

Android Application Development

For Aptech Centers Use Only

Android Application Development

Trainer's Guide

© 2016 Aptech Limited

All rights reserved.

No part of this book may be reproduced or copied in any form or by any means – graphic, electronic or mechanical, including photocopying, recording, taping, or storing in information retrieval system or sent or transferred without the prior written permission of copyright owner Aptech Limited.

All trademarks acknowledged.

APTECH LIMITED

Contact E-mail: ov-support@onlinevarsity.com

First Edition - 2016



Dear Learner,

We congratulate you on your decision to pursue an Aptech Worldwide course.

Aptech Ltd. designs its courses using a sound instructional design model – from conceptualization to execution, incorporating the following key aspects:

➤ **Scanning the user system and needs assessment**

Needs assessment is carried out to find the educational and training needs of the learner.

Technology trends are regularly scanned and tracked by core teams at Aptech Ltd. TAG* analyzes these on a monthly basis to understand the emerging technology training needs for the Industry.

An annual Industry Recruitment Profile Survey[#] is conducted during August - October to understand the technologies that Industries would be adapting in the next 2 to 3 years. An analysis of these trends & recruitment needs is then carried out to understand the skill requirements for different roles & career opportunities.

The skill requirements are then mapped with the learner profile (user system) to derive the Learning objectives for the different roles.

➤ **Needs analysis and design of curriculum**

The Learning objectives are then analyzed and translated into learning tasks. Each learning task or activity is analyzed in terms of knowledge, skills and attitudes that are required to perform that task. Teachers and domain experts do this jointly. These are then grouped in clusters to form the subjects to be covered by the curriculum.

In addition, the society, the teachers, and the industry expect certain knowledge and skills that are related to abilities such as learning-to-learn, thinking, adaptability, problem solving, positive attitude etc. These competencies would cover both cognitive and affective domains.

A precedence diagram for the subjects is drawn where the prerequisites for each subject are graphically illustrated. The number of levels in this diagram is determined by the duration of the course in terms of number of semesters etc. Using the precedence diagram and the time duration for each subject, the curriculum is organized.

➤ **Design & development of instructional materials**

The content outlines are developed by including additional topics that are required for the completion of the domain and for the logical development of the competencies identified. Evaluation strategy and scheme is developed for the subject. The topics are arranged/organized in a meaningful sequence.

The detailed instructional material – Training aids, Learner material, reference material, project guidelines, etc.- are then developed. Rigorous quality checks are conducted at every stage.

➤ **Strategies for delivery of instruction**

Careful consideration is given for the integral development of abilities like thinking, problem solving, learning-to-learn etc. by selecting appropriate instructional strategies (training methodology), instructional activities and instructional materials.

The area of IT is fast changing and nebulous. Hence, considerable flexibility is provided in the instructional process by specially including creative activities with group interaction between the students and the trainer. The positive aspects of Web based learning –acquiring information, organizing information and acting on the basis of insufficient information are some of the aspects, which are incorporated, in the instructional process.

➤ **Assessment of learning**

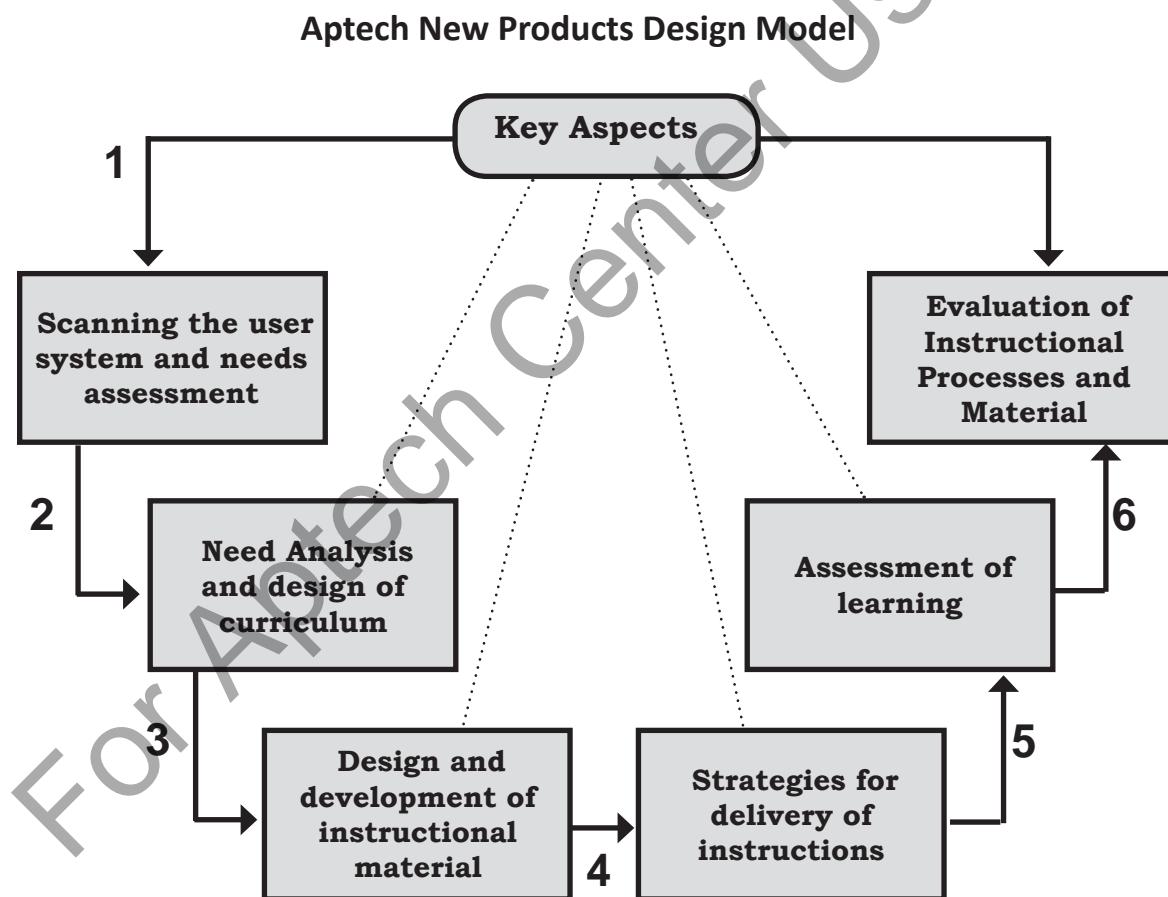
The learning is assessed through different modes – tests, assignments & projects. The assessment system is designed to evaluate the level of knowledge & skills as defined by the learning objectives.

➤ **Evaluation of instructional process and instructional materials**

The instructional process is backed by an elaborate monitoring system to evaluate - on-time delivery, understanding of a subject module, ability of the instructor to impart learning. As an integral part of this process, we request you to kindly send us your feedback in the reply pre-paid form appended at the end of each module.

*TAG – Technology & Academics Group comprises of members from Aptech Ltd., professors from reputed Academic Institutions, Senior Managers from Industry, Technical gurus from Software Majors & representatives from regulatory organizations/forums.

Technology heads of Aptech Ltd. meet on a monthly basis to share and evaluate the technology trends. The group interfaces with the representatives of the TAG thrice a year to review and validate the technology and academic directions and endeavors of Aptech Ltd.



“

A little learning is a dangerous thing,
but a lot of ignorance is just as bad

”

For Aptech Center Use Only

Preface

The book ‘Android Application Development’ Trainer’s Guide’ serves understanding on new features and functionalities of Android 5.0 Lollipop and Android 6.0 Marshmallow. The faculty/trainer should teach the concepts in the theory class using the slides. This Trainer’s Guide will provide guidance on the flow of the session and also provide tips and additional examples wherever necessary. The trainer can ask questions to make the session interactive and also to test the understanding of the students.

This book is the result of a concentrated effort of the Design Team, which is continuously striving to bring you the best and the latest in Information Technology. The process of design has been a part of the ISO 9001 Certification for Aptech-IT Division, Education Support Services. As part of Aptech’s quality drive, this team does intensive research and curriculum enrichment to keep it in line with industry trends.

We will be glad to receive your suggestions.

Design Team

“

Practice is the best of
all instructors.

For Aptech Center Use Only

Table of Contents

Sessions

1. Introduction to Android
2. Getting Started with Android
3. Android System Overview
4. Android User Interface (UI)
5. More UI Elements
6. Media Handling
7. Data Handling and Content Providers
8. Services, Broadcast Receivers, and Intent Filters
9. Google API
10. Web Services in Android
11. Wireless and Networking
12. Telephony, SMS, and VoIP
13. Sensors
14. Google Play Store
15. Android Studio 2.1 and Android Marshmallow 6.0
16. Material Design
17. Android Interface Definition Language
18. Android Native Development Kit (NDK)

“ The future depends on what
we do in the present.”

For Aptech Center Use Only

Session 1: Introduction to Android

1.1 Pre-Class Activities

You should familiarize yourself with operating systems and acquired good understanding of Linux and the internals of Android. You should study the architecture and history of Android. You practice the steps to setup the development environment.

1.1.1 Objectives

After the session, learners will be able to:

- Explain Android
- Explain the history of Android
- Describe the architectural framework of Android
- Explain the downloading and installation process of Android SDK and IDEs

1.1.2 Teaching Skills

You should be familiar with the Android and its history. You should be able to explain the concepts of Operating Systems. You should know the IDEs available for Android development. Additionally, you should know the process for downloading and installing the IDEs and setting up the development environment.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

1.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the basics of Android and Android application development.

1.2 In-Class Explanations

Introduction

Slide 3

Introduction

- An Operating System is a software program that enables communication and utilization of the hardware resources by the software programs.
- Each OS can be broadly divided into:
 - Application Layer
 - Kernel Layer
 - Hardware and device driver layers
- Kernel translates requests from one layer to another.
- Android is a Mobile Operating System.
- Mobile market is dominated by Android and iOS.
- Applications are OS specific.



© Aptech Ltd. Introduction to Android/Session 1 3

Using slide 3, give an outline of what is going to be covered in the session. Define the term Operating System. Explain the components of an OS and the functionality provided by each of them. List the features of Android.

Introduction to Android

Slides 4 to 8

Introduction to Android 1-5

- **What is Android ?**
 - A Mobile OS running on the Linux Kernel.
 - Android applications are written in Java.
 - Applications run on a virtual machine.
 - Dalvik VM was used until Android Kitkat.
 - Lollipop introduces Android Run Time (ART).
- **History of Android**
 - Android 1.0 released in 2008.
 - Latest version is Android Lollipop released in November 20, 2014.
 - Several improvements made over years.
 - User base has grown exponentially.



© Aptech Ltd. Introduction to Android/Session 1 4

Using slide 4, explain the Android operating system. Define Virtual Machine (VM) and explain its importance. Mention the previous VM used by Android and convey the fact that it has been replaced with ART.

