|  |
| --- |
| **A Functional Testing Test Cases** |
| The functional testing of Mobiles normally consists in the areas of testing user interactions as well as testing the transactions. The various factors which are relevant in functional testing are |
| Type of application based upon the business functionality usages (banking, gaming, social or business) |
| Target audience type (consumer, enterprise, education) |
| Distribution channel which is used to spread the application (e.g. Apple App Store, Google play, direct distribution) |
| The most fundamental test scenarios in the functional testing can be considered as : |
| To validate whether all the required mandatory fields are working as required. |
| To validate that the mandatory fields are displayed in the screen in a distinctive way than the non-mandatory fields. |
| To validate whether the application works as per as requirement whenever the application starts/stops. |
| To validate whether the application goes into minimized mode whenever there is an incoming phone call. In order to validate the same we need to use a second phone, to call the device. |
| To validate whether the phone is able to store, process and receive SMS whenever the app is running. In order to validate the same we need to use a second phone to send sms to the device which is being tested and where the application under test is currently running. |
| To validate that the device is able to perform required multitasking requirements whenever it is necessary to do so. |
| To validate that the application allows necessary social network options such as sharing, posting and navigation etc. |
| To validate that the application supports any payment gateway transaction such as Visa, Mastercard, Paypal etc as required by the application. |
| To validate that the page scrolling scenarios are being enabled in the application as necessary. |
| To validate that the navigation between relevant modules in the application are as per the requirement. |
| To validate that the truncation errors are absolutely to an affordable limit. |
| To validate that the user receives an appropriate error message like “Network error. Please try after some time” whenever there is any network error. |
| To validate that the installed application enables other applications to perform satisfactorily, and it does not eat into the memory of the other applications. |
| To validate that the application resumes at the last operation in case of a hard reboot or system crash. |
| To validate whether the installation of the application can be done smoothly provided the user has the necessary resources and it does not lead to any significant errors. |
| To validate that the application performs auto start facility according to the requirements. |
| To validate whether the application performs according to the requirement in all versions of Mobile that is 2g, 3g and 4g. |
| [To perform Regression Testing to uncover new software bugs in existing areas of a system after changes have been made to them. Also rerun previously performed tests to determine that the program behavior has not changed due to the changes.](https://www.guru99.com/regression-testing.html) |
| To validate whether the application provides an available user guide for those who are not familiar to the app |

|  |
| --- |
| **B Performance Testing Test Cases** |
| This type of testing’s fundamental objective is to ensure that the application performs acceptably under certain performance requirements such as access by a huge number of users or the removal of a key infrastructure part like a database server.  The general test scenarios for Performance Testing in a Mobile application are: |
| To determine whether the application performs as per the requirement under different load conditions. |
| To determine whether the current network coverage is able to support the application at peak, average and minimum user levels. |
| To determine whether the existing client-server configuration setup provides the required optimum performance level. |
| To identify the various application and infrastructure bottlenecks which prevent the application to perform at the required acceptability levels. |
| To validate whether the response time of the application is as per as the requirements. |
| To evaluate product and/or hardware to determine if it can handle projected load volumes. |
| To evaluate whether the battery life can support the application to perform under projected load volumes. |
| To validate application performance when network is changed to WIFI from 2G/3G or vice versa. |
| To validate each of the required the CPU cycle is optimization |
| To validate that the battery consumption, memory leaks, resources like GPS, Camera performance is well within required guidelines. |
| To validate the application longevity whenever the user load is rigorous. |
| To validate the network performance while moving around with the device. |
| To validate the application performance when only intermittent phases of connectivity is required. |

