



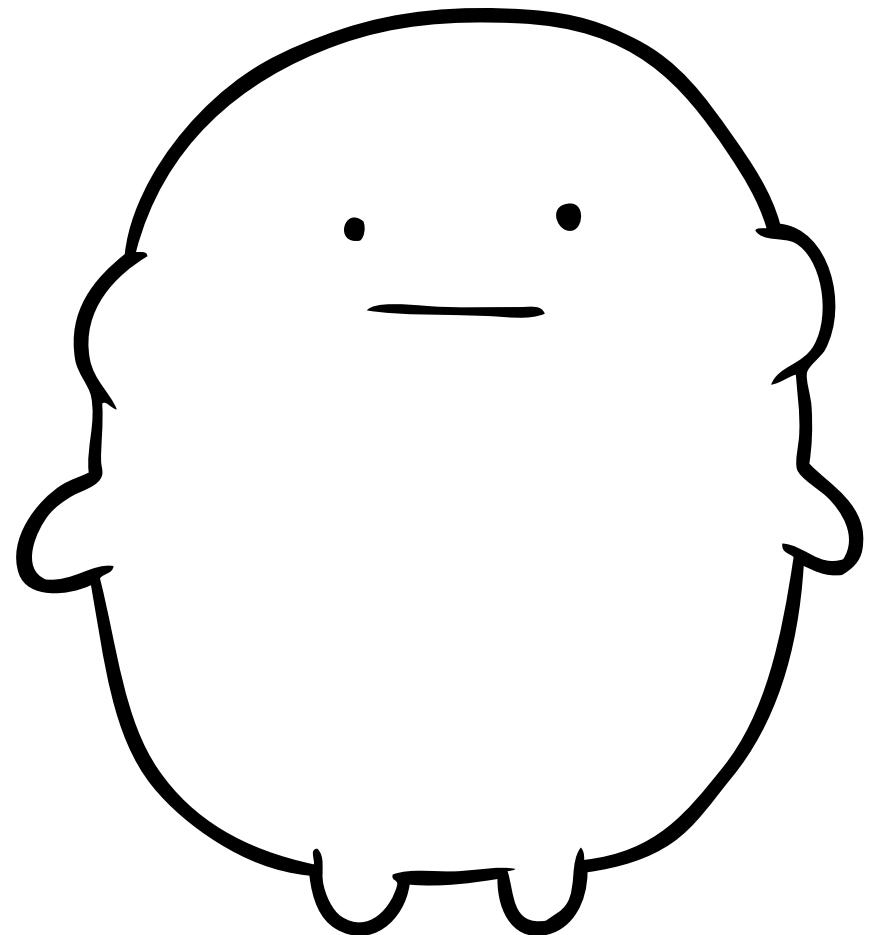
# JavaScript

---

Introduction

# 講師紹介

About



## Profile

名前

紅 尚志(べに たかし)

出身

宮城県仙台市

専門

WEB開発や制作など

趣味

筋トレ、ゲーム(古い)など

メッセージ

WEBで最も幅広く使われる言語です。ある意味なんでもできてしまうので、体系的に理解しましょう。

プログラム  
専門です！

みなさんの話を率直に  
聞かせてください！



学びたいこと、目標など

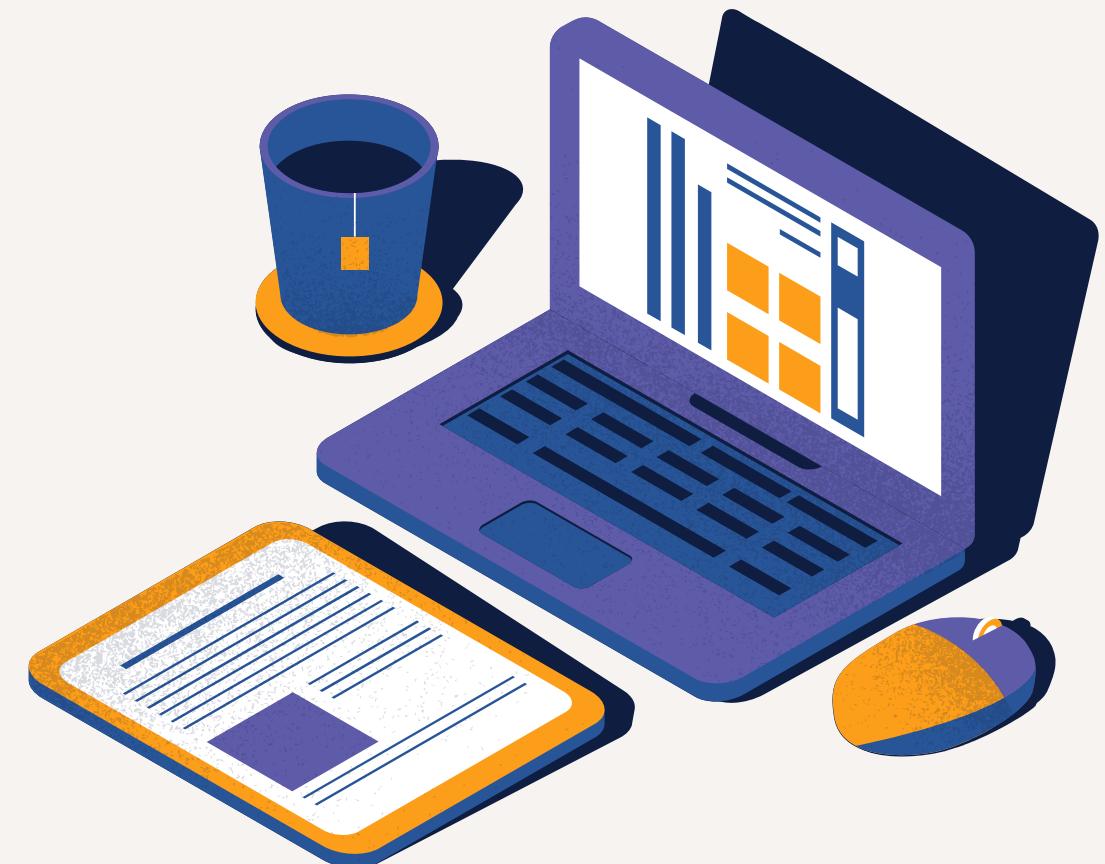
# 講義のルール

---

共通のルールにしたがいましょう

## 【この授業のルール】

- ・私用以外の休みは素直に申告してください  
課題なら課題で構いません
- ・演習中は体をほぐしても大丈夫です
- ・自分でテーマ設定する課題は60~70%程度で終わらせられるくらいにしましょう  
高いモチベがあるなら自主製作がおすすめです
- ・その日の課題が終わらなそうなときも教えてください



