プログラミングB
（2022年度後期）

井田 正明

2022年10月28日

プログラミングの概要と開発環境

プログラミングの概要

- プログラミングとは

- ウェブでのプログラミングの概要
- （サーバとクライアント）

- ホームページの仕組み（**コンテンツ，スタイル（デザイン），プログラミング**）
  - **HTML**
  - **CSS**
  - **JavaScript**

- プログラミング開発環境
  - テキストエディタ，
  - サーバへのファイルupload（「一般公開」はしない予定）

さまざまなWebページの作成方法

- テキストエディタを使用して作成
  - 基本，HTML・CSS・JavaScript のファイルを直接編集
  - 最も自由度がある
  - 技術的な知識が必要
  - 慣れるまで作業がむつかしい
- ホームページ作成ソフトを利用
  - ホームページビルダー等
- CMSツール（ブログツール）を利用
  - WordPress等
- ホームページ作成サービスを利用
  - WiX等
- SNS, YouTube
  - Twitter, Instagram, ….

プログラミングの開発環境

- ブラウザ（**閲覧用**ソフト）
  - Google Chrome

- エディタ（**編集用**ソフト）
  - テキストエディタ（Mery，Sakura, メモ帳）
    - ファイルの拡張子を「**.html**」「.css」などとする（「.txt」とはならないように注意）
    - ファイルの文字コードは，**utf-8** とする
    - (Mery:　https://forest.watch.impress.co.jp/library/software/mery/　)

- デベロッパーツール
  - Google Chrome の「その他のツール」→「デベロッパーツール」
  - または，「**F12**キー」で起動・表示

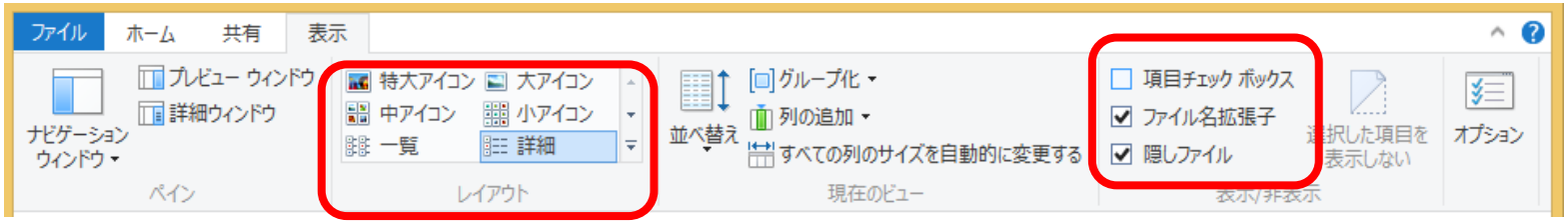
- サーバへのファイルアップロード用のソフト（検討中）
  - WinSCP（大学内での使用）
    - Host名：ikuta1.isc.senshu-u.ac.jp
    - 「ユーザ名」と「パスワード」を入力

作業フォルダの作成と「課題」の提出方法

- 作業フォルダの作成
  - 各自のパソコン内に新しく作業用のフォルダを作成
  - 「**webfiles_2022_1007**」（**webfiles_日付**）
  - フォルダ内に必要なファイルをコピー
  - フォルダに移動しその中で表示や編集作業を行う

- 作業フォルダの提出
  - **フォルダをzipファイルへ変換**（複数のファイルをひとつにまとめる）
  - 右クリックして，「送る」，「ZIPファイルに圧縮」→「.zip」
  - zipファイル（webfiles_2022_1007.zip）を「in Campus」へ提出（アップロード）

- フォルダ内のファイルの「拡張子」（.html, .png, .jpg など）は表示させておく



情報基礎II-第6章「ホームページの作成」

- 教科書（**2021年**度版）第6章（2022年からは無い）:
  - http://www.isc.senshu-u.ac.jp/jtext/text2021/text2_dl.html

- (1) ホームページのしくみを理解する.

- (2)「タグ」を使って，「HTML」を記述できるようになる.

- (3) インターネットのしくみやコンピュータでの記述方法を理解する.

- 動画による説明
  - http://www.isc.senshu-u.ac.jp/jtext/textmovies2/kiso2_movies.html
  - SmartArtによる図の作成，Excelのグラフを利用，透明pngで保存，Google Spredsheetによるグラフ作成

エディタ(プログラム作成・編集用のソフト)

- プログラム作成・編集用のエディタが必要
  - Meryエディタ(Windows 64bit版)
    - 大学内のパソコンにはインストールされている
    - 画面がプログラムの内容で色分けされる
    - 自宅等で利用の場合
    - https://forest.watch.impress.co.jp/library/software/mery/
    - (その他:さくらエディタ　https://sakura-editor.github.io/)
    - Macについて，同等の「テキストエディタ」を使用してください.
  - ファイルを保存する際には，(文字化けしないように)文字コードを「utf-8」にする

「index.html」ファイル
（フォルダ内の基準ファイル）

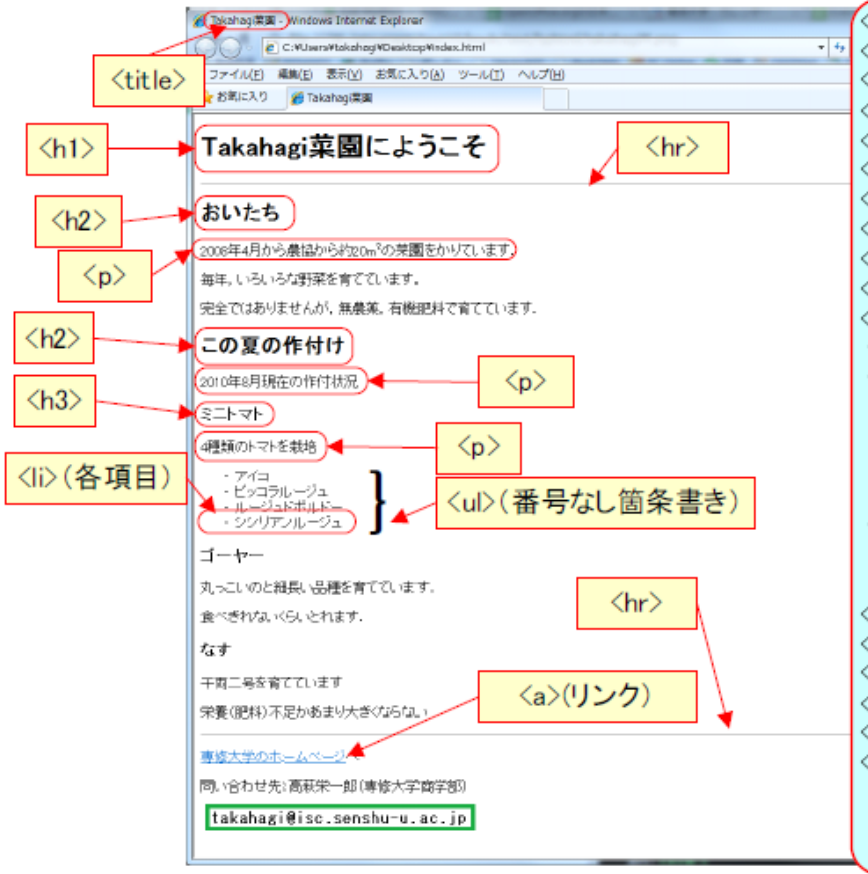
- index.html ファイルの表示と編集

- ブラウザで開く（左）
  - ダブルクリックする（IEが起動されることがあるので注意）
  - 「Chrome」からファイルを開いて読み込むこと
  - **ファイルをブラウザの画面にドロップする方がわかりやすい**

