

情報科学演習

第6回

WWW と HTML

目次

1	本日の目標	1
2	用語の解説	1
3	本日の実習	2
3.1	HTML について	2
3.2	テキストベースブラウザを使う	4
3.3	スタイルシートを書く	5
3.4	Web ページを公開してみる	6
3.5	残りの時間の課題: Web ページの改良とバックアップ	8

1 本日の目標

- HTML の基本構造を知る.
- コンピュータ内の文書は, 人間とコンピュータ (ソフトウェア) の両方が読む事を意識する.
- 文章の構造と見栄えの分離という考え方を理解する.

これから, 学生生活の上でも, あるいはその後社会人になってからも, 皆さんには, 「文書の作成」という業務が必然的に要求されます. 例えばそれは, レポートの作成であったり, 論文の作成であったり, 報告書の作成です. これらの作成には, 最近は通常コンピュータが使われます.

皆さんが作成する文書は, 単純に「文字列の並び」ではなく, それなりの構造を持っています. 例えば, 表題があったり, 章があったり, 節があったり, 結論部分があったりです.

コンピュータでこのような文書を作成する時には, その文書の構造に従った補助機能が利用できます. というより, むしろ積極的にその機能を利用して文書を作成します. ワードプロセッサでは, それはマウスを使った指示 (あるいはそれと同値なキーボードショートカット) で通常は実現されています. ワードプロセッサを使う際の最大の問題点は, 文章構造を意識するところが曖昧になる事です. なぜ文字の大きさや形を変えるかという理由を, 論理的に意識すべきなのです.

この講義では, 「文書の構造の記述をテキスト (文字列) でする」と言う事を体験していただきます. この講義では HTML を使いますが, 2 年次の「計算機概論 I」では, 数式の記述で絶大な威力を発揮する $\text{\LaTeX}$ で同じような内容を習う予定です.

2 用語の解説

WWW World Wide Web の略. 世界最初の Web browser が CERN(Conseil Européen pour la Recherche Nucléaire, 現在は L'Organisation européenne pour la recherche nucléaire, <http://www.cern.ch/>) で 1989 年頃から, Tim Berners Lee によって開発されたが, そのアプリケーションが WorldWideWeb.app (<http://www.math.u-ryukyu.ac.jp/~suga/www.png>). 開発者に敬意を表して, World Wide Web, WWW, W3, Web 等という. 日本で, 「インターネット」という言葉が使われている事が良くありますが, インターネットとはネットワーク同士をつなぐ事で, Web を見る以外の, もっと大きな内容を持つ言葉です.

HTML Hyper Text Markup Language の略. Hyper Text というのは相互参照をする文書の集合. World Wide Web で用いられる相互参照を記述するための言語. 現在は非営利団体 W3C (World Wide Web Consortium) で仕様が決められている.

http Hyper Text Transfer Protocol の略. Protocol とはここでは通信規約, すなわちネットワーク通信に於ける様々な約束事の事.

Web browser HTML 書かれた内容を解釈して, 画面に表示するためのソフトウェア. この講義では主に Firefox を使います. 過去に開発された Web browser が <http://browsers.evolt.org/> にまとめて置かれています. また, 今日は elinks というブラウザも使います.

最初の講義で紹介した本,

「ユニバーサル HTML/XHTML, 神崎正英著, 毎日コミュニケーションズ」

をこのテキストでは, 「参考書」と書いてあります. 必要となったら, 図書館で借りて, 参照するようにして下さい.

3 本日の実習

3.1 HTML について

まずは復習です. フォルダ「xxx のホーム」の中にある, 前回作成した index.html をダブルクリックして firefox で表示して下さい, さらに index.html を右クリックして gedit(GNOME テキストエディタの略) で開いて, 両方を見比べて下さい.

gedit で見える内容と firefox で見える内容が異なります. どちらも開いているのは index.html というファイルです. 同一の内容のデータ (ファイル) でもソフトウェアによって解釈が異なり, 表示が違って来る事を良く理解しておいて下さい.

このことは文字コードでも同じです. ASCII コードでは文字の 1 は数の 49(10 進) です. コンピュータ内の 49 という数を ASCII コードと解釈すると文字の 1 になるのです.