# 講義の流れ

---

予習・復習



講義



例題



解説



演習・課題



解説



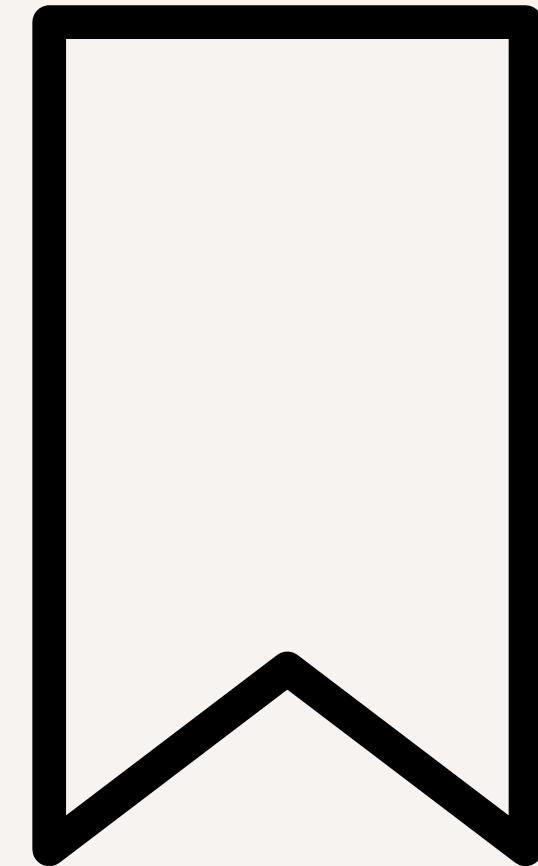
まとめ

# 教科書

---



Mana, 1冊ですべて身につく JavaScript入門講座,  
SBクリエイティブ(<https://www.sscr.jp/product/4815615758/>)

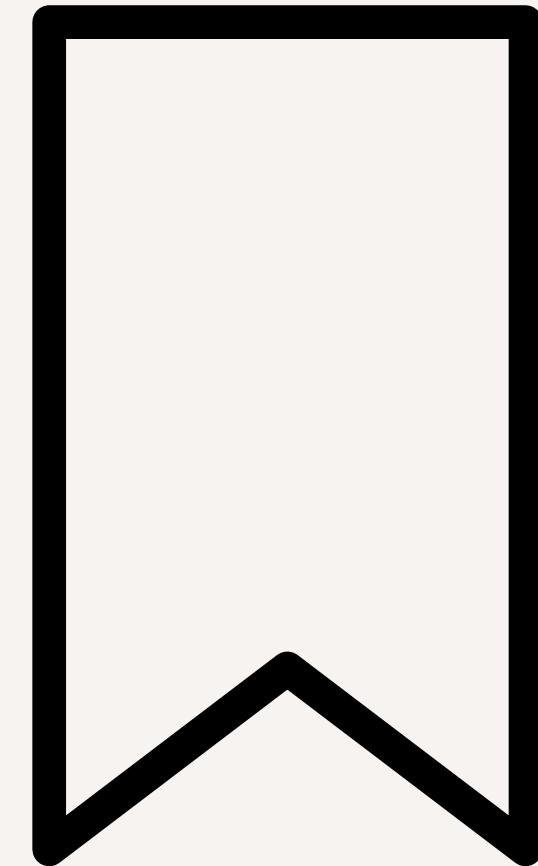


# 参考として

---



Dustin Boswell、Trevor Foucher 著、角 征典 訳、リーダブルコード、  
オライリージャパン(<https://gihyo.jp/book/2022/978-4-297-13128-9>)



最大目的：納品できる！



# 到達目標

これから1年間をかけて身につけていく技術的な要件です。

クライアントサイドのJSを理解する

JSは非常に範囲が広いです。その中でも、クライアント、つまり各端末で動かす方法を理解します。JSにおける最もスタンダードな使い方です。

外部データとの連携ができるようになる

JSではAPI連携というものをよく使います。動的なデータ変更や、サーバー処理の肩代わりをして高速化など、活用用途は計り知れません。ぜひ習得しましょう。

エラーの解決方法を導き出せる

JSはHTML等と違いプログラミング言語のひとつです。CSSなどよりも解決に時間がかかりやすく、より早く正確に原因を特定する能力を求められます。

# 年間の計画を立てよう



4月

- ・前期授業開始

5月

6月

7月

8月

9月

10月

11月

12月

1月

2月

3月

- ・WordPress制作完了
- ・講義終了



ここまで  
質問はありますか？

---

遠慮せずに挙手・発言してください！

# 小休止



# 今回の内容

---

1. JavaScriptとは？
2. 基本的な書き方
3. まとめ

# JavaScriptとは？



# まずは読んでみよう

---



- Javaとの違いは何でしょうか
- 何のために使われるでしょうか
- CSSアニメーションとの違いは何ですか

1冊で身につくJavaScript P14~P26

## 時間軸を与えることができる

JSを使うことによって、時間差や条件に合致した時のみ動作させるような動きが可能になります。きめ細かなアニメーション処理や、より適材適所なデザイン、あるいはカラーリング変更やフォントサイズ変更といったアクセシビリティ対応なども可能になります。

