

# AutoCloseable vs Closeable



Tell me about autoclosable interface and aslo compare autoclosable with closable

Good question 👍 Let's break it down step by step.

## 1. AutoCloseable Interface

- Introduced in **Java 7**.
- Package: `java.lang`
- Purpose: Allows an object to be used with the **try-with-resources** statement, so that resources (like files, sockets, database connections) are closed automatically after use.
- **Method:**

```
java  
  
void close() throws Exception;
```

- Any class that implements `AutoCloseable` can be used inside a **try-with-resources** block.

### Example:

```
java  
  
class MyResource implements AutoCloseable {  
    @Override  
    public void close() throws Exception {  
        System.out.println("Resource closed!");  
    }  
}  
  
public class Main {  
    public static void main(String[] args) {  
        try (MyResource res = new MyResource()) {  
            System.out.println("Using resource...");  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
}
```

### ✓ Output:

```
sql  
  
Using resource...  
Resource closed!
```

## 2. Closeable Interface

- Introduced earlier in **Java 5**.
- Package: `java.io`
- Designed specifically for **I/O classes** (e.g., `InputStream`, `OutputStream`, `Reader`, `Writer`).
- **Method:**

```
java

void close() throws IOException;
```

- Used mainly for **stream-based resources**.



## 3. Key Differences (AutoCloseable vs Closeable)

Feature	AutoCloseable	Closeable
Package	<code>java.lang</code>	<code>java.io</code>
Introduced	Java 7	Java 5
Method Signature	<code>void close() throws Exception;</code>	<code>void close() throws IOException;</code>
Exception type	Can throw any <b>Exception</b> (broader)	Can only throw <b>IOException</b>
Usage Scope	General resources (DB, sockets, custom)	Specifically I/O streams/files
Try-with-resources	Fully supported	Fully supported
Preferred Today	<b>AutoCloseable</b> (more generic, flexible)	Still used for I/O classes (legacy + compatibility)

## 4. Which One to Use?

- If you're writing a **custom resource class** (like a DB connection handler or network socket manager), implement **AutoCloseable** because it's more generic.
- If you're writing something specifically related to **I/O streams**, implement **Closeable**.

### Rule of Thumb:

-  **Closeable** is a specialized, I/O-focused version of **AutoCloseable**.
-  All **Closeable** classes are also **AutoCloseable** (because **Closeable** extends **AutoCloseable**).

Do you want me to also **show you what happens if a class implements both** and how Java decides which `close()` is called?