

Day0 事前学習

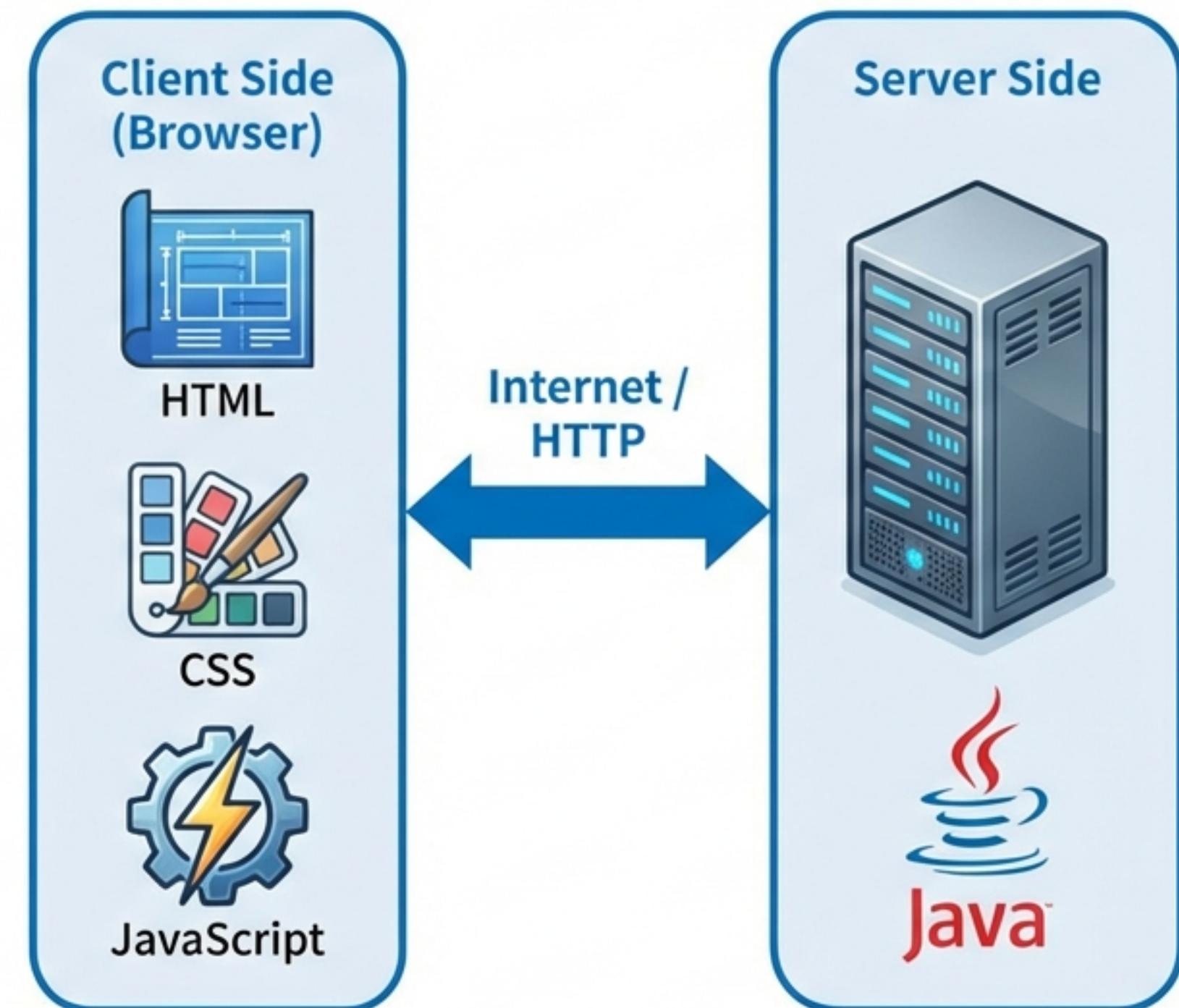
Web技術とJavaプログラミングの基礎

基本構文とコーディング規約



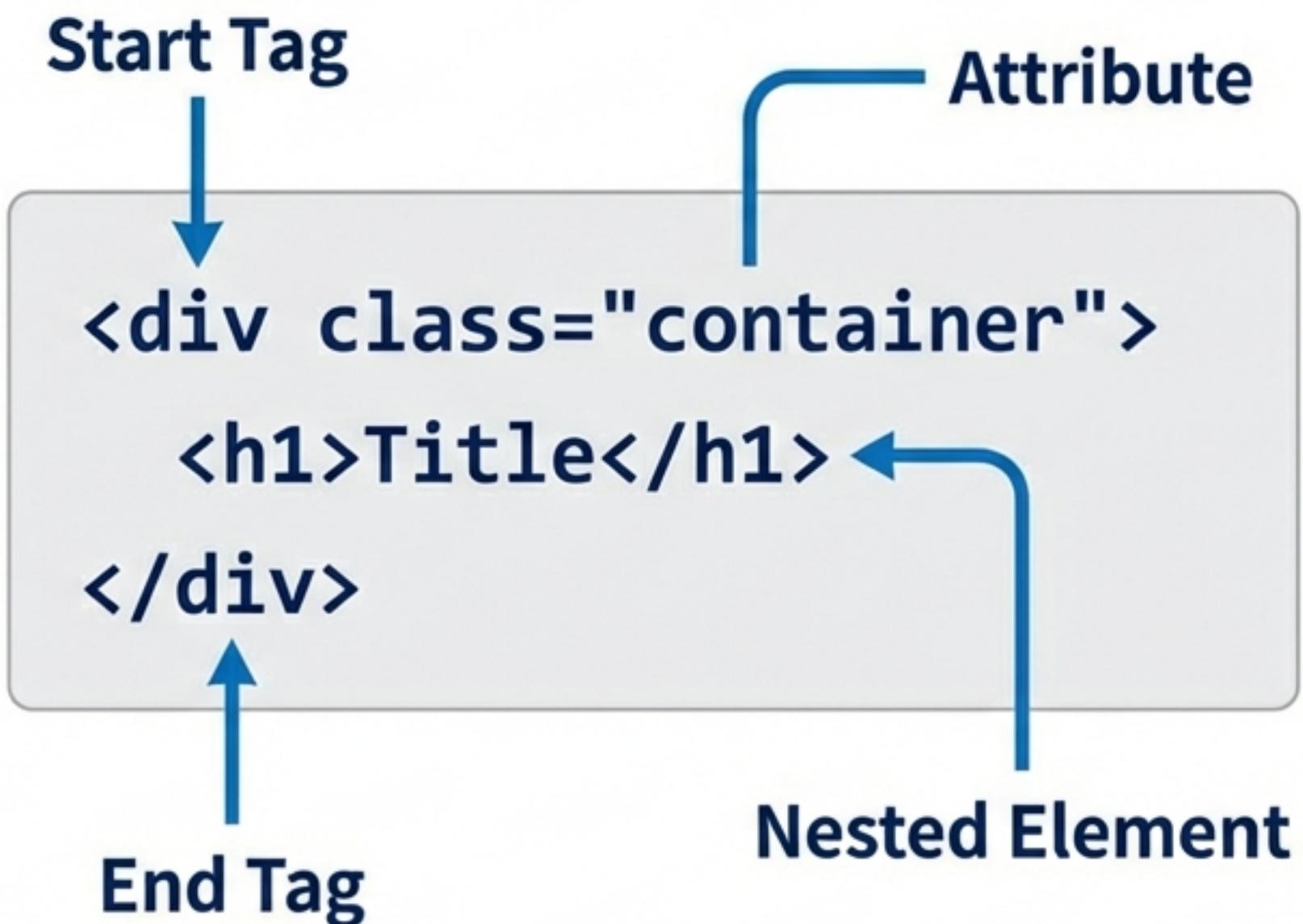
Webアプリの全体像

- HTML: 画面の骨組み・構造
- CSS: 画面の見た目・装飾
- JavaScript: 画面の動作・動き
- Java: サーバー側の業務処理



HTML (構造)