HTML は, Hyper Text Markup Language の名が示す通り, 一種の (Markup = 印付け) のための人工的な言語です. Hyper Text というのは相互参照を持つ文書の集まりの事で, 1960 年代にはこの概念は存在していました. この概念を考案したのは Douglas Engelbart という人で, マウスの発明家でもあります. この概念に Hyper text という名前をつけたのは Ted Nelson という人のようです (詳しくは, 参考書第 1 章 1.1, 1.2 を読んで下さい.). HTML 以前にもハイパーテキストを扱うソフトウェアは存在していました (有名なのは Apple の HyperCard). HTML がこれだけ浸

透したのは、ネットワーク経由でのアクセスを想定した事とともに、インターネットの発展と時期を同じにしたと言う理由です。

HTML は文書の内容とともに「文書の相互参照と論理構造」を記述していきます。Browser は、HTML に書かれた論理構造に従った表示をするように設計されています。従って、例えばもとの文章にあった改行は、browser でみると無視されます。改行は論理構造に関係ないからです。また、論理構造に従って文字の大きさを変えたり、改行したりリストの中黒・をつけたりしています。

もう1つ注意して欲しいのは、HTML は、テキストファイルを利用している事です。前回、テキストファイルと画像ファイルの大きさの比較をしましたが、大きさの桁が違っています。最近ネットワーク通信のスピードが速くなりましたが、それでも他のデバイス (DVD やハードディスク) から比べると遅いものです。その通信に、firefox で表示されている index.html の形の画像を送っていたのでは、効率が悪すぎます。実際、index.html の大きさは 1KB(キロバイト) 程度ですが、firefox で表示されるものを画像にすると、例えば PDF ファイルでは、50KB 程度になり、約 50 倍の大きさになります。通信回線は、多量のデータをやり取りするのに時間がかかる場合が、多くあります。そこで、通信データは少なくし、表示する方で工夫するという「分散処理」の考え方が、ネットワーク通信では基本です。

HTML にはいくつかのバージョン (版) がありますが、この講義では 2014 年に確定した最新バージョン HTML5 に基づいた話をします。HTML 4.0 の仕様策定の頃から、見た目ではなく、browser や使う人に対する依存性をなくする事が重視されています。

HTML では、論理構造と相互参照はタグ (tag) によって記述されます。タグは不等号 <, > で囲まれます。タグは基本的に開始タグと終了タグがあります。例えばパラグラフ (段落) の開始タグは、<p> でそれに対する終了タグは</p>です。タグは大文字、小文字の区別がありません。<html> と<HTML> は同じ意味です。index.html を見ればわかりますが、タグは入れ子構造で使います。

HTML の開発動機は、効率的な情報共有です。この考え方は今も変わっていません。このことも頭の隅に入れておいて下さい。

index.html で使われているタグ (要素)

詳しくは、参考書 2.1, 2.2(pp 13-23) を見て下さい。参考書は、HTML 4.01 に基づいているため、少し変化している部分があります。

<html> - </html> HTML 文書全体の開始と終了。lang="ja" は、このページが日本語を使っている事を宣言しています。タグには、その要素に対して固有の役割を持たせる事が出来ます。これを「属性値 (attribute)」と言います。属性値は、属性名="値" という形 (=と"で囲む) で指定します。

<head> - </head> Header(ヘッダ、適切な日本語訳は無い)。ブラウザやサーチエンジンなど、この文書を利用するソフトウェア (まとめてユーザーエージェントという) に対して、補助的な情報を記述する。

<meta ... > meta 要素タグ。Header の中で文書の付加的な情報を記述し、ユーザーエージェント (ブラウザやサーチエンジン) 等に対して、その動作を指示するために使う。index.html では、文書の文字コードの情報を与えるものを記述した。終了タグは無し。

`<link ... >` 上と同様, ユーザーエージェントに対してこの文書とリンクしてるファイルや, 補助的に用いるファイルを記述する. 終了タグは無し. `index.html` では, スタイルシート (後述) の指定をしてある.

`<title>` - `</title>` 文書の表題. グラフィカルなブラウザでは, タイトルバーにここの文字列が現れる事が多い.

`<body>` - `</body>` 文書の本体 (本来の内容)

`<h1>` - `</h1>`, `<h2>` - `</h2>` 見出しタグ. 数字は, 見出しのレベル. `<h1>` から `<h6>` まで 6 つのレベルがある. グラフィカルなブラウザでは, レベルに応じて文字の大きさが変わるが, 文字の大きさを変えるために使うのではなく, 文書の中での意味としての見出しとして用いる.

`<hr>` 水平線を引く. 閉じタグは無し.

`<p>` - `</p>` パラグラフ (段落). 終了タグは省略可. 段落と言う概念を持っていない人が多いようですが, どんな文書も「段落の集まり」で形成されています. 文書を作るとき (今の皆さんだとレポート課題が多いと思いますが) には, 常に「段落」を意識して作ってください.