Additional Reference:

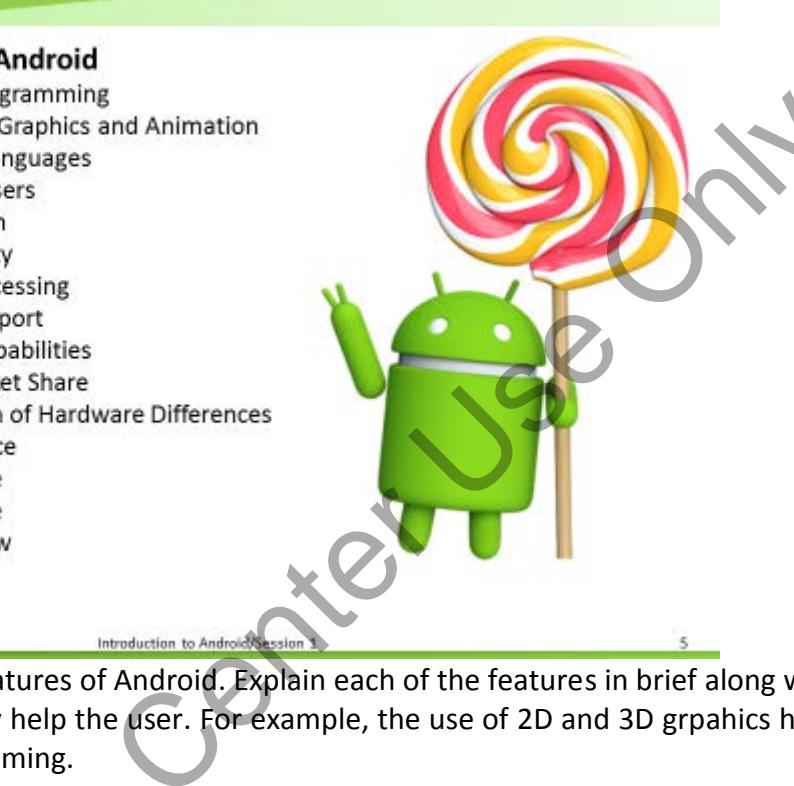
You can refer to the following link(s) for more information regarding the history of Android:

<https://www.android.com/history/>

https://en.wikipedia.org/wiki/Android_version_history

Introduction to Android 2-5

- ◆ **Features of Android**
 - ◆ Ease of Programming
 - ◆ 2D and 3D Graphics and Animation
 - ◆ Multiple Languages
 - ◆ Web Browsers
 - ◆ Multi Touch
 - ◆ Connectivity
 - ◆ Media Processing
 - ◆ Sensor Support
 - ◆ Storage Capabilities
 - ◆ Large Market Share
 - ◆ Abstraction of Hardware Differences
 - ◆ Open Source
 - ◆ Free to Use
 - ◆ Ease of Use
 - ◆ Google Now




© Aptech Ltd.

Introduction to Android/Session 1

5

Using slide 5, list out the features of Android. Explain each of the features in brief along with small examples of how they help the user. For example, the use of 2D and 3D grpahics have lead to the rise of mobile gaming.

Introduction to Android 3-5

- ◆ **New Features in Android Lollipop**
 - ◆ Improved Notification System
 - ◆ Optimized Battery consumption
 - ◆ OpenGL ES 3.1
 - ◆ New Virtual Machine – ART
 - ◆ UI Overhaul
 - ◆ 'Google Now' became a full fledged personal assistant



© Aptech Ltd.

Introduction to Android/Session 1

6

Using slide 6, list the new features introduced in Android Lollipop. Provide a brief

explanation of each of the feature and how it enhances the user experience. For example, for Google Now, you can explain the convinience of having a digital personal assistant and voice commands to get things done more efficiently.



In-Class Question: What are the various features supported by Android?

Answer: The various features supported by Android are as follows:

- Ease of Programming
- 2D and 3D Graphics and Animation
- Multiple Languages
- Web Browsers
- Multi Touch
- Connectivity
- Media Processing
- Sensor Support
- Storage Capabilities
- Large Market Share
- Abstraction of Hardware Differences
- Open Source
- Free to Use
- Ease of Use
- Google Now

Introduction to Android 4-5

◆ Device types running Android

- ◆ Smartphones
- ◆ Tablets
- ◆ Televisions
- ◆ Cars and In-Vehicle Entertainment
- ◆ Android Wear Devices
- ◆ Google Glass



Using slide 7, list the device types running Android. Use the images to identify the devices. Explain the importance of Android for each of these devices. For example, Android for smart watches has eliminated the need to constantly interact with the mobile device. Android on TVs has enhanced the TV experience and led to the rise of 'Smart TVs'.

Introduction to Android 5-5

- ◆ Challenges to Developing for Android

- ◆ Hardware Fragmentation
- ◆ Software Fragmentation
- ◆ Lack of Hardware Software Integration Standards
- ◆ Lack of Quality Control on Android devices

- ◆ Android and Open Source platform

- ◆ Android is developed privately by Google.
- ◆ Later, source code is made available to everyone.
- ◆ Anyone can make changes.
- ◆ Huge community of mod'ers and developers.
- ◆ Custom ROMs available to most devices.



Using slide 8, explain the various challenges to developing for Android. You may compare them to other platforms such as iOS or WP. Explain the concept of Open Source and the benefits of the model. Explain the community involvement and how it benefits the end user and the developer.

Additional Reference:

You can refer to the following link(s) for more information on how Android compares to other platforms:

<http://blog.bugsense.com/post/26918984231/android-vs-windows-phone-from-a-developers-scope>

<http://www.aurosyssolutions.com/mobile-application-development-android-vs-ios-vs-windows-developers-view/>



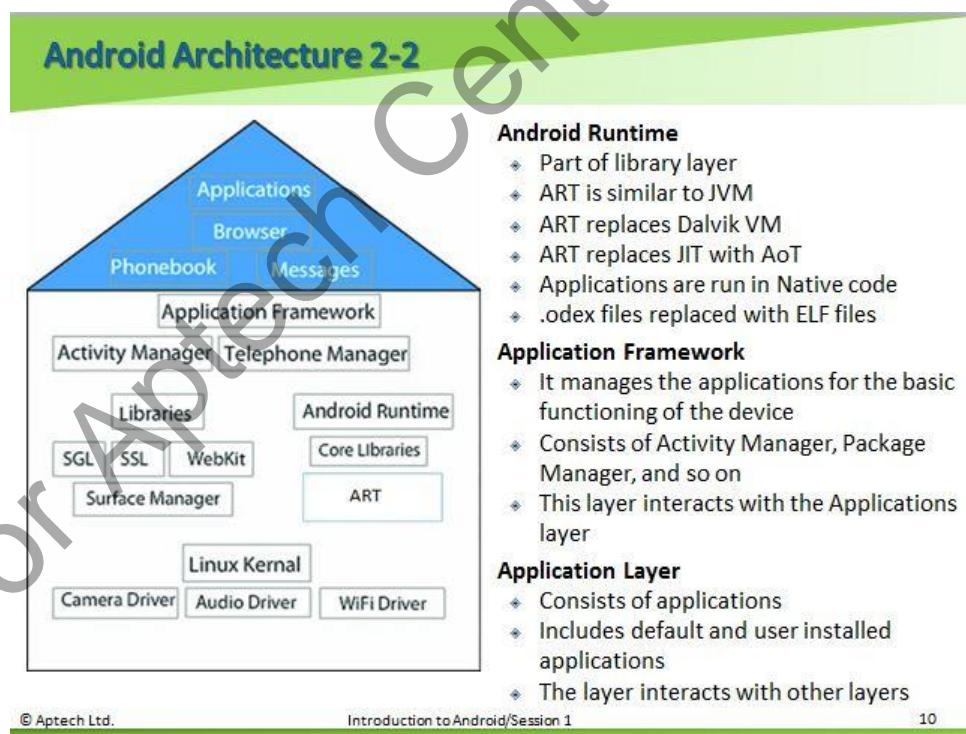
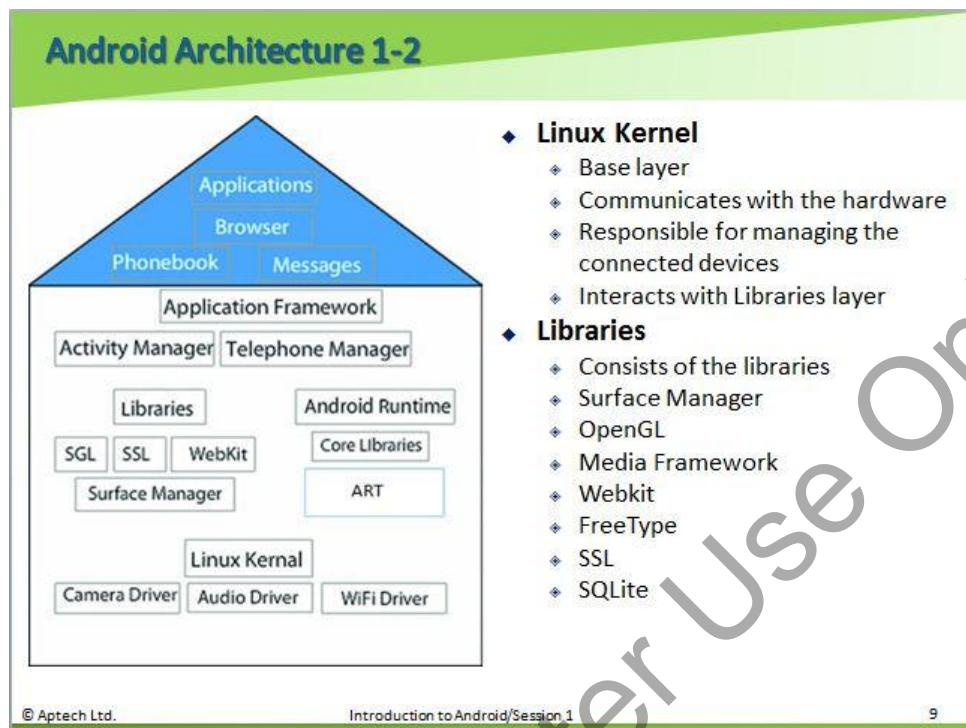
In-Class Question: What are the devices capable of running Android?

Answer: The devices capable of running Android are as follows:

- Mobiles
- Tablets
- Televisions
- Cars and In Vehicle Entertainment
- Android Wear Devices
- Google Glass

Android Architecture

Slides 9 and 10



Using slides 9 and 10, explain the Android architecture. Using the diagram provided, explain each of the layer and its functionality. Explain the fact that each layer communicates with the neighboring layers. Use the bulleted points to emphasize on the important information.

Additional Reference:

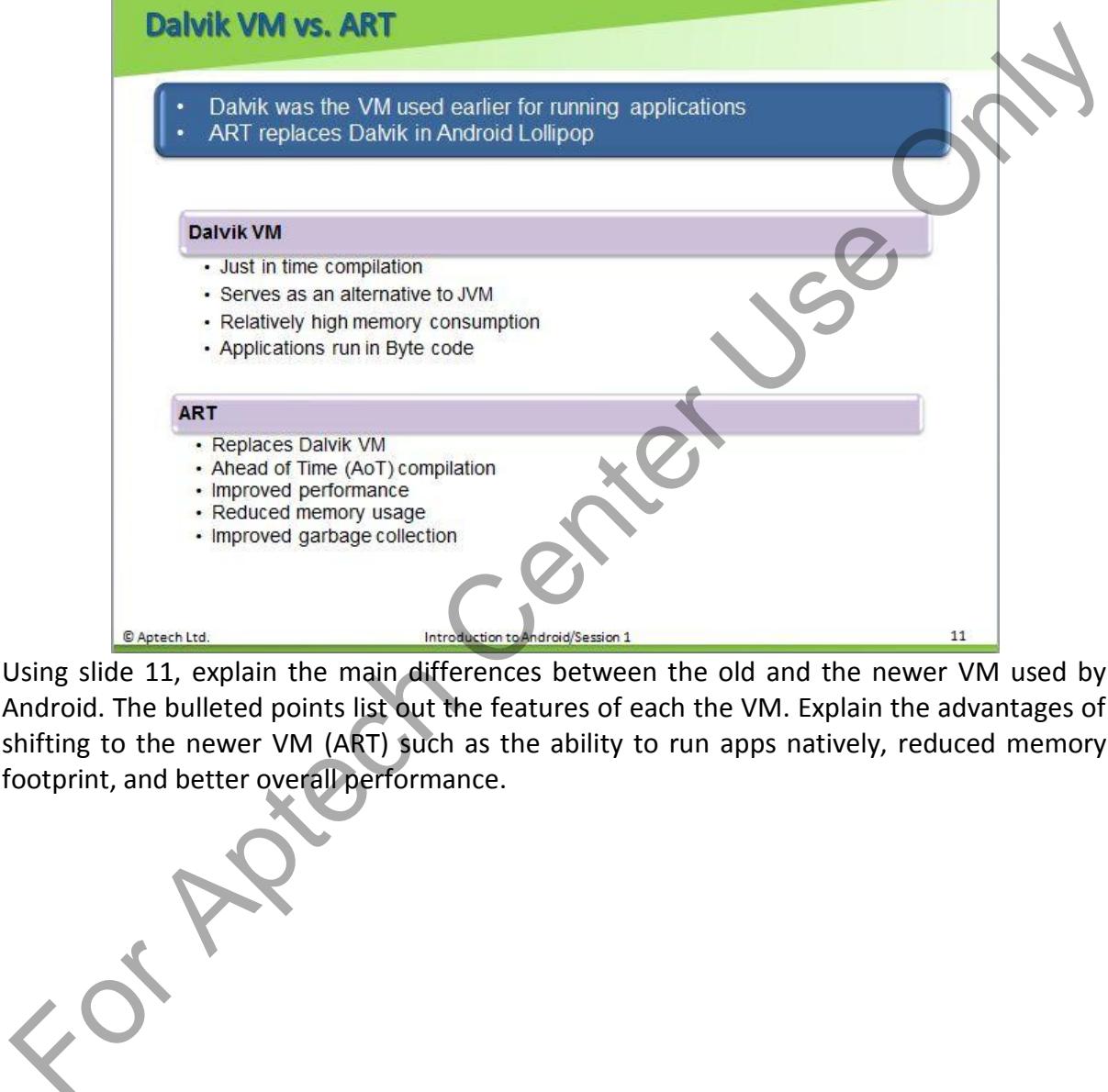
You can refer to the following link(s) for more information regarding Android Architecture:

