Unit 5 Testing and Market Distribution 7 Lecture

Testing and debugging mobile application, Versioning mobile apps, signing mobile apps, Packaging mobile apps, Distributing apps on market place. Designing an App using various services provided by android, Application promotion strategies using Google analytics.

https://www.softwaretestinghelp.com/beginners-guide-to-mobile-application-testing/

Types of Mobile Testing

There are broadly 2 kinds of testing that take place on mobile devices:

#1. Hardware testing:

The device including the internal processors, internal hardware, screen sizes, resolution, space or memory, camera, radio, Bluetooth, WIFI etc. This is sometimes referred to as, simple "Mobile Testing".

#2. Software or Application testing:

The applications that work on mobile devices and their functionality are tested. It is called the "Mobile Application Testing" to differentiate it from the earlier method. Even in mobile applications, there are few basic differences that are important to understanding:

a) Native apps: A native application is created for use on a platform like mobile and tablets.
b) Mobile web apps are server-side apps to access website/s on mobile using different browsers like Chrome, Firefox by connecting to a mobile network or wireless network like WIFI.
c) Hybrid apps are combinations of native app and web app. They run on devices or offline and are written using web technologies like HTML5 and CSS.

There are few basic differences that set these apart:

- Native apps have single platform affinity while mobile web apps have the cross-platform affinity.
- Native apps are written in platforms like SDKs while Mobile web apps are written with web technologies like HTML, CSS, asp.net, Java, PHP.
- For a native app, installation is required but for mobile web apps, no installation is required.
- A native app can be updated from the play store or app store while mobile web apps are centralized updates.
- Many native apps don't require an Internet connection but for mobile web apps, it's a must.
- Native app works faster when compared to mobile web apps.
- Native apps are installed from app stores like Google play store or app store where mobile web are websites and are only accessible through the Internet.

The significance of Mobile Application Testing

Testing applications on mobile devices is more challenging than testing web apps on the desktop due to

- **Different range of mobile devices** with different screen sizes and hardware configurations like a hard keypad, virtual keypad (touch screen) and trackball etc.
- **Wide varieties of mobile devices** like HTC, Samsung, Apple and Nokia.
- **Different mobile operating systems** like Android, Symbian, Windows, Blackberry and IOS.
- **Different versions of operation system** like iOS 5.x, iOS 6.x, BB5.x, BB6.x etc.
- **Different mobile network operators** like GSM and CDMA.
- Frequent updates – (like Android- 4.2, 4.3, 4.4, iOS-5.x, 6.x) – with each update a new testing cycle is recommended to make sure no application functionality is impacted.

As with any application, Mobile application testing is also very important, as the clientele is usually in millions for a certain product – and a product with bugs is never appreciated. It often results in monetary losses, legal issue, and irreparable brand image damage.

Basic Difference Between Mobile and Desktop Application Testing:

Few obvious aspects that set mobile app testing apart from the desktop testing

- On the desktop, the application is tested on a central processing unit. On a mobile device, the application is tested on handsets like Samsung, Nokia, Apple, and HTC.
- Mobile device screen size is smaller than a desktop.
- Mobile devices have less memory than a desktop.
- Mobiles use network connections like 2G, 3G, 4G or WIFI where desktop use broadband or dial-up connections.
- The automation tool used for desktop application testing might not work on mobile applications.

Types of Mobile App Testing:

To address all the above technical aspects, the following types of testing are performed on Mobile applications.

- **Usability testing**– To make sure that the mobile app is easy to use and provides a satisfactory user experience to the customers
- **Compatibility testing**– Testing of the application in different mobiles devices, browsers, screen sizes and OS versions according to the requirements.
- **Interface testing**– Testing of menu options, buttons, bookmarks, history, settings, and navigation flow of the application.
- **Services testing**– Testing the services of the application online and offline.
- **Low-level resource testing**: Testing of memory usage, auto-deletion of temporary files, local database growing issues known as low-level resource testing.
- **Performance testing**– Testing the performance of the application by changing the connection from 2G, 3G to WIFI, sharing the documents, battery consumption, etc.
- **Operational testing**– Testing of backups and recovery plan if a battery goes down, or data loss while upgrading the application from a store.
- **Installation tests–** Validation of the application by installing /uninstalling it on the devices.
- **Security Testing**– Testing an application to validate if the information system protects data or not.

