

2011 年度 メディアプロジェクト演習 1

HTML 講座 発展編 1

立命館大学情報理工学部 メディア情報学科

進捗が遅れている場合に限り、テキスト中で【省略可】マークのついた節は省略してもよい。

1 リンクタグ

1.1 リンクを張る

WWW の最大の特徴として、あるページから他のページへ自由にジャンプして閲覧できる点があります。このようなジャンプをリンクと呼びます。

別のページへのリンクを張るには、`~`タグを使います。このタグで囲まれた範囲をクリックすると、「href」属性に指定した URL（`http://www.ritsumei.ac.jp` など）やファイル名（`index.html` など）へジャンプできます。

タグ	<code>~</code>
機能	リンク先の URL やファイル名で指定された先などへのリンクを張ります。

(sample14.html)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.0 Transitional//EN">
<html>
  <head>
    <title>リンクを張る</title>
  </head>
  <body>
    <p>html では他のホームページへリンクを張る
    ことも可能です。下のテストページをクリックして
    ください。 </p>
    <ul>
      <li><a href="http://www.ritsumei.ac.jp/">
        立命館大学</a></li>
      <li><a href="http://www.media.ritsumei.ac.jp/">
        メディア情報学科</a></li>
    </ul>
  </body>
</html>
```

1.2 複数のページを作ってリンクを張る

他の URL へリンクを張るだけでなく、ローカルな自分の所の文書にもリンクすることができます。今までは、最初に開くトップページ（インデックスページ）だけを編集してきましたが、Web サイトは、通常、1 つのページだけではなく、いくつかのページをハイパーリンクで有機的につなげて、

全体で1つのサイトを構成するようにするのが普通です。

同じディレクトリ内にあるファイルにリンクを張る場合、href 属性の記述は、

```
<a href="ファイル名">~</a>
(例:<a href="index.html">~</a>や
<a href="hobby.html">~</a>)
```

のように、単にファイル名のみとなります。

あるファイルから、別のディレクトリ内にあるファイルへリンクを張る際には、

```
<a href="ディレクトリ名/ファイル名">~</a>
```

のようにファイルの位置を相対パスで記述します。

いくつかのページを作る前に、まず、それぞれのページの内容をどうするか決め、その内容に沿ったファイル名を考えます。例えば経歴や趣味などについて詳しく説明するページを別途設けるのであれば、そのページのファイル名は、「profile.html」や「hobby.html」などが考えられます。他の良い名前を思いついたら、それでもかまいません。日本語のローマ字綴りでも良いでしょう。shumi.html などでも良いわけです。ただし、WWW のリンクでは、今の所日本語のファイル名には対応していません。

(sample15.html)

```
<h3>立命太郎のホームページへようこそ</h3>
<ul>
  <li><a href="profile.html">経歴</a></li>
  <li><a href="hobby.html">趣味</a></li>
</ul>
<hr>
<p><a href="index.html">トップページにもどる
</a></p>
```

基本課題 8： 自己紹介ページから、立命館大学のホームページにリンクを張ってください。

2 色の指定（発展編）

2.1 背景・基本の文字色

body 要素の属性「bgcolor」を使って、バックグラウンドの色を指定することができます。

タグ	<code><body bgcolor="背景色" text="基本の文字色">~</body></code>
機能	HTML 文書本体の背景色および文字色を定義する

例えば、背景色を `black` にしたい場合は、次のように書きます。

(sample16.html)

```
<body bgcolor="black">
```

背景色を指定した場合は、同時に基本の文字色も指定しておく方が良くとされています。背景色とテキストの色とが似た色になってしまうと、読みづらくなる可能性があるからです。

(sample16.html)

```
<body bgcolor="black" text="white">
```

リンクしているテキストの色についても同様に指定した方が良くとされています。

タグ	<code><body bgcolor="背景色" text="基本の文字色" link="リンクが張られている文字の色" alink="アクティブな(マウスが上になっている)リンクの文字の色" vlink="既訪問のリンクの文字の色">~</body></code>
機能	HTML 文書本体の背景色および文字色に加えてリンクが張られている文字の色を設定する

(sample17.html)

```
<body bgcolor="white" text="black" link="blue" alink="red" vlink="green">
```

ここで、`link` が未訪問のリンク、`vlink` が既訪問のリンク、`alink` がアクティブな（マウスが文字の上になっている）リンクです。

2.2 16 進数を使った色の表し方

基礎編で登場した、基本の 16 色以外の色だけでは、例えばピンクやオレンジなどの中間色を使うことができません。基本の 16 色では表現できる色の種類が少なすぎますので、16 進数による色の指定方法が用意されています。

「`bgcolor="..."`」のふたつの「`"`」のくくりの中に、まず「`#`」を書き、その後ろに、RGB の順、つまり、赤、緑、青の順に、2 桁ずつの 16 進数を書きます。大きい数字ほどその色が明るくなります。例えば、「`#ff0000`」は赤、「`#00ff00`」は緑、「`#0000ff`」は青となります。

(sample18.html)

```
<body bgcolor="#ddffdd">
```

などとすると、緑がかった薄い色の背景となります。

ピンクやオレンジに近い色も 16 進数を用いて指定する場合は、それぞれ、「`#ffbcc`」「`#ffa300`」とすると表示されます。

16 進数による色指定で表示できる色の種類が、以下の URL に記載されていますので、参考にしてください。

http://www.page.sannet.ne.jp/mtoga/html/bih-h_x1.htm

なお、16 進数は、0 から、15 までの数を一桁で表すもので、0, 1 から始まって、9 より後ろは、順に、a, b, c, d, e, f までとなります。a, b, c, d, e, f は、10 進数だと、10, 11, 12, 13, 14, 15 の数を表します。

基本課題 9：自己紹介ページの背景と基本の文字色を、16 進数による色表現を使用して、好きな色に設定してください。

3 リストタグ

基礎編では、文章の前に「`・`」を表示させるリストの書き方について説明しましたが、ここでは数字や項目入りのリストの書き方について紹介します。

3.1 番号入りのリスト

タグ	<code>~</code>
機能	番号入りのリストを定義します。

(sample19.html)

```
<ol>
<li>リスト 1</li>
<li>リスト 2</li>
<li>リスト 3</li>
</ol>
```

3.2 項目名入りのリスト

タグ	<code><dl>~</dl></code>
機能	項目名入りのリストを定義します。

タグ	<code><dt>~</dt></code>
機能	リストの項目名を定義します。

タグ	<code><dd>~</dd></code>
機能	リストの内容名を定義します。

(sample19.html)

