

実習内容

プログラミング実習「Web アプリ編」では、以下の言語を学びます。

- ・HTML：マークアップ言語
- ・CSS：スタイルシート言語
- ・JavaScript：プログラミング言語（スクリプト言語）

これら言語を利用して「文字・数字を扱う」「画像や音声を扱う」「個人制作」の3段階で進めていきます。画像やアニメの表現に比べると地味に感じるかもしれませんが、プログラミングを学ぶ上で重要な内容になります。

開発環境

- ・ブラウザ: Google Chrome (Dev Tools)
- ・エディタ: VS Code (拡張機能なし)
- ・画像: Adobe Photoshop、Adobe Illustrator 等

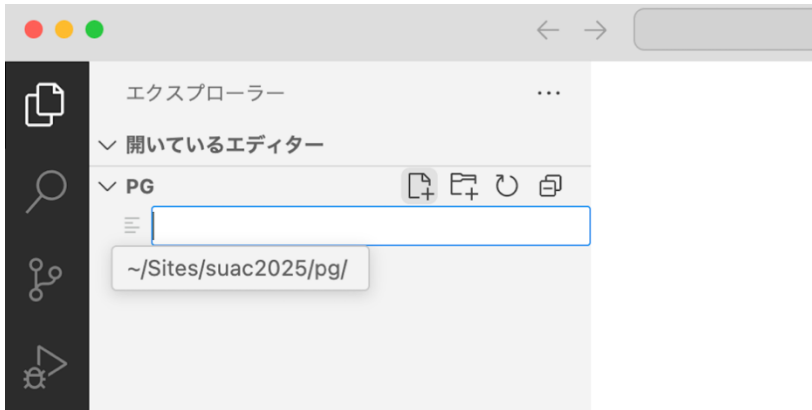
注意点

- 1 プログラムを改変したら、そのつど名前を書き変えてバックアップしてください。
(一つのファイルに上書きを繰り返してしまうと、エラー発生の際に、ちゃんと動いていた状態に戻りにくくなります)
- 2 プログラムは基本的に、半角の英数字と記号で記述します。
(コメント記法は全角文字を記述できます。)
- 3 JavaScript では大文字／小文字は異なる文字として扱われます。
- 4 読みやすさのために、文字の間に半角の空白を入れたり、行間に空白行を入れて構いません。
(ただし、空白を入れてはいけない箇所がまれにあるので注意。別途説明します)
- 5 コメントの記号を書くと、その右側の記述はプログラムでは実行されない。
プログラムを分かり易くする上で重要です。コメントには全角文字も使えます
- 6 同じ内容を実行するプログラムの書き方は何通りもあります。
(教員がサンプルとして示した書き方だけが正解というわけではありません)

Step0 / 準備

演習用のフォルダを用意します。

- ・ デスクトップに演習用のフォルダ「pg」を作成してください。
- ・ VS Code を起動して、ファイル > フォルダーを開く > 「pg」フォルダを選択してください。
- ・ エクスプローラー > ファイル作成 > step0.html と入力してファイルを作成してください。



step0.html に以下の文字を記述して、ファイルをブラウザ（Chrome）にドラッグ&ドロップしてください。

Hello world !

ブラウザに「Hello world !」と表示されれば準備は完了です。

URL は、**file:///C:/Users/（あなたのユーザー名）/Desktop/pg/step0.html** になります。
通常、Web アプリは Web サーバ経由で取得するため、https:// もしくは http:// ですが、
ローカル PC（自分のパソコン）のファイルを参照する場合は、file:/// でも表示ができます。
ただし、JavaScript を扱う場合は、いくつか制約が発生します。現時点では割愛します。

