TestNG Reports

TestNG provides several **default reports** after test execution that help you analyze the results. These reports are automatically generated and stored in the `test-output` directory of your project. Below are the **default reports provided by TestNG**:

🔹 1. index.html

- **Location:** `test-output/index.html`
- **Purpose:** Main TestNG HTML report.
- **Contents:**
  - Summary of test results (passed, failed, skipped)
  - Package-wise, class-wise, and method-wise results
  - Time taken by each test
  - Links to detailed reports

🔹 2. emailable-report.html

- **Location:** `test-output/emailable-report.html`
- **Purpose:** A concise summary of the test execution.
- **Contents:**
  - Test methods and their status (pass/fail/skip)
  - Easily sharable via email (hence the name)
- **Advantage:** Clean and simple summary for stakeholders.

🔹 3. testng-results.xml

- **Location:** `test-output/testng-results.xml`
- **Purpose:** XML format result for integrations with CI tools or for custom parsing.
- **Contents:**
  - Structured data of test results
  - Can be used for generating custom reports or integrating with reporting tools like Allure, Jenkins, etc.

🔹 4. testng-failed.xml

- **Location:** `test-output/testng-failed.xml`
- **Purpose:** Contains only the failed test cases.

- **Usage:**
  - Can be rerun easily by referencing this file in your TestNG runner.
  - Helpful for debugging failures without re-running all tests.

🔹 5. Reporter Output (Optional)

- If you use `Reporter.log()` in your tests, these logs are shown:
  - In the `index.html` under each test method
  - In the console output
- **Purpose:** Helps in debugging and adds custom log information in the report.

Summary Table

Report Name	Format	Purpose
`index.html`	HTML	Main interactive report
`emailable-report.html`	HTML	Summary report (sharable)
`testng-results.xml`	XML	Machine-readable result for integrations
`testng-failed.xml`	XML	Rerun failed tests easily
Reporter Output	Text	Custom logs added using `Reporter.log()`

Let me know if you'd like to learn how to **customize or enhance** these reports using tools like **ExtentReports** or **Allure Reports**.

Reading and analyzing **TestNG reports** helps you quickly understand the **status of your test execution**, **failures**, and **overall test coverage**. Here's a step-by-step guide on how to read and analyze each of the default TestNG reports:

✅ 1. `index.html` – Main Detailed Report

📍 Location: `test-output/index.html`

🔍 How to Read:

- **Overview Summary Table**: Top section with number of tests passed, failed, and skipped.
- **Package/Class/Method-wise Details**:
  - Expand sections to view which classes/methods passed or failed.
  - Click on a failed method to view the **stack trace**.
- **Execution Time**: See how long each test took (helps identify performance bottlenecks).
- **Reporter Output**: If you used `Reporter.log()`, logs are visible here under each method.

🔍 What to Analyze:

- Which **methods failed** and **why** (check stack trace).
- How many **tests passed vs failed**.
- **Trends**: Are failures concentrated in a particular class or package?

✅ 2. `emailable-report.html` – Concise Summary Report

📍 Location: `test-output/emailable-report.html`

🔍 How to Read:

- Table showing:
  - **Test method name**
  - **Status** (Passed/Failed/Skipped)
  - **Duration**
  - **Class name**
- Sorted view to quickly scan the results.

🔍 What to Analyze:

- Useful for quick overviews to share with stakeholders.
- Check test distribution and pass/fail ratio.
- Use it to **monitor stability** during CI runs.

✅ 3. `testng-results.xml` – Machine-Readable XML Report

📍 **Location:** `test-output/testng-results.xml`

🔍 How to Read:

- Open in a browser/editor or feed into reporting tools.
- Tags like `<test-method>`, `<status>`, and `<exception>` show detailed results.

🔍 What to Analyze:

- Use in **CI pipelines** for report integration.
- Parse it with tools like Allure, Jenkins plugins, or custom scripts.

✅ 4. `testng-failed.xml` – Rerun Failed Tests

📍 **Location:** `test-output/testng-failed.xml`

🔍 How to Read:

- Contains only the **failed test methods**.
- Used to rerun just the failures.

🔍 What to Analyze:

- Helps isolate and debug failing tests.
- Great for continuous integration pipelines to retry only failures.

✅ 5. Console Output + Reporter Logs

📍 **Usage:**

- Any `Reporter.log("message")` will appear in:
  - `index.html` (under each test method)
  - Console logs

🔍 What to Analyze:

- Useful for debugging or confirming execution steps.
- Add checkpoints to see what values or conditions were met during test.

🧠 Tips for Effective Analysis

Action	Purpose
🔁 Rerun failed tests	Use **`testng-failed.xml`** to rerun and confirm flakiness or stability
📈 Track trends	Over multiple runs, compare pass/fail rates and execution times
🐛 Deep dive into failures	Read stack traces and use logs to identify root causes
🛠️ Integrate with Jenkins/Extent/Allure	For better dashboards and long-term visibility
📦 Group by package/class	Helps you find problematic areas in code coverage

---

Would you like help **setting up a custom report** (like **ExtentReports** or **Allure Reports**) for better visuals and trend tracking?

Here's a clear and detailed **overview of the `test-output` folder** generated by **TestNG** after running your test suite:

📁 TestNG `test-output/` Folder Overview

When you run a TestNG suite, it generates a **`test-output`** folder that contains all test execution results, logs, and reports.

📂 Structure and Contents:

File/Folder	Description
`index.html`	🔹 Main HTML report with expandable summary of passed/failed/skipped tests. 🔹 Interactive, best for in-depth analysis.
`emailable-report.html`	🔹 Concise summary of test results. 🔹 Suitable for emailing to team members.
`testng-results.xml`	🔹 XML file with full structured results. 🔹 Used for custom reporting tools, CI integration.
`testng-failed.xml`	🔹 Contains only the tests that failed. 🔹 Useful for rerunning failed tests.
`junitreports/`	📂 Contains JUnit-style XML reports for compatibility with tools like Jenkins. 🔹 Useful when using **`<junit>`** listener.
`testng-reports/`	📂 Contains internal HTML and XML files used to build the **`index.html`**.
`logs/` (optional)	📂 If enabled, contains log output of the test execution.
`Output by TestNG`	Console output from the test suite execution.

📌 Quick Summary of Key Files

File	Type	Main Use
`index.html`	HTML	Detailed interactive report (most used)
`emailable-report.html`	HTML	Sharable summary report
`testng-results.xml`	XML	Parsing & integration with tools
`testng-failed.xml`	XML	Re-running failed tests
`testng-reports/`	Folder	Backend files used by **`index.html`**
`junitreports/`	Folder	JUnit-style reports for CI tools

📸 Screenshot Representation (Example)

```
plaintext

test-output/
├── emailable-report.html
├── index.html
├── testng-results.xml
├── testng-failed.xml
├── testng-reports/
│   ├── testng.css
│   ├── testng.js
│     ├── suite1_test1_results.html
│     └── ...
├── junitreports/
│     ├── TEST-TestSuite.xml
│     └── ...
└── Output by TestNG
```

🛠️ Optional Enhancements

You can extend the reporting by:

- Integrating with **ExtentReports** or **Allure Reports** for better visuals
- Exporting `testng-results.xml` to CI/CD tools like Jenkins
- Adding custom logs with `Reporter.log()` to enrich output

Would you like a hands-on demo or setup guide for **enhancing your TestNG reports** with Extent or Allure?