- ・タグ: <>で囲んで意味を持たせる
- ・属性: タグに追加情報を与える
- ・入れ子構造: 箱の中に箱を入れる
イメージ



CSS（見た目）

- HTMLを装飾する役割
- 色、フォント、配置などを指定
- HTMLとは別のファイルで管理するのが基本

HTML Only

Click Me

HTML + CSS

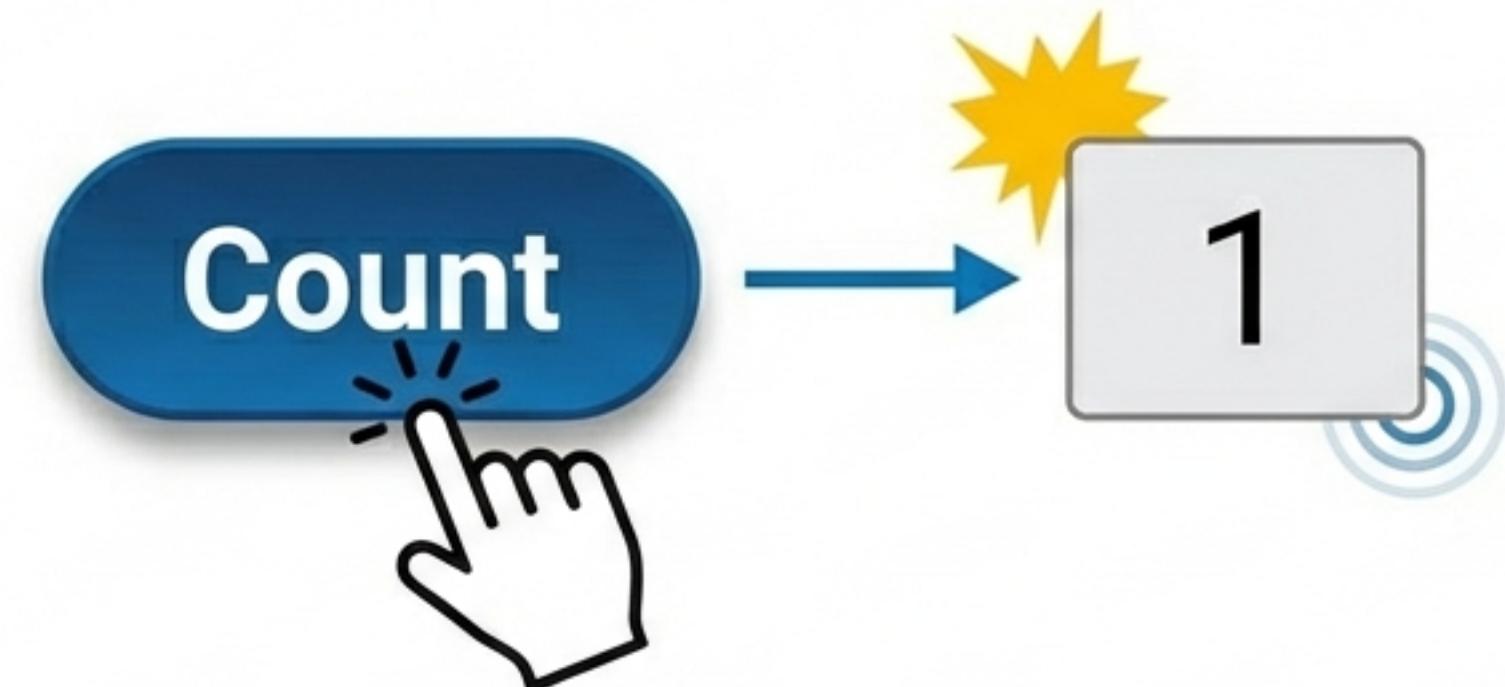
Click Me

background: blue;
color: white;

JavaScript (動作)