## 内外とのデータ連携ができる

JSのバックグラウンド処理を併用すると、ページを遷移することなく表示を変更することができます。無駄な描画がなくなり、高速な動作が可能です。

例: GoogleMap検索

## 発展領域が非常に広い

サーバーサイドで動かすnode.js、ウェブからスマホアプリまで作れるReact、PCアプリに強いElectronなど、なんでも作れるといつていいほど範囲が広いです。一昔前までは2大ゲームエンジンの1つにも対応していたのですが廃止となりました。それを差し引いても、ゲームとマイコン以外はおよそ何とかなるだけのポテンシャルを持っています。



ここまで  
質問はありますか？

---

遠慮せずに挙手・発言してください！

# 保存場所などについて

---

<https://isbn2.socr.jp/15758/>

↓

<https://www.socr.jp/support/4815617728/>

→ 「【ダウンロード】『1冊ですべて身につくJavaScript入門講座』サンプルデータ」：  
「JS\_Data-2.zip」

【保存場所(候補)】

「~/school/cr3/html-css-js/」

# コードを書いてみよう(2-3)

---

- ✓自分のプロジェクトを作って実際に書いてみましょう
- ✓コードは「自分で」書きましょう(単元によって変える可能性があります)

【保存場所(候補)】

「~/school/cr3/html-css-js/」, ディレクトリ名: 「JS\_MyProject/chapter2/demo○○」  
デモファイルと同じ場所で構いません(日本語は使わない)。後期では動かないリスクのある  
単元では保存場所に制限をかける可能性が高いです。

名前	更新日時	種類	サイズ
JS_Data-2	4/16/2025 9:48 AM	ファイル フォルダー	
JS_MyProject	4/16/2025 9:51 AM	ファイル フォルダー	

# 内部と外部の記述方法(2-3)

---

## 【インラインスクリプト】

HTML文書内にスクリプトを書く行為を「インラインスクリプト」などと呼びます。『インラインで書いて』と言われたらこの書き方です。

- ✓ 通常のHTMLでは非効率な場合が多いですが、CMSやフロントエンドでは活躍します
- ✓ 記述順序が煩雑になるリスクがあるので乱用には注意しましょう

## 【外部スクリプト】

外部のスクリプトを「src」属性で読み込むものを「外部スクリプト」などと呼びます。jQueryをはじめとしたライブラリを使う場合は全てこちらにあたります。

- ✓ 同期、非同期によって表示速度やスコアに影響があります(いずれやります)
- ✓ ファイルにまとまっているので、流れを掴みやすく原因の特定がしやすいです



ここまで  
質問はありますか？

---

遠慮せずに挙手・発言してください！

# 書き方の基本ルール(2-4)

---

※教科書の「パラメター」は「パラメータ(ー)」表記が一般的です

- ・オブジェクトやメソッドについてはいずれやりますのでさらっと理解していくください
- ・セミコロンは必要です。ただし、自動で挿入する機能あるため省略できる仕様です
- ・文字列の指定にはプラスで「バッククォート」を使った方法があり、特定の条件で真価を発揮します(P68)
- ・エスケープはよくある混乱ポイントです。必要になったら学びなおしましょう