Mobile Application Testing Strategy

The Test strategy should make sure that all the quality and performance guidelines are met. A few pointers in this area:

1) Selection of the devices – Analyze the market and choose the devices that are widely used. (This decision mostly relies on the clients. The client or the app builders consider the popularity factor of certain devices as well as the marketing needs for the application to decide what handsets to use for testing.)

2) Emulators – The use of these is extremely useful in the initial stages of development, as they allow quick and efficient checking of the app. The emulator is a system that runs software from one environment to another environment without changing the software itself. It duplicates the features and works on the real system.

Types of Mobile Emulators

- Device Emulator- provided by device manufacturers
- Browser Emulator- simulates mobile browser environments.
- Operating systems Emulator- Apple provides emulators for iPhones, Microsoft for Windows phones and Google Android phones

Testing Types for Mobile Apps

Following types of testing are performed in order to certify an Android application:

1) Functional Testing:

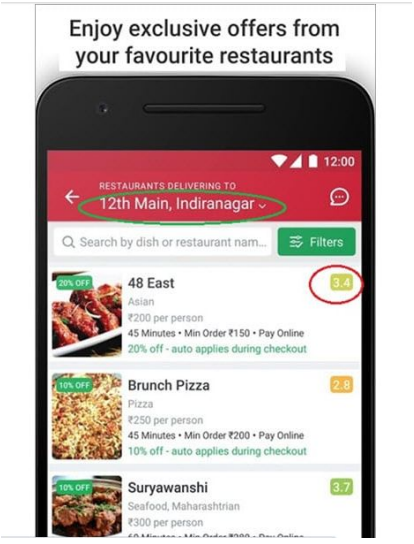
Testing is normally achieved by user interface initiated test flows. Not just the flow of a use case is tested, but the various business rules are also tested. Testing is done by certifying the requirements. i.e. whether the application is working based on the requirements.

Let us take the **Example** of Zomato app.

When you launch the app, based on the location, the list of restaurants shows up. Now as a QA, you have to test the listing of restaurants.

This is functionality testing and here you will test functionalities like:

- Verify whether the restaurant cards are shown w.r.t the location.
- Verify whether the restaurants shown are in the given range (km) of the location.
- Verify whether the review count for each card is in sync with the reviews given.
- Check whether adding a new restaurant for a location reflects in the list.
- Verify whether the restaurants are listed based on the filtering rule of Zomato etc.



2) Android UI Testing:

This is an user-centric testing of the application. In this test phase, items such as visibility of text in various screens of the app, interactive messages, alignment of data, the look and feel of the app for different screens, size of fields etc are tested under this.

For the same above Example of Zomato, the UI tests would be:

- Verify whether the cards are shown properly aligned with the mobile screen size.
- Verify whether the image of the restaurant is shown as expected.
- Verify whether the card details, the rating of the restaurant, cuisine type, address etc, are shown properly aligned.
- Verify whether the scrolling of the list is glitch free and the cards are not shown misaligned when a long list of cards is scrolled etc.

3) Compatibility Testing:

This testing is done mostly in the form of two matrices of OS Vs app and Device model Vs app. Usually, a list of supported OS (and sometimes devices) is provided by the product owner or customer.

The need for this testing is:

- When an OS like Android is taken into account, 7 base versions are found neglecting the number of patch releases/EPs.
- *Can you guess the types of Android devices currently working all over the globe? 1000? 2000? Wrong!* The correct answer is 24000. There are 24000 types of unique Android devices that are present and active in the world.
- With variations such as these, compatibility testing plays a vital role in certifying an android app.
- We not only need to validate the OS and type of the device but also validate few other features which fall into compatibility testing. **The features include:**
  - Screen size
  - Screen resolution
  - Network connectivity

4) Interface Testing:

In other words, it is also termed as Integration testing. This testing is done after all the modules of the app are completely developed, tested individually and all the bugs are fixed verified.