Step1 / HTML の基本

HTML は HyperText Markup Language（ハイパーテキスト・マークアップ・ランゲージ）の略です。Web アプリを作成するための基本的なマークアップ言語であり、文章の構造をコンピューターに伝えるための「タグ」と呼ばれる指示記号で構成されています。

step1.html を作成して、以下のコードを記述し、ブラウザで開いてみましょう。

HTML の基本的な記述：

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <title>Demo</title>
</head>
<body>
  <!-- ここに HTML を記述します -->
  <h1>Hello world !</h1>
</body>
</html>
```

解説：

- 基本的に <html></html>、<head></head> のように開始タグと終了タグがあります。一部例外があり、例えば、後述の
タグ等には開始終了タグはありません。
- <body></body> の間に記述したコードが画面に表示されます。
- <!-- ○○○ --> は、コメントといいます。○○○の箇所は自由に文字を書くことができ、画面には表示されず、プログラムにも影響しません。

HTML タグには色々な種類があります。いくつか例を以下に挙げます。

基本構造：

タグ	説明	例
<!DOCTYPE html>	HTML5 であることを宣言	<!DOCTYPE html>
<html>	HTML 文書の最外枠	<html lang="ja">...</html>
<head>	ページの設定や情報を入れる	<meta>, <title>, <link> などを含む
<body>	実際にブラウザに表示される部分	<body>...</body>
<title>	検索結果に表示されるタイトル	<title>自己紹介ページ</title>
<meta>	文字コード・説明・SEO などの情報	<meta charset="UTF-8">

テキスト・構造に関するタグ：

タグ	説明	例
<h1>～<h6>	見出し（数字が小さいほど大きな見出し）	タイトル
<p>	段落（paragraph）	こんにちは。
 	改行（ 閉じタグなし ）	こんにちは。 こんばんは。
	強調（太字＋意味的強調）	重要
	強調（イタリック）	大切
	文中の一部を囲む（インライン）	色付き文字
<div>	区切り・まとまりを作る（ブロック）	<div class="section">内容</div>

リンクやメディア関連のタグ：

タグ	説明	例
<a>	ハイパーリンク （他ページへのリンク）	プロフィールへ
	画像表示	
<video>	動画を埋め込み	<video src="movie.mp4" controls></video>
<audio>	音声を再生	<audio src="bgm.mp3" controls></audio>

フォーム（入力欄）関連のタグ：

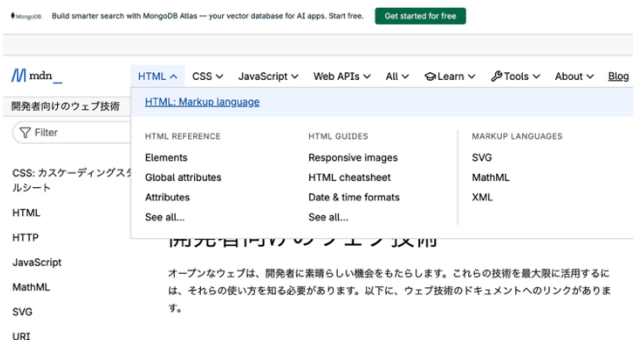
タグ	説明	例
<form>	入力フォーム全体	<form action="send.php" method="post">...</form>
<input>	入力欄（テキスト・ボタンなど）	<input type="text">, <input type="submit">
<textarea>	複数行のテキスト入力	<textarea></textarea>
<select>	プルダウンメニュー	<select><option>選択肢 1</option></select>
<button>	ボタン	<button>送信</button>
<label>	入力項目の説明	<label>名前：</label>

POINT：

HTML 要素（タグ）の調べ方

「MDN Web Docs」という、ウェブ開発者向けの技術ドキュメントサイトがあります。HTML、CSS、JavaScript などに関するリファレンスなどに関する学習コンテンツが提供されています。

（「mdn」で検索すると「開発者向けのウェブ技術 | MDN」というページが検索結果に表示されます。）



Step2 / CSS の基本

CSS とは、「Cascading Style Sheets（カスケーディング・スタイル・シート）」の略で、ウェブサイトの色、レイアウト、フォント、サイズなどを指定するための言語です。HTML がウェブページの骨組み（構造）を定義するのに対し、CSS は見た目（デザイン）を整える役割を担います。

step1.html を複製して、step2.html を作成してください。

その上で、以下のコードに変更したうえで、ブラウザで表示してください。

CSS の基本的な記述：

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <title>Demo</title>
  <style>
    /* ここに CSS(スタイル)の記述します */
    body {
      background: blue;
    }
  </style>
</head>
<body>
  <!-- ここに HTML を記述します -->
  <h1 style="color: white;">Hello world !</h1>
</body>
</html>
```