- エディタで開く（右）
  - （注：zipは解凍しておくこと）
  - Meryエディタを起動させて，ファイルを開いて読み込み
  - **ファイルをエディタの画面にドロップする方がわかりやすい**

開発環境，編集作業の状況

ブラウザ画面（左）　　エディタ画面（右）




```
<body>
<h1>Takahagi菜園にようこそ</h1>
<hr>
<h2>おいたち</h2>
<p>2008年4月から農協から約20平方メートルの菜園をかりています．
</p>
<p>毎年，いろいろな野菜を育てています．</p>
<p>完全ではありませんが，無農薬，有機肥料で育てています．</p>
<h2>この夏の作付け</h2>
<p>2010年8月現在の作付状況</p>
<h3>ミニトマト</h3>
 <p>4種類のトマトを栽培</p>
 <ul>
  <li>アイコ</li>
  <li>ピッコラルージュ</li>
  <li>ルージュドボルドー</li>
  <li>シシリアンルージュ</li>
 </ul>
```

途中略

```
<hr>
<p><a href="http://www.senshu-u.ac.jp/" >専修大学のホームページ
</a>へ</p>
<p>問い合わせ先：高萩栄一郎（専修大学商学部）</p>
<p> takahagi@isc.senshu-u.ac.jp </p>
</body>
```

プログラムの編集と実行

- 以下の手順で行う
  - エディタでプログラムを編集する
  - エディタでプログラムファイルを上書き保存する
  - 「実行」:ブラウザで再読み込み（リロード，更新）」（上方のボタン．またはF5）

HTMLについて

HTML

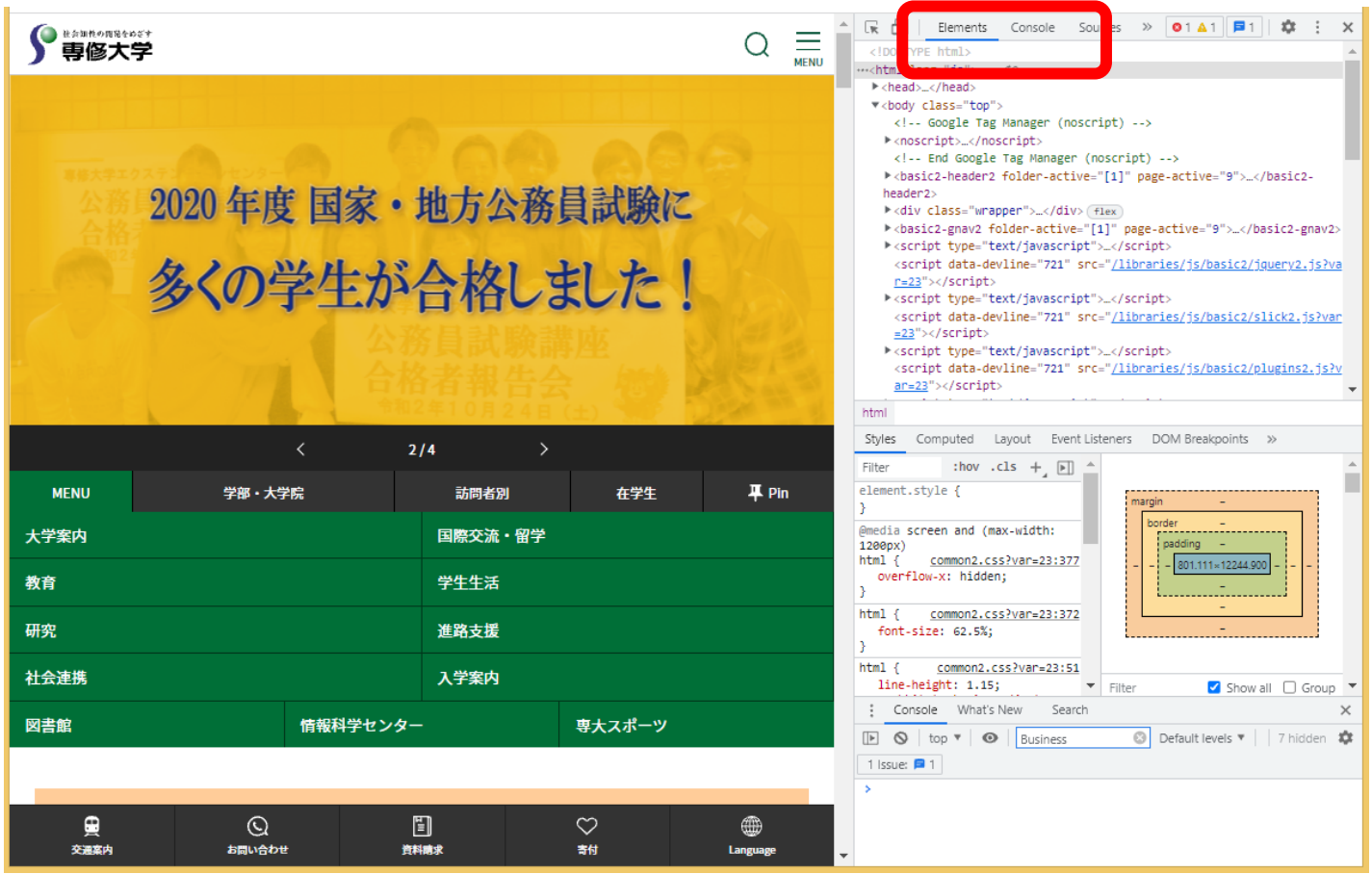
- Web のページは，HTML（Hyper Text Markup Language）という言語の文法にしたがって書かれている
- 通常，拡張子(ファイル名の最後のピリオドより右の文字) は，「html」のファイル
- タグ（ <h1> , <p> など）の集合によって構成されている

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Takahagi 菜園</title>
<!--
<link rel="stylesheet" href="sample.css">
-->
</head>
<body>
<h1>Takahagi 菜園にようこそ</h1>
<hr>
```

「デベロッパーツール」でソースを確認する
（複雑なファイルの例）

- Google Chrome で，「F12キー」を押す
  （または「その他のツールの中」）

「Elements」，「Console」など
重要な機能がある



「HTML（HTML5）」の基本構造

```
<!DOCTYPE html>    DOCTYPE宣言
<html lang="ja">    html開始タグ
  <head>    head開始タグ
    <meta charset="UTF-8">
     <title>    title開始タグ
      タイトル
    </title>    title終了タグ
  </head>    head終了タグ
  <body>    body開始タグ
        ここに，内容となる様々なタグを記述
        （<h1>タグ，<p>タグ，<a>タグ，<ul>タグ，<table>タグ，・・・）

