What are desktop applications?

An application that runs stand-alone in a desktop or laptop computer. Contrast with "Web-based application," which requires the Web browser to run. The term may be used to contrast desktop applications with mobile applications that run-in smartphones and tablets.

Some examples of desktop applications for Windows:

- Windows File Explorer (to find files on your computer)
- Microsoft Office applications (Word, Excel, etc.)
- Web browsers (Chrome, Firefox, Internet Explorer)
- Adobe Photoshop

What is a web app?

A web app is an internet-enabled application that runs through a web browser, designed and developed for many devices, independent of the operating system.

Some examples of web applications:

- Google search (to find stuff on the web)
- Google Docs (like Microsoft Office, but they it runs inside a web browser)
- Facebook
- Twitter

What does Mobile Application (Mobile App) mean?

A mobile application, most commonly referred to as an app, is a type of application software designed to run on a mobile device, such as a smartphone or tablet computer.

Apps are generally small, individual software units with limited function. This use of app software was originally popularized by Apple Inc. and its App Store, which offers thousands of applications for the iPhone, iPad and iPod Touch.

Mobile app pros

- Faster Speed: experience quicker overall usability.
- Native promotional options: market your app within the app store.
- Push notifications: send reminders to users via badges and banners.
- Integrated device functionality allows access to system features including hardware and other software.
- Quick access: add shortcuts to your native app.

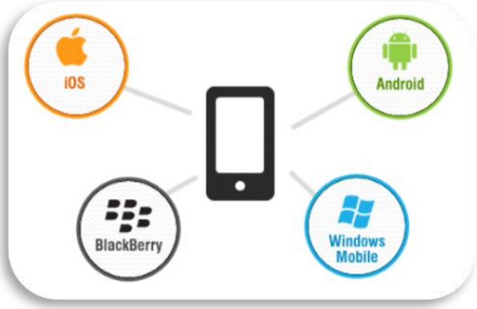
Mobile app cons

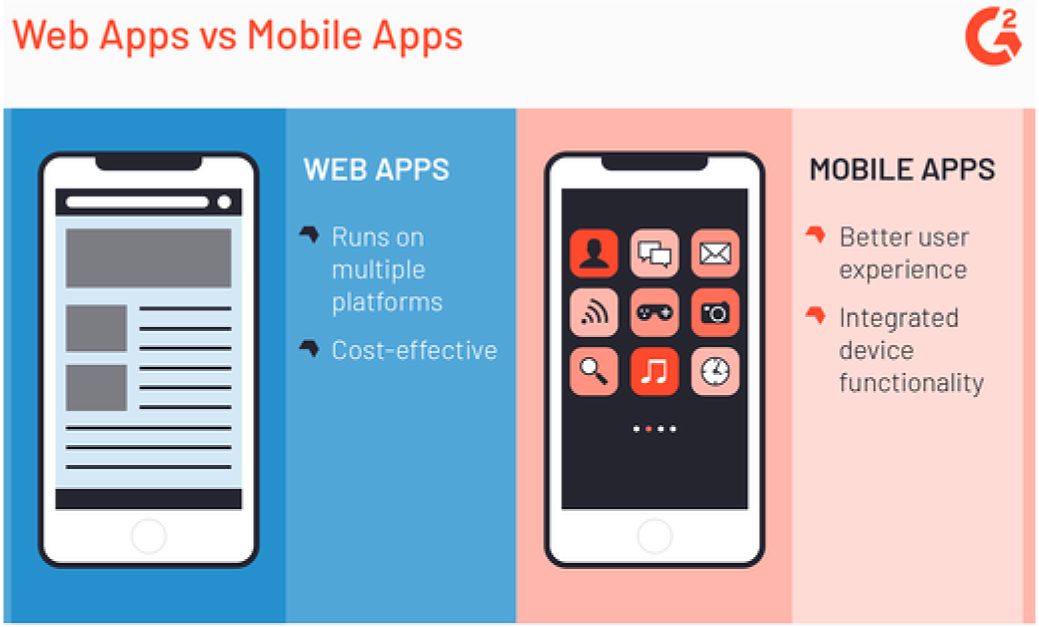
- More expensive: typically have higher development cost than web apps.
- Approval process: mobile apps are all verified in their respective app stores, which can prove tricky at times.
- Installation: users must download the app manually in order to use it, taking up storage space.

- Updates: periodic system releases mean a lot of continual maintenance work for app developers.

Differences between mobile and desktop apps:

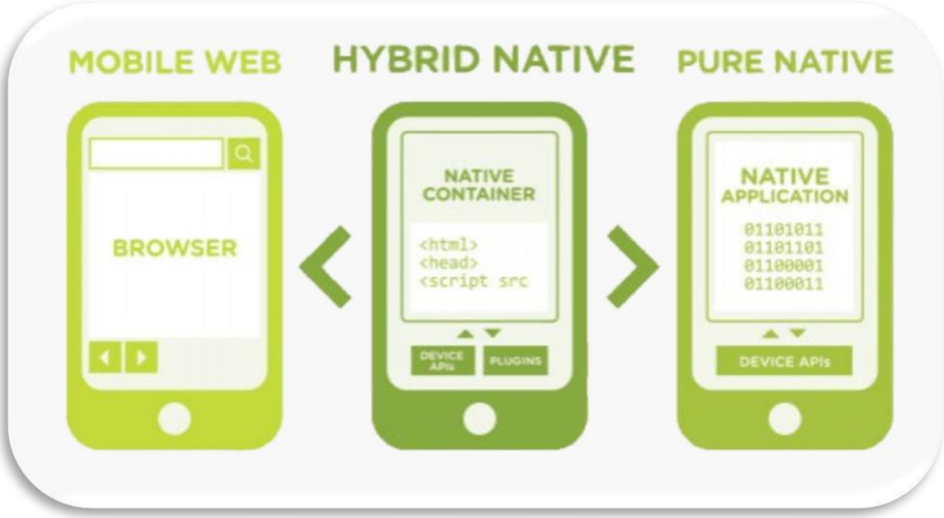
- The mobile device is a system, that has not powerful stuffing. So, it cannot work as a personal computer.
- The mobile application testing in provided on handsets (Apple, Samsung, Nokia, etc.), while the desktop app is tested on a central processor.
- Mobile devices screens variety, their extensions, and colors. Mobile phone screen size is smaller than desktop ones.
- Making and receiving calls is the main task of the phone, that is why the application should not interfere with this major function.
- A wide variety of specific operating systems and component configurations: Android, iOS, BlackBerry etc.
- Mobile phone OS quickly becomes obsolete. In addition, there is a limit to updating their OS.
- Mobile devices use network connections (3G, 4G, Wi-Fi), desktop use broadband connection or Wi-Fi.
- Mobile devices constantly search the network. That is why you should test the application at different data rates.
- Tools, which are good for the desktop apps testing, are not fully suitable for the mobile application testing.
- Mobile applications must support multiple input channels (keyboard, voice, gestures, etc.), multimedia technologies and other features that increase their usability.





Type of Mobile application:

Three main types of the mobile apps are divided: Mobile Web Apps, Native (Pure native) Apps, and Hybrid Apps.



Mobile Web application, in fact, is the website opened in the gadget (smartphone or tablet) with the help of the mobile browser.

Some merits of the Mobile Web Apps:

- Easy development.

- Easy access.
- Easy update.
- Mobile Web App requires no installation.

Some demerits of the Mobile Web Apps:

- No offline capabilities support.
- Limited functionality in the comparison with Hybrid and Native Apps. (no access to the file system and local resources).
- Problems with redistribution: Google Play and App Store don't support redistribution of the Mobile Web Apps.

Native App is the application, which has been developed specifically for one platform (Android, iOS, Tizen, Windows 10 M0bile, BlackBerry).

Some merits of the Native Apps:

- Native app works offline.
- It can use all features of its device.
- Advanced user experience.
- Push notifications can be used for user's alert.

Some demerits of the Native Apps:

- Native Apps creation is expensive in comparison to the Mobile Web apps.
- It requires high costs for the maintenance.

How native mobile apps are built

Native mobile apps are built in platform-specific languages. Native app developers use Swift or Objective-C for iOS apps, Java or C++ for Android apps and C# for Windows Phone apps. Yes, that means if you want your native app to be available for download on Android and iOS devices, you'll need to build multiple versions that, most likely, won't have identical user interfaces.

Hybrid App is the mix of the Native App and Mobile Web App. It can be defined like mobile website content exposition in the application format.

Some merits of the Hybrid Apps:

- More cost effective in comparison to the Native App.
- Easy distribution.
- Embedded the browser.
- Device features.

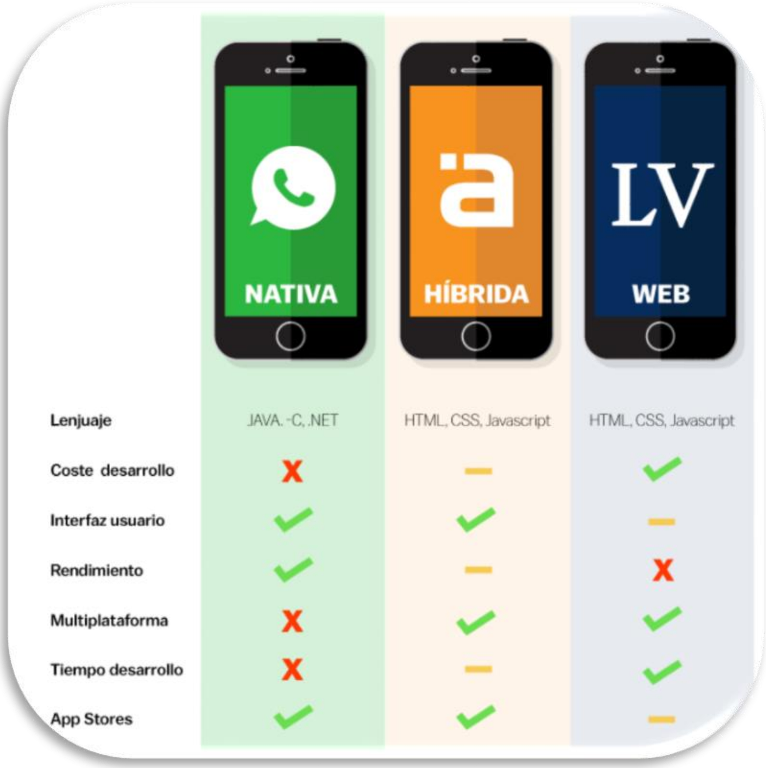
Some demerits of the Hybrid Apps:

- It works not so fast as Native App.
- Graphics are less accustomed to the OS in comparison to Native App.