Interface testing includes tests like a complete end to end testing of the app, interaction with other apps like Maps, social apps etc, usage of Microphone to enter text, usage of Camera to scan a barcode or to take a picture etc.

Again considering Zomato, the integration tests would be like:

- Verify whether the user is able to book a table for a restaurant.
- Verify whether the user is able to view the menu and order food online.
- Verify whether the user is able to avail a PayTM coupon while ordering food.
- Verify whether the user is able to view the location of the restaurant on Google Maps.
- Verify whether the user is able to open the phone dialer and call the restaurant etc.

5) Network Testing:

The key features of Network Testing include:

- The app should talk to the intermediate service so as to carry out the process.
- During this testing, request/response to/from the service is tested for various conditions.
- This test is mainly done to verify the response time in which the activity is performed like refreshing data after sync or loading data after login etc.
- This is done for both strong wifi connection and the mobile data network.
- This is an in-house testing.

6) Performance Testing:

Performance of the application under some peculiar conditions are checked.

Those conditions include:

- Low memory in the device.
- The battery in extremely at a low level.
- Poor/Bad network reception.

Performance is basically tested from 2 ends, application end, and the application server end

7) Installation Testing:

There are two types of apps on an Android device i.e, Pre-installed applications and the applications which are installed later by the user.

For both of the above, installation testing needs to be carried out. This is to ensure smooth installation of the application without ending up in errors, partial installation etc.

Upgrade and uninstallation testing are carried out as part of Installation testing.

8) Security Testing:

Privacy and security are the 2 major requirements of an app. However, in Banking, healthcare, this becomes the primary requirement.

Testing of the data flow for encryption and decryption mechanism is to be tested in this phase. Access to stored data is also tested in this phase.

9) Field Testing:

Field testing is done specifically for the mobile data network and not in-house but by going out and using the app as a normal user. This testing is done 'only' after the whole app is developed, tested and regressed (for bugs and test cases).

It is basically done to verify the behavior of the app when the phone has a 2G or 3G connection. Field testing verifies if the app is crashing under slow network connection or if it is taking too long to load the information.

10) Interrupt Testing:

This type of testing is also known as Offline Scenario Verification. Conditions where the communication breaks in the middle are called as offline conditions.

Some of the conditions where interruptions of a network can be tested are as follows:

- Data cable removal during data transfer process.
- Network outage during the transaction posting phase.
- Network recovery after an outage.
- Battery removal or Power On/Off when it is in the transactional phase.

Best Practices in Android App Testing

There are certain factors to be considered while putting a strategy for mobile app testing.

They are:

1) Device Selection:

- This is one of the most critical steps before starting the android application testing process.
- Decide which devices are to be taken into account for the testing process.
- The selection is to be done so as to maximize the number of target customers.
- Factors such as OS version, Screen resolutions and Form factors [Tablet or smart phones] play a vital role in the selection phase.
- If required, even the help of Emulators can be taken into account.
- But, Emulators should not replace the physical device testing process.
- Device emulators are cost effective and they come in handy during the initial development phase.
- But, to test the real-life scenarios, physical devices are the must. Both emulators and physical devices are to be used in a balanced manner for an optimized result.

2) Beta Testing of the Application:

- Beta testing is very effective in testing with the real-world users, real devices, actual networks and applications installed in a wide geography.
- This gives a clear picture of the network density, network variations [Wi-Fi, 4G, 3G, and 2G] and the impact on the application.
- Beta testing in the real world is one of its kind and cannot be replicated in a controlled environment.

3) Connectivity:

- Normally, Android applications are connected to the internet for various requirements.
- The connectivity on different devices plays a key role in putting up the strategy.
- Mostly the connectivity is controlled by simulation software which helps in regulating the network speed, latency, and limited connectivity while testing.
- It is said that testing under real network connections is always advisable for real-time result/data.

4) Manual or Automated Testing:

- Though automation testing takes ample amount of time for the first run, it comes in handy when the testing has to be repeated. This also reduces the overall time span of testing during the different development stages.
- Android Automation should be clubbed with Manual testing when the regression testing repetition is high in the application development phase, compatibility testing has to be done for the same application on different OS versions, backward compatibility checkpoints etc.