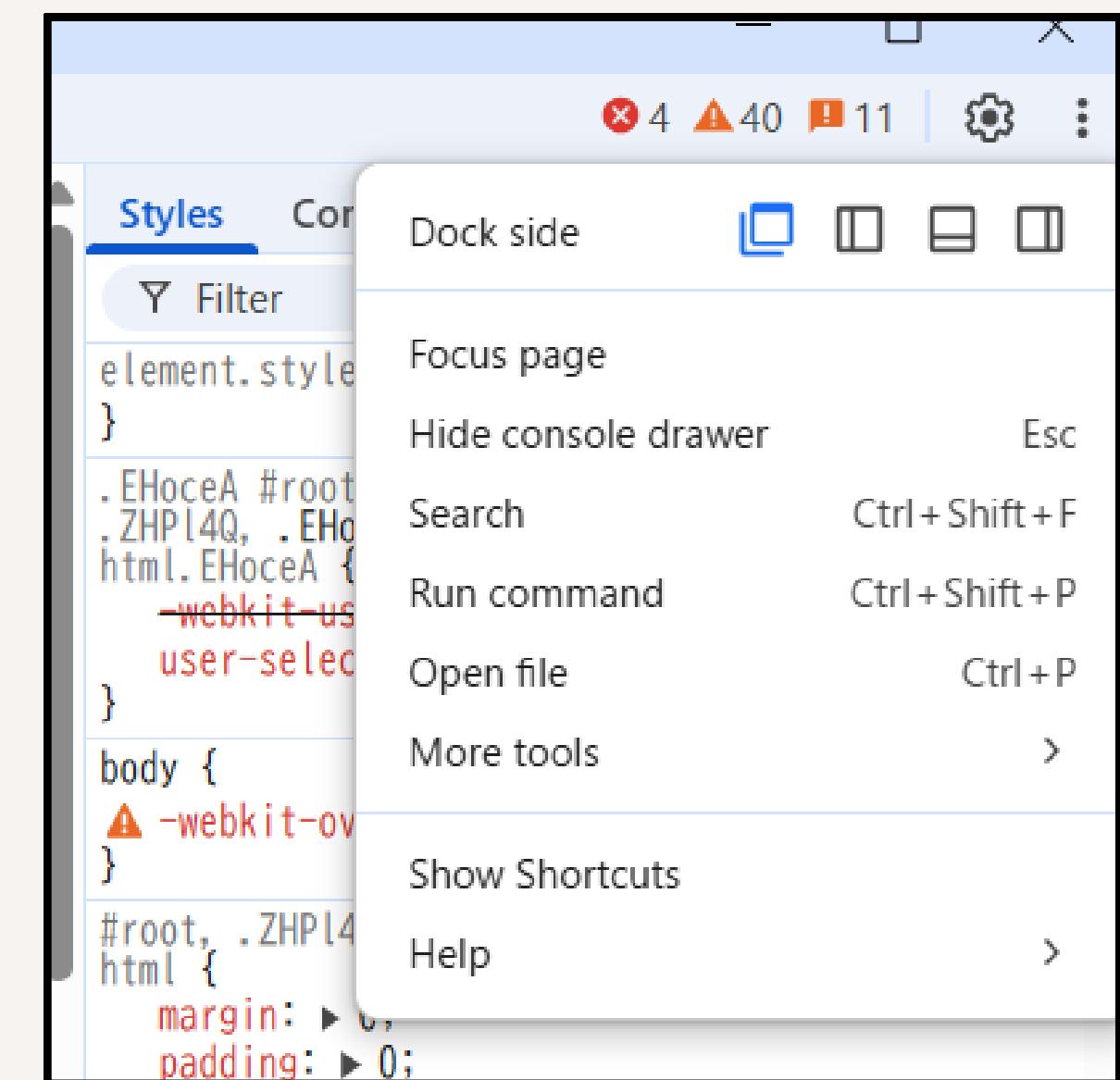
ここまで  
質問はありますか？

---

遠慮せずに挙手・発言してください！

# コンソールを使ってみよう(2-5)

- ・ ウィンドウ幅が小さく感じる場合は、別ウィンドウ化してみましょう(ちなみに別ウィンドウ派)
- ・ 例題もやってみましょう





ここまで  
質問はありますか？

---

遠慮せずに挙手・発言してください！

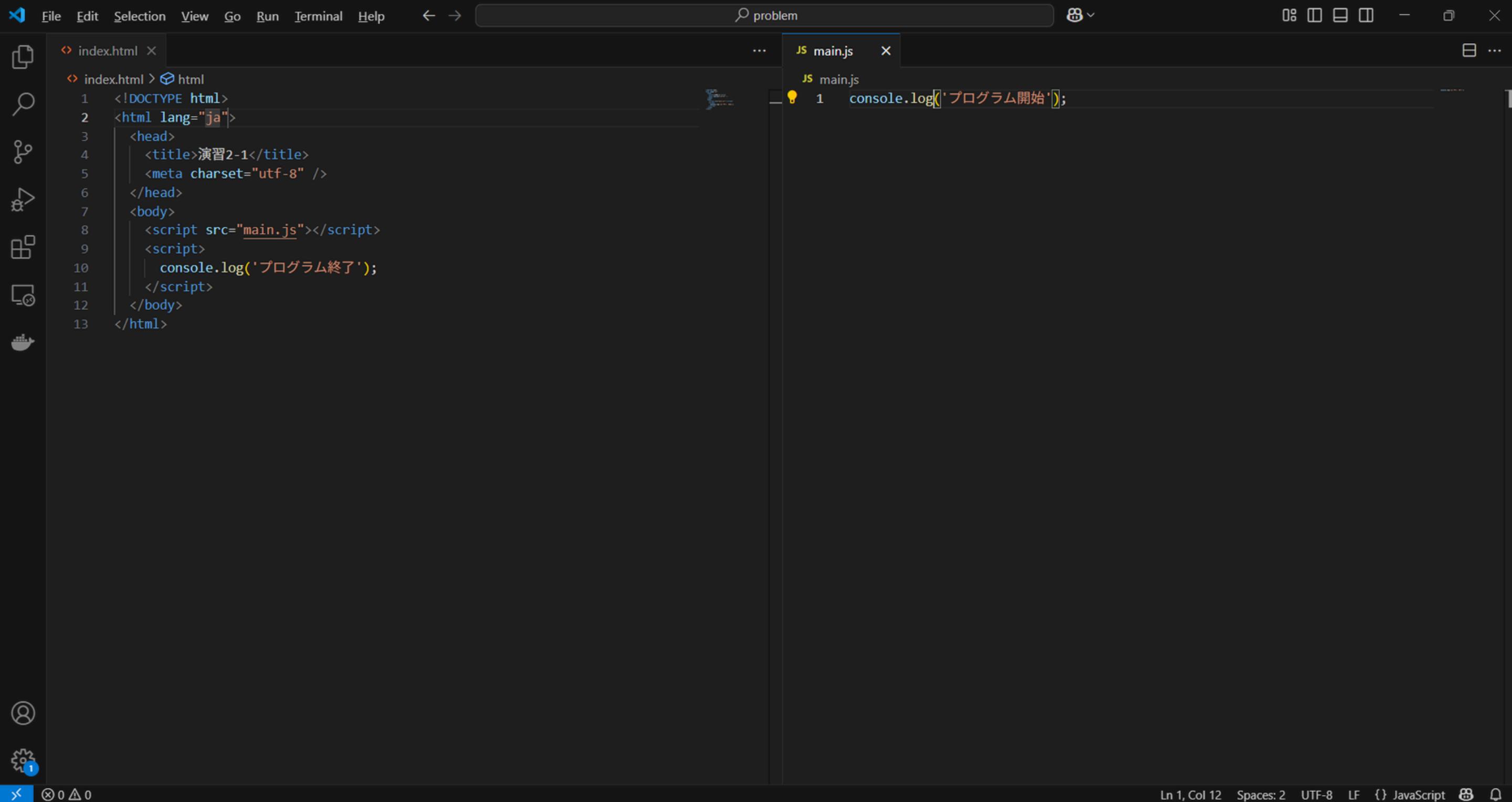
# 演習2-1

---

以下の条件に合致するコードを記述してください。

1. 「JS\_MyProject/chapter2/problem」 ディレクトリを作成する
2. 上記に 「index.html」 と 「main.js」 を作成する
3. 「index.html」 に、 **コンソール**で「プログラム終了」と表示するプログラムを作成する
4. 「main.js」 では、 **コンソール**で「プログラム開始」と表示するプログラムを作成する
5. タイトルなどは任意

# 演習2-1 - 回答例



The screenshot shows a dark-themed code editor interface with two files open:

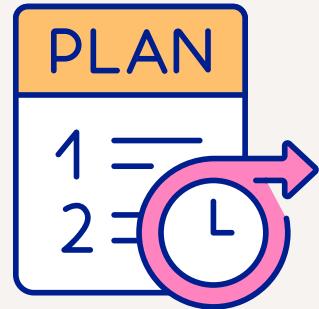
- index.html**:  
A simple HTML document with a title and a script tag linking to `main.js`. The script contains a single line of code to log a message to the console.
- main.js**:  
A JavaScript file with one line of code that logs the string 'プログラム開始' to the console.

