Becoming a Pro

IN Mobile Applications Testing

Overview: Mobile APPS

- Categories
- Types
- Distribution/Installation/Logs
- Mobile Test Industry Standards
- Remote Device Access (RDA)
- Emulators
- Simulators
- Troubleshooting Guide
- App Risk Analysis

Mobile APPS: Distribution/Installation/Logs

What is ADB in Android Studio

Android Debug Bridge (adb) is a versatile command line tool that lets you communicate with an emulator instance or connected Android-powered device.

It is a client-server program that includes three components:

A client, which sends commands. The client runs on your development machine. You can invoke a client from a shell by issuing an adb command. Other Android tools such as DDMS also create adb clients.

A daemon, which runs commands on a device. The daemon runs as a background process on each emulator or device instance.

A server, which manages communication between the client and the daemon. The server runs as a background process on your development machine.

Mobile APPS: Distribution/Installation/Logs

What is ADB LOCATS?

Logcat is a command-line tool that dumps a log of system messages, including stack traces when the device throws an error and messages that you have written from your app with the Log class

ANDROID MONITOR includes a logcat Monitor that displays debug messages.

The logcat Monitor displays system messages, such as when a garbage collection occurs, as well as messages that you can add to your app using the LOG class.

It displays messages in real time and also keeps a history so you can view older messages.

Mobile APPS: Distribution/Installation/Logs

What is ADB LOCATS?

To set a LOG LEVEL : in the log level MENU Select the Following Options

Verbose - Show all log messages (the default).

Debug - Show debug log messages that are useful during development only, as well as the message levels lower in this list.

Info - Show expected log messages for regular usage, as well as the message levels lower in this list.

Warn - Show possible issues that are not yet errors, as well as the message levels lower in this list.

Error - Show issues that have caused errors, as well as the message level lower in this list.

Assert - Show issues that the developer expects should never happen.

Mobile APPS: Distribution/Installation/Logs

What is ADB LOCATS?

HOMEWORK : http://adbshell.com/commands/adb-logcat

Some most useful commands

adb logcat *:V ***lowest priority, filter to only show Verbose level***

adb logcat *:D ***filter to only show Debug level***

adb logcat *:I ***filter to only show Info level***

adb logcat *:W ***filter to only show Warning level***

adb logcat *:E ***filter to only show Error level***

adb logcat *:F ***filter to only show Fatal level***

adb logcat *:T ***filter to show steps leading up to errors and warnings***

Mobile APPS: Distribution/Installation: Android .APK

- Through an App Marketplace *(Google Play)*
- by Email *(Android system recognizes the APK and displays an Install Now button in the email message)*
- Through a Website *(host the release-ready APK file on your website and provide a download link)*
- Google Drive
- Android Studio
- Test Fairy TOOL
- *enable allow "Unknown Sources" on the device (Settings > Applications > Unknown Sources*

Mobile APPS: Distribution/Installation: Android .APK

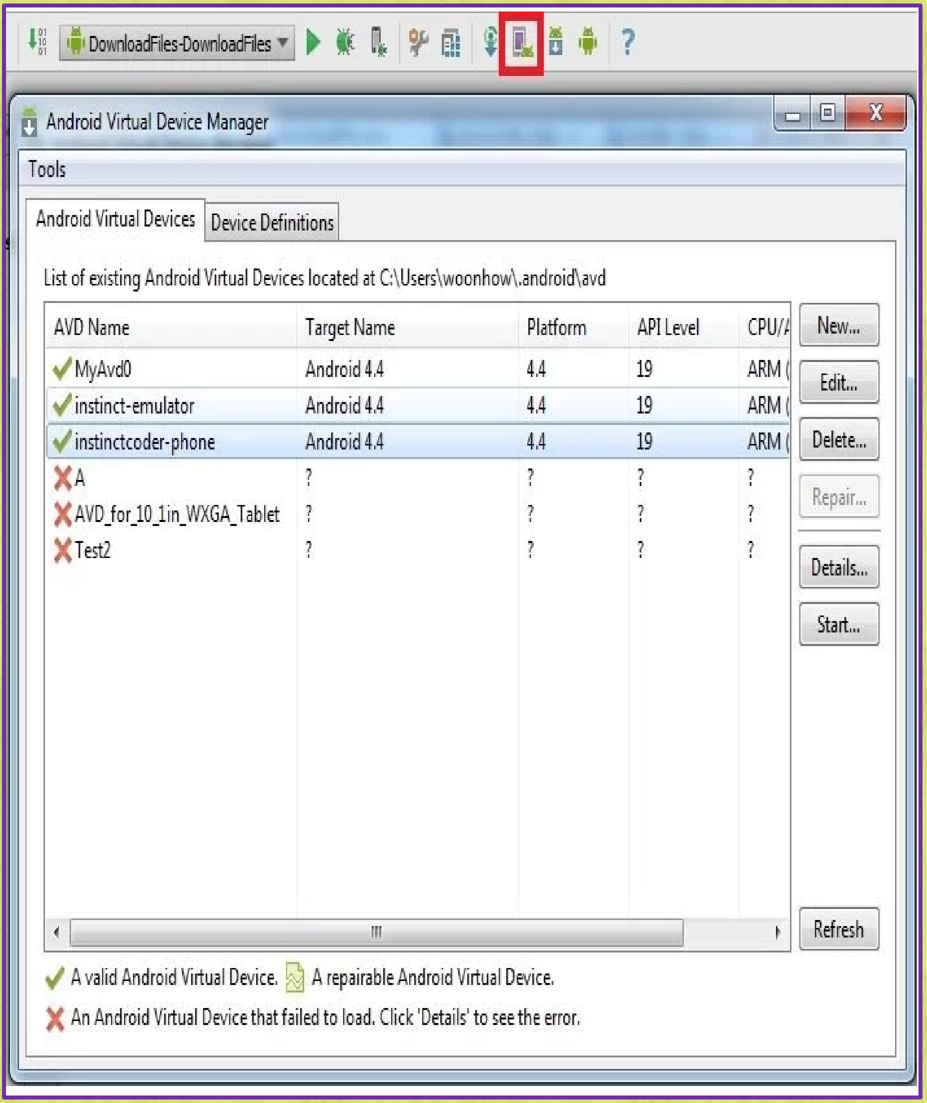
Manually Install APK in Android Studio Emulator

1. Verify the presence of X:\Program Files (x86)\Android \android-studio\sdk\ platform-tools

2. Copy APK file into X:\Program Files (x86)\Android \android-studio\sdk\ platform-tools

3. go to Android Studio > Run Android Virtual Device Manager (AVD) > Start emulator

In this session we will use APK file get from "Candy Crush"



List of existing Android Virtual Devices located at C:\Users\woonhow\.android\avd

AVD Name	Target Name	Platform	API Level	CPU/A
✓ MyAvd0	Android 4.4	4.4	19	ARM (
✓ instinct-emulator	Android 4.4	4.4	19	ARM (
✓ instinctcoder-phone	Android 4.4	4.4	19	ARM (
✗ A	?	?	?	?
✗ AVD_for_10_1in_WXGA_Tablet	?	?	?	?
✗ Test2	?	?	?	?

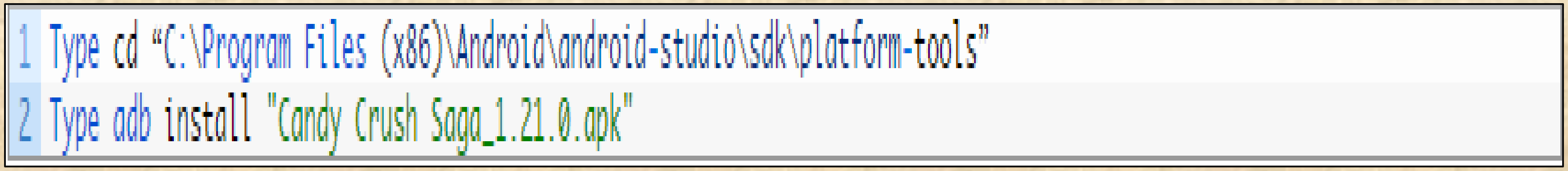
✓ A valid Android Virtual Device. A repairable Android Virtual Device.

✗ An Android Virtual Device that failed to load. Click 'Details' to see the error.

Mobile APPS: Distribution/Installation: Android .APK

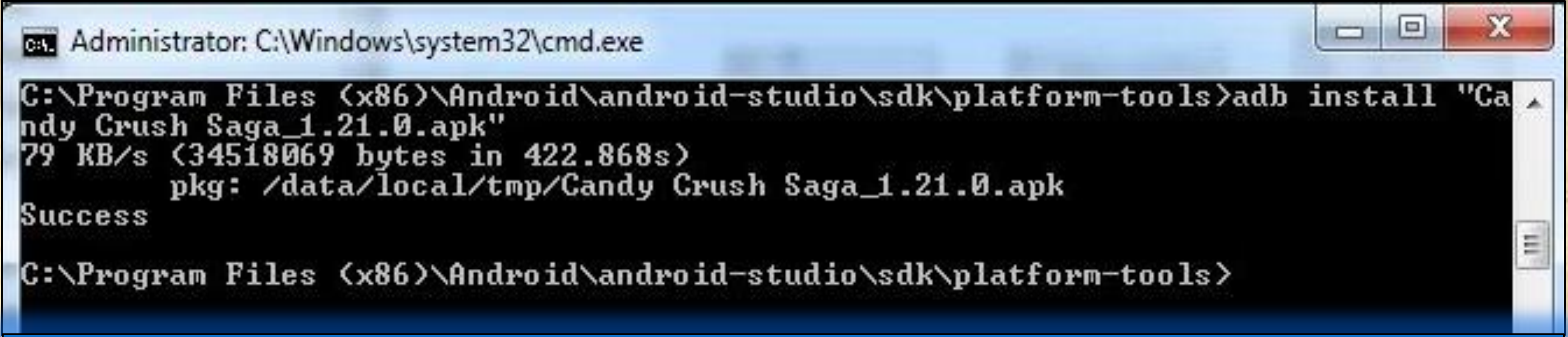
Cont. : Manually Install APK in Android Studio Emulator from Shell

4. Go to **Start > Run > Cmd**



```
Type cd "C:\Program Files (x86)\Android\android-studio\sdk\platform-tools"
Type adb install "Candy Crush Saga_1.21.0.apk"
```

5. After successfully installed, you will see result in command prompt like below



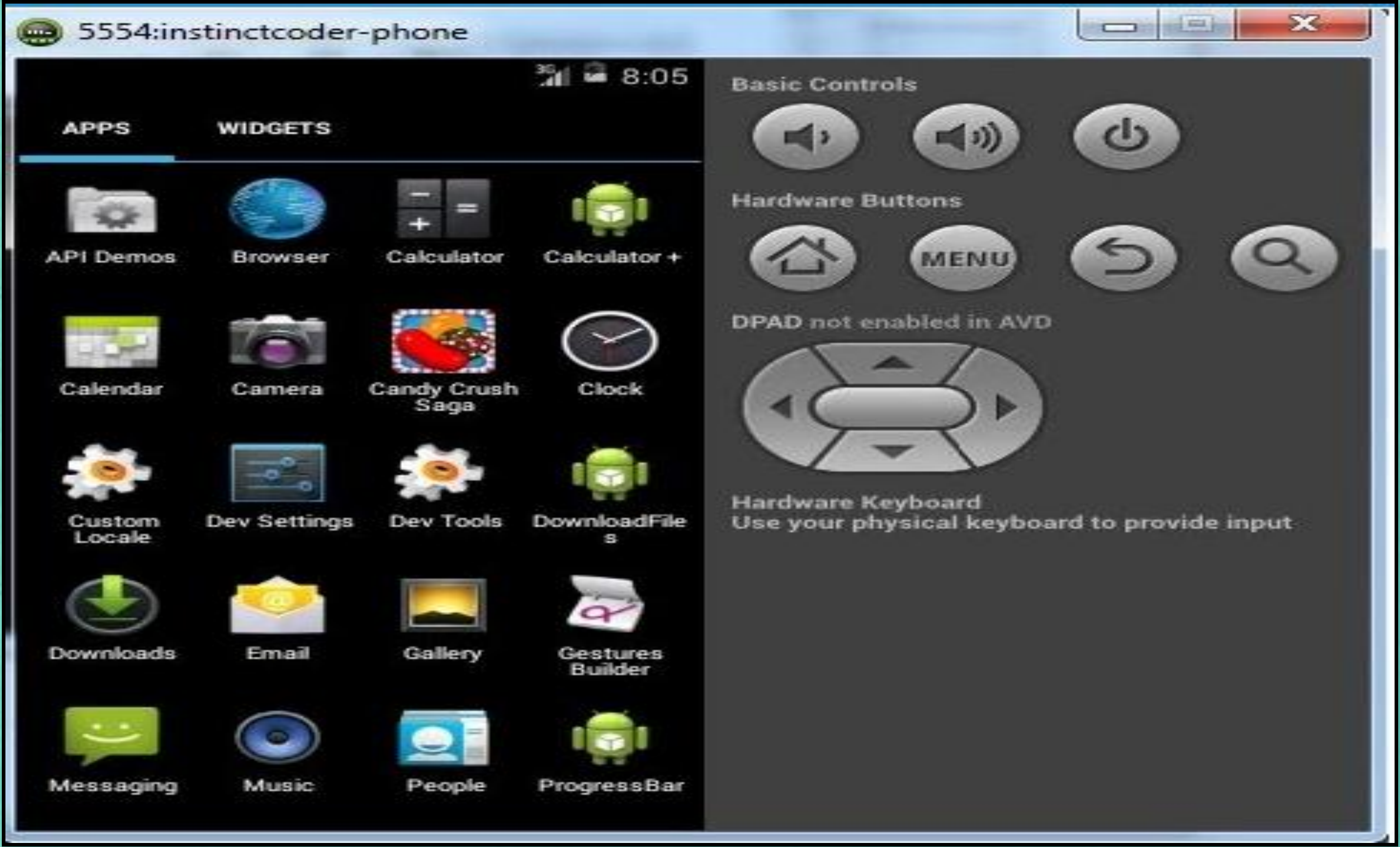
```
C:\Program Files (x86)\Android\android-studio\sdk\platform-tools>adb install "Ca
ndy Crush Saga_1.21.0.apk"
79 KB/s (34518069 bytes in 422.868s)
        pkg: /data/local/tmp/Candy Crush Saga_1.21.0.apk
Success

C:\Program Files (x86)\Android\android-studio\sdk\platform-tools>
```

It will take about 1 minute to install successfully

Mobile APPS: Distribution/Installation: Android .APK

Manually Install APK in Android Studio Emulator

6.Go to the emulator and you will see Candy Crush install in the emulator like below



Mobile APPS: Distribution/Installation: Android .APK

What is TEST FAIRY ?

TestFairy offers some great features for app developers. One of the stand out features is client side Video recording and not just screen shots.

TestFairy provides a video recording of the exact test from the client side, including CPU, Memory, GPS, Network monitoring, logs, crash reports and more.

