

アクセシビリティに対応する為のスタイルシートを使ったWebページレイアウトについての考察

近藤 研二

倉敷芸術科学大学芸術学部

(2005年9月30日 受理)

1. はじめに

近年、高齢者や障害者など心身の機能に制約がある人でも、年齢的・身体的条件に関わらず、ウェブで提供されている情報にアクセスし利用できることを実現する「ウェブアクセシビリティ」が注目されている。「ウェブアクセシビリティ」を実現するためのウェブページを制作するためには、カスケーディング・スタイルシート（CSS）という技術を使用することが不可欠となっているが、スタイルシートを使用することによって、「ウェブアクセシビリティ」を実現できる以外にも、サイトの更新・管理が簡単になるなど、多くのメリットも出てくる。しかし、現在のところインターネットブラウザによっては、スタイルシートに一部対応してなかったり、スタイルシート対応に関するバグや解釈の違いによって崩れて表示されたり、表示に違いが出てくる等の問題点もあり、それらに対する問題解決の為の方法は、参考文献も少なく、スタイルシートの利用を困難にしている。本論では、スタイルシートを使ってウェブページをレイアウトする際に、発生する不具合を避けてアクセシビリティやユーザビリティに対応したウェブページを実現するための方法について考察する。

2. ウェブアクセシビリティ

パソコンやインターネットが一般に普及し、高齢者や障害者にとっても、ホームページは重要な情報源となりつつある。視覚障害者は、これまで新聞や雑誌等の文字情報を入手するためには、文字情報は点字に置き換えてもらうか、朗読してもらうなど、誰かの助けが必要であったが、インターネットでは、音声ブラウザなどを使用する事によって、毎日のニュースをリアルタイムに読む事ができるようになっている。インターネットが広く普通に使用されるようになった今日、公共性の高い情報を提供しているサイトにおいては、誰もが同じように情報を得られることが特に求められている。アメリカでは1998年に、連邦政府による開発、調達、維持に使用する電子・情報技術に対する障害者のアクセシビリティの保障を義務付ける法律ができ、日本でも日本工業規格としての基準作成が進められ、2004年に「第3部:ウェブコンテンツ」として制定されており、公的機関においては、今後ますますアクセシビリティがウェブ制作の重要な要件として取り組まれることが見

込まれる。

3. テーブルを使ったレイアウト

これまで、見やすく読みやすいレイアウトをウェブページで実現するためには、HTMLの<table>を使って制作するのが一般的であった。しかし、もともと<table>は、表を作成するときのタグであり、<table>を使ったレイアウトは、アクセシビリティという観点からすると様々な問題がある。

例えば、目に障害を持つ人が使用する音声ブラウザは、HTMLで書かれたソースを上から順番に読み上げてくれるのだが、テーブルタグを使うことによって文章構造が崩れてしまい、読み上げる順番も崩れて、内容がきちんと伝わらない場合が出てくるのである。また、ウェブアクセシビリティは、高齢者や障害者のためだけに必要なわけではない。ウェブサイトは、携帯電話やPDAなどからの利用も拡大しているが、これら非パソコン端末は、パソコンと比べると表現できる情報量が限られており、テーブルでレイアウトされたコンテンツは、表示されないものや部分的に情報を取得できないことがある。

4. スタイルシートを使ったレイアウト

そもそも、HTMLは、本来ページのエレメントの構造を定義する言語であり、ウェブページの見栄えなどの表示方法を制御しようとするものではない。そこで、Webページの文章構造を記述するのはHTMLで行い、Webページの見栄えは別の言語で記述しようということで、Web技術の標準化をすすめるW3Cという組織によって、カスケーディング・スタイルシート（CSS）が提唱された。

