情報システム演習 2：D 分野

執筆

河原和好

はじめに

この演習では，D 分野（コンピュータと通信）で主に学習する情報システムの開発において必要なコンピュータと通信の技術について，ウェブサイトの作成を通して基本的な内容を具体的に学習する。

まず，ウェブの仕組みを理解し，実際に HTML とスタイルシート(CSS)を用いてウェブページを作成する手法を学習する。

次に，JavaScript によるプログラミングを取り入れたウェブサイトの制作について学習し，さらにフォームについても学習する。

最後には個人で何らかのウェブサイトを作成し，発表することで，学習の成果を確認する。

学習する内容

この演習で学習する内容は以下のとおりである。

1. WWW のしくみ
2. HTML
3. CSS
4. JavaScript
5. フォーム
6. ウェブサイト作成

演習の進め方や内容は進み具合により調整する場合があります。

1. WWW のしくみ

この章では，本演習で使用する用語について説明する。

1.1 インターネット

インターネットとは，全世界のコンピュータのネットワークを相互に接続した巨大なコンピュータネットワークのことである。元はアメリカ国防総省高等の研究計画局(ARPA)の分散型コンピュータネットワーク(ARPANET)であり，軍事目的で研究されていた分散型ネットワーク（全体を統括するコンピュータが存在しないネットワーク）であった。当初のインターネットは学術機関における電子メールなどで利用されていたが，WWW(World Wide Web)が開発されてから利用が拡大し，世界規模の情報通信インフラとなった。通信プロトコルには TCP/IP を使用している。

インフラ：インフラストラクチャー(infrastructure)の略。基盤や下部構造などの意味を持つ英単語で，一般的には上下水道や道路などの社会基盤のことを指す。IT の世界では，何らかのシステムや事業を有効に機能させるために基盤として必要となる設備や制度のことを指す。

プロトコル：一般的には複数の者が対象となる事項を確実に実行するための手順等について定めたもののことで，日本語では規定，議定書，儀典と訳される。コンピュータの通信の場合は，コンピュータ同士で通信を行うための取り決めのことを表し，日本語では通信プロトコル，通信手順，通信規約という。これから取り上げる，TCP/IP, HTTP, FTP, Telnet などがこれに当たる。

TCP/IP：Transmission Control Protocol / Internet Protocol の略。インターネット上で通信を行う際のプロトコルのこと。デファクト・スタンダード（事実上の標準）になっている。仕様が RFC（Request For Comments）として公開されている。コンピュータメーカー固有のプロトコルでないため，誰でもコンピュータに組み込むことができる。ネットワーク利用者が使いながら改良，開発して来たプロトコルなので，使い勝手がよく，幅広い互換性をもち，日々進歩しているプロトコルである。動作検証済，動作実験済のソフトウェアがすぐに入手できるという利点がある。

1.2 インターネット上のサービス

インターネット上では，以下に挙げるような様々なサービスが提供されている。優良なもの，無料なもの，登録制など利用が制限されているものなど，様々なものがある。

- WWW（ウェブページの閲覧）
- 電子メール(E-mail)
- ファイル転送
- リモートアクセス
- SNS
- ショッピングサイト
- ネットオークション
- ネットバンキング

本演習ではこれらのうち，WWW について主に学習する。

1.3 WWW

WWW とは，World Wide Web の略語であり，ワールドワイドウェブ，略してウェブという。インターネットで標準的に用いられるドキュメント（文書）システムで，インターネットを通してドキュメント（文書以外に画像や動画なども）をやりとりすることができる。ウェブ(web)は蜘蛛（クモ）の巣の意味で，ネットワークが蜘蛛の巣に見えることから来ている。

欧州核物理学研究所(CERN)の Tim Berners-Lee 氏が所内の論文閲覧システムとして 1989 年に考案したシステムで，HTML(Hyper Text Markup Language)という言語を用いて文書の論理構造を記述する。文書内に文字以外に画像や音声などのデータや，他の文書の位置などを埋め込めるハイパーリンクという仕組みが用いられている。

1.3.1 ブラウザ

ブラウズ(browse)とは「閲覧する」という意味の英単語で有り，ブラウザ(browser)とは，一般的には何らかの情報を閲覧するためのソフトウェアのことを指すが，現在はウェブのドキュメント（ウェブページ）を見るためのソフトウェアであるウェブブラウザ（WWW ブラウザ）のことを指す。

主なブラウザとしては，Internet Explorer, Mozilla Firefox, Google Chrome, Safari などがある。ブラウザの種類によって見た目が変わったり機能が変わったりするので注意する必要がある。

1.3.2 ホームページ

ホームページという言葉の本来の意味は，ブラウザを立ち上げたときや，ホームボタンを押したときに表示されるページ（スタートページ）のことであったが，これが転じてトップページ（企業やウェブサービスのサイトの中で一番上のページ）を「ホームページ」というようになり，さらに現在では，一般的なウェブページのことを「ホームページ」と呼ぶようになった。

ホームページを HP と略す(Home Page の略)場合も多くなってきているが，英語では PC メーカーであるヒューレット・パッカード社(Hewlett Packard)を公式に HP と略すため，と紛らわしい場合があるので注意が必要である。

1.3.3 URL

URL とは，Uniform Resource Locator の略語であり，インターネット上に存在する情報資源（リソース，文書や画像などのデータ）の場所を指し示す（ロケーター）記述方式である。ブラウザにこの URL を入力したり，ハイパーリンクで URL を指定したりすることにより，その場所にある情報資源をブラウザ上に表示させることが出来る。

URL の例）http://www.nuis.ac.jp/~kawahara/index.html

- http　　使用されるプロトコル
- www　　サーバ(コンピュータ)名
- nuis　　新潟国際情報大学を表す
- ac　　大学などの高等教育機関を表す
- jp　　日本国内にあることを表す
- ~kawahara　ユーザ名
- index.html　ファイル名

1.3.4 HTTP

HTTP とは，HyperText Transfer Protocol の略語であり，ウェブサーバとクライアント（ウェブブラウザなど）との間で，データ（ハイパーテキスト：HTML など）を送受信する際に使われるプロトコルである。データの送受信はクライアントサーバモデルを使用している。

1.4 クライアントサーバモデル

サーバ（情報資源を集中管理し，その情報を提供するコンピュータ）とクライアント（情報を提供され，それを利用するコンピュータ）が接続されたコンピュータネットワークの仕組みことで，クラサバ，クライアントサーバシステ

ムともいう。例として，ウェブ，メール，プリンタ，ファイルのシステムがある。

ウェブの場合，サーバはウェブサーバ（WWW において情報送信を行うコンピュータ，またはその機能を持ったソフトウェア）で，クライアントはウェブブラウザ（ブラウザ，ウェブページを見るためのソフトウェア）となる。

ウェブプログラミング（ウェブの仕組みを使ってプログラムを作成すること）を行う際，プログラムがクライアント側で動作する場合と，サーバ側で動作する場合があり，それぞれ，クライアントサイド，サーバサイドという。

1.4.1 クライアントサイド

通常のウェブページが表示される仕組みと同様で，ウェブサーバから提供されたデータをウェブブラウザに表示する仕組みがクライアントサイドとなる。HTML ファイルの中にプログラムがある場合（スクリプト）や，ダウンロードモジュールとして別ファイルとなっている場合があり，クライアントであるウェブブラウザがプログラムを解釈して実行する。

表 1-1 クライアントサイドの例

おもな技術	おもな開発言語
文書作成	HTML, CSS
スクリプト	JavaScript, VBScript など
ダウンロードモジュール	Flash, Java アプレットなど