<http://source.android.com/devices/>

<http://www.eazytutz.com/android/android-architecture/>

Dalvik VM vs. ART

Slide 11



Dalvik VM vs. ART

- Dalvik was the VM used earlier for running applications
- ART replaces Dalvik in Android Lollipop

Dalvik VM
<ul style="list-style-type: none">• Just in time compilation• Serves as an alternative to JVM• Relatively high memory consumption• Applications run in Byte code

ART
<ul style="list-style-type: none">• Replaces Dalvik VM• Ahead of Time (AoT) compilation• Improved performance• Reduced memory usage• Improved garbage collection

© Aptech Ltd. Introduction to Android/Session 1 11

Using slide 11, explain the main differences between the old and the newer VM used by Android. The bulleted points list out the features of each the VM. Explain the advantages of shifting to the newer VM (ART) such as the ability to run apps natively, reduced memory footprint, and better overall performance.

Google Play Platform

Slide 12

Google Play Platform

- ◆ It was originally called Android Market
- ◆ Play Store is the official market place for Android Applications
- ◆ It comes pre-installed in almost all Android Devices
- ◆ Primary source for installing applications
- ◆ Millions of customers
- ◆ Available in 135 countries
- ◆ Music, Videos, Movies, and Books are also purchasable


Google play

© Aptech Ltd. Introduction to Android/Session 1 12

Using slide 12, explain Google Play and how it benefits the developer. List the features specified in bulleted points and explain its importance. You may compare it to other market places such as the iTunes or the WP Store and explain how Google Play differs from them.

Additional Reference:

You can refer to the following link(s) for more information about Google Play and how it compares to other app stores:

https://en.wikipedia.org/wiki/Google_Play

<https://play.google.com/store?hl=en>

<http://www.androidauthority.com/google-play-store-vs-the-apple-app-store-601836/>

<http://www.androidauthority.com/google-play-vs-windows-phone-marketplace-2012-78170/>



In-Class Question: List the differences between Dalvik and ART.

Answer: The differences between Dalvik and ART are as under:

Dalvik

- Just-In-Time compilation.
- Serves as an alternative to JVM.
- Relatively high memory consumption.
- Applications run in Byte code.

ART

- Ahead of time compilation.
- Replaces Dalvik VM.
- Improved performance.
- Reduced memory usage.

Android Software Development Kit (SDK)

Slides 13 to 15

Android Software Development Kit (SDK) 1-3

- ◆ A SDK is a package that consists of all the tools required for developing applications
- ◆ Android SDK comes with:
 - ◆ Documentation
 - ◆ Libraries
 - ◆ Sample Code
 - ◆ Emulator
 - ◆ Guides
 - ◆ Android Studio IDE (Bundle Only)
- ◆ Android Studio is Google's official IDE for developing Android Applications
- ◆ Eclipse can also be used as an IDE for Android development



Using slide 13, explain the concept of an SDK. You can begin by explaining that an IDE is the abbreviation for Integrated Development Environment and is primarily useful for writing code for the application, debugging, and running the application. Proceed to explain that the SDK contains all the tools for development including the IDE, the compiler, libraries, and anything else necessary. List the various components of the Android SDK and explain each of them in brief. Explain how the two IDEs namely, Eclipse and Android Studio differ from each other.

Additional Reference:

You can refer to the following link(s) for more information regarding the Android SDK:

<https://developer.android.com/sdk/index.html>

Android Software Development Kit (SDK) 2-3

- The software requirements to install and use the Android SDK are given in the following table:

Requirement	Description
Operating System	<ul style="list-style-type: none"> • Windows 2003 (32-bit), Windows XP (32-bit), Vista (32- or 64-bit) or Windows 7 (32- or 64-bit). • Mac OS X 10.5.8 or later (x86 only). • Linux.
Development Environment	<ul style="list-style-type: none"> • Optional if using Android Studio: <ul style="list-style-type: none"> ✓ Eclipse 4.0 (Juno) or higher. ✓ Eclipse Java Development Tools (JDT) plug-in. <ul style="list-style-type: none"> ❖ Eclipse IDE. ❖ ADT plug-in. • Mandatory: JDK 7 or higher - Java Runtime Environment (JRE) alone is not sufficient.

Android Software Development Kit (SDK) 3-3

- The hardware requirements to install and use the Android SDK are shown in the following table:

Requirement	Description
Hardware Requirements	<ul style="list-style-type: none"> • Minimum 2 Gigabyte (GB) of Random Access Memory (RAM). • Intel Core2Due or equal processor for x86 architecture. • 4 GB hard disk space.

Using slides 14 and 15, list the software and hardware requirements for the Android SDK. Ensure that all the requirements are met on the target systems.

Installing JDK

Slides 16 to 21

Installing JDK 1-6

- Check system requirements
- Go to download link and download the installation file as shown in the following figure:

The screenshot shows the Oracle Java SE Downloads page. On the left, there's a sidebar with links like Java SE, Java EE, Java ME, Java SE Support, Java SE Advanced & Java, Java Embedded, Java DB, Java TEE, Java Card, Java TV, New to Java, Community, and Java Magazines. The main content area has tabs for Overview, Downloads, Documentation, Community, Technologies, and Training. Under the Downloads tab, it says 'Java SE Downloads'. It features links for Java Platform (JDK) 8u5, NetBeans, and Java Platform, Standard Edition. Below these are sections for Java EE 8u4F (with a note about security fixes), Java Platform (JDK) 8u5, and Server JRE.

© Aptech Ltd. Introduction to Android/Session 1 16

Installing JDK 2-6

- Start the installation running the downloaded setup file.
- Click Next to proceed with the installation as shown in the following figure:

The screenshot shows the Java SE Development Kit 8 Update 25 - Setup window. It has a red header with the Java logo and ORACLE. The main area says 'Welcome to the Installation Wizard for Java SE Development Kit 8 Update 25.' It states that the wizard will guide the user through the installation process. A note at the bottom says 'The Java Mission Control profiling and diagnostics tools suite is now available as part of the JDK.' At the bottom are 'Next >' and 'Cancel' buttons.

© Aptech Ltd. Introduction to Android/Session 1 17

Installing JDK 3-6

- Select the components as shown in the following figure:



© Aptech Ltd.

Introduction to Android/Session 1

18

Installing JDK 4-6

- Click Next
- The installation begins as shown in the following figure:



© Aptech Ltd.

Introduction to Android/Session 1

19

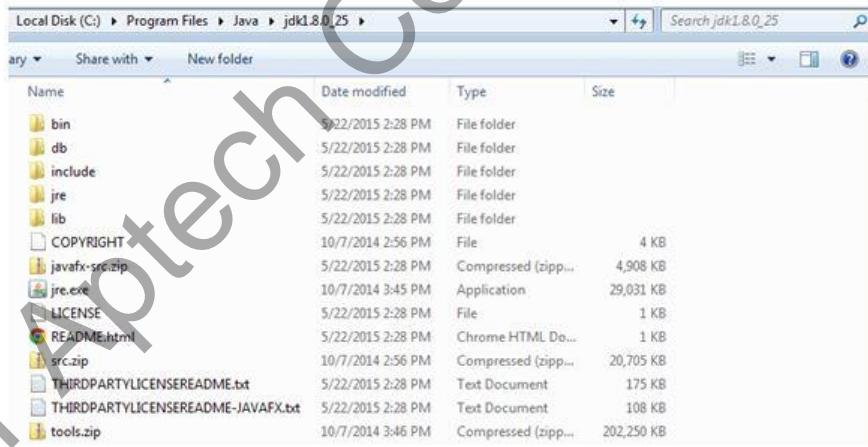
Installing JDK 5-6

- Click Close to complete the installation as shown in the following figure:



Installing JDK 6-6

- The installation folder can be opened to verify the installation as shown in the following figure:



Using slides 16 to 21, demonstrate the procedure for installing Java Development Kit (JDK). JDK contains the Java Compiler, Libraries, and the Java Virtual Machine (JVM) to run the compiled byte code. Android development is done in Java and as a Java compiler is necessary for compiling and running Java code, the JDK is mandatory. The steps in most slides are self-explanatory. Read the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Downloading Android Studio Bundle

Slides 22 to 24

Downloading Android Studio Bundle 1-3

- Go to download link and click download as shown in the following figure:

© Aptech Ltd. Introduction to Android/Session 1 22

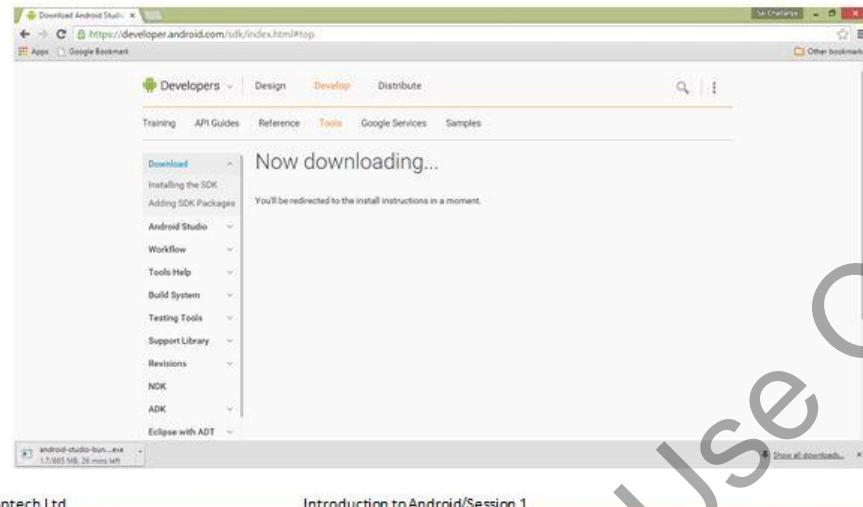
Downloading Android Studio Bundle 2-3

- In the Terms and Conditions page, select the 'I have read and agree with the above terms and conditions' check box as shown in the following figure:

© Aptech Ltd. Introduction to Android/Session 1 23

Downloading Android Studio Bundle 3-3

- Click the 'Download Android Studio for Windows' button. The download begins as shown in the following figure:



Using slides 22 to 24, demonstrate the procedure for downloading the Android Studio Bundle. You may start by explaining that Android Studio is Google's official IDE for android application development. Hence, it will receive more attention from Google and immediate bug fixes. The steps in most slides are self-explanatory. Read out the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Additional Reference:

You can refer to the following link(s) for more information regarding Android Studio:

https://en.wikipedia.org/wiki/Android_Studio

<http://developer.android.com/tools/studio/index.html>

Downloading Android SDK Standalone

Slides 25 to 27

Downloading Android SDK Standalone 1-3

- Go to download link and click download as shown in the following figure:

The screenshot shows the 'Download' section of the Android developer website. On the left, there's a sidebar with links like 'Installing the SDK', 'Adding SDK Packages', 'Android Studio', 'Workflow', 'Tools Help', etc. The main area features a large image of a laptop displaying the Android Studio interface with a smartphone on the screen. Below the image is a green button labeled 'Download Android Studio for Windows'. A grey callout box points to this button with the text 'Go to download link and click download as shown in the following figure:'.