結果：

青い背景に、白い文字「Hello world !」が表示されたら正しく記述できています。

解説：

- スタイルの指定方法には 2 種類あります。
 - 1) <head>内に<style></style>を記述し、そのなかでスタイル指定。（body）
 - 2) タグに style="属性"を付与して、スタイルを指定。（h1）どちらの方法でも構いませんが、一つのタグに複数のスタイルを指定することがあるため 1)の方法で記述すると可読性があがります。
- HTML のコメントは、<!-- ○○○ --> ですが、CSS のコメントは /* ○○○ */ です。HTML のコメント同様に、コメントといいます。○○○の箇所は自由に文字を書くことができ、画面には表示されず、プログラムにも影響しません。

CSS のプロパティには色々な種類があります。いくつか例を以下に挙げます。

色・背景：

プロパティ	説明	例
color	文字の色	color: red;
background-color	背景色	background-color: #f0f0f0;
opacity	透明度 (0~1)	opacity: 0.8;

文字：

プロパティ	説明	例
font-size	文字の大きさ	font-size: 20px;
font-family	フォントの種類	font-family: "Arial", "Hiragino Sans", sans-serif;
font-weight	太さ (normal, bold など)	font-weight: bold;
font-style	スタイル (斜体など)	font-style: italic;
text-align	文字の位置 (左・中央・右)	text-align: center;
line-height	行の高さ (行間)	line-height: 1.6;
letter-spacing	文字間隔	letter-spacing: 2px;

余白：

プロパティ	説明	例
margin	外側の余白	margin: 20px;
padding	内側の余白	padding: 10px;

レイアウト：

プロパティ	説明	例
display	要素の表示方法 (block, inline, flex など)	display: block;
position	位置指定 (relative, absolute, fixed など)	position: absolute;
top / left / right / bottom	要素の位置	top: 50px; left: 100px;
z-index	要素の重なり順	z-index: 10;
overflow	はみ出した内容の扱い	overflow: hidden;
width	幅	width: 300px;
height	高さ	height: 200px;

装飾：

プロパティ	説明	例
border	枠線 (まとめて指定)	border: 2px solid #333;
border-radius	角を丸くする	border-radius: 10px;
cursor	カーソルの形	cursor: pointer;
transform	回転・拡大・移動など	transform: rotate(15deg);

単位：

単位	意味	例
px	ピクセル（絶対値）	font-size: 16px;
%	親要素に対する割合	width: 80%;
em	親要素のフォントサイズ基準	padding: 1.5em;
rem	ルート（html）フォント基準	font-size: 1.2rem;
vh / vw	画面の高さ・幅に対する割合	height: 100vh;

Step3 / 要素に名前をつける (id と class とは?)