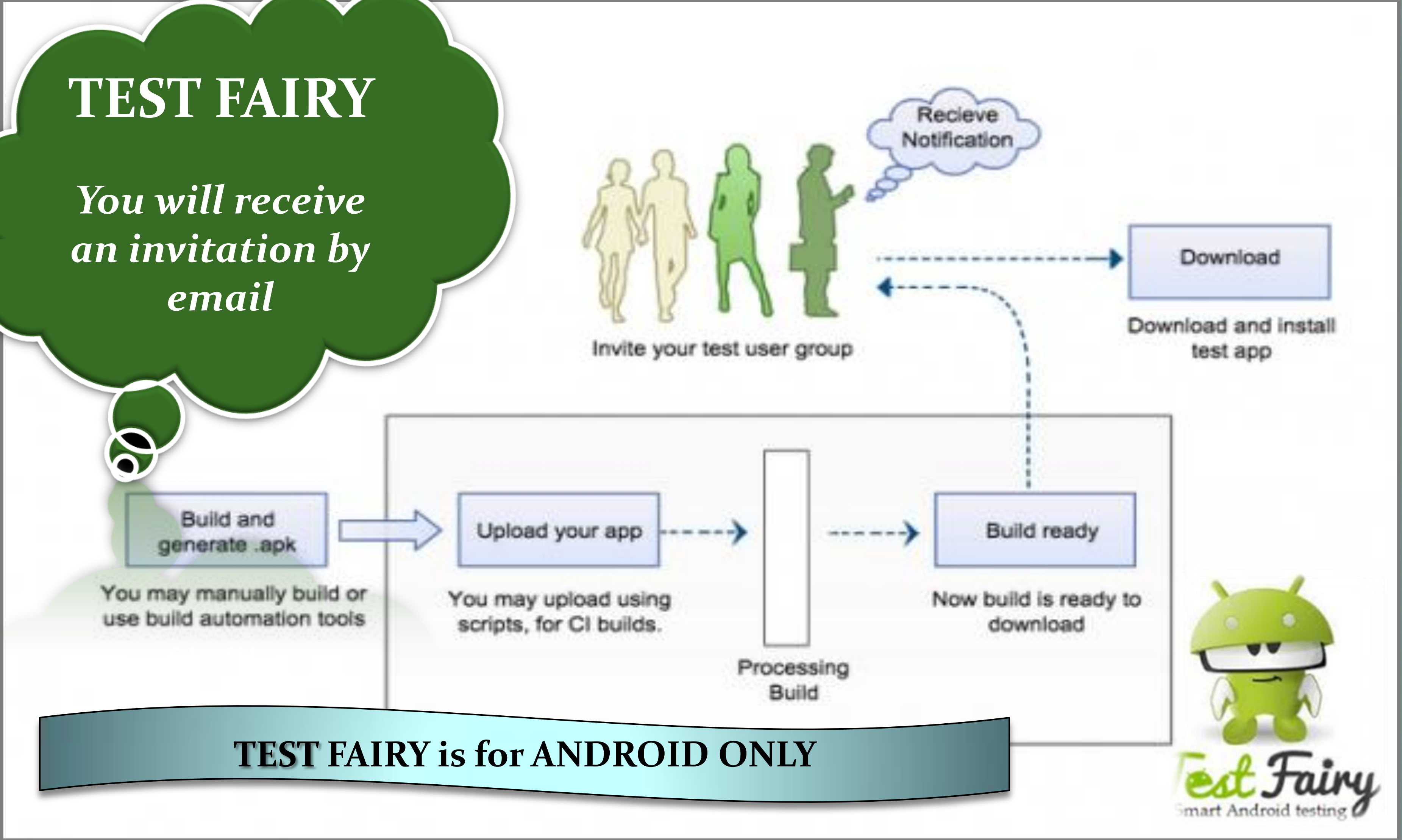
With Android apps you don't need to integrate any SDK or APIs into your app builds. You upload your APK (Android application file) to the TestFairy platform.

TEST FAIRY is for ANDROID ONLY

Mobile APPS: Distribution/Installation: Android .APK

TEST FAIRY

You will receive an invitation by email


Recieve Notification
Invite your test user group
Download
Download and install test app
Build and generate .apk
You may manually build or use build automation tools
Upload your app
You may upload using scripts, for CI builds.
Processing Build
Build ready
Now build is ready to download
Test Fairy
Smart Android testing

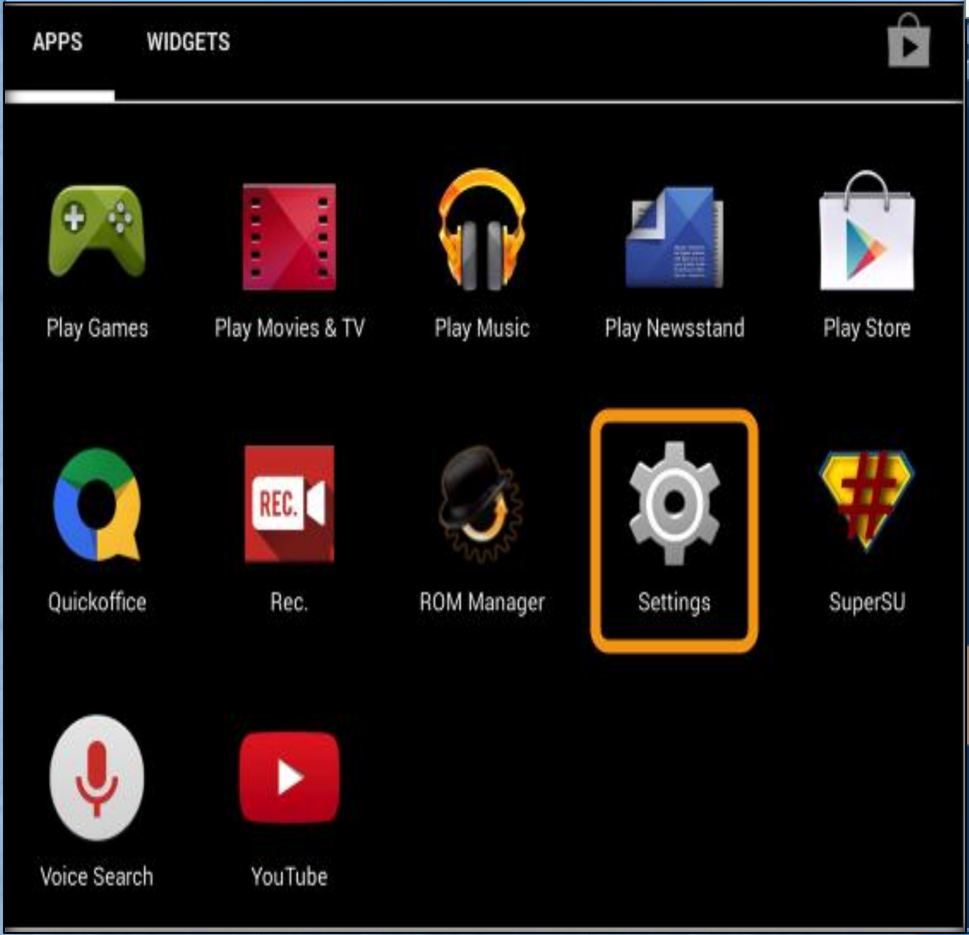
TEST FAIRY is for ANDROID ONLY

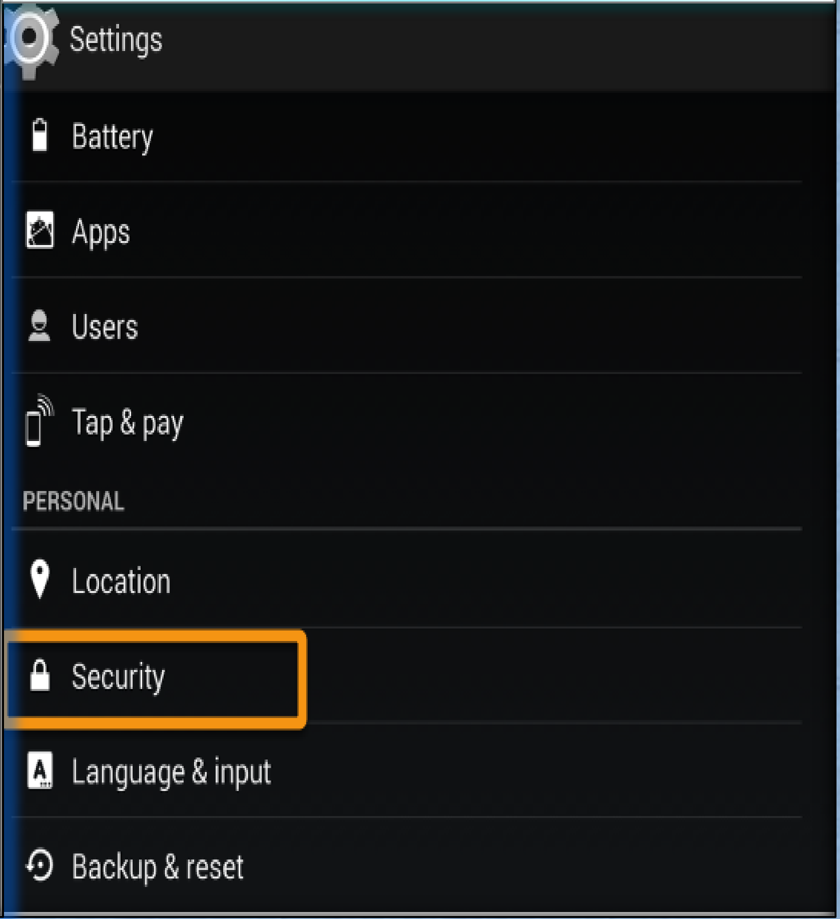
Mobile APPS: Distribution/Installation: Android .APK

Installing THIRD PARTY APK

1. Go to the "Settings" application on your Android device

2. Choose the "Security" option located under the heading "Personal"

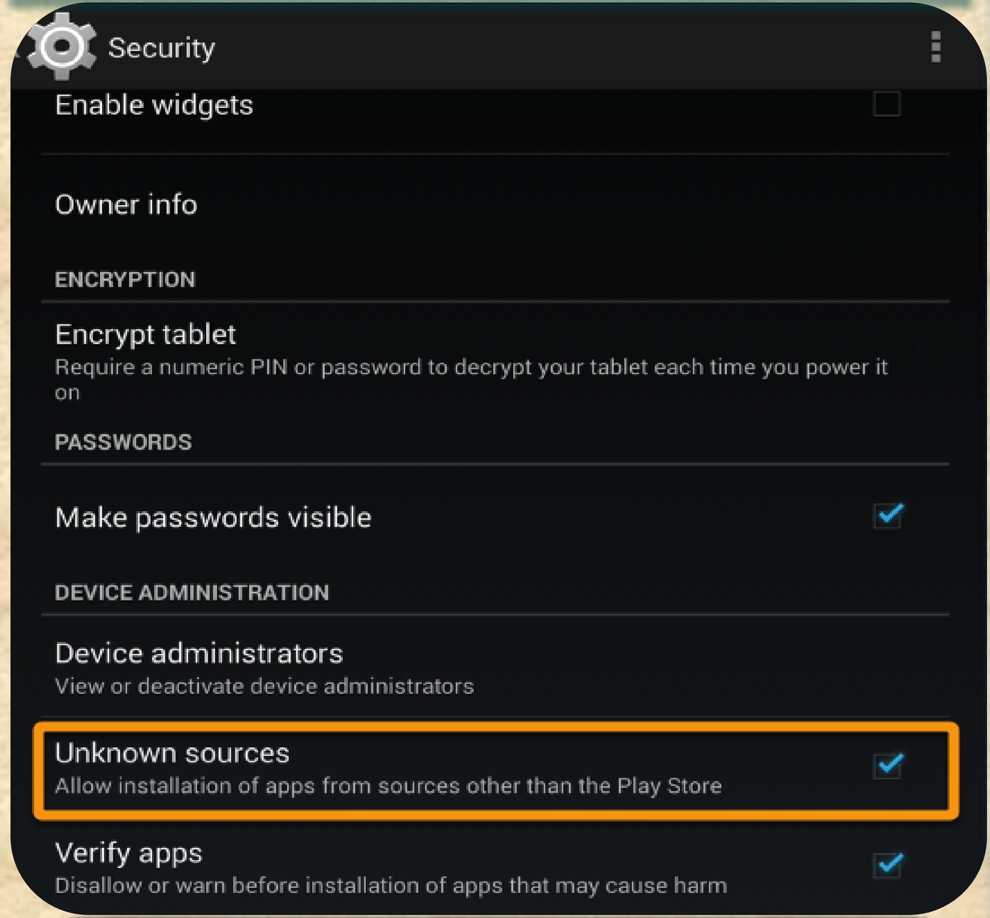




Mobile APPS: Distribution/Installation: Android .APK

Installing THIRD PARTY APK

3. Under "Device Administration" place a checkbox next to the option "Unknown Sources"



Security

Enable widgets ☐

Owner info

ENCRYPTION

Encrypt tablet
Require a numeric PIN or password to decrypt your tablet each time you power it on

PASSWORDS

Make passwords visible ✓

DEVICE ADMINISTRATION

Device administrators
View or deactivate device administrators

Unknown sources ✓
Allow installation of apps from sources other than the Play Store

Verify apps ✓
Disallow or warn before installation of apps that may cause harm

This allows you to install applications on your Android device that are not downloaded directly from the Google Play store

".

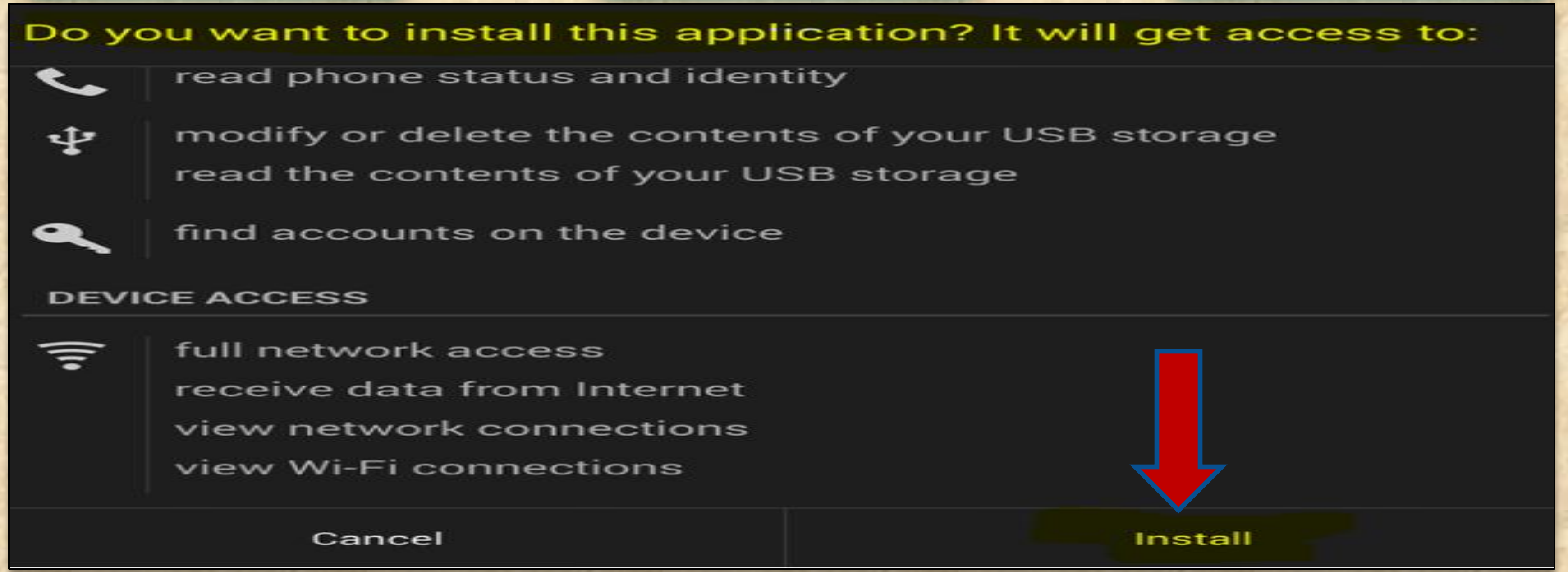
Mobile APPS: Distribution/Installation: Android .APK

Transferring APK to your Device

1. On your computer, attach the ".apk" file to an e-mail and send it to an account that you can access via your Android device.