```
<dl>
  <dt>リストの項目 1</dt>
  <dd>項目 1 の内容</dd>
  <dt>リストの項目 2</dt>
  <dd>項目 2 の内容</dd>
</dl>
```

3.3 リストの中にリストを作る

sample19 のようにリストの中にまたリストを定義することも可能です。

(sample19.html)

```
<ul>
  <li>リスト 1
  <ol>
    <li>リスト 1-1
    <li>リスト 1-2
  </ol>
  <li>リスト 2
</ul>
```

基本課題 10： HTML 基礎編 課題 5 で作成した趣味の箇条書きを、3.1～3.3 のどれかのタグを使って書き換えてみてください。

4 画像を用意する

4.1 ファイルサイズに注意

テキストだけのページだと、文字数にもよりますが、通常はあまり大きなファイルサイズ（バイト数）にはなりません。しかし、画像ファイルの場合は、ファイルサイズが大きい場合があります。ファイルサイズが大きいと、Web ページを閲覧するときに、なかなか画像が表示されない原因となります。よって、Web ページに張り付ける画像ファイルは、ファイルサイズを必要最小限にすることが肝心です。

RAINBOW では、皆さんが使えるホームディレクトリの容量は 50 MB までとなっています。ホームディレクトリの中のデータが 50 MB を超えると、プログラムがコンパイルできなくなるなどの現象も発生しますので、大きなファイルサイズの画像を利用しないように注意しましょう。

4.2 ファイル形式

画像の圧縮形式には様々なものがあります。アイコンなど、使用している色の総数が少ないコンピュータ・グラフィック的な画像の場合は GIF、PNG 形式が多く使われ、写真などの自然画像の場合は JPEG 形式が多く使われます。

4.3 フリー素材の利用

インターネット上には、アイコンや壁紙などの画像材料を置いているページが沢山あります。自由にダウンロードして使って良いものがありますので、気に入ったものがあれば使っても良いでしょう。

ただし、使用に当たっては制限が有る場合が多々ありますので、そのページの説明や指示をよく読んで使うようにしましょう。Web 材料であるという事を唱っていないページの画像データは、ほとんどの場合、二次利用を認めていないので、むやみにゲットしたり、自分のページで使ったりしないようにしましょう。場合によっては法に抵触したり、訴えられたりする可能性もないとは言えません。

4.4 デジタルカメラの利用【省略可】

最近では、デジタルカメラが普及し、写真をコンピュータで扱う事が容易になりました。Web ページに貼付けたり、E-mail にデジタル写真を添付して送るといったような事も多くなるでしょう。ただ、最近ではデジタル写真の画素数が増え、1000 万画素などという写真が撮れるデジタルカメラもめずらしくなくなり、これからますます画素数は増えて行くでしょう。当然ですが、画素数が増えるとともに画像自体のデータサイズが大きくなります。圧縮された形式の JPEG ファイルであっても、画素数が大きいと、容易に 1 MB を越してしまったりします。

では、ファイルサイズを小さくするにはどうすればいいのでしょうか。いろいろな方法が考えられますが、ここでは Windows のアプリケーションを使ってファイルサイズを小さくする方法を紹介しておきます。

以下のいずれの方法も元のファイルの情報を減らす方向での作業になりますから、作業を行なった後からは、元の画像を復元できません。必ず元のファイルのバックアップをとっておきましょう。

(1) 解像度を落とす

アプリケーションによって「解像度を小さくする」という表現をしたり、「画像サイズを小さくする」という言い方になったりします。後者の場合は、単に表示サイズを小さくしているだけで実際のファイル自体に変化のない場合もありますから、前出の方法で、しっかりファイルサイズを調べて確かめた方がいいでしょう。操作を行なった後で、保存します。

Microsoft Photo Editor では、「イメージ」→「サイズ変更」で、画像の解像度を変更することがで

きます。

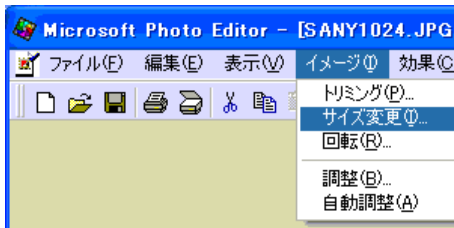


図5 画像の解像度を変更する

(2) トリミング

画面の外側の方の不要な部分をカットしてしまう方法で、「切り抜き」という言葉が使われる場合もあります。必要な部分だけを選択し、後はアプリケーションの「トリミング」や「切り抜き」といった機能を使って不要な部分を切り取って保存します。

Microsoft Photo Editor では、「選択」アイコンで切抜きしたい範囲を指定し、「イメージ」→「トリミング」を選択することで、画像の切り抜きを行うことができます。

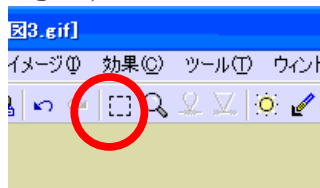


図6 「選択」アイコンをクリックし
画像中の切り抜きしたい範囲を指定する

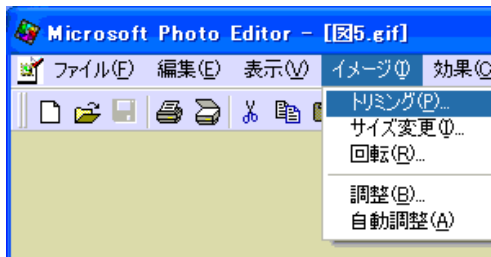


図7 「イメージ」→「トリミング」を選択

(3) 圧縮率を高める

JPEG などの圧縮を伴ったフォーマットの変換時には、オプションで圧縮率を調節する事ができます。圧縮率を高める事で、ファイルサイズを小さくすることができますが、引き換えに画像の質は落ちる事になります。ただ、ほとんど違いがわからない範囲でかなり圧縮率を上げる事は可能です。（ただし、この方法は、Microsoft Photo Editor では利用できません）

以上の方法を併用する事で効果を高める事ができるでしょう。

元のファイルのバックアップ

画像ファイルを編集すると、やっぱり元の方が良かったとか、元に戻して別の編集をしたいなどと思う事も時々起こります。そこで、下のファイルを前もって別ディレクトリにコピー・アンド・ペーストしたり、別のドライブやディスクにコピーしておく事でバックアップをとっておくのが賢明です。

5 画像の張り付け

5.1 画像の大きさを調整する

基礎編で勉強した src, alt 属性以外の img 要素で良く使われる属性としては、width と height 属性があります。

width, height 属性は、表示される画像の幅と高さを与えます。width, height 属性を与えることによって、Web ブラウザは、画像データが到着する前にページのレイアウトを始めることができるようになります。

タグ	
機能	画像の幅と高さを定義する

(sample20.html)

htmlでは画像の大きさを調整することが可能です。