- スクリプト : HTML の中にプログラムが記述され，ウェブブラウザ（クライアント）がそれを解釈して実行する仕組みのこと。
  例）JavaScript, VBScript
- ダウンロードモジュール : HTML と一緒にプログラムをダウンロードし，ウェブブラウザ（クライアント）にインストールしてあるプラグインがそれを解釈して実行する仕組みのこと。プラグインがインストールされていないと実行できない。リッチクライアントともいう。
  例） Flash, Java アプレット

本演習では，このクライアントサイドのスクリプト(JavaScript)を学習する。

1.4.2 サーバサイド

クライアントからの要求（リクエスト）によってサーバ上でプログラムを実行し，その処理結果を HTML ファイルとして応答（レスポンス）しクライアントに送る仕組みがサーバサイドである。HTML ファイルの中にクライアントサイドで動作するウェブプログラムを含めたり，ダウンロードモジュールというプログラムを出力したりする場合がある。

表 1-2 サーバサイドの例

おもな技術	おもな開発言語
CGI	Perl, C 言語, Ruby など
PHP	PHP 言語
JSP, サーブレット	Java
ASP, .NET	Visual Basic, C#など

本演習ではサーバサイドについては触れない。授業においては，情報コースの専門演習において，サーバサイドの演習を行う予定である。

1.5 ウェブプログラミングの基礎

通常のウェブページは，表示するたびに同じ内容の文書が表示される。このようなウェブページを「静的」なページと呼び，後述する HTML と CSS を用いて作成することができる。

これに対し，表示する時に変化するようなページを作成することができる。このようなウェブページを「動的」なページと呼ぶ。後述する JavaScript などを用いることによって作成することができる。

プログラムによって動的なウェブページを作成することを「ウェブプログラミング」という。

ウェブプログラミングの例としては，アニメーションのような動きのあるウェブページ，インタラクティブ（双方向性。ユーザの操作に対し何らかの反応が返ってくる）なウェブページ，掲示板，チャット，ブログ，オンラインショップサイト等が挙げられる。

1.5.1 プログラミングとは

コンピュータに行わせる処理の手順を記述したものを「プログラム」といい，人間がプログラムを作成することを「プログラミング」という。プログラムはプログラミング言語（JavaScript, Java, C 言語など）という言語を用いて記述する。ウェブプログラミングの場合は，HTML と CSS に JavaScript などのプ

ログラミング言語（スクリプト言語）を組み合わせて行う。

1.6 本演習で使用するソフトウェア

この演習では，HTML ファイルを作成するためにテキストエディタ TeraPad を使用し，HTML ファイルを表示するためにウェブブラウザ Internet Explorer(IE)または Mozilla Firefox を使用する。ファイルをウェブサーバに転送する場合は，FTP クライアントソフト FFFTP を使用する。

全てのソフトウェアは情報センターのコンピュータにインストールされており，使用することが出来る。また，IE は Windows パソコンに標準で付属しており，他の TeraPad，Firefox，FFFTP はフリーソフトなため，個人でダウンロードして使用することが出来る。

1.6.1 テキストエディタ

どの演習も，まずはテキストエディタを用いて HTML 等を入力することから始まる。本演習で使用する TeraPad では，編集モードに HTML モードがあるので，最初に指定しておくと作成しやすくなる。

また，最初に保存する場合，「ファイル」メニューから「文字／改行コード指定保存」を選択して文字コード「UTF-8N」を指定して保存する必要がある。これは，HTML5（HTML のバージョン 5）において UTF-8（Unicode という文字コード）で文書を作成することが推奨されているからである。

保ファイルの拡張子として .html（ドット HTML）を付ける。 .htm（ドット HTM）でも動作する。ファイル名に日本語を使用すると動作しない場合もあるので，できるだけファイル名は半角英数字を使用するようにする。

1.6.2 ウェブブラウザ

作成した HTML ファイルは，ウェブブラウザで表示させて確認する。本学の PC 環境では，IE と Firefox を使用することが出来る。

表示する方法は，TeraPad にあるブラウザのボタンを押すか，ファイル自体をダブルクリックする（IE で表示）か，ファイルをブラウザにドラッグ＆ドロップするか，で行う。

一度表示させているファイルを修正して保存した場合，ブラウザの更新ボタンを押すことで，修正した内容が反映される。

1.6.3 FTP クライアント

ブラウザで表示させることが出来た HTML ファイルは，個人の環境で表示できるだけである。授業を進めていく上ではこのままでも構わないが，作成した

ウェブサイトを教員も含め他の人が見られる状態にするためには，WWW の仕組みを使ってインターネット上で見られるようにする必要があり，ファイルをウェブサーバにアップロードしなければならない。そこで使用されるソフトウェアが FTP クライアントソフトであり，本学では FFFTP というソフトを使用することができる。

FFFTP を起動するとユーザ ID とパスワードを聞いてくるので入力しログインする。成功すると画面が表示される。

左側が転送元（ローカル，自分が使用しているコンピュータ）を，右側が転送先（リモート，今回はウェブサーバ）を表示している。

ローカルのファイルをドラッグ＆ドロップすることにより，サーバにファイルを転送すること（アップロード）が出来る（メニューやアイコンからも行うことが出来る)。逆の操作で，サーバからローカルにファイルを転送することも出来る（ダウンロード)。

他にも，フォルダの作成やファイルの削除なども行うことが出来る。

演習の進め方

準備

HTML ファイル等は各自の USB メモリに作成するので忘れずにもってくること。また，演習用のフォルダを作成してファイルをまとめておくと管理しやすい。

手順 1 HTML ファイルの作成

テキストエディタ TeraPad を用いてファイルを作成する。

手順 2 HTML ファイルの確認

ブラウザを用いてファイルを表示させる。表示されなかったり，動作が変なところがあったりしないか確認し，不具合がある場合はテキストエディタで元のファイルを修正する。

通常の演習はここまでで構わない。前述の通り，作成した Web サイトを他の人からも閲覧可能にするためには，以降の手順を行う必要がある。

手順 3 HTML ファイルのアップロード

動作が確認できたファイルを FTP クライアント FFFTP を用いてウェブサーバに転送する。本学では，学生がウェブページを作成して公開できるような設定がなされているので，リモート側の public_html というフォルダにファイルを転送すればよい。

USB メモリの場合と同様で，直接ファイルを転送するよりも，フォルダを作成してそこにファイルを転送するようにした方が管理しやすい。

フォルダの作成については，各先生の指示に従うこと。

手順 4 アップロードしたファイルの確認

ブラウザを起動し，URL 欄（アドレス欄）に以下の URL を入力し表示する。

http://www. nuis.ac.jp/~sxxxxxxx/ファイル名

sxxxxxx の所は各自のユーザ ID を入れる。フォルダを作成している場合は，ファイル名の前にフォルダ名を書き，続けて / の後にファイル名を書く。

注意

ウェブサーバに転送することにより，全世界からアクセスできることになるので，著作権には気をつけること。これに関する事については，「学生 Web ページ利用ガイドライン」を熟読し遵守すること。

http://www.ic.nuis.ac.jp/sis/guide/webguideline/

2. HTML

この章では，ウェブで文書を表示する際の一番の基本となる HTML について学習する。

2.1 HTML とは

Hyper Text Markup Language の略で，ウェブページを記述するためのマークアップ言語のこと。

- ハイパーテキスト：文書の中に他の文書の位置情報(ハイパーリンク)を埋めこみ，複数の文書を相互に連結できるようにした文書。リンクによって画像や音声なども埋め込むことができる。
- マークアップ言語：文書の一部にタグと呼ぶ特殊な文字で囲んだ部分を付加することによって，文書の構造や修飾情報を記述する言語のこと。