Downloading Android SDK Standalone 2-3

- Scroll down to the Other Download Options section as shown in the following figure:

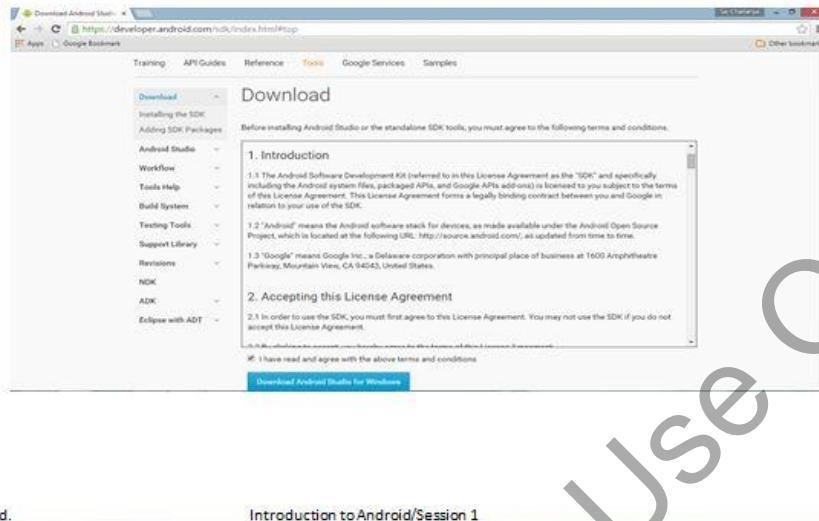
The screenshot shows the 'Other Download Options' section for the 'SDK Tools Only' package. It includes a table with columns for Platform, Package, Size, and SHA-1 Checksum. The table has three rows: one for Windows (installer/.24.2-windows.exe) and two for Mac OS X (android-sdk_r24.2-macosx.zip). Below this, there's another table for 'All Android Studio Packages' with a single row for Windows (android-studio-bundle-141.189065-windows.exe).

Platform	Package	Size	SHA-1 Checksum
Windows	installer/.24.2-windows.exe (Recommended)	107849819 bytes	e764ea93aa727667379be309fb3e42d879ab599
Mac OS X	android-sdk_r24.2-macosx.zip	155944165 bytes	2011ee9a080f4858f164e55172801714af8a1660
Linux	android-sdk_r24.2-linux.tgz	168119905 bytes	1a299827e9395a95dbd292096e3825e2dd8089

Platform	Package	Size	SHA-1 Checksum
Windows	android-studio-bundle-141.189065-windows.exe (Recommended)	928285594 bytes	470e67749409fd710c059faaff22d9191c47ae9d0

Downloading Android SDK Standalone 3-3

- In the Terms and Conditions page, select the 'I have read and agree with the above terms and conditions' check box as shown in the following figure:



© Aptech Ltd.

Introduction to Android/Session 1

27

Using slides 25 to 27, demonstrate the procedure for downloading Android SDK Standalone. The standalone package does not contain the Android Studio IDE. Hence, it can be downloaded instead, if another IDE is planned to be used. The steps in most slides are self-explanatory. Read out the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Installing Android SDK

Slides 28 to 37

Installing Android SDK 1-10

- Go to the directory where the SDK has been downloaded
- Double-click the Installer. The wizard appears as shown in the following figure:



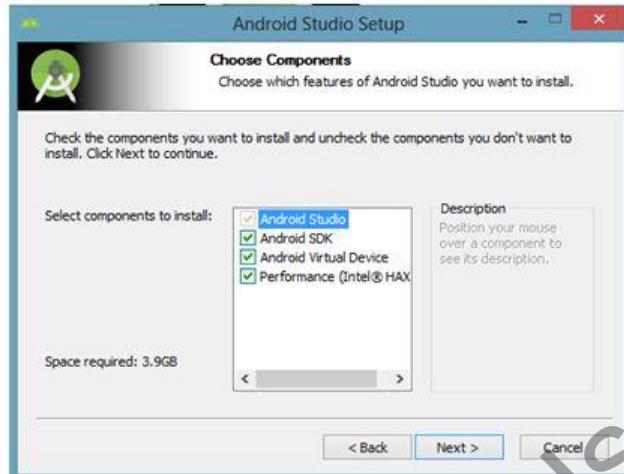
© Aptech Ltd.

Introduction to Android/Session 1

28

Installing Android SDK 2-10

- Select all the components in the components screen as shown in the following figure and click Next.



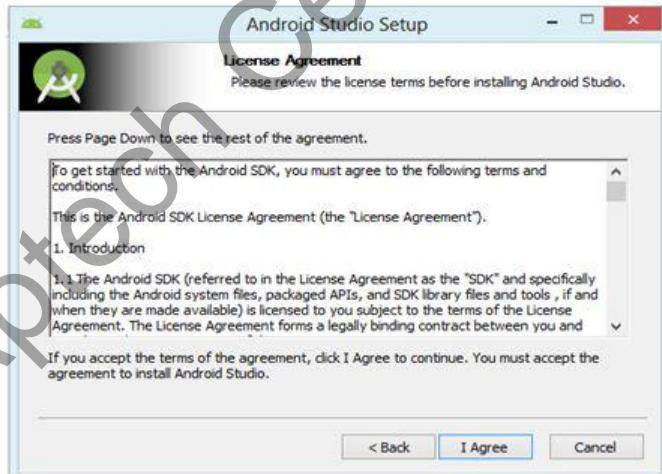
© Aptech Ltd.

Introduction to Android/Session 1

29

Installing Android SDK 3-10

- The SDK license agreement appears. Click the 'I Agree' button as shown in the following figure:



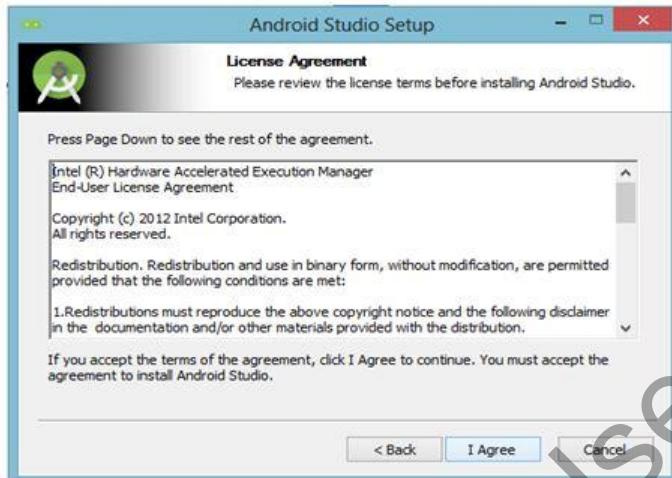
© Aptech Ltd.

Introduction to Android/Session 1

30

Installing Android SDK 4-10

- The Intel Hardware Accelerated Execution Manager License Agreement appears if the component was selected as shown in the following figure:



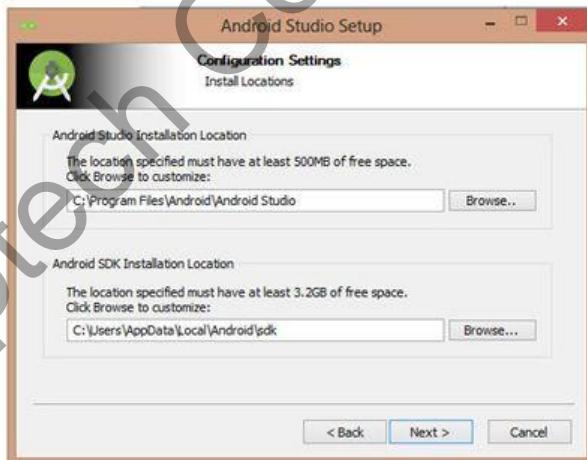
© Aptech Ltd.

Introduction to Android/Session 1

31

Installing Android SDK 5-10

- Select 'I Agree'
- The installation path as shown in the following figure:



© Aptech Ltd.

Introduction to Android/Session 1

32

Installing Android SDK 6-10

- Select the path of installation and click Next. The AVD configuration page appears as shown in the following figure:
- Select the default RAM size for the device and click Next



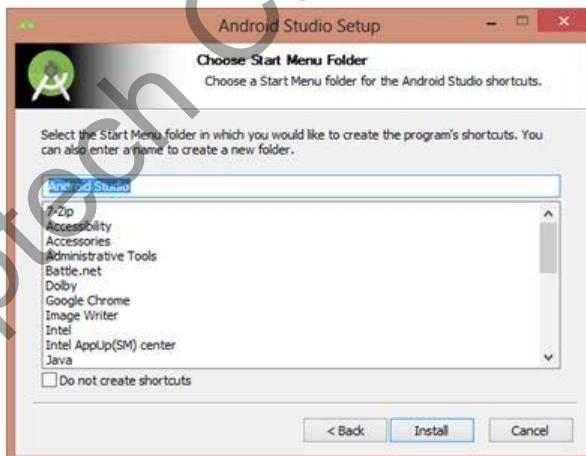
© Aptech Ltd.

Introduction to Android/Session 1

33

Installing Android SDK 7-10

- The start menu selection screen appears as shown in the following figure:
- Click Install



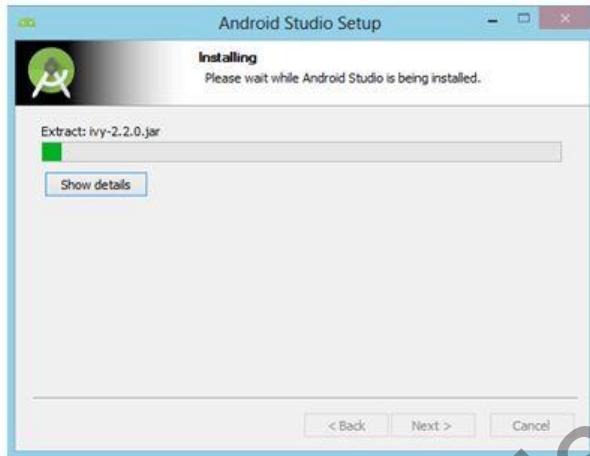
© Aptech Ltd.

Introduction to Android/Session 1

34

Installing Android SDK 8-10

- The installation begins as shown in the following figure:



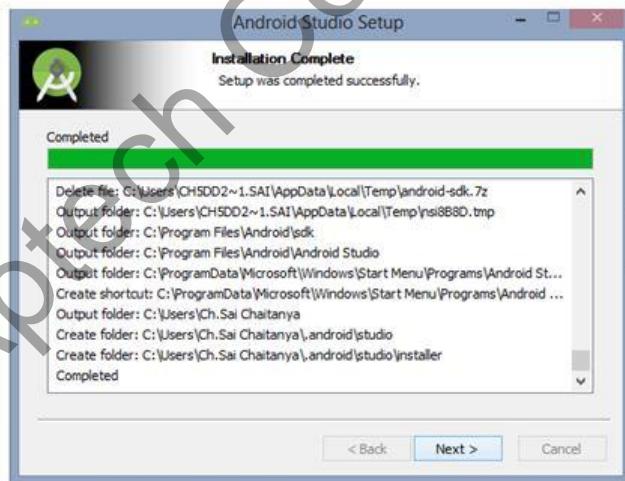
© Aptech Ltd.

Introduction to Android/Session 1

35

Installing Android SDK 9-10

- Click Next after the installation completes as shown in the following figure:



© Aptech Ltd.

Introduction to Android/Session 1

36

Installing Android SDK 10-10

- The successfully installed screen appears as shown in the following figure:



- Click Finish

© Aptech Ltd. Introduction to Android/Session 1 37

Using slides 28 to 37, demonstrate the procedure for installing Android SDK. The steps in most slides are self-explanatory. Read out the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Installing Android SDK from Zip File

Slide 38

Installing Android SDK from Zip File

- ◆ Navigate to the .zip file
- ◆ Extract contents to a known location
- ◆ Make a note of the name and location of the SDK directory
- ◆ Use the command line to use the SDK tools
- ◆ If the OS is Windows, the SDK manager GUI can be used

© Aptech Ltd. Introduction to Android/Session 1 38

Using slide 38 demonstrate the procedure for installing Android SDK from zip file. This is an alternative method of installation. The steps are self-explanatory. Read out the step and explain it in detail whenever necessary.

