1. What Is a Variable?

A **variable** is like a labeled container that stores data in memory so your program can use it later.

👉 Think of it as a box:

- The **label** = variable name
- The **content** = stored value

Example:

```
String name = "Alex";
```

Here:

- `String` → type of data
- `name` → variable label
- `"Alex"` → stored value

---

2. Java Data Types

Java is a **strongly typed** language. This means you *must* tell Java exactly what type of data a variable will hold before you use it, and you cannot change that type later.

Start by teaching just these five essential types:

Category	Data Type	What it stores	Example
Number	`int`	Whole numbers (integers).	`10`, `-5`, `0`
Number	`double`	Numbers with decimals.	`3.14`, `-99.9`
Character	`char`	A single letter, number, or symbol. *Always uses single quotes.*	`'A'`, `'7'`, `'?'`
Logic	`boolean`	Only two possible values: true or false.	`true`, `false`
Text	`String`	A sequence of characters. *Always uses double quotes. Note the capital 'S'!*	`"Hello"`, `"Java"`

> **Teacher's Note:** It is crucial to emphasize that `String` starts with a capital 'S' while the others are lowercase. This introduces the idea that `String` is a slightly different breed (a Reference type) than the basic Primitives, without needing to overcomplicate the explanation early on.

---

3. Syntax: Declaration vs. Initialization

Students need to learn how to actually write this in code. Teach this as a two-step process that can be combined into one.

Step 1: Declaration (Building the empty box) You tell Java the *type* and the *name*.

```
int playerScore;
String playerName;
```

Step 2: Initialization (Putting the first value in the box) You use the equals sign (`=`) to assign a value. Read `=` as "gets the value of" rather than "equals".

```
playerScore = 100;
playerName = "Alex";
```

The Standard Way (Combined) Show them that programmers usually do both on the exact same line to save time:

```
int playerScore = 100;
String playerName = "Alex";
boolean isGameOver = false;
```

---

4. Naming Rules and Best Practices

Finally, cover the rules for the "labels" (variable names). If they learn good habits now, they will avoid frustrating errors later.

- **Rule 1:** No spaces allowed. (`player score` will crash the program).
- **Rule 2:** Cannot start with a number. (`1stPlayer` is bad; `player1` is good).
- **Rule 3 (The Golden Convention):** Use **camelCase**. Start with a lowercase letter, and capitalize the first letter of every subsequent word.
- *Example:* `myFavoriteColor`, `accountBalance`, `isRainyDay`.

The "Greet Me" Program

```
public class HelloName {
    public static void main(String[] args) {

        // Variable storing text
        String name = "Alex";

        // Print message using variable
        System.out.println("Hello, " + name + "!");
    }
}
```

Code Breakdown

1️⃣ Variable Creation

```
String name = "Alex";
```

This line:

- Creates a variable called `name`
- Stores the text `"Alex"` inside it

2️⃣ Concatenation (Joining Text)

```
"Hello, " + name + "!"
```

The `+` operator joins text and variables together.

Result printed:

```
Hello, Alex!
```

3️⃣ Output Statement

```
System.out.println(...);
```

Displays text on the screen.

📌 Why types matter: They tell Java how much memory to use and what operations are allowed.

Rules for Naming Variables

✔ Allowed

```
name
studentAge
totalMarks
```

❌ Not allowed

```
1name
student age
class
```

Tips

- Must start with a letter, `_`, or `$`
- Cannot use spaces
- Cannot use Java keywords
- Use camelCase → `favoriteNumber`

Tasks

Task 1: The Detective (Spot the Errors)

The following Java code is broken. There are **five** lines of code, and each one has exactly **one** mistake. Can you find and fix them?

```
1. int player Score = 100;
2. string heroName = "Zelda";
3. boolean hasMagicKey = "true";
4. char letterGrade = 'A+';
5. double 1stLapTime = 45.2;
```

Task 2: The Architect (Choose the Right Box)

Look at the real-world information below. Which Java data type (`int`, `double`, `boolean`, `char`, or `String`) would be the **best** choice to store each piece of info?

1. The price of a cup of coffee.
2. The number of lives a player has left in a video game.
3. A student's middle initial.
4. Whether or not a door is currently locked.
5. A user's email address.

Task 3: The Builder (Write the Code)

Write the exact Java code to create the following variables. Remember to combine declaration and initialization, and use proper **camelCase** naming conventions!

1. Create a variable for a dog's name and set it to `Buddy`.
2. Create a variable for the dog's age and set it to `5`.

3. Create a variable to track if the dog is hungry and set it to `true`.
4. Create a variable for the dog's weight in pounds and set it to `45.5`.

Related Topics

- Standard Input
- Java Expressions: Types, Examples & Practice Questions | Complete Guide