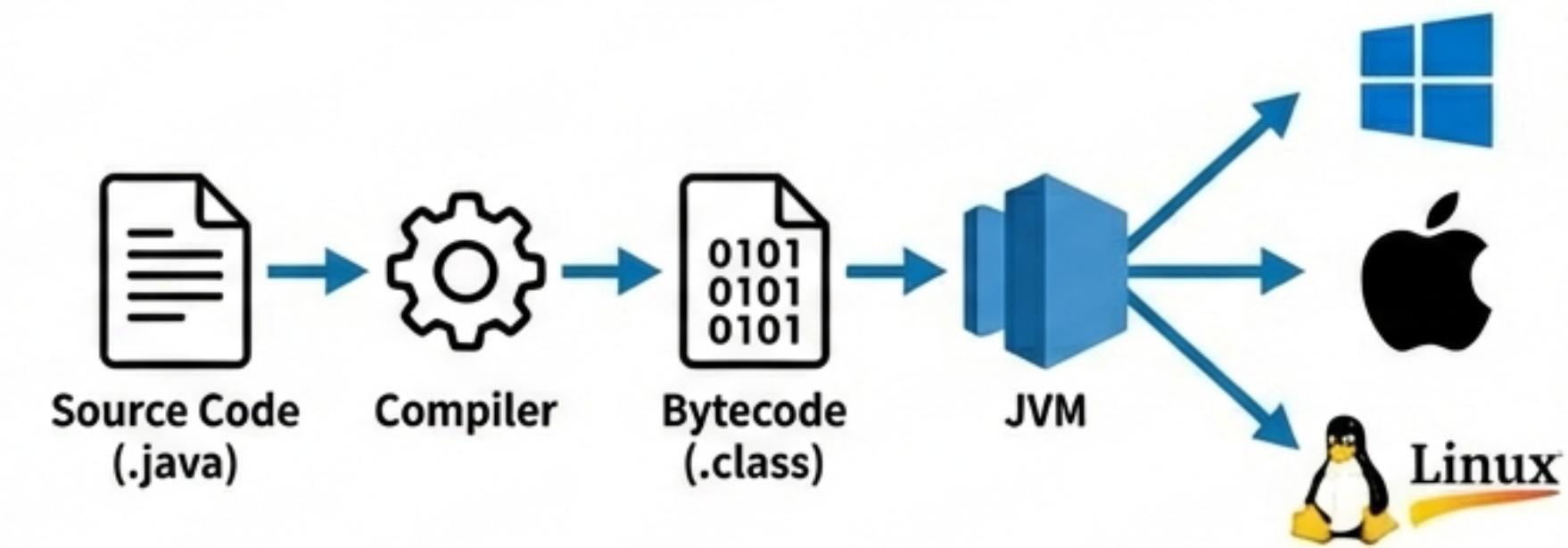
- ・ブラウザ上で動作するプログラミング言語
- ・DOM操作による画面更新
- ・クリック時の処理などを担当

```
button.addEventListener("click", ...)  
}
```



JavaとJVM（仕組み）

- コンパイル: 人間が読めるコードを機械語に変換
- JVM: OSごとの差異を吸収して実行
- 特徴: Write Once, Run Anywhere



Java基本構文1 (Hello World)

- **class**: プログラムの単位
- **main**メソッド: プログラムの実行開始地点
- **System.out.println**: 画面出力の命令

```
public class Hello {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
}
```

Entry Point
(実行開始地点)

ソースコードの基本ルール① (文字とコメント)

- ・半角英数を使用する: 全角（日本語など）は命令文ではNG
- ・可読性: 「誰が見ても分かりやすい」コードを書く
- ・コメント: 人間のための注釈。コンパイル時は無視される

コメント (Comments)

// 行コメント (Line Comment)

/* 範囲コメント (Block Comment) */

文字 (Characters)

OK Example

int price = 100;