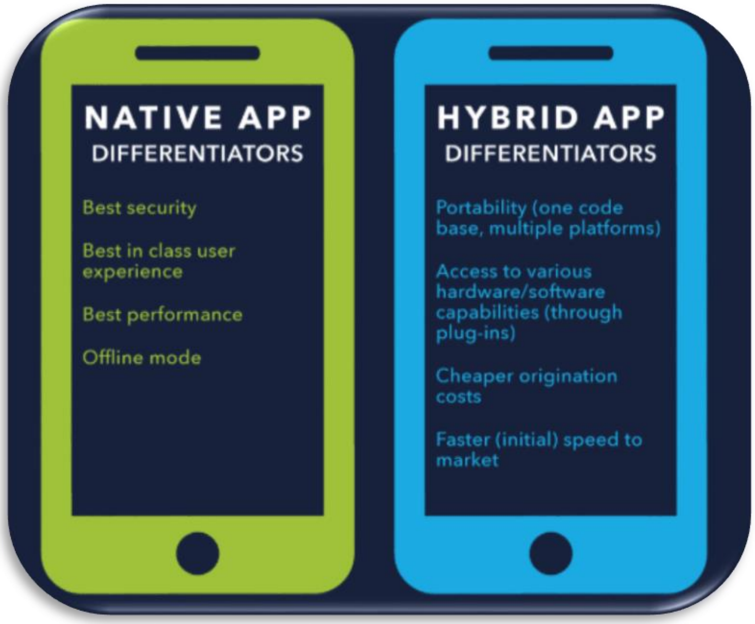
How hybrid mobile apps are built

You guessed it—hybrid apps combine the best of both web and native apps. Technically a kind of mobile app, a hybrid app is installed like a native app, but when you run it, it functions as a web app using a platform's WebView. (WebView is kind of a mini web browser that can be configured to run Fullscreen.)

Hybrid apps are also built specifically for the operating system, and can therefore also access device capabilities like a native app. However, similar to web apps, hybrid apps are written in HTML, CSS, and JavaScript, but are then packaged for different platforms.



	NATIVA	HÍBRIDA	WEB
Lenjuaje	JAVA. -C, .NET	HTML, CSS, Javascript	HTML, CSS, Javascript
Coste desarrollo	X	–	✓
Interfaz usuario	✓	✓	–
Rendimiento	✓	–	X
Multiplataforma	X	✓	✓
Tiempo desarrollo	X	–	✓
App Stores	✓	✓	–



NATIVE APP DIFFERENTIATORS

Best security

Best in class user experience

Best performance

Offline mode

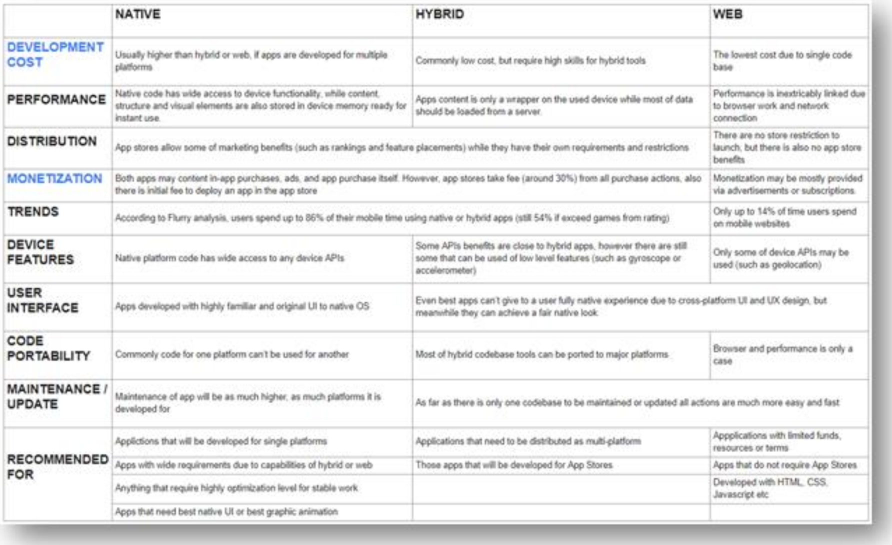
HYBRID APP DIFFERENTIATORS

Portability (one code base, multiple platforms)

Access to various hardware/software capabilities (through plug-ins)

