Difference between an object reference and a java object

**1. Java Object**

* An **object** is a real instance of a class created in memory at runtime using the new keyword.
* It **contains actual data (state/fields) and behavior (methods)** defined in its class.
* Stored in the **heap memory**.

👉 Example:

class Student {

String name;

int age;

}

Student s1 = new Student(); // creates a new object

Here, new Student() creates an **object** of type Student in the heap.

**🔹 2. Object Reference**

* A **reference** is a variable that **stores the memory address (or reference) of an object**.
* The reference does not contain the object itself, but rather a "pointer" to it.
* Stored in the **stack memory**.

👉 Example:

Student s1 = new Student();

* s1 is an **object reference**.
* It holds the memory address that points to the actual object in the heap.

**🔹 3. Key Differences**

| **Feature** | **Object (Java Object)** | **Object Reference** |
| --- | --- | --- |
| **Meaning** | Instance of a class with actual data & methods. | Variable that points to an object. |
| **Where stored** | Heap memory | Stack memory (holds address of object in heap). |
| **What it contains** | Fields + methods (state + behavior). | Address (or "handle") to the object. |
| **Created by** | new keyword (or via frameworks, reflection). | Declaring a variable of class type. |
| **Example** | new Student() | Student s1 |

**🔹 4. Analogy**

Think of a **TV remote and a TV set**:

* The **TV set** = the **object** (it does the actual work).
* The **remote control** = the **reference** (you use it to access/control the object).

Without the TV, the remote is useless; without the remote, the TV exists but you can’t control it.

✅ So in short:

* **Object** = actual thing in memory (data + behavior).
* **Reference** = variable that refers to (points to) the object.