  </body>    body終了タグ
</html>    html終了タグ
```

代表的なタグ

通常は「開始タグ」と「終了タグ」の対になっている（ <title> ****</title> ）

- <title> タグ
  - ウインドウのタイトルバーに表示する文字列
    <title> ＊ ＊ ＊ ＊菜園</title>

- <h1> タグ
  - 見出し（heading）
    <h1>＊ ＊ ＊ ＊にようこそ</h1>
  - 他に，<h2>, <h3>, <h4>, ・・・

- タグ
  - 段落（paragraph）
    2008年4月から約20平方メートルの菜園をかりています

代表的なタグ（続き）

- タグ， タグ
  - 箇条書き（先頭が「・」，または「数字付」）

```
<ul>
 <li>アイコ
 <li>ピッコラルージュ
</ul>
```

- <table> タグ （タグの階層構造）
  - 表（テーブル） <tr> １行を意味する <th>見出しタグ <td>データタグ

```
<table>
    <caption>収穫個数</caption>
    <tr> <th>年度</th> <th>トマト</th> <th>ゴーヤー</th> <th>なす</th> </tr>
    <tr> <th>2012</th> <td>128</td> <td>23</td> <td>38</td> </tr>
    <tr> <th>2013</th> <td>158</td> <td>43</td> <td>52</td> </tr>
</table>
```

代表的なタグ（続き）

- 特殊なタグ（終了タグ無し）
  - <hr> タグ　横罫線（horizontal rule）
  -
 タグ　改行（break）
- コメントタグ
  - <!--　　コメント　　-->
    - コメントを記入
    - プログラムの一部を「未使用部分に指定」する際にも利用する

代表的なタグ（続き）

- <a> タグ （href 属性で指定）
  - リンクを生成

    **** 専修大学のホームページ ****

- <img> タグ （src 属性で指定）
  - 画像表示（image）

    <img src=“photofile.png" style="width:500px">

  - 画像のダウンロード（フリー画像，個人利用）
    - マウス右クリック →「名前を付けて画像を保存」

代表的なタグ（続き）

- タグ
  - ブロックを作成（これ自体は領域（場所）を確保しているだけ）

```
<div style="text-align:center;">
    ...
</div>
```

  - スタイルやidの利用など様々な用途がある

- タグ
  - ** と似たタグとして タグ**は，「文中の一部分」に対して利用

（実習）

- インターネットから「フリー画像」を取得して，<img> タグで，画像を表示してみましょう
  - ダウンロードの仕方
  - ファイルを格納するフォルダについて

- （注意）プログラムを記述する際の注意
  - 「全角」と「半角」の違いに気をつけましょう
    - ダブルクォート「“ ”」と「” ”」は異なります
    - シングルクォート「‘ ’」と「’ ’」は異なります
  - 「>」や「)」が欠落してないか気をつけましょう

- 問題解決：ネットでの検索を利用

代表的なタグ（続き）

- <form>タグ （入力インタフェース）

```
<form>
 <input type="button"  value="クリック"  onclick = “ alert( ‘こんにちは！’); "  >
 <button type=“button”  onclick = “alert(‘こんばんは！’);”  >押す</button>
</form>
```

パワーポイント利用して「図のファイル（png）」を作成

- パワーポイントを起動
- スライドのレイアウトを「白紙」
- 図の作成
  - 挿入 → 図 → SmartArt
    - **図を作成**（動画:Web:SmartArt による図の作成）
    - http://www.isc.senshu-u.ac.jp/jtext/textmovies2/SmartArt.html
  - **Excel で作成したグラフ**もパワーポイントのスライドに貼り付け
    - http://www.isc.senshu-u.ac.jp/jtext/textmovies2/Excel2pp.html
  - その他にも「図形ツール」を使って図やアイコンの作成を各自で行う

- 最後に作成した図を**「png 形式」で保存**
  - http://www.isc.senshu-u.ac.jp/jtext/textmovies2/pp2png.html
    - パワーポイントの「図として保存」では自動で背景色は透明
    - パワーポイントで，作成した図やグラフをすべて選択
    - 右クリックして，「図として保存」を選択（png 形式）
      - **ファイル名には日本語文字を使用しない**
      - htmlファイルと同じフォルダに保存（移動）

パワーポイントのツールの利用

- 図形ツールでの図形を組み合わせて，新しいアイコンなどを作る
  - 組み合わせた図形を「グループ化」する
  - 図をpng形式で保存

パワーポイントを利用して「スライドショー動画のファイル（mp4）」を作成

- mp4ファイルの作成
  - ファイル → エクスポート → ビデオの作成
  - ファイルの種類は「標準」（MPEG4-ビデオ(*.mp4)）
  - 「記憶されたタイミングとナレーションを使用しない」
    - 各スライドの所要時間（秒）：例えば，5秒
  - ファイルの保存：ファイル名には日本語を使用しない（videofile.mp4など）

- HTMLファイルにテキストエディタで下記のコードを追加する

```
<video width="500" height="450" controls>
    <source src="videofile.mp4" type="video/mp4">
</video>
```

HTML インターネット資源の利用など

- 教科書 第6章
  - 「8.1 リンクの作成」
  - 「8.2 自分のページに埋め込む」　Google Map
  - 「8.3 Google chart tools」
    - https://developers.google.com/chart/
  - 「8.4 Goole Spreadsheet」

  - 「9 PDF ファイル」
  - 「10 複数のWeb のページの作成」

    - **（ここまでの内容のプログラム（htmlファイル）を作成する．必要なファイルをフォルダに入れ，zip化，提出）**

  - 「11 HTML 作成ソフトウエアを使用する」
  - 「12 Wiki」

  - 「13 公開作業」

インターネットとプログラミング

- **Webサービス，Web API**
  - RSS
    - Yahoo News：　https://news.yahoo.co.jp/rss
    - Allabout：　https://rss.allabout.co.jp/
  - ビジネスとプログラミング
    - 財務情報とXBRL
    - EDINET：　https://disclosure.edinet-fsa.go.jp/
    - 分析ツール
  - オープンデータ
    - e-Stat：　https://www.e-stat.go.jp/
    - RESAS：　https://resas.go.jp/
    - 公的データやビジネス関連のさまざまなオープンデータ
    - COVID19

CSSについて

CSS（Cascading Style Sheets）

- HTMLドキュメントの要素（h1, h2, p など）のプロパティの設定

- CSSの表記法（セレクタ，プロパティ，値）
  - （「どこ」の「何」を「どのように」するか）
  - **セレクタ {プロパティ : 値;}**

    （例） **h1** { color : red ; } （見出し(h1)の色を赤にする）

    p { font-size:20px; color:red; } （サイズを20, 色を赤にする）

  - セレクタの種類
    - 要素，子孫，子（>），隣接（+），属性（[=]），**id（#）**，**クラス（.）**，疑似クラス（div:hover, a:visited）

  - Cascading: プロパティの値の指定が段階的に引き継がれて文書に適用される

CSSを記述する場所（3種類）

- (1)「**インライン形式**」 HTMLのタグの内部（「属性」）に直接記述する

```
<p style="color:blue;" >文字</p>
<h1 style="color:red; font-size:24pt;" >CSSの説明</h1>
```

- (2)「 **内部スタイルシート形式**）」 ファイル内の <head> </head> の内部に記述する