webページの文章構造部分をHTMLで、見栄えを定義する部分をスタイルシートと、分離することによって、HTMLはシンプルなままで残り、そのコンテンツが正しくマークアップ（機械に情報を理解できるようにタグをつける）されていれば、たとえコンテンツの見栄えを定義するスタイルシートが読み込めなくても、アクセシビリティを向上させることが可能になるというわけだ。また、スタイルシートを使用することによって、アクセシビリティが向上するだけでなく、HTMLとスタイルシートを個別に編集でき、一つのスタイルシートを複数のページに適応させることができるので、大量のページを短時間で簡単に管理・更新できるようになる。また、レイアウトのためのgif画像等も必要なく、複雑に入れ子になったテーブルを使用しないので、ブラウザでの表示も早いなどのメリットもあり、Internet Explorerは、3.0から、Netscape Navigatorは、4.0から対応している。しかし、これらのブラウザはかなり不完全にしかスタイルシートに対応されておらず、Internet Explorer 4.0, Netscape Navigator 6.0以降では、実用的に問題のないレベルまでスタイルシートに対応してきているが、まだ、完全に対応しているとはいえない。そのために、ブラウザによっては、意図したレイアウトとは異なった表示（段組みが崩れる）に

なるなどの不具合が生じることもある。

5. ボックスを使用して段組みを作る

スタイルシートを使って段組みを用いたレイアウトを行う場合、ボックスという概念を理解しておく必要がある。スタイルシートにおけるボックスとは、それぞれの要素（タグ）が持っている長方形の範囲内のことであり、要素のコンテンツ領域、borderという枠、コンテンツと枠の間に相当するpadding、枠と隣接するボックスの境界までの余白であるmarginという要素で構成される。ボックスを定義するには、要素（タグ）名を指定して、このボックスのプロパティであるmarginやpaddingそしてpaddingを囲むborderの設定を行うが、それぞれのプロパティは、意味を明確に理解し、正確な値を入力する必要がある。例えば、ボックスのwidthは、要素の幅ではなく、コンテンツの幅であり、paddingやborderのサイズは含まれていない。したがって、見た目のボックスの幅は、borderのサイズとpaddingのサイズ、コンテンツのwidthを足したものであるため、特にボックスの中にボックスを入れる場合注意が必要である。

また、div要素は、複数の段落をまとめるなどブロックレベル要素をグループ化するとき使用する。div要素に class属性やid属性を指定することで、きめ細かなスタイル設定ができる。class属性は、グループ名と考えると分かりやすい。

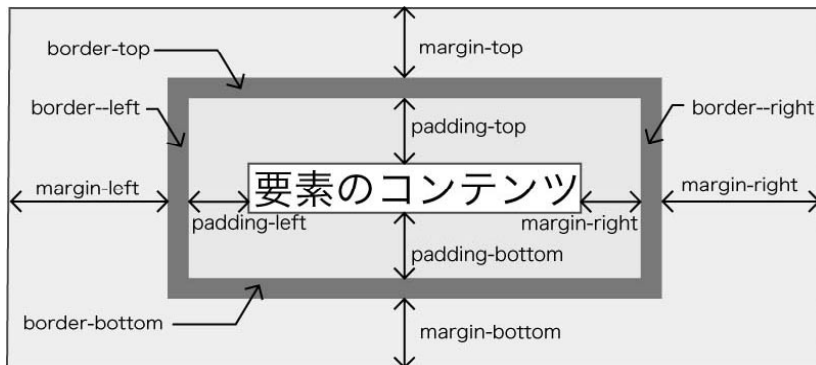
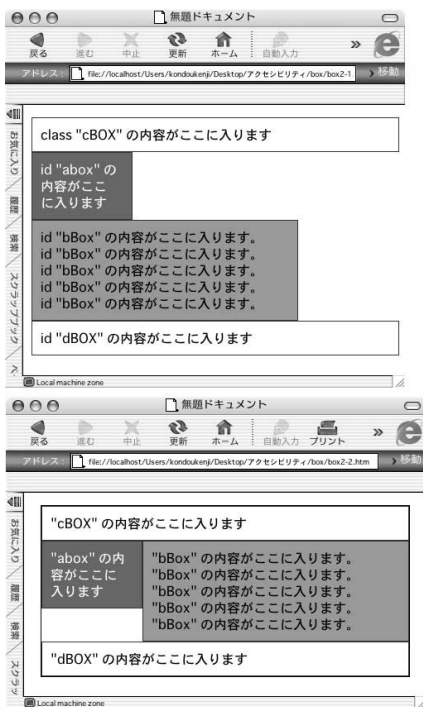