HTML のタグに「名前をつける」ための属性として「id」「class」があります。
それを使って、CSS で「どの要素にスタイルを当てるか」を指定できます。

step2.html を複製して、step3.html を作成してください。

その上で、以下のコードに変更したうえで、ブラウザで表示してください。

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <title>Demo</title>
  <style>
    body {
      background: blue;
    }
    #title {
      color: white;
    }
    .note {
      color: red;
    }
  </style>
</head>
<body>
  <h1 id="title">Hello world !</h1>
  <h2 class="note">こんにちは !</h2>
  <p class="note">今日はいい天気ですね</p>
</body>
</html>
```

結果：

青い背景に、白い文字「Hello world !」が表示され、赤い文字で「こんにちは！」「今日はいい天気ですね」が表示されたら正しく記述できています。

解説：

- id は、HTML に「id="***"」で指定し、スタイルには「#***」で指定します。
- class は、HTML に「class="***"」で指定し、スタイルには「.***」で指定します。

注意点：

単位	意味	例
id 属性	ページ内で 1 つだけ使う ※ 複数箇所は NG	<p id="note">こんにちは</p> <p id="note">今日はいい天気ですね</p>
class 属性	ページ内で何個でも使える	<p class="note">こんにちは</p> <p class="note">今日はよい天気ですね</p>

Step4 / CSS でレイアウト・装飾する

CSS は、配色だけでなく、レイアウト、余白、文字の見た目、装飾、アニメーションなども指定できます。今回は、ブロックレイアウト（div）と余白（margin / padding）等について解説します。

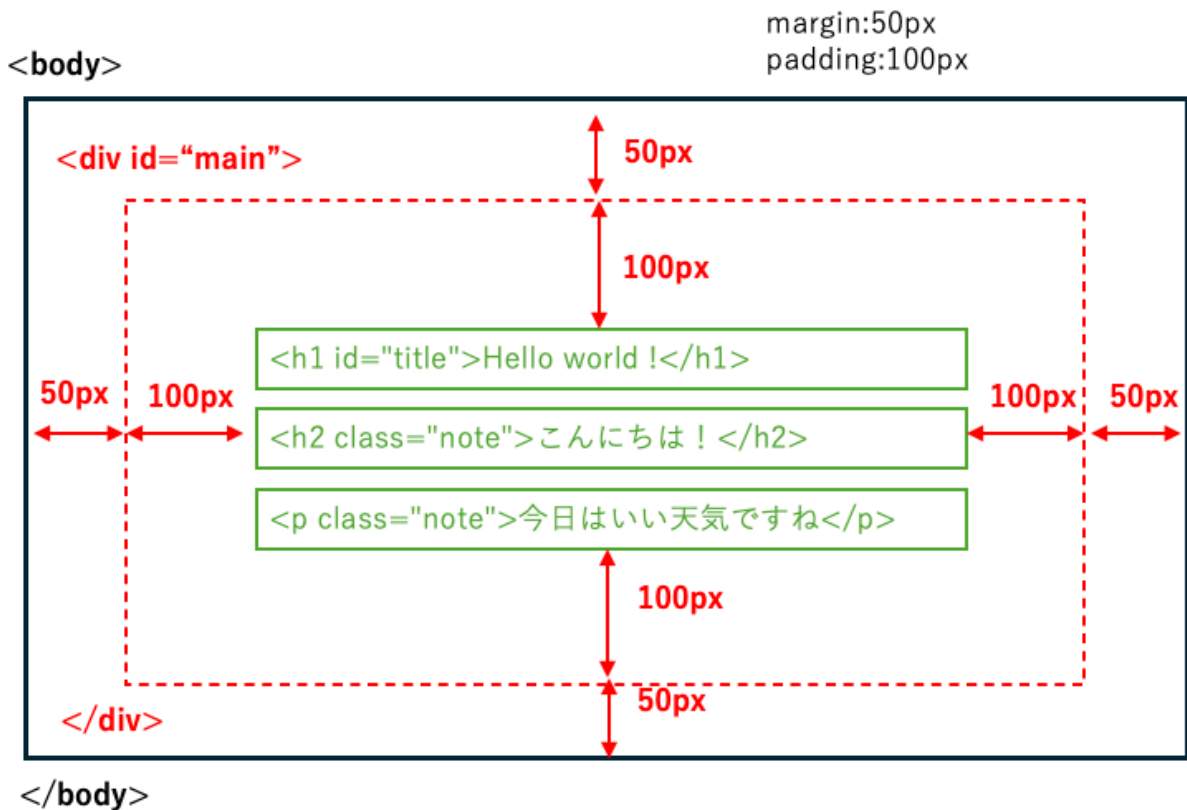
Step3.html を複製して、step4.html を作成してください。

その上で、以下のコードに変更したうえで、ブラウザで表示してください。

全体を囲うように <div id="main">を追加しています。

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <title>Demo</title>
  <style>
    body {
      background: blue;
    }
    #main {
      margin: 50px;
      padding: 100px;
      background: #ffffff;
      text-align: center;
    }
    #title {
      color: black;
      border: 2px solid #000000;
    }
    .note {
      color: red;
    }
  </style>
</head>
<body>
  <div id="main">
    <h1 id="title">Hello world !</h1>
    <h2 class="note">こんにちは !</h2>
    <p class="note">今日はいい天気ですね</p>
  </div>
</body>
</html>
```

