**FUNCTIONAL TESTING** is a type of software testing that validates the software system against the functional requirements/specifications. The purpose of Functional tests is to test each function of the software application, by providing appropriate input, verifying the output against the Functional requirements.

**End To End Testing** is a software testing method that validates entire software from starting to the end along with its integration with external interfaces. The purpose of end-to-end testing is testing whole software for dependencies, data integrity and communication with other systems, interfaces and databases to exercise complete production like scenario.

Along with the software system, it also validates batch/data processing from other upstream/downstream systems. Hence, the name **“End-to-End”**. End to End Testing is usually executed after functional and [System Testing](https://www.guru99.com/system-testing.html). It uses actual production like data and test environment to simulate real-time settings. End-to-End testing is also called **Chain Testing**.

**Ad hoc Testing**

**Ad hoc Testing** is an informal or unstructured software testing type that aims to break the testing process in order to find possible defects or errors at an early possible stage. Ad hoc testing is done randomly and it is usually an unplanned activity which does not follow any documentation and test design techniques to create test cases.



Ad hoc Testing does not follow any structured way of testing and it is randomly done on any part of application. Main aim of this testing is to find defects by random checking. Adhoc testing can be achieved with the Software testing technique called **Error Guessing.** Error guessing can be done by the people having enough experience on the system to “guess” the most likely source of errors.

This testing requires no documentation/ planning /process to be followed. Since this testing aims at finding defects through random approach, without any documentation, defects will not be mapped to test cases. This means that, sometimes, it is very difficult to reproduce the defects as there are no test steps or requirements mapped to it.

**Risk Based Testing**

**Risk Based Testing (RBT)** is a software testing type which is based on the probability of risk. It involves assessing the risk based on software complexity, criticality of business, frequency of use, possible areas with[Defect](https://www.guru99.com/defect-management-process.html)etc. Risk based testing prioritizes testing of features and functions of the software application which are more impactful and likely to have defects.

Risk is the occurrence of an uncertain event with a positive or negative effect on the measurable success criteria of a project. It could be events that have occurred in the past or current events or something that could happen in the future. These uncertain events can have an impact on the cost, business, technical and quality targets of a project.

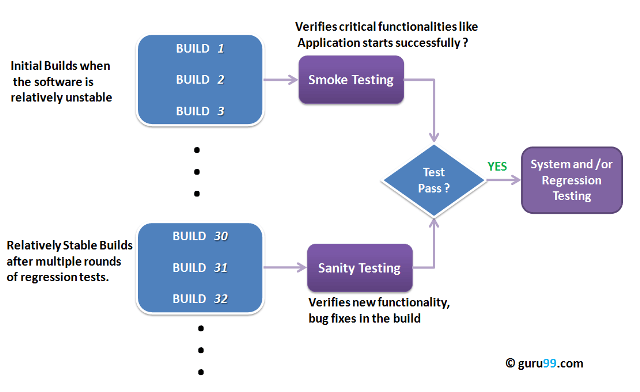
Risks can be positive or negative.

* **Positive risks** are referred to as opportunities and help in business sustainability. For example investing in a New project, Changing business processes, Developing new products.
* **Negative Risks** are referred to as threats and recommendations to minimize or eliminate them must be implemented for project success.

# Sanity Testing Vs Smoke Testing: Introduction and Differences

Smoke and Sanity testing difference is the most misunderstood topic in Software Testing. There is an enormous amount of literature on the subject, but most of them are confusing. The following article makes an attempt to address the confusion.

The key differences between Smoke Testing and Sanity Testing can be learned with the help of the following diagram –



## Smoke Testing

**Smoke Testing** is a software testing technique performed post software build to verify that the critical functionalities of software are working fine. It is executed before any detailed functional or regression tests are executed. The main purpose of smoke testing is to reject a software application with defects so that QA team does not waste time testing broken software application.

In [Smoke Testing](https://www.guru99.com/smoke-testing.html), the test cases chose to cover the most important functionality or component of the system. The objective is not to perform exhaustive testing, but to verify that the critical functionalities of the system are working fine.  
For Example, a typical smoke test would be – Verify that the application launches successfully, Check that the GUI is responsive … etc.

## What is Sanity Testing?

Sanity testing is a kind of Software Testing performed after receiving a software build, with minor changes in code, or functionality, to ascertain that the bugs have been fixed and no further issues are introduced due to these changes. The goal is to determine that the proposed functionality works roughly as expected. If sanity test fails, the build is rejected to save the time and costs involved in a more rigorous testing.

The objective is “not” to verify thoroughly the new functionality but to determine that the developer has applied some rationality (sanity) while producing the software. For instance, if your scientific calculator gives the result of 2 + 2 =5! Then, there is no point testing the advanced functionalities like sin 30 + cos 50.

**Smoke Testing Vs Sanity Testing – Key Differences**

Following is the difference between Sanity and Smoke testing:

|  |  |
| --- | --- |
| **Smoke Testing** | **Sanity Testing** |
| Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine | Sanity Testing is done to check the new functionality/bugs have been fixed |
| The objective of this testing is to verify the “stability” of the system in order to proceed with more rigorous testing | The objective of the testing is to verify the “rationality” of the system in order to proceed with more rigorous testing |
| This testing is performed by the developers or testers | Sanity testing in software testing is usually performed by testers |
| Smoke testing is usually documented or scripted | Sanity testing is usually not documented and is unscripted |
| Smoke testing is a subset of Acceptance testing | Sanity testing is a subset of [Regression Testing](https://www.guru99.com/regression-testing.html) |
| Smoke testing exercises the entire system from end to end | Sanity testing exercises only the particular component of the entire system |
| Smoke testing is like General Health Check Up | Sanity Testing is like specialized health check up |

# Difference Between Retesting and Regression Testing

## Retesting

**Retesting** is a process to check specific test cases that are found with bug/s in the final execution. Generally, testers find these bugs while testing the software application and assign it to the developers to fix it. Then the developers fix the bug/s and assign it back to the testers for verification. This continuous process is called Retesting.

