

# unknown vs any

In TypeScript, both `unknown` and `any` are special types that can represent any value, but they differ significantly in their approach to type safety.

## `any` Type:

The `any` type essentially disables type checking for the variable it's assigned to. You can assign any value to an `any` type variable, and you can perform any operation on it without TypeScript raising an error, even if the operation is invalid for the actual runtime type of the value. This provides maximum flexibility but sacrifices type safety, potentially leading to runtime errors that TypeScript was designed to prevent. `any` is often used when integrating with untyped JavaScript libraries or when dealing with highly dynamic data structures where strict typing is impractical.

## `unknown` Type:

The `unknown` type is a type-safe counterpart to `any`. Like `any`, you can assign any value to an `unknown` type variable. However, unlike `any`, TypeScript enforces type checking when you try to perform operations on an `unknown` variable. Before you can use a value of type `unknown` in an operation (e.g., accessing a property, calling a method), you must first narrow its type using type guards (like `typeof` checks or `instanceof`). This ensures that you explicitly handle the possible types of the value, promoting safer and more predictable code.

## Key Differences Summarized:

| Feature       | <code>any</code>                           | <code>unknown</code>                                     |
|---------------|--------------------------------------------|----------------------------------------------------------|
| Type Checking | Bypasses all type checking.                | Enforces type checking before operations.                |
| Operations    | Allows any operation without prior checks. | Requires type narrowing (type guards) before operations. |
| Safety        | Less type-safe, prone to runtime errors.   | More type-safe, helps prevent runtime errors.            |

|       |                                                                 |                                                      |
|-------|-----------------------------------------------------------------|------------------------------------------------------|
| Usage | For maximum flexibility, or when strict typing is not feasible. | For type-safe handling of values with unknown types. |
|-------|-----------------------------------------------------------------|------------------------------------------------------|

**In essence:**

Use `any` when you need to quickly bypass TypeScript's type system and accept the associated risks.

Prefer `unknown` when you want to handle values of an uncertain type in a type-safe manner, ensuring that you explicitly check and handle their potential types before using them.