```
<head>
 <style type="text/css">
        p {color:blue;}
 </style>
</head>
```

- (3)「**外部スタイルシート形式**」 外部CSSファイル（例：style1.css）にCSSを記述し，<link>でファイルを読み込む

```
<head>
 <link rel="stylesheet" href="style1.css" type="text/css">
</head>
```

「id属性（#）」および「class属性（. ）」

「id, #」や「class, . 」による場所の指定方法

```
<head>
  <style>
    #id1001 { background : red; }    （これは，id=“id1001”の場所についての指示）
    .c2001 { color : blue; }         （これは，class=“c2001”の場所についての指示）
  </style>
</head>

<body>
  <h1 id=“id1001”>ここの背景は赤色</h1>      （id は1つしか使えない ）
  <h2 class=“c2001”>ここの文字は青色</h2>    （class は複数使える ）
</body>
```

なお，「属性」による指定は，「CSS」以外にも，「JavaScript」のプログラムでも頻繁に使用する

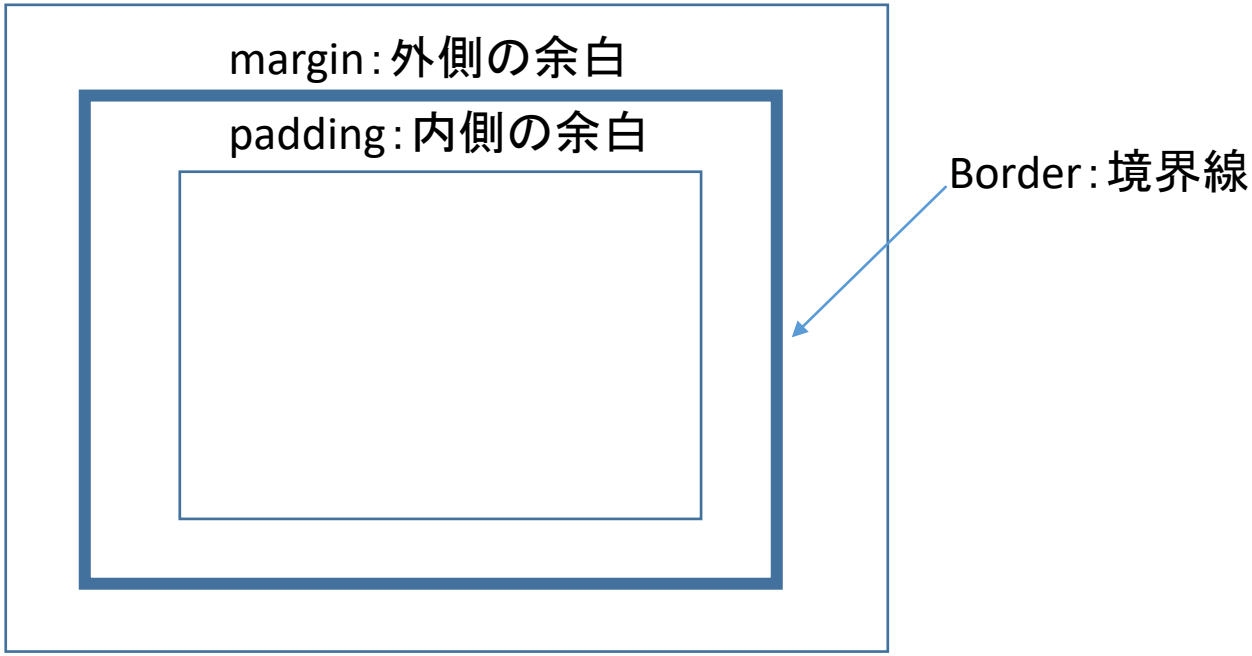
（例）

基本プロパティの例

- フォント関連
  - color: red
  - background-color: #3366ff
  - font-size: 120%
  - font-weight: 400
  - font-family: serif
- テキスト関連
  - text-align: center
  - text-decoration: underline
  - line-height: 1.5em
  - letter-spacing: 0.2em
  - text-indent: 1em
- リスト関連
  - list-style-type: circle
  - list-style-position: inside
  - list-style-image: url('./img123.gif')
- ボックス（上 右 下 左）
  - margin: 5px 10px 3px 15px
  - padding: 5px
  - border: dashed 2px
- イメージ（画像）関連
  - background-image:url('./img123.gif');
  - background-position: left
  - background-repeat: no-repeat

border, margin, padding

```
p {
  border: solid 3px #0000ff;    （境界線を実線，線の幅3px，青色）
  margin: 20px;     （外側の余白20px）
  padding: 20px;      （内側の余白20px）
}
```



カラーについて（カラーコードによる色の表現）

- 色の英語による指定
  - aqua（水色）
  - gray（灰色）
  - navy（紺）
  - silver（銀色）
  - black（黒）
  - green（緑）
  - olive（オリーブ色）
  - teal（青緑）
  - blue（青）
  - lime（黄緑）
  - purple（紫）
  - white（白）
  - fuchsia（桃色）
  - maroon（茶）
  - red（赤）
  - yellow（黄色）

- 色の詳細な指定：16進数表記（**2桁ずつ3色**）

- 16進数とは：「0,1,2,3,4,5,6,7,8,9,**A,B,C,D,F**」
- 16進数で2桁：　00（弱）から　FF（強）までの 256 段階
- 最初に「#」をつける

- 基本の3色
  - 赤（R）　#FF0000
  - 緑(G)　#00FF00
  - 青(B)　#0000FF

- 基本3色の組み合わせ
  - **Black　#000000**
  - **White　#FFFFFF**
  - Gray　#808080
  - Yellow　#FFFF00
  - Pink　#FFC0CB
  - Aqua　#00FFFF

- 詳細は，ネットで「カラーコード」と検索して調べる

（実習）

- （ここまでの知識で）実際のページを参考にページを作成してみましょう
  - https://www.senshu-u.ac.jp/about/campus/
  - 表，写真，図，箇条書き，・・・・・・

- ヒント

```
<table border="1" width="60%" style="border-collapse: collapse; border:1px solid #333; margin: auto;">
 <tr>
  <td width="420px"><img src="abm00003150.png" style="width:400px; padding:10px"></td>
  <th style="text-align:left;">
   <h4>ラーニング・コモンズ（５号館）</h4>
   <p style=“font-weight:nomal; font-size:10pt”>らせん状に配置された、開放空間の「アクティブラウンジ」・・・・・・<p>
  </th>
 </tr>
</table>
```

CSS追加

- さまざまな機能・困難な調整
  - ブロックにして全体を構成
    -
    - container の中に（header, sidebar, content, footer など）
    - それぞれに，margin, border, padding, や background-color
  - hover
    - a { text-decoration:none; transition:3s;}
    - a:hover { text-decoration: underline;background-color:lightgreen;transition:3s;}
  - 上下間隔（微小な空白）
  - 表（テーブル）の様々な設定

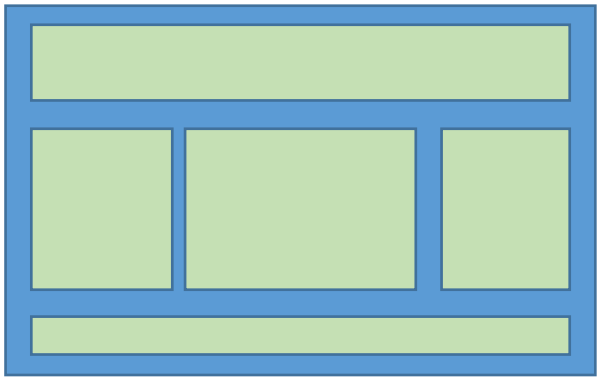
- インターネットでの調査
  - 検索の仕方

CSSの利用：floatによる「マルチコラム」の作成

- floatとclearによる位置指定