解説：



- `<div>` は“division(区切り)”の略で、レイアウトやグループ分けのために使うブロック要素です。
- `margin` は、対象要素の外側余白、`padding` は、対象要素の内側余白です。
- `margin: 50px` は、`margin: 50px 50px 50px 50px;` とも書き換えられます。
`margin: {上} {右} {下} {左}` で記述できます。`padding` も同様です。
- `text-align: center;` は、文字の位置を左寄せ、中央寄せ、右寄せで指定できます。
- `border: 2px solid #000000;` は、枠線表示です。`border: {太さ} {線種} {色}` で指定します。
- `color`、`background-color`、`border-color` などの色指定は、主に以下の指定方法があります。
 - 1) `color: red;` ・ ・ ・ 色名。種類は限られる。
 - 2) `color: #ff0000;` ・ ・ ・ カラーコード（16進数）。Webでは一般的。
 - 3) `color: rgb(255, 0, 0);` ・ ・ ・ RGB値。赤・緑・青を0～255の範囲で指定。

各プロパティ（`margin` 等）の値を自由に変更して、表示が変わること確認してみてください。
また、Chrome の DevTool を利用すると簡単にスタイル要素の適用状態が確認できます。

Step5 / ゲームのスタート画面を作る

これまでに学んだ HTML と CSS を使い、ゲームのスタート画面を作ります。

以下は、ワイヤーフレームです。ワイヤーフレームとは **Web サイトやアプリのレイアウト構成を線や枠で表した設計図**のことです。



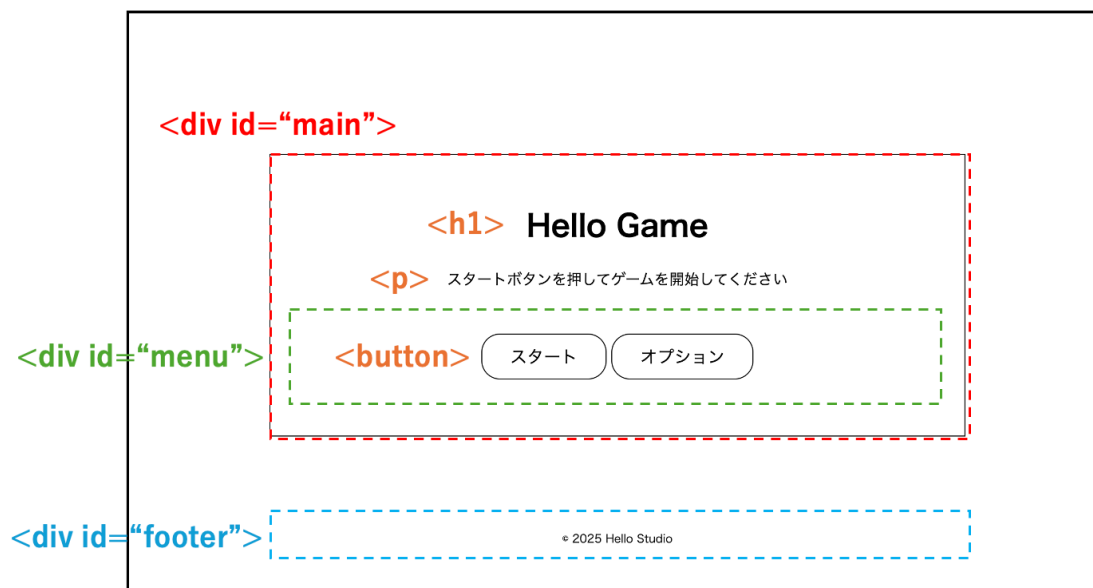
Step5 (1)

step1.html を複製して、step5.html を作成してください。

まずは、以下を参考に HTML のコードを書いてください。（CSS はまだ使いません）

その上で、以下のコードに変更したうえで、ブラウザで表示してください。

<body>



HTML :

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <title>Demo</title>
</head>
<body>
  <div id="main">
    <h1>Hello Game</h1>
    <p>スタートボタンを押してゲームを開始してください</p>
    <div id="menu">
      <button>スタート</button>
      <button>オプション</button>
    </div>
  </div>
  <div id="footer">
    (c) 2025 Hello Studio
  </div>
</body>
</html>
```

結果 :

Hello Game

スタートボタンを押してゲームを開始してください

© 2025 Hello Studio

Step5 (2)

続けて、目標の画面に近づけるように、CSS のコードを書いてください。

コードを記述したらブラウザで表示確認をして、コードを調整してください。

使用するプロパティは、「text-align」「margin」「padding」「border」の4つです。

HTML :

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <title>Hello Game</title>
  <style>
    #main {
      text-align: center;
      padding: 30px;
      border: 1px solid #000;
    }
    #menu {
      padding: 30px;
    }
    #footer {
      text-align: center;
      margin: 50px;
    }
    button {
      border: 1px solid #000;
      padding: 10px 30px;
    }
  </style>
</head>
<body>
  <div id="main">
    <h1>Hello Game</h1>
    <p>スタートボタンを押してゲームを開始してください</p>
    <div id="menu">
      <button>スタート</button>
      <button>オプション</button>
    </div>
  </div>
  <div id="footer">
    &copy; 2025 Hello Studio
  </div>
</body>
</html>
```

