জাভাস্ক্রিপ্টে **let** এবং **const** হলো দুটি কিওয়ার্ড যা ভেরিয়েবল ডিক্লেয়ার করার জন্য ব্যবহৃত হয়। এগুলি ES6 (ECMAScript 2015) এ প্রবর্তিত হয় এবং var এর চেয়ে বেশি মডার্ন এবং নিরাপদ। নিচে let এবং const এর মধ্যে পার্থক্য এবং ব্যবহার সম্পর্কে বিস্তারিত ব্যাখ্যা দেওয়া হলো:

**১. let:**

let কিওয়ার্ড ব্যবহার করে আপনি একটি **পরিবর্তনযোগ্য (mutable)** ভেরিয়েবল ডিক্লেয়ার করতে পারেন। এটি ব্লক-স্কোপড, অর্থাৎ এটি শুধুমাত্র যে ব্লকে ডিক্লেয়ার করা হয়েছে সেই ব্লকের মধ্যে অ্যাক্সেস করা যায়।

**বৈশিষ্ট্য:**

1. **পরিবর্তনযোগ্য**: let ভেরিয়েবলের মান পরিবর্তন করা যায়।
2. **ব্লক-স্কোপড**: এটি শুধুমাত্র যে ব্লকে ডিক্লেয়ার করা হয়েছে সেই ব্লকের মধ্যে অ্যাক্সেস করা যায়।
3. **হোইস্টিং**: let ভেরিয়েবল হোইস্ট হয়, কিন্তু এটি টেম্পোরাল ডেড জোন (TDZ) এ থাকে যতক্ষণ না ডিক্লেয়ারেশন লাইনে পৌঁছানো হয়।

**সিনট্যাক্স:**

javascript

Copy

let variableName = value;

**উদাহরণ:**

javascript

Copy

let name = "John";

name = "Alice"; // ভেরিয়েবলের মান পরিবর্তন করা যায়

console.log(name); // আউটপুট: Alice

if (true) {

let age = 25; // ব্লক-স্কোপড

console.log(age); // আউটপুট: 25

}

// console.log(age); // Error: age is not defined (ব্লকের বাইরে অ্যাক্সেস করা যায় না)

**২. const:**

const কিওয়ার্ড ব্যবহার করে আপনি একটি **অপরিবর্তনীয় (immutable)** ভেরিয়েবল ডিক্লেয়ার করতে পারেন। এটি ব্লক-স্কোপড এবং একবার মান অ্যাসাইন করার পর তা পরিবর্তন করা যায় না।

**বৈশিষ্ট্য:**

1. **অপরিবর্তনীয়**: const ভেরিয়েবলের মান পরিবর্তন করা যায় না।
2. **ব্লক-স্কোপড**: এটি শুধুমাত্র যে ব্লকে ডিক্লেয়ার করা হয়েছে সেই ব্লকের মধ্যে অ্যাক্সেস করা যায়।
3. **হোইস্টিং**: const ভেরিয়েবল হোইস্ট হয়, কিন্তু এটি টেম্পোরাল ডেড জোন (TDZ) এ থাকে যতক্ষণ না ডিক্লেয়ারেশন লাইনে পৌঁছানো হয়।

**সিনট্যাক্স:**

javascript

Copy

const constantName = value;

**উদাহরণ:**

javascript

Copy

const age = 25;

// age = 30; // Error: ভেরিয়েবলের মান পরিবর্তন করা যায় না

console.log(age); // আউটপুট: 25

if (true) {

const city = "Dhaka"; // ব্লক-স্কোপড

console.log(city); // আউটপুট: Dhaka

}

// console.log(city); // Error: city is not defined (ব্লকের বাইরে অ্যাক্সেস করা যায় না)

**let এবং const এর মধ্যে পার্থক্য:**

| **বৈশিষ্ট্য** | **let** | **const** |
| --- | --- | --- |
| **পরিবর্তনযোগ্যতা** | পরিবর্তনযোগ্য | অপরিবর্তনীয় |
| **স্কোপ** | ব্লক-স্কোপড | ব্লক-স্কোপড |
| **হোইস্টিং** | হোইস্টেড (TDZ) | হোইস্টেড (TDZ) |
| **ইউজ কেস** | পরিবর্তনযোগ্য ভেরিয়েবল | অপরিবর্তনীয় ভেরিয়েবল |

**কোনটি কখন ব্যবহার করবেন?:**

* **let ব্যবহার করবেন**:
  + যখন ভেরিয়েবলের মান পরিবর্তন করার প্রয়োজন হয়।
  + উদাহরণ: লুপের মধ্যে ভেরিয়েবল, কন্ডিশনাল স্টেটমেন্ট।
* **const ব্যবহার করবেন**:
  + যখন ভেরিয়েবলের মান পরিবর্তন করার প্রয়োজন না হয়।
  + উদাহরণ: কনস্ট্যান্ট ভ্যালু, অবজেক্ট বা অ্যারের রেফারেন্স।

**উদাহরণ:**

javascript

Copy

let count = 0;

count = 1; // মান পরিবর্তন করা যায়

console.log(count); // আউটপুট: 1

const PI = 3.1416;

// PI = 3.14; // Error: মান পরিবর্তন করা যায় না

console.log(PI); // আউটপুট: 3.1416

const person = { name: "John", age: 25 };

person.age = 30; // অবজেক্টের প্রপার্টি পরিবর্তন করা যায়

console.log(person); // আউটপুট: { name: "John", age: 30 }

**সারসংক্ষেপ:**

* let এবং const উভয়ই ব্লক-স্কোপড এবং হোইস্টেড (TDZ সহ)।
* let ব্যবহার করা হয় যখন ভেরিয়েবলের মান পরিবর্তন করার প্রয়োজন হয়।
* const ব্যবহার করা হয় যখন ভেরিয়েবলের মান পরিবর্তন করার প্রয়োজন না হয়।
* const অবজেক্ট বা অ্যারের ক্ষেত্রে রেফারেন্স অপরিবর্তনীয় রাখে, তবে অবজেক্টের প্রপার্টি বা অ্যারের উপাদান পরিবর্তন করা যায়।

এই পার্থক্যগুলি বুঝলে আপনি আপনার প্রয়োজনের ভিত্তিতে সঠিক কিওয়ার্ড বেছে নিতে পারবেন। 😊