```
<html>

<head>
<style>
    body {text-align:center;}

    #container {width:700px; border: thin solid #F00;
    background-color:#FF0; padding:5px;}

    #header {width:700px; height:100px; border:thin solid #F00;
    background-color:#FFF; text-align:left; margin-bottom:5px;}

    #sidebarLeft {width:120px; height:300px; float:left;}
    #content {width:420px; height:300px; float:left;}
    #sidebarRight {width:120px; height:300px; float:right;}
    #footer {width:700px;height:50px; clear:both;}

</style>
</head>
```

```
<body>
<div id="container">
  <div id="header">
      header
  </div>
  <div id="sidebarLeft">
      sidebarLeft
  </div>
  <div id="content">
      content
  </div>
  <div id="sidebarRight">
      sidebarRight
  </div>
  <div id="footer">
      footer
  </div>
</div>
</body>
</html>
```



（Webページの作成練習）：このファイルにこれまでの HTML, CSS の学習内容を追記

multicolの追加説明

```
#container {
        margin-right: auto;
        margin-left : auto;
        text-align:center;
        width:1200px;
        border: thin solid #DDD0D0;
        background-color: #DDFFEE;
        padding:20px;
}
#header {
        border-top-left-radius: 30px;
        border-top-right-radius: 30px;
        border-bottom-left-radius: 30px;
        border-bottom-right-radius: 30px;
        width:1200px;
        height:90px;
        border: thin solid #DDCCDD;
        background-color: aqua;
        margin-bottom:20px;
        box-shadow: 0 10px 25px 0 rgba(0, 0, 0, .5);
}
#sidebarLeft {
        width:300px;
        border: double green;
        float:left;
        height:1840px;
        margin-right:15px;
}
```

```
#content {
        width:700px;
        float:left;
        background-color: white;
        height:1800px;
        padding : 20px ;
        margin-bottom:10px;
        text-align:left;
}
#sidebarRight {
        width:120px;
        border: double red;
        float:right;
        height:1840px;
}
#footer {
        width:1200px;
        height:50px;
        text-align:right;
        border: solid #CCCCCC;
        clear:both;
}
```

文字などの表示位置の指定方法

CSSによる位置の固定　position:absolute;

top　上からの位置,　　left　左からの位置

（例1）

```
<p style="position:absolute; top:3%; left:45%; font-size:200%;">専修大学</p>
```

（例2）

```
<p style="position:absolute; top:160px; left:600px; font-size:150%;">入学案内</p>
```

「Bootstrap」を利用することによる マルチコラムの作成

Bootstrapを利用

```
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.1/dist/css/bootstrap.min.css" >
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.1/dist/js/bootstrap.min.js" ></script>
```

スマートフォンで1列（12グリッド），タブレットで2列（6グリッド），スマートフォンで3列（4グリッド）

```
<div class="row">
    <div class="col-12 col-md-6 col-lg-4">
        <h1>項目1</h1>  <p>専修大学は、1880年（明治13年）、米国留学から帰国した４人の若者により創立されました。</p>
    </div>
    <div class="col-12 col-md-6 col-lg-4">
        <h1>項目2</h1>  <p>専修大学は、1880年（明治13年）、米国留学から帰国した４人の若者により創立されました。</p>
    </div>
    <div class="col-12 col-md-6 col-lg-4">
        <h1>項目3</h1>  <p>専修大学は、1880年（明治13年）、米国留学から帰国した４人の若者により創立されました。</p>
    </div>
</div>
```

プログラミング言語
JavaScript について

JavaScriptの基本型(J01)

- JavaScriptの記述場所
  - HTMLに埋め込んで実行
  - **<script>　タグを使う**
  - **<head>内に，<body>内に**
  - 文の終わりは「;」

- 出力（内容の確認）
  - **デベロッパーツール(F12) の console を開く**
  - console.log( )
  - 文字列は「”　”」で囲むこと

- <meta charset=“utf-8”>
  を指定する
- コメント方法：「//」，「/*　*/」
  - HTMLでは　「**<!–　コメント　-->**」

- **ファイル名は「.html」にする**
- **ファイルを保存で文字コードは「utf-8」にする**
- **全角文字に注意（文字列以外は「半角英数字」）**

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>JavaScript練習</title>
  <script>
    alert("こんにちは");
    console.log("こんばんは");
    //  コメント
    /* コメント */
  </script>
</head>

<body>
  <h1>JavaScriptの基本</h1>
  <!-- コメント(HTML) -->
</body>
</html>
```

変数 (J02)

- 変数宣言:「var」
- 変数の決め方:
  - 一文字目: 英字, $, _
  - 2文字目以降: ・・・, 数字
  - 大文字と小文字は区別される
  - 変数は小文字からはじめる
  - 予約語
  - 命名規則
    - **英単語(内容が分かりやすい)**
    - 変数, 関数:camelCase 記法
  - (Pascal)
- スコープ

  「ローカル変数」

  「**グローバル変数**」

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>JavaScript練習</title>
  <script>
    var msg1;
    msg1 = “こんにちは”;
    console.log(msg1);
</script>
</head>

<body>
  <h1>JavaScriptの基本</h1>
</body>
</html>
```

変数：初期値設定

- var inputNum = 123;
  - 「宣言」と同時に「初期値設定」

- 注意：同じ変数に異なる
  データ型代入

  var inputNum = “こんにちは";

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>JavaScript練習</title>
</head>
<body>
  <h1>JavaScript基本</h1>
  <script>
    var inputNum = 123;
    console.log(inputNum);
  </script>
</body>
</html>
```

関数 (J03)

- function 宣言
  - return 命令
- **引数**を渡す
  - **戻り値**を受け取る
- コード分離
  - 関数が増加・複雑化
  - 「src属性」で読み込む
  - 「double.js」（外部ファイル）

```
function   doubleValue(x) {
  var y = x * 2;
  return  y;
 }
```

```
<!DOCTYPE html>
<html>
<head>
 <meta charset="utf-8">
 <title>JavaScriptの基本</title>
 <script>
 function doubleValue(x) {
  var y = x * 2;        // ローカル変数
  return y;
 }
 </script>
</head>

<body>
 <script>
var inputNum = 100;
var outputNum = doubleValue(inputNum);
   cosole.log(outputNum);
 </script>
</body>
</html>
```

入出力(J04)