2. On the Android device, click on the ".apk" attachment in the e-mail in order to download it.

3. Follow the on-screen instructions to install the application.



Do you want to install this application? It will get access to:

read phone status and identity

modify or delete the contents of your USB storage
read the contents of your USB storage

find accounts on the device

DEVICE ACCESS

full network access
receive data from Internet
view network connections
view Wi-Fi connections

".

Mobile APPS: Android LOGS : EXTRA

Installing LOGCAT APP to your Device

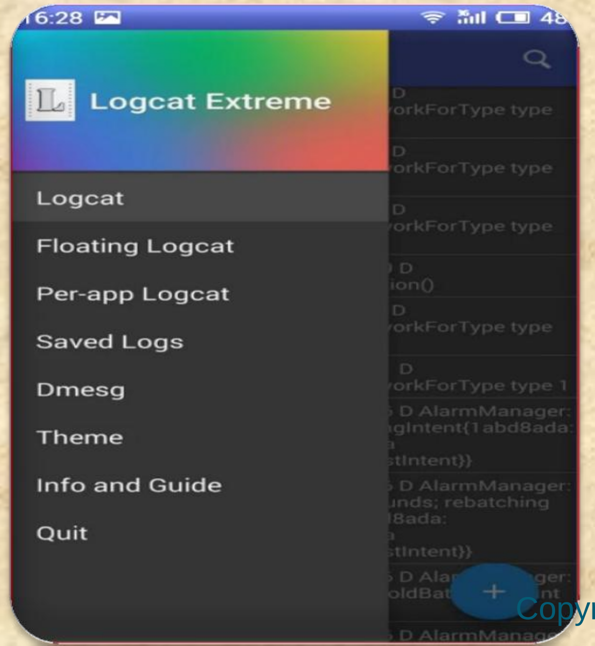
Download

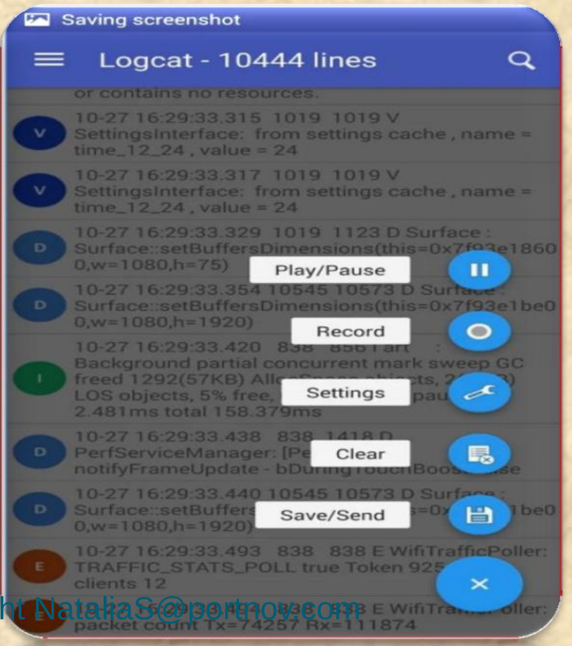
LOGCAT EXTREME

from PlayStore to your Device directly *(this is only an example, there are many other similar apps)*

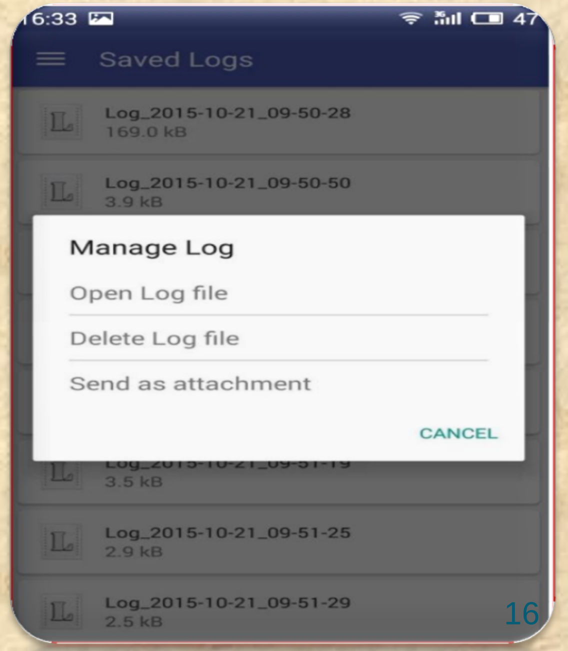
LOGCAT EXTREME *is* an enhanced Logcat reader and Logcat recorder which comes with a rich set of features and handy user interface.

Please note: From Android 4.1 onwards ANY logcat app needs root access in order to show logs properly.





```
or contains no resources.
V 10-27 16:29:33.315  1019  1019 V
SettingsInterface:  from settings cache , name =
time_12_24 , value = 24
V 10-27 16:29:33.317  1019  1019 V
SettingsInterface:  from settings cache , name =
time_12_24 , value = 24
D 10-27 16:29:33.329  1019  1123 D Surface :
Surface::setBuffersDimensions(this=0x7f93e1860
0,w=1080,h=75)
D 10-27 16:29:33.354 10545 10573 D Surface :
Surface::setBuffersDimensions(this=0x7f93e1be0
0,w=1080,h=1920)
I 10-27 16:29:33.420   838   856 I art     :
Background partial concurrent mark sweep GC
freed 1292(57KB) All
LOS objects, 5% free,                   pau
2.481ms total 158.379ms
D 10-27 16:29:33.438   838  1418 D
PerfServiceManager: [Pe
notifyFrameUpdate - bDuringTouchBoos
D 10-27 16:29:33.440 10545 10573 D Surface :
Surface::setBuffers                 =0x      1be0
0,w=1080,h=1920)
E 10-27 16:29:33.493   838   838 E WifiTrafficPoller:
TRAFFIC_STATS_POLL true Token 925
clients 12
E                                838 E WifiTra        oller:
packet count Tx=74257 Rx=111874
```



Log_2015-10-21_09-50-28
169.0 kB
Log_2015-10-21_09-50-50
3.9 kB
Log_2015-10-21_09-51-19
3.5 kB
Log_2015-10-21_09-51-25
2.9 kB
Log_2015-10-21_09-51-29
2.5 kB

Mobile APPS: Distribution/Installation: iOS .IPA

- **Through an App Marketplace** (APP STORE)
- **XCODE**
- **ITUNES**
- **TEST FLIGHT**

Mobile APPS: Distribution/Installation: .IPA

What is XCode?

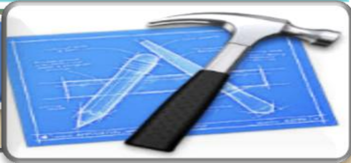
<ins>Xcode is an Intergrated Development Environment by Apple containing a suite of software Development Tools for macOS, iOS, WatchOS and tvOS</ins>

Xcode supports source code for programing languages C, C++, Objective-C, Objective-C++, Java, AppleScript, Python, Ruby, ResEdit (Rez), and Swift

Also supports variety of programming models, including but not limited to Cocoa, Carbon, and Java

Mobile APPS: Distribution/Installation: .IPA

Distributing .IPA through XCode ?

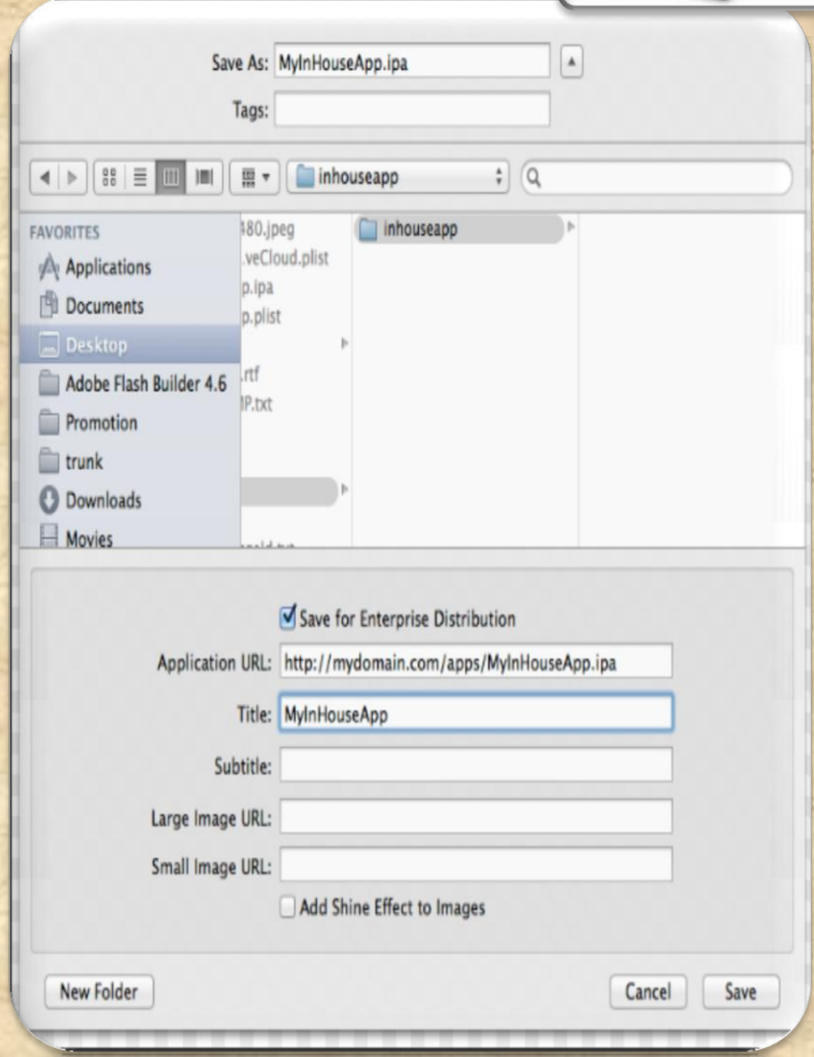


Connect the device to your Mac.

In Xcode, choose Window > Devices and select the device under Devices.

In the Installed Apps table, click the Add button (+) below the table.

In the dialog that appears, choose the iOS App file and click Open.


Save As: MyInHouseApp.ipa
Tags:

☑ Save for Enterprise Distribution
Application URL: http://mydomain.com/apps/MyInHouseApp.ipa
Title: MyInHouseApp
Subtitle:
Large Image URL:
Small Image URL:
☐ Add Shine Effect to Images

Mobile APPS: Collecting LOGS: .IPA

How to do it through Xcode on MAC ?

1. Install XCode

2. Connect your iPhone to the Mac

3. Select Trust this computer on the iPhone pop-up request

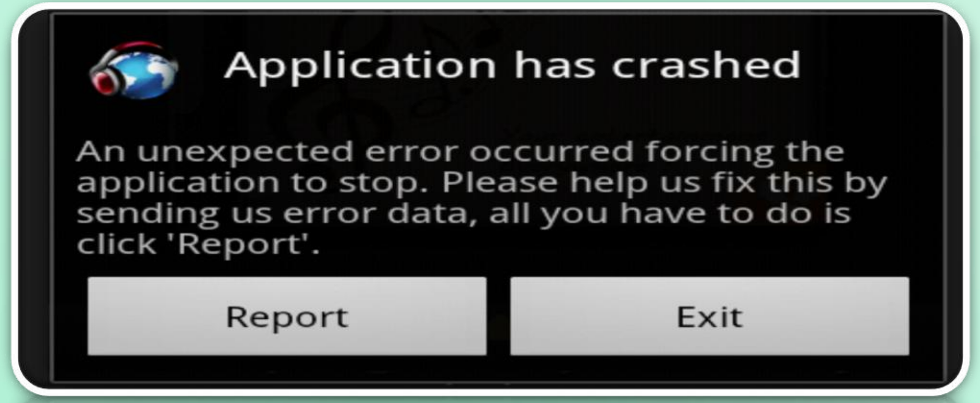
4. Start xCode (Menu) →Window → Devices (Select your iPhone and press the arrow button in the bottom right

5. Reproduce the problem

6. Press the arrow button and download the logs

How can I debug a deployed app without Xcode debugger ?

- A: Once you have deployed your app, either through the *App Store* or as an *Ad Hoc* or *Enterprise* build, you won't be able to attach *Xcode's* debugger to it.
- To debug problems, you need to analyze *Crash Logs and Console* output from the device.



Application has crashed

An unexpected error occurred forcing the application to stop. Please help us fix this by sending us error data, all you have to do is click 'Report'.

Getting Crash Logs and Console Output

Getting Crash Logs Directly From a Device

Without Xcode

Users can retrieve crash reports from their device and send them to you via email by following these instructions.

(It is not possible to get device console logs directly from a device)

- Open Settings app
- Go to Privacy, then Diagnostics & Usage
- Select Diagnostics & Usage Data
- Locate the log for the crashed app. The logs will be named in the format: *<AppName>_<DateTime>_<DeviceName>*
- Select the desired log. Then, using the text selection UI select the entire text of the log. Once the text is selected, tap Copy
- Paste the copied text to Mail and send to an email address as desired

Mobile APPS: Distribution/Installation: .IPA

What is iTunes?

iTunes is a media player, media library, online radio broadcaster, and mobile device management application developed by Apple Inc.

It is used to play, download, and organize digital downloads of music and video (as well as other types of media available on the iTunes Store) on personal computers running the macOS and Microsoft Windows operating systems.

Mobile APPS: Distribution/Installation: .IPA

- *Drag-and-drop IPA* file into '*Apps*' tab of iTunes BEFORE you connect the device
- Connect your device
- Select *your device* on iTunes
- Select '*Apps*' tab
- *Search app* that you want to install
- Click on '*Install*' button. This will change to '*Will Install*'
- Click on '*Apply*' button on right corner

LOGS: Collecting from iTunes

- Sync your device with iTunes on your desktop.
- After syncing, look for crash logs in the correct directory.

The Following few slides will give an instructions . Lets start !

LOGS: Collecting from iTunes

- **Mac OS X:**

- Open Finder (found in the Dock)
- Click on the 'Go' menu at the top of your screen, and select 'Go to Folder'
- Type (or paste): *~/Library/Logs/CrashReporter/MobileDevice/<DEVICE_NAME>*
- Open the folder with the same 'name' as your device.

  (Note: your device name appears in iTunes on the left side, under 'Devices').
- Open the folder called 'Retired'
- You will see at least one item starting with 'ReadItLaterPro'.

LOGS: Collecting from iTunes

- **Windows Vista or 7:**

- Open any Windows Explorer Window (My Computer, My Documents, etc.)
- Enter %appdata%, and press enter →Navigate to Roaming
  C:\Users\<USERNAME>\AppData\Roaming\Apple
- ***Computer\Logs\CrashReporter\MobileDevice\<DEVICE_NAME>***
- (Note: your device name appears in iTunes on the left side, under 'Devices')
- You will see at least one item starting with 'ReadItLaterPro'.

LOGS: Syncing your device with iTunes

- **Windows XP:**

- Locate your Application Data folder.
- Navigate to Apple computer ***C:\Documents and Settings\<USERNAME>\Application Data\Apple***
- ***Computer\Logs\CrashReporter\MobileDevice\<DEVICE_NAME>***
- (Note: your device name appears in iTunes on the left side, under 'Devices')
- You will see at least one item starting with 'ReadItLaterPro'.