```
<br>
もとのサイズ<br>
<br><br>
幅を200pixelに変更<br>
<br><br>
幅を200pixel, 高さを50pixelに変更<br>

```

5.2 画像の配置【省略可】

画像の配置を、「左寄せ」「中央寄せ」「右寄せ」と変更することも可能です。このような配置に関するタグは第7章で説明します。

5.3 画像にリンクを張る【省略可】

発展編の第2章で文字にリンクを張る方法について説明しましたが、HTML では、文字と同様、タグを~タグで囲むことによって、画像にもリンクを張ることができます。

(sample21.html)

```
<a href="リンク先の URL, ファイル名"></a>
画像ファイルをクリックするとリンク先の URL が
開きます
```

上記 sample24.html をブラウザで表示すると、画

像の周りにリンクが張られていることを示す枠線が入ります。この枠線がデザインの邪魔になる場合は、以下のように「border」属性を「0」にすることで枠線を非表示にすることができます。

(sample21.html)

```
<a href="リンク先の URL, ファイル名"></a>
border 属性で 0 を指定すると画像の外枠の線を非
表示にする事ができます
```

5.4 背景に画像を張り込む 【省略可】

2.1 節 (sample16.html) では背景に色を設定する方法を説明しましたが、背景に画像を張り込むこともできます。body 要素の属性「background」を使って、バックグラウンドの画像を指定します。

タグ	<body background=" 画像ファイル">~</body>
機能	「ファイル名」で指定された画像を背景に定義する

(sample22.html)

```
<body background="61.gif">
<p>htmlではページ全体の背景に画像を表示させる
ことができます。 </p>
</body>
```

発展課題 1: HTML 基礎編 課題 6 で貼り付けた画像の大きさが 2 倍になるように変更してください。

6 表の表示

6.1 表の外観を変更する

基礎編で作成した表には、枠線が表示されず、表の要素が見にくかったと思います。そこで、table 要素の「border」「cellspacing」「cellpadding」属性を使って、表の枠線やその太さ、要素（セル）内のマージンを指定することができます。

タグ	<table border="外枠の線" cellspacing="枠線の太さ" cellpadding="セル内のマージン">~</table>
機能	表の外枠と枠線の太さ、セル内のマージンを定義する

(sample23.html)

表の枠線とその太さと要素内のマージンを設定する。

```
<table border="1" cellspacing="0" cellpadding="0">
<tr>
<td>セル 1 </td>
```

```
<td>セル 2 </td>
</tr>
</table><br>
<table border="1" cellspacing="3" cellpadding="0">
<tr>
<td>セル 1 </td>
<td>セル 2 </td>
</tr>
</table><br>
<table border="3" cellspacing="0" cellpadding="0">
<tr>
<td>セル 1 </td>
<td>セル 2 </td>
</tr>
</table>
<table border="1" cellspacing="0" cellpadding="3">
<tr>
<td>セル 1 </td>
<td>セル 2 </td>
</tr>
</table>
```

6.2 表・セルの幅を指定する 【省略可】

table 要素、および td 要素の「width」属性を使って、表全体の幅や、それぞれのセルの幅を指定することができます。

タグ	<table width="表の幅, 単位はピクセル">~</table>
機能	表の幅を定義する

タグ	<td width="セルの幅, 単位はピクセル">~</td>
機能	セルの幅を定義する

(sample24.html)

表・セルの幅を設定する。

```
<table border="1" width="200">
<tr>
<td>セル 1 </td>
<td>セル 2 </td>
</tr>
</table><br>
<table border="1" width="200">
<tr>
<td width="60">セル 1 </td>
<td width="140">セル 2 </td>
</tr>
</table><br>
<table border="3" width="200">
<tr>
<td width="140">セル 1 </td>
<td width="60">セル 2 </td>
</tr>
```

```
</table>
```

6.3 複数の行や列にわたる項目を作る【省略可】

表を使って時間割を作成する際、2 時限連続で行われる実験などもあります。このような場合には、table 要素の「colspan」「rowspan」属性を使って、複数の行や列にわたってセルを統合することができます。

タグ	<code><td colspan="横にまたがるセルの数" rowspan="縦にまたがるセルの数">~</td></code>
機能	複数の行や列にわたる項目を定義する

(sample25.html)

```
表のセルの配置を設定する。
<table border="1">
  <tr>
    <td colspan="2">セル 1 </td>
  </tr>
  <tr>
    <td>セル 2 </td>
    <td>セル 3 </td>
  </tr>
</table><br>
<table border="1">
  <tr>
    <td rowspan="2">セル 1 </td>
    <td>セル 2 </td>
  </tr>
  <tr>
    <td>セル 3 </td>
  </tr>
</table>
```

6.4 文字や画像の表示位置を指定する【省略可】

td 要素の「align」属性を使って、セル中の文字や画像の表示位置を指定することができます。

タグ	<code><td align="内容の表示位置">~</td></code>
機能	セルの中で内容が表示される位置を定義。left (左寄せ), center (中央寄せ), right (右寄せ) となる。

(sample26.html)

```
表のセルの配置を設定する。
<table border="1" width="300px">
  <tr>
    <td align="left">左寄せ</td>
  </tr>
</table><br>
<table border="1" width="300px">
  <tr>
    <td align="center">中央寄せ</td>
  </tr>
</table>
```

```
</table><br>
<table border="1" width="300px">
  <tr>
    <td align="right">右寄せ</td>
  </tr>
</table>
```

6.5 表の各項目の色を変える【省略可】

table 要素の「bgcolor」属性を使って、表全体の背景色を指定することができます。td 要素の「bgcolor」属性を使って、セルの背景色をそれぞれ指定することもできます。

タグ	<code><table bgcolor="背景色">~</table></code>
機能	表全体の背景色を定義する

タグ	<code><td bgcolor="背景色">~</td></code>
機能	セルの背景色を定義する

(sample27.html)