- **ファイル読み込み**

  <script src="plus.js"></script>

- DOM: Document Object Model

- **データの取得（DOM）**

document.getElementById(“id100").**value**

- **データの表示（DOM）**

document.getElementById(“id200").**innerHTML**

```
<!DOCTYPE html>
<html>
<head>
 <meta charset="utf-8">
 <title>JavaScript基本</title>
<script src=“double.js"></script>
 <script>
 function showValue() {
var inputValue = document.getElementById("id100").value;
var resultValue= doubleValue(inputValue);
document.getElementById(“id200").innerHTML = resultValue;}
 </script>
</head>

<body>
<form>
 数値入力：<input type="text" id="id100">
 <button type="button" onclick="showValue()">
  クリックしてください</button>
</form>

<div id=“id200">ここに表示されます</div>
</body>
</html>
```

DOM: Document Object Model

- データの取得（DOM）

  document.getElementById(“id100").**value**

  id=“id100”を見つけて，そこの値を取得する

- データの表示（DOM）

  document.getElementById(“id200”).**innerHTML = 値**

  id=“id200”を見つけて，そこに値を表示する

「値」が HTMLである場合，その内容（次の例では図）が表示される

document.getElementById(“id200”).innerHTML = ‘<img src=“zu3.png” style=“width:200px;”>’;

（注意）ダブルクォートが使われている場合，その外側はシングルクォートで囲む

さらにその外側は，ダブルクォートで囲む

HTML：入出力の処理

```
<form>
  <input type="text" id="inputValue">
  <button type=“button” onclick = "showValue()“ >クリックして</button>
</form>

<div id="outputValue">ここに表示されます</div>
```

「整数」として精度が保証されるのは

-9,007,199,254,740,992 〜9,007,199,254,740,992

配列（複数の変数）・繰返し（J10）

- 配列（代入）

var testArray = [“専”, “修”, “大”, “学”];

- i番目の要素

  testArray[i]

- 配列の長さ（例では，4）

  testArray.length

- 繰り返し　for文
  - （例では，0,1,2,3 となる）

```
for (var i = 0 ; i < testArray.length ; i++ ){
        （ここが繰り返される）
}
```

```
<!DOCTYPE html>
<html><head>
 <meta charset="utf-8">
 <title>配列と繰り返し</title>
 <script>
 function showArray() {
  var testArray = [“専", "修", "大", "学“];
  var resultValue = "";
  for (var i = 0 ; i < testArray.length ; i++ ){
   resultValue += i + "番目の要素： " + testArray[i] + "<br>";
  }
  document.getElementById("outputValue").innerHTML =
   resultValue;
 }
 </script>
</head>

<body>
<form>
 <button type="button" onclick="showArray()">クリック</button>
</form>

<div id="outputValue"></div>
</body>
</html>
```

文字列(J11)

- 文字列の長さ
  testString.length

- 文字の場所指定
- 3番**(注意:0番から始まる)**
  - teststr.charAt(4)
- 7から10未満
  - teststr.substring(7, 10)
- 7から3つ
  - teststr.substr(7, 3)

- 文字の結合
  - teststr.concat('商学部')

```
<!DOCTYPE html>
<html>
<head></head>
<body>
<script>        // コンストラクタとリテラル
   var str = new String('神奈川県川崎市多摩区専修大学') ;
   console.log(str);
   var teststr = '神奈川県川崎市多摩区専修大学‘ ;
   console.log(teststr);

   // 文字の長さ
   console.log( teststr.length );

   // 文字の場所指定
   console.log( teststr.charAt(4) );
   console.log( teststr.substring(7, 10) );
   console.log( teststr.substr(7, 3) );

   // 文字の結合
   console.log( teststr.concat('商学部') );
</script>
</body>
</html>
```

文字列操作（J12）

- 文字列の長さ
  testString.length
  （例では，9）
- 繰り返し（for）
- +=　追加の意味
- 文字列の「**i 番目**から n 文字」取得
  testString.substr(i, n)

```
<!DOCTYPE html>
<html>
<head>
 <meta charset=“utf-8">
 <title>文字列の操作</title>
 <script>
 function showString() {
  var testString = "プログラミング応用";
  var resultValue = "";
  for (var i = 0 ; i < testString.length ; i++) {
    resultValue +=   i + "番目の文字： "
          + testString.substr(i, 1) +"<br>";
  }
   document.getElementById("outputValue").innerHTML
          = resultValue;
 }
 </script>
</head>
<body>
<form>
 <button type="button" onclick="showString()">クリック
</button>
</form>
<div id="outputValue"></div>
</body>
</html>
```

数学 Math Object（J13）

```
<script>
  // 絶対値
        console.log( Math.abs(-10) );

  // 最大値・最小値
        console.log( Math.max(3, 4) );
        console.log( Math.min(-5, 0) );

   // 乱数
        console.log( Math.random() );

// 切り上げ，切り捨て，四捨五入
        console.log( Math.ceil(1.6) );
        console.log( Math.floor(1.6) );
        console.log( Math.round(1.6) );
```

```
  // 指数　2の4乗
        console.log( Math.pow(2, 4) );


  // 平方根　 2の平方根
        console.log( Math.sqrt(2) );


  // 円周率
        console.log( Math.PI );


  // 三角関数
        console.log( Math.cos(3) );
        console.log( Math.sin(3) );
</script>
```

簡単なゲーム(「左右当てゲーム」:left-right)

```
<!DOCTYPE html>
<html>
<head>
 <meta charset="utf-8">
 <title>条件文とグローバル変数</title>
 <script>
 ningenWin = 0,  computerWin = 0;     //グローバル変数設定

 function resetGame() {
   ningenWin = 0,   computerWin = 0;
 document.getElementById("result").innerHTML =
       “右ですか？左ですか？” ;
 document.getElementById("status").innerHTML = “ “ ;
 }

 function hantei(x) {
  var sayu, hanteiString;
  var randomValue = Math.floor(2 * Math.random());
  switch (randomValue) {
   case 0: sayu = "左";  break;
   case 1: sayu = "右";  break;
  }
  var hanteiValue = Math.abs(x - randomValue);
  switch (hanteiValue) {
   case 0: hanteiString = “予想と一致しましたので,
                人間の勝ち";
           ningenWin++ ;  break;
   case 1: hanteiString = “予想と一致しませんので,
                コンピュータの勝ち";
           computerWin++ ;  break;
  }
 document.getElementById("result").innerHTML =
   “コンピュータは" + sayu + "でした。" + hanteiString + "です";
 document.getElementById("status").innerHTML =
   “人間:” + ningenWin + “勝コンピュータ:" + computerWin + "勝";

  if (computerWin == 5) {
   alert(“3勝” + ningenWin + “敗でコンピュータが勝ちました");
  }

  if (ningenWin == 5) {
   alert(“3勝” + computerWin + “敗で人間の勝ちです．");
  }
 }
 </script>
</head>

<body>
<h2>簡単なゲーム(左右当て)</h2>
<div>勝負</div>
<form>
 <button type="button" onclick="hantei(0)">左</button>
 <button type="button" onclick="hantei(1)">右</button>
 <button type="button" onclick="resetGame()">リセット</button>
</form>

<div id="result">右ですか？左ですか？</div>
<div id="status"></div>
</body>
</html>
```