結果：



Step5 (3)

ここからは、CSS を使って自由にデザインを工夫してみましょう。
フォントの種類や色、文字の大きさ、ボタンを角丸にするなど
自分の好みに合わせて変更してみてください。

Step6 / JavaScript

JavaScript（ジャバスクリプト）とは、主にウェブサイト動きや対話性を持たせるために使われるプログラミング言語です。HTML がウェブページの構造を、CSS がデザインを定義するのに対し、JavaScript はユーザーの操作に応じて要素を動的に変化させたり、計算を行ったりします。

JavaScript の記述：

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <title>Demo</title>
</head>
<body>
  <h1>Hello world</h1>

  <script>
    // ここに JavaScript の記述します
    alert("ケロケロ");
  </script>
</body>
</html>
```

解説：

- <script></script> のように、ブラウザに対して「ここに JavaScript 等のプログラムが記述されている」と認識させるために、プログラム部分をこれらのタグ（開始タグと終了タグ）で囲んで記述します。
- alert("〇〇〇"); という関数は、〇〇〇という文字列をブラウザにダイアログ出力します。
- JavaScript のコメントは 一行の場合、// と記述します。
複数行の書くこともできます。その場合は CSS コメントと同じ /* 〇〇〇 */ で囲みます。

Step7 / 入力・出力

JavaScript からの入力・出力方法にはいくつかの方法があります。

- ・ ダイアログ：「alert()」「confirm()」「prompt()」
- ・ コンソール：「console.log()」 ※ 開発中に利用
- ・ HTML（DOM 操作）：「document.getElementById("msg").textContent」 等

alert() :

```
<script>
  alert( "ケロケロ" );
</script>
```

alert() は""の文字をダイアログ表示します。「OK」のみのボタンを表示します。

confirm() :

```
<script>
  // 確認ダイアログを表示
  confirm("カエルが好きですか?");
</script>
```

- confirm() は「OK」「キャンセル」の2択ボタンを表示します。
- OK を押すと true、キャンセルを押すと false が返ります。

prompt() :

```
<script>
  // 入力ダイアログを表示
  prompt("あなたの名前を教えてください:");
</script>
```

- prompt() は「入力欄」と「OK」「キャンセル」のボタンを持つダイアログです。

console.log() :

```
<script>
  console.log( "こんにちは" );
</script>
```

alert()のようにJavaScript に書いた文字を表示します。

ただし、alert()はダイアログ表示することに対して、console.log()は Dev Tools に出力します。

DOM 操作：

```
<p id="note"></p>

<script>
  document.getElementById("note").textContent = "いい天気ですね！ ";
</script>
```

alert()はダイアログ表示することに対して、html タグを指定して操作し内容を置き換えます。

これを DOM 操作といいます。

DOM (Document Object Model)は、HTML などのマークアップ言語で書かれた文書を、JavaScript などのプログラミング言語から操作・変更できるようにするための API (Application Programming Interface) です。

なお、基本的に、プログラムの行末には「;」の記号を書きます(省略することもできますが、習慣として付けるようにしましょう)。

Step8 / 変数

const : 定数 = 再代入が不可。