Cheaper origination costs

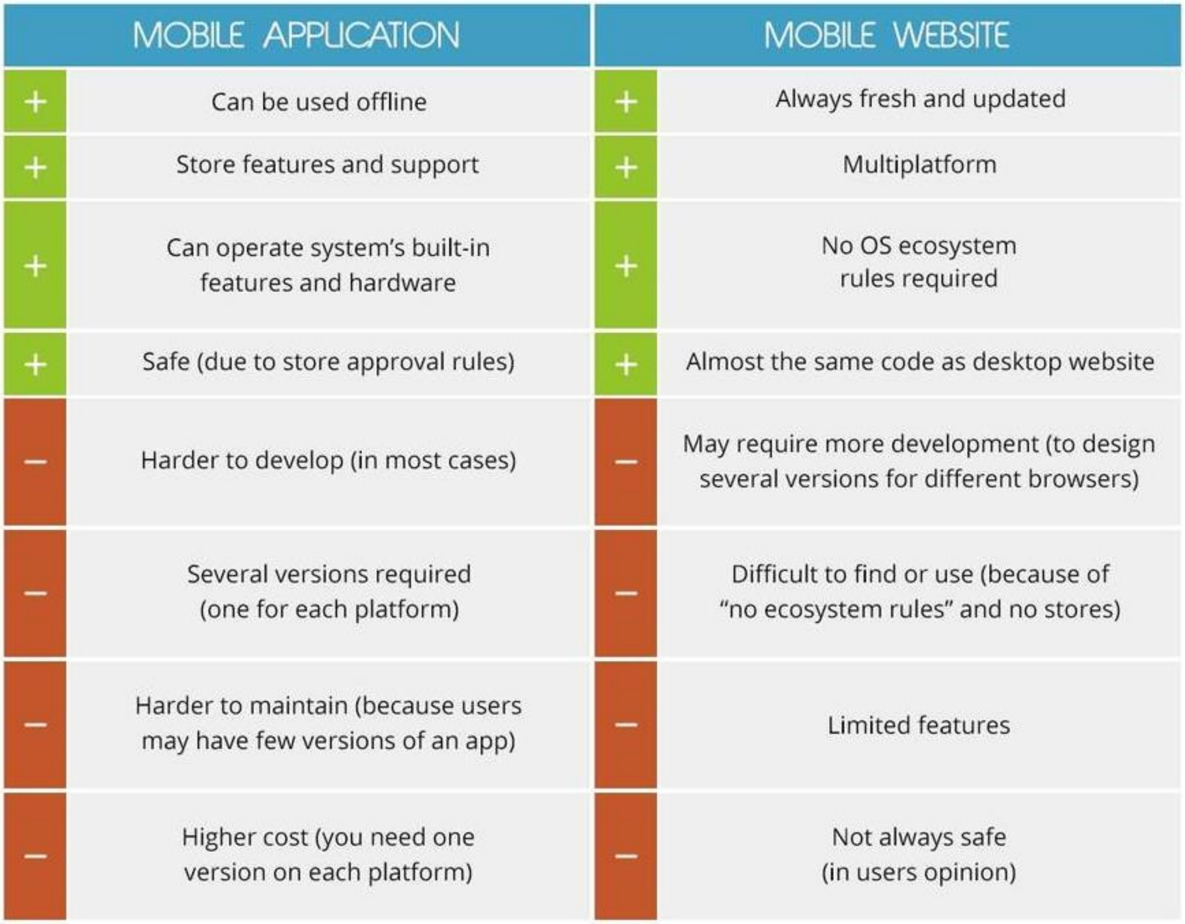
Faster (initial) speed to market



	NATIVE	HYBRID	WEB
DEVELOPMENT COST	Usually higher than hybrid or web, if apps are developed for multiple platforms	Commonly low cost, but require high skills for hybrid tools	The lowest cost due to single code base
PERFORMANCE	Native code has wide access to device functionality, while content, structure and visual elements are also stored in device memory ready for instant use	Apps content is only a wrapper on the used device while most of data should be loaded from a server	Performance is inextricably linked due to browser work and network connection
DISTRIBUTION	App stores allow some of marketing benefits (such as rankings and feature placements) while they have their own requirements and restrictions		There are no store restriction to launch, but there is also no app store benefits
MONETIZATION	Both apps may content in-app purchases, ads, and app purchase itself. However, app stores take fee (around 30%) from all purchase actions, also there is initial fee to deploy an app in the app store		Monetization may be mostly provided via advertisements or subscriptions
TRENDS	According to Flurry analysis, users spend up to 86% of their mobile time using native or hybrid apps (still 54% if exceed games from rating)		Only up to 14% of time users spend on mobile websites
DEVICE FEATURES	Native platform code has wide access to any device APIs	Some APIs benefits are close to hybrid apps, however there are still some that can be used of low level features (such as gyroscope or accelerometer)	Only some of device APIs may be used (such as geolocation)
USER INTERFACE	Apps developed with highly familiar and original UI to native OS	Even best apps can't give to a user fully native experience due to cross-platform UI and UX design, but meanwhile they can achieve a fair native look	
CODE PORTABILITY	Commonly code for one platform can't be used for another	Most of hybrid codebase tools can be ported to major platforms	Browser and performance is only a case
MAINTENANCE / UPDATE	Maintenance of app will be as much higher, as much platforms it is developed for	As far as there is only one codebase to be maintained or updated all actions are much more easy and fast	
RECOMMENDED FOR	Applictions that will be developed for single platforms	Applications that need to be distributed as multi-platform	Appplications with limited funds, resources or terms
	Apps with wide requirements due to capabilities of hybrid or web	Those apps that will be developed for App Stores	Apps that do not require App Stores
	Anything that require highly optimization level for stable work		Developed with HTML, CSS, Javascript etc
	Apps that need best native UI or best graphic animation		

Which type of app should I use?