`<ul>` - `</ul>` Unordered list (番号の付かない箇条書). 箇条書に付いては, 後の講義でもう 1 度取り上げる.

`<li>` - `</li>` List item (箇条書の中のリスト)

index.html で使われているタグでない html の要素

`<!DOCTYPE html>` 文書型定義 (Document Type Definition) HTML5 では, 単純に HTML と書くだけ. 以前は, HTML のどのバージョンかの記述があった.

`<!-- ... -->` コメント. 複雑な処理等をする場合に, それを作った人が, 処理内容などについてのメモ書きを残すための仕組み. 多くの場合, 中身を改変する際に自分自身へのメッセージとなる. ブラウザはこの部分を完全に無視する.

コメントという仕組みが最初は理解できないかも知れませんが, プログラミング言語や, 表計算ソフト等複雑な処理をする処理系では, 必ず 1 部分を処理系が無視をしてくれるようにする仕組みがあります. これは, その処理内容を読む人間のための仕組みです.

3.2 テキストベースブラウザを使う

Web ブラウザは Firefox や Safari の様なグラフィカルなものだけではなくありません. 前回もいいましたが, グラフィカルな環境が使えない事もあります. 例えば, この講義の Web ページや情報処理センターの Web ページには, 個人情報保護の観点やセキュリティ上の問題から, 大学外からの閲覧を制限してる情報があります. では, それらは本当に大学外から見えないかというと, 琉球大学のアカウントとネットワーク環境とテキストベースブラウザがあれば, 読む事ができるのです. 問題はどうやって大学のマシンを大学外から利用するかという事ですが, それについてはこの講義では触れません. 興味のある方は調べて下さい. 2 年次向けの計算機概論 I では, 講義します. ここでは, CentOS に入っている `elinks` というテキストブラウザを使ってみます. 次の操作をして下さい.

1. Gnome 端末を起動する.
2. elinks index.html (elinks と index.html の間には空白を入れる.) と押して ENTER キーを押すと, elinks が起動して, 今まで作った index.html が読めます.

カーソル移動は, カーソルキー (矢印の付いたキー) でできます.

他の URL に移るには, 大文字の G (Shift + G) キーを押します. Enter URL: とでますから, URL (例えば `http://www.math.u-ryukyu.ac.jp/~suga/joho/2017`) 入力して ENTER を押すとそこに飛びます. リンクのページに飛ぶには, そのリンクの場所にカーソルを持って行き, ENTER を押せばリンク先に飛びます. 1 つ前のウェブページに戻るには, 大文字の B キーです.

終了は小文字の q キーで, 押すと終了するかどうかを尋ねられますから y を押します.

3.3 スタイルシートを書く

HTML は, 文章の内容を論理構造を表すタグとともに記述するように考えられており, その見栄えや表現方法は browser に任せるようにしています. これは, Web をアクセスする人が, さまざまな環境である得る事を想定した仕様で, 当然の事です.

しかし, 多くの場合 Web にアクセスするのはグラフィカルな browser で, 文書の多彩な表現が可能です. このような環境に対して内容の色や配置を指定するには, スタイルシートを利用します. この講義では, 最も標準的に用いられている カスケイディングスタイルシート (Cascading style sheet, CSS) を利用します. スタイルシートには, CSS の他にも XSL (Extensible Stylesheet Language) などいくつかの種類があります.

gedit を起動します. メニューを「表示→強調モード→その他→CSS」とたどってスタイルシートの入力モードにします. 下の 4 行の内容を index.css という名前のファイルで保存して下さい.

```
body {background-color: rgb(224,255,224)}
h1 {color: maroon; text-align: center}
h2 {color: #000088}
p {text-indent: 1em}
```

保存が終わったら, Firefox で再読み込みボタン (青色の円形の矢印のついたもの) を押して, 表示がどのように変化するかを見て下さい.

もし何も変化しなければ, ファイル名が index.css となっているか, index.html の 上から 7 行目付近の `<link rel="stylesheet" type="text/css" href="index.css">` の部分が間違っていないかを確かめて下さい.

CSS は, 次のような形で記述されます.

```
セレクタ { 種類: 指定値; 種類: 指定値; ... }
何の どの部分を どの値に ...
```

複数の部分を指定する場合には, ; で区切って書きます. 例えば上で,

```
h1 {color: maroon; text-align: center}
```

としてありますが, 第 1 レベルの見出しの色は maroon にし, そのテキストは画面の中央に配置すると言う意味です.