1993 年に公開され，現在ではインターネットで文書をやりとりする際の標準になっている。

HTML の仕様は W3C(World Wide Web Consortium)によって勧告されていて，以下のような仕様が存在する。

- 1993 年　HTML1.0
- 1995 年　HTML2.0
- 1997 年　HTML3.2
- 1997 年　HTML4.0
- 1999 年　HTML4.01
- 2014 年　HTML5
- 2016 年　HTML5.1

本演習では，HTML5 について学習していく。

2.2 HTML の基本構造

HTML の基本構造は以下のようになっている。

- 文書型宣言
- <html>～</html>　HTML 要素
  <head>～</head>　head 要素（設定などを書く）
  <body>～</body>　body 要素（本文を書く）

以下で説明する。

2.2.1 文書型宣言(Document Type Definition, DTD)

マークアップ言語において，文書構造（文書型）が決められた規則のことで，どの様な要素があるのか，どの様な順序で書けばよいか，どの様な属性が指定できるか，などが決められている。

HTML の場合，先頭行に書くことになっていて，ブラウザがこれを見て HTML 文書がどのバージョンで書かれているかを判断している。

HTML5 においては，文書型宣言は以下のようになる。

```
<!DOCTYPE html>
```

2.2.2 html 要素

HTML において開始タグと終了タグで囲まれたものを「要素」という。開始タグは<と>で囲まれ，終了タグは</と>で囲まれる。

html 要素は一番基本となる要素で，この中に head 要素と body 要素を記述する。以下のように言語を指定して書く。

```
<html lang="ja">        標準言語の指定（日本語）
```

2.2.3 head 要素

HTML に対する指定を記述する要素で，以下のような指定が出来る。

title 要素

文書のタイトルを指定する。

meta 要素

ページがどんな性質であるかという情報を設定する。

文字コード

コンピュータでは文字を表現するために数字（コード）を使っている。日本語は，ISO-2022-JP(JIS コード)，EUC-JP(EUC コード)，Shift_JIS(シフト JIS コード)の 3 つがこれまでは使われていたが，最近は UNICODE（ユニコード，UTF-8, UTF-16）が使用されている。

- ISO-2022-JP　　：インターネットで使用
- EUC-JP　　：UNIX で使用
- Shift_JIS　　：PC(Windows)で使用
- UNICODE (UTF-8)　　：世界中の文字を統一して使うことができる

HTML5 では文字コードとして UTF-8 が推奨されている。文字コードの指定は，meta 要素を用いて，<meta charset="UTF-8">のように指定する。UTF-8 の文字コードは，エディタで保存するときに指定する。

2.2.4 body 要素

HTML の本文を記述する要素である。実際にどの要素があるかについては後述する。

2.2.5 HTML 文書の構成

HTML で文書を作成する際は，見出しや段落を用いて文書の構造（階層構造）を意識して作る必要がある。歴史的な経緯から，HTML でデザイン（文書の修飾情報）を指定することもできるが，デザインは CSS（スタイルシート(次回)）を用いることになっているからである。

2.3 HTML の書き方

マークアップ（意味付け：文書の構造や修飾情報の記述）は文書を要素で区切って行う。

・ 要素は，タグを使って記述（開始タグと終了タグ）する。
  <タグ名 > ・・・ </タグ名>
  例）<title>**文書のタイトル**</title>

・ 終了タグが無い要素もある
  例）

・ 必要であれば，要素に属性を記述することができる
  <タグ名 属性名="属性値"> ・・・ </タグ名>
  例）HOME

2.3.1 基本的なタグ

ここに挙げるもの以外にも様々なタグが用意されている。

・ <h1>～<h6> 見出しを作成するタグ。数字が小さいほど見出しは大きくなる。文字の大きさ指定では無いので注意（文字を大きくするためにこのタグを使用するのは良くない）。
・ <p>　段落を作成するときのタグ。
・
 改行を作成するときのタグ。終了タグは不要。

・ 番号なしの箇条書きを作成するときのタグ。項目は<li>タグで指定する。<li>タグの終了タグは省略可能である。
・ 番号つきの箇条書きを作成するときのタグ。項目は<li>タグで指定する。<li>タグの終了タグは省略可能である。

例）rei2-1.html

2.4 ハイパーリンク

文書の中に，参照先の文書の情報をリンクとして埋め込んでおくことにより，ブラウザで表示した際に下線で表示され，クリックすると参照先の文書にジャンプすることができるしくみのことである。a タグを使用して以下のように記述する。

・ 例 1：同じフォルダにある他の文書へのリンク
  次の文書
・ 例 2：他の URL へのリンク
  NUIS

例）rei2-2.html

2.5 画像の表示

画像や音楽などのマルチメディアファイルへのリンクを HTML 内に組み込むことができる。一般のブラウザにおいては，対応しているフォーマットであれば画像は表示できる。動画や音楽の再生機能は無いので，これらを補助的に再生するソフトウェアとして，プラグインというソフトウェアをブラウザに組み込むことによって再生していた。HTML5 になって動画や音楽の再生も対応するようになったが，ブラウザによって動作が異なるので注意が必要である。

画像を表示するには，img タグを使用して記述する。

・ 例 1：文書と同じ場所にある画像へのリンク
  <img src="image1.jpg">

・ 例 2：URL を利用した画像へのリンク

<img src="http://www.nuis.ac.jp/pub/common2/objfiles/f01_1426732404457302_img.jpg">

例）rei2-3.html

2.6 テーブル（表）の作成

table タグを使用することで，テーブル(表)を作成することができる。オプションとして，border="1"と書くことにより，罫線を追加することができる。

- tr タグ：行を追加する
- td タグ：セルを追加する
- th タグ：見出しのセルを追加する（省略可能）
- caption タグ：タイトルや説明を追加する（省略可能）

例）rei2-4.html

2.7 サンプル

rei2-1.html 基本タグの例

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>HTML1</title>
</head>
<body>
<h1>自己紹介</h1>
<p>
はじめまして。<br>
よろしくお願いします。
</p>
<h2>番号なし箇条書き</h2>
<ul>
<li>あああ</li>
<li>いいい</li>
<li>うううう</li>
</ul>
<h2>番号つき箇条書き</h2>
<ol>
<li>あああ</li>
<li>いいい</li>
<li>うううう</li>
</ol>
</body>
</html>
```

rei2-2.html ハイパーリンクの例

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>HTML2</title>
</head>
<body>
<h1>リンク</h1>
<ul>
<li><a href="rei2-1.html">基本タグ</a></li>
<li><a href="rei2-3.html">画像表示</a></li>
<li><a href="rei2-4.html">テーブル作成</a></li>
<li><a href="http://www.nuis.ac.jp/">大学ホームページ</a></li>
</ul>
</body>
</html>
```

rei2-3.html 画像表示の例

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>HTML3</title>
</head>
<body>
<h1>画像の表示</h1>
<p>画像の表示には img タグを使う</p>
<h2>ファイルで画像を指定</h2>
<p>
image1.jpg の所を各自で用意した画像ファイルに変えてください。
</p>
<img src="image1.jpg">
<h3>URL で画像を指定</h3>
<img src=
"http://www.nuis.ac.jp/pub/common2/objfiles/f01_1426732404457302_img.jpg">
</body>
</html>
```

rei2-4.html テーブル作成の例

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>HTML4</title>
</head>
<body>
<h1>テーブル(表)の作成</h1>
<table border="1">
<tr><td>111</td><td>222</td><td>333</td></tr>
<tr><td>444</td><td>555</td><td>666</td></tr>
</table>
<h2>説明</h2>
<table border="1">
<caption>タグ一覧</caption>
<tr><th>タグ</th><th>役割</th></tr>
<tr><td>table</td><td>テーブル(表)を作成</td></tr>
<tr><td>tr</td><td>行を追加</td></tr>
<tr><td>td</td><td>セルを追加</td></tr>
<tr><td>th</td><td>見出しのセルを追加</td></tr>
<tr><td>caption</td><td>タイトルや説明を追加</td></tr>
</table>
</body>
</html>
```