The code editor includes standard navigation and search tools, and a status bar at the bottom indicating the current file type is JavaScript.

```
index.html
<!DOCTYPE html>
<html lang="ja">
<head>
<title>演習2-1</title>
<meta charset="utf-8" />
</head>
<body>
<script src="main.js"></script>
<script>
| console.log('プログラム開始');
</script>
</body>
</html>

main.js
console.log('プログラム開始');
```

# 今日の授業のまとめ



## 学習計画

大まかに今年度の取り組みについて紹介しました。今後も疑問点が出れば隨時質問してください。



## 2つの記述方法

文書内に直接書く方法と、外部ファイルに分ける方法を学びました。状況に応じて使い分けてください。



## 簡単なメソッドを使用

alertやconsoleなどのオブジェクトに属するメソッドを使用しました。今後も使っていくので、この使い方を忘れないようにしてください。



# JavaScript

---

実習2 - ページの書き換え／定数／条件分岐

# 今回の内容 - 教科書準拠

---

1. JavaScriptの基本を学ぼう

# JavaScriptの基本を学ぼう



# 3-2 defer関数について

- 
- defer: スクリプトの非同期読み込みで、deferで宣言した順に実行される
  - async: スクリプトの非同期読み込みで、順不同で実行される

## defer

```
<script src="jquery.js" defer></script>    順1  
<script src="jquery.ext.js" defer></script>  順2
```

defer同士の順序が守られるため基本はこちらを使いたい。deferとasync、deferと未記載(同期読み込み)の混在では順序が崩れるため、依存するファイル同士は同じルールで記述しよう。

## async

```
<script src="jquery.js" async></script>    順?  
<script src="jquery.ext.js" async></script>  順?
```

読み込み順が不明なため、完全に独立したファイルでのみ使うことができる。いずれにせよ最終的には読み込むため、最も早いというものでもない。

## 3-3 コメントアウトについて

- 
- 「//」を記述した行はコメントとして処理されJSが実行されない
  - 「/\*」と「\*/」で囲んだ範囲は全てコメントとして処理される

### 複数行コメントの注意点

```
1  /*  
2   <script>  
3     /*console.log('コメントアウト1');  
4     console.log('コメントアウト2');//  
5   </script>  
6 */
```

複数行コメントで囲う場合、二つ以上のコメントアウトが干渉する可能性がある。  
良く起こることではないが、まとめて作業する時など見逃さないように注意すること。

### コメントの鉄則(リーダブルコードより)

- コードから読み取れることを書かない、価値の提供できないコメントは書かない
- コメントを書くならコメントのメンテナンスもセットで行う
- 変数の名前、条件の指定など、プログラムのロジックで解決できることをコメントで済まそうとしない

# 3-7 定数constのルール

- 
- 数値や文字列などを指定した場合は再代入ができない
  - オブジェクトや配列を指定した場合は、プロパティや中のデータ編集は可能
    - あくまで「参照先」を保存しているだけ

例)

top



under

```
const books = under;  
under.append( 'js' : 'javascript入門講座' );
```

あくまで本棚の一番下の場所を記憶したような処理になる。  
したがって、本を取り出したり、入れたりといったことは可能。  
ただし、「books」を改めて一番上「top」の段に設定し直すなどは不可。

JavaScriptの基本を学ぼう

## 3-11 條件分岐の考え方

- 
- 日常生活に置き換えて考えてみよう



「もしも」障害物があったら



「もしも」プリンに名前が  
書いていなかったら



## 3-11 條件分岐と「型」

- 
- 「==」と「==='の違いとは?
    - 「==」では型を考慮しない(暗黙的に型変換を行う)
    - 「==='では型を考慮する
  - 「<=」や「>」などの不等号でも型の変換が行われる

====

- 両方とも一致しない

```
if (1 === '1' )
```

```
if (0 === null)
```

==

- 両方とも一致する

```
if (1 == '1' )
```

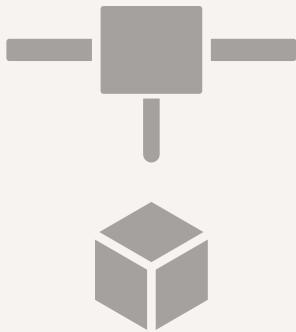
```
if (0 == null)
```

# 今日の授業のまとめ

F

## 関数

いつまでも使うであろう関数の基本を学ぶ。アロー関数と通常の関数の違いを理解する。



## 定数

定数の宣言方法と例外的なルールを学ぶ。大きなプロジェクトでもバグの少ないプロジェクトの構成を理解する

iF

## 条件分岐

基本的な条件分岐の使い方を理解する。ほんの少しの違いでバグの原因にもなる「型」違いを理解する。



# JavaScript

---

実習3 - イベントで操作しよう！ - ロードイベント／クリックイベント

# 今回の内容 - 教科書準拠

---

1. イベントで操作しよう！ - ロードイベント／クリックイベント

イベントで操作しよう！ - ロードイベント／クリックイベント

# 4-1 イベントの基本

---

## イベントの基本

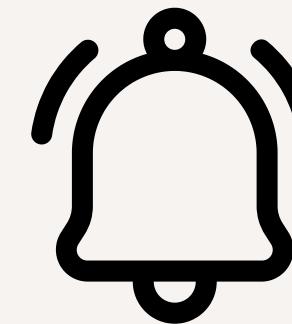
HTML／CSSなどで作ったページにJSで時間軸を与えることができる。

そして、具体的な行動(アクション)があったら特定の行動をとるための仕組みを「イベント」という。



スマホを鳴らす

電話がきたら...



