JS/TS基礎・環境構築

🗓️ Week 1：JavaScript / TypeScript の基礎固め & 開発環境準備

🎯 目的

この週は「モダンなWebアプリ開発の共通言語」である JavaScript（JS）と TypeScript（TS）の**基礎力を身につけること**が目的です。

1️⃣ 開発環境を整えよう

✅ 使うツール一覧（すべて無料）

ツール名	用途
Node.js	JavaScriptをPC上で実行するランタイム（サーバー用JS）
npm / yarn	パッケージ管理ツール（Node.jsに付属）
VSCode	コードを書くためのエディタ
Git	バージョン管理ツール（チーム開発でも必須）
Docker Desktop for Mac	開発環境の仮想化。今後の環境構築で使用予定

🛠️ インストール手順（Intel Mac 用）

Node.js（LTS版）

- 公式：https://nodejs.org

  または

- Homebrew 使用（ターミナルで）

```
brew install node
```

VSCode

- https://code.visualstudio.com/

Git（インストールされていない場合）

ターミナルで以下を実行するとインストールが始まります。

```
git --version
```

Docker Desktop for Mac（後で使うけど入れておこう）

- https://www.docker.com/products/docker-desktop/

2️⃣ JavaScript（JS）の基礎を学ぼう

🌱 なぜ学ぶの？

- LaravelはPHPを使っていましたが、モダンなWebフロントは**JSが主役**
- React/Next.js、NestJS も全て **JS または TypeScriptベース**

📚 覚えるべき構文（実行環境：Node.js）

✅ 変数の宣言

```
let name = "野呂";
const age = 23;
```

✅ データ型と演算

```
let score = 85;
let isPass = score >= 60;
console.log(isPass); // true
```

✅ 配列・オブジェクト

```
const fruits = ["apple", "banana", "grape"];
const user = { name: "野呂", age: 23 };
```

✅ if文・for文

```
for (let i = 0; i < fruits.length; i++) {
  console.log(fruits[i]);
```

```
}
```

✅ 関数定義

```
const greet = (name) => {
  return `こんにちは、${name}さん！`;
};
```

✅ 非同期処理：Promise / async-await

```
const getData = async () => {
  const res = await fetch("https://api.example.com/data");
  const json = await res.json();
  console.log(json);
};
```

3️⃣ TypeScript（TS）の導入と型の概念

🎓 TypeScriptとは？

> JavaScriptに型の仕組み（型注釈）を追加した言語。
> 間違いに開発中に気づける → バグを防ぎやすい。

🧪 簡単なTSコード例（`example.ts`）

```
function add(a: number, b: number): number {
  return a + b;
}
console.log(add(1, 2)); // 3
```

🛠️ 実行方法

```
npm install -g typescript
tsc example.ts
node example.js
```

🧠 よく使う型

種類	例
number	`let age: number = 23;`
string	`let name: string = "野呂";`
boolean	`let ok: boolean = true;`
array	`let items: string[] = ["a", "b"];`
object	`type User = { name: string; age: number };`

4️⃣ DOM操作（HTMLとJSの連携）

💡 目的

- React学習前に「ブラウザの中でJSがどう動くか」をイメージできるようにする。

👨‍💻 例：ボタンクリックで文字を表示

`index.html`

```
<!DOCTYPE html>
<html>
  <head><title>Hello JS</title></head>
  <body>
    <button id="btn">クリックしてね</button>
    <p id="msg"></p>
    <script src="script.js"></script>
  </body>
</html>
```

`script.js`

```
document.getElementById("btn").addEventListener("click", () ⇒ {
  document.getElementById("msg").textContent = "こんにちは、JS！";
});
```

✅ 開く方法

- Finderでファイルを右クリック → 「ブラウザで開く」

📝 ミニ課題（アウトプット用）

No.	内容	ヒント
1	`hello.js`：Node.jsでHello Worldを表示	`console.log()`
2	`calculator.ts`：TSで足し算・引き算関数を作る	`number`型を使う
3	`api-fetch.ts`：fetchで天気API取得して表示	`async` / `await`
4	Gitで管理 → GitHubにpush	`git init`, `git add`, `git commit`

📖 参考教材

- MDN Web Docs（JavaScript）
- TypeScript公式ハンドブック（日本語）
- Qiita記事：TypeScript初心者が基礎を学ぶ
- 書籍：『サバイバルTypeScript』（入門におすすめ）

✅ 今週のゴールまとめ

- Node.jsとTypeScriptを使って**「手元でコードを書く → 実行」**までできるようにする。
- 変数・関数・非同期処理・型の概念を**「書いて、動かして、覚える」**
- Gitでコードを管理し、**バージョン管理の流れ**にも慣れる。