3. CSS

本来，HTML は文章の論理構造を記述する言語であったが，ブラウザメーカーによる拡張の結果，見栄えを記述するタグが大量に取り込まれてしまったため，HTML 文書が複雑化してしまったり，ブラウザによって表示が異なってしまったりするような問題が起きるようになった。

そこで，HTML 4.0（1997 年の規格）では，文書の論理構造を記述するという本来の目的に立ち返り，見栄えの記述は CSS（Cascading Style Sheets，スタイルシート）を使って行うように改められた。つまり，HTML で文書の構造を記述し，スタイルシートで視覚的構造を記述しようということである。

3.1 スタイルシート

見栄えと構造を分離するという目的で用いられるもので，HTML などのマークアップ文書や，オフィスソフト等で用いられている。

特に, HTML で用いられるスタイルシートは CSS（Cascading Style Sheets，カスケーディング・スタイル・シート，カスケード・スタイル・シート，段階スタイルシート）と呼ばれ，HTML の要素をどのように修飾（表示）するかを指示する仕様を指定する。文書の構造と体裁を分離させるという理念を実現する為に W3C により提唱された。

3.1.1 スタイルシートの記述ルール

適用するタグ（セレクタ）に続いて｛ ｝内に設定する見栄えの内容を記述（宣言）する。宣言部では，プロパティ(属性)と設定値を：（コロン）で区切って書く。

例）

h1 { color : blue }	h1 タグに文字色青の指定をする
body{	同じセレクタに複数の設定ができる
background-color:yellow;	セミコロンで区切る
color:blue;	背景色黄色と文字色青の指定
}	
h2,h3,h4{color:red}	違うセレクタに同じ設定をすることができる

例）rei3-1.html

3.1.2 スタイルシートを書く場所

以下のような記述方法がある。

1. タグに直接書く（インライン）
   スタイルを指定したい開始タグの中に，style 属性としてルールの宣言部だけを記述する。手軽にスタイルを指定できるが，まとめて修正することは難しい。
2. style タグに書く
   HTML の<head>～</head>の中に style タグを用いてルールを記述する。ページ内でまとめた変更がしやすい。
3. 外部ファイルに書く
   ルールを記述したファイルを別途作成し，ルールを適用したい HTML の<head>～</head>内でそのファイルを指定する。複数ページに同じスタイルを適用することができるので，統一感のあるサイトを作成することができる。

それぞれを併用することもできる（例：rei3-2.html）。スタイルは 1,2,3 の順で優先される。

3.2 CSS の主なプロパティ

主なものを挙げる（rei3-3.html）。これ以外にもさまざまな指定があるので，各自で調べて試してみて欲しい。

- 文字色 color: 色の指定
- 背景色 background-color: 色の指定

例）

h1{color:red}	色の名前で指定。
h2{color:#00ff00}	RGB(red,green,blue)の各成分を 00(最小)～ff(最大)の 16 進数で指定
h3{color}#00f}	RGB の各成分を 0(最小)～f(最大)の 16 進数で指定

例）

h1{background-color:rgb(255,255,128)}
RGB の各成分を 0(最小)～255(最大)の 10 進数で指定

h2,h3{color:rgb(80%,80%,60%)}
RGB の各成分をパーセントで指定

・ 文字の大きさ font-size

例)

```
<p style="font-size:small;">small</p>
<p style="font-size:large;">large</p>
        (xx-small,x-small,small,medium,large,x-large,xx-large)
<p style="font-size:2.5em;">2.5em</p>
        (基準フォントの 2.5 倍)
```

・ 文字のスタイル font-style

例)

```
<p style="font-style:italic">Italic</p>
        (イタリック指定)
```

・ 文字の太さ font-weight

例)

```
<p style="font-weight:bold">Bold</p>
        (太字指定)
<p style="font-weight:normal">Normal</p>
        (通常指定)
```

3.3 span タグと div タグ

どちらもスタイル規則をある範囲に対して適応したいときに使用するタグである。

3.3.1 span タグ

文字列の一部（インライン要素という）にスタイル規則を適用したい場合に使用するタグである。前後に改行が入らない(rei3-4.html)。

例)

```
<h1>情報<span style="color:red">システム</span>演習 D</h1>
```

3.3.2 div タグ

段落など，複数のタグ（ブロック要素という）に対してまとめてスタイル規則を適用したいに使用するタグである。前後に改行が入る(rei3-4.html)。

例）

```
<div style="background-color:yellow">
<p>段落 1</p>
<p>段落 2</p>
</div>
<p>段落 3</p>
```

3.4 class 属性

class 属性を使用すると，複数のタグに同じスタイル規則を設定することができる。

3.4.1 宣言方法

・ タグ名.class 属性の値 { 宣言 }　指定したタグにのみ指定
・ .class 属性の値 { 宣言 }　任意のタグに使用可能

3.4.2 使用例

midori と aka と名前を付けたスタイル規則は p タグでしか指定できないが，back_yellow と名前を付けたスタイル規則は任意のタグに指定することが出来る。

・ CSS

```
p.midori { color : green }
p.aka { color : red }
.back_yellow { background-color:yellow }
```

・ HTML

```
<h1>class の使用例 1</h1>
<p class="midori">段落 1</p>
<p class="aka">段落 2</p>
<p>段落 3</p>
<h2 class="back_yellow">class の使用例 2</h2>
<p class="back_yellow">段落 4</p>
```

例）rei3-5.html

3.5 スタイルシートのまとめ

3.5.1 スタイルシートの利点

- 見栄えを一度に指定できる
  - ひとつひとつ font タグ等で飾り付けを指定しなくてよい
  - class 属性や外部スタイルシートを使えば統一したウェブサイトを作成できる
- 自分の好きな見栄えに変更できる
  - ブラウザによっては，スタイルシートを指定することで自分好みの表示にすることができる
- 見栄えと意味を分離できる
  - ウェブページをデータベース化する場合や，目の不自由な人が見る場合など，飾りを外して文章だけを取り出すことができる

3.5.2 スタイルシートの注意点

- ブラウザによって動作が異なる場合がある（スタイルに対応していないブラウザがある）
- 前述のとおり，自分好みのスタイルを設定している人や，スタイルシートをオフにしている人もいるため，その場合の動作を考えてウェブページを作成する必要がある
- 外部スタイルシートを使っている場合は，HTML 文書以外にスタイルシートもアップロードしなければならない

