JavaScript Asynchronous Programming — Beginner Essentials

This document gives the **clean, simple, foundation-level explanation** your students need **before** learning Promises and async/await.

It contains: - Synchronous JavaScript - Asynchronous JavaScript - Event Loop (simple) - Callbacks (with error-first pattern) - Clear Task → Explanation → Code → Output format

---

1. Synchronous JavaScript

Definition

JavaScript runs code **line by line**, and each line must finish before the next begins.

Example

```
console.log("A");
console.log("B");
console.log("C");
```

Output:

```
A
B
C
```

---

2. Asynchronous JavaScript

Definition

Some operations run in background. JavaScript does not wait for them.

Example

```
console.log("Start");
setTimeout(() => console.log("Inside Timeout"), 1000);
console.log("End");
```

Output:

```
Start
End
Inside Timeout
```

3. Event Loop — Simple Explanation

- JS has one main thread.
- Async tasks (timeouts, promises) run outside the main thread.
- When complete, callbacks/promises move into queues.
- Event loop moves them back into main thread when free.

4. Callback Functions

Definition

A function passed to another function to run later.

Basic Syntax

```
function doTask(callback) {
  callback();
}
```

Example 1 — Simple callback

Task: Print greeting then print message after callback.

```
function greet(name, callback) {
  console.log("Hello " + name);
  callback();
}

greet("John", () => console.log("Greet finished"));
```

Output:

```
Hello John
Greet finished
```

5. Error-First Callback Pattern

What is this pattern?

A standardized callback style where the **first parameter is error**, second is result.

```
callback(error, result)
```

Why it exists?

Callbacks do not have automatic error handling. This pattern allows controlled failure.

Example — Safe Division

Task: Divide two numbers. If denominator is zero, return error.

```
function divide(a, b, callback) {
  if (b === 0) callback("Cannot divide by zero", null);
  else callback(null, a / b);
}

divide(10, 0, (err, result) => {
  if (err) console.log("Error:", err);
  else console.log("Result:", result);
});
```

Output:

```
Error: Cannot divide by zero
```