Android Testing Framework

There are a handful of Android testing frameworks that are available.

In this tutorial, we are going to discuss 3 different types of most commonly used frameworks:

- Robotium test framework
- Robo-electric test framework
- Appium test framework

1) Robotium Test Framework:

- This framework is used to write sophisticated and robust black box test cases for Android applications.
- It supports both native as well as hybrid clients.
- Functions, system test cases, and User acceptance test cases can be written using this framework.
- Robotium supports Android 1.6 and above and also support for Dialogs, Menus, Activities etc.
- This framework handles multiple Android activities automatically.
- A handful of methods are given as a part of Robotium for interacting with different graphical components of the Android application. ***Some of them are as below:***
  - goBack();
  - getButton();
  - isRadioButtonChecked();
  - searchText("User");
  - click on button("Logout");

2) Robo-electric Test Framework:

- This framework helps in testing Android applications on the JVM based on the JUnit4 framework.
- It uses Android API.
- This helps in writing test cases and running them on the JVM.
- Under this, all the classes are replaced by something called shadow objects.
- Whenever a method is implemented, Robo-electric internally sends the call to the shadow object.
- Based on the implementation, if a method is implemented by shadow object then a value is returned. Else, NULL is returned.
- Because of the shadow objects and JVM, the execution becomes faster.

3) Appium Test Framework:

- This framework works for native, hybrid and mobile –web apps for Android devices.
- Appium is free to use utility.
- Single API works for both Android as well as the iOS platform. This is one of the frameworks which supports cross-platform testing.
- It uses Selenium Web driver to interact with the Android application.
- Appium supports script writing using a lot of programming languages such as Java, C#, Python, PHP, Ruby etc.

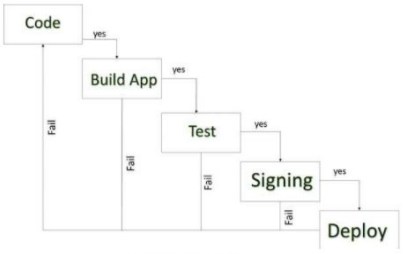
Conclusion

Android application automation can be clubbed with manual as per the need. However, if Androind Automation testing is added to the strategy, selecting the correct tool is a big task. While selecting the automation tool factors like Multi-platform support, Test workflow, Price of the tool, service/support etc. should be taken into account.

There are several challenges involved in Android application testing. There are different factors to be taken into account before the Android testing process can really be implemented but once done this becomes a very interesting task.

Publishing Android Applications:

Android application publishing is a process that makes your Android applications available to users. Infect, publishing is the last phase of the Android application development process.



Android development life cycle

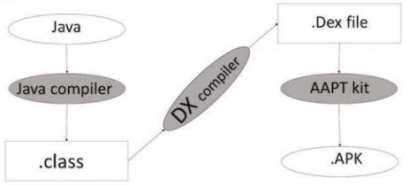
Once you developed and fully tested your Android Application, you can start selling or distributing free using Google Play (A famous Android marketplace). You can also release your applications by sending them directly to users or by letting users download them from your own website.

You can check a detailed publishing process at Android official website, but this tutorial will take you through simple steps to launch your application on Google Play. Here is a simplified check list which will help you in launching your Android application −

Step	Activity
1	**Regression Testing** Before you publish your application, you need to make sure that its meeting the basic quality expectations for all Android apps, on all of the devices that you are targeting. So perform all the required testing on different devices including phone and tablets.
2	**Application Rating** When you will publish your application at Google Play, you will have to specify a content rating for your app, which informs Google Play users of its maturity level. Currently available ratings are (a) Everyone (b) Low maturity (c) Medium maturity (d) High maturity.
3	**Targeted Regions** Google Play lets you control what countries and territories where your application will be sold. Accordingly you must take care of setting up time zone, localization or any other specific requirement as per the targeted region.
4	**Application Size** Currently, the maximum size for an APK published on Google Play is 50 MB. If your app exceeds that size, or if you want to offer a secondary download, you can use APK Expansion Files, which Google Play will host for free on its server infrastructure and automatically handle the download to devices.
5	**SDK and Screen Compatibility** It is important to make sure that your app is designed to run properly on the Android platform versions and device screen sizes that you want to target.
6	**Application Pricing** Deciding whether you app will be free or paid is important because, on Google Play, free app's must remain free. If you want to sell your application then you will have to specify its price in different currencies.