講義が終わる

チャイムが鳴ったら...

# 4-3 windowやdocument

---

- 「window」や「document」など、どのイベントに設定するかはリファレンスを確認する
  - <https://developer.mozilla.org/en-US/docs/Web/API/Window>
  - <https://developer.mozilla.org/en-US/docs/Web/API/Document>

習うより慣れよ

# 今日の講義のまとめ

E

## イベント

イベントの基本を学び、この後学んでいく条件分岐やデータとの連携を見据えた下地を作る。



# JavaScript

---

実習4 - イベントで操作しよう！ - 条件分岐と組み合わせ／スクロールイベント

# 今回の内容 - 教科書準拠

---

- 1.4-6 if / else文で表示する文字を切り替えよう
- 2.4-7 入力した文字数を数えてみよう
- 3.4-8 lengthでカウントしよう
- 4.4-9 文字数によって表示を変えよう
- 5.4-10 チェックをいれるとボタンを押せるようにしよう
- 6.4-11 チェックでボタンを有効化しよう
- 7.4-12 より効率のいい書き方を考えよう
- 8.4-13 ページのスクロール量を表示しよう
- 9.4-14 スクロール量を取得しよう
- 10.4-15 ページのサイズを取得しよう
- 11.4-16 計算式を書いてみよう

重要!

4-8 lengthでカウントしよう

# lengthとは？

---

JavaScriptの標準オブジェクト

JavaScriptでは、様々なものが「オブジェクト」として定義されています。

イベントで使用している「document」や「window」だけでなく、数値や文字列も実はオブジェクトです。

したがって、同様に標準機能として「length」をはじめとしたプロパティやメソッドが提供されています。

[https://developer.mozilla.org/ja/docs/Web/JavaScript/Reference/Global\\_Objects/String](https://developer.mozilla.org/ja/docs/Web/JavaScript/Reference/Global_Objects/String)

## 4-12 より効率のいい書き方を考えよう

# 論理式の応用方法

---

論理式の計算結果は真理値である

※真理値: 「true」 / 「false」

if文の中で使われる真理値は、計算結果が真理値になります。

チェック状態などの変数だけでなく、関数の判定などにも使われます。

例)

```
function isPositiveValue(num) {  
    return num >= 0; // 0以上ならtrue, 0未満ならfalse  
}
```

## 4-16 計算式を書いてみよう

# 論理演算子とビット演算

---

if文でのビット演算ミスに注意

JavaScriptを始め多くのプログラミング言語でビット演算が使用できます。

よく似た式なので、記述ミスに注意しましょう。

今回は「AND」を取り上げていますが、「OR」や「XOR」(ほぼ使わない)なども存在します。

【論理式】

```
if (a === b && a === c)
```

【ビット演算(エラーにならず式としても成り立つ)】

```
if (a === b & a === c)
```

ビット演算「&」の詳細)

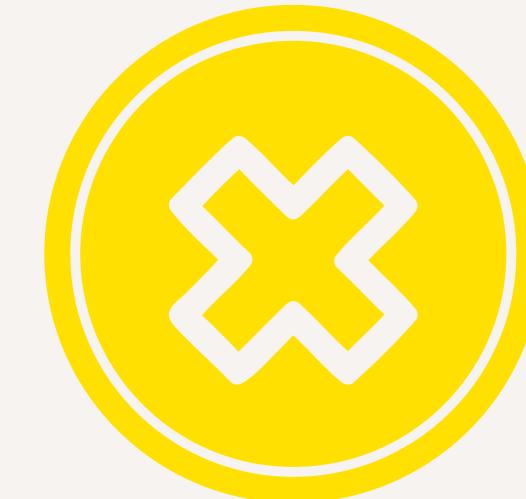
[https://developer.mozilla.org/ja/docs/Web/JavaScript/Reference/Operators/Bitwise\\_AND](https://developer.mozilla.org/ja/docs/Web/JavaScript/Reference/Operators/Bitwise_AND)

# 今日の講義のまとめ

E

イベント

イベントの基本を学び、この後学んでいく条件分岐やデータとの連携を見据えた下地を作る。



組み合わせ

if

条件分岐

基本的な条件分岐の使い方を理解する。ほんの少しの違いでバグの原因にもなる「型」違いを理解する。



# JavaScript

---

実習5 - 複数のデータを使ってみよう！ - 繰り返し文

# 今回の内容 - 教科書準拠

---

- 1.5-1 作成する画像一覧ページの紹介
- 2.5-2 InsertAdjacentHTMLでHTMLタグを挿入しよう
- 3.5-3 配列で複数の画像のファイル名をまとめてみよう **重要!**
- 4.5-4 配列の中にある画像を表示しよう
- 5.5-5 for分の繰り返し処理を理解しよう **重要!**
- 6.5-6 for分で画像を一覧表示しよう