3.6 サンプル

rei3-1.html スタイルシートの記述例

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>CSS1</title>
<style>
h1 { color : blue }
body{
 background-color:yellow;
 color:blue;
}
h2,h3,h4{color:red}
</style>
</head>
<body>
<h1>見出し 1</h1>
<p>あいうえお</p>
<h2>見出し 2</h2>
<p>かきくけこ</p>
<h3>見出し 3</h3>
<p>さしすせそ</p>
</body>
</html>
```

rei3-2.html スタイルシートの記述場所

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>CSS2</title>
<link rel="stylesheet" href="mycss.css">
<style>
 body {background-color:yellow}
 h2 {color:green}
</style>
</head>
<body>
<h1 style="color:red">タグに記述する方法</h1>
<p>あいうえお</p>
<h2>head で記述する方法</h2>
<p>かきくけこ</p>
<h3>外部ファイルに記述する方法</h3>
<p>さしすせそ</p>
</body>
</html>
```

mycss.css　　外部スタイルファイルの例

```
h3 {color:blue}
p {background-color:white}
```

rei3-3.html スタイルシート例 3

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>CSS3</title>
<style>
h1{color:red}
h2{color:#00ff00}
h3{color:#00f}
h1{background-color:rgb(255,255,128)}
h2,h3{background-color:rgb(80%,80%,60%)}
</style>
</head>
<body>
<h1>文字の大きさ</h1>
<p style="font-size:small;">small</p>
<p style="font-size:large;">large</p>
<p style="font-size:2.5em;">2.5em</p>
<h2>文字のスタイル</h2>
<p style="font-style:italic">Italic</p>
<h3>文字の太さ</h3>
<p style="font-weight:bold">Bold</p>
<p style="font-weight:normal">Normal</p>
</body>
</html>
```

rei3-4.html span タグと div タグの例

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>CSS4</title>
<style>
span{
 color:red;
 background-color:yellow;
}
div{
 color:cyan;
 background-color:green;
}
</style>
</head>
<body>
<h1>情報<span style="color:blue">システム</span>演習 D</h1>
<h2>span タグ</h2>
<p>文字列の<span>一部</span>にスタイル規則を<span>適用</span>
したいとき</p>
<h2>div タグ</h2>
<div>
<p>段落にまとめて</p>
<p>スタイルを適用</p>
</div>
<p>したいとき</p>
</body>
</html>
```

rei3-5.html class 属性の例

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>CSS5</title>
<style>
p.midori { color : green }
p.aka { color : red }
.back_yellow { background-color:yellow }
</style>
</head>
<body>
<h1>class の使用例 1</h1>
<p class="midori">段落 1</p>
<p class="aka">段落 2</p>
<p>段落 3</p>
<h2 class="back_yellow">class の使用例 2</h2>
<p class="back_yellow">段落 4</p>
</body>
</html>
```

4. JavaScript

この章では，JavaScript というプログラミング言語を用いて，動きがあるようなウェブページの作成方法を学習する。

4.1 JavaScript とは

ウェブページを記述するプログラミング言語の 1 つで，Sun Microsystems 社と Netscape Communications 社が開発したプログラミング言語である。これまで，静的な表現しかできなかったウェブページに，動きや対話性（インタラクティブ性という）を付加することを目的に開発された。現在は主要なウェブブラウザのほとんどに搭載されている。

4.1.1 JavaScript の変遷

HTML の経緯と同様に，ブラウザによって実装に違いがあるため，使えない機能があったり同じプログラムでも挙動が異なったりする問題があった。そのため，ヨーロッパの標準化団体 ECMA がソフトウェアメーカーに呼びかけて，通称「ECMAScript」と呼ばれる標準を発行し，現在では ECMAScript 準拠の処理系を JavaScript と呼ぶことになっている。また，独自の拡張を施した JavaScript には独自の名称を付けるようになっている。

例）

Microsoft の Internet Explore 等で用いられたものは「JScript」

Adobe の Flash で用いられたものは「ActionScript」

4.1.2 JavaScript についての注意

Java は，Sun Microsystems 社が開発（現在は Oracle 社が管理）したプログラミング言語である。様々なプラットフォームで動作するので，PC はもちろん，ブラウザ上でも動き，サーバから携帯電話・スマートフォンまで動作する。

Java と JavaScript いずれも C 言語を祖先として生まれた言語なので，文法はよく似ているが，まったく別のものなので使い分けに注意が必要である。

JavaScript は元々 LiveScript という名前だったが，Java が流行しはじめたことや，Netscape 社と Sun 社が技術提携したこともあり，JavaScript と名前を変更したので同じような名前となっている。

4.2 JavaScript の記述方法

JavaScript には以下のような記述方法がある。

(1) script タグを使用して，HTML の中に記述する
　通常のプログラムを作る場合(rei4-1.html)
(2) HTML の中でプログラムを設定したいタグの中に記述する
　マウスのクリック（イベントという）などに応じて動作させたい場合(後述)

4.3 プログラミングに関する基礎事項 1：変数・代入・演算

JavaScript のプログラミングで使用する用語や概念を説明する。

4.3.1 変数

プログラムの中で使うデータを記憶しておくための領域に名前を付けたものである。データの入れ物（箱）のようなものと考えることが出来る。変数を前もって準備（用意）することを「宣言」という。数学の「変数」と似たような概念である。

JavaScript の場合，変数名は，先頭が英文字で以降は英数字と＿（下線，アンダーバー，アンダーライン）とで構成される。大文字と小文字は区別される。

宣言せずに使用することができるが，宣言した方がわかりやすいプログラムとなる。宣言は var **変数名**; とする。

初期値は「undefined」という何も値が入っていないことを表す値が入る。

例）rei4-2.html

4.3.2 代入

変数などにデータを入れることである。使い方は a=1 のように，＝(イコール)記号を使うが，数学の「＝」とは意味が違うので注意が必要である。数学の場合，a=1 は変数 a と 1 が等しいという意味であるが，プログラムの場合，代入は右辺の値を左側の変数に入れることを表す（これを代入という）ので，a=1 は変数 a に 1 という値を入れることを意味する。

JavaScript の場合，代入することで変数にどんな型のデータも代入することができる。型とは変数に入れる事ができるものの種類のことで，数値型や文字列型などがある。つまり，JavaScript では文字でも数字でも変数に代入することができる。宣言と同時に値を代入することもできる。

例）rei4-2.html

4.3.3 演算

演算子という記号を使って計算をすることを演算という。数学で使う演算子

には，＋－×÷などがあるが，プログラミングの場合は違う記号を使うことがある。

JavaScript の場合（C 言語や Java と同様），演算子は四則演算として +-*/ が使用でき，それ以外に余りを求める演算の % と，変数の値を 1 増やす（インクリメント）++と，1 減らす（デクリメント）-- を使用できる。

さらに代入と演算を合成した複合演算子というものも存在する。

a=a+5 → a+=5, a=a-5 → a-=5,

a=a*5 → a*=5, a=a/5 → a/=5, a=a%5 → a%=5

例）rei4-2.html, rei4-3.html

4.4 オブジェクトとプロパティ・メソッド

4.4.1 オブジェクト

プログラミングで使用する，機能を備えた部品のことをオブジェクトという。ウェブプログラミングの場合，ウィンドウやドキュメント（文書）や各タグがこれに当たる。オブジェクトの特性や性質など，何らかのデータを設定できる（プロパティ）と，オブジェクトに対し与えられた指示に応じた処理を行うことができる（メソッド）を持つ。

このような仕組みを，Browser Object Model(BOM)，Document Object Model(DOM)という。

オブジェクトの例）

- window　　ウィンドウ
- document　　文書
- form　　フォーム

4.4.2 プロパティ

前述の通り，部品に設定するデータ（特性，性質）のことをプロパティという。以下のように「オブジェクト.プロパティ」として使用する。

プロパティの例）（rei4-4.html）

- document.bgColor="yellow";　　文書の背景色
- document.fgColor="blue";　　文書の文字色
- document.title="たいとる";　　文書のタイトル
- window.location.href = "http://www.nuis.ac.jp/";

location は window(ウィンドウ)の中にある位置に関するオブジェクトで，
href プロパティは現在表示している場所(URL)を表すプロパティであるので，
それを変更することにより，表示する URL を変更することが出来る。

4.4.3 メソッド

部品（オブジェクト）に与えられた処理をメソッドという。プログラムで使用される関数のようなものである。以下のように「オブジェクト.メソッド名（引数)」として使用する。

例）rei4-5.html

・ window.alert("てすと");　　ポップアップウィンドウを表示
・ window.document.write("こんにちは");　文書内に文字を出力
・ window.prompt("あなたの年齢は？", 20); 入力用ダイアログを表示
　初期値を指定できる。入力された値を返すようになっている（後述)。

・ window オブジェクトは省略可能なため，alert( )と書くことができる

4.5 プログラムの構造

プログラムの流れを言葉や記号で記述したものをアルゴリズムという。アルゴリズム（プログラム）の基本構造は次の 3 つである。

(1) 順次構造　順番に実行する
(2) 分岐構造　条件によって処理を分ける
(3) 反復構造　一定の回数を繰返す

順次構造はこれまでに出てきた例のように，順番に実行する処理である。続いて，分岐構造と反復構造について説明する。

4.5.1 分岐構造

ある条件に従って，処理を分岐させたいときに使用する。

・ <u>使用法 1</u>

