**What is a Data Type in Java?**

**Definition**:  
A **data type** specifies the type of data that a variable can hold. In Java, every variable must be declared with a data type before it is used.

**Why?**  
To let the compiler know:

* What kind of data to expect (like numbers, characters, true/false).
* How much memory to allocate.
* What operations can be done on that variable.

**🔸 Types of Data Types in Java**

Java categorizes its data types into **two main categories**:

Data Types

|

-----------------

| |

Primitive Non-Primitive

(Built-in) (Reference)

**1️⃣ Primitive Data Types – (8 Types)**

These are built-in, low-level types that are not objects.

| **Data Type** | **Size** | **Default** | **Example** | **Description** |
| --- | --- | --- | --- | --- |
| byte | 1 byte (8 bits) | 0 | byte age = 25; | Small integers (−128 to 127) |
| short | 2 bytes | 0 | short temp = 30000; | Larger than byte |
| int | 4 bytes | 0 | int salary = 50000; | Most commonly used integer |
| long | 8 bytes | 0L | long population = 8000000000L; | Very large numbers |
| float | 4 bytes | 0.0f | float pi = 3.14f; | Decimal values (less precision) |
| double | 8 bytes | 0.0d | double gpa = 8.55; | Decimal values (more precision) |
| char | 2 bytes | '\u0000' | char grade = 'A'; | A single character |
| boolean | 1 bit | false | boolean isAlive = true; | True or False |

**✅ Real Life Analogies:**

| **Data Type** | **Real-World Analogy** |
| --- | --- |
| int | Number of apples |
| float | Price of milk: 45.75 rupees |
| char | A single key pressed on keyboard |
| boolean | Light switch: ON or OFF |
| byte | Age of a small child (0–127) |
| long | Distance from Earth to Sun |
| double | Scientific measurements like mass of Earth |
| short | Temperature in a city (–32,000 to 32,000) |

**🔍 Details of Each Primitive Type**

**🟤 byte**

* Range: –128 to 127
* Use when memory saving matters (e.g., image pixel values)

**🟠 short**

* Range: –32,768 to 32,767
* Use when int is too much

**🔵 int**

* Most used
* Ideal for counting, indexing, loops, etc.

**🔴 long**

* Large integers: phone numbers, bank account numbers
* Always add **L**: long a = 12345678900L;

**🟢 float**

* Less precise decimals (6–7 digits)
* Add **f** at the end: 3.14f

**🟣 double**

* More precise decimals (15 digits)
* Default for decimal values
* Use in calculations: double area = 3.14159 \* r \* r;

**🟤 char**

* Holds a single Unicode character
* char c = 'A';

**⚫ boolean**

* True/False
* Used for decisions, conditions

**2️⃣ Non-Primitive Data Types (Reference Types)**

These store the **address/reference** of the value in memory.

| **Type** | **Example** | **Description** |
| --- | --- | --- |
| String | "Hello" | Sequence of characters |
| Arrays | int[] arr = {1, 2, 3}; | Collection of elements |
| Classes | Student s = new Student(); | User-defined data structures |
| Interfaces | Runnable, Comparable, etc. | Contracts to define behavior |
| Enums | enum Color { RED, GREEN }; | Set of constants |

**✅ Difference Between Primitive and Non-Primitive**

| **Feature** | **Primitive** | **Non-Primitive** |
| --- | --- | --- |
| Memory | Stores value | Stores reference |
| Built-in | Yes | No (user-defined or library) |
| Null | Can't be null | Can be null |
| Methods | No methods | Have methods |
| Examples | int, char | String, Array, Class |

**🧠 Memory Chart Summary**

| **Data Type** | **Memory** | **Range** |
| --- | --- | --- |
| byte | 1 byte | –128 to 127 |
| short | 2 bytes | –32,768 to 32,767 |
| int | 4 bytes | –2B to 2B |
| long | 8 bytes | Very large |
| float | 4 bytes | ~6-7 decimals |
| double | 8 bytes | ~15 decimals |
| char | 2 bytes | Unicode characters |
| boolean | ~1 bit | true/false |

**📚 Sample Java Code**

public class DataTypeDemo {

public static void main(String[] args) {

int age = 25;

float price = 99.99f;

double weight = 65.8;

char grade = 'A';

boolean isPassed = true;

System.out.println("Age: " + age);

System.out.println("Price: " + price);

System.out.println("Weight: " + weight);

System.out.println("Grade: " + grade);

System.out.println("Passed? " + isPassed);

}

}

**✅ Summary**

* Java has 8 primitive data types.
* Non-primitives (like String, Arrays, Classes) are reference types.
* Every variable **must** have a declared type.
* Use the **right type** to optimize memory and performance.
* Understanding data types is **fundamental** for interviews, coding, and Java programming.