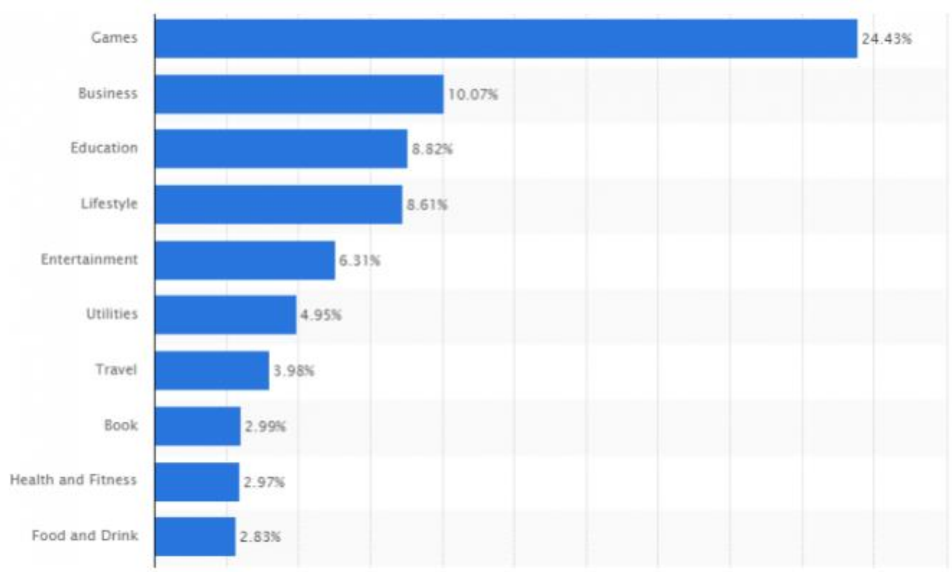
The short answer is it depends. Can you get by with a "fits all" app, or should you look for something that fits your specifications a little better? It's based on the occasion. Think about what features are most important to you, your company goals, and/or the app's main function.



	MOBILE APPLICATION		MOBILE WEBSITE
+	Can be used offline	+	Always fresh and updated
+	Store features and support	+	Multiplatform
+	Can operate system's built-in features and hardware	+	No OS ecosystem rules required
+	Safe (due to store approval rules)	+	Almost the same code as desktop website
–	Harder to develop (in most cases)	–	May require more development (to design several versions for different browsers)
–	Several versions required (one for each platform)	–	Difficult to find or use (because of "no ecosystem rules" and no stores)
–	Harder to maintain (because users may have few versions of an app)	–	Limited features
–	Higher cost (you need one version on each platform)	–	Not always safe (in users opinion)

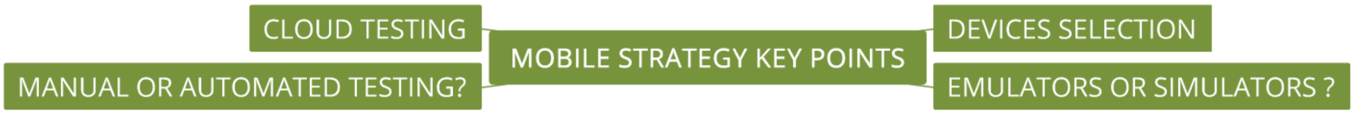
Categories of apps

There are 33 categories of apps in Google Play and 24 categories in Apple's App Store. And there are just 7 app categories that have managed to reach more than 3% of users, according to Statistica research.



Mobile site testing strategy key points

Let's consider the main points and challenges we should face to.



CLOUD TESTING

MANUAL OR AUTOMATED TESTING?

MOBILE STRATEGY KEY POINTS

DEVICES SELECTION

EMULATORS OR SIMULATORS ?

Devices selection

There is no doubt, that the real device is the best decision if you want to test mobile application. Testing on a real device always gives you the highest accuracy of results.

In fact, this is really not easy to choose the most appropriate device. Anyway, here are some actions you should do while selecting device for the mobile testing:

- Make the analysis to define the most popular and used gadgets in the market.
- Choose devices with different OS.
- Choose devices with different screen resolutions.
- Pay attention to the next factors: compatibility, memory size, connectivity etc.

As it was mentioned before you have lots of **advantages** for testing mobile apps on the real devices:

- High accuracy of the testing result.
- Simple bug replication.
- The points like battery drainage, geolocation, push notifications, devices built-in sensors are easy for testing.
- Ability to test incoming interrupts (calls, SMS).
- Ability to test mobile application in the real environment and conditions.
- No false positives.

And also some **disadvantages**:

- A huge number of the often used devices.
- Additional expenses for the maintenance of the devices.
- Limited access to the devices often used in the foreign countries.

As you can see testing on the real devices is the good decision, but also it has some limitations. You should overcome them to make mobile apps testing process real effective.

Emulators or simulators?

There is no difficult to guess, that they are special tools which emulate/simulate the functionality and behavior of the mobile devices.

“Emulator” and “simulator meanings are often confused. Despite theirs almost similar pronunciation, they have no equal meaning.

In fact, an emulator is the original device replacement. Though you can run soft and apps on your gadget, you have no ability to modify them.

The simulator doesn't replicate device's hardware, but you have an ability to set up the similar environment as the original device’s OS.

So, it is better to use mobile simulators to test mobile application. Emulators are more appropriate for the mobile site testing.

Some **advantages** of using the simulators to test mobile application:

- Easy setup.
- Fast working.
- Helps to verify and explore the behavior of your mobile app.
- Cost effective.

Some **disadvantages** of using the simulators to test mobile application:

- Device hardware is not taken into the consideration.
- False positives are possible.
- Incomplete data of the simulation results, which makes some difficulties for the complete analysis of the test results.

Mobile manual and automated testing

Nowadays many specialists support the opinion that manual testing is going to die. Sure, it is not true. Of course, we can not do without test automation, but there also situations when manual testing is preferable.