|  |
| --- |
| **C Security Testing Test Cases** |
| The fundamental objective of security testing is to ensure that the application’s data and networking security requirements are met as per guidelines.  The following are the most crucial areas for checking the security of Mobile applications. |
| To validate that the application is able to withstand any brute force attack which is an automated process of trial and error used to guess a person’s username, password or credit-card number. |
| To validate whether an application is not permitting an attacker to access sensitive content or functionality without proper authentication. |
| To validate that the application has a strong password protection system and it does not permit an attacker to obtain, change or recover another user’s password. |
| To validate that the application does not suffer from insufficient session expiration. |
| To identify the dynamic dependencies and take measures to prevent any attacker for accessing these vulnerabilities. |
| [To prevent from SQL injection related attacks.](https://www.guru99.com/sql.html) |
| To identify and recover from any unmanaged code scenarios. |
| To ensure whether the certificates are validated, does the application implement Certificate Pinning or not. |
| To protect the application and the network from the denial of service attacks. |
| To analyze the data storage and data validation requirements. |
| To enable the session management for preventing unauthorized users to access unsolicited information. |
| To check if any cryptography code is broken and ensure that it is repaired. |
| To validate whether the business logic implementation is secured and not vulnerable to any attack from outside. |
| To analyze file system interactions, determine any vulnerability and correct these problems. |
| To validate the protocol handlers for example trying to reconfigure the default landing page for the application using a malicious iframe. |
| To protect against malicious client side injections. |
| To protect against malicious runtime injections. |
| To investigate file caching and prevent any malicious possibilities from the same. |
| To prevent from insecure data storage in the keyboard cache of the applications. |
| To investigate cookies and preventing any malicious deeds from the cookies. |
| To provide regular audits for data protection analysis. |
| Investigate custom created files and preventing any malicious deeds from the custom created files. |
| To prevent from buffer overflows and memory corruption cases. |
| To analyze different data streams and preventing any vulnerabilities from these. |

|  |
| --- |
| **D Usability Testing Test Cases** |
| The usability testing process of the Mobile application is performed to have a quick and easy step application with less functionality than a slow and difficult application with many features. The main objective is to ensure that we end up having an easy-to-use, intuitive and similar to industry-accepted interfaces which are widely used. |
| To ensure that the buttons should have the required size and be suitable to big fingers. |
| To ensure that the buttons are placed in the same section of the screen to avoid confusion to the end users. |
| To ensure that the icons are natural and consistent with the application. |
| To ensure that the buttons, which have the same function should also have the same color. |
| To ensure that the validation for the tapping zoom-in and zoom-out facilities should be enabled. |
| To ensure that the keyboard input can be minimized in an appropriate manner. |
| To ensure that the application provides a method for going back or undoing an action, on touching the wrong item, within an acceptable duration. |
| To ensure that the contextual menus are not overloaded because it has to be used quickly. |
| To ensure that the text is kept simple and clear to be visible to the users. |
| To ensure that the short sentences and paragraphs are readable to the end users. |
| To ensure that the font size is big enough to be readable and not too big or too small. |
| To validate the application prompts the user whenever the user starts downloading a large amount of data which may be not conducive for the application performance. |
| To validate that the closing of the application is performed from different states and verify if it re-opens in the same state. |
| To ensure that all strings are converted into appropriate languages whenever a language translation facility is available. |
| To ensure that the application items are always synchronized according to the user actions. |
| To ensure that the end user is provided with a user manual which helps the end user to understand and operate the application who may be not familiar with the application’s proceedings |
| Usability testing is normally performed by manual users since only human beings can understand the sensibility and comfort ability of the other users. |

|  |
| --- |
| **E Compatibility Testing Test Cases** |
| Compatibility testing on mobile devices is performed to ensure that since mobile devices have different size, resolution, screen, version and hardware so the application should be tested across all the devices to ensure that the application works as desired.  The following are the most prominent areas for compatibility testing. |
| To validate that the user Interface of the application is as per the screen size of the device, no text/control is partially invisible or inaccessible. |
| To ensure that the text is readable for all users for the application. |
| To ensure that the call/alarm functionality is enabled whenever the application is running. The application is minimized or suspended on the event of a call and then whenever the call stops the application is resumed. |

|  |
| --- |
| **F Recoverability Testing Test Cases** |
|  |
| Crash recovery and transaction interruptions |
| Validation of the effective application recovery situation post unexpected interruption/crash scenarios. |
| Verification of how the application handles a transaction during a power failure (i.e. Battery dies or a sudden manual shutdown of the device) |
| The validation of the process where the connection is suspended, the system needs to re-establish for recovering the data directly affected by the suspended connection. |