また, 色指定は 3 通りを例示してあります. `rgb(224,255,224)` と `#E0FFE0` は同じ意味になります. どちらも光の 3 原色赤 (red), 緑 (green), 青 (blue) の輝度を, 数値で指定しており, 始めの方

は 10 進で分離して書いてあり、後の方は 16 進でまとめて書いてあります。輝度の値の範囲は $0 \sim 255 (= 2^8 - 1)$ です。代表的な色は、色の名前 (例えば maroon=栗色) で指定する事も出来ます。html で定義されている色の名前はそう多くなく 16 色で、参考書の p79 にあります。これ以外の色の名前の定義されており、グラフィカルなブラウザでは、その名前を使ってもほとんどの場合は、大丈夫です。ただし、色の名前については自分で調べて下さい。

最後のセクタ p の指示での text-indent は、段落の最初の字下げの指定です。1em は 1 文字と言う単位の指定です。単位の指定方に関する詳しい事は、参考書を見て下さい。

スタイルシートを書いた後のページも、elinks を使って見て下さい。文字の色を変えるとか、文字を中央に置くとかの指示は、elinks でどのように反映されるか、反映されないかを見て下さい。elinks では、見出しタグに対して、文字の大きさが変わらないことも、把握して下さい。ウェブページを書き換えた時には、必ず elinks でどう見えるかもチェックするようにして下さい。

HTML のタグやスタイルシートの指定などは英語がもとになっていますが、この程度の英単語の意味は、各自調べておくとともに、この際覚えてください。(会話よりもむしろこういうものの方が、使える英語だと私は考えます。)

次に行く前に、一旦 gedit は終了して下さい。

3.4 Web ページを公開してみる

前回も言いましたように、今のままではネットワークを使って皆さんのウェブページにアクセスする事は出来ません。それが出来るようになるには、Web サーバと呼ばれるコンピュータの然るべき場所にファイルを置いて、アクセスできるように然るべき設定をする必要があります。

いわゆるプロバイダを利用する場合には、ファイル転送ソフトを用いて Web ページのファイルを転送しますが、琉球大学の情報処理センターでは、皆さんのホームディレクトリ (フォルダ) は、Web サーバからアクセスできるような設定になっています。このホームディレクトリは、「xxx のホーム」ではなく、3 回目の授業で述べた、「Linux」の 1 つ上のホームです。ホームディレクトリの全てのファイルが、Web サーバ越しに見えたのでは、(公開しない) 個人的なファイルが置けませんので特定のディレクトリ (フォルダ) 内だけが Web サーバ越しに見えます。情報処理センターの場合、ホームディレクトリの中の「WWW」(W は大文字) というディレクトリに HTML 関連ファイルを置いて、そのディレクトリが Web サーバのソフトウェアから読み取り可能にしなければなりません。3 回目の講義で、このフォルダはすでに作成済みですが、欠席をしてまだ作成していない人は、<http://www.math.u-ryukyu.ac.jp/~suga/joho/2017/03/node7.html> に従って作成して下さい。作成の際には、ページの先頭行から書かれている内容を正確に実行して下さい (途中から実行してはダメ)。

Web サーバが、このフォルダを読めるようにするには、次のようにします。

1. Gnome 端末を開く

(a) `cd ..`

としてエンターを押す (..`はピリオド 2 つの連続, cd と..の間はスペース (半角の空白) が入る).`

(b) `pwd`

としてエンターを押すと

```
/rains/eve/e-sci/e1731xx
```

最後に自分の login 名があらわれることを確認する. このようになっていたら (c) に進む
もしそうになっていないなら,

```
cd
```

としてエンターを押し (a) にもどる.

(c) `chmod 711 .`

としてエンターを押す (`はピリオド 1 つ, chmod と 711 の間, 711 と . の間にはスペース (半角の空白) が入る).`

2. `chmod 711 WWW` としてエンターを押す.

3. デスクトップの「xxx のホーム」をダブルクリックして開き, 「index.html」と「index.css」をデスクトップに作った「WWW へのリンク」にドラッグ&ドロップで移動する.

4. デスクトップの「WWW へのリンク」をダブルクリックし, index.html を右クリックして, プロパティのアクセス権のタブを選び, グループ, その他の「アクセス」の欄を「読み込みのみ」になっているかを確認する. もしなっていないければ, そのように変更する.