Some **merits** of the manual mobile application testing:

- It is more cost effective in the short-termed period.
- Manual testing is more flexible.
- Better simulation of user actions.

Some **demerits** of the manual mobile application testing:

- Manual test cases are hard to be reused.
- Less effective of execution certain and constant task.
- Test running process is slow.
- Some kinds of test cases couldn't be executed manually (load testing).

Some **advantages** of the app automation testing:

- Test running process is quite faster.
- Cost effective in the long-termed period.
- Automated test cases are easy to be reused.
- The only decision for some kind of testing (perfomance testing).
- Test results are easy to be shared.

Some **disadvantages** of the app automation testing:

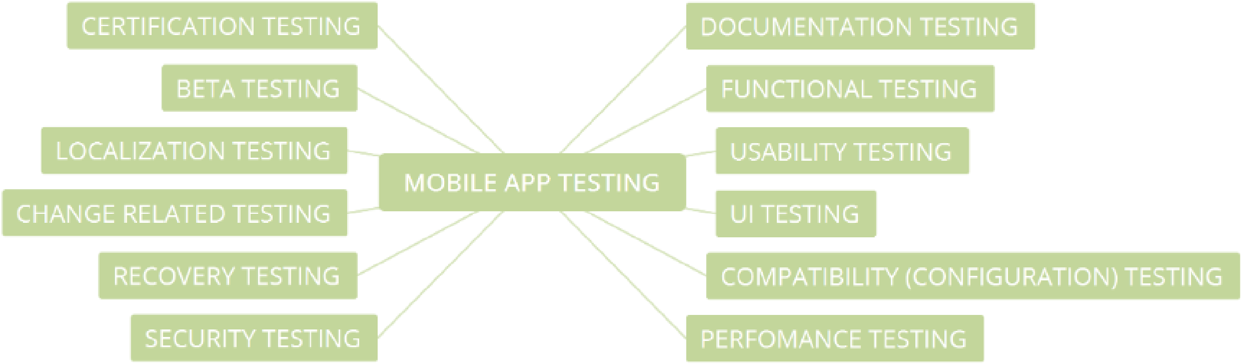
- Automated mobile testing tools have some limitations.
- Time-consuming process.
- Automated testing has less efficiency in the defining user-friendliness or positive customer experience.

As you can see you should make different decisions creating your strategy for the mobile testing. Of course, there are no univocal answers on them.

The combination of different approaches seems to be the optimal way. For example, you can use simulators in the earliest stages of your testing process. But is better to use real devices (physical or cloud-based) in the final stages. Automated testing is preferable for the load and regression testing. But manual mobile testing tools are better to be used for usability and exploratory testing.

Mobile application testing stages

So, let's start to consider the main stages of the mobile app testing process. They more mostly similar to the website testing stages. Mostly, but not quite similar. As you have read before, there are some basic differences between mobile and desktop applications. Therefore, we need to pass some additional stages and make some additional verifications.



1. Documentation Testing

Documentation testing is the necessary preparatory stage of the mobile application testing process.

Actually, testing begins before software development process. Testers get navigational charts, screen layouts, other requirements invisible on the design. These requirements are analyzed for completeness and inconsistency. Contradictions in the requirements must be resolved before the start of development.

Artifacts like Requirements (Specification, PRD), Test Plan, Test Cases, Traceability Matrix are created and analyzed on this stage.

2. Functional testing

Functional testing is aimed to ensure that it is working as per the defined requirements. In simple terms, we check whether the application performs the expected functions, which are usually described in the specification or correspond to the logic of business processes.Pay attention to the next important factors while providing functional testing of your mobile app:

- The application type, which is defined by its business functionality (social networks, banking, education, ordering and delivery of food, tickets, the game industry etc.).
- Target audience (companies, users, educational environment etc.).
- Distribution channels (direct delivery, Google Play, App Store, etc.)

Now, let's consider the main verifications, which should be passed to test mobile application functionality.

Installing and running the application

- The installation of the application should take place without significant errors, if the device meets the system requirements.
- Verify the application automatically starts correctly.
- Ensure the user manual is available.
- Ensure the application's operation during startup/exit meets the basic requirements.

Fields testing

- Verify the required fields work correctly.
- Make sure that mandatory and optional fields are displayed in different ways.

Business functionalities testing

- Verify the declared price and content correspond to the user got information.
- Ensure the user can perform typical operations: buying, adding goods to the cart, ordering goods etc.
- Make sure the application supports payment transactions through payment systems like Visa, Mastercard, Paypal etc.
- Check the recovery of the purchase regardless of the device, but with an account binding.

Interruptions testing

- Incoming and outgoing calls, SMS, and MMS.
- Battery discharge/removal.
- Disconnecting and connecting the network/Wi-Fi.
- Disconnecting and connecting the SD-card.
- Charging the device.

Constant users feedback testing

- Downloading content messages
- Progress bar.
- The appropriate reaction of the buttons on pressing.
- Network access error messages.
- Attempt to delete important information messages.
- Availability and synchronization of sound, vibration, and visual notifications.
- The appearance of a screen (message) at the end of the process (game).

Update testing

- All user data is saved after updates.
- Ensure the update progress is displayed properly.
- Make sure updates are supported by older operating systems.
- Testing various ways of installing updates (Wi-Fi, Bluetooth, USB)

Device resources testing

- Lack of space to install or run the application.
- Memory leaks. Pay attention to windows, with a lot of information, and tasks with long workflow.
- Installing/replacement the app on the SD-card.
- The absence of some functions supported by the application (3G, SD-card, etc.).
- Ensure the installed application does not interfere with the normal operation of other apps and does not consume their memory.