Note – Installing Android SDK is as simple as extracting the zip. Moreover, it is an alternative to installing the Bundle or the standalone. If they are already done, then there is no need for doing this.

Requirements for Eclipse IDE

Slide 39

Requirements for Eclipse IDE

The requirements to install and use the Eclipse IDE are as shown in the following table:

Requirement	Description
Operating System	<ul style="list-style-type: none">Windows 2003 (32-bit), Windows XP (32-bit), Vista (32- or 64-bit) or Windows 7 (32- or 64-bit).Mac OS X 10.5.8 or later (x86 only).Linux.
Development Environment	<ul style="list-style-type: none">JDK 7 or higher - Java Runtime Environment (JRE) alone is not sufficient.
Recommended Hardware Requirements	<ul style="list-style-type: none">Minimum 2 Gigabyte (GB) of Random Access Memory (RAM).Intel Core2Due or equal processor for x86 architecture.1 GB hard disk space.

Using slide 39, list the requirements for Eclipse IDE. Ensure that the target systems meet the requirements.

Installing Eclipse IDE

Slides 40 to 42

Installing Eclipse IDE 1-3

- Navigate to the location of the Eclipse setup is downloaded
- Right-click the file and Click 'Extract All' as shown in the following figure:

© Aptech Ltd. Introduction to Android/Session 1 40

Installing Eclipse IDE 2-3

- Provide the path for extracting the files as shown in the following figure:

© Aptech Ltd. Introduction to Android/Session 1 41

Installing Eclipse IDE 3-3

- Click Extract
- The contents are extracted in the mentioned folder as shown in the following figure:

A screenshot of a Windows File Explorer window titled 'eclipse'. The window shows a single item in the list view: 'eclipse' (File folder) located at 'Computer > Windows8_OS (C) > Program Files > eclipse'. The file was modified on 5/8/2015 at 10:58 PM. The status bar at the bottom of the window displays '© Aptech Ltd.' and 'Introduction to Android/Session 1'.

Using slides 40 to 42 demonstrate the procedure for installing Eclipse IDE. Begin by giving a brief history on the previous versions of Eclipse and the features of the current version, such as Java 8 support. Mention that the version used in the given example is Eclipse Luna. The steps in most slides are self-explanatory. Read out the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Additional Reference:

You can refer to the following link(s) for more information regarding Eclipse and its history:

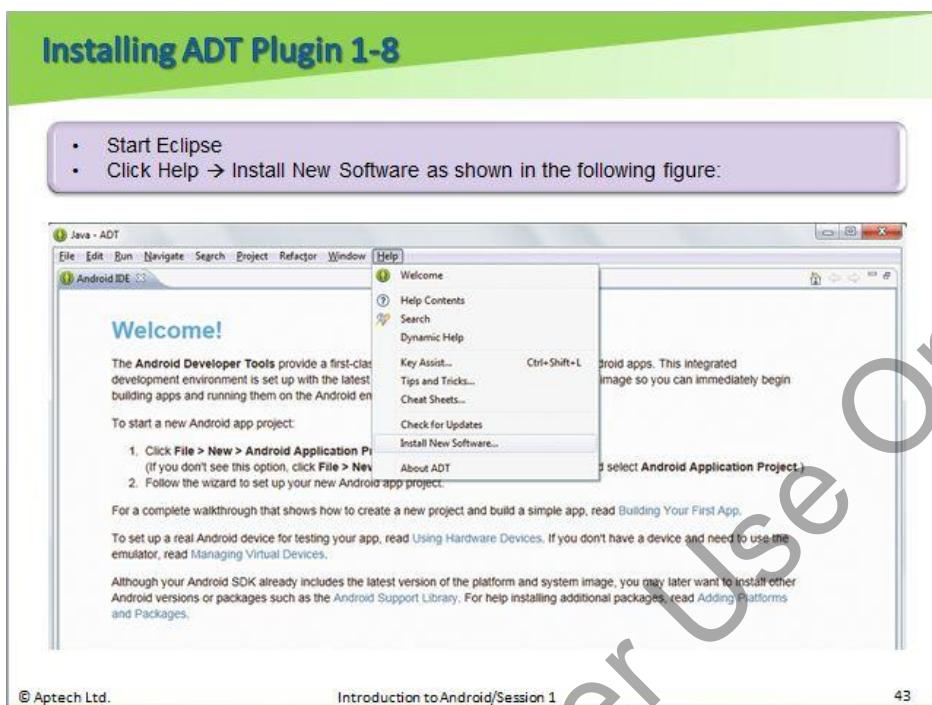
<https://eclipse.org/luna/>

[https://en.wikipedia.org/wiki/Eclipse_\(software\)#Releases](https://en.wikipedia.org/wiki/Eclipse_(software)#Releases)

[https://wiki.eclipse.org/Older Versions Of Eclipse](https://wiki.eclipse.org/Older_Versions_of_Eclipse)

Installing ADT Plugin

Slides 43 to 50



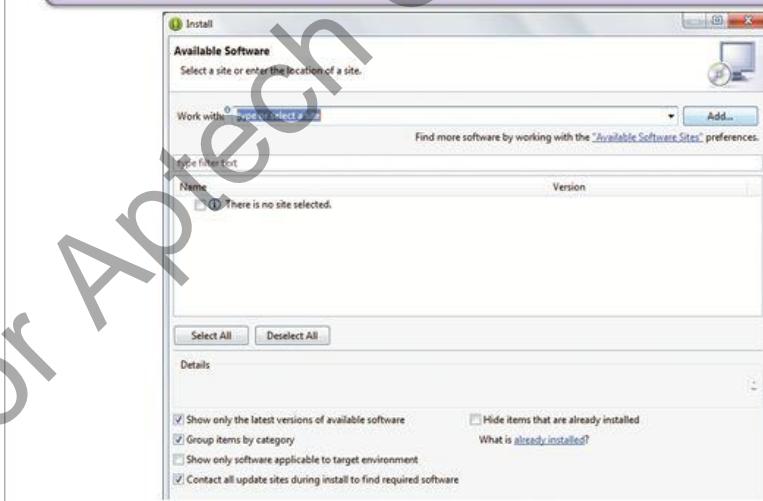
© Aptech Ltd.

Introduction to Android/Session 1

43

Installing ADT Plugin 2-8

- In the Available Software pane of the Install dialog box, click Add, next to the Work with list, as shown in the following figure:



© Aptech Ltd.

Introduction to Android/Session 1

44

Installing ADT Plugin 3-8

- In the Add Repository dialog box, in the Name box, type ADT Plugin as shown in the following figure:



- In the Location box, type the URL <https://dl-ssl.google.com/android/eclipse/> and click OK as shown in the following figure:



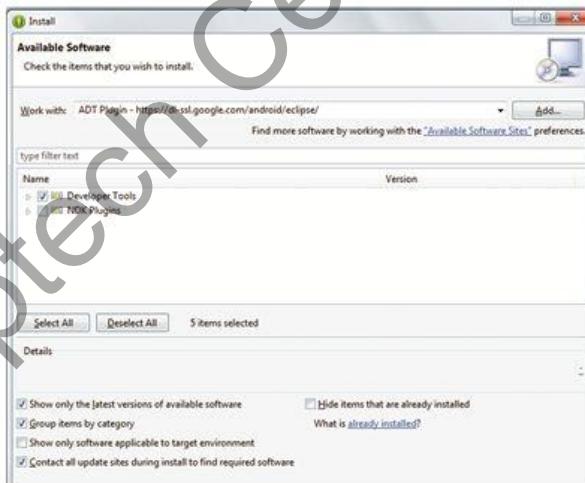
© Aptech Ltd.

Introduction to Android/Session 1

45

Installing ADT Plugin 4-8

- In the Available Software pane of the Install dialog box, select Developer Tools as shown in the following figure:



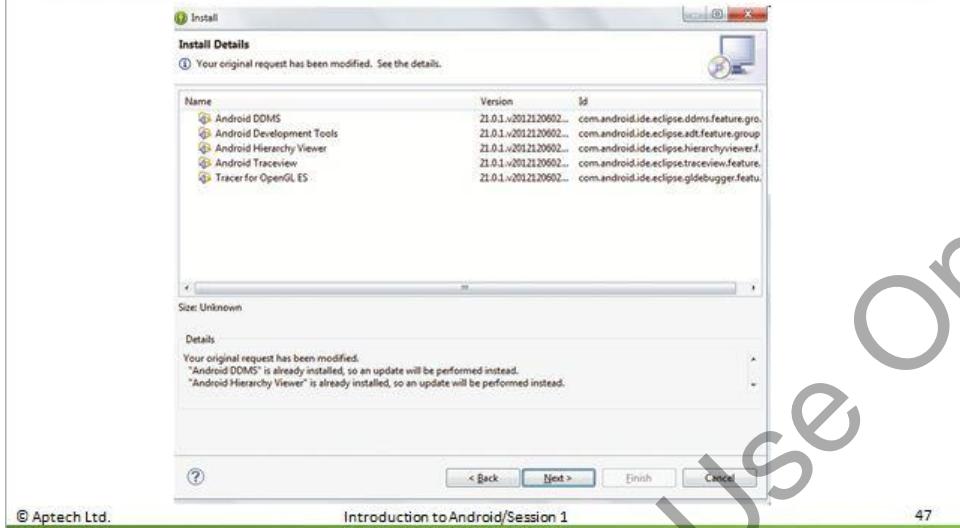
© Aptech Ltd.

Introduction to Android/Session 1

46

Installing ADT Plugin 5-8

- In the Install Details pane of the Install dialog box, click Next as shown in the following figure:



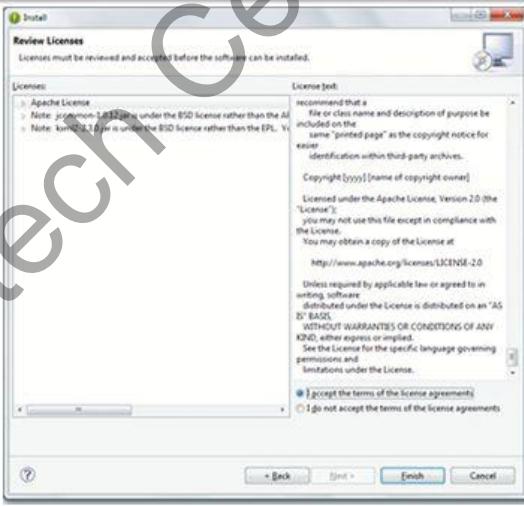
© Aptech Ltd.

Introduction to Android/Session 1

47

Installing ADT Plugin 6-8

- Read and accept the license agreement and click Finish as shown in the following figure:



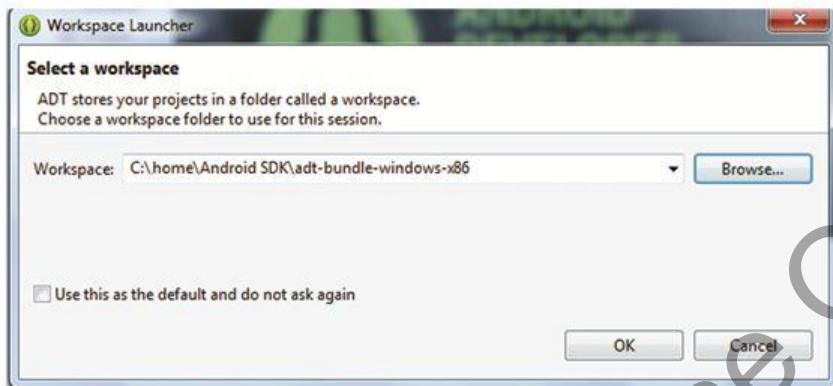
© Aptech Ltd.