```
<script>
  const b = 3;
  alert( b );
</script>
```

let : 変数 = 再代入が可能。

```
<script>
  let a;
  a = 3;
  alert( a );
</script>
```

2 行を 1 行にまとめて

```
<script>
  let a = 3;
  alert( a );
</script>
```

以下に箇条書きで「変数」の説明を行います。

- ・ 変数は、値の入れ物、あるいは値を入れる箱と考えることができる
- ・ 変数は、英数字で（文字数は自由）で表される（例：a x b1 b2 time sokudo 等々）
- ・ ユーザの入力や、プログラムの進行によって、中身の値は変化することがある
（変化させることができる）
- ・ 「=」の記号を使って、変数に値を代入できる（「=」の右側から左側へ代入する）
- ・ 変数は自由に作れるが、「if」や「alert」等、プログラム自体が機能を提供する

ために使う「予約語」は変数として使うことはできない

- ・ 変数は、なるべくわかりやすい名前をつけることで、他の人（あるいは時間が経った後の自分）がプログラムを見たときに理解しやすくなります。
- ・ JavaScript では、プログラム中で使用する変数は、プログラムの冒頭で、let という書式を使って「この文字(列)を、変数として使用する」ことを宣言しておくのが一般的な書き方になります。（これらの記述は省略することもできます）。

Step7 では `alert("ケロケロ");` の書式で直接的に表示内容を指定しましたが、`alert(a);` と記述することで、変数 `a` の中身を表示することができます(上述のプログラムを実行すると、メッセージとして「3」が表示されます。

Step9 / データの種類

変数に格納できるデータの種類：

```
<script>
  let name = "ケロケロ";    // 文字列(String)
  let age = 10;              // 数値(Number)
  let isFrog = true;         // 真偽値(Boolean)
  let colors = ["赤", "青"]; // 配列(Array)
  let info = { name: "カエル", color: "緑" }; // オブジェクト(Object)

  alert( name );
</script>
```

文字列は “” で囲み、数値は 囲み不要です。

真偽値は特殊で `true/false` の文字ですが囲み不要です。

配列、オブジェクトは 別途説明します。ここでは異なる記法になることを理解してください。

また、未定義 (`undefined`) や `null` といったデータの種類がありますが、割愛します。

Step10 / 演算子 (四則演算等)

```
<script>
  // 変数の準備
  let a = 5;
  let b = 2;

  // 各演算
  let add = a + b;    // 足し算
  let sub = a - b;    // 引き算
  let mul = a * b;    // 掛け算
  let div = a / b;    // 割り算
  let mod = a % b;    // 余り
  let pow = a ** b;   // べき乗(aのb乗)
</script>
```

注意点：

データ型の違いにより演算結果が変わります。以下の変数 a と 変数 b の値は何になるでしょうか。

```
<script>
  let x = "60"; // 文字列
  let y = 60;   // 数値

  let a = x + x;
  let b = y + y;
</script>
```

Step11 / ランダム生成

```
<script>
  let num = Math.random();
  alert( num );
</script>
```

以下に箇条書きで「**Math.random()**」の説明を行います。

- ・ゲーム等に不可欠な「ランダムさ」を導入するための方法です。
- ・プログラム中でランダム性が必要になる局面は様々な場面が考えられますが、JavaScriptの場合は、すべてこの命令を基にして作ります。
- ・「Math.random ()」は呼び出すたびに（使用するたびに）異なる値を返します。
- ・「Math.random ()」は0以上1未満の範囲の小数を、一つ選んでくれる（ゲットできる）関数です。「Math.random ()」の Math は数学 (mathematics) の意味で、Math.random () の他に、サイン関数の Math.sin () や、絶対値を求める関数の Math.abs ()、小数点以下を切り捨てる関数 Math.floor () 等の、数学的な機能が用意されています。（これらの詳細は別途説明します）
- ・上述のプログラムは、変数 num を使用せずに、

```
alert( Math.random() );
```

という一行を書くことで、同じ動作が実現できます（複数の括弧が組み合わされていますが、基本的に括弧の内側から処理が行われます）。ただし、上述のプログラムのように、一旦変数にランダムな値を代入して(確保して)、以降の処理を、その変数を使って行う方法が一般的です。

とりあえず、0 以上 1 未満のランダムな小数がそのまま表示されるだけのプログラムですが、これはもっとも原始的な「おみくじ」（あるいは「占い」）と言えると思います。例えば値が 1 に近ければめでたく、0 に近いとめでたくない、というルールで考えることができます。

Step12 / 条件分岐