Some other verifications:

- Games concerned verifications: correctness of connecting/disconnecting players, players connection via different networks etc.
- Make sure the information error messages are correct on time and appropriate.
- Verify connection to the analytical tools like Google Analytics.
- Testing the power consumption.
- Verify the necessary options correct work with social networks - **Share, Publish, Navigation.**

Some useful tools to test mobile application functionality: **Appium**, **Selendroid**, **Robotium, Ranorex.**

3. Usability testing

Usability testing is aimed to ensure the convenience of using the application, creates an intuitive interface that conforms to accepted standards. It is performed to create fast and easy-to-use applications. Here are 3 main basic criteria for the apps evaluation:

- Satisfaction
- Efficiency
- Effectiveness

Let's consider the simple checklist to test mobile application usability:

- Make sure that the buttons are of the normal size and placed in one area of the screen
- Verify the app works in multitasking mode, when necessary.
- Check the navigation of the important application modules.
- Ensure the icons and pictures look natural in the app environment.
- Verify the color of the buttons that perform the same function is the same.
- The text should be simple, clear and visible to the user. Short sentences and paragraphs are possible to read.
- Define the optimal font size.
- Ensure correct operation of the Zoom-in and Zoom-out system.
- Verify the context menus are not overloaded.
- Make sure that the application can be terminated by any state and that it resumes operation in the same state.
- Ensure that the application components are synchronized with the user's actions.
- Verify the user can return or cancel the action if he/she pressed the wrong button.
- Verify the speed of response of the element is high enough

Some useful tools to test mobile application usability: **User Zoom**, **Reflector, Loop**[11]**.**

4. UI (User Interface) testing

User Interface (UI) testing is performed to ensure the graphic user interface of your app meets the specifications.

Here are some verifications to test mobile application UI:

- Ensure the compliance with the standards of UI
- Check your app's UI with the standard screen resolutions: 640 × 480, 800 × 600, 1024 × 768, 1280 × 800, 1366 × 768, 1400 × 900, 1680 × 1050.
- Verify responsiveness of applications on different devices.
- Test the main design element: buttons, icons, colors, links, fonts, font sizes, layout, text boxes, text formatting, labels, captions, buttons, lists etc.
- Verify advertising does not overlap application control buttons.
- Ensure the advertising has an accessible closing button.
- Make sure the correct display of various elements on retina and non-retina screens.
- Verify all elements display with portrait and landscape page orientation.

Some useful tools to test mobile application interface: **FitNesse, iMacros, Coded UI, Jubula, LoadUI.**

5. Compatibility (Configuration) testing

Compatibility (Configuration) testing is conducted in o

rder to ensure optimal application performance on different devices - taking into account their size, screen resolution, version, hardware, etc. You should pay attention to the next points:

- OS Configuration
- Browser Configuration
- Database Configuration
- Device Configuration
- Network Configuration

Cross-platform testing helps you to test mobile application in different OS: Windows, iOS, Android, and BlackBerry etc.

Cross-browser testing allows ensuring the correct work of the app in different browser configurations: Mozilla Firefox, Google Chrome, Opera Mini etc.

Database testing is aimed to verify the correct work of your application in different database configurations: Oracle, DB2, MySql, MSSQL Server, Sybase.

Device Configuration testing should take into account such parameters:

- Device type: smartphone, tablet, etc.
- Device configuration: RAM, processor type, screen resolution, battery capacity, etc.

Network configuration testing is performed to ensure the correct work in different network configurations (GSM, TDMA) and standards (2G, 3G, 4G).

Some tips to test your mobile application compatibility:

- Create a coverage matrix (the table in which all possible configurations are entered).
- Prioritize configurations.
- Check each configuration, step by step, in accordance with the set priorities.

Some useful tools to test mobile application perfomance compatibility: **BrowserStack, CrossBrowserTesting by Smart Bear, Litmus, Browsera, Rational Clearcase by IBM, Ghostlab.**

6. Perfomance testing

Performance testing is a set of types of testing, the purpose of which is to determine the operability, stability, resource consumption and other attributes of application quality under different usage scenarios and loads.

The main aims of the performance testing:

- Checking the response time of the application to various types of requests, in order to make sure that the application is working according to the requirements for the normal user load. **(Load testing).**
- Testing the working capacity of the application at loads exceeding the user's several times. **(Stress testing).**
- Examine the operability of the application for long time work, under normal load. **(Stability testing).**
- Check work in the conditions of the "expanded" database, under the normal time. **(Volume testing).**
- Determine the number of users who can simultaneously work with the application. **(Concurrency testing).**

Some verifications for performance testing your mobile app:

- Determine whether the application is running the same under different network conditions.
- Find various application and infrastructure bottlenecks that reduce application performance.
- Evaluate the ability of the app to cope with planned load volumes.
- Verify the response time of the application meets the requirements.
- Check the application stability under conditions of a hard user load.
- Ensure the performance of the application if it works under conditions of a non-permanent connection to the Internet.
- Make sure the existing client-server configuration provides optimal performance.

Some useful tools to test mobile application perfomance: **NeoLoad by Neotys**, **Apteligent (formerly Crittercism)**, **New Relic.**

7. Security testing

Security testing is aimed to check the security of the system, as well as to analyze the risks associated with providing a holistic approach to application protection, hackers, viruses, unauthorized access to sensitive data.