Introduction to Android/Session 1

48

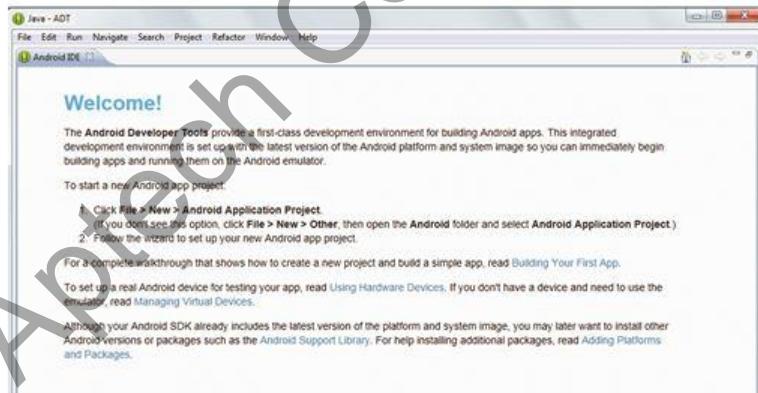
Installing ADT Plugin 7-8

- In the Select a workspace pane in the Workspace Launcher dialog box browse and select the Android SDK directory
- Click OK as shown in the following figure:



Installing ADT Plugin 8-8

- The installation is complete as shown in the following figure:



Using slides 43 to 50, demonstrate the procedure for installing the ADT Plugin. You may begin by explaining that ADT plugin adds Android development specific features to Eclipse. This includes the Android GUI editor replacing the desktop application GUI, Android project creation, and apk packaging. The steps in most slides are self-explanatory. Read the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Summary**Slide 51****Summary**

- ◆ An Operating System (OS) is a software program that enables communication and utilization of the hardware resources by the software programs
- ◆ Android is an Operating System (OS) for mobile devices running on the Linux Kernel
- ◆ Android Lollipop aims to provide major improvements in terms of performance and 64 bit CPU support along with a brand new UI scheme
- ◆ The Android architecture consists four layers namely Kernel, Libraries and Runtime, Application Framework and Applications
- ◆ Google Play (originally called Android Market) is Google's official market place for Android
- ◆ The Linux Kernel is the base layer. This is the OS layer upon which the entire framework is built
- ◆ The Android Studio Bundle comes with the official IDE for developing Android application recommended by Google, the Android Studio

Using slide 51, summarize the session. Make them revise the following points:

- An Operating System (OS) is a software program that enables communication and utilization of the hardware resources by the software programs.
- Android is an Operating System (OS) for mobile devices running on the Linux Kernel.
- Android Lollipop aims to provide major improvements in terms of performance and 64 bit CPU support along with a brand new UI scheme.
- The Android architecture consists four layers namely, Kernel, Libraries and Runtime, Application Framework, and Applications.
- Google Play (originally called Android Market) is Google's official market place for Android.
- The Linux Kernel is the base layer. This is the OS layer upon which the entire framework is built.
- The Android Studio Bundle comes with the official IDE for developing Android application recommended by Google, the Android Studio.

1.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes Application Development Basics. Familiarize yourself with the IDEs (Eclipse and Android Studio) and the process of creating projects. Learn the pre-existing code and how it is important.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 2: Getting Started with Android**2.1 Pre-Class Activities**

You should familiarize yourself with Android Application basics. You should also have a good understanding of the fundamentals of the Android Application. You should study the Android Application components and the Application framework. You should be able to list and explain the contents of an Android Project.

Familiarize yourself with the topics of the current session in-depth.

2.1.1 Objectives

After the session, learners will be able to:

- Explain the process of creating an Android application
- Explain the fundamentals of Android application
- Explain the composition of Android applications framework
- Explain communication components
- Explain pre-existing components

2.1.2 Teaching Skills

You should be familiar with the IDEs used for Android Application Development that is, Eclipse and Android Studio. You should be able to create projects in both the IDEs and explain the auto-generated code. The trainer must have a clear understanding of the various components of a project and their importance.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

2.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Tell the students that the previous session explained the basics of Android and Android history. It also described the procedure to set up the development Environment.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the process of creating Android projects and understanding the code used to develop Android Applications.

2.2 In-Class Explanations

Introduction

Slide 3

The slide has a green header bar with the title 'Introduction'. Below the header, there is a bulleted list of requirements for developing Android applications:

- ◆ A developer needs to understand the basics of Android Applications
- ◆ A supported IDE needs to be used –
 - ◆ Android Studio
 - ◆ Eclipse IDE
 - ◆ IntelliJ
- ◆ Applications can be deployed on AVDs or real devices

On the left side of the slide, there is a small icon of a computer monitor displaying the word 'android'. On the right side, there is an icon of a smartphone displaying a home screen. Between these icons is a screenshot of an AndroidManifest.xml file. The file contains the following XML code:

```
<manifest>
    <uses-sdk android:minSdkVersion="3" android:targetSdkVersion="4" />
    <application android:allowBackup="true" android:icon="@drawable/icon" android:label="@string/app_name" android:theme="@style/AppTheme" />
    <activity android:name=".MainActivity" android:label="@string/app_name" android:theme="@style/AppTheme" />
</manifest>
```

At the bottom of the slide, there is a footer bar with the text '© Aptech Ltd.' on the left, 'Getting Started with Android/Session 2' in the center, and the number '3' on the right.

Using slide 3, explain an outline of what is going to be covered in the session. List and explain the various IDEs available for Android Application Development. Explain the importance of an AVD and state the advantages and disadvantages when compared to deploying an application on a real device. AVDs lets you check compatibility on multiple device configurations without having to invest in the real devices.

Creating an Android Application

Slide 4

Creating an Android Application

- ◆ What is an Android Project
 - ◆ An Android project consists of the entire source code for an Android application
 - ◆ It gets compiled and packaged into an apk
- ◆ An Android project can be created in two ways:
 - ◆ Eclipse IDE
 - ◆ Android Studio



The slide features logos for two popular Android IDEs: Eclipse (with its blue and orange logo) and Android Studio (with its green and white logo).

© Aptech Ltd. Getting Started with Android/Session 2 4

Using slide 4, explain about Android project and its contents. Give a brief outline of the contents of an Android project and the importance of each of them. Explain the process of compilation and packaging into an APK.

Explain about the IDEs that can be used for Android App development and give a brief idea about their history. Mention that Android Studio is Google's official IDE for Android Application development.



In-Class Question: What is an IDE?

Answer: IDE is the abbreviation for Integrated Development Environment. It is a software tool which eases the process of developing Applications.

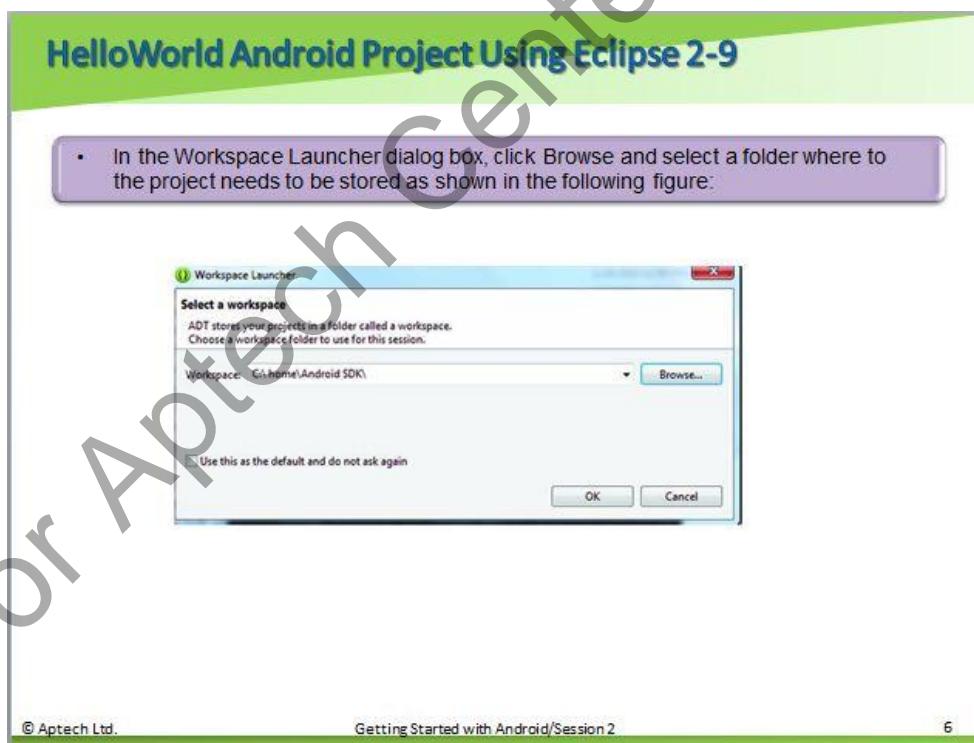
Additional Reference:

You can refer the following link(s) for more information regarding SDK and IDE:

<https://developer.android.com/sdk/index.html>

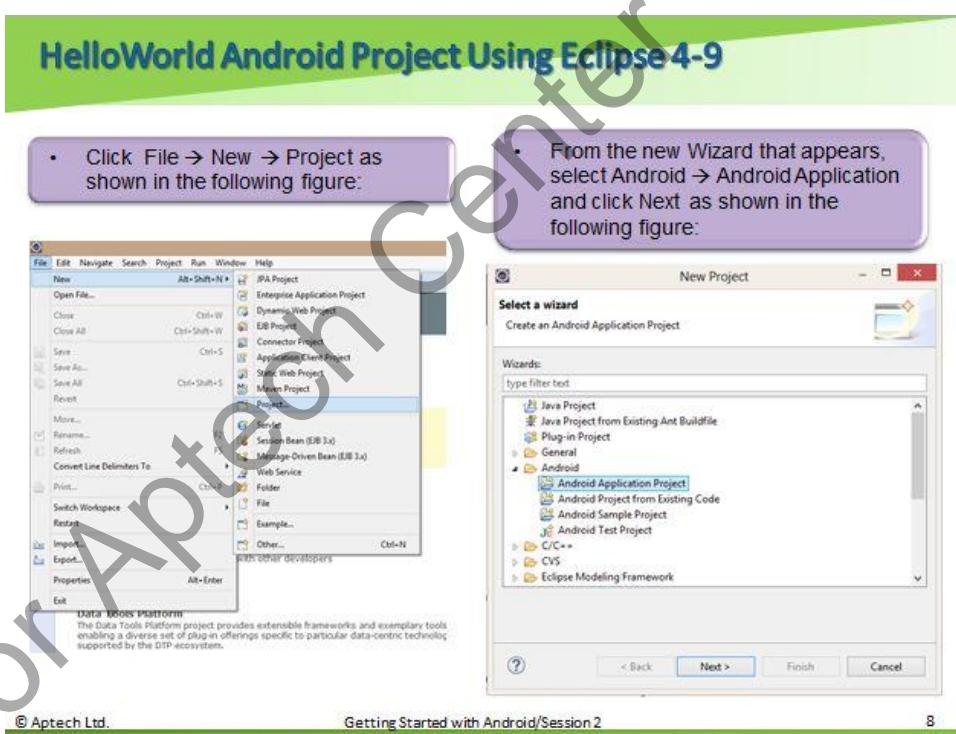
HelloWorld Android Project Using Eclipse

Slides 5 to 13





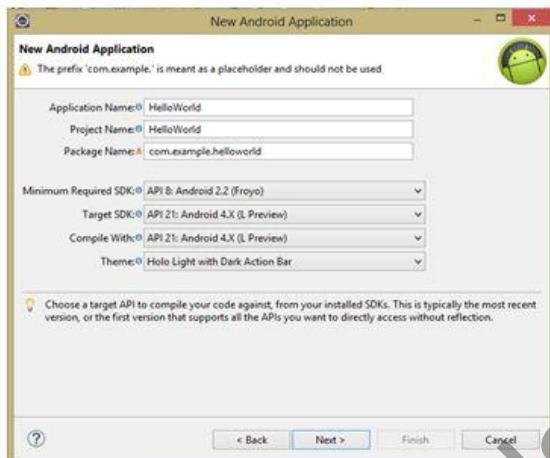
7



8

HelloWorld Android Project Using Eclipse 5-9

- In the Application Name box, type HelloWorld as shown in the following figure:



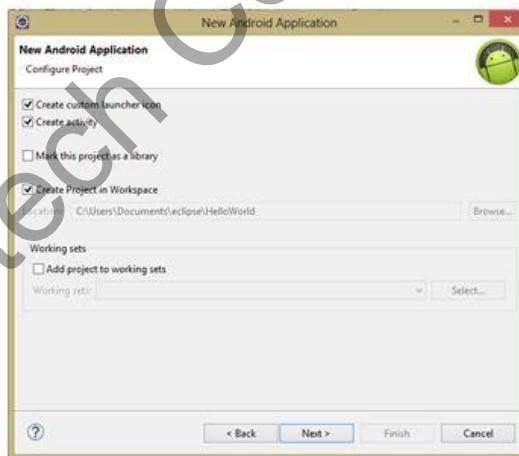
© Aptech Ltd.