```
表・セルの背景色を設定する。
<table border="1" bgcolor="#ffffcc">
  <tr>
    <td>セル 1 </td>
    <td>セル 2 </td>
  </tr>
</table><br>
<table border="0">
  <tr>
    <td bgcolor="#ccccff">セル 1 </td>
    <td bgcolor="#ffdddd">セル 2 </td>
  </tr>
</table>
```

発展課題 2： HTML 基礎編 課題 7 で作成した時間割表の中で「実験」など複数時限にわたる科目の項目について、複数行・列を統合してください。

7 横方向の配置

ブロックレベルの要素の配置をページ内横方向で整列すると、視覚的にとても良い効果が得られる事があります。ここで、ブロックとは、テーブルのセルや、h タグによる見出しのように、ブロック状に囲まれた部分のことを便宜上指しています。

7.1 center 要素

視覚的に効果的なものとして、ブロックレベルでの配置、特に中央寄せが挙げられます。最も簡単に実現するには、center 要素を使います。

center 要素は、中にブロックレベルの要素をいくつでも含む事ができますので、h 要素や p 要素

をいくつも入れる事ができます。

タグ	<code><center>~</center></code>
機能	タグで囲まれた内容を中央寄せする

(sample28.html)

```
<center>
<p>中央に配置されます</p>
</center>
```

7.2 div 要素

ひとつひとつのブロックレベルの要素で個別に属性を指定しなくても良いように、便利な要素が用意されています。それは `div` 要素で、`div` 要素自体ブロックレベルで、内側にいくつでもブロックレベルの要素を含む事ができますから、`div` 要素の属性として、`align` 属性値を与えておけば、その中の `h` 要素や `p` 要素の配置を全部まとめてコントロールできます。

タグ	<code><div>~</div></code>
機能	タグで囲まれた内容をひとまとめにします

タグ	<code><div align=" 内容の表示位置">~</div></code>
機能	内容が表示される位置を定義。 <code>left</code> (左寄せ), <code>center</code> (中央寄せ), <code>right</code> (右寄せ) となる。

(sample29.html)

```
<div align="left">左に寄せます</div>
<div align="center">中央に配置します</div>
<div align="right">
  <p>右に寄せます</p>
  <h1>h タグとも一緒に利用できます</h1>
</div>
```

発展課題 3: 自己紹介ページの見出し部分を中央に配置してください。

8 その他のタグ

HTML では、その他にもフレームタグなどいろいろなタグや属性が存在します。以下の URL に整理されていますので、参考にしてください。

<http://www.tagindex.com/>

2011 年度 メディアプロジェクト演習 1

HTML 講座 発展編 2 ~ CSS と Web アクセシビリティ

立命館大学情報理工学部 メディア情報学科

進捗が遅れている場合に限り、テキスト中で【省略可】マークのついた節は省略してもよい。

1 はじめに

本演習ではこれまで、様々なタグを用いた基礎的な Web ページの作成方法・レイアウト方法について学んできました。ここで、タグは本来、Web ページ上の文章構造を決定するために用いるものです。また、タグでレイアウトを決定すると、HTML 文章の構造が複雑になり、Web ページの運用・保守が非常に困難となります。そのため、Web ページのレイアウトに関しては、Cascading Style Sheets (カスケーディング・スタイル・シート: CSS) を用いてそれを実現することが推奨されてきました。

そこで発展編 2 では、CSS の基礎とその記述方法を説明し、CSS を用いた Web ページ作成について理解を深めることを目標とします。また、世界中の人々が分け隔てなく情報資源にアクセスできるようにすることを目的とした、Web アクセシビリティの考え方についても学びます。

2 Cascading Style Sheets

CSS は、基本的には、各タグで囲まれた文章をどのように表示するかを決定します。CSS を用いる際には、必ず以下の記述が必要となります。

タグ	<code><meta http-equiv="Content-Style-type" content="text/css"></code>
機能	CSS を使用することを宣言します。

(sample34.html)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.0 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Style-type"
content="text/css">
</head>
<body>

</body>
</html>
```

なお、ページ幅の都合上改行してありますが、改行はしなくて結構です。しかし、この改行部分に

相当するところに、半角スペースを入力してください。

3 CSS の記述方法

3.1 CSS の書式

CSS は、基本的には、以下の書式をとります。

```
セレクタ {プロパティ:値;...}
```

ここで、セレクタは適用範囲、プロパティは何を指定するか(文字の色、形など)、値は具体的な内容(赤、MS 明朝など)を指定します。また、セレクタに示した適用範囲に複数の指定(文字の色と形の両方など)をしたい場合は、「;」で区切ります。このとき、「;」で区切った後に改行すると見やすい場合があります。

3.2 CSS の 3 つの実現方法

CSS には以下に示す 3 つの実現方法があります。

- style 属性を使う方法
- HTML ファイルの<head> ~ </head>間に CSS を記述する方法
- CSS のみを記述した別ファイル(***.css)を用意し、HTML ファイルからリンクを貼る方法

style 属性を使う場合は以下のように記述します。

タグ	<code><要素名 style="プロパティ:値"></code>
機能	要素のスタイルを指定します。

(sample35.html)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.0 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Style-type"
content="text/css">
</head>
<body>
<p style="color:#FF0000">赤い文字</p>
</body>
</html>
```

次に、HTML ファイルの<head> ~ </head>間に CSS を記述する場合は、以下のように記述します。

タグ	<style type="text/css">~</style>
機能	CSS を HTML ファイルの中で記述します

(sample36.html)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.0 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Style-type"
content="text/css">
<style type="text/css">
<!--
      p{color:#FF0000}
-->
</style>
</head>
<body>
<p>赤い文字</p>
</body>
</html>
```

ここで、<!-- -->は、CSS に対応していないブラウザで閲覧したときに、この中の部分がそのままテキストとして表示されないようにするための配慮です。

最後に、CSS のみを記述した別ファイルを用意し、HTML ファイルからリンクを貼る場合は、以下のように記述します。

タグ	<link rel="stylesheet" href="***.css" type="text/css">
機能	***.css ファイル内で記述した CSS を、HTML ファイルの中で使用します。