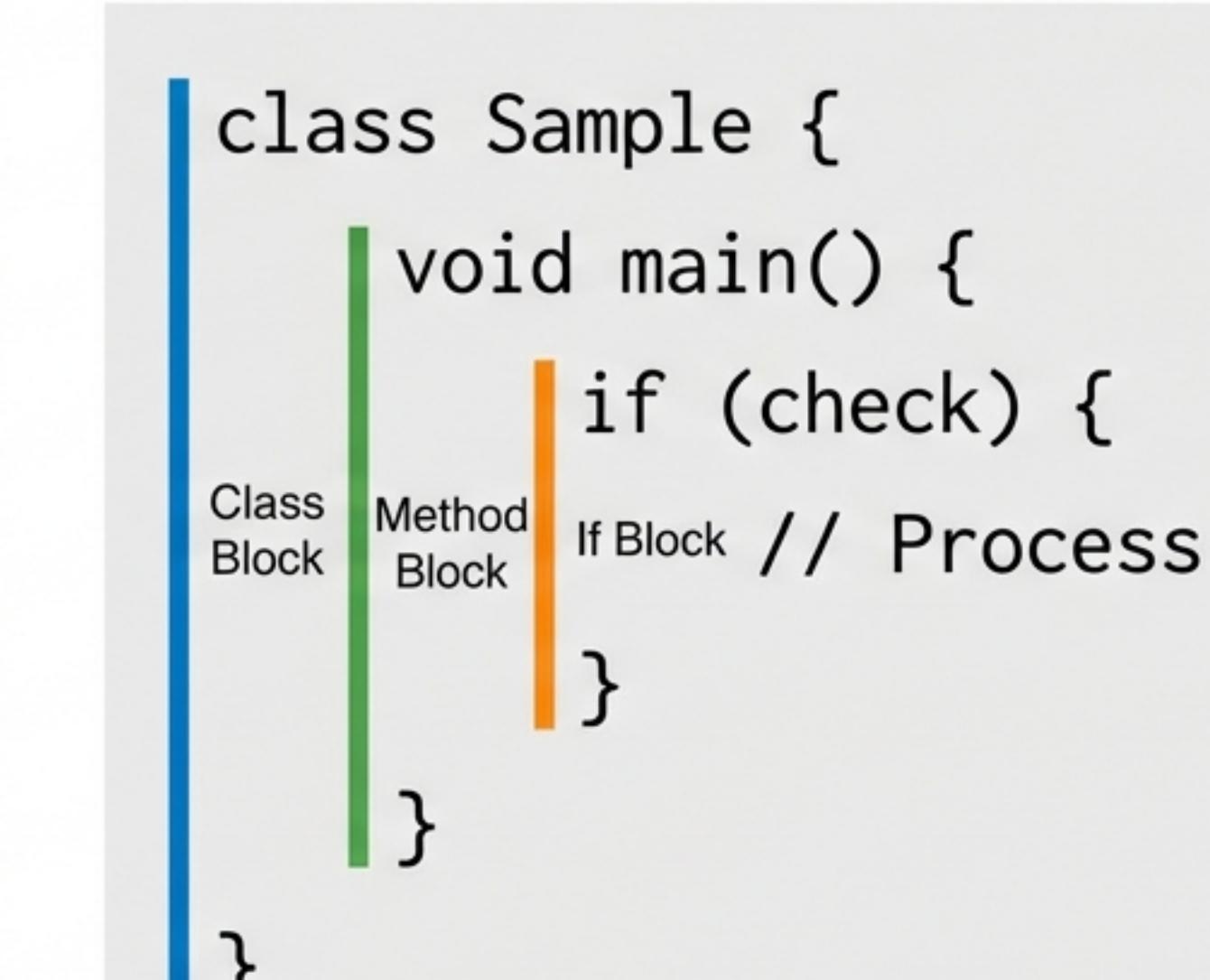
NG Example

i n t ~~p r i c e~~ e

ソースコードの基本ルール② (ブロックとインデント)

- ブロック {}: 処理の有効範囲
- ネスト: ブロックの中にブロックを入れる構造
- インデント: 字下げ (タブ/スペース4つ) で階層を視覚化

```
class Sample {  
    void main() {  
        if (check) {  
            // Process  
        }  
    }  
}
```



The diagram illustrates the nesting of code blocks. A blue vertical bar on the left represents the 'Class Block'. Inside it, a green vertical bar represents the 'Method Block'. Within the Method Block, an orange vertical bar represents the 'If Block'. The code itself is written in Java, showing a class named 'Sample' with a 'main' method containing an 'if' block that processes something if 'check' is true. The labels are placed to the left of their respective blocks: 'Class Block' is to the left of the outermost brace, 'Method Block' is to the left of the inner brace, and 'If Block' is to the left of the innermost brace.