図 1

- Margin 他の要素たちとの余白の部分を設定する
- Padding 要素のまわりのすき間を設定する
- Border 枠線の幅、色、及びスタイルを設定する
- widthとheight 要素の幅、高さではなく、要素のコンテンツの幅、高さを設定する
- floatとclear 親要素の右（左）端に移動して反対側にテキストの回り込みを設定する。
- clear 回り込みを終了させる。

6. 2段組み固定幅のレイアウト

スタイルシートを使って、段組みを用いたレイアウトを実現するには、このdiv要素を使ってコンテンツを幾つかのグループに分け、グループ毎にfloatやclear等のdiv要素のボックスのプロパティを適切に設定することになる。また、div要素の中に、入れ子にdiv要素を入れて使用することもできる。グループ化されたいいくつかのdiv要素を全て一つのdiv要素の中にまとめることによって、ブラウザのウィンドウの横幅を狭めると右側に配置した段組みが下にズレる現象を避けることができる。下のサンプルは、float:leftに設定したdiv要素aBOXとbBOXを幅を444pxで固定したmainBOXの中にまとめたもので、bBOXがaBOXの下回り込むことを避けている。



```
//CSS//
.mainBOX {
    width: 444px;
    margin-right: auto;
    margin-left: auto;
    margin-top: 10px;
}
.aBox {
    width: 100px;
    float: left;
    border: 1px solid #333333;
    padding: 10px;
}
.bBox {
    width: 300px;
    float: left;
    border: 1px solid #333333;
    padding: 10px;
}
.cBOX {
    width: 422px;
    border: 1px solid #333333;
    padding: 10px;
}
.dBOX {
    clear: both;
    border: 1px solid #333333;
    padding: 10px;
    width: 422px;
}

// HTML//
<body>
<div class="mainBOX">
<div class="cBOX"> "cBOX" の内容がここに
    入ります</div>
    <div class="aBox"> "aBox" の内容がここに
    入ります</div>
    <div class="bBox"> "bBox" の内容がここに
    入ります。<br>
    "bBox" の内容がここに入ります。<br>
    "bBox" の内容がここに入ります。<br>
    "bBox" の内容がここに入ります。<br>
    </div>
    <div class="dBOX">"dBOX" の内容がここに
    入ります</div>
</div>
</body>
```

図 2

7. 2段組み左ボックスのみ固定幅のレイアウト

右側にメニューボックスを、左側にコンテンツを配置しするレイアウトは、一般的にみられるレイアウトである。ウィンドウの横幅を拡げると、メニュー部分の左側のボックスは固定しながら右側のコンテンツも表示領域が広がるレイアウトは、文字を拡大して表示するユーザーにとって、画面を効率よく表示させることができるので、アクセシビリティという点で優れているといえる。