|  |
| --- |
| **G Important Checklist** |
|  |
| Installation testing (whether the application can be installed in a reasonable amount of time and with required criterion) |
| Uninstallation testing (whether the application can be uninstalled in a reasonable amount of time and with required criterion) |
| Network test cases (validation of whether the network is performing under required load or not, whether the network is able to support all the necessary applications during the testing procedures) |
| Check Unmapped keys (Unmapped keys: Each and every screen in the app might require a different set of active keys. Keys which are not required by the screen can be left unmapped.) |
| Check application splash screen (Splash screens may be an innocuous part of the user experience. It’s just a launch screen, there’s not much to it. But first impressions count and the devil is in the details.  A splash screen is a screen which appears when you open an app on your mobile device. Sometimes it’s referred to as a launch screen or startup screen and shows up when your app is loading after you’ve just opened it. When the loading is finished, you’ll be taken to a more functional screen where you can complete actions. Splash screens appear on your screen for a fleeting moment – look away and your might miss them. Traditionally, you’ll see a logo and company name and, if you’re lucky, the company motto.) |
| Continued keypad entry during interrupts and other times like network issues |
| Methods which deal with exiting the application |
| Charger effect while an application is running in the background |
| Low battery and high performance demand |
| Removal of battery while an application is being performed |
| Consumption of battery by application |
| Check Application side effects |
| Native Apps  Native apps live on the device and are accessed through icons on the device home screen. Native apps are installed through an application store (such as Google Play or Apple’s App Store). They are developed specifically for one platform, and can take full advantage of all the device features — they can use the camera, the GPS, the accelerometer, the compass, the list of contacts, and so on. They can also incorporate gestures (either standard operating-system gestures or new, app-defined gestures). And native apps can use the device’s notification system and can work offline.  Mobile Web Apps  Web apps are not real applications; they are really websites that, in many ways, look and feel like native applications, but are not implemented as such. They are run by a browser and typically written in HTML5. Users first access them as they would access any web page: they navigate to a special URL and then have the option of “installing” them on their home screen by creating a bookmark to that page.  Hybrid apps  Hybrid apps are part native apps, part web apps. (Because of that, many people incorrectly call them “web apps”). Like native apps, they live in an app store and can take advantage of the many device features available. Like web apps, they rely on HTML being rendered in a browser, with the caveat that the browser is embedded within the app. |
| Mixed strategy: Testing on all available devices is desired but might not be feasible in all cases. In such scenarios, you can compensate for the lack of devices with simulators and emulators. |

|  |
| --- |
| **H Benefits of app testing** |
|  |
| Testing of mobile apps ensures that only the high-performance apps are published. |
| Testing also makes sure that the application operates with consistent performance on across network bandwidths. |
| Stability and reliability of the mobile apps are confirmed during the testing process. |
| When untested apps are released to the end users it develops many bugs and scores less on the performance metrics which in turn lead to uninstalls. Testing provides a ready-to-market mobile app in a stipulated period of time which helps in customer retention. |
| Testing also ensures that the mobile app has the best in class UI and UX suitable for the target audience. |

==============================================================================

**Native Apps**

Native apps live on the device and are accessed through icons on the device home screen.

Native apps are installed through an application store (such as Google Play or Apple’s App

Store). They are developed specifically for one platform, and can take full advantage of all

the device features — they can use the camera, the GPS, the accelerometer, the compass, the

list of contacts, and so on. They can also incorporate gestures (either standard operating-

system gestures or new, app-defined gestures). And native apps can use the device’s

notification system and can work offline.

**Mobile Web Apps**

Web apps are not real applications; they are really websites that, in many ways, look and

feel like native applications, but are not implemented as such. They are run by a browser

and typically written in HTML5. Users first access them as they would access any web page:

they navigate to a special URL and then have the option of “installing” them on their home

screen by creating a bookmark to that page.

**Hybrid apps**

Hybrid apps are part native apps, part web apps. (Because of that, many people

incorrectly call them “web apps”). Like native apps, they live in an app store and can take

advantage of the many device features available. Like web apps, they rely on HTML being

rendered in a browser, with the caveat that the browser is embedded within the app.

====================================================================================

