイント	オプション・
23 24 25 26	<d <d <s< th=""><th>文字サイズ ・ オプション ・</th><th>▲ ipt" src="http ba3/" accesske</th><th>vndefined google_ad_wi</th><th>0319155421.h dth = 120;</th><th>tml (35 行目) 🕎</th></s<></d </d 	文字サイズ ・ オプション ・	▲ ipt" src="http ba3/" accesske	vndefined google_ad_wi	0319155421.h dth = 120;	tml (35 行目) 🕎
27 28 29	<s <b< th=""><th>エディタで開く ▶</th><th>ipt" src="http</th><th></th><th></th><th></th></b<></s 	エディタで開く ▶	ipt" src="http			
30	<d <s< th=""><th>iv id="ad_code3"> cript type="text/javascr</th><th>ipt"><!--</th--><th></th><th></th><th></th></th></s<></d 	iv id="ad_code3"> cript type="text/javascr	ipt"> </th <th></th> <th></th> <th></th>			
32 33 34	go // go	ogle_ad_client = "pub-39 120x600, 作成済み 08/01/ ogle_ad_slot = "98479080	09642330912231 10 59";			
35 36 37	g0 g0 //	ogle_ad_width = 120; ogle_ad_height = 600; >				
38	<s sr</s 	cript type="text/javascr c="http://pagead2.google	ipt" syndication.co 🔻			
<u> </u>						

すると、?が行頭につくので、そこがブレイクポイントとなります。ブレイクポイントを反映させるために、表示->更新を選択します。

😻 Fireb	ougの使い方 - Firebug	
ファイル((E) 表示 ヘルプ	
調査	all • 0319155421.html 0319155421.html 🌔	
יעכב	-ル HTML CSS スクリプト → DOM 接枝	監視 スタック ブレークポイント オブション・
4	(meta nity equive content type content-	🔽 0319155421 ht保引9155421.html (35 行日) 👩
с а	(META name= generator content="http://w	woowle ad width = 120
7	<pre><milt <title="" content="filebogo/" hame="keywords">Firebugの使い方</milt></pre>	800816_80_WIGTH - 1209
8	k rel="stylesheet" href="http://7go.	
9	k rel="alternate" type="application/	
10	k rel="alternate" type="application/	
11	<link container"="" href="http://7go.biz/mo</td><td></td></tr><tr><th>15</th><td></td><td></td></tr><tr><th>16</th><td><script type=" javascript"="" next"="" prev"="" rel="start" src="http</td><td></td></tr><tr><th>17</th><td></head></td><td></td></tr><tr><th>18</th><th><body></th><th></th></tr><tr><th>19</th><th><div id=" text=""/>	
20	<divalign="center"></divalign="center">	
•	··· · · · · · · · · · · · · · · · · ·	
<u> </u>		

すると自動的にブレイクポイントで止まります。ブレイクポイントで止まると

、

、

、

、

、

のほうでも設定したブレイクポイントで読み込みが停止します。

```
Continue
```

次のブレイクポイントが見つかるまでプログラムが実行されます。ブレイクポイントが見つからない場合は最後 までプログラムを実行します。

Step Over

コードを一行づつ実行する機能です。関数/メソッドがあった場合は以降スキップします。

```
function a(){
var c = 1;
debugger;←中断A
c = b(c);←スキップ
alert(c);←実行再開
}
```

```
function b(s){
var d = s + 1;
return d;
}
```

実行すると「中断A」で停止します。その後 Step Over をクリックすると c = b(c); は処理されず、 alert(c); か ら実行再開されます。

💿 Step Into

コードを一行づつ実行する機能です。関数/メソッドがあった場合はその中にはいって処理を実行します。

```
function a(){
var c = 1;
debugger;←中断A
c = b(c);←実行再開
alert(c);
}
```

```
function b(s){
var d = s + 1;
return d;
}
```

実行すると「中断A」で停止します。その後 Step Into をクリックすると c = b(c); から実行再開されます。



コードの実行を再開し、関数呼び出しの内側から呼び出し元の関数に戻り実行を中断します。

```
function a(){
var c = 1;
c = b(c);一中断B
alert(c);
}
function b(s){
var d = s + 1;
debugger;一中断A
return d;
```