Some verifications you have to pass to test mobile application security:

- Ensure the data of users of the application **(logins, passwords, bank card numbers)** are protected from network attacks of automated systems and cannot be found by selection.
- Verify the application security system requires a strong password and does not allow the attacker to seize the passwords of other users.
- Make sure that the application does not give access to sensitive content or functionality without proper authentication.
- Protect the application against attacks of the SQL injection type.
- Protect the application and the network from DoS Attacks.
- Protect the application from malicious attacks on clients.
- Protect the system from malicious implementations when the program is running.
- Provide session management to protect information from unauthorized users.
- Prevent possible malicious consequences of file caching.
- Examine user files and prevent their possible harmful effects.
- Analyze the interaction of system files, identify and correct vulnerabilities.
- Prevent possible malicious actions of cookies.

Some useful tools to test mobile application security: Retina CS Community, OWASP Zed Attack Proxy, **Veracode,** Google Nogotofail, and SQL Map.

8. Recovery testing

Recovery test verifies the app under test in terms of its ability to withstand and successfully recover from possible failures caused by software errors, hardware failures, or communication problems.

Here is the list of the verifications for the recovery testing:

- Verify the effective recovery of the application after unforeseen crash scenarios.
- Ensure the process of data recovery after a break in the connection.
- Test the recovery after a system failure and a transaction failure.
- Verify the ability of the application to process transactions in the event of a power failure (low battery, incorrect application shutdown etc.).

9. Localization testing

Localization testing allows you to test mobile application adaptation for a specific target audience in accordance with its cultural specifics.

Some verifications for the localization testing:

- Determine languages supported by the application.
- Ensure the correctness of the translation.
- Verify the correctness of the translation in accordance with the theme of the application
- Check the date formats.
- Check the delimiters in numbers.

Of course, the native speakers are preferred to perform localization testing of the mobile app.

Ubertesters, **eggPlant** can be useful to test mobile application localization.

10. Change related testing

So, you passed all mentioned stages and found some bugs. Therefore, some changes have been made to the code of your app.

The key goals of the change related testing:

- Verify your team has successfully fixed all detected bugs (**Re-testing or Confirmation testing**). Put it simply, the test cases that originally detected the bugs are run again. And this time they should be passed with no bugs.
- Verify the new changes did not lead to the appearance of new bugs. (**Regression testing**). Actually, providing regression testing, you should pass not only test cases with detected bugs, but also test cases checking all functionalities of your app.

Some useful tools for change related testing of your app: **Appium**, **Robotium, Ranorex.**

11. Beta testing

Finally, you have the prerelease full functionality version of your mobile app. It would be better to evaluate the possibilities and stability of the program in terms of its future users.

Beta testing is the stage of debugging and checking the beta version of the program. Its main purpose is identifying the maximum number of errors in its work for their subsequent elimination before the final release of the app to the market.

People who have experience with working with similar type apps, better yet, with the previous version of the application are chosen to the role of beta testers.

You should pay attention to the next factors before providing beta testing of your mobile app:

- A number of testing participants.
- Testing duration.
- Shipping
- Demographic coverage
- Testing costs.

Though you need to spend some money for beta testing, it could be a good investment in the quality of your mobile app.

Some popular platforms for beta testing of the mobile apps: **HockeyApp**, **Ubertesters**, **TestFlight** .

12. Certification testing

There are certain rules for organizing an installation file **(.apk)** and rules for applications design for each application store. Certification testing verifies the app meets the requirements of the most popular stores like Google Play, the App Store, and Windows Phone.

Let's consider the main criteria for application compliance with standards, licensing agreements and terms of use.

Android:

- The installation file for the application (.apk) matches with **Program Policies** .
- The application meets the requirements of the **UIG** .
- There are no viruses in the app. Android market semi-automatically checks the application for viruses and could block you account if detect them.
- You should follow the order of version control in the case of publishing an updated version of your app.

iOS:

- The application meets the requirements of the **Human Interface Guidelines**.
- The application must have a unique name.
- You need to provide a link for feedback from the developer.
- The application should be put to the determined particular category.
- App Store test the app for compatibility.
- App doesn't contains prohibited materials, unforeseen delays in work or repetition of existing functions.

Windows Phone

- The application meets the requirements of the **App certification requirements**.
- Clear description of the hardware and network requirements.
- The functions mentioned in the description or shown in the screenshots are fully realized
- Option to control auto-playable sound is required.

Tips to test mobile application

Let's systematize our knowledge and try to determine the main tips for mobile application testing.

1. Learn the app you are going to test.
2. Remember the differences between desktop and mobile apps.
3. Take into account the operating system and hardware specifics
4. Use real devices when it is possible.
5. Don't Try to Find the "Swiss Army Knife" of Testing. Use the tools you are familiar with.
6. Use the advantages of the cloud mobile testing.
7. Confirm your findings with screenshots, logs and videos.
8. Provide your mobile app testing both for portrait and landscape screen mode.
9. Use the development menu options for iOS and Android.
10. Do not neglect (but do not abuse) emulators and simulators for testing.
11. Verify the performance of your app.

12. Don’t automate everything
13. Get real users to test your app
14. Release the time to work out more complex, unconventional test scenarios (f.e. use test "monkeys").
15. Consider the human factor