```
if（条件式）{
    条件が正しい場合に
    実行する文
}
```

・ 使用法 2

```
if(条件式){
    正しいとき
        実行する文
}else{
    正しくないとき
        実行する文
}
```

・ 使用法 3

```
if(条件 1){
    条件 1 が成り立った時
}else if(条件 2){
    条件 2 が成り立った時
}else if(条件 3){
    条件 3 が成り立った時
}else{
    すべて成り立たない時
}
```

{}で囲まれた部分をブロックという。if 文や，次に説明する for 文では，ブロックを用いることで複数の処理を実行させることができる。

例）rei4-6.html ～ rei4-10.html

4.5.2 反復構造

一定の回数を繰返したい場合に使用する。繰返しをする前に初期値などを指定する部分と，ある条件が成立している限り繰返す条件を書く部分と，繰返しを行なうたびに実行する部分からなる。

```
for(初期設定; 条件; 繰返すたびの処理){
  繰返したい文
}
```

例）rei4-11.html, rei4-12.html

4.6 イベント処理

ユーザが何らかの操作を行ったときに発生する信号のことをイベントという。イベントの例としては，「マウスを動かした」「マウスのボタンをクリックした」「値を入力した」などがある。

このイベントに応じて，JavaScript のプログラムを動作させることができ，
<タグ名 イベント名＝"JavaScript の命令"＞のように記述する。

例）rei4-13.html

- **<h1 onmouseover="document.bgColor='red'"＞この上にマウスが来ると</h1>**
- **<h2 onclick="alert('クリックしたね')"＞ここをクリックすると</h2>**

※本演習ではこの記述方法を採用するが，伝統的な書き方であり，現在はイベントリスナという仕組みを使った記述方法が推奨されている。

4.6.1 主なイベント

主なイベントとしては以下のようなものがある。

対象	イベント名	意味
マウス	onclick	マウスボタンがクリックされた
	onmouseover	マウスポインタが上に乗った
	onmouseout	マウスポインタが領域から外れた
	onmousemove	マウスポインタが動いた
	onmousedown	マウスボタンが押された
	onmouseup	マウスボタンが（押されたあと）戻された
キー	onkeydown	キーが押された
	onkeypress	キーが押された（Shift,Ctrl,Alt には無反応）
	onkeyup	キーが離された
フォーム（後述）	onreset	リセットボタンが押された
	onsubmit	送信ボタンが押された
	onchange	項目が変化した
	onselect	選択された
その他	onload	ページの読み込みが終わった
	onunload	ページが破棄された(次ページに移動した)

4.7 関数

プログラムにおける関数とは，何らかのデータを受取り，決まった仕事をし，値を返す命令の集まりのことである。イベントが起こったときに関数を実行するようにすれば，複数の処理を実行できるようになる。

例）rei4-14.html （関数 msg）

4.8 style オブジェクトとオブジェクトの指定方法

style オブジェクトのプロパティを変更することにより，そのオブジェクト（部品）のスタイルを設定することができる。

また，id 属性を用いることにより，特定したいオブジェクトに id（識別番号のようなもの）を設定しておくことで，関数等からオブジェクトを指定するようにすることができる。

例）rei4-15.html

4.10 サンプル

rei4-1.html 一般的な記述方法

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript1</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
alert("JavaScript は");
alert("書いた順番どおりに");
alert("実行されます");
</script>
</body>
</html>
```

説明

- script タグ内に JavaScript のプログラムは記述する
- 実行したいことを順番に書いていき，最後に ; (半角のセミコロン)を書く
- 書いた順番に実行される
- alert("文字")は，ポップアップウィンドウというウィンドウが表示され，中に指定した"文字"が表示される

rei4-2.html 変数と代入と演算

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript2</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
var x=100;
var y=50;
alert(x+y);
alert(x-y);
alert(x*y);
alert(x/y);
</script>
</body>
</html>
```

説明

- 変数 x と変数 y を準備し，変数 x には 100 を，変数 y には 50 を入れて(代入)している
- alert(数式)と書くと，数式を計算した結果がポップアップウィンドウの中に表示される
- 順番に変数 x と変数 y を足し算，引き算，掛け算，割り算した結果が表示される

rei4-3.html 演算いろいろ

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript3</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
var z=3;
z++;
alert(z);
z--;
alert(z);
alert(z%2);
</script>
</body>
```

説明

- 変数 z を用意し，3 を代入している
- z++ により，変数 z の値が 1 増えるので，alert(z)で 4 が表示される
- z-- により，変数 z の値が 1 減るので，alert(z)で 3 が表示される
- z%2 は z を 2 で割ったときの余りを計算する演算なので，alert(z%2)では 3 を 2 で割ったときの余り 1 が表示される

rei4-4.html プロパティの使用例

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript4</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
document.bgColor="yellow";
document.fgColor="blue";
document.title="たいとる";
alert("移動します");
window.location.href = "http://www.nuis.ac.jp/";
</script>
</body>
</html>
```

説明

- プロパティで指定した部分が変化していることを確認する
  （文字色，背景色，タイトル）
- ポップアップウィンドウを閉じると，URL が設定されてページが移動することを確認する

rei4-5.html メソッドの使用例

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript5</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
window.alert("てすと");
window.document.write("こんにちは<br>");
var age;
age=prompt("あなたの年齢は？", 20);
document.write("あなたは",age,"歳なんですね。");
</script>
</body>
</html>
```

説明

- 前半の window.alert, window.document の window は省略できる
- document.write メソッドを使用すると画面に文字を表示することが出来る。HTML のタグを表示すれば，そのタグが解釈されて表示される。
- prompt メソッドはこの例のように返ってくる値を変数に入れることにより，値を入力することが出来る

rei4-6.html 分岐の使用法 1 の例(1)

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript6</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
var price=1000;
document.write("基本料金は"+price+"円です<br>");
document.write("20 歳未満は 1 割引となります<br>");
var age;
age=prompt("年齢を入力してください", 20);
if(age < 20){
  price*=0.9;
}
document.write("あなたの料金は"+price+"円です");
</script>
</body>
```

rei4-7.html 分岐の使用法 1 の例(2)

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript7</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
var score;
score=prompt("点数を入力してください", 100);
if(score>=60){
  alert("合格！");
}
</script>
</body>
</html>
```

rei4-8.html 分岐の使用法 2 の例(1)