LOGS: Collecting from your Device

- To obtain iOS crash logs, please select your device and go to:
- ***Settings > Privacy > Diagnostics & Usage (for iOS 8 or newer)***
- ***Settings > General > About > Diagnostics & Usage (for iOS 7 or older)***

- Select a ***Chrome*** crash from the list.

  This will start with ***"Chrome_"*** and contain the timestamp of the crash.
- Tap on the crash and you will see a text field with a crash log.

  Long press to *Select All* and then *Copy* the crash text.
- Paste it into something you can get off of your device (for example, an email to yourself).

Mobile APPS: Distribution/Installation: .IPA

What is TestFlight?

TestFlight is an online service for over-the-air installation and testing of mobile applications, currently owned by Apple Inc and only offered to developers within the iOSDeveloper Program

Developers signed up with the service to distribute applications to internal or external beta testers, who could subsequently send feedback about the application to developers

The TestFlight SDK additionally allowes developers to receive remote logs, crash reports and tester feedback.

HOMEWORK : READ AN ENTIRE ARTICLE ABOUT TestFLIGHT
https://www.raywenderlich.com/48750/testflight-sdk-tutorial

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Challenges

- Device fragmentation
- In-house vs. outsourced testing
- Availability of mobile testing tools
- Application Lifecycle Testing

- Like any desktop or web application testing, mobile application testing is also focused on the quality and performance of the final product.
- However, mobile app testing becomes far more challenging because of the following key factors

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

- Device Model
- OS Version
- Screen Resolution
- Form Factor
- Emulators vs. Physical Devices

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

- Network density
- How the app behaves on specific devices
- How real-world users interact with the app
- Different battery states on the devices
- Multiple networks (Wi-Fi, 4G, 3G, etc.)

Beta Testing of your Mobile App

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

applications can be deployed, tested, and managed

saves businesses from setting up on-premise test environments

capability to support complex apps

provides real-time testing results

Mobile App Testing on Cloud

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

testing mobile apps in real network environments

network simulation tools are available

test mobile apps in various network speeds, bandwidths variations

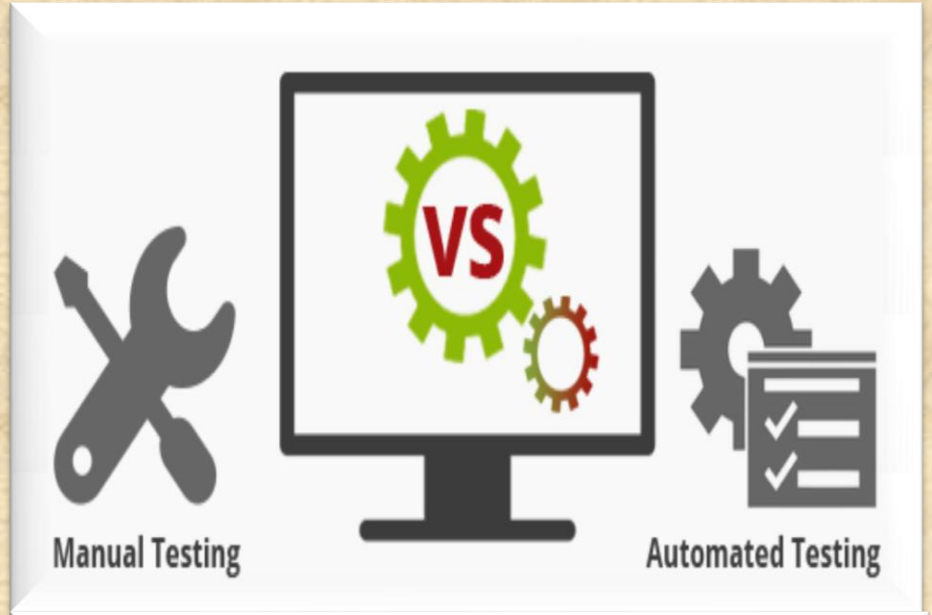
testing the app in a full internet connectivity scenario and other factors

Mobile Test Industry Standards :

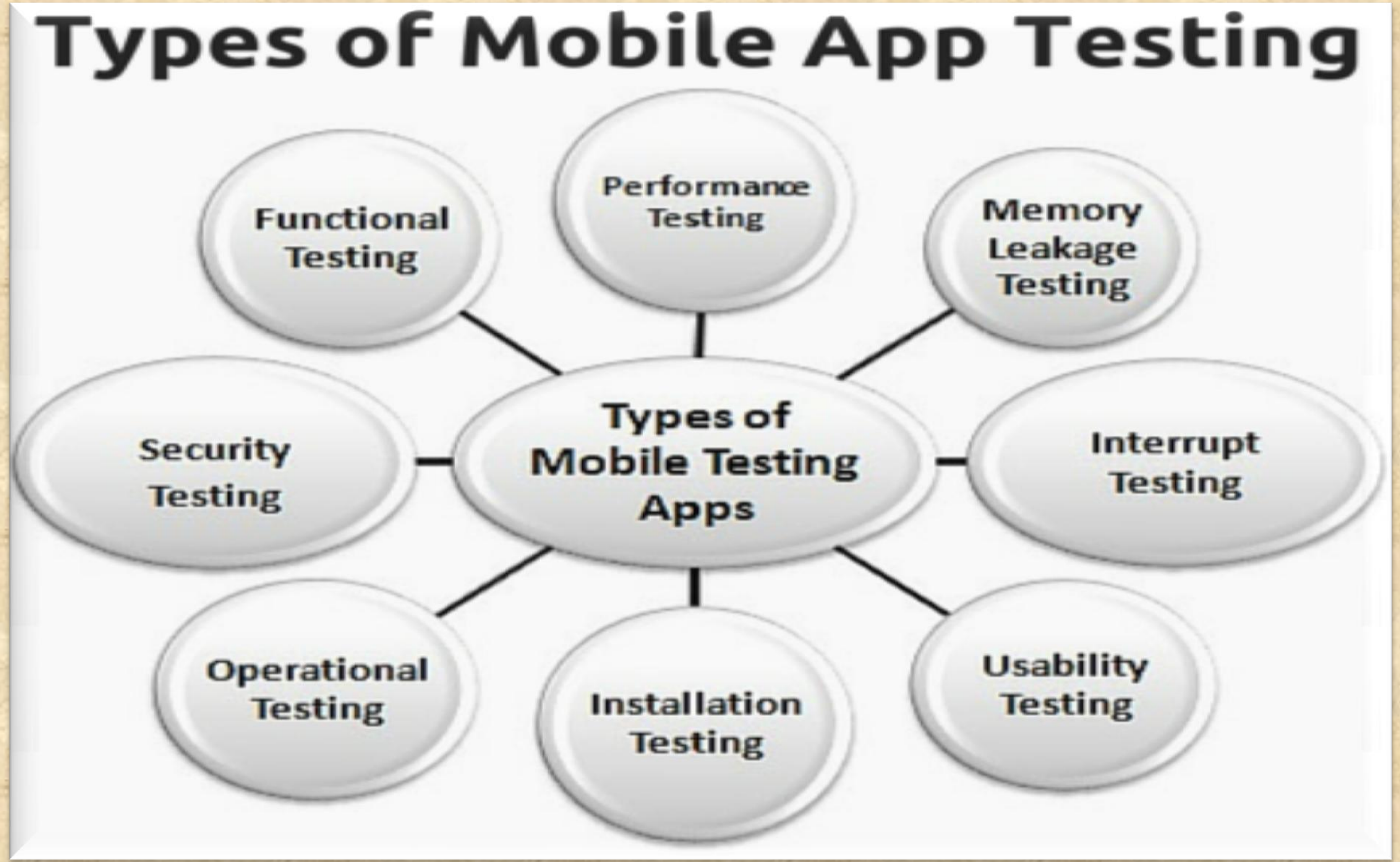
Testing Strategies for Mobile Apps

Automated testing is highly effective in consistently repeating a test procedure in regression testing as well as testing during the development stages.

However, test automation requires significant amount of initial investment.



Mobile Test Industry Standards :
Testing Strategies for Mobile Apps



Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

FUNCTIONAL TEST

- Verifying that all documented requirements are implemented.
- Verifying that all features work as expected.
- Validating texts, logos, images, text captions and other UI elements.
- Validating localization and globalization.
- Evaluating ease of navigation and screen transitions.
- Examining response speed.
- Evaluating the intuitiveness of the touch interface.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

PERFORMANCE TEST

- Performance with low battery power
- Performance while network out of coverage area
- Performance during poor bandwidth
- Performance while changing internet connection mode
- Performance while transferring heavy file
- Testing from Application’s server and client side

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Memory Leakage TEST

- Verifying if program runs for an extended time and consumes additional memory
- Verifying if memory is allocated frequently for one-time tasks
- Verifying where the program can request memory — such as shared memory that is not released
- Verifying where memory is very limited, such as in an embedded system or portable device
- Verifying where the leak occurs within the operating system or memory manager
- Verifying when a system device driver causes the leak

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

INTERRUPT TEST

- Battery low
- Battery full- when charging
- Incoming phone call
- Incoming SMS
- Incoming Alert from another mobile application
- Plugged in for charging
- Plugged out from charging
- Device shut off
- Application Update reminders
- Alarm
- Network connection loss
- Network connection restoration

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

USABILITY TEST

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.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

INSTALLATION TEST

- Verify application gets installed properly
- Verify user can uninstall application successfully
- Verify app updates are properly installed
- Verify aborting installation does not affect other features
- Check app behavior on trying to install it on non-supported version/device.
- Verify app is installed properly from app store and from side loading

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

SECURITY TEST

Data flow -- Can you establish an audit trail for data, what goes where, is data in transit protected, and who has access to it?

Data storage -- Where is data stored, and is it encrypted? Cloud solutions can be a weak link for data security.

Data leakage -- Is data leaking to log files, or out through notifications?

Authentication -- When and where are users challenged to authenticate, how are they authorized, and can you track password and IDs in the system?

Server-side controls -- Don't focus on the client side and assume that the back end is secure.

Points of entry -- Are all potential client-side routes into the application being validated?

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps EXTRA

FUNCTIONAL VS Non-FUNCTIONAL TEST

FUNCTIONAL	NON- FUNCTIONAL
Unit Testing Smoke testing / Sanity testing	Load and Performance Testing
Integration Testing (Top Down, Bottom up Testing)	Ergonomics Testing
Interface & Usability Testing	Stress & Volume Testing
System Testing	Compatibility & Migration Testing
Regression Testing	Data Conversion Testing
Pre User Acceptance Testing (Alpha & Beta)	Penetration Testing
User Acceptance Testing	Operational Readiness Testing
White Box & Black Box Testing	Installation Testing
	Security Testing

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Platform/OS TEST

- Different OS ->Android, IOS, Windows
- Different browsers -> Firefox, Google Chrome, IE, Safari
- Different Screen Size and resolution
- OS versions and memory size
- Hardware capable of interrupt handling without getting hanged
- Multilingual Support
- Different Time Zones Support

Mobile Test Industry Standards :
Testing Strategies for Mobile Apps EXTRA

ACCESABILITY TEST (What is SCREEN READER ?)

Mobile Accessibility is critical to reaching all audiences. A product is accessible when a person with a disability can have an experience equivalent to that of a person without a disability

Users who are blind will use a screen reader to navigate and access information on mobile devices.

The screen readers are included in the device operating system and can be turned on in the device settings.

When Screen Reader is turned on, the gestures and keyboard shortcuts change.

In the 2014 Webaim survey shows that 82% of Screen Reader users will use a mobile device

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps EXTRA

ACESSABILITY TEST (SCREEN READER)

Web Content Accessibility Guidelines (WCAG)

- A person who is blind using a screen reader or a talking browser can navigate your information and interact with it.
- A person with low-vision can magnify the screen and understand the content.
- A person who is deaf or hard-of-hearing can read captions in multimedia presentations.
- A person with a dexterity limitation can use the alternative input devices for all interaction, or can use speech recognition software.
- A person with ADHD or dyslexia can use and understand the content and complete tasks
- Please refer to this link to learn more https://www.w3.org/TR/WCAG20/

Screen reader testing on mobile

Zooming site/application

Color ratios

Readability of the site

Navigation

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA

Workshop : ANSWER THESE QUESTIONS

1. *What do you consider to be the biggest security issues with mobile phones?*
2. *How seriously are consumers and companies taking these threats?*
3. *What can be done about these threats?*

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA

- Attacks on mobile devices range in volume and severity, but all have the potential to cause chaos at both a device and network level.

Just like in the conventional fixed Internet world, attacks come in all shapes and sizes – such as:

- Phishing (criminals attempt to trick users into sharing passwords etc)
- Spyware (tracks user's activity, perhaps selling data to advertisers)
- Worms (a program that copies itself onto multiple devices via network connections)
- Trojans (a program that looks genuine but hides malicious intent)
- Man-In-The-Middle Attacks (where a criminal intercepts and manipulates messages between two devices or device and computer).

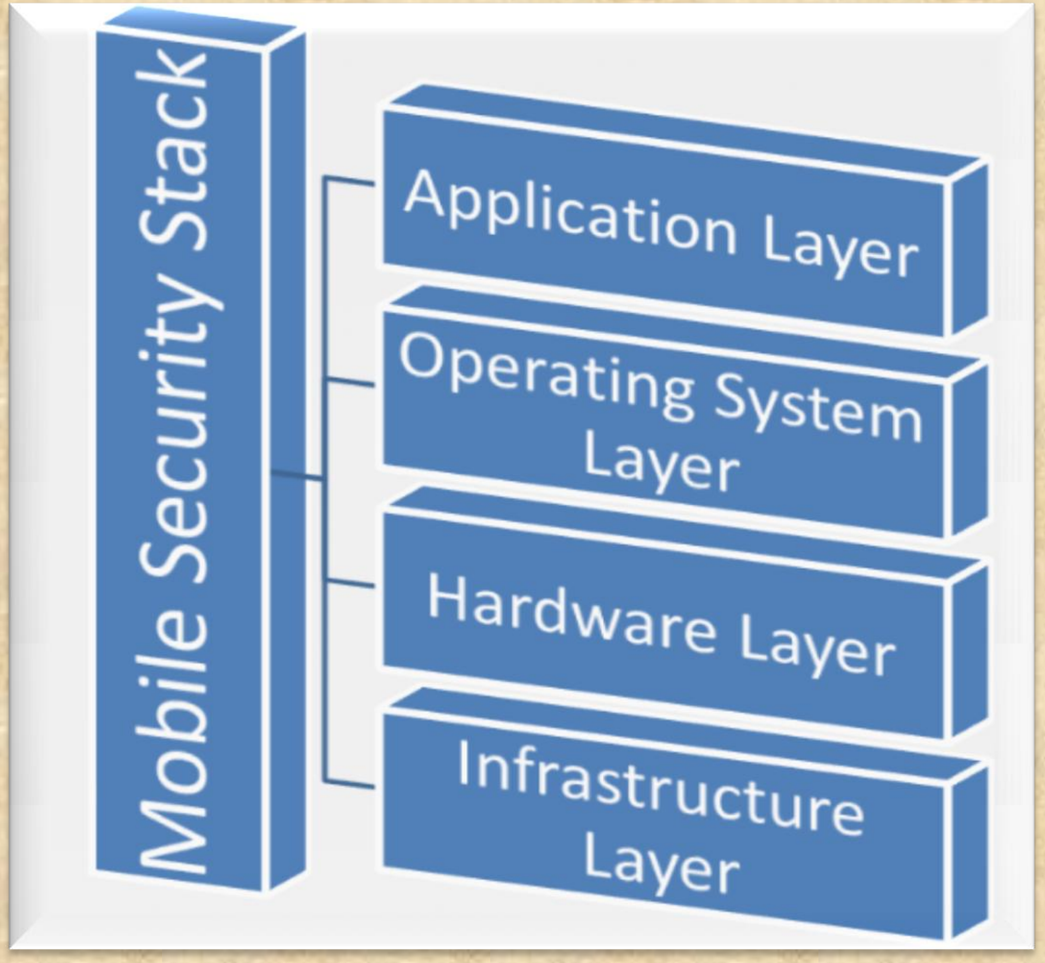
Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA

The Mobile Code Security Stack

- The mobile code security stack can be broken up into four distinct layers.
- Each layer of the mobile code security model is responsible for the security of its defined components and nothing more.
- The upper layers of the stack rely on all lower layers to ensure that their components are appropriately safe

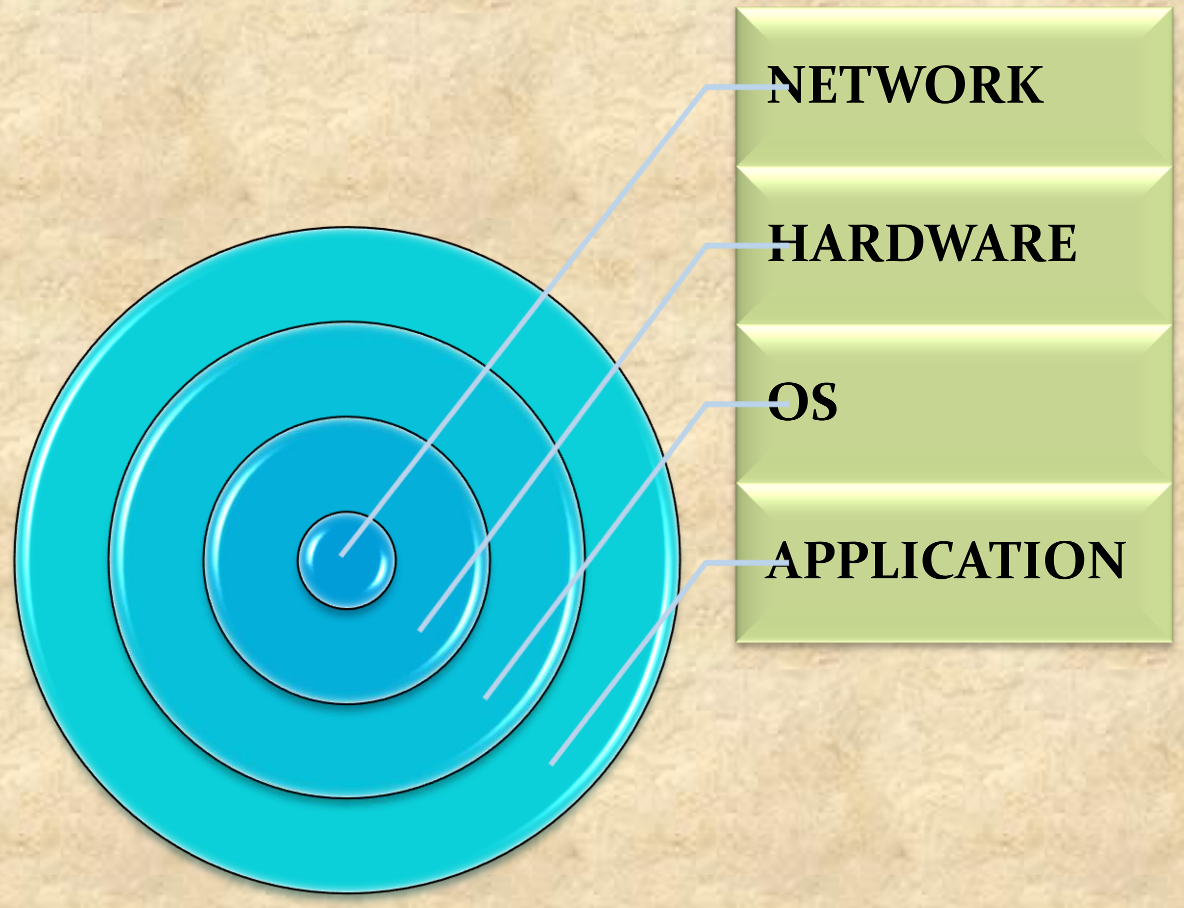


Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA

Mobile Device Risks at Every Layer



Example :

Your device isn't rooted but all your email and pictures are stolen, your location is tracked, and your phone bill is much higher than usual.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

OWASP Mobile Security Project

Security Test EXTRA

What is OWASP ?

- The Open Web Application Security project is an online community which creates freely-available articles, methodologies, documentation, tools, and technologies in the field of Web App Security

OWASP Top Ten:

- The Top Ten was first published in 2003 and is regularly updated.
- Its goal is to raise awareness about application security by identifying some of the most critical risks facing organizations.
- The Top 10 project is referenced by many standards, books, tools, and organizations, including MITRE, PCI DSS, Defense Information Systems Agency, FTC, and many more.

CWE – COMMON WEAKNESS ENUMERATION :
https://cwe.mitre.org/about/

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN

There are two main categories of mobile code security risks:

MALICIOUS FUNCTIONALITY

- The category of malicious functionality is a list of unwanted and dangerous mobile code behaviors that are stealthily placed in a Trojan app that the user is tricked into installing.
- Users think they are installing a game or utility and instead get hidden spyware, phishing UI or unauthorized premium dialing.

VULNERABILITIES.

- The category of Mobily Security vulnerabilities are errors in design or implementation that expose the mobile device data to interception and retrieval by attackers.
- Mobile code security vulnerabilities can also expose the mobile device or the cloud applications used from the device to unauthorized access.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN 2017

- A1-Injection
- A2-Broken Authentication and Session Management
- A3-Sensitive Data Exposure
- A4-XML External Entities (XXE)
- A5-Broken Access Control
- A6-Security Misconfiguration
- A7-Cross-Site Scripting (XSS)
- A8-Insecure Deserialization
- A9-Using Components with Known Vulnerabilities
- A10-Insufficient Logging&Monitoring

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN

A1-Injection

Injection flaws, such as SQL, OS, XXE, and LDAP injection occur when untrusted data is sent to an interpreter as part of a command or query.

Example:

- The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN

2. A2-Broken Authentication

Application functions related to authentication and session management are often implemented incorrectly,

Attackers can compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities (temporarily or permanently).

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN

3. A3-Sensitive Data Exposure

Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII.

Example:

Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes.

Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN

4. A4-XML External Entities (XXE)

Many older or poorly configured XML processors evaluate external entity references within XML documents.

External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN

5. A5-Broken Access Control

Restrictions on what authenticated users are allowed to do are often not properly enforced.

Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN

6. A6-Security Misconfiguration

Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information.

Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN

7. A7-Cross-Site Scripting (XSS)

XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript.

XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN

8. A8-Insecure Deserialization

Insecure deserialization often leads to remote code execution.

Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN

9. A9-Using Components with Known Vulnerabilities

Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover.

Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test EXTRA- OWASP TOP TEN

10. A10-Insufficient_Logging &Monitoring

Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data.

Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Security Test

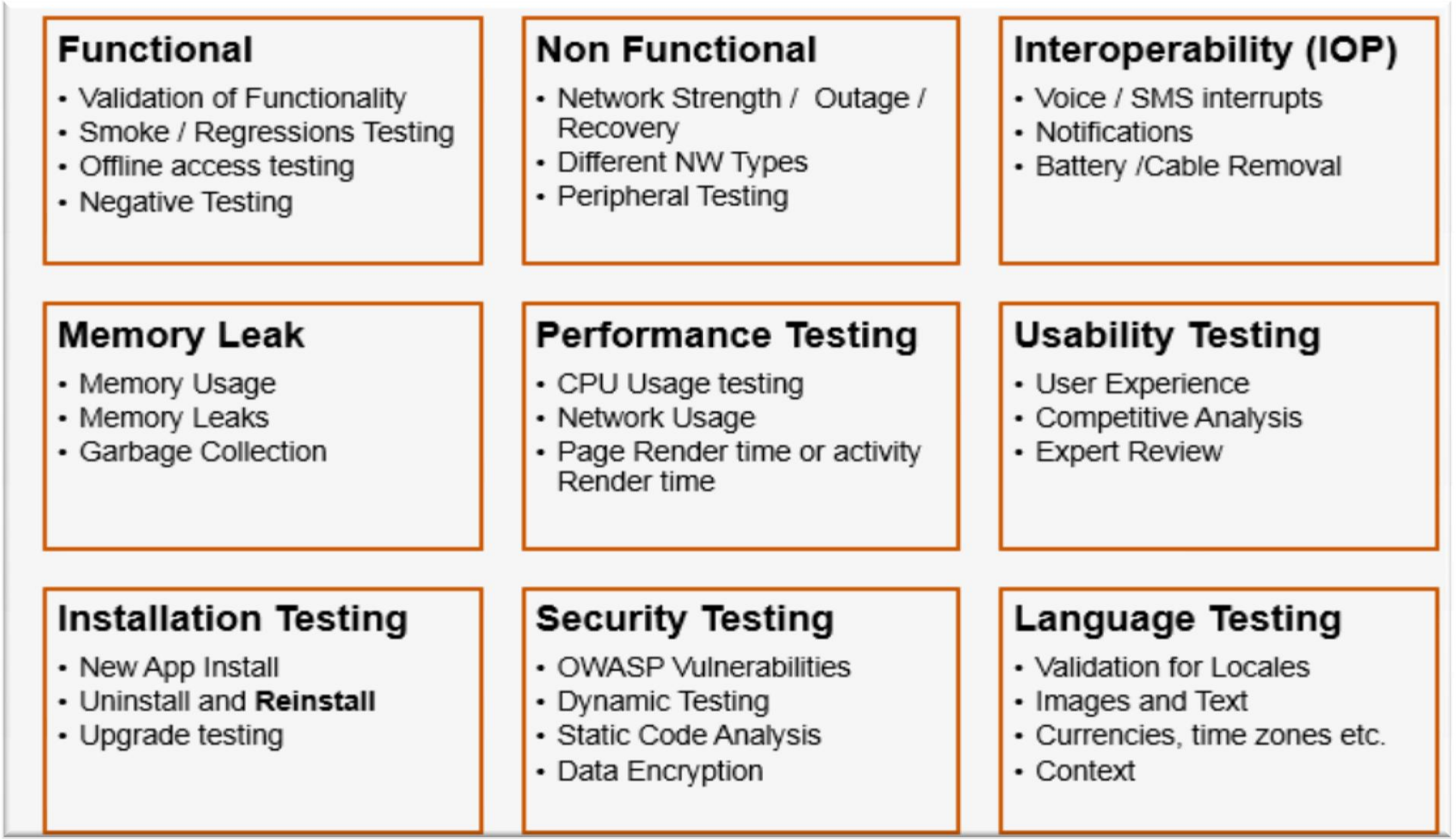
CREATE CHECK LIST BEFORE

- Phone identifiers such as (IMSI or IMEI)
- Address Book
- Account Details
- E-maiL
- Stock application data
- Banking Data
- GPS Location(s)
- Web History
- User's Dictionary
- Images
- Notes
- Calendar Appointments
- Call Logs
- Encryption Keys

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps EXTRA

SUMMARY



Functional

- Validation of Functionality
- Smoke / Regressions Testing
- Offline access testing
- Negative Testing

Non Functional

- Network Strength / Outage / Recovery
- Different NW Types
- Peripheral Testing

Interoperability (IOP)

- Voice / SMS interrupts
- Notifications
- Battery /Cable Removal

Memory Leak

- Memory Usage
- Memory Leaks
- Garbage Collection

Performance Testing

- CPU Usage testing
- Network Usage
- Page Render time or activity Render time

Usability Testing

- User Experience
- Competitive Analysis
- Expert Review

Installation Testing

- New App Install
- Uninstall and **Reinstall**
- Upgrade testing

Security Testing

- OWASP Vulnerabilities
- Dynamic Testing
- Static Code Analysis
- Data Encryption

Language Testing

- Validation for Locales
- Images and Text
- Currencies, time zones etc.
- Context

Mobile Test Industry Standards
Testing Strategies for Mobile Apps : LETS PRACTICE

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

How to Start Testing a Mobile Page

1. Find and divide page into components/modules
2. Create a list/table of modules/testing types/ testing
3. Create a variety of variables and conditions within each module
4. Specify the Expected and leave place for Actual Results
5. FOR TROUBLESHOUTING Create a list/table to view ANY problem from different angels
6. Create a Check List for Testing Categories:
7. Functionality Testing
8. Usability testing
9. Interruption testing
10. Connectivity testing
11. Compatibility testing
12. Performance testing
13. Security testing

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

Consumers behaviour only on the basis of experience delivered by app

29%

of smartphone users will immediately switch to another site or app if it doesn't satisfy their needs

70%

of them do so because of lagging load times

67%

will switch if takes too many steps to purchase or get desired information

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

GUI TEST Checklist

- Navigation
- Formatting
- Color and fonts
- Scrolls
- Controls and alignments
- Spelling and grammar
- Justification
- Look and feel
- Default and shortcut keys
- Tab
- Opening input
- Alternatives
- Behavior
- Modality and multiple windows
- Contrast
- Images

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

top considerations for creating a release CHECK LIST for mobile app testing

- Application Installation/Update
- Application Sign Up & Log in
- Subscription scenarios
- Application Sanity Suit
- APP works in **different Mobile modes**
- User Friendly
- Network connectivity
- Data save conditions
- Mobile interruptions
- Battery Consumption
- Mobile memory utilization
- Mobile data utilization
- Screen scrolling application screen
- New OS release support
- correct implementation of AdMob or other mobile ad platform

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

Test case	Sub-Category	Description	Result
1. Installation		Verify that app can be Installed Successfully	App should be able to install Successfully
2. Un-installation		Verify that app can be Uninstall Successfully	User should be able to uninstall the app successfully
3. Network Test Cases		Verify the behavior of app when there is Network problem and user is performing operations for data call	User should get proper error message like "Network error. Please try later"…
		Verify that User is able to establish data call when Network is back in action	User should be able to establish data call when Network is back in action

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

TC	Sub-Category	Description	Result
4. Voice Call Handling	Call Accept	Verify that User can accept Voice call at the time when app is running and can resume back in app from the same point	User should be able to accept Voice call at the time when app is running and can resume back in app from the same point
	Call Rejection	Verify that User can reject the Voice call at the time when app is running and can resume back in app from the same point	User should be able to reject the Voice call at the time when app is running and can resume back in app from the same point
	Call Establish	Verify that User can establish a Voice call in case when app data call is running in background	User should be able to establish a Voice call in case when app data call is running in background

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

TC	Sub-Category	Description	Result
5. SMS Handling		Verify that User can get SMS alert when app is running	User should be able to get SMS alert when app is running
		Verify that User can resume back from the same point after reading the SMS	User should be able to resume back from the same point after reading the SMS
6. Unmapped Keys		Verify that unmapped keys are not working on any screen of app	Unmapped key should not work on any screen of app
7. Application Logo		Verify that app logo with App Name is present in app manager, on the App screen page, widgets (opt.) and user can select it	Application Logo with App Name should be present in app manager, on the App screen page, widgets (opt.) and User can select it.