(sample37.html)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.0 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Style-type"
content="text/css">
<link rel="stylesheet" href="sample1.css"
type="text/css">
</head>
<body>
<p>赤い文字</p>
</body>
</html>
```

(sample1.css)

p{color:#FF0000}

なお、一般に CSS を別ファイルに記述する方法が推奨されています。これは、複数の HTML ファイルから CSS を参照できるようにすることで、

HTML 作成・保守・運用を簡単にするためです。本資料でも、CSS を別ファイルに記述する方法に絞って説明して行きます。

3.3 Class

これまで、指定した要素にのみ適用可能なスタイルの指定方法について説明してきました。しかしながら CSS では、class (クラス) と呼ばれるものを用いて好きな名前スタイルを指定することが出来ます。これにより、異なる要素に対して共通のスタイルの指定、また、同じ要素に対して異なるスタイルの指定などが可能になります。まず、クラスを用いた CSS を適用する際の HTML タグの記述方法は以下のようになります。

タグ	<要素名 class="クラス名">
機能	要素にクラス名で指定されたスタイルを適用

なお、クラス名の後にスペースを空けてその他のクラス名を記述することにより、2 つのスタイルを 1 つの要素に適用することが出来ます。次に、CSS 側は、以下のような指定方法になります。

.クラス名 (任意に決定) {プロパティ: 値;...}

要素名.クラス名 (任意に決定) {プロパティ: 値;...}

ここで、前者は要素を指定していないので、どのような要素にも適用させることが可能です。また、後者は、指定された要素の一部にのみ、適用させることが可能となります。

定義したクラスを html ファイルに適用する方法は、sample38.html のとおりです。ただし、<p class="red it">のように複数のクラスを適用する場合は、クラス名とクラス名の間にスペースを入れます。

(sample38.html)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.0 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Style-type"
content="text/css">
<link rel="stylesheet" href="sample2.css"
type="text/css">
</head>
<body>
<p class="red">赤い文字</p>
<p class="red it">赤いイタリック文字</p>
```

```
<h1 class="blue">青い文字</h1>
<p class="blue">青くない文字</p>
</body>
</html>
```

(sample2.css)

```
.red{color:#FF0000}
.it{font-style:italic}
h1.blue{color:#0000FF}
```

4 CSS の例

以下に、代表的な CSS について説明します。なお、これら以外にも様々な CSS が存在します。また、それぞれの値の指定方法についても、これ以外のものが沢山在ります。それらを勉強する際には、<http://htmlhelp.com/ja/reference/css/> が良い参考ページとして役立つと思います。

4.1 CSS の例 1

タグの範囲の背景色を指定

CSS	background-color:色
機能	背景色を指定
値	数値 16 進数で色を指定 色名 色名で色を指定

タグの範囲の文字色を指定

CSS	color:色
機能	文字色を指定
値	数値 16 進数で色を指定 色名 色名で色を指定

文字の大きさを指定

CSS	font-size:文字の大きさ
機能	文字の大きさを指定
値	%, em, px, pt など指定
備考	%:標準文字サイズを 100%とした割合 em:親要素の文字サイズを 1 em としたとき の比 px:ピクセル数で指定 pt:ポイント数で指定

文字の太さを指定

CSS	font-weight:文字の太さ
機能	文字の太さを指定
値	bold, normal

文字の斜体を指定

CSS	font-style:文字の種類
機能	斜体とするかしないかを指定
値	italic 斜体 normal 標準の文体

文字のフォントを指定

CSS	font-family:フォント名 or 文字の種類名
機能	フォント名や種類名を指定
値	文字の種類名 serif 文字の先端にはねがある。 sans-serif 文字の先端にはねが無い cursive 筆記体 monospace 幅が全て同じフォント フォント名 フォント名には様々なものがあります。皆さんで調べてみてください。

行間を指定

CSS	line-height:行間
機能	複数行にわたるブロックの行間を指定
値	%, em, px などの数値

文字の表示位置を指定

CSS	text-align:表示位置
機能	文字の表示位置を指定
値	left 左揃え center 中央揃え right 右揃え justify 両端揃え (半角のみ)

(sample39.html)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.0 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Style-type"
content="text/css">
<link rel="stylesheet" href="sample3.css"
type="text/css">
</head>
<body>
<p class="sec">章用スタイル</p><br>
<p class="subsec">節用スタイル</p><br>
<p class="text">本文用スタイル本文用スタイル本
文用スタイル<br>
本文用スタイル本文用スタイル本文用スタイル
</p>
<p>スタイル適用していない本文 (上と行間を比べ
てみてください) <br>
スタイル適用していない本文(上と行間を比べてみ
てください) </p>
<h1 class="center">中央寄せ：文字色赤：背景黒
</body>
</html>
```

(sample3.css)

```
.sec
{
    font-size:16pt;
    font-weight:bold;
    font-family:"M S P ゴシック";
}
.subsec
{
    font-size:14pt;
    font-family:"M S P ゴシック";
}
.text
{
    font-size:12pt;
    font-family:"M S P 明朝";
    line-height:1.5em;
}
h1.center
{
    text-align:center;
    color:#FF0000;
    background-color:#000000;
}
```

基本課題 11：自己紹介ページの背景色やフォントなどのスタイルを別途 CSS ファイルとして作成し、そのスタイルを自己紹介の HTML ページに反映させてください

4.2 CSS の例 2 【省略可】

文字の周囲の余白を指定

CSS	padding:間隔
機能	文字の大きさを指定
値	数値 %, em, px, pt など指定
備考	%:標準文字サイズを 100%とした割合 em:親要素の文字サイズを 1emとしたときの比 px:ピクセル数で指定 pt:ポイント数で指定

文字の周囲にそれぞれ異なる余白を指定

padding:a b c d	文字の上(a)右(b)下(c)左(d)の余白を指定
padding:a b c	文字の上(a)左右(b)下(c)の余白を指定
padding:a b	文字の上下(a) 左右(b)の余白を指定
padding:a	文字の上下左右(a)の余白を指定

上下左右の特定の部分に余白を指定

padding-top	文字の上の余白を指定
padding-right	文字の右の余白を指定
padding-bottom	文字の下の余白を指定
padding-left	文字の左の余白を指定

ブロックの周囲の余白を指定

CSS	margin:間隔
機能	ブロック枠の周囲への余白を指定
値	数値 %, em, px, pt など指定 auto 状況に応じて自動調整
備考	%:標準文字サイズを 100%とした割合 em:親要素の文字サイズを 1emとしたときの比 px:ピクセル数で指定 pt:ポイント数で指定 ブロックとは、テーブルのセルや、h タグによる見出しのように、ブロック状に囲まれた部分のことを便宜上指しています。

ブロックの周囲にそれぞれ異なる余白を指定

margin:a b c d	ブロックの上(a)右(b)下(c)左(d)の余白を指定
margin:a b c	ブロックの上(a)左右(b)下(c)の余白を指定
margin:a b	ブロックの上下(a) 左右(b)の余白を指定
margin:a	ブロックの上下左右(a)の余白を指定

上下左右の特定の部分に余白を指定

margin-top	ブロックの上の余白を指定
margin-right	ブロックの右の余白を指定
margin-bottom	ブロックの下の余白を指定
margin-left	ブロックの左の余白を指定

(sample40.html)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Style-type" content="text/css">
<link rel="stylesheet" href="sample4.css" type="text/css">
</head>
<body>
<p class="sec">文字余白確認：章用スタイル</p>
<p class="subsec">文字余白確認：節用スタイル</p>
<p class="sec_b">ブロック余白確認：章用スタイル</p>
<p class="subsec_b">ブロック余白確認：節用スタイル</p>
```

```
ル</p></body>
</html>
```