## 5-1 作成する画像一覧ページの紹介

# 概要と目的

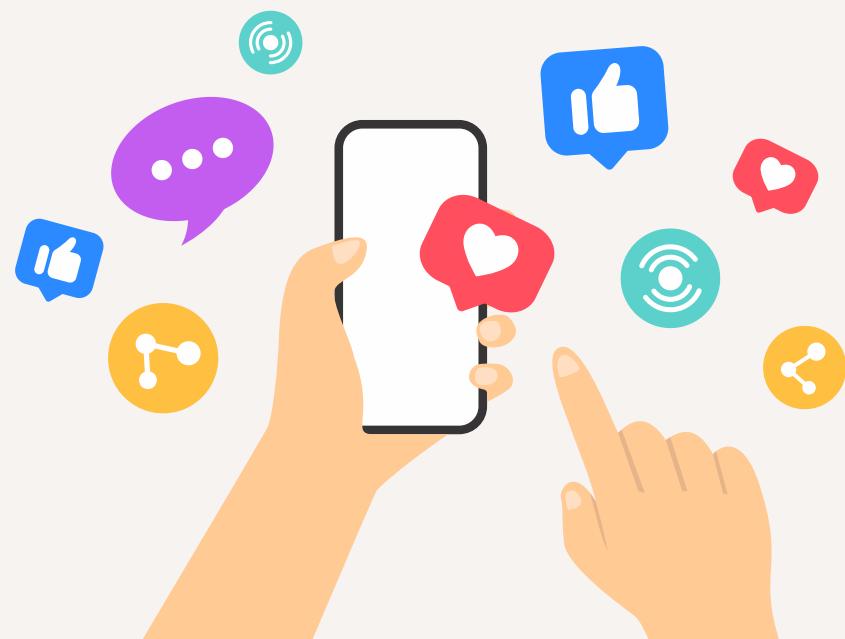
---

## 概要

- JavaScriptで表示する画像や情報を記述する

## 目的

- あらかじめ設定した情報を元にページを生成できるようになる
- 動的に情報を変更する、ということを理解する - SNSにならって



個別にHTMLを一々書かなくとも、SNSなら投稿、商品なら商品登録で、決められたフォーマットで情報を生成できる。

X/Youtube/Instagram/Amazon/楽天/その他

## 5-2 InsertAdjacentHTMLでHTMLタグを挿入しよう

# 変数とWebフォント

---

### 変数とIDの制約について

変数やIDとして使えない場合(予約済みの記号など)を除いて、大きな制約はありません。

【○同じIDを複数の変数で指定する】

```
const menu1 = document.querySelector( '#menu' );
const menu2 = document.querySelector( '#menu' );
```

### Webフォントをcssから読み込む方法

- @font-faceを使うと、CSSから読み込むことも可能です。
- <https://developer.mozilla.org/ja/docs/Web/CSS/@font-face>

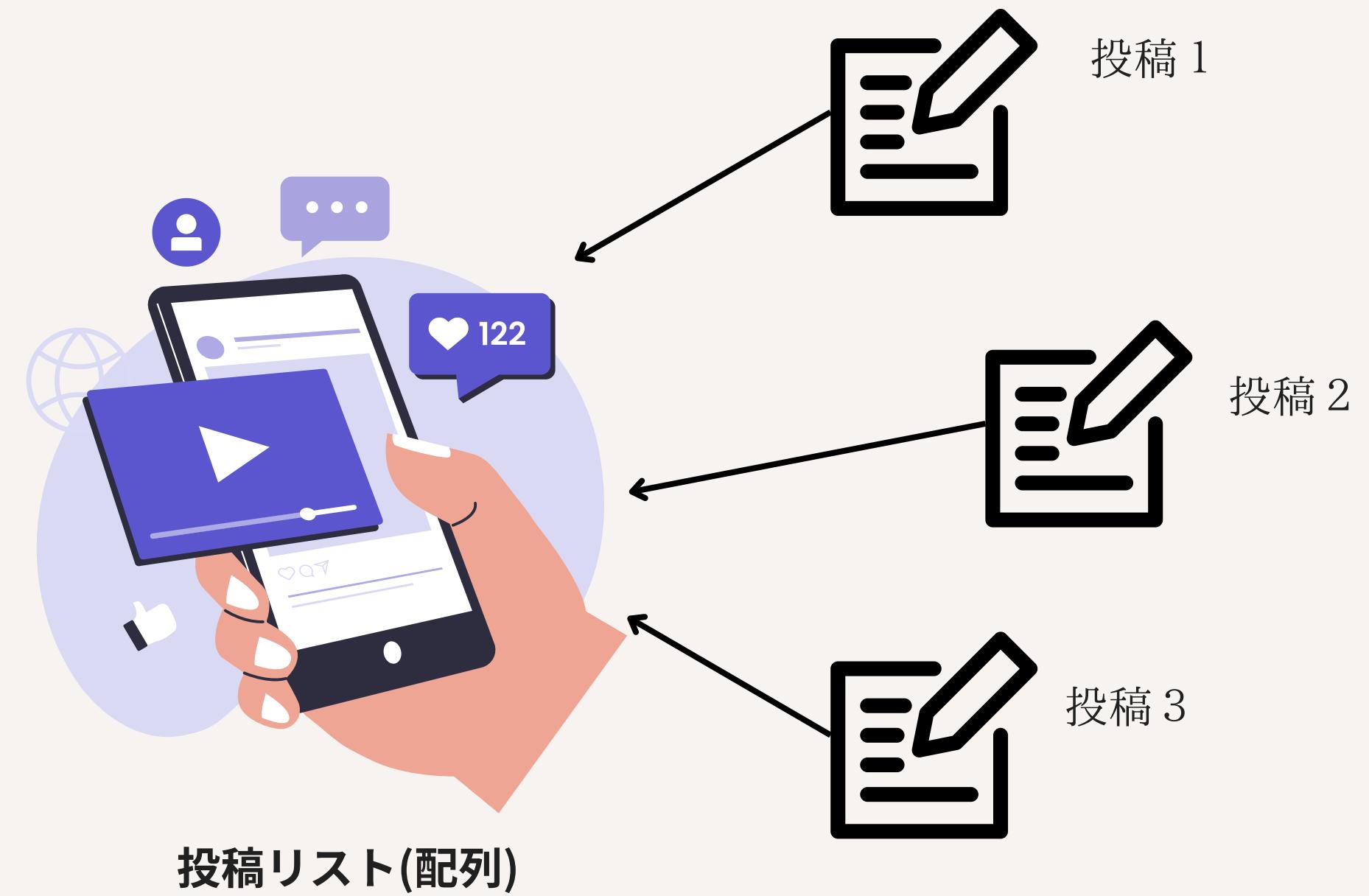
5-3 配列で複数の画像のファイル名をまとめてみよう

# 配列の考え方

---

SNSで考えてみよう

配列×オブジェクト(5-8)