}

実行すると「中断A」で停止します。その後 Step Out をクリックすると「中断B」で停止します。

😻 Fireb	ugの使い方 - Firebug		- D ×
ファイル(④ 表示 ヘルプ		
調査	all + 0319155421.html		
יעכב	-ル HTML CSS スクリプト ▼ DOM 接枝	監視 スタック ブレークポイント	オブション・
1	html PUBLIC "-//W3C//DTD XHTML</th <th>新しい監視点</th> <th></th>	新しい監視点	
2	<html content="</th" content-type"="" xmlns="http://www.w3.org/1999/xhtm</th><th>woowle ad beight and</th><th></th></tr><tr><th>3</th><th><head></th><th>80081e_ad_ne18nt 280</th><th></th></tr><tr><th>4</th><th><meta http-equiv="><th>google_ad_width 336</th><th></th></html>	google_ad_width 336	
5	<meta name="generator" content="http://w</th> <th>«oo«le ad slot</th> <th></th>	«oo«le ad slot	
6	<META name="Keywords" content="Firebugの</th> <th></th> <th></th>		
7	<title>Firebugの使い方</title>		
8	<link rel="stylesheet" href="http://7go.</th> <th></th> <th></th>		
9	k rel="alternate" type="application/		
10	k rel="alternate" type="application/		
11	<link rel="start" href="http://7go.biz/m</th> <th></th> <th></th>		
12	<link rel="prev" href="http://7go.biz/mo</th> <th></th> <th></th>		
13			
14	<link rel="next" href="http://7go.biz/mo</th> <th></th> <th></th>		
15	-		
16	<pre><script type="text/javascript" src="http</pre></pre>		
17			
4			

その他の機能として、監視、スタックがあります。監視は、JavaScript 実行後の変数になにが値としてはいっているかをチェックできます。スタック(データを後入れ先出しで保持する構造体)はスタックトレースがおこなえます。ブレイクポイントとあわせてつかってみてください。

6.Web Developer を使用したデバッグ

😻 Webdeveloperのダウンロード:ミナトラボ - Mozilla I	Firefox	_ 🗆 🗙					
ファイル(E) 編集(E) 表示(V) 履歴(S) ブック	ウマーク(18) ツール(17) ヘルプ(14)						
🔇 💽 🗸 🔥 🗋 http://lab.tubonotubo.jp/tools/webdeveloper/index.html 👷 🔹 💽 Google 🔎							
🥥 無効化・ 💍 Cookie・ 🔤 CSS・ 📰 フォーム	ム・ 🔳 画像・ 🕕 情報・ 🐑 その他・ 🌽 枠表示・ 💱 サイズ変更・ 🧬 ツール・ ⊵ ソーン	ス・ 🤌 設定					
キャッシュを無効にする(D)							
 Javaを無効にする(J)							
JavaScriptを無効にする(S) ・	全てのJavaScript(<u>A</u>)						
Meta要素をによるリダイレクトを無効にする(R)	✓ Strict警告(S)						
 最小フォントサイズを無効する(E) 色表示を無効にする(P) ポップアップブロックを無効にする(B) プロキシを無効にする(D) リファラー送信を無効にする(E) 	こめにウェブ開発に役立つツールを開発し配布・掲載しています。どうぞご自由にお使いください。 エンコーダ バスワード生成 数値文字参照変換 GoogeサイトマップXML作成 ■	NEW					

インストールするとアドレスバーの下に Web Developer Toolber が表示されます。ツールバーの表示/非表示の切り替えはメニューから表示->ツールバーで選択できます。

このアドオンを追加するとページに関するさまざまな情報をわかりやすく解説してくれます。各機能は主にブラウザの機能動作の ON/OFF からスタイルシート適用範囲の表示とさまざまです。

- 以下、代表的な機能
 - ▶ クッキー、Java、JavaScript、CSS などを無効にする
 - ▶ CSS をその場で編集する
 - > フォームの各属性値を表示する
 - ▶ 画像を非表示にする
 - ▶ alt 属性のない、あるいは alt 属性値が空の画像を枠で囲む
 - ▶ リンクのパスを表示する
 - ▶ ブラウザの画面内を拡大あるいは縮小する
 - ▶ テーブルのセルを表示する
 - ▶ ブラウザのウィンドウおよび表示領域の幅と高さを表示する
 - ▶ HTML および CSS の構文をチェックする
 - > W3C/WCAG 1.0 あるいは米国リハビリテーション法 508 条でアクセシビリティをチェックする
 - ▶ Web ページのソースコードを表示する



数ある機能のなかで一番便利なのが、枠表示機能です。これは、テーブルやブロックレベル、h要素、フレームなどの範囲を色がついた枠でかこってくれます。この機能とあわせて Firebug で CSS の値を変更してみてください。枠がその変更のつど適用されると思います。

これでいちいちスタイルシートを変更、保存、再表示という手間が省けます。

7.Color Zilla を使用した開発補助



http://www.colorzilla.com/firefox/

からアドオンをインストールします。

ColorZilla はブラウザのなかのマウスカーソルにある位置の色情報を HTML で扱えるものに変換して表示してくれ ます。Web 開発などでは、使いたい色をコードから調べるのが億劫な場合があるので画面から直接色情報を得ら れる ColorZilla はスタイルシートの編集などにとても便利です。