左のボックスのfloatをleft、widthを固定、右のボックスのfloatをleft、widthを指定しない設定をすると、Internet Explorerでは正しく表示されるが、Safariなどの他のブラウザでは、右のボックスが、左のボックスの下に回りこんでしまう。すべてのブラウザで正しく表示させるためには、右のボックスのfloatの設定をしないで、margin-leftをaBOXの幅分だけ設定すると左ボックスの下に回り込んでしまうのを防ぐことができた。

```

//CSS//
.abox {
    width: 100px;
    float: left;
    border: 1px solid #333333;
    padding: 10px;
}
.bBox {
    border: 1px solid #333333;
    padding: 10px;
    margin-left: 122px;
}
.cBOX {
    border: 1px solid #333333;
    padding: 10px;
}
.dBOX {
    clear: both;
    border: 1px solid #333333;
    padding: 10px;
}

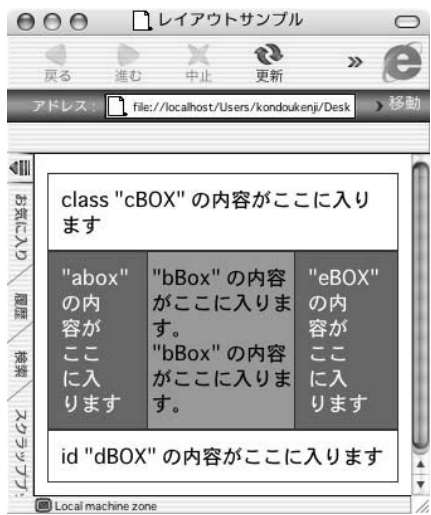
// HTML//
<body>
<div class="mainBOX">
  <div class="cBOX"> "cBOX" の内容がここに
  入ります</div>
  <div class="abox"> "abox" の内容がここに
  入ります</div>
  <div class="bBox"> "bBox" の内容がここに
  入ります。<br>
  "bBox" の内容がここに入ります。<br>
  "bBox" の内容がここに入ります。<br>
  "bBox" の内容がここに入ります。<br>
</div>
  <div class="dBOX">"dBOX" の内容がここに
  入ります</div>
</div>
</body>
    
```

図 3

8. 3段組左右ボックスのみ固定幅のレイアウト

右側にメニューボックスを、左側にサブメニューを配置し、センターにコンテンツを配置するレイアウトも、よくみられるレイアウトである。下の3段組のレイアウトは、ウィンドウの横幅を上げるとメニュー部分の左右ボックスは固定しながらセンターのコンテンツは表示領域が広がるようにしてある。

左のボックスのfloatをleft 右のボックスのfloatをrightに設定し、widthを左右のボックスとも固定にして、センターのボックスは左右のボックスの幅の分だけ左右のmarginをとることによって、うまく表示することができる。



```

//CSS//
.aBOX {
    width: 50px;
    float: left;
    border: 1px solid #333333;
    padding: 10px;
}
.bBOX {
    border: 1px solid #333333;
    padding: 10px;
    margin-left: 72px;
    margin-right: 72px;
}
.cBOX {
    border: 1px solid #333333;
    padding: 10px;
}
.dBOX {
    clear: both;
    border: 1px solid #333333;
    padding: 10px;
}
.eBOX {
    padding: 10px;
    float: right;
    width: 50px;
    border: 1px solid #333333;
}

// HTML//
<body>
<div class="cBOX"> class "cBOX" の内容がここに入ります</div>
<div class="aBOX"> "aBOX" の内容がここに入ります</div>
<div class="eBOX"> "eBOX" の内容がここに入ります</div>
<div class="bBOX"> "bBOX" の内容がここに入ります。<br>
"bBOX" の内容がここに入ります。<br>
</div>
<div class="dBOX"> id "dBOX" の内容がここに入ります</div>
</body>

```

図 4

9. Positionを使った段組みレイアウト

floatを使って段組みをつくる方法以外にも、Positionを使って段組みをすることもできる。Positionは、ボックスの配置方法を指定するプロパティーで、relative（相対位置）absolute（絶対位置）fixed（固定位置）を指定できる。ボックスをabsoluteで指定すると、ボックスの表示位置に関係なくどこにでも置くことができるので、音声ブラウザなどでメニューをページの最後に読ませることが可能になる。下の例は、右のボックスのpositionをabsolute（絶対位置）にして、右のボックスのmargin-leftを左のボックスの幅の分だけ設定して2段の段組みを実現したものである。

fixedは、スクロールしても位置が固定されたままにする設定で、フレームを使ったレイアウトのようにメニューを固定できるので便利であるが、ほとんどのブラウザでサポートされてないようだ。