ソースコードの基本ルール③ (命令文と区切り)

- 記述場所: 命令文はメソッドブロック内に書く
- セミコロン; 文末には必ずつける
- 順次進行: 上から順に1行ずつ実行される

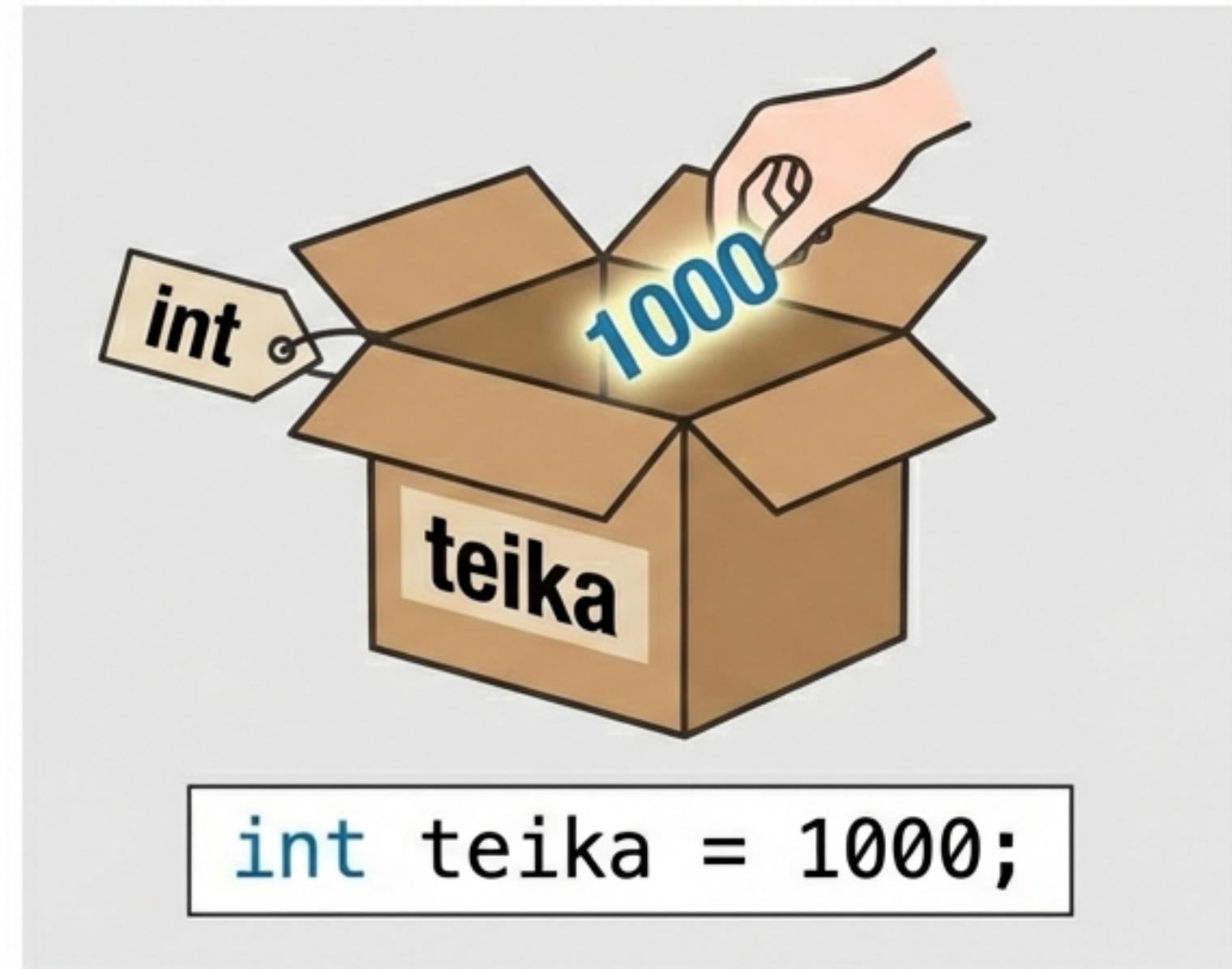
```
int a = 1;  
int b = 2;  
int c = a + b;
```

End of instruction
(文の区切り)

Execution
Flow
(順次進行)

Java基本構文2（変数とデータ型）

- 変数: データを一時的に保存する「箱」
- 宣言: 箱の名前と種類（型）を決める
- 代入: 箱に値を入れる
- データ型: int（整数）, String（文字列）など



Java基本構文3 (制御構文)