Getting Started with Android/Session 2

9

HelloWorld Android Project Using Eclipse 6-9

- Click Next twice in the Configure Project pane, as shown in the following figure:



© Aptech Ltd.

Getting Started with Android/Session 2

10

HelloWorld Android Project Using Eclipse 7-9

- In the Configure Launcher Icon screen that is displayed, customize the appearance of the launcher icon as shown in the following figure:



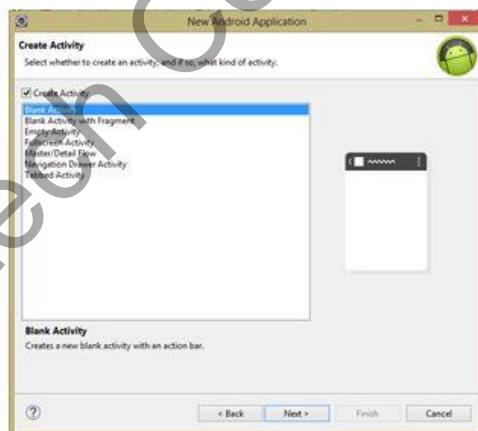
© Aptech Ltd.

Getting Started with Android/Session 2

11

HelloWorld Android Project Using Eclipse 8-9

- Click Next. The Create Activity screen is displayed as shown in the following figure:



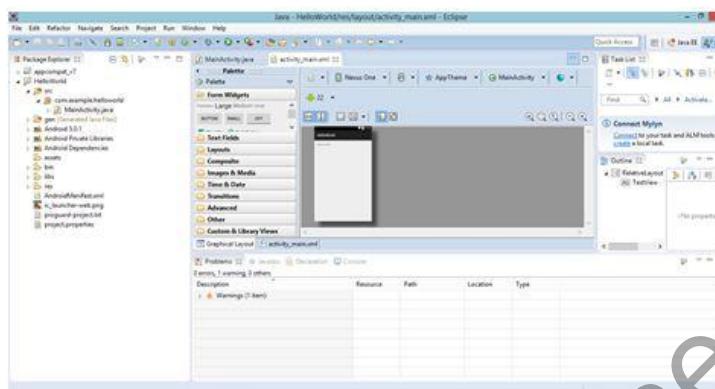
© Aptech Ltd.

Getting Started with Android/Session 2

12

HelloWorld Android Project Using Eclipse 9-9

- Select Blank Activity and click Next.
- Click Finish. The project is created as shown in the following figure:



© Aptech Ltd.

Getting Started with Android/Session 2

13

Using slides 5 to 13, demonstrate the procedure for creating a project in Eclipse. Start by specifying that we begin with the process to install Eclipse IDE. Using slide 6, explain the process to setup the workplace. Explain that workplace setup is usually a one-time process for setting up the IDE.

Using slide 7, explain the welcome screen of the Eclipse IDE. Explain that the welcome screen contents may change over time.

Explain that the type of activity selected will affect the auto generated code. This can be changed later on by manually modifying the code. The steps in most slides are self-explanatory. Read out the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.



In-Class Question: What is an AVD?

Answer: AVD stands for Android Virtual Device which is an Android emulator. It functions similar to a Real Device.

Android Virtual Device

Slide 14

Android Virtual Device

- ◆ An Android Virtual Device is an Android Emulator
- ◆ The emulator functions exactly similar to a mobile device
- ◆ AVD consists of:
 - ◆ Hardware features
 - ◆ Software features
 - ◆ The look of the screen



© Aptech Ltd. Getting Started with Android/Session 2 14

Using slide 14, explain about Android Virtual Device. Explain the term Emulator and the importance of AVD in Android application development. An emulator is hardware or software that enables one computer system (called the host) to behave like another computer system (called the guest). Explain the contents of an AVD and how each of them is essential to the AVD.

Additional Reference:

You can refer the following link(s) for more information regarding the AVD Manager and AVDs:

<http://developer.android.com/tools/help/avd-manager.html>

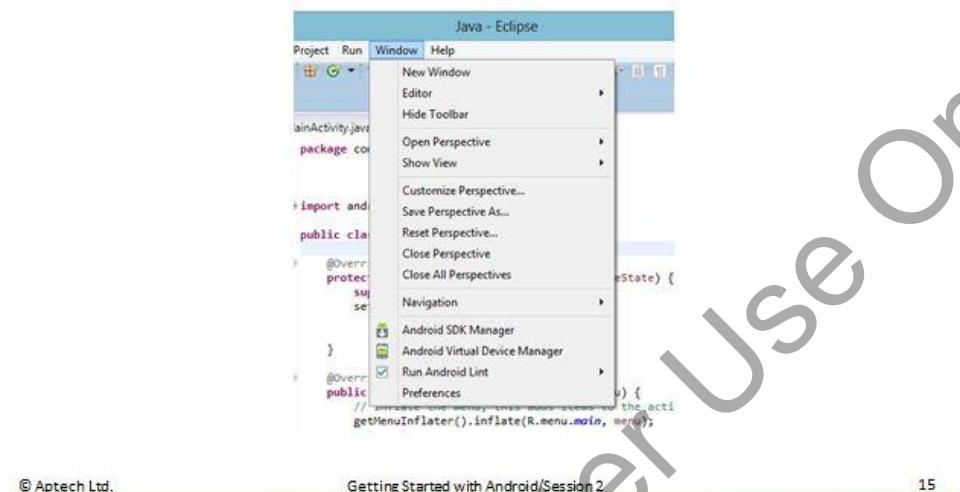
<https://developer.android.com/tools/devices/index.html>

Creating an AVD

Slides 15 to 20

Creating an AVD 1-6

- Click Window → Android Virtual Device Manager to start the AVD Manager as shown in the following figure:



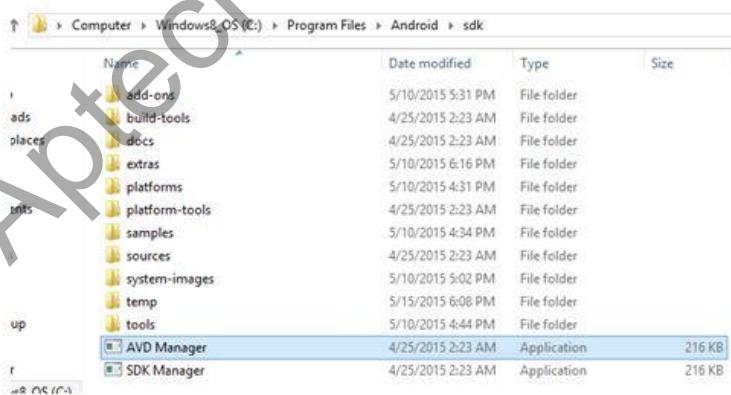
© Aptech Ltd.

Getting Started with Android/Session 2

15

Creating an AVD 2-6

- Alternately, navigate to the Android SDK installation directory and start the AVD Manager as shown in the following figure:



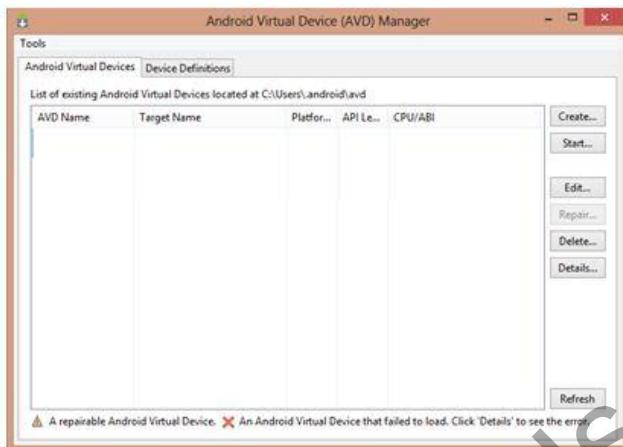
© Aptech Ltd.

Getting Started with Android/Session 2

16

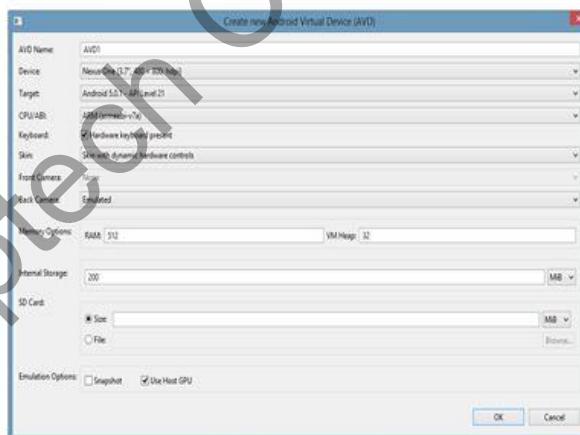
Creating an AVD 3-6

- The Android Virtual Device (AVD) Manager dialog box is displayed as shown in the following figure:



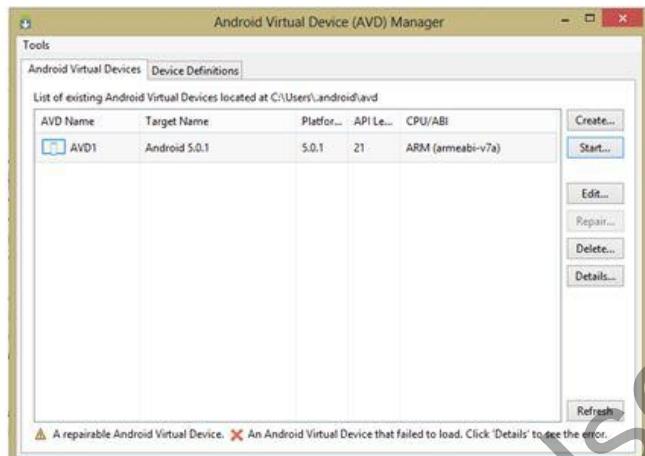
Creating an AVD 4-6

- Click Create to create a new AVD. The Create new Android Virtual Device (AVD) dialog box is displayed.
- Enter the AVD details as shown in the following figure:



Creating an AVD 5-6

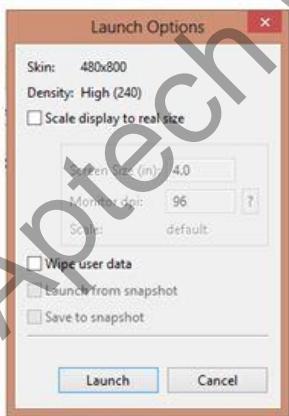
- Click OK after all the requirements are complete.
- The Android Virtual Device Manager dialog box appears displaying the newly created AVD, as shown in the following figure:



Creating an AVD 6-6

- Select AVD1 and click Start to start the emulator.
- Click Launch in the Launch Options dialog box as shown in the following figure:

The emulator is launched as shown in the following figure:



Using slides 15 to 20, demonstrate the process of creating an AVD. The steps in most slides are self-explanatory. Using slide 15, explain the process to start the AVD manager from within the IDE itself. This can be done from the Window menu. Using slide 18, explain the process to create a new AVD. The options chosen here will be used for creating the AVD used for testing future applications.

Read the steps and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Creating Launch Configurations

Slide 21

Creating Launch Configurations

- ◆ A launch configuration is required to run the app on the AVD
- ◆ There are two type of launch configurations –
 - ◆ Run Configurations
 - ◆ Debug Configurations



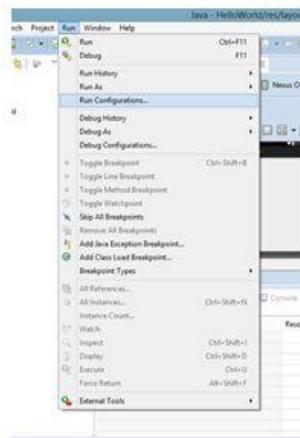
Using slide 21, explain about Launch Configuration and its requirement. Explain the two types of Launch configurations and how they differ from each other. A Launch Configuration specifies the information to deploy the application such as the project, the main activity, target device, and so on. Run configuration contains information to deploy the application in a real time environment whereas debug configuration is used to deploy the application in a debugging environment and testing.

