7 Things I Learned About Java 11

Index

Section 1	HTTP Client API
Section 2	String Methods
Section 3	File Methods
Section 4	Optional Enhancements
Section 5	Garbage Collector
Section 6	Single File Programs
Section 7	Deprecation

HTTP Client API

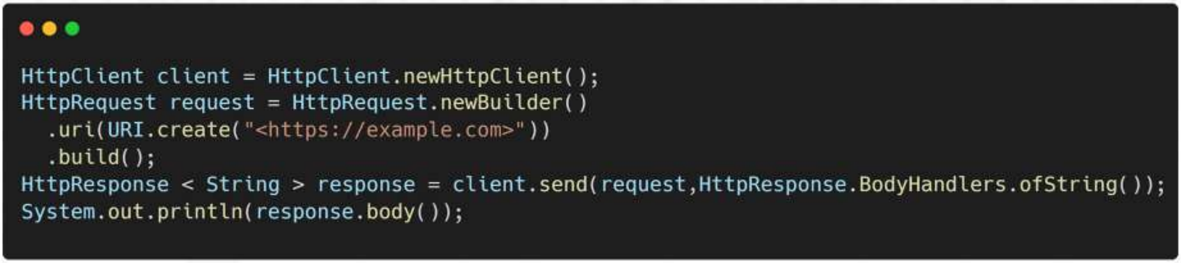
Why it's cool:

A new way to send HTTP requests, replacing the old HttpURL Connection method.

Real-world usage:

- Automating API calls in testing
- Fetching data from REST APIs with HTTP/2 support (more speed!)

Code Snippet:



```java
HttpClient client = HttpClient.newHttpClient();
HttpRequest request = HttpRequest.newBuilder()
  .uri(URI.create("<https://example.com>"))
  .build();
HttpResponse < String > response = client.send(request,HttpResponse.BodyHandlers.ofString());
System.out.println(response.body());
```

Why it matters:

Streamlines API testing and interaction. Faster, more efficient HTTP requests.

New String Methods

Why it's cool:

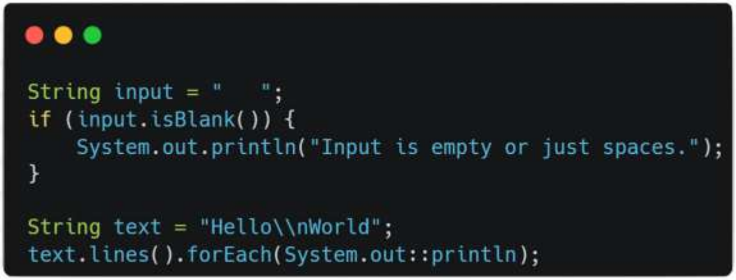
Java made string handling simple with these handy methods:

- **isBlank():** Checks if a string is empty or full of spaces.
- **lines():** Splits text into lines.
- **strip():** Removes unnecessary spaces from both ends.

Real-world usage:

✓ Validating user inputs.
📄 Reading logs or multi-line inputs.
🧹 Cleaning up strings before processing.

Code Snippet:



```java
String input = "   ";
if (input.isBlank()) {
    System.out.println("Input is empty or just spaces.");
}

String text = "Hello\\nWorld";
text.lines().forEach(System.out::println);
```

Why it matters:

These methods make your code cleaner and easier to read! Perfect for input validation in tests.

📂

File Methods

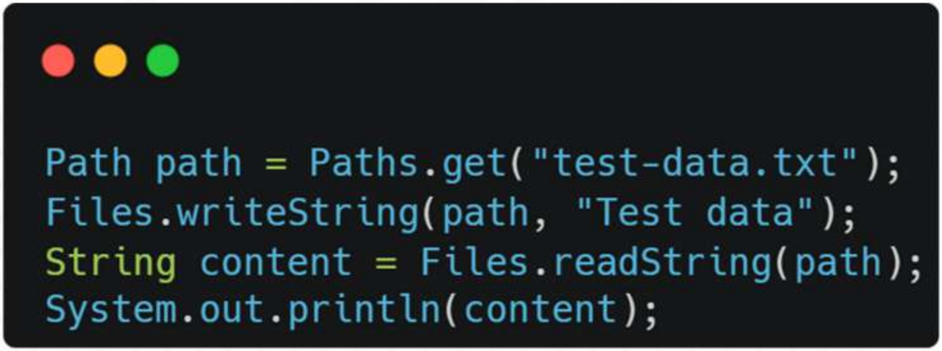
Why it's cool:

Working with files is now as easy as writing and reading strings. Less code, less hassle!