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

TC	Sub-Category	Description	Result
8. Splash Screen		Verify that when User selects app Logo in Splash is displayed. **Note:** Splash do not remain for fore than 3 sec **Note:** A splash screen is an image that appears while a game or program is loading.	When User selects app, Logo in app Splash should be displayed
9. Low Memory		Verify that app displays proper error message when device memory is low and exits gracefully from the situation	App should display proper error message when device memory is low and exits gracefully from the situation
10. Clear/Back Key		Verify that Clear key should navigate the User to previous screen	Clear Key should navigate the User to previous screen
11. End/Home Key		Verify that End Key should navigate the User to native Device screen	End Key should navigate the User to native Device screen

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

TC	Sub-Category	Description	Result
12.Visual Feedback		Verify that there is visual feedback when response to any action takes more than 3 sec	There should be visual feedback given when response time for any action is longer than 3 sec
13. Continual Keypad Entry		Verify that continual key pad entry do not cause any problem. **Note:** Continual Keypad test consist in a multiple key press, done quickly as possible, in order to load at maximum capacity the handset's memory	Continual key pad entry should not cause any problem in app
14. Exit Application		Verify that User is able to exit from app with every form of exit modes such as Flap, Slider, Home Key or Exit option from any point of app	User should be able to exit with every from of exit mode such as Flap, Slider, Home Key or Exit option from any point of app
15. Charger Effect		Verify that when app is running then inserting and removing charger do not cause any problem and proper message is displayed when charger is inserted in device	When app is running, then insertion or remove of charger not cause any problem, and proper message displayed .

Mobile Test Industry Standards
Testing Strategies for Mobile Apps : LETS PRACTICE

TC	Sub-Category	Description	Result
16. Low Battery		Verify that when app is running and battery is low, then proper message is displayed to the User.	When app is running and battery is low, there should be proper message displayed to the User
17. Removal of Battery		Verify that removal of battery at the time of app data call is going on do not cause interruption and data call is completed after battery is inserted back in the device	Removal of battery at the time of app data call is going on should not cause interruption and data call should be completed after battery is inserted back in the device
18. Battery Consumption		Verify that app does not consume battery excessively	The app should not consume battery excessively
19. Application Start/Restart		Find the app icon and select it. Press tab on the Device to launch the app. Observe the app launch in the timeline defined.	App must not take longer than 25 sec to start

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

TC	Sub-Category	Description	Result
20. Application Side Effects		Make sure that your app is not causing other apps of device to hamper	Installed app should not cause other apps of device to hamper
21. External incoming communication infrared		App should gracefully handle the condition when incoming communication is made via InfraRed	When incoming communication enters the device, the app must at least respect one of the following: A. Go into pause state during InfraRed session and automatically continue from the point it was suspended at after the InfraRed session is done B. Give a visual or audible notification The app must not crash or hung.

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

TC	Sub-Category	Description	Result
22. Bluetooth interrupt:		When a file transfer is taking place with Bluetooth, the application must be paused and should be resumed from the same point after the transfer is done	

Mobile Test Industry Standards
Testing Strategies for Mobile Apps : LETS PRACTICE

HOMEWORK

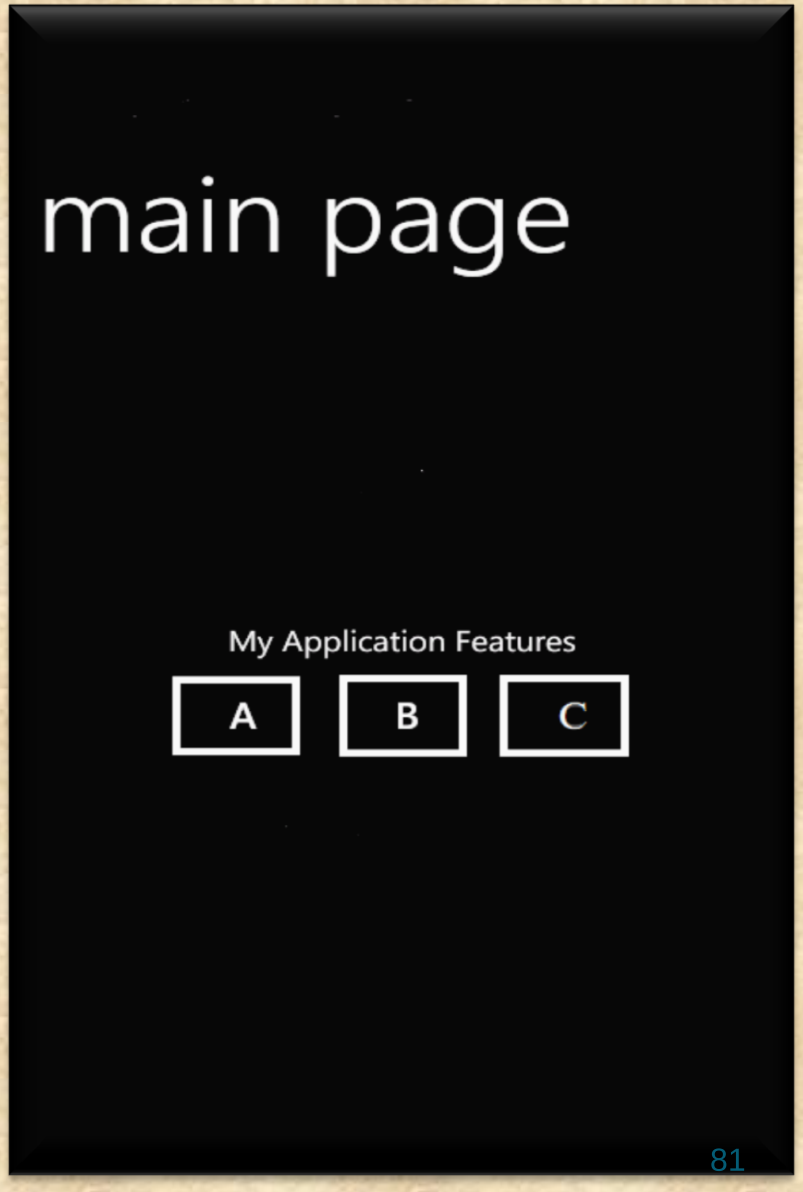
Write as many Test Cases you can for this simple app on Mobile device with three buttons (A, B and C) that making different sounds upon tapping on it.

A - for Audio 1
B - for Audio 2
C - for Audio 3

You are free to create conditions and Rules for each button , but be consistent.

Write Test Cases (use previous slides for hints).

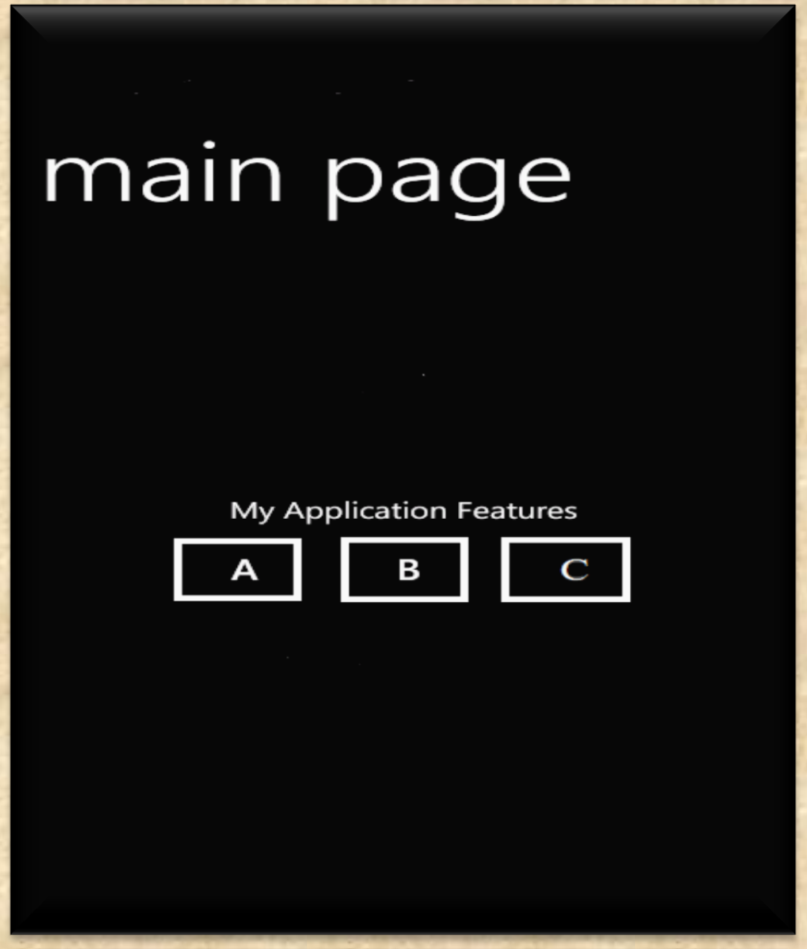
HAVE FUN !



Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

HOMEWORK REVIEW

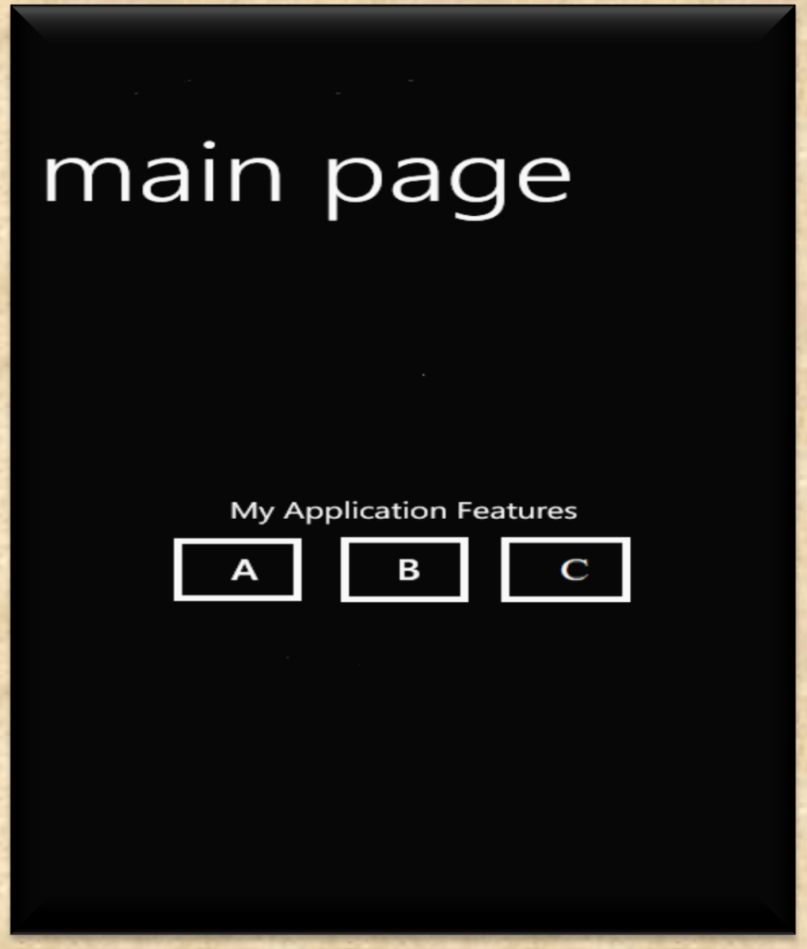


Functional Test

Case	Description	Result
Button A	Verify that when Button A is pressed, sound tone A appeared	When button A is pressed the sound tone A should be audible
Button B	Verify that when Button B is pressed, sound tone B appeared	When button B is pressed the sound tone B should be audible
Button C	Verify that when Button C is pressed, sound tone C appeared	When button C is pressed the sound tone C should be audible
Combination of buttons and sounds	Verify that when A,B,C buttons are pressed consecutively, the specific sound A,B,C is appeared	When buttons A,B, C are pressed consecutively, the audible tones A, B, C should be observed

Mobile Test Industry Standards
Testing Strategies for Mobile Apps : LETS PRACTICE

HOMEWORK REVIEW



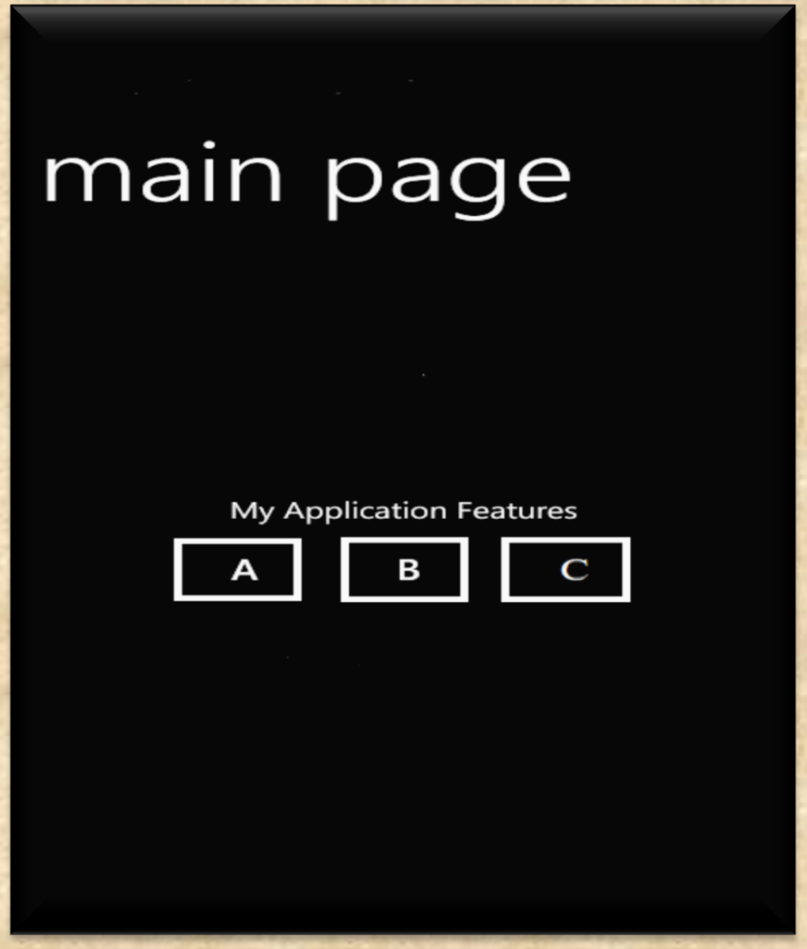
UI Test

Case	Description	Result
Panning (sliding horizontally left-right) **Swiping**	Verify that when main Page is panned/swiped, the sound buttons A,B,C remains in the same order, the same position on the page screen, and do not make sound	**The buttons A,B,C, should not loose the order or make any sound during panning/swiping gestural input procedures**
Rotation	Verify that when device is rotated, Buttons ABC should not loose it's order and make any sound	**During device's rotation Buttons ABC should not loose it's order and make any sound**
Zooming	Verify that buttons A,B, C should not loose the order or make any sound during the Zooming gestural procedure	**Buttons A,B,C should not loose the order or make and sound during the Zooming procedure**