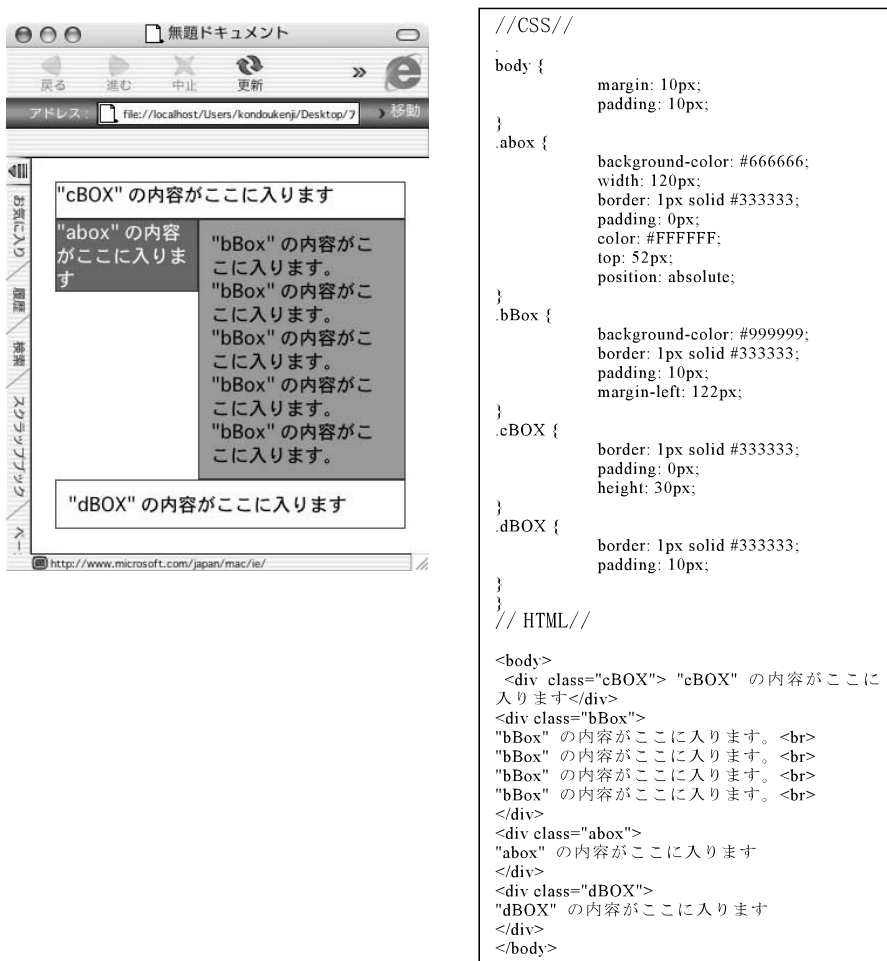


図 5

10. おわりに

今年9月9日より第60回国民体育大会が、11月5日より第5回全国障害者スポーツ大会が岡山で開催される。岡山で開催される全国障害者スポーツ大会では、岡山県下の大学、専門学校計8校で構成される大会サポーターが組織されており、今回初めての試みとして、サポーター養成各校が横の連携を取って、意識の向上をはかることを目的に、全国障害者スポーツ大会の公式ホームページとは別に、大会サポーター独自のホームページが立ち上げられた。倉敷芸術科学大学 映像デザイン学科 webデザイン研究室では、このホームページのデザイン協力の要請を受けて今回ホームページのデザインに取り組んだので、事例報告として紹介する。