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript8</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
var x,message;
x=prompt("整数を入力");
if(x%2==0){
   message="偶数";
}else{
   message="奇数";
}
alert(message);
</script>
</body>
</html>
```

rei4-9.html 分岐の使用法 2 の例(2)

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript9</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
var score;
score=prompt("点数を入力してください", 100);
if(score>=60){
  alert("合格！");
}else{
  alert("不合格！");
}
</script>
</body>
</html>
```

rei4-10.html 分岐の使用法 3 の例

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript10</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
var score;
score=prompt("点数を入力してください", 100);
if(score>=80){
  alert("評価 A");
}else if(score>=70){
  alert("評価 B");
}else if(score>=60){
  alert("評価 C");
}else{
  alert("不合格！");
}
</script>
</body>
</html>
```

rei4-11.html 反復の例 1

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript11</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
var i;
for(i=1; i<=3; i++){
   alert( i+"回目");
}
</script>
</body>
</html>
```

rei4-12.html 反復の例 2

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript12</title>
</head>
<body>
<h1>JavaScript</h1>
<script>
var i;
for(i=5; i>0; i--){
alert(i);
}
</script>
<p>スタート！</p>
</body>
</html>
```

rei4-13.html イベントの例 1（onclick, onmouseover）

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript13</title>
</head>
<body>
<h1 onmouseover="document.bgColor='red'">この上にマウスが来ると</h1>
<h2 onclick="alert('クリックしたね')">ここをクリックすると</h2>
</body>
</html>
```

rei4-14.html イベントの例２（関数）

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript14</title>
<script>
function msg()
{
 alert('クリックしたね');
 alert('複数の処理を');
 alert('実行するよ');
}
</script>
</head>
<body>
<h1 onclick="msg()">この見出しをクリックすると・・・</h1>
</body>
</html>
```

説明

- function が関数の定義を表し msg がここで定義している関数の名前を表す
- msg( )の( )は，ここでは関数は何も値を受け取らないことを表している
- 関数の中身は {} で囲まれた部分
- onclick で msg( )を実行するように設定している
- クリックされたら関数の本体に実行が移り，そこに定義されている命令を実行して戻ってくる

rei4-15.html イベントの例 3（関数，style，　id による指定）

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>JavaScript15</title>
<script>
function changeColor(c)
{
 var e = document.getElementById('ptag');
 e.style.backgroundColor=c;
}
</script>
</head>
<body>
<p id="ptag" onmouseover="changeColor('yellow')"
 onmouseout="changeColor('white')">
この上にマウスが来ると・・・外れると・・・
</p>
</body>
</html>
```

説明

- 関数 changeColor を定義している。
- この関数は引数として色の指定を受け取るようになっていて，受け取った色の指定を背景色として指定する動作を行なう。
- p タグの上にマウスが来ると yellow と言う文字を，マウスが外れると white という文字を送るようになっている。
- 背景色を指定したいオブジェクト（タグ）に対して id 属性を指定しておき，id 属性から getElementById メソッドを用いてオブジェクトを取得し，背景色を変えるようにしている
  - getElementById の最後の 2 文字は大文字の I(アイ)と小文字の d(ディー)であり，数字の 1 や小文字の l(エル)と間違えやすいので注意。

5. フォーム

ウェブページ上でデータを入力したり選択したりするための部品の集まりをフォームという。一つ一つの部品のことをコントロールという。下図のように，form タグの中にソースコードを記述することで作成することができる。name 属性でフォームを識別するための名前を付ける。

```
<form name="myform">
  ・・・
</form>
```

5.1 コントロール（部品）の作成方法

コントロールには，input タグを使うものと，専用のタグを使うものがある。

5.1.1 input タグを使うもの

テキストボックス

input タグで type 属性を text とすることで，文字を入力したり表示したりすることができるテキストボックスを作成することができる。name 属性でこの部品を特定する名前を付ける。これにより，後述する JavaScript や CGI から部品を特定して値を取得することができるようになる(rei5-1.html)。size 属性で大きさを指定することができる。

例：`<input type="text" name="username">`

ラジオボタン

input タグで type 属性を radio とすることでラジオボタンを作成することができる。同一項目から 1 つだけ選択することができる。value 属性で選択されたときの値を設定することができる（後にプログラムで使用する）。checked 属性を付けると最初に選択された状態となる(rei5-1.html)。

例：`<input type="radio" name="ans2" value="value2" checked>`

チェックボックス

input タグで type 属性を checkbox とすることでチェックボックスを作成することができる。複数項目にチェックを付けることができる。value 属性で選択

されたときの値を設定することができる（後にプログラムで使用する）。checked 属性を付けると最初に選択された状態となる(rei5-1.html)。

例：<input type="checkbox" name="ans1" value="value1" checked>

ボタン

input タグで type 属性を button とすることでボタンを作成することができる。このボタンを押すことで，JavaScript のイベント（onclick）でプログラムを起動したり，CGI プログラムにデータを送ったりすることができる。value 属性でボタンに表示される文字を設定することができる(rei5-1.html)。

例：<input type="button" name="send" value="送信">

5.1.2 input タグを使わないもの

ドロップダウンメニューとリストボックスは全く同じもので，表示形式（表示する桁数）が違うだけとなっている。

ドロップダウンメニュー

select タグ内に option タグで項目を指定することで，ドロップダウンメニューを作成することができる。選択することで，value 属性で指定した値がプログラムに送信されるようになる(rei5-2.html)。selected を指定することで初期値を設定することができる。

```
例：<select name="dropdownmenu">
      <option value="value1">選択肢 1</option>
      <option value="value2" selected>選択肢 2</option>
    </select>
```

リストボックス

前述の select タグで size を指定して選択肢を多く表示することで，リストボックスを作成することができる(rei5-2.html)。

```
例：<select name="listbox" size="2">
      <option value="value1">選択肢 1</option>
      <option value="value2" >選択肢 2</option>
    </select>
```

テキストエリア

textarea タグにより，複数のテキストが入力可能な入力欄を作成することができる。cols 属性で 1 行あたりの文字数を，rows 属性で行数を指定する(rei5-2.html)。

5.2. JavaScript からのフォームの利用

前述の通り，フォームに作成したボタンをクリックしたときに，イベント onclick で JavaScript の関数を呼び出すように設定することで，プログラムを動作させることができる。

それぞれの部品に入力した設定した値は，フォームをオブジェクトとして，その中に各部品オブジェクトがあり，各部品オブジェクトの value プロパティとして取り出すことができる(rei5-3.html～rei5-9.html)。

例：`myform.username.value`（前述したテキストボックスの値の場合）

ただし，ラジオボタンとチェックボックスは少し違った形で処理することになる(rei5-5.html, rei5-6.html)。

5.3 サンプル

rei5-1.html フォームの作成 1

（テキストボックス，チェックボックス，ラジオボタン，送信ボタン）

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>form1</title>
</head>
<body>
<h1>Form 1</h1>
<form name="myform">
<p>お名前：<input type="text" name="username"></p>
<p>使用機器：<br>
<input type="checkbox" name="ans1" value="iphone">iPhone
<input type="checkbox" name="ans1" value="android">Android
<input type="checkbox" name="ans1" value="keitai" checked>携帯電話
<input type="checkbox" name="ans1" value="other">その他
</p>
<p>次に買いたい機器：<br>
<input type="radio" name="ans2" value="iphone">iPhone
<input type="radio" name="ans2" value="android">Android
<input type="radio" name="ans2" value="keitai">携帯電話
<input type="radio" name="ans2" value="other" checked>その他
</p>
<p><input type="button" name="send" value="送信"></p>
</form>
</body>
</html>
```

説明

- このサンプルはフォームのコントロール（部品）を作成するだけで，入力やチェックは出来るが，ボタンを押しても動作しない

rei5-2.html フォームの作成２
（ドロップダウンメニュー，リストボックス，テキストエリア）

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>form2</title>
</head>
<body>
<h1>Form 2</h1>
<form name="myform">
<p>年齢
<select name="age">
<option value="10">10代以下</option>
<option value="20" selected>20代</option>
<option value="30">30代</option>
<option value="40">40代以上</option>
</select>
</p>
<p>出身
<select name="hometown" size="3">
<option value="shinai" selected>新潟市内</option>
<option value="kennai">新潟県内</option>
<option value="kengai">新潟県外</option>
</select>
</p>
<p>コメント<br>
<textarea cols="40" rows="5">
ご自由にどうぞ。
</textarea>
</p>
<p><input type="button" name="send" value="送信
"></p>
</form>
</body></html>
```