(sample4.css)

```
.sec
{
    font-size:16pt;
    font-weight:bold;
    font-family:"M S P ゴシック";
    background-color:#000000;
    color:#FF0000;

    padding:7pt;
}
.subsec
{
    font-size:14pt;
    font-family:"M S P ゴシック";
    background-color:#000000;
    color:#FF0000;

    padding:7pt 7pt 1pt 24pt;
}
.sec_b
{
    font-size:16pt;
    font-weight:bold;
    font-family:"M S P ゴシック";
    background-color:#000000;
    color:#FF0000;

    margin:0;
}
.subsec_b
{
    font-size:14pt;
    font-family:"M S P ゴシック";
    background-color:#000000;
    color:#FF0000;

    margin-top:0pt;
    margin-left:24pt;
}
```

4.3 CSS の例 3 【省略可】

文字の周囲に枠線を表示

CSS	border-style:種類
機能	枠線の種類を指定
値	solid 枠を1本線で表示 dotted 枠を点線で表示 dashed 枠を破線で表示 double 枠を二重線で表示 groove 枠がへこんでいるように表示 ridge 枠が出っ張っているように表示 inset 枠の内側がへこんでいるように表示

表示	表示
outset	枠の内側が出っ張っているように表示
none	枠無し。(初期値)

上下左右に、異なる枠線を指定

border-style:a b c d	上(a)右(b)下(c)左(d)
border-style:a b c	上(a)左右(b)下(c)
border-style:a b	上下(a)左右(b)
border-style:a	上下左右(a)

枠線の太さを指定

CSS	border-width:太さ
機能	枠線の太さを指定
値	px, em

上下左右に、異なる枠線の太さを指定

border-width:a b c d	上(a)右(b)下(c)左(d)
border-width:a b c	上(a)左右(b)下(c)
border-width:a b	上下(a)左右(b)
border-width:a	上下左右(a)

枠線の色を指定

CSS	border-color:色
機能	枠線の太さを指定
値	px, em

上下左右に、異なる枠線の色を指定

border-color:a b c d	上(a)右(b)下(c)左(d)
border-color:a b c	上(a)左右(b)下(c)
border-color:a b	上下(a)左右(b)
border-color:a	上下左右(a)

枠線の種類、太さ、色を一括指定

CSS	border:種類 太さ 色
機能	枠線の種類、太さ、色を指定
値	border-style/border-width/border-color と同じ

特定の部分に枠線の種類、太さ、色を指定

border-top	上
border-right	右
border-bottom	下
border-left	左

(sample41.html)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.0 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Style-type"
content="text/css">
```

```
<link rel="stylesheet" href="sample5.css"
type="text/css">
</head>
<body>
<span class="solid">solid</span>
<span class="dotted">dotted</span>
<span class="dashed">dashed</span>
<span class="double">double</span><br><br><br>
今までの CSS を下にまとめてみました。
<p class="solid2">border-style:solid; border-width:1px;
border-color:#666666; padding:5px;<br>
background-color:#CCCCCC; color:#666666;
font-family:"Arial"; font-weight:bold; font-size:11pt;
line-height:1.5em;</p>
</body>
</html>
```

(sample5.css)

```
.solid{border-style:solid}
.dotted{border-style:dotted}
.dashed{border-style:dashed}
.double{border-style:double}

.solid2
{
    border-style:solid;
    border-width:1px;
    border-color:#666666;

    padding:5px;

    background-color:#DDDDDD;
    color:#666666;