```
<script>
  let num = Math.random();

  if( num > 0.5 ){
    alert( "吉" );
  }

  if( num <= 0.5 ){
    alert( "凶" );
  }
</script>
```

if 文の役割は、

- ・条件が満たされている場合は、指定された処理内容を実行する
- ・条件が満たされていない場合は指定された処理は実行しない（スキップする）

となります。if 文の書き方は実行内容が 1 行であれば、

```
if (条件) { 条件が満たされた場合に行う処理 ; }
```

と書けますが、一般的に行う処理は複数行にわたる場合が多く、そのような場合は、下記のように書きます。なお、{ } の中に記述される実行内容は、左側にスペースを入れて記述を右側にずらすことで、それらが if 文の実行内容であることがわかりやすくなります。（これをインデントといいます）

```
if (条件) {
  条件が満たされた場合に行う処理 1 ;
  条件を満たされた場合に行う処理 2 ;
  条件を満たされた場合に行う処理 3 ;
}
```

「条件」の部分では、基本的に二つの値の比較を行います。値としては、直接「0.5」等の数値を置くことや変数を置くことができます。二つの値の比較のために、以下の記号を使うことができます。

比較演算子

- > : 左辺の値が右辺の値より大きい場合は・・・
- < : 右辺の値が左辺の値より大きい場合は・・・
- > = : 左辺の値が右辺の値以上の場合は・・・

<= : 右辺の値が左辺の値以上の場合は・・・
== : 左右の値が同じ場合は・・・
!= : 左右の値が異なる場合は・・・

1 頁の 7 番目の項目で、文字の間に空白を入れることで、プログラムが見やすくなると説明しましたが、上記の「==」や「>=」等、if 文の条件指定に使う記号については、間に空白を入れてはいけないことに注意してください。また、== と書くべきところを、単に = と書いてしまう間違いが多発しますので、注意して下さい。「=」を 1 個だけ使うと、既に説明した右辺の値を左辺の変数に入れるための「代入」の意味になり、「==」とは別の意味になります。javascript をはじめとする様々なプログラミング言語がこの「==」を採用していますが、私は間違いやすい良くない仕様だと思っています。

Step13 / if 文を増やしてメッセージの種類を多様にする

```
<script>
  let num = Math.random();

  if( num > 0.5 ){
    alert( "大吉です" );
  }

  if( num > 0.25 && num <= 0.5 ){
    alert( "吉です" );
  }

  if( num <= 0.25 ){
    alert( "凶です" );
  }
</script>
```

上述のプログラムで、これまで 2 種類だったメッセージを 3 種類に増やすことができました。プログラムの記述から、大吉、吉、凶、それぞれの出現確率を考えてみて下さい。プログラム中に新しい記号「&&」が含まれています。この「&&」を使っている二つめの if 文では、変数の値の上限と下限の両方が指定されています。このことは、中学、高校の数学では、

$0.25 < \text{num} \leq 0.5$

というようなコンパクトな書き方で条件を指定できたのを憶えている人もいますが、JavaScript ではこのように書くことができず、二つの不等号に分けた上で、それを組み合わせる記号を

使う必要があります。今回は、二つ不等号で示された条件の両方を満たす必要があるので「and（かつ）論理」を適用する必要がありますが、and 論理を示す記号が「&&」になります。条件の組み合わせ方には、この and（かつ）の他に or（または）もありますが、or は「||」で記述します（「|」の記号は、「¥」のキーのシフトで打つことができます）。なお、これらの && と || は、文字の間に空白を入れてはいけません。

上述のプログラムの、「メッセージ」「それぞれの出現確率」「異なるメッセージの数」を自由に変えて試してみてください。

Step14 / if と else の組み合わせ

```
<script>
  let num = Math.random();

  if( num > 0.5 ){
    alert( "吉です" );
  } else {
    alert( "凶です" );
  }
</script>
```

```
if (条件) {
  条件が満たされた場合に行う処理；
} else {
  条件が満たされなかった場合に行う処理；
}
```

---=

課題：

Step14 のプログラムは、else を使う if 文の別の書き方を使って同じ動作を実現することも可能です。

```
if (条件 1) {
  条件 1 が満たされた場合に行う処理；
} else if (条件 2) {
  条件 1 を満たさず、条件 2 が満たされた場合に行う処理；
} else {
  条件 1 も 2 も満たされなかった場合に行う処理；
}
```