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

HOMEWORK REVIEW



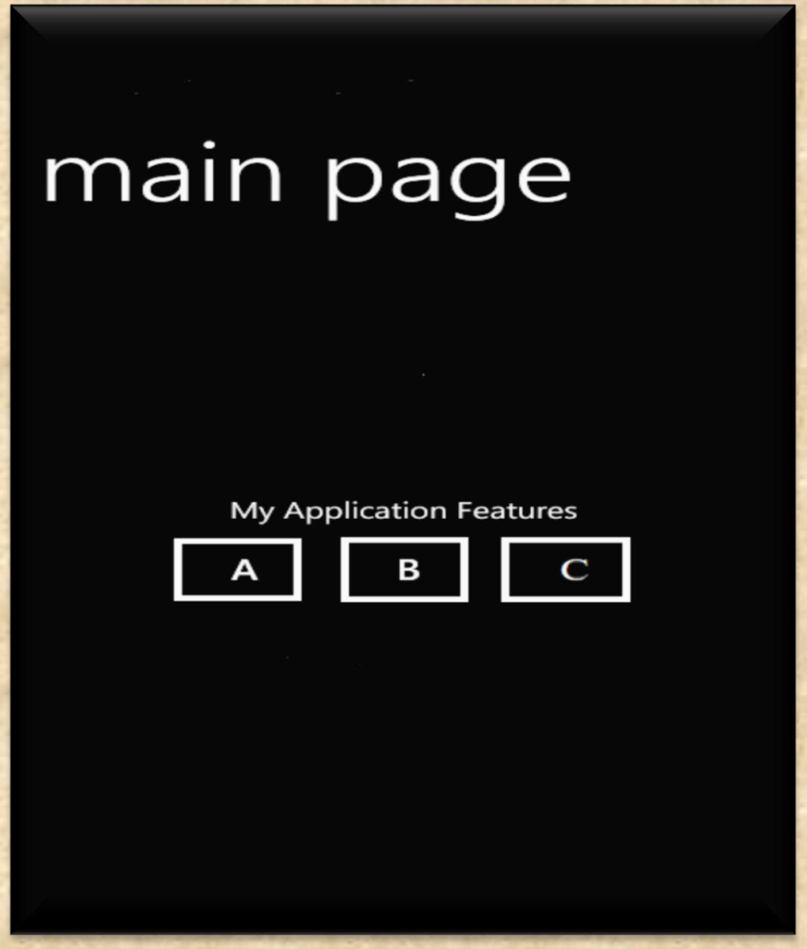
Interruption Test

Case	Description	Result
Phone Call Interruption	Verify that when Phone Call is initiate, buttons ABC are in "pause" mode and do not perform assigned sound tones.	When Phone Call is occurred, the Buttons ABC should be saved in 'pause" mode and do not perform assigned sound tone.
Text message interruption	Verify that when SMS notifications/ message appears, the main app page will response with safe, end session	When SMS action occurs, proper error message should be displayed and app will be closed gracefully with saved information
Verge App Notification (w/ TuneTone)	Verify that when TechNews Notification with the Ringtone occurs, buttons ABC will pause and perform assigned sound tones after Notification Ringtone is done.	When TechNews Notification (w/Ringtone) occurs the Buttons ABC should be pause until Ringtone tune are done, and continue to perform ABC assigned sound after no more than 3 sec delay.

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

HOMEWORK REVIEW



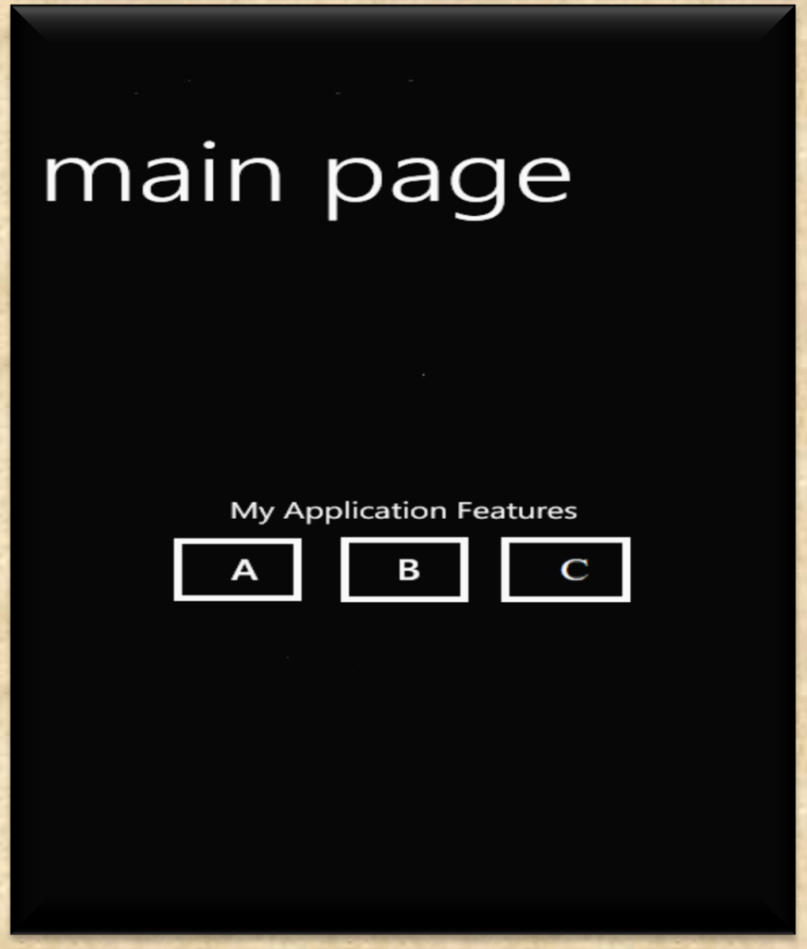
Connectivity Test

Case	Description	Result
Flight Mode of Mobile Device	Verify that when Device has Flight Mode ON, the Buttons ABC are still active and performing sound	Buttons ABC should be active and perform assigned sounds when Mobile Device is in Offline Mode.
Bluetooth Connection active with Wearable Device	Verify that when Wearable Device BT connected and play Music, the Buttons ABC are still active and performing sound	Buttons ABC should be active and perform assigned sounds when Mobile Device is in active Bluetooth Mode.
Low bandwidth Network	Verify that when Device is in Frequently changed "hopping" area the Buttons ABC are still active and performing sound	Buttons ABC should be active and perform assigned sounds when Mobile Device is in the "hopping mode"

Mobile Test Industry Standards

Testing Strategies for Mobile Apps : LETS PRACTICE

HOMEWORK REVIEW



Performance Test

Module	Description	Result
Define the maximum amount of load that a system can handle	Verify that when 10,000 Users press A,B,C buttons pressed simultaneously, the designed combination of three sound tone is appeared	When buttons ABC are pressed simultaneously the tune combined of three sounds should appeared
The number of concurrent user that application can handle	Verify that when 10,000 User concurrently press Buttons A, there is not drop in functionality and sound quality.	When 10,000 User concurrently press Button Ait should be not affect the functionality or sound quality
Check application scalability	Verify that during the Device OS/Firmware/ Native App upgrades application can run without drop in performance	When Device OS/or Phone Firmware/or Phone Native App upgrades occurs the application runs without significant performance degradation

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Smoke TEST aka “build verification test”

Performed after software build to ascertain that the critical functionalities of the program is working fine.

Executed "before" any detailed functional or regression tests

The purpose is to reject a badly broken application, so that the QA team does not waste time installing and testing the software application.

Example :

a smoke test may address basic questions like **"Does the app run?", "Does it open a page ?"**, or **"Does tapping on the home key do anything?"**

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Sanity TEST aka "tester acceptance test"

After receiving a software build, with minor changes in code, or functionality, Sanity testing is performed

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.

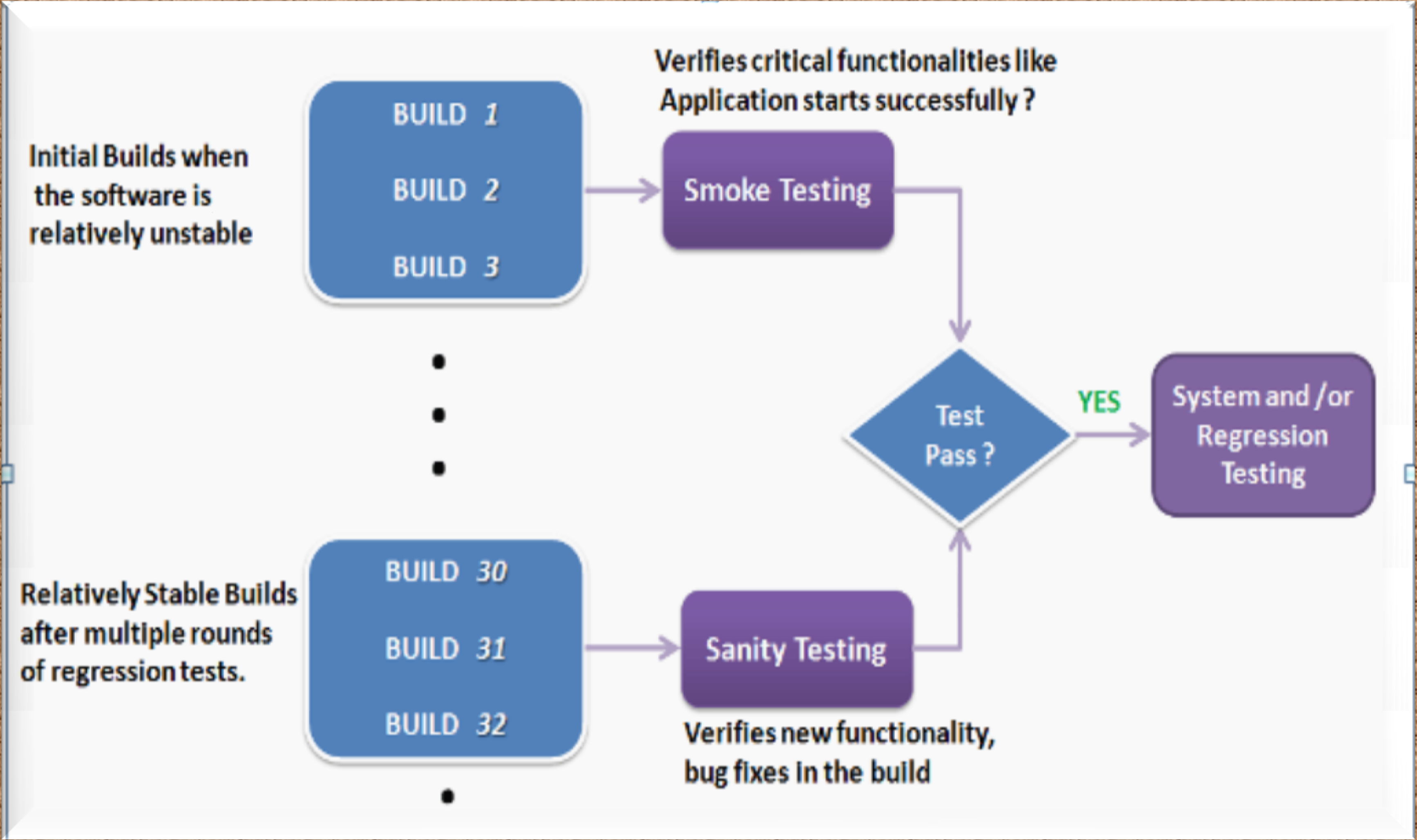
Example :

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

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Smoke vs Sanity TEST



Verifies critical functionalities like Application starts successfully ?
Initial Builds when the software is relatively unstable
BUILD 1
BUILD 2
BUILD 3
Smoke Testing
Test Pass ?
YES
System and /or Regression Testing
Relatively Stable Builds after multiple rounds of regression tests.
BUILD 30
BUILD 31
BUILD 32
Sanity Testing
Verifies new functionality, bug fixes in the build

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Comparison SUMMARY

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 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
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

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Smoke and Sanity TEST Checklist –UI

1. APP/Webpage title as per the page's functionality.

2. Spelling/ grammatical mistake (e.g. Text, Caption, Label).

3. Proper field alignment (Left margin, right margin, bottom margin, top margin).

4. Same font size/style or as per the requirement.

5. Proper space between texts, text lines, fields.

6. Standard format and size of button.

7. Textbox: Border, alignment, size, length, Data Type.

8. Combo box: Size, alignment, showing valid value.

9. Date picker (Not by keyboard, from date to date range).

10. Mandatory field identified with an identification like (*) sign.

11. Image length, size, alignment

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Smoke and Sanity TEST Checklist –Functional

1. Mandatory and composite field validation.

2. Error message not mandatory for optional field.

3. Numeric field does not accept the alpha numeric and proper error message display.

4. Max length checking for specific input field (e.g. Credit card number, Account number).

5. Confirmation message for Insert/update/ delete operations.

6. Correct format of amount value.(Should be numeric)

7. Uploaded documents are opened and generated properly.

8. Validation (Equivalence partitioning/Boundary value analysis/Positive testing/Negative/Page Refreshing).

9. System works properly with multiple browsers.

10. Pagination works and number shows properly.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Smoke and Sanity TEST Checklist –Database

1. Database name, Tables, columns name, column types matches according to requirement.

2. Data saves properly into the database after the each page submission.

3. Data display on the front end and make sure it is same in the back end.

4. Is any difference between Live and Test environment (Database Name, Table Name, Column Name, Data Type, Entity Relationship Key – Primary, Foreign, Unique key)

5. Checking Procedure/Function Create/Update related information(Entity Name, Author, Create/Update Date, Description/Purpose)

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps

Smoke and Sanity TEST Checklist –Security

1. Session timeout checking. Whether the page is expiring after the specific time.

2. Does the page browse if I paste it in a newly open browser?

3. Browser back-forward button checking if the page consist any calculation or information submission.

4. Does the browser's back-forward button re-submit the page?

5. Does this application has admin/user log in the database?

6. Password, Account number, credit card number display in encrypted format.

7. Access the secured App/web page directly without login

8. User account gets locked out if the user is entering the wrong password several times.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps : GAMES

Mobile game testing differs from the regular mobile app testing.

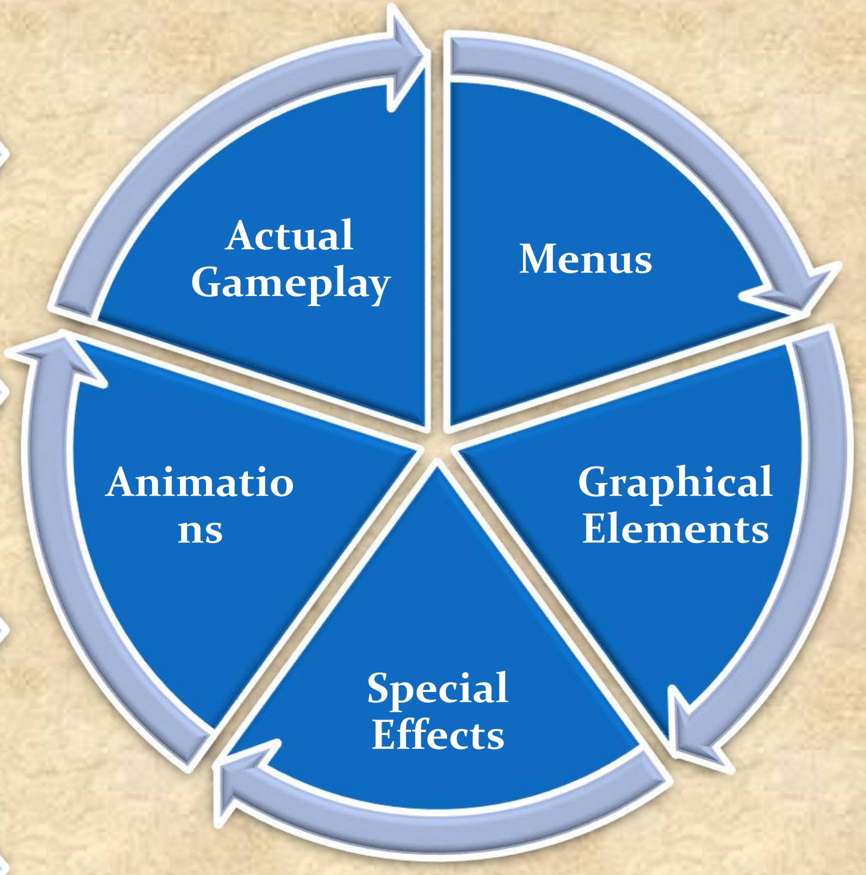
Effective mobile game testing derives from a well-structured and systematic approach, use of test automation framework and seamless integration with your agile process.