    font-family:"Arial";
    font-weight:bold;
    font-size:11pt;
    line-height:1.5em;
}
```

5 Web アクセシビリティ

WWWにおいては、世界中の人々が分け隔てなく世界中の情報資源にアクセスできるようにしようという理念があります。この事を、アクセシビリティと呼んでいます。アクセシビリティの向上のために、Webの表現方法を直接左右するHTMLの規格について、いろんな改良、変更が加えられてきました。HTMLの要素の中には、アクセシビリティについての考慮の足りなかった要素も含まれており、現在、そのような要素については非推奨という指定がされています。

5.1 非推奨な要素や属性

HTMLの仕様の中には、色の指定や、文字の装飾やサイズの指定、テキストや画像の配置など、見た目の構造を制御する要素や属性がかなりあります。WWWでは、楽しいページや目立つページ、見やすいページを目指して、そういった様々な物理的な構造の工夫をしています。

しかし、実は、そのような物理的な構造を表現する要素や属性は、現在のHTMLの規格においては、ほとんどのものが「非推奨」となっています。ただ、非推奨だからと言って、見た目の工夫を否定してしまったら、そのページの目的によっては意味のないものとなったり、楽しくないものになってしまう可能性もあります。

これは、物理的な構造を否定するのではなく、言語仕様の上で、HTMLは論理的な構造を表現する役割に専念し、物理的な構造については別の言語仕様で面倒をみるという、役割分担を目指した結果だと考えられます。そこで、物理的な構造を表現するために、本資料で勉強したCSS (Cascading Style Sheets) という言語仕様が提案され、実際に使われています。

ただ、現実的にはまだまだ見た目を表現するための要素や属性は多く使われており、TransitionalなDTDを用いる限り、禁止されているわけではありません。扱いが容易なので、本講座でも、フォントの指定や色の指定などの見た目の表現についてのHTML要素や属性についても学びました。

5.2 Webアクセシビリティ

Webのアクセシビリティを押し進めている活動として、Web Accessibility Initiative (WAI) が挙げられます。Webアクセシビリティについてのガイドラインの日本語訳がありますので、そのページのURLを載せておきます。

<http://www.zspc.com/documents/wcag10/index.html>

また、本資料最後の付録ページに、チェックリストを載せておくので、自分のホームページをチェックしてみてください。

さて、なぜHTMLの仕様の中から物理的な構造についての要素や属性を分離しなければならなかったかと言うと、これはアクセシビリティの問題があったからです。例えば、読み上げソフトを使ってWWWを閲覧している目の不自由な人たちもいます。文字の大きさや文字の色を直接指定してあっても、そのような人にとっては、その事自体にはあまり意味がありません。

しかし、文字の大きさを直接指定するのではな

く、h1要素のように、「見出し」であるという論理的な構造を、HTML要素として指定しておけば、読み上げソフトはそれに則してその部分が見出しであると言う事を閲覧者に伝える事ができるでしょう。そのような意味で、HTMLにおいては論理的な構造を表現する方法を確立させておき、物理的な構造の表現については別の手段を用意するという役割分担の考え方が、アクセシビリティの向上にとってとても有効であると言えます。

参考文献

[1] HTMLとスタイルシートによる最新Webサイト作成術—ホームページでなにを伝える?どう作る?, エクスナレッジ, December 2002.

付録: アクセシビリティガイドライン

1	優先度[1] 必ず配慮したい項目
2	優先度[2] できるだけ配慮したい項目
3	優先度[3] できれば配慮したい項目

番号	ガイドライン	優先度
全体		
A-1	ページの制作サイズはブラウザの横幅800 pxを超えないこと。	1
A-2	HEAD に付ける<TITLE>は内容が解るように適切なタイトルを付けること。	1
A-3	背景色・文字色等は、コントラストが十分にあり、見やすいこと。	2
A-4	緑と赤の区別の難しい人がいることを配慮し、この色の組み合わせは極力避けること。	2
A-5	1 ページは適切な長さにし、やむを得ず長くなるときは、アンカーをつけること。	2
A-6	ページのレイアウトのためにテーブルを利用するのは極力避けること。	2
A-7	フォントサイズ・色・表示位置等の制御にはできるだけスタイルシートを使うこと。	2
A-8	色の指定には色名 (red や blue) ではなく、数値 (#ffffff) などを利用すること。	2
A-9	ステータスバーには、できるだけメッセージを流さないこと。	2
A-10	タグの属性値は引用符 (") で囲むこと。	2
A-11	できれば Web サイトの概略を記述した英文ページを用意すること。	3

A-12	どうしてもページをアクセシブルにできない場合は、同等の情報を持つテキストページを用意する。	2
A-13	サイト全体の構成を示すために、サイトマップや目次を提供すること。	2
A-14	サイトで使用する文章の基本となる言語を指定しておくこと。	3
A-15	レイアウト、ナビゲーション、文章表現などは、サイト内で一貫性のあるルールで作ること。	3

画像

B-1	画像には ALT 属性を付けること。	1
B-2	リンクボタンになっている画像の全てに ALT 属性を使ってリンク先を明確にする事。	1
B-3	意味の持たないワンポイントの画像には ALT にスペースを入れること。	1
B-4	画像は極力軽く作ることを心がけ、指定には width, height 属性を付けること。	2
B-5	イメージマップのリンクに ALT 属性を付ける。	1
B-6	重要な画像 (図表やグラフ) には、解説をリンクさせること。	3
B-7	文字の透過画像は、反転させても見えるように配慮すること。	2
B-8	画像中の文字は見やすくすること。	2
B-9	点滅やスクロールなど変化する画像の使用は、コントラスト・点滅スピードに注意すること。	2

リンク

C-1	リンクの下線はむやみに消さないこと。	2
C-2	リンクするボタン画像などは、リンクである事が解るように配慮すること。	2
C-3	キーボード操作だけで適切なリンクができること。	2
C-4	「戻る」等のサイト内基本操作は、解りやすいように同じ位置・形状にすること。	2
C-5	リンクする画像が隣接する場合は、誤操作をしないように間を開けること。	2
C-6	リンクを張ったテキストは幅が狭すぎないこと。	2
C-7	リンクによって、新しいウィンドウを開くことは必要最低限とすること。	2
C-8	リンクの文字は、文章の前後関係を見なくても意味が通り、あまり長くないこと。	3

C-9	連続したリンクが 1 つのものと誤認されないように、間に仕切り記号を入れる等配慮をする。	3
C-10	文頭等にナビゲーションを付ける場合は、読み飛ばせるように、本文へのアンカーを付けること。	2
音声・映像		
D-1	音声情報に対してはできるだけ文字でも情報を提供すること。	2
D-2	ビデオ情報に対しては、できるだけそれと同期した音声の情報も提供すること。	2
D-3	重要な情報を警告音等で知らせる場合は、同等のメッセージを表示すること。	2
D-4	音声情報は、再生・停止・音量調整などのコントロールを可能にすること。	2
テキスト		
E-1	日本語のページでは外国語の乱用はせず、誰にでも解る様に配慮すること。	1
E-2	色によって伝えられる情報は、色がなくても情報が伝わるようにすること。	1
E-3	略語や一般的でない言葉には、理解しやすいように説明を付けるなどして配慮すること。	3
E-4	日付の表記には「/」を使わず、できるだけ日本語を使うこと。	3
E-5	全角と半角の使い方は、できるだけサイト内で統一すること。	3
E-6	文字サイズ・行間はスタイルシートを使用し、読みやすい物にすること。	2
E-7	見出し要素は正しい順番を守ること。	2
E-8	一単語内にスペースや強制改行を入れないこと。	2
E-9	重要な意味を持つ物には記号を用いないこと。	2
E-10	点滅したりスクロールしたりするテキストの使用は極力避けること。	2
E-11	リストは正確に構造を記述し、見栄えのためには用いないこと。	2
E-12	リストの番号表示等は音声では読み上げられないので注意すること。	2
E-13	丸数字等の機種依存文字は使わないこと。	1
E-14	PDF での情報提供は極力避けること。	2
テーブル		
F-1	表組は、行と列の関係を考慮してなるべ	2

	く解りやすくすること。	
F-2	表の始まりには「表題」を付けること。	2
F-3	行や列の見出し項目には<TH>を使って指定すること。	2
F-4	セルの結合は、必要最小限にすること。	2
F-5	レイアウトのテーブルは、情報が適切に音声ブラウザで読み上げられること。	1
フレーム		
G-1	フレームは必要最小限にすること。	1
G-2	それぞれの FRAME 要素にタイトルを付けること。	2
G-3	画面が小さくても表示できるようにスクロールバーの非表示には注意すること。	2
G-4	フレームに直接画像を入れずに別の HTML ファイルを使うこと。	2
フォーム		
H-1	submit ボタンに画像はできる限り使わないこと。	2
H-2	入力ボックス、テキストエリアにはできるだけデフォルト値を持たせること。	2
H-3	フォームによる入力をさせる場合には、できるだけ代替え手段を用意すること。	3
H-4	プルダウン形式の選択メニューはキーボードからのアクセスを可能にすること。	2
JavaScript		
I-1	JavaScript には対応していない音声ブラウザを考慮して代替え手段を用意すること。	2
I-2	Java アプレットには対応していない音声ブラウザを考慮して代替え手段を用意すること。	2
I-3	利用者の意図に関わらず、ページの更新や移動を自動的に行わないこと。	2

2011 年度 メディアプロジェクト演習 1

HTML 講座 発展編 3 ~ JavaScript

立命館大学情報理工学部 メディア情報学科

1 JavaScript

JavaScript は HTML 文書の中に埋め込むことができるプログラミング言語であり、ユーザのマウス操作に応じて表示内容が切り替わるインタラクティブな Web ページを作るために使用されている言語である。

JavaScript のプログラムは HTML 文書の中で特定のタグに挟んで埋め込むだけで実行され、コンパイルは必要ない。プログラムは Web ブラウザによって実行され、ユーザのマウス操作に応じて HTML の内容が書き換えられる。本資料では JavaScript の基本的な部分だけを述べるが、画像の描画から音楽の再生、サーバとの通信に至るまで、JavaScript で実現可能な機能は多岐にわたる。

2 JavaScript の機能

JavaScript における最も基本的な機能として、マウスイベントとそれによる関数の呼び出しについて述べる。

2.1 マウスイベント

JavaScript ではブラウザ上でユーザが行ったマウス操作を「イベント」として捕らえることができる。これをきっかけとして関数を呼び出すことで、ブラウザとユーザの間でのインタラクティブなやりとりを実現する。

以下はマウスを上に乗せた場合に画像を入れ替える HTML 文書である。ただしこの HTML 文書と同じディレクトリに image1.jpg, image2.jpg という名称で作成した画像を置いておくこと。

ここでは onmouseover="..." と onmouseout="..." の部分で JavaScript を使用している。変数 this は onmouseover や onmouseout が定義された HTML 要素（この場合 img 要素）を指し、その src（ソース）となる画像を切り替えている。

ピリオドの後に属性名（メンバ名）をつけて参照するのは C における構造体と同じである。

onmouseover や onmouseout はマウス操作によって発生するイベントであり、「マウスイベント」と呼ばれる。

(sample42.html)