## 5-5 for分の繰り返し処理を理解しよう

# インクリメント処理

---

### インクリメントの意味

for (let i = 0; i < 10; **i++**) ← 「**i++**」：インクリメント、変数の値を『1ずつ増加』させる時に使う

#### ■以下のコードは意味として同じ

- `i = i + 1;` // 標準形
- `i += 1;` // 上記の短縮記法(加算代入)
- `i++;` // iを『1ずつ増分する』という状況でのみ使用できる構文、主にループ用
- **X** `i++++;` // iを2ずつ増やすときは、標準形か短縮記法を使う

#### ■デクリメント処理

インクリメントとは逆に、『変数の値を1ずつ減少』させる時に使う

- `i--;` // その他の表記を含めて、加算の記号「+」を「-」に変えるだけ

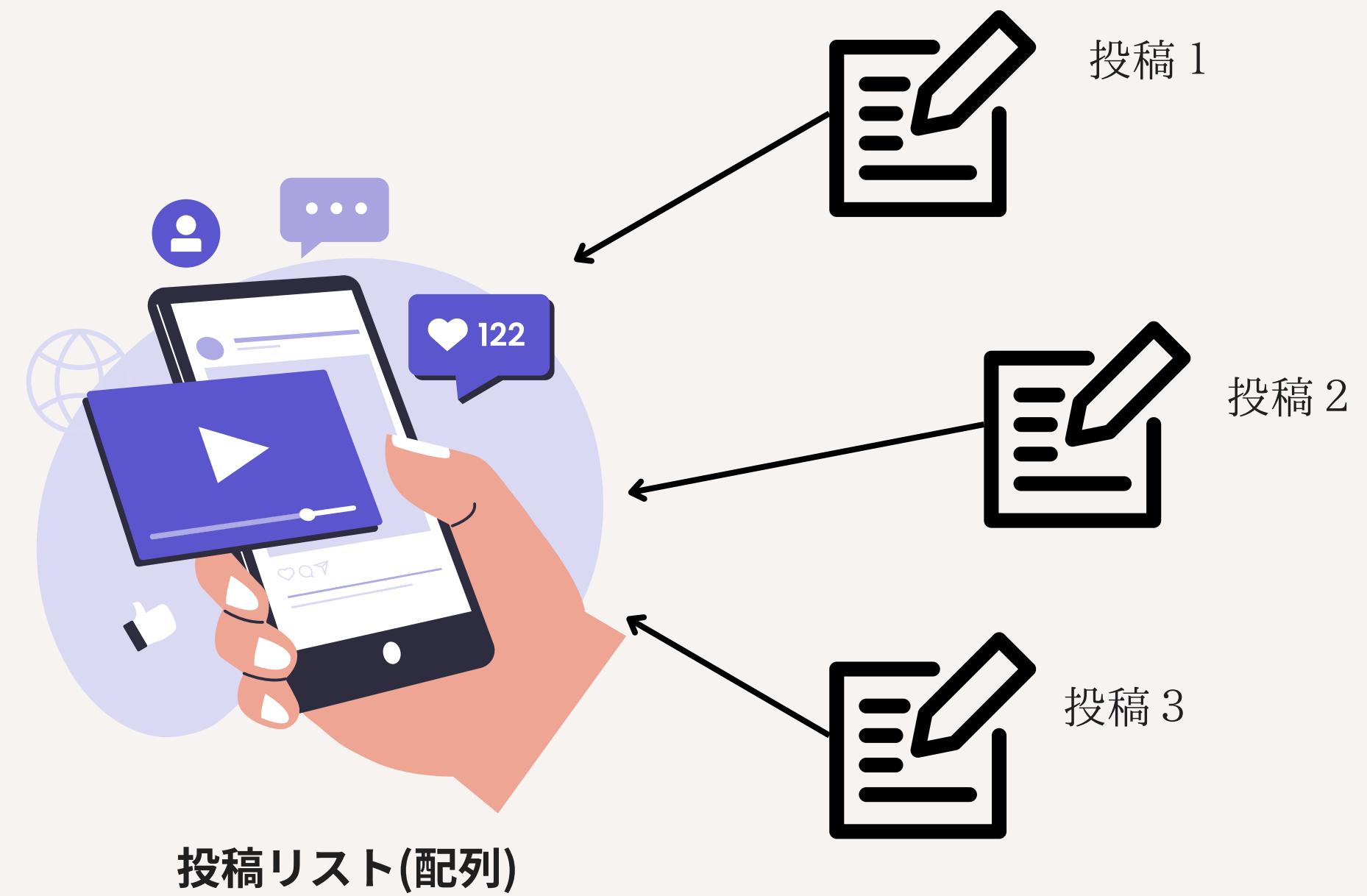
5-3 配列で複数の画像のファイル名をまとめてみよう

# 配列の考え方

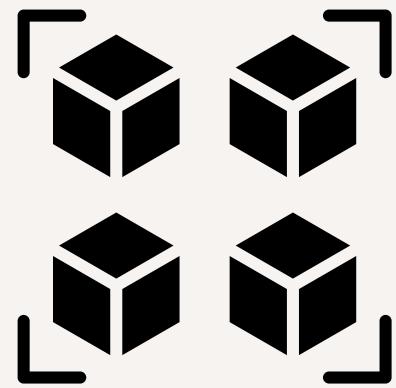
---

SNSで考えてみよう

配列×オブジェクト(5-8)



# 今日の講義のまとめ



配列

データを一元的に、効率よく管理する方法を学ぶ(活用方法は次回以降)



繰り返し

繰り返しはプログラム化(自動化)の第一歩。しっかり使いこなせるようになろう。



# JavaScript

---

実習5 - イベントで操作しよう！ - 条件分岐と組み合わせ／スクロールイベント