

## জাভাস্ক্রিপ্ট মডিউল: ইম্পোর্ট (Import) এবং এক্সপোর্ট (Export)

আধুনিক জাভাস্ক্রিপ্ট ডেভেলপমেন্টে **মডিউল (Module)** একটি অত্যন্ত গুরুত্বপূর্ণ ধারণা। আগে সব জাভাস্ক্রিপ্ট কোড একটি ফাইলে লেখা হতো, যা বড় প্রজেক্টের ক্ষেত্রে রক্ষণাবেক্ষণ করা কঠিন হয়ে পড়ত। ES6 (ECMAScript 2015) আসার পর জাভাস্ক্রিপ্টে আনুষ্ঠানিকভাবে মডিউল সিস্টেম যুক্ত করা হয়।

সহজ কথায়, একটি মডিউল হলো একটি সাধারণ **জাভাস্ক্রিপ্ট ফাইল**। মডিউলের প্রধান কাজ হলো কোডগুলোকে ছোট ছোট ভাগে ভাগ করা এবং **প্রয়োজন অনুযায়ী** অন্য ফাইলে ব্যবহার করা। এই আদান-প্রদানের জন্য দুটি প্রধান কি-ওয়ার্ড ব্যবহার করা হয়: **export** এবং **import**.

### ১. এক্সপোর্ট (Export) কী?

**export** স্টেটমেন্টটি ব্যবহার করা হয় যখন আপনি একটি ফাইলের কোনো ভেরিয়েবল, ফাংশন বা ক্লাস অন্য ফাইলে ব্যবহার করতে চান।

জাভাস্ক্রিপ্টে মূলত দুই ধরনের এক্সপোর্ট রয়েছে:

1. **Named Export** (নামড এক্সপোর্ট)
2. **Default Export** (ডিফল্ট এক্সপোর্ট)

#### ক. Named Export (নামড এক্সপোর্ট)

যখন একটি মডিউল থেকে আপনি **একাধিক** ভ্যাবু বা ফাংশন এক্সপোর্ট করতে চান, তখন Named Export ব্যবহার করা হয়। ইম্পোর্ট করার সময় আপনাকে **অবশ্যই একই নাম** এবং **`{}` (curly braces)** ব্যবহার করতে হবে।

**উদাহরণ:** mathUtils.js

```
// ভেরিয়েবল এক্সপোর্ট
export const pi = 3.1416;

// ফাংশন এক্সপোর্ট
export function add(a, b) {
  return a + b;
}

// আলাদাভাবে এক্সপোর্ট করা
function subtract(a, b) {
  return a - b;
}

export { subtract };
```

#### খ. Default Export (ডিফল্ট এক্সপোর্ট)

যখন আপনি একটি ফাইল থেকে **শুধুমাত্র একটি** (যেমন একটি ক্লাস বা একটি ফাংশন) এক্সপোর্ট করতে চান, তখন Default Export ব্যবহার করা হয়। প্রতি ফাইলে **কেবল একটিই** default export থাকতে পারে।

**উদাহরণ:** User.js

```
export default class User {  
  constructor(name) {  
    this.name = name;  
  }  
  
  sayHello() {  
    console.log(`Hello, ${this.name}!`);  
  }  
}
```

## ২. ইম্পোর্ট (Import) কী?

import স্টেটমেন্টটি অন্য **মডিউল** বা **ফাইল** থেকে এক্সপোর্ট করা কোড **বর্তমান ফাইলে** নিয়ে আসতে ব্যবহৃত হয়।

### ক. Named Export ইম্পোর্ট করা

Named export ইম্পোর্ট করার সময় **{}** ব্যবহার করতে হয় এবং **নাম হুবহু** মিলতে হয়।

**উদাহরণ:** main.js

```
import { pi, add } from './mathUtils.js';  
  
console.log(pi); // 3.1416  
console.log(add(10, 5)); // 15
```

### খ. Default Export ইম্পোর্ট করা

Default export ইম্পোর্ট করার সময় **{}** বা কালি ব্রেসের **প্রয়োজন নেই**। আপনি যেকোনো নামে এটি ইম্পোর্ট করতে পারেন।

**উদাহরণ:** main.js

```
// User.js থেকে ইম্পোর্ট করা হচ্ছে, আমরা যেকোনো নাম দিতে পারি (যেমন: Person)  
import Person from './User.js';  
  
const user1 = new Person("Rahim");  
user1.sayHello(); // Output: Hello, Rahim!
```