条件文（もしこうなら・・）

・（1） if 文，if ～ else 文

```
if ( 条件式 ) {
        ******;
} else {
        ******;
}
```

「==」,「>」,「<」,「>=」,「<=」,
「true」,「false」,「||」,「&&」

```
if ( iWin == 3 ) {
  alert( “3勝" + youWin +
   "敗でわたしが勝ちました");
  resetGame();
}
```

（2） switch 文

```
switch ( x ) {
    case 0: watashi = "左";
            break;
    case 1: watashi = "右";
            break;
}
```

テキストボックスに入力された数字は，明確に数値に変換してから入力する

```
 y = Number(x);
Switch( y ) {
   case 0: ****
   case 1:  ****
}
```

グローバル変数，乱数，アラート

●**グローバル変数**（とローカル変数）
ningenWin = 0, computerWin = 0 ;

●**乱数の利用**
Math.random()　「0から1」の値をとる．「 0.1, 0.6, 0.9, 0.4. 0.2・・・など」
Math.floor(2 * Math.random()**)**
2倍する．つまり，「0.2, 1.2, 1.8, 0.8, 0.4・・・など」
切り捨てする．つまり，「0,　1,　1,　0,　0,・・・など」**（0か1の乱数生成）**
（「ceil」は切り上げ、「floor」は切捨て、「round」は四捨五入）

Math.abs(x - rndValue**)**　差の「絶対値」をとることで「一致」を調べる

●**アラートと変数**
alert(“3勝” **+** ningenWin **+** “敗でコンピュータが勝ちました” **)**

●変数の値の増加
computerWin++（値を1だけ増やす）

プログラムの実行順序

（1）基本：プログラムの上から下へ順に実行されていく

（2）**「ボタン」をクリックして**ある関数の実行を開始させたい場合

```
<form>
    <button  type=“button”  onclick=“関数()” >  クリックしてください  </button>
</form>
```

（3）HTMLの読み込み後に**「自動的に特定場所から実行開始」**させたい場合
（jQueryを使う場合）

```
$(function () {
      // ここに，プログラムの読み込み完了時に実行するプログラムを書く.
      // ここから実行が始まる
 });
```

(重要)プログラム作成時の注意事項

- 例題のプログラムを十分に理解する
- 開発ツール(F12)のconsoleで，変数の内容(console.log(変数))を確認する
- 例題のプログラムを少しだけ追加・修正し，変化(画面の変化やcolsole.logの変化)を確認する.
  - (注意)例題のプログラムを急に大きく変化させないこと．少しずつ変化させて確認することが重要
  - うまくいかないときには，うまく進んでいるところまで戻ること
- 「エラーメッセージ」の意味を考えてみる
  - Refference Error (not defined): 変数名や関数名が，存在しない.
  - Syntax Error: かっこ「 { } 」の対応が取れていない(一つ少なかったり多かったり).
  - Failed to load resource: 読み込むファイルが存在しない.
  - エラーと思われる変数の内容を **console.log() を追加**して出力し，確認する.
- 問題解決方法：インターネットでの検索を利用
  - プログラム作成において，「ネットで調査」する能力も重要である

（実習）プログラムの改良（「左右当て」ゲーム）

- デベロッパーツールを使う

- 改良（少しずつ修正を加えていく）
  - 「5勝」で終了を，「3勝」で終了に変更
  - 表示されている文章を少し変える
  - ゲームに適した画像（クリップアート）を加える
  - 最後のアラートの表示を通常の表示に変える
    - getElementByID を利用
  - デベロッパーツールのコンソールにつぎの変数の内容を表示させる
    - console.log("ningenWin = ", ningenWin);
    - console.log("computerWin = ", computerWin);
  - 最後の場面で画像を表示させる（人間の場合とコンピュータの場合）
  - リセットの内容を改良する
  - 最後の場面に「リセットボタン」を移動させる
  - 「左」「右」に「上」を加えて「3択」のゲームにする

  - （その他の工夫）

  - 解答例は，配布ファイル

オブジェクト

- オブジェクト：
  - 「**プロパティ**」と「**メソッド**」から構成
  - インスタンス化
    - 「new演算子」：オリジナルのコピーを生成し操作する

      new 変数名 = オブジェクト名
  - インスタンス変数（オブジェクト変数）　「．」ピリオド

    変数名．プロパティ名

    変数名．メソッド名
      - （注意：例外）
  - **組み込みオブジェクト**：String, Array, Function, Math, Date
    - var str1 = new String(‘専修大学’) ;
    - var str2 = ‘専修大学’ ;

データの入力，フォーム（form）

```
<form>
  <input type="button"  value="クリック"
    onclick =  “ alert( ‘こんにちは！’); " >
  <button type=“button”
    onclick = “alert(‘こんばんは！’); ”> 押す</button>
</form>
```

さまざまなフォーム（サンプルファイルを参照）

- テキストボックス(F01, F02)　<form> <input>
- プルダウンメニュー（F03）　<form> <select> <option>
- ラジオボタン（F04）　<form> <label> <input>　（ .checked　をさがす）
- 複数チェックボックス（F05）　<form> <label> <input>（ .checked　をさがす）
- 複数ドロップダウン（F06）<form> <select> <option>　.selected

テキスト入力（基本）

```
<!DOCTYPE html>
<head>
<meta charset="UTF-8">
<title>テキスト入力（基本）</title>
<script>
//テキストボックスの文字取得
function show_textbox1(){
  var str1 = document.getElementById("id101").value;
  //var str1 = document.form101.name101.value;
  alert("こんにちは、" + str1 + " さん。");
  document.getElementById("id201").innerHTML = 'こんばんは、' + str1 + " さん。";
  console.log(str1);
}
</script>
</head>
<body>
<form name="form101">
お名前は？<input type="text" id="id101" name="name101" value="神田花子"><br>
<input type="button" value="表示" onclick="show_textbox1()"><br>
</form>
<div id="id201"></div>
</body>
</html>
```

プルダウンメニュー

```
function show101(){
   var item101 = document.getElementById("id101").value;
   //var item101 = document.form1.kenmei.value;

   document.getElementById("id201").innerHTML = '出身県は、' + item101 ;
   console.log(item101);
}
```

```
<form name="form1">
            出身県は？<br>
            <select name="kenmei" id="id101">
                        <option value="神奈川県">神奈川県</option>
                        <option value="東京都">東京都</option>
                        <option value="埼玉県">埼玉県</option>
                        <option value="千葉県">千葉県</option>
            </select>
   <input type=“button” value=“表示” onclick=“show101()”>    <br>
</form>
```

```
<div id="id201"></div>
```

ラジオボタン

```
function getRadio(elem101){
        var result2 = '';
        for(var i = 0; i < elem101.length; i++){
                if (elem101[i].checked){
                        result2 = elem101[i].value;     break;
                }
        }
        return result2;
}
```

```
function show101(){
  //var item2 = getRadio(document.form101.addr);
  var item2 = getRadio(document.getElementById("id101"));
    ＊＊＊＊
}
```

```
<form name="form101" id="id101">
        住所は？<label><input type="radio" name="addr" value="神奈川県" />神奈川県</label>
            <label><input type="radio" name="addr" value="東京都" />東京都</label>
            <label><input type="radio" name="addr" value="埼玉県" />埼玉県</label>
            <label><input type=“radio” name=“addr” value=“千葉県” />千葉県</label>    <br>
  <input type="button" value="表示" onclick="show101()">
</form>
```