ホームページのデザインは、3年次webデザイン実習の授業の一環として学生に課題として取り組んだ。岡山大会サポーターのホームページでは、サポーターや一般の方の他に障害者などがアクセスすることが想定され、アクセシビリティに対応することが特に求められた。一般の方が見やすく分かりやすいホームページであると同時に、障害者の方が利用できるホームページを実現するために、ホームページのレイアウトはスタイルシートを使用した。

図7は採用された案のトップページであるが、2段組み固定幅のレイアウトであり、ヘッダーのボックスの下に右側にメニューボックス、左側にコンテンツボックス、その下にフッターのボックスを配した。メニューのボックスは、スタイルシートの設定でfloatをleftに設定し、コンテンツボックスのfloatをrightに設定し、HTMLではコンテンツボックスの下に記述して、音声ブラウザで読まれたときにコンテンツのトップに戻らなくてもメニューを選択できるようにしている。

メニューのボタンは、画像を使用せずテキストを使用しており、マウスオーバーしたときのボタンのカラーや背景カラーは、スタイルシートで設定したので、ユーザーがブラウザの設定で文字を拡大したときに、メニューの文字も拡大してみせることができ、読み込みのデータが軽く、携帯でも表示可能である。

なお、今回紹介したスタイルシートを使ったレイアウトは、Mac版Internet Explorer 5.2、

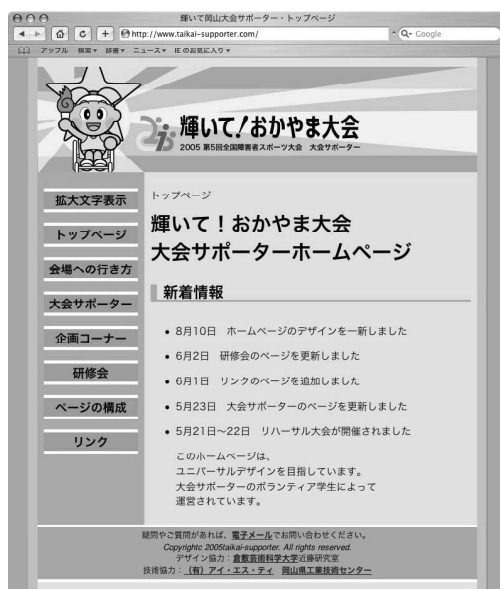


図7 <http://www.taikai-supporter.com/>

Windows版 Internet Explorer 6.0、Safari 1.3で検証を行った。

参考文献

- ・「スタイルシート スタANDARD・デザインガイド」エ・ビスコム・テック・ラボ 著
(株)毎日コミュニケーションズ 2004年
- ・「WebアクセシビリティJIS規格完全ガイド」アライド・ブレインズ編 日経BP社2004年

Prospect concerning Web page layout using style sheets to respond to accessibility

Kenji KONDO

College of Arts,

Kurashiki University of Science and the Arts,

2640 Nishinoura, Tsurajima-cho, Kurashiki-shi, Okayama 712-8505, Japan

(Received September 30, 2005)

Recently, “Web Accessibility”, which enables access and use of information in the web sites, is drawing the attention of aged people and physically-disabled people whose physical and mental functions are limited, regardless of their ages or somatic conditions.

For actualization of the “Web Accessibility”, it is essential to use a technique called “Cascading Style Sheet”, which is recommended by an organization, W3C, promoting standardization of the web technologies. The use of such Style Sheet is extremely advantageous that the up-dating and control of the web sites can be done easily on top of the easy actualization of the “Web Accessibility”.

Although there are many advantages with the Style Sheet, there are still some problems as follows: some internet browsers do not correspond to the Style Sheet, the page is displayed improperly due to differences of bug concept and interpretation of the Style Sheet, and the indications on the page are wrong. There are just a few reference books which explain about how to solve such problems, and this made the use of the Style Sheet difficult and limited.

This article explores the method to actualize the web pages corresponding to the accessibility and usability while preventing aforementioned problems which may occur when laying out the web pages using the Style Sheet.