7	**Promotional Content** It is a good marketing practice to supply a variety of high-quality graphic assets to showcase your app or brand. After you publish, these appear on your product details page, in store listings and search results, and elsewhere.
8	**Build and Upload release-ready APK** The release-ready APK is what you you will upload to the Developer Console and distribute to users. You can check complete detail on how to create a release-ready version of your app: Preparing for Release .
9	**Finalize Application Detail** Google Play gives you a variety of ways to promote your app and engage with users on your product details page, from colourful graphics, screen shots, and videos to localized descriptions, release details, and links to your other apps. So you can decorate your application page and provide as much as clear crisp detail you can provide.

Export Android Application Process



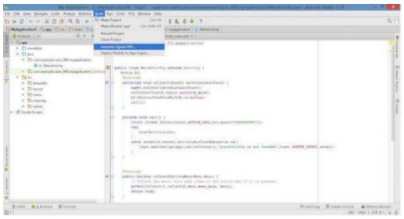
Apk development process

Before exporting the apps, you must some of tools

- **Dx tools**(Dalvik executable tools): It going to convert **.class file** to **.dex file**. it has useful for memory optimization and reduce the boot-up speed time
- **AAPT**(Android assistance packaging tool):it has useful to convert **.Dex file** to**.Apk**
- **APK**(Android packaging kit): The final stage of deployment process is called as .apk.

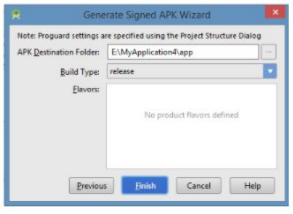
You will need to export your application as an APK (Android Package) file before you upload it Google Play marketplace.

To export an application, just open that application project in Android studio and select **Build → Generate Signed APK** from your Android studio and follow the simple steps to export your application −



Next select, **Generate Signed APK** option as shown in the above screen shot and then click it so that you get following screen where you will choose **Create new keystore** to store your application.

Enter your key store path,key store password,key alias and key password to protect your application and click on **Next** button once again. It will display following screen to let you create an application −

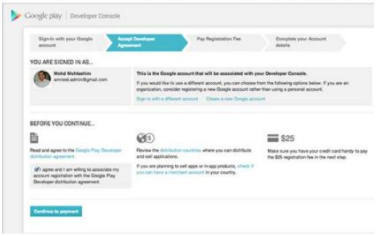


Once you filled up all the information,like app destination,build type and flavours click **finish** button While creating an application it will show as below

Finally, it will generate your Android Application as APK formate File which will be uploaded at Google Play marketplace.

Google Play Registration

The most important step is to register with Google Play using Google Play Marketplace. You can use your existing google ID if you have any otherwise you can create a new Google ID and then register with the marketplace. You will have following screen to accept terms and condition.



You can use **Continue to payment** button to proceed to make a payment of $25 as a registration fee and finally to complete your account detail.

Once you are a registered user at Google Play, you can upload **release-ready APK** for your application and finally you will complete application detail using application detail page as mentioned in step 9 of the above mentioned checklist.

Signing Your App Manually

You do not need Android Studio to sign your app. You can sign your app from the command line using standard tools from the Android SDK and the JDK. To sign an app in release mode from the command line −

- Generate a private key using keytool

```
$ keytool -genkey -v -keystore my-release-key.keystore
-alias alias_name -keyalg RSA -keysize 2048 -validity 10000
```

- Compile your app in release mode to obtain an unsigned APK
- Sign your app with your private key using jarsigner

```
$ jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1
-keystore my-release-key.keystore my_application.apk alias_name
```

- Verify that your APK is signed. For example −

```
$ jarsigner -verify -verbose -certs my_application.apk
```

- Align the final APK package using zipalign.

```
$ zipalign -v 4 your_project_name-unaligned.apk your_project_name.apk
```