説明

・ このサンプルもフォームのコントロール（部品）を作成するだけで，入力やチェックは出来るが，ボタンを押しても動作しない

rei5-3.html フォームの使用例：円の面積の計算
（テキストボックス，送信ボタン）

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>form3</title>
<script>
function keisan()
{
  var r=myform.hankei.value;
  var s=r*r*3.14;
  myform.menseki.value=s;
}
</script>
</head>
<body>
<h1>円の面積の計算</h1>
<form name="myform">
<p>半径を入力<input type="text" name="hankei" value="1"></p>
<p><input type="button" value="計算" onclick="keisan()"></p>
<p>面積<input type="text" name="menseki"></p>
</form>
</body>
</html>
```

説明

- ボタンを押して動作させるには，以前学習した onclick イベントをボタンに適用し，関数を呼び出すようにすれば良い
- 関数の中からフォームの部品の値を取得するには，
  フォーム名.コントロール名.value
  とすれば良い（テキストボックスの場合）
- 反対にフォーム名.コントロール名.value に値を入れると，その値がコントロールに表示される（テキストボックスの場合）

rei5-4.html フォームの使用例：偶数と奇数の判定
（テキストボックス，送信ボタン，if 文）

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>form4</title>
<script>
function hantei()
{
  var n, msg;
  n=myform.num.value;
  if(n%2==0){
    msg="偶数";
  }else{
    msg="奇数";
  }
  myform.kekka.value=msg;
}
</script>
</head>
<body>
<h1>偶数と奇数の判定</h1>
<form name="myform">
<p>整数を入力<input type="text" name="num"></p>
<p><input type="button" value="判定" onclick="hantei()"></p>
<p><input type="text" name="kekka" value="----">です</p>
</form>
</body>
</html>
```

説明

・ 以前学習した奇数と偶数の判定をテキストボックスに入力した値で行うようにしたサンプル

rei5-5.html フォームの使用例：占い？（ラジオボタン，for 文，if 文）

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>form5</title>
<script>
function uranai()
{
  var i, msg="未選択";
  for(var i=0; i<myform.blood.length; i++){
    if(myform.blood[i].checked){
      msg="あなたは"+myform.blood[i].value+"ですか？";
    }
  }
  myform.kekka.value=msg;
}
</script>
</head>
<body>
<h1>ラジオボタン</h1>
<form name="myform">
<p>血液型は？<br>
<input type="radio" name="blood" value="神経質">A
<input type="radio" name="blood" value="自己中">B
<input type="radio" name="blood" value="大ざっぱ">O
<input type="radio" name="blood" value="変人">AB
</p>
<p><input type="button" value="占い"
 onclick="uranai()"></p>
<p><input type="text" name="kekka" size="30"
 value="----"></p>
</form>
```

説明

ラジオボタンはテキストボックスのように単純に値を取り出すことが出来ない。この例にあるように，全てのボタンにチェックが入っているかどうか調べなければならない。

length オブジェクトはラジオボタンの数

checked オブジェクトはチェックが入っているかどうか（入っていたら true）

blood[i]は配列という構造を表し，ラジオボタンが作成された順に 0,1,2 という番号がつく

以上より，ラジオボタンの数だけ for 文で繰返し，書くボタンにチェックが入っているかを調べ，入っていたらそのボタンに設定されている value を取り出している

rei5-6.html　カロリーチェック（チェックボックス，for 文，if 文）

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>form6</title>
<script>
function check()
{
  var i, count=0, calorie=0;
  for(var i=0; i<myform.menu.length; i++){
    if(myform.menu[i].checked){
      count++;
      calorie+=Number(myform.menu[i].value);
    }
  }
  myform.kekka1.value=count;
  myform.kekka2.value=calorie;
}
</script></head>
<body>
<h1>チェックボックス</h1>
<form name="myform">
<p>何を食べたいですか？（複数可）<br>
<input type="checkbox" name="menu" value="235">ごはん
<input type="checkbox" name="menu" value="480">ラーメン
<input type="checkbox" name="menu" value="410">鶏のから揚げ
<input type="checkbox" name="menu" value="85">マグロの刺身
<input type="checkbox" name="menu" value="123">サラダ
<input type="checkbox" name="menu" value="50">みそ汁</p>
<p><input type="button" value="計算" onclick="check()"></p>
<p><input type="text" name="kekka1" size="4" value="--">品
<input type="text" name="kekka2" size="30" value="----">カロリー</p>
</form>
</body></html>
```

説明

チェックボックスもラジオボタンと同様に，全てのボックスにチェックが入っているかどうか調べなければならない。

この例では，全てのチェックボックスに対しチェックが入っているかどうか調べ，チェックが入っていたらその数をカウントするとともに，そこに設定してある value(カロリー)を加えていき，最終的に何個チェックが入り，カロリーの合計がいくらかを表示している

rei5-7.html フォームの使用例：rei5-5.html(占い)のドロップダウンメニュー版

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>form7</title>
<script>
function uranai()
{
  var i, msg;
  msg="あなたは"+myform.blood.value+"ですか？";
  myform.kekka.value=msg;
}
</script>
</head>
<body>
<h1>ドロップダウンメニュー</h1>
<form name="myform">
<p>血液型は？
<select name="blood">
<option value="神経質" selected>A</option>
<option value="自己中">B</option>
<option value="大ざっぱ">O</option>
<option value="変人">AB</option>
</select></p>
<p><input type="button" value="占い" onclick="uranai()"></p>
<p><input type="text" name="kekka" size="30" value="----"></p>
</form>
</body>
</html>
```

説明

- select タグを使うドロップダウンメニュー・リストボックスは，ラジオボタン・チェックボックスのような煩わしい処理をせずに，テキストボックスと同様に選択したものを value として取り出すことが出来る

rei5-8.html フォームの使用例：rei5-5.html(占い)のリストボックス版

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>form8</title>
<script>
function uranai()
{
  var i, msg;
  msg="あなたは"+myform.blood.value+"ですか？";
  myform.kekka.value=msg;
}
</script>
</head>
<body>
<h1>リストボックス</h1>
<form name="myform">
<p>血液型は？<br>
<select name="blood" size="4">
<option value="神経質" selected>A</option>
<option value="自己中">B</option>
<option value="大ざっぱ">O</option>
<option value="変人">AB</option>
</select></p>
<p><input type="button" value="占い" onclick="uranai()"></p>
<p><input type="text" name="kekka" size="30" value="----"></p>
</form>
</body>
</html>
```

説明

- rei5-7.html とほとんど同じ（select タグでサイズを指定しているかどうかの違いのみ）

rei5-9.html フォームの使用例：テキストエリアの使用例

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8">
<title>form9</title>
<script>
function shori()
{
  alert(myform.comment.value);
}
</script>
</head>
<body>
<h1>テキストエリア</h1>
<p>textarea に入力された値の取り出し方の例。<br>
alert で表示。</p>
<form name="myform">
<textarea name="comment" cols="40" rows="5"">
感想をどうぞ。
</textarea><br>
<input type="button" value="送信" onclick="shori()">
</form>
</body>
</html>
```

説明

・ value プロパティを用いて入力された文字列を取り出すことができる