- **if文 (分岐)** : 条件によって処理を分ける
- **for文 (反復)** : 指定した回数だけ繰り返す
- ロジック構築に必須の要素



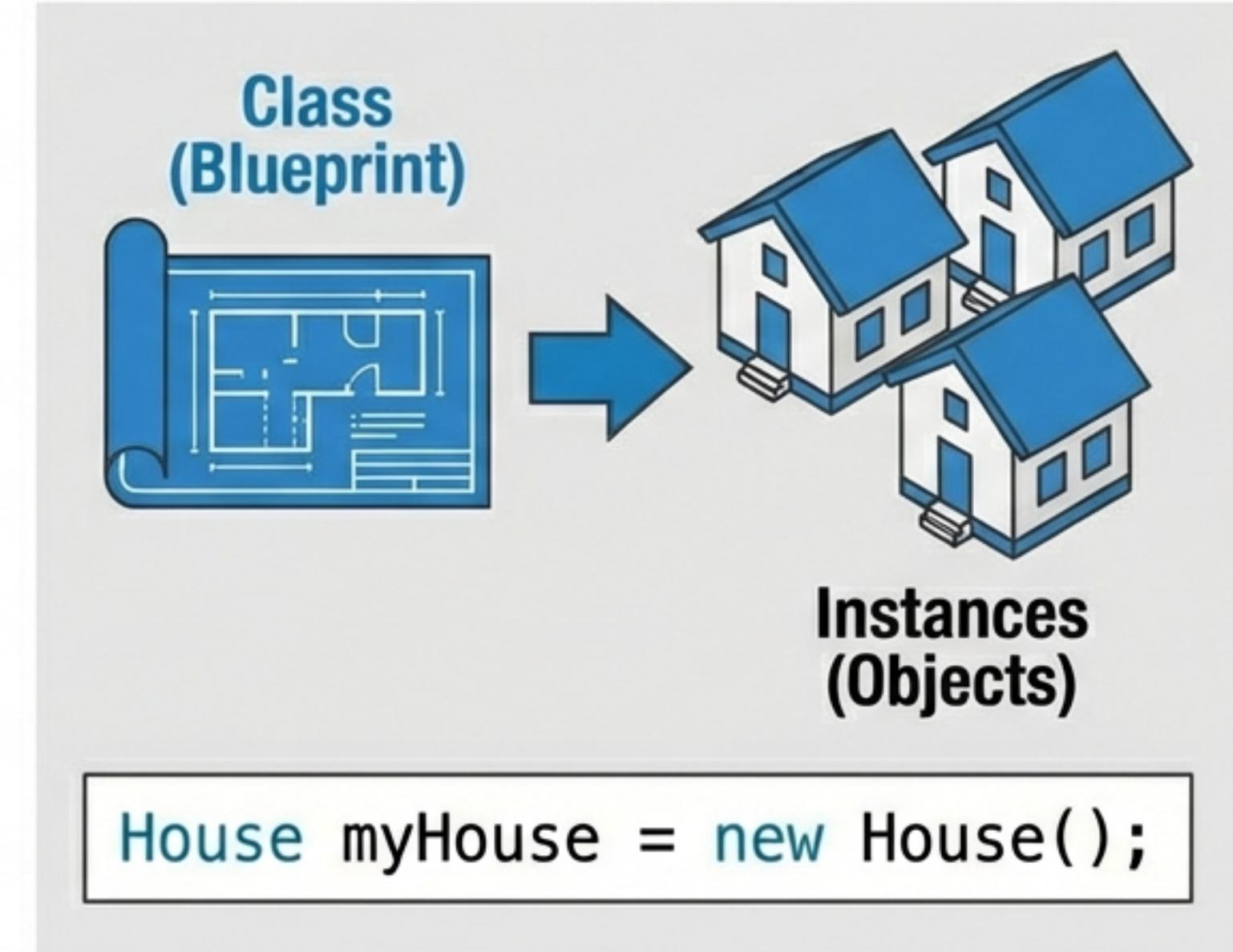
```
if (score > 80) {  
    ...  
}
```



```
for (int i=0; i<3; i++) {  
    ...  
}
```

Java基本構文 4 (クラスとオブジェクト)

- クラス (Class) : 設計図
- インスタンス (Instance) : 実体 (オブジェクト)
- new: 実体を作成するキーワード
- new: 実体を作成するキーワード



まとめ

- Webの役割：
HTML/CSS/JS/Javaの分担
- Javaの基礎：変数、制御、
クラス
- 基本ルール：「人間が読み
やすい」コードを書く



Messy Code = Buggy Code