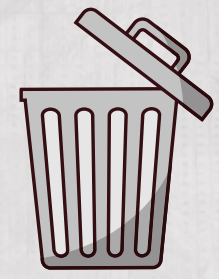
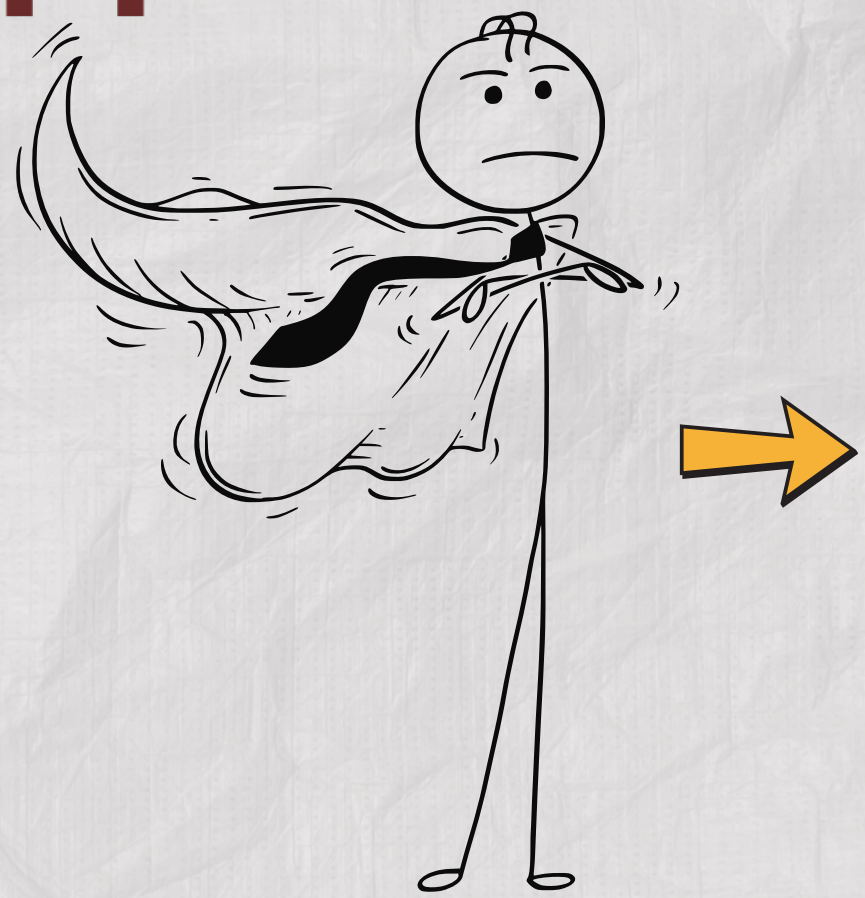
@HamzaELHANAFY



Java Garbage Collection: The Hidden Hero Behind Your Apps



A simple question

Have you ever found yourself wondering how much memory you need to manage in your Java app

If your answer is “**No**”, you’re not alone …

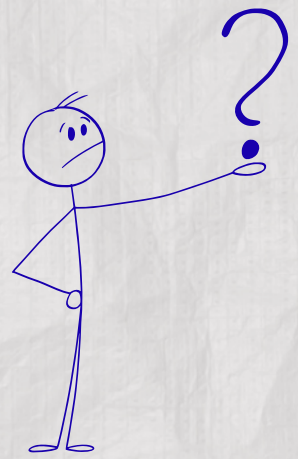
most Java developers don’t worry about it, and that’s exactly the point!

@HamzaELHANAFY

Java has a hidden hero called the **Garbage Collector (GC)** 🧹 that automatically cleans up memory you no longer use.

Ok, but

What is Garbage Collection?



In a simple way

GC *It's the automatic process in Java that removes objects no longer referenced by your code.*

➡️ *You focus on writing your app — Java handles memory!* ✅

How It Works

Let's break down this very simple example.

```
public class Demo {
    public static void main(String[] args) {
        String name = new String("Hamza");
        name = null; // 🗑️ Now eligible for GC
        System.gc(); // Hint to run GC (optional)
    }
}
```

Soo, After **name = null**, the object "Hamza" is no longer used → GC will clean it up. 🧹

Other Case Studies: When Does Java Delete Your Objects? 🧹

Here are other quick scenarios where your objects become eligible for Garbage Collection 👇

1. Local Variables Out of Scope ⏳

So, when a method ends, all its local objects disappear from memory.

```
public void example() {
    String temp = new String("Temp");
} // After the method ends, temp is gone → GC can delete it!
```

2. Reassigning an Object 🔄

If you assign a new object to a variable, the old object becomes unreachable.

Donc man9drosh nwslo lih, hit mab9ax 3ndo htta reference

```
String str = new String("Old");
str = new String("New");
// "Old" is no longer used (because it loses its reference)→ GC will delete it!
```

Why It Matters

- *Keeps your app fast ⚡*
- *Prevents OutOfMemoryError ❌*
- *Lets you focus on coding, not memory management 👨‍💻*

That's a quick look at Java Garbage Collection!

Have you ever needed to manage memory manually? Let us know in the comments 👇