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps : GAMES

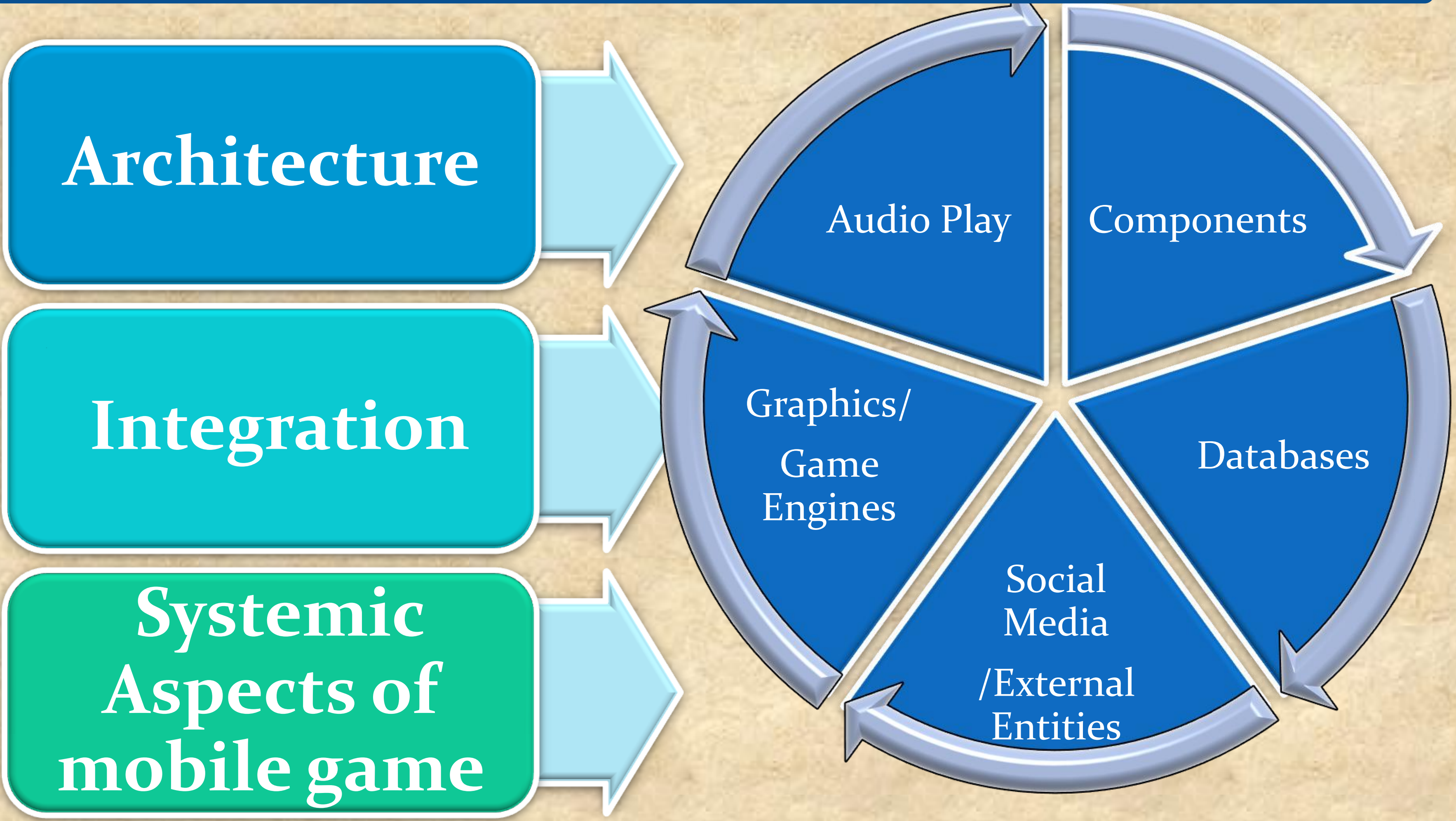
Black-Box Testing Approach focuses on :

- Functional and overall Playability aspects of the game
- GUI
- User Experience
- Any Visual Appearance



Mobile Test Industry Standards :
Testing Strategies for Mobile Apps : GAMES

White Box Testing Approach focuses on :



Mobile Test Industry Standards :

Testing Strategies for Mobile Apps : GAMES

Test Type	Purpose
Functional	reveal issues related to user interface (and graphics), stability, game flow/mechanism, and integration of graphics assets.
Compatibility	reveal incompatibility issues with any parts of the game, its third-party components or integrations with those actual devices that end-users use.
Performance	important to understand how used device ecosystem varies and what are actual requirements of the game for its users.
Localization	your game titles, texts and content needs to be translated and tested with devices in multiple languages.
Regression	needs to happen when anything changes in software : server-client interaction, requiring a login, uploading of data (e.g. results) and downloading of data (e.g. data, images).
Load	tests the limits of a system, such as the number of players on a server, the graphic content on the screen (e.g. frames per second, FPS), or memory consumption (allocation and deallocation of it).

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps : GAMES

KEY AREAS IN MOBILE GAME TESTING

- User Interface and Functionality
- Graphics Performance
- Usability and User Experience
- Multi-player/User Features
- Social Integrations
- Security and Liabilities

Mobile Test Industry Standards :
Testing Strategies for Mobile Apps : Banking

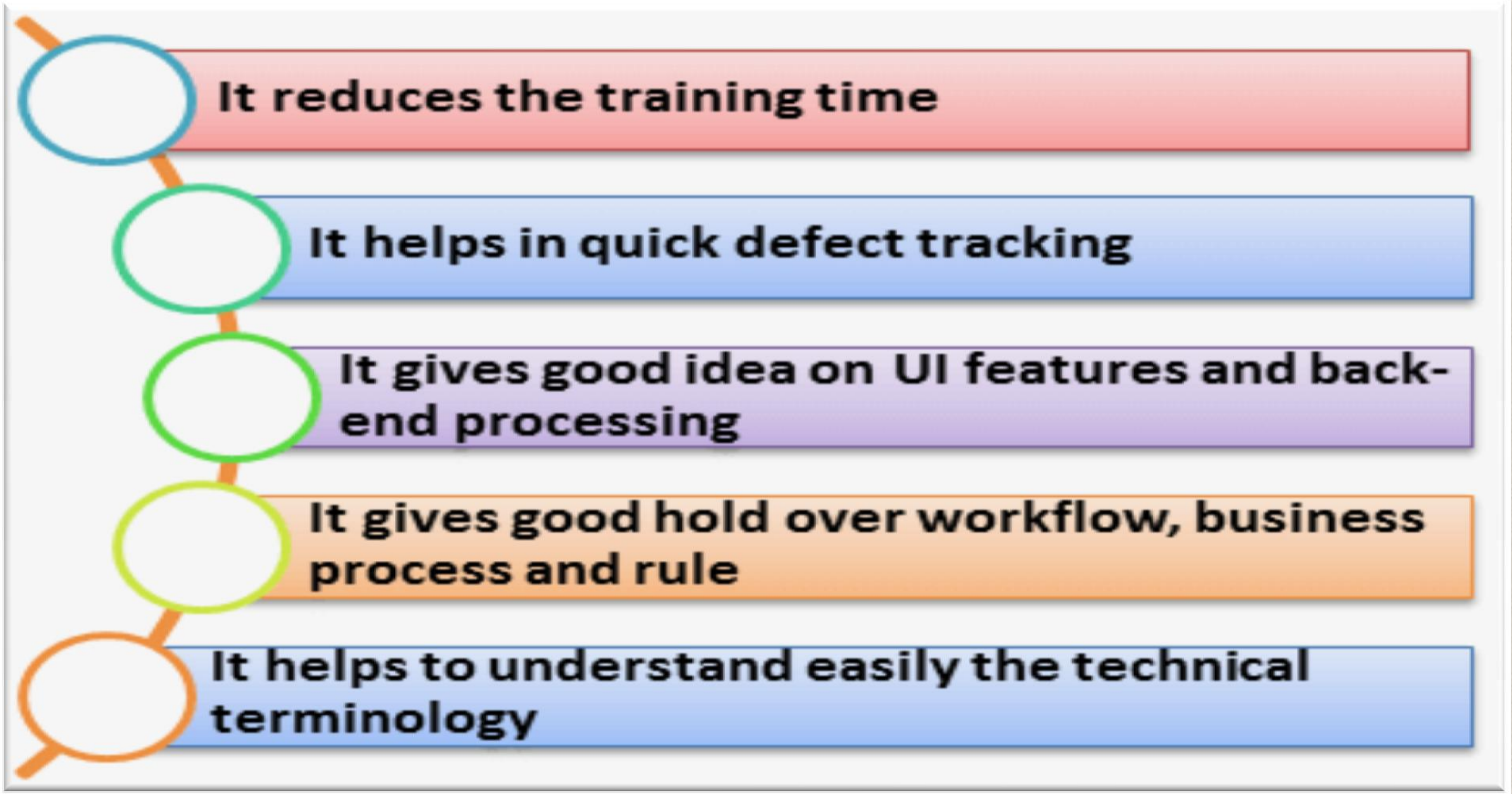
Banking applications are considered to be one of the most complex applications in development and testing industry.

What makes Banking application so complex?

What approach should be followed in order to test the complex workflows involved?

Mobile Test Industry Standards :
Testing Strategies for Mobile Apps : Banking

Why Domain Knowledge Matters?



It reduces the training time

It helps in quick defect tracking

It gives good idea on UI features and back-end processing

It gives good hold over workflow, business process and rule

It helps to understand easily the technical terminology

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps : Banking

BANKING DOMAIN in TESTING

Traditional Banking sector

- Core Banking
- Corporate Banking
- Retail Banking

Service based Banking sector

- Core
- Corporate
- Retail
- Loan
- Trade Finance
- Private Banking
- Consumer Finance
- Islamic Banking
- Customer Delivery Channels/Front End Delivery

Mobile Test Industry Standards : Testing Strategies for Mobile Apps : Banking

12 most important characteristics of a Banking application

- It should support thousands of concurrent user sessions
- A banking application should integrate with other numerous applications like trading accounts, Bill pay utility, credit cards, etc.
- It should process fast and secure transactions
- It should include massive storage system.
- To troubleshoot customer issues it should have high auditing capability
- It should handle complex business workflows
- Need to support users on multiple platforms (Mac, Linux, Unix, Windows)
- It should support users from multiple locations
- It should support multi-lingual users
- It should support users on various payment systems (VISA, AMEX, MasterCard)
- It should support multiple service sectors (Loans, Retail banking etc.)
- Foolproof disaster management mechanism

Mobile Test Industry Standards : Testing Strategies for Mobile Apps : Banking

Banking applications have multiple tiers involved in performing an operation. For Example, a banking application may have:

- Web Server to interact with end users via Browser
- Middle Tier to validate the input and output for web server
- Data Base to store data and procedures
- Transaction Processor which could be a large capacity Mainframe or any other Legacy system to carry out Trillions of transactions per second.

- Requirement Analysis
- Requirement Review
- Business Reqs Documentation
- Database Testing
- Integration Testing
- Functional Testing
- Security Testing
- Usability Testing
- User Acceptance Testing

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps : Banking Security

Many banks failed when it came to proper SSL encryption, authentication and secure feature implementation.

90% of tested apps initiated connections without proper SSL encryption

70% didn't have alternative authentication solutions

50% used an iOS featured called UIWebView (designed to display web content in native apps) insecurely

40% didn't validated the authenticity of digital certifications received from a server

20% were complied without using features designed to limit the risk of memory corruption attacks

Many apps exposed sensitive information through iOS system logs and crash logs

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps : Banking Security

	Sample Test Cases For Banking Application (OVERVIEW)
1	Verify that user is able to login with valid username and password
2	Verify that user is able to perform basic financial transactions
3	Verify that user is able to add a beneficiary with valid name and account details
4	Verify that user is able to make financial transactions to added beneficiary
5	Verify that user is able to add decimal number into amount (limited by 2 numbers)
6	Verify that user is not able to add negative number into amount field
7	Verify that user is allowed to transfer money only if there is proper account balance.
8	Verify that there is a confirmation check for financial transactions

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps : Banking Security

	Sample Test Cases For Banking Application
9	Verify that user is given an acknowledgment receipt upon successful financial transaction.
10	Verify that customer is able to send money to multiple people
11	Verify that user is allowed to change password
12	Verify that account details reflect financial transactions also.
13	Verify that user with invalid password is not allowed to login.
14	Verify that after repeated attempts to login with incorrect password(as per the limits), user should be blocked.
15	Verify that time-out feature is implemented
16	Verify that if either of the username or password is blank, user is not allowed to login. User should be given an alert also.

Mobile Test Industry Standards :

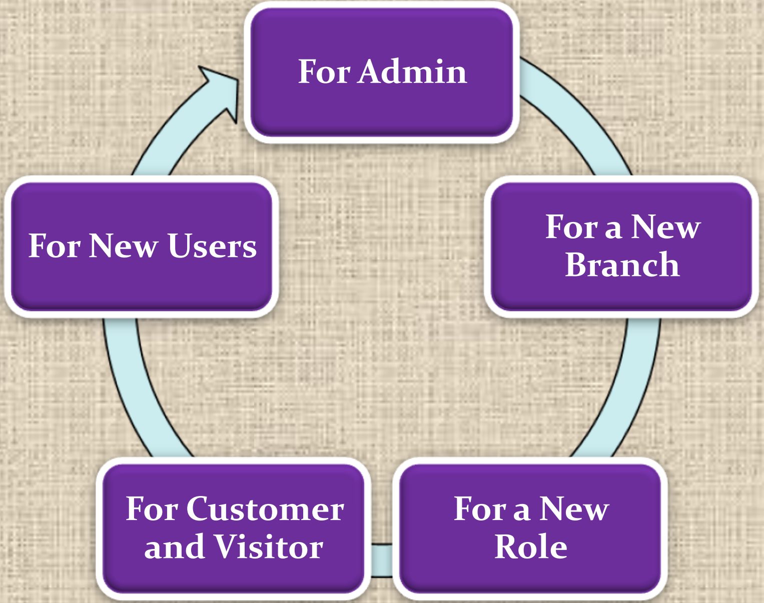
Testing Strategies for Mobile Apps : Banking Security

	Challenges faced by QA
1	Getting access to production data and replicating it as test data, for testing is challenging
2	The biggest challenge in testing banking system is during the migration of the system from the old system to the new system like testing of all the routines, procedures and plans. Also how the data will be fetched, uploaded and transferred to the new system after migration
3	There may be the cases where requirements are not documented well and may lead to functional gaps in test plan Many non-functional requirements are not fully documented, and testers do not know whether to test it or not
4	The most important point is to check whether the said system follows the desired policies and procedures
5	The scope and the timelines increases as banking application are integrated with other application like internet or mobile banking

Mobile Test Industry Standards :

Testing Strategies for Mobile Apps : Banking Security

Guidance For Banking Application: Scope



SUMMARY

- Majority of banking software are developed on Mainframe and Unix
- Testing helps to lessen possible glitches encounter during software development
- Proper testing and compliance to industry standards, save companies from penalties
- Good practices help develop good results, reputation and more business for companies
- Both manual and automated testing have respective merits and usability