## ৩. ইম্পোর্ট এবং এক্সপোর্টের বিভিন্ন কৌশল

### নাম পরিবর্তন করা (as কিওয়ার্ড)

কখনো কখনো দুটি ভিন্ন মডিউলে **একই নামের ফাংশন** থাকতে পারে বা আপনি নাম ছোট করতে চাইতে পারেন।

সেকেন্ডে as ব্যবহার করা হয়।

### Export এর সময় নাম পরিবর্তন:

```
export { add as sum }; // add ফাংশনটি sum নামে এক্সপোর্ট হবে
```

### Import এর সময় নাম পরিবর্তন:

```
import { add as jogKora } from './mathUtils.js';  
console.log(jogKora(2, 3));
```

### সবকিছু একসাথে ইম্পোর্ট করা (Namespace Import)

যদি একটি ফাইলে অনেকগুলো Named Export থাকে এবং আপনি সবগুলো একবারে ইম্পোর্ট করতে চান, তবে **\*** (asterisk) ব্যবহার করে একটি অবজেক্ট হিসেবে ইম্পোর্ট করতে পারেন।

```
import * as MathTools from './mathUtils.js';  
  
console.log(MathTools.pi);  
console.log(MathTools.add(5, 5));
```

### রি-এক্সপোর্ট (Re-export)

কখনো কখনো আমরা একটি ফাইলে কিছু ইম্পোর্ট করে সেটিকে আবার সাথে সাথেই এক্সপোর্ট করে দিতে চাই (সাধারণত index.js ফাইলে প্যাকেজ বানানোর সময়)।

```
export { add } from './mathUtils.js';  
export { default as User } from './User.js';
```

## 8. ব্রাউজারে মডিউল ব্যবহার (HTML)

ব্রাউজারে সরাসরি মডিউল ব্যবহার করতে হলে `<script>` ট্যাগে `type="module"` অ্যাট্রিবিউট যুক্ত করতে হয়।

```
<!DOCTYPE html>  
<html lang="en">  
<body>  
  <script type="module" src="main.js"></script>  
</body>  
</html>
```

### গুরুত্বপূর্ণ নোট:

1. **CORS Policy:** মডিউল লোকাল ফাইল সিস্টেম (file://) থেকে সরাসরি কাজ করে না। আপনাকে লাইভ সার্ভার (Live Server) বা লোকাল ওয়েব সার্ভার ব্যবহার করতে হবে।
2. **Strict Mode:** মডিউলগুলো স্বয়ংক্রিয়ভাবে "use strict" মোডে রান করে।
3. **Deferred:** মডিউল স্ক্রিপ্টগুলো ডিফল্টভাবে defer আচরণ করে (অর্থাৎ HTML পার্স হওয়ার পর রান করে)।

## ৫. ডায়নামিক ইম্পোর্ট (Dynamic Import)

সাধারণ import ফাইলের শুরুতে থাকে এবং স্ট্যাটিক হয়। কিন্তু কখনো কখনো আমাদের শর্তসাপেক্ষে বা ফাংশনের ভেতরে মডিউল লোড করার প্রয়োজন হয় (পারফরমেন্স অপটিমাইজেশনের জন্য)। একে ডায়নামিক ইম্পোর্ট বলে। এটি import() ফাংশন ব্যবহার করে যা একটি প্রমিস (Promise) রিটার্ন করে।

```
// উদাহরণ: বাটন ক্লিক করলে মডিউল লোড হবে
button.addEventListener('click', async () => {
  try {
    const module = await import('./mathUtils.js');
    console.log(module.add(10, 20));
  } catch (error) {
    console.error("মডিউল লোড করা যায়নি", error);
  }
});
```

### সারসংক্ষেপ (Summary)

বিষয়	বিবরণ	সিনটাক্স উদাহরণ
<b>Named Export</b>	একাধিক ভ্যালু এক্সপোর্ট করতে।	export const x = 1;
<b>Default Export</b>	ফাইলে একটি প্রধান ভ্যালু এক্সপোর্ট করতে।	export default class ...
<b>Named Import</b>	নির্দিষ্ট নাম ধরে ইম্পোর্ট করতে।	import { x } from './file.js';
<b>Default Import</b>	যেকোনো নামে ইম্পোর্ট করতে (ব্রেস ছাড়া)।	import X from './file.js';
<b>Import All</b>	সব কিছু একটি অবজেক্টে আনতে।	import * as Obj from './file.js';

মডিউল ব্যবহারের মাধ্যমে আপনার জাভাস্ক্রিপ্ট কোড হবে আরও গোছানো, রিইউজেবল এবং বাগ-মুক্ত।

### রেফারেন্স:

- [MDN Web Docs: export](#)
- [JavaScript.info: Import and Export](#)