```
<html><body>

</body></html>
```

JavaScript ではクォーツ (“ や ‘) が入れ子になることが多いため、閉じ忘れのないよう注意すること。

(sample43.html)

```
<html><body>
<table><tr>
<td align=center bgcolor=#0000ff onClick=
"document.getElementById('word1').style.color=
'ff0000'">
<font size=7 color=#ffffff>Click here!</font>
</td></tr></table>
<font size=7>Have fun with
<font id="word1">JavaScript!</font></font>
</body></html>
```

基本課題 12 : JavaScript を使用して、自己紹介ページの画像の上にマウスを重ねると別の画像に変わるように HTML ページを変更してください。

2.2 関数の定義

JavaScript では関数を定義して使用できる。定義された関数を onClick 等の引数として与えることで、マウスイベント発生時に実行させることができる。

関数は script 要素内で <!-- と --> に挟んで記述する。script 要素は HTML 文書の head 要素内に記述する。

予約語 function の後に関数名を書き、C と同様の形で定義する。

たとえば以下の HTML 文書では、「Multiples of 2」あるいは「Multiples of 3」をクリックすると、2 の倍数や 3 の倍数が表示される。JavaScript では C と同じように for ループや if 文による条件分岐などの制御構造が利用できる。

(sample44.html)

```
<html><head>
<script type="text/javascript">
<!--
function showNumbers(num) {
```



```

for (i = 1; i<=10; i++) {
document.getElementById("area1").innerHTML +=
(i*num) + " ";
}
document.getElementById("area1").innerHTML +=
"<br>";
}
// --></script></head>
<body>
<font size=7 onClick="showNumbers(2)">
Multiples of 2</font><br>
<font size=7 onClick="showNumbers(3)">
Multiples of 3</font><br>
<font size=7 id="area1"></font>
</body></html>

```

なお、JavaScript では変数を宣言なしで利用できる。また、型の指定も不要である。そのため仮引数 (num) にも型をつけなくてよい。

文字列は演算子「+」を使って連結できる。整数型の変数は必要に応じて文字列型に自動変換される。

2.3 タイマー

JavaScript では一定時間後に HTML を書き換えるといった処理が可能である。これは `setTimeout` 関数によって実現される。

HTML 文書をブラウザが読み込む際、`onLoad` というイベントが発生するため、`body` 要素の `onLoad` の引数として `setTimeout` 関数を与え、その第一引数として与えた任意の関数を実行させることができる。第二引数が実行までの待ち時間である。時間はミリ秒を単位として指定する。

以下の HTML 文書ではブラウザへの読み込みから 3 秒後に背景色が切り替わる。`document.bgColor` は HTML 文書の背景色を決め、`document.fgColor` はフォント色を決めている属性である。

(sample45.html)

```

<html>
<head>
<script type="text/javascript">
<!--
function changeColor() {
    document.bgColor = 'black';
    document.fgColor = 'white';
}
// --></script></head>

<body onLoad="setTimeout('changeColor()',3000)">
<font size=7>Day and night.</font>
</body></html>

```

基本課題 13：自己紹介ページに、ブラウザに表示して 3 秒経過すると背景色が変わる機能を追加してください。

参考文献

JavaScript の詳細な機能に関する説明は Web 上に多数見受けられるので、適宜調査し利用すること。例として以下のサイトが挙げられる。

- [1] とほほの JavaScript リファレンス
<http://www.tohoho-web.com/js/>

また、以下の文献もリファレンスとして使用するのに適している。

- [2] JavaScript 辞典 第3版, 株式会社アंक著, 翔泳社, 2008.