

Introduction to Software Testing

Introduction to Software Testing

(A concise, easy-to-read guide)

1. What Is Software Testing?

Aspect	Description
Definition	The systematic process of evaluating a software product to find defects, verify that it meets requirements, and assess its quality.
Primary Goals	1. Detect defects early. 2. Ensure the product behaves as expected. 3. Provide confidence to stakeholders.
Key Question	<i>Does the software do what it is supposed to do, and does it do it well?</i>

2. Why Test? (The “Why” Behind Testing)

1. **Quality Assurance** – Improves reliability, performance, and user satisfaction.
 2. **Risk Reduction** – Prevents costly failures in production (financial loss, brand damage, safety hazards).
 3. **Compliance** – Meets regulatory or contractual obligations (e.g., ISO, HIPAA).
 4. **Feedback Loop** – Gives developers early insight into design or implementation flaws.
 5. **Maintainability** – Facilitates future changes by establishing a safety net of regression tests.
-

3. Fundamental Concepts

Concept	Meaning	Typical Use
Defect / Bug	A deviation from expected behavior.	Reported, tracked, and fixed.
Test Case	A set of inputs, execution conditions, and expected results.	Basis for manual or automated testing.
Test Suite	Collection of related test cases.	Organized execution (e.g., smoke suite, regression suite).

Introduction to Software Testing

Concept	Meaning	Typical Use
Test Plan	Document describing scope, approach, resources, schedule, and deliverables.	Guides testing activities.
Test Strategy	High-level description of testing objectives and methods (e.g., "focus on security testing").	Aligns testing with project goals.
Verification vs. Validation	<i>Verification</i> : "Are we building the product right?" (conformance to specifications). <i>Validation</i> : "Are we building the right product?" (meeting user needs).	Both are essential; verification is often static, validation is dynamic.

4. Levels of Testing

Level	Focus	Typical Activities
Unit Testing	Individual components/functions.	Write/execute test code (often automated).
Integration Testing	Interaction between combined units/modules.	Test interfaces, data flow, API contracts.
System Testing	Whole system as a black box.	Functional, performance, security, usability tests.
Acceptance Testing	Business/Stakeholder perspective.	User Acceptance Testing (UAT), Contract Acceptance, Alpha/Beta testing.

5. Types of Testing (Cross-cutting)

Type	When Used	Goal
Functional	Throughout development	Verify features against requirements.
Non-Functional	After functional baseline	Assess performance, security, usability, reliability, etc.
Static Testing	Early (requirements, design)	Review, walkthrough, static analysis – find defects without executing code.
Dynamic Testing	After code is runnable	Execute software, observe behavior.
Regression Testing	After changes/fixes	Ensure existing functionality remains intact.

Introduction to Software Testing

Type	When Used	Goal
Smoke / Sanity Testing	Build verification	Quick checks to confirm basic stability before deeper testing.
Exploratory Testing	When requirements are vague or to uncover hidden issues	Simultaneous learning, test design, and execution.