# **Specialized Testing Techniques**

## **📌 Table of Contents**

1. Introduction
2. Regression Testing
3. Re-Testing
4. Smoke Testing
5. Sanity Testing
6. Exploratory Testing
7. Ad-hoc Testing
8. Comparison Summary
9. Best Practices
10. Real-World Examples
11. References

## **📖 Introduction**

Specialized testing techniques ensure software quality at different stages of development. This guide covers:  
✔ Regression Testing → Checks for new bugs in old features.  
✔ Re-Testing → Verifies fixed defects.  
✔ Smoke Testing → Basic "health check."  
✔ Sanity Testing → Quick feature validation.  
✔ Exploratory Testing → Unscripted, creative testing.  
✔ Ad-hoc Testing → Random, informal testing.

## **🔄 Regression Testing**

### **What is it?**

Testing existing functionality after changes (bug fixes, new features) to ensure no unintended side effects.

### **When to Use?**

* After code changes.
* Before major releases.

### **Techniques**

* Full Regression: Tests entire system (time-consuming).
* Partial Regression: Focuses on impacted modules.
* Automated Regression: Uses scripts (Selenium, Cypress).

### **Example**

* Scenario: After updating a payment gateway, re-test checkout flow.
* Tools: Selenium, TestNG.

### **Pros & Cons**

| **Pros** | **Cons** |
| --- | --- |
| ✅ Prevents feature breaks. | ❌ Can be time-consuming. |
| ✅ High ROI when automated. | ❌ Requires maintenance of test scripts. |

## **🔁 Re-Testing**

### **What is it?**

Re-executing failed test cases to confirm fixes.

### **Key Points**

* Tests only the fixed defects.
* Uses same data/environment as initial test.

### **Example**

* Defect: Login fails with special characters.
* Re-Test: Verify login works after fix.

### **Regression vs Re-Testing**

| **Aspect** | **Regression Testing** | **Re-Testing** |
| --- | --- | --- |
| Scope | Entire system/modules. | Only fixed defects. |
| Goal | Find new bugs. | Confirm bug fixes. |

## **🔥 Smoke Testing**

### **What is it?**

A quick check to verify if the build is stable enough for further testing.

### **Key Features**

* Tests critical functionalities only.
* Also called "Build Verification Testing (BVT)."

### **Example**

* Test Cases:
  1. App launches.
  2. User can log in.
  3. Homepage loads.

### **When to Use?**

* After every new build.
* Before deep testing (e.g., regression).

## **🧠 Sanity Testing**

### **What is it?**

A narrow, deep test to confirm specific functionality works after minor changes.

### **Smoke vs Sanity Testing**

| **Aspect** | **Smoke Testing** | **Sanity Testing** |
| --- | --- | --- |
| Scope | Broad (critical paths). | Narrow (specific feature). |
| Depth | Shallow. | Deep. |
| When? | After build. | After minor changes. |

### **Example**

* Change: Updated search algorithm.
* Sanity Test: Verify search results are accurate.

## **🔍 Exploratory Testing**

### **What is it?**

Unscripted testing where testers explore the app dynamically.

### **Key Features**

* No predefined test cases.
* Relies on tester’s intuition.
* Finds unexpected bugs.

### **Example**

* Scenario: Randomly navigate an e-commerce site to find UI glitches.

### **Pros & Cons**

| **Pros** | **Cons** |
| --- | --- |
| ✅ Finds edge cases. | ❌ Hard to document. |
| ✅ Flexible & creative. | ❌ Depends on tester skill. |

## **🎯 Ad-hoc Testing**

### **What is it?**

Informal, random testing without planning or documentation.

### **When to Use?**

* Time constraints.
* Quick feedback needed.

### **Ad-hoc vs Exploratory**

| **Aspect** | **Ad-hoc Testing** | **Exploratory Testing** |
| --- | --- | --- |
| Planning | None. | Light (session-based). |
| Documentation | None. | Minimal notes. |

### **Example**

* Test: Try crashing the app by clicking buttons randomly.

## **🆚 Comparison Summary**

| **Technique** | **Purpose** | **Scripted?** | **Best For** |
| --- | --- | --- | --- |
| Regression | Prevent new bugs in old features. | Yes (often automated). | Post-change validation. |
| Re-Testing | Confirm bug fixes. | Yes. | Fixed defects. |
| Smoke | Check build stability. | Yes. | Initial build test. |
| Sanity | Validate specific fixes. | Yes. | Minor changes. |
| Exploratory | Find unexpected issues. | No. | Creative testing. |
| Ad-hoc | Quick, informal checks. | No. | Rapid feedback. |

## **🏆 Best Practices**

✔ Automate regression tests for frequent runs.  
✔ Combine smoke + sanity tests for build confidence.  
✔ Use exploratory testing for UX/edge cases.  
✔ Document ad-hoc findings for future reference.

## **🌍 Real-World Examples**

### **Case 1: Windows 10 Update (Regression Fail)**

* Issue: New update broke printer drivers.
* Root Cause: Skipped regression testing on peripheral devices.

### **Case 2: Airbnb (Exploratory Success)**

* Bug Found: Calendar glitch allowed double bookings.
* How: Tester randomly navigated date picker.

### **Case 3: Facebook (Smoke Test Critical)**

* Process: Every build must pass 5-minute smoke test before deployment.

## **📚 References**

* [ISTQB Glossary](https://www.istqb.org/glossary)
* [Exploratory Testing (James Bach)](https://www.satisfice.com/exploratory-testing)
* [Microsoft Test Guidelines](https://docs.microsoft.com/en-us/devops/develop/test/)

🔹 Conclusion:

* Scripted tests (Regression, Smoke, Sanity) ensure stability.
* Unscripted tests (Exploratory, Ad-hoc) find hidden bugs.
* Balance both for maximum coverage.