5. index.css についても index.html と同じアクセス権になっている事を確認する.

上が全部終わったら, firefox で URL を入力する所に, `http://www.cc.u-ryukyu.ac.jp/~e1731xx` (1731xx は自分の学籍番号を入れる) と入力してみてください. それは, ネットワーク越しにアクセスしたページです. 今までは, 自分のファイルを Firefox が直接読んでましたが, この読み方では, 一旦琉球大学の Web サーバが読み, さらにネットワークを通じて Firefox が読んでいます. この状態で, ネットワークに繋がった世界中から読む事ができるようになっています. (実際に読みに来る人がいるかどうかは, また別な問題.) この方法で, 自分のページが見えない人は, 手を挙げて私か TA の瑞慶覧君を呼んでください.

自分のページが見えたら, 隣の人の学籍番号に変えてみて他の人のページも見れるようになって
いることを確認して下さい. 携帯電話を持っている人は, 携帯電話の Web ブラウザで, 自分のページ
を見てください.

ウェブページの公開を中止する

ウェブページを公開を中止するには, WWW というディレクトリを別の名前に変えると言うのが, 1 つの方法です.

これ以外にも, ディレクトリ WWW のアクセス権で, グループとその他のアクセス権をなしに
すると言う手もあります.

特定の場所からはアクセスを許すが他からは許さないとかの設定も可能で, これをする場合には,
.htaccess というファイルを用います. これに関しては, 後の講義で触れます.

3.5 残りの時間の課題: Web ページの改良とバックアップ

残った時間で、次の事をやって下さい。

1. index.html のプロフィールの下に簡単な自己紹介を付け加えるなどして、内容を充実させて下さい。ただし、あまり個人が特定できる内容は書き込まないで下さい。
2. バックアップ (現時点で、USB メモリの使い方に対する情報処理センターの回答はありません。)

時間に余裕がある人は、index.css を変更してみてください。ただし、見づらい色使いにならないよう注意して下さい。文書作成時には、次の事も注意して下さい。

HTML 文書作成上の注意 (規格外と非推奨)

講義で紹介していないタグも、使いかたと意味を調べて積極的に使ってみてください。参考書の 2 章、3 章に (HTML 4.01 版ですが) 詳しく書いてあるはずですが、いろいろなウェブページの HTML を参考にするのも 1 つの手です。Firefox の「表示」メニューから、「ページのソース」を選ぶと、今見ているページの HTML が表示されます。

ただし、WWW が成長して行く過程で、さまざまな独自拡張のタグが現れました。例えば「文字列を点滅させる」という独自拡張が以前はありましたが、現在では HTML 規格外です。最初の方でも述べましたが、HTML の考え方は文書の論理構造と相互参照なので、「文字列を点滅させる」のような概念は規格に入りません (文字列を強調するタグとして、`<strong>` や `<em>` がある)。規格外のタグを使うと、ブラウザでの表示は保証されませんから、使うべきではありません。HTML の (古い) 参考書の中には、このような内容のことを書いてあるものもありますので、本を選ぶときには注意してください。指定した参考書には、HTML 4.01 の正確なタグの解説が書いてありますので、使う前に参考書の解説をまず読んで下さい。HTML5 についても幾つかの解説書がでています。どの本が良いのかは、今のところ判断ができませんが、「初心者向け」を謳っている本は、「悪い本」の割合が高いのは、昔からです。

同様に HTML 文書内で「文字の大きさや色を変更する」と言うような属性指定は、非推奨です。HTML5 では、HTML 本体に見せ方を指定せず、見せ方の指定はスタイルシートと browser にまかせます。これも文書の論理構造や相互参照と関係ないからです。HTML は目印をつける (markup) ための物で、レイアウトやデザインを整えるための物ではありません。また、上で述べた相互運用の概念に反しています。目の見えない人、色の区別のつかない人が世の中にはいらっしゃいますが、これらの人たちへの配慮も必要です。目の見えない人は文書を点字印刷したり、音声に変換して読む事ができます。色の区別が付かない人は、スタイルシートを無効にして自分に都合の良い表示で読む事ができます。レイアウトや配置は、スタイルシートを使うべきです。

なお、HTML5 の策定は終了しており、今流通している主な browser の多くは、この主要部分には対応しています¹。特に、スマートフォン向けの Web は、HTML5 が主流になっていますので、HTML5 のタグを (自分で調べて) 利用する事は、積極的に行ってください。

¹完全に HTML5 に準拠したブラウザを作るのは、マルチメディア対応の部分で大変だと思います