### What is Regression Testing?

[Regression Testing](https://www.guru99.com/regression-testing.html) is a type of software testing executed to check whether a code change has not unfavorably disturbed current features & functions of an Application

### Retesting vs Regression Testing

|  |  |
| --- | --- |
| **Regression Testing** | **Re-testing** |
| * Regression Testing is carried out to confirm whether a recent program or code change has not adversely affected existing features | * Re-testing is carried out to confirm the test cases that failed in the final execution are passing after the defects are fixed |
| * The purpose of Regression Testing is that new code changes should not have any side effects to existing functionalities | * Re-testing is done on the basis of the[Defect](https://www.guru99.com/defect-management-process.html)fixes |
| * Defect verification is not the part of Regression Testing | * Defect verification is the part of re-testing |
| * Based on the project and availability of resources, Regression Testing can be carried out parallel with Re-testing | * Priority of re-testing is higher than regression testing, so it is carried out before regression testing |
| * You can do automation for regression testing, [Manual Testing](https://www.guru99.com/manual-testing.html) could be expensive and time-consuming | * You cannot automate the test cases for Retesting |
| * Regression testing is known as a generic testing | * Re-testing is a planned testing |
| * Regression testing is done for passed test cases | * Retesting is done only for failed test cases |
| * Regression testing checks for unexpected side-effects | * Re-testing makes sure that the original fault has been corrected |
| * Regression testing is only done when there is any modification or changes become mandatory in an existing project | * Re-testing executes a defect with the same data and the same environment with different inputs with a new build |
| * Test cases for regression testing can be obtained from the functional specification, user tutorials and manuals, and defect reports in regards to corrected problems | * Test cases for retesting cannot be obtained before start testing. |

# What is Exploratory Testing

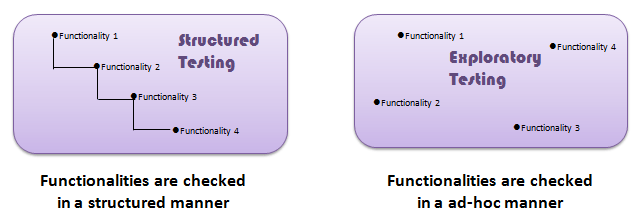
## What is Exploratory Testing?

**Exploratory Testing** is a type of software testing where Test cases are not created in advance but testers check system on the fly. They may note down ideas about what to test before test execution. The focus of exploratory testing is more on testing as a “thinking” activity.

Exploratory Testing is widely used in Agile models and is all about discovery, investigation, and learning. It emphasizes personal freedom and responsibility of the individual tester.

Under scripted testing, you design test cases first and later proceed with test execution. On the contrary, exploratory testing is a simultaneous process of test design and test execution all done at the same time.

Scripted Test Execution is usually a non-thinking activity where testers execute the test steps and compare the actual results with expected results. Such test execution activity can be automated does not require many cognitive skills.



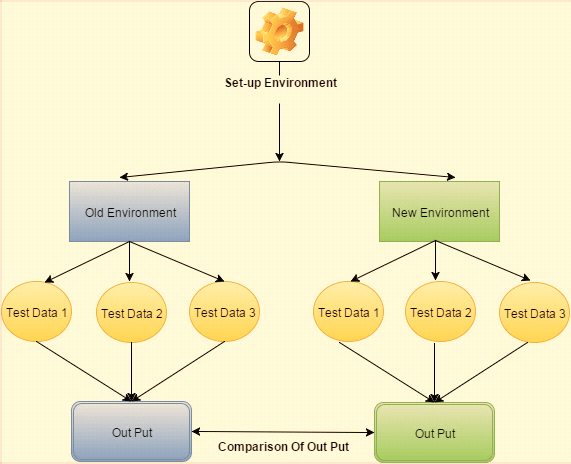
**Differences between Scripted and Exploratory Testing**

|  |  |
| --- | --- |
| **Scripted Testing** | **Exploratory Testing** |
| Directed from requirements | Directed from requirements and exploring during testing |
| Determination of test cases well in advance | Determination of test cases during testing |
| Confirmation of testing with the requirements | Investigation of system or application |
| Emphasizes prediction and decision making | Emphasizes adaptability and learning |
| Involves confirmed testing | Involves Investigation |
| Is about Controlling tests | Is about Improvement of test design |
| Like making a speech – you read from a draft | Like making a conversation – it’s spontaneous |
| The script is in control | The tester’s mind is in control |

## Parallel Testing

**Parallel Testing** is a software testing type in which multiple versions or subcomponents of an application are tested with same input on different systems simultaneously to reduce test execution time. The purpose of parallel testing is finding out if legacy version and new version are behaving the same or differently and ensuring whether new version is more efficient or not.

The below image demonstrate the parallel testing.



### Parallel Testing Example

When any organization is moving from old system to new system, legacy data is an important part. Transferring this data is a complex process.

In software testing, verifying compatibility of the newly developed system with the old system is done through “parallel testing.”



### What is Concurrency Testing?

Concurrency Testing is defined as a testing technique to detect the defects in an application when multiple users are logged in. In other words monitoring the effect while multiple users perform the same action at the same time. The image below show the concurrent testing

Concurrent testing is also referred as **multi-user** testing. Testing concurrent program is more challenging then testing sequential program, due to non-determinism and synchronization issues.