フォームのまとめ

2種類の場所の指定方法（id属性，name属性）

id による方法

```
var str1 = document.getElementById("id101").value;
```

name による方法

```
var str1 = document.form101.name101.value;
```

フォームでチェックされている場所を探す方法

```
For ( i = 0;  i < elem101.length;  i++ ){
        if ( elem101[i].checked ) {
                result2 = elem101[i].value;
                break;
        }
}
```

（実習）プログラムの改良２（「左右当て」ゲーム）

- 「F01（テキスト入力）」を利用して，名前の入力と表示をしましょう
- 「F03（プルダウン）」を利用して，「左，右，上」の入力するプルダウンメニューを追加しましょう.
- （「F04」を利用して，「左，右，上」の入力するラジオボタンを追加しましょう．）

（実習）「じゃんけんゲーム」

```
ningenWin = 0;
computerWin = 0;
hikiwake = 0;

var rndValue = Math.floor(3 * Math.random());
//0,1,2 の3つの数をランダムに生成

document.getElementById("result").innerHTML =
"わたし（コンピュータ）は" + computer_value + "でした。" + hanteiString + "です";

document.getElementById("status").innerHTML =
"あなた（人間）:" + ningenWin + "勝.　わたし（コンピュータ）:" + computerWin + "勝.　引き分け:" + drow ;
```

```
var hanteiValue = x - rndValue;
    //人間の値からコンピュータの値を引く
switch (hanteiValue) {
    case 0: hanteiString = "引き分け";
    drow++ ; break;
    case 1: hanteiString = "あなた（人間）の【負け】";
    computerWin++ ; break;
    case 2: hanteiString = "あなた（人間）の【勝ち】";
    ningenWin++ ; break;
    case -1: hanteiString = "あなた（人間）の【勝ち】";
    computerWin++ ; break;
    case -2: hanteiString = "あなた（人間）の【負け】";
    ningenWin++ ; break;
}
```

（実習）じゃんけんゲームの改良１

入力ボタンの部分を改良して「プルダウンメニュー」，または「ラジオボタン」に変更する

- 条件分岐の「switch」にいれる x は数値にする（ Number( ) を使うと，文字を数値に変換できます）

```
function show101(){
    var item101 = document.getElementById("id101").value;
    console.log(item101);
    console.log(typeof(item101));  // 型をしらべる．string

    var item102 = Number(item101);  // 入力値を文字から数値に変換している
    console.log(item102);
    console.log(typeof(item102));    //型を調べる．number

    syoubu( item102 );  // あらためて syoubu関数を呼び出す．
}
```

（続き）

```
function getRadio(elem101){   // ラジオボタンのどれが選択されたかを調べる
        var result2 = '';
        for(var i = 0; i < elem101.length; i++){
                if (elem101[i].checked){
                        result2 = elem101[i].value;
                        break;
                }
        }
        return result2;
}
```

```
function show101(){
   var item101 = getRadio( document.form101.addr );
   console.log(item101);
   console.log(typeof(item101));  // 型をしらべる. string

   var item102 = Number(item101);  // 入力値を文字から数値に変換している
   console.log(item102);
   console.log(typeof(item102));   //型を調べる. number

   syoubu( item102 );  // あらためて syoubu関数を呼び出す.
}
```

（実習）じゃんけんゲームの改良２

- 「少しずつ」プログラムを追加・修正していく

- 表示の改良
  - 人間が1勝すると，「一勝目（初勝利）ですね！」と表示する
  - 終盤に，「あと1勝で優勝ですね！」と表示する
  - （左右ゲームと同様に）表示する文章を充実させる
  - ゲームに適した画像（クリップアート）を加える
  - リセットの内容を改良する

- 連勝についての表示の改良
  - 連勝すると「調子が良いですね」と表示する
  - 連勝用の変数の追加
  - 連勝でなくなると，その表示を消す

- 「注意」
  - console.logで変数の動きを確認しながら，繰り返し実行する

メッセージの追加と削除について

（例）

```
if (ningenWin == 1   &&   (hanteiValue==2 || hanteiValue==-1)  ) {
   document.getElementById(“****").innerHTML = "1勝目ですね“ ;
}
else {
   document.getElementById(“****”).innerHTML = “ ” ;    //「空白」表示. 何も表示させない
}
```

じゃんけんゲームの改良の例(4)

入力を「ラジオボタン」に変更．グーで勝つと2点，その他は1点，引き分け0点，5点先取で優勝と修正

ポイント

- 「switch (hanteiValue) {　**********　}」の部分の改良

JavaScriptのライブラリの活用（ jQuery ）

- jQueryの特徴
  - コード量を短縮できる
  - ブラウザの違いを意識せずに記述
  - DOM操作を簡単に記述
  - Ajax処理を簡単に記述

- jQuery 読み込みの記述方法
  - **CDNを使用する場合**

    <script src = "https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js" ></script>
  - 直接ファイルを読み込む場合

    <script src = "ファイルパス/jquery-3.6.0.min.js“ ></script>

プログラムの実行順序

- 基本：プログラムの上から下へ順に実行

- HTMLの読み込みが完了してから実行させたい場合（2種類）

```
window.onload = function () {
    // ここに，プログラムの読み込み完了時に実行するプログラムを書く
};
```

jQueryを使う場合：

```
$(function () {
    // ここに，プログラムの読み込み完了時に実行するプログラムを書く
});
```

ネットワークデータの活用（後半に向けて）

- 方法
  - ネットワークからのデータの取得（ ajax ）
  - JSONデータから必要な一部のデータの抽出（ json.****.**** ）
  - HTML, CSSでの表示

- 対象となるネットワーク経由のデータ
  - RSS（ニュース，ベストセラー，献立，旅行，etc.）
  - XBRL（財務）
  - オープンデータ
    - E-stat（日本の統計）
    - Resas（日本の統計）

RSSの基本プログラム（後半に向けて）

```
<!DOCTYPE html>
<html>
<head><meta charset='UTF-8'>
<title>RSS表示</title>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
<script>
function feedGet(url) {
   $.ajax({
      type: 'GET',
      url: url,
      dataType: 'jsonp',    //または, JSON
      jsonpCallback: 'callback',
      success: function(json){
         console.log(json);              // 取得したデータの確認
         container = document.getElementById("feed");
         var doc = "";  // ここに必要なデータを詰め込んでいく

         for(num in json.channel.item){
            doc = doc + '<h3>' + json.channel.item[num].title + '</h3>';    // データが追加された
         }

         container.innerHTML = doc;
      })
}

$(function(){    // ページの読み込み後に実行
   feedGet( 'http://cgi.isc.senshu-u.ac.jp/~thz2368/webapi/jsonencode.php?feed=https://news.yahoo.co.jp/rss/topics/top-picks.xml' );
});
</script>
</head>
<body>
<h1>表示</h1>
<div id="feed"></div>      <!-- 結果の表示場所 -->
</body>
</html>
```