**✅ When NOT to use int in Java:**

**1️⃣ When the number is too big**

If the number is **larger than 2,147,483,647** (which is the max for int), then:

// ❌ This will overflow

int population = 8000000000; // Too big!

// ✅ Use long

long population = 8000000000L;

**2️⃣ When you need decimal/fraction values**

int only stores **whole numbers**. If you need **decimal values**, use float or double.

// ❌ Wrong: decimal gets cut off

int price = 99.99; // Error

// ✅ Correct

float price = 99.99f;

double weight = 67.54;

**3️⃣ When memory usage matters (e.g., embedded systems)**

If you're handling **thousands of small values**, like age of kids in a school, use byte or short to save memory.

// ✅ Save memory with byte

byte age = 12;

**4️⃣ When you need characters**

If you're dealing with **letters or symbols**, use char, not int.

char grade = 'A'; // ✅ Correct

**5️⃣ When you're storing true/false**

For logical yes/no, use boolean.

boolean isEligible = true; // ✅ Not int

**6️⃣ When you're working with objects or collections**

Java’s **Collections (like ArrayList)** don’t allow int. You must use **Integer (wrapper class)**.

ArrayList<Integer> list = new ArrayList<>();

**7️⃣ When the number is used in scientific precision**

If precision is important (like physics, chemistry, finance):

* Avoid int
* Use double

double pi = 3.1415926535; // ✅ High precision

**8️⃣ When working with strings or text**

Don’t use int to store names, addresses, etc.

// ❌ int name = "John";

String name = "John"; // ✅ Use String

**🔁 Summary Table**

| **Situation** | **Don’t Use** | **Use Instead** |
| --- | --- | --- |
| Large numbers | int | long |
| Decimal values | int | float/double |
| Characters | int | char |
| True/False | int | boolean |
| Collections | int | Integer |
| High memory efficiency | int | byte/short |
| Text values | int | String |

**✅ Interview Tip:**

**"How do you choose a data type in Java?"**

I start by asking:

* Do I need decimals? → Use float/double
* Is the number very large? → Use long
* Is it a single character? → Use char
* Is it true/false? → Use boolean
* Is it text? → Use String
* Is it a small whole number and memory is a concern? → Use byte or short  
  Otherwise, I default to int for general whole numbers.

**✅ What is a Variable in Java?**

A **variable** is like a **named container** in memory that stores a value, which can be used and changed during the program.

**➕ Think of it like:**

📦 *A labelled box where you store something temporarily.*

* You can open the box (read its value)
* You can replace what's inside (change the value)

**🔹 Variable Declaration Syntax**

datatype variableName = value;

Example:

int age = 25;

* int = data type
* age = variable name
* 25 = value stored in it

**🔸 Variables with Primitive Data Types**

Let’s go one by one with **examples**:

**1️⃣ int — Whole numbers**

int students = 50;

System.out.println("Students: " + students); // 50

**2️⃣ float — Decimal with less precision**

float price = 99.99f;

System.out.println("Price: ₹" + price); // ₹99.99

**3️⃣ double — Decimal with more precision**

double gpa = 9.35;

System.out.println("GPA: " + gpa); // 9.35

**4️⃣ char — Single character**

char grade = 'A';

System.out.println("Grade: " + grade); // A

**5️⃣ boolean — True or False**

boolean isPassed = true;

System.out.println("Passed: " + isPassed); // true

**6️⃣ long — Large numbers**

long population = 8000000000L;

System.out.println("Population: " + population); // 8000000000

**7️⃣ byte — Small whole numbers**

byte age = 12;

System.out.println("Child's age: " + age); // 12

**8️⃣ short — Medium range numbers**

short temperature = 24000;

System.out.println("Temperature: " + temperature); // 24000

**🔹 Variables with Non-Primitive Data Types**

**🔹 String — Sequence of characters**

String name = "Bhargavi";

System.out.println("Name: " + name); // Bhargavi

**🔹 int[] — Array**

int[] marks = {90, 80, 85};

System.out.println("First mark: " + marks[0]); // 90

**🔁 Changing Variable Values**

You can update a variable's value:

int age = 18;

age = 21; // updated

System.out.println("Age now: " + age); // 21

**🔒 Rules for Naming Variables**

✅ Valid:

int age;

float totalAmount;

String firstName;

❌ Invalid:

int 2marks; // cannot start with digit

int class; // 'class' is a reserved keyword

**🧠 Memory Tip:**

* int age = 25;  
  → A 4-byte memory box is labelled age, and it stores the number 25.

**✅ Summary Table: Variable Examples**

| **Data Type** | **Variable Example** | **Meaning** |
| --- | --- | --- |
| int | int score = 95; | Whole number |
| float | float price = 49.5f; | Decimal value |
| double | double pi = 3.14159; | High precision |
| char | char symbol = '#'; | Single character |
| boolean | boolean isOn = false; | True/False |
| long | long distance = 15000000000L; | Large number |
| String | String name = "Siri"; | Text |