🌎 Real-world usage:

- 📥 Reading large data sets for tests from files.
- 📥 Writing test results or logs quickly.

🖥️ Code Snippet:



```
Path path = Paths.get("test-data.txt");
Files.writeString(path, "Test data");
String content = Files.readString(path);
System.out.println(content);
```

⚠️ Why it matters:

Quick file read/write operations without complex handling. Useful for working with test data files.

❓

Optional Enhancements

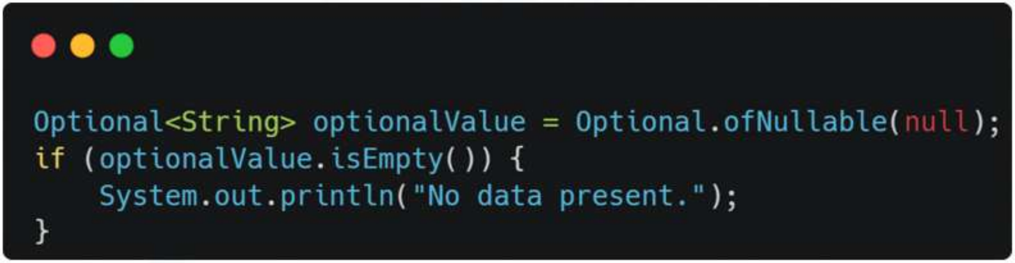
🤔 Why it's cool:

Optional just got better with the new isEmpty() method for cleaner null checks.

🌎 Real-world usage:

🔍 Checking if API responses or database values are present without worrying about null

🧑‍💻 Code Snippet:



```java
Optional<String> optionalValue = Optional.ofNullable(null);
if (optionalValue.isEmpty()) {
    System.out.println("No data present.");
}
```

⚠️ Why it matters:

Simplifies handling missing values, especially in test cases where null checks are frequent.

♻️

Z Garbage Collector (ZGC)

Why it's cool:
The Z Garbage Collector minimizes pauses and optimizes memory management.

🌎 **Real-world usage:**
⚡ Ideal for performance-sensitive apps that require minimal delays, like real-time systems or high-frequency trading apps.

⚠️ **Why it matters:**
Reduces GC pauses to milliseconds, improving app performance during tests, especially for performance and load testing scenarios.

Launch Single-File Source-Code Programs

🤷 Why it's cool:

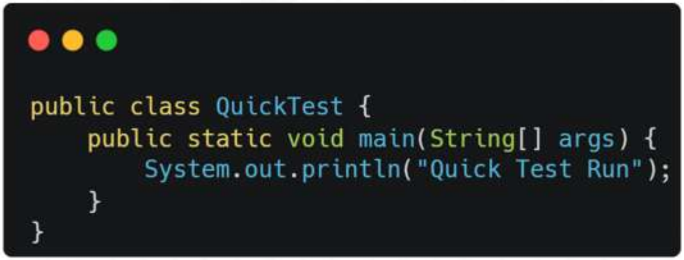
Now you can run a single Java file without compiling it. Perfect for quick tasks or scripts.

🌎 Real-world usage:

🏃 Rapid test script execution without needing a full project structure.

🚀 Prototype test utilities and tools on the fly.

💻 Code Snippet:



```java
public class QuickTest {
    public static void main(String[] args) {
        System.out.println("Quick Test Run");
    }
}
```

⚠️ Why it matters:

Boosts productivity by eliminating extra steps for simple, quick-run Java programs.

❌

Deprecation & Removal of Features

Why it's cool:

Deprecated and removed features clean up Java, making way for modern, more efficient methods.

🌎 **Real-world usage:**

🔄 Migrating old codebases to modern Java practices.

✅ Ensuring your test automation frameworks don't rely on deprecated APIs.

⚠️ **Why it matters:**

Staying updated means fewer bugs, more efficient code, and better performance in test automation suites.