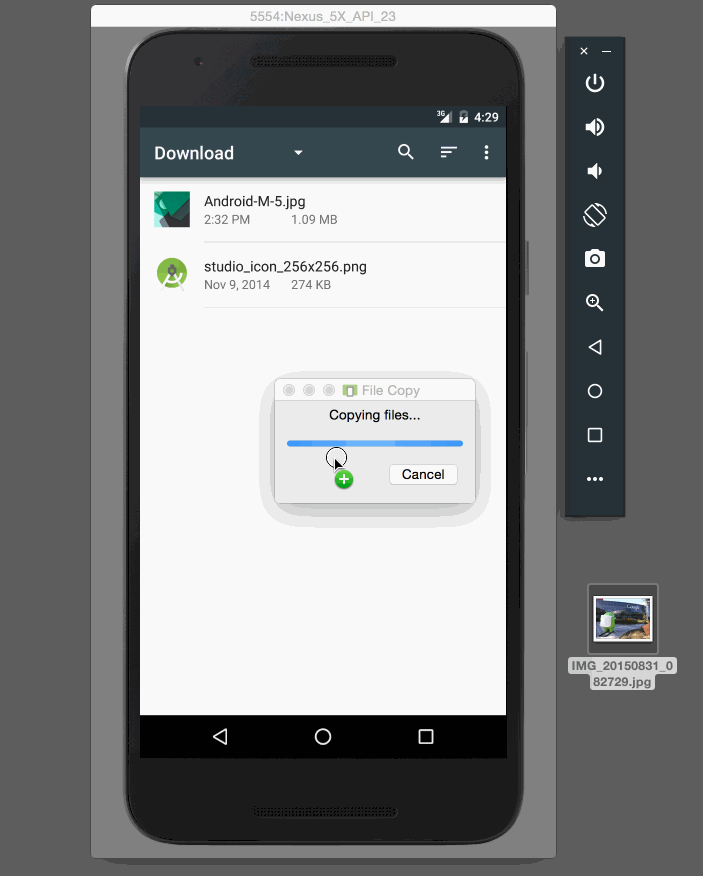
Real Devices Vs Emulator/Simulator in Mobile Testing: Which is Best?

Posted on October 16, 2017 at 12:01 pm.

Written by [Indium](https://huddle.eurostarsoftwaretesting.com/members/indiumsoft/profile/)

*Reading Time: 3 minutes*

Mobile testing has [become a nightmare](https://huddle.eurostarsoftwaretesting.com/5-mobile-development-pitfalls/) due to increasing number of devices and configurations that mobile apps and websites need to be cross-checked against. While both Emulators and Simulators look like great alternatives for testing apps, since having a wide range of real devices also accounts for the cost involved.  These tests should confirm the seamless running of applications. Thankfully, today we have a wide range of testing tools for thorough testing.



Emulators, Simulators, or Real Devices

These tools can be categorised into three broad categories: emulators, simulators, or real devices. Well, few may say that emulators and simulators can be clubbed in a single group. But, real device testing does what the name implies – it tests the application as they run on the users’ smartphones.

Let’s take a deeper look at the categories.

Real device mobile testing requires the use of the actual smartphone. But emulators and simulators are based on the concept of virtual testing. Virtual testing involves testing the software that provides the same functionality as provided by the real phone.

For testing mobile apps, an emulator is a desktop application that mimics the hardware and OS of the applications that should be tested. A simulator does not mimic the hardware/OS, but rather it mimics the basic behavior of a device. While simulators are usually simpler in function than emulators, they are not as useful as emulators. Real device testing checks the functionality of mobile apps thoroughly and ensures the appropriate working of the apps.

Emulators provide better results compared to simulators as they can be used to test specific situations or cases, and also can mimic multiple devices. Emulators are often used more as they are relatively cheaper – specifically if compared to real devices.

Emulator/Simulators VS Real Devices

Let us try to compare what provides better benefits in the following scenarios:

* **Situation-based scenarios:** In situation-based scenarios, buying required mobile devices may be tough – while using the emulator/simulator may take care of the need. This means, not all the scenarios can be tested in simulators/emulators – this is possible only if testing is done on the mobile.
* **Ease-of-use scenarios:** Emulators or Simulators make testing the plethora of apps on the multiple mobile devices available today much easier, but they miss out the UI and UX aspects of an app, including the color and brightness display.
* **Ease-of-access scenarios:** Testing on mobile devices is reliable – however, testing on Emulators and Simulators makes it easier as it requires only the URL for the app.
* **Validating battery scenarios:** Mobile devices can perform this test as required whereas emulators and simulators usually are not able to mimic the issues related to the battery.
* **Validating Performance:** This is a very crucial part of testing a mobile application. Testing the performance of an app on a mobile device usually gives a more accurate result as compared when tested on emulators or simulators.

Disadvantages of Emulators and Simulators

While both Emulators and Simulators look like great alternatives for testing apps, since having a wide range of real devices also accounts for the cost involved. But, they have issues that may generate both false positive and negative results, which will surely have a negative impact on the business ROI. Another disadvantage of emulator/simulator is that howsoever much close their feature may be to the real app, emulator/simulator may still not cover all features/scenarios and the results obtained may not be as accurate as we would want them to be.

[Real device testing](http://www.indiumsoft.com/mobile-apps-testing-services/) wins the challenge over either of the two. With an efficient testing process, real devices can validate for every possible scenario with almost 100% accuracy.

However, having all three options have their own benefits, and can be used according to the need, helping organisations meet their goals.