Creating Run Configurations

Slides 22 to 25

Creating Run Configurations 1-4

- In Eclipse, click Run → Run Configurations as shown in the following figure:



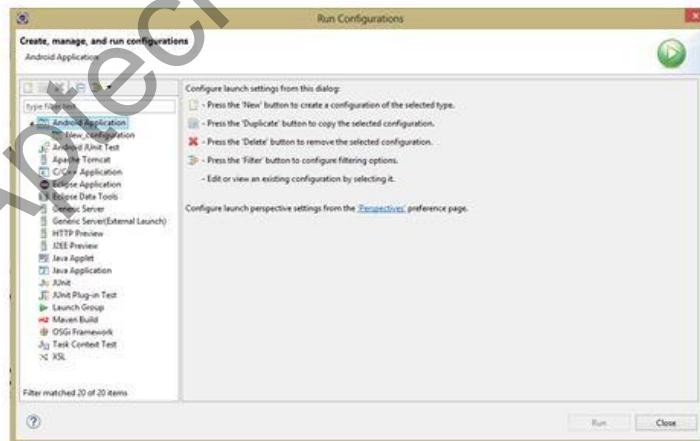
© Aptech Ltd.

Getting Started with Android/Session 2

22

Creating Run Configurations 2-4

- In the Create, manage, and run configurations dialog box, select Android Application and click New launch configuration icon as shown in the following figure:



© Aptech Ltd.

Getting Started with Android/Session 2

23

Creating Run Configurations 3-4

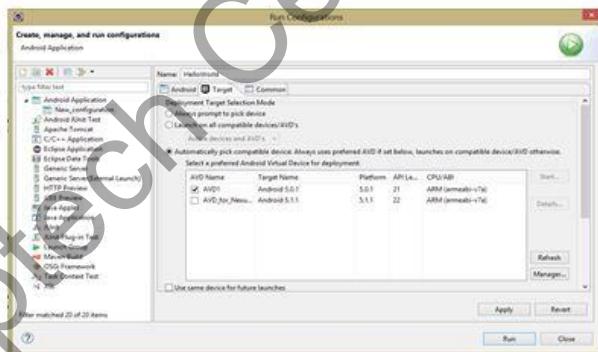
- In the Name box type the name of the new configuration as Hello World
- Click Browse to display the Project Selection dialog box as shown in the following figure:



- Select HelloWorld and click OK

Creating Run Configurations 4-4

- Click the Target tab and select the newly created AVD, as shown in the following figure:



- Click Apply
- Click Close to exit the Create, manage, and run configurations dialog box

Using slides 22 to 25, demonstrate the process of creating Run Configuration. This can be done from the Run menu. Using slide 24, explain the process of selecting the project for creating the Run Configuration. Run Configurations are unique to each application. The steps in most slides are self-explanatory. Read the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.



In-Class Question: What is debugging?

Answer: Debugging is the process of analyzing application execution and determining and fixing bugs.

Additional Reference:

You can refer the following link(s) for more information regarding debugging and the adb Android debugger:

<https://en.wikipedia.org/wiki/Debugging>

<http://developer.android.com/tools/help/adb.html>

Creating Debug Configurations

Slides 26 and 27

Creating Debug Configurations 1-2

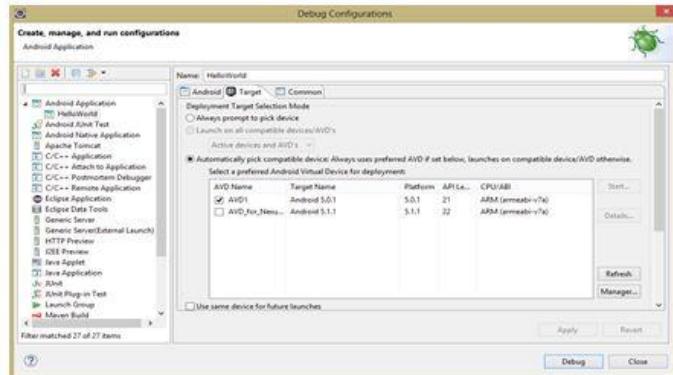
- In Eclipse, click Run Debug → Configurations
- In the Debug Configurations dialog box, in the Android Application list, select HelloWorld as shown in the following figure:

The screenshot shows the 'Debug Configurations' dialog box in Eclipse. On the left, there's a tree view with 'Android Application' selected, showing 'HelloWorld' under it. The main right panel has fields for 'Name' (set to 'HelloWorld'), 'Project' (set to 'HelloWorld'), and 'Launch Action' (radio button selected for 'Launch Default Activity'). At the bottom, there are 'Apply' and 'Debug' buttons.

© Aptech Ltd. Getting Started with Android/Session 2 26

Creating Debug Configurations 2-2

- Click the Target tab. The AVD for the configuration is selected by default as shown in the following figure:



- Click Debug
- Click Run → Run

© Aptech Ltd.

Getting Started with Android/Session 2

27

Using slides 26 and 27, demonstrate the process of creating Debug Configuration explained in slide 21. The steps in most slides are self-explanatory. Read the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Running and Debugging Hello World Project

Slide 28

Running and Debugging Hello World Project

- The apk is installed on the target
- Running and Debugging can be done on –
 - Real Android Device
 - Emulator
- Its deployed automatically using adb, by the IDE



© Aptech Ltd.

Getting Started with Android/Session 2

28

Using slide 28, explain the difference between running and debugging a project. Explain the importance of debugging. List the methods of running and debugging and explain the merits and demerits of each. State the fact that the application is automatically deployed using adb

by the IDE.

Additional Reference:

You can refer the following link(s) for more information regarding debugging:

<https://en.wikipedia.org/wiki/Debugging>

Running the Project

Slides 29 and 30

Running the Project 1-2

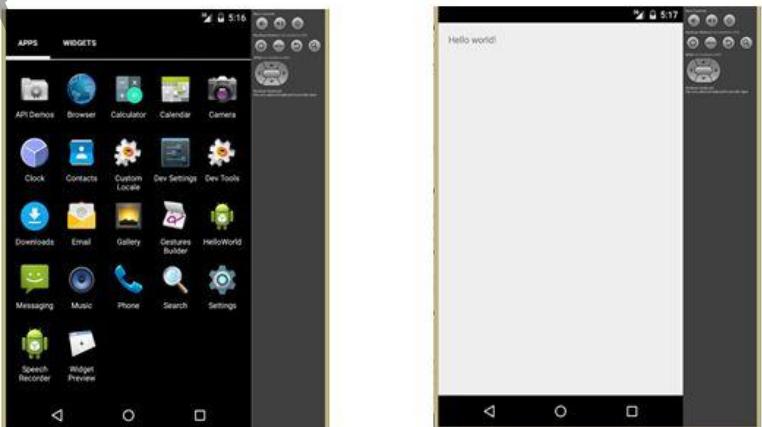
- In Eclipse, click Run → Run
- The emulator is displayed with the lock screen as shown in the following figure:
- Click and drag the lock icon upwards to unlock the home screen
- On the home screen, click the circle and click OK as shown in the following figure:



© Aptech Ltd. Getting Started with Android/Session 2 29

Running the Project 2-2

- When the apps load, click the HelloWorld app icon as shown in the following figure:
- The text Hello world! is displayed on the screen as shown in the following figure:



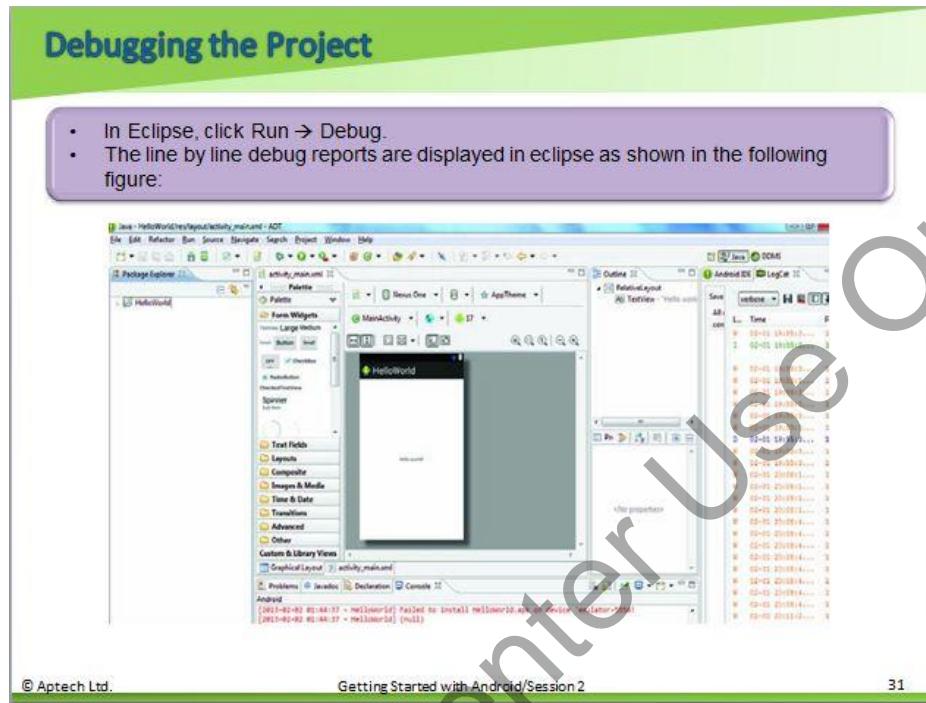
© Aptech Ltd. Getting Started with Android/Session 2 30

Using slides 29 and 30, demonstrate the process of running the project. You can start by

explaining that once the basic code is laid down for the application, the developer can run and test it. The steps in most slides are self-explanatory. Read the steps and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Debugging the Project

Slide 31



Using slide 31, demonstrate the process of debugging an Android application. You can start by explaining that bugs are a very common phenomenon in software projects and it is necessary to find the source of the said bug and fix it. Using the figure given on the slide, explain how the interface can be used to debug the application and the various windows and their purpose.

Understanding the HelloWorld Project

Slide 32

The screenshot shows the Eclipse IDE's Project Explorer view. The project name is 'HelloWorld'. Inside the project, there are several folders and files:

- src**: Contains the Java source file `MainActivity.java`.
- gen**: Contains auto-generated files, including `[Generated Java Files]`.
- assets**: Contains the image file `ic_launcher-web.png`.
- bin**: Contains compiled class files.
- res**: Contains XML files for layouts, drawables, and values, including subfolders for different screen densities (hdpi, ldpi, mdpi, xhdpi) and API levels (values-v11, values-v14, values-v20dp).
- AndroidManifest.xml**: The manifest file for the application.
- project.properties**: Properties file for the project.
- proguard-project.txt**: ProGuard configuration file.
- ic_launcher-web.png**: Application icon.
- activity_main.xml**: Layout XML file.
- .java files**: Other Java source files.

At the bottom of the slide, there is a watermark reading "For Aptech Center Only".

Using slide 32, explain the various components of a project. List each component while providing a brief explanation of their purpose and their contents. For example, you may explain that the 'src' folder contains the Java source files for the project and the 'gen' directory consists of the auto-generated content by the compiler such as the .R file.

Additional Reference:

You can refer the following link(s) for more information regarding the code components:

http://www.tutorialspoint.com/android/android_hello_world_example.htm

Android Application Fundamentals

Slide 33

Android Application Fundamentals

- ◆ Each Application consists of several components
- ◆ An application may include Activities, Services, Content Providers, and so on

◆ **Security Sandboxes in Android**

- ◆ Every application has a different user ID
- ◆ Permission is granted to the application files so that only valid users can access them
- ◆ The presence of a Virtual Machine (VM) isolates the application

◆ **Concept of Least Privilege**

- ◆ Two applications can share Linux User ID and VM and access each other's files
- ◆ Permissions granted at time of installation



© Aptech Ltd.

Getting Started with Android | Session 2

33

Using slide 33, explain the various components of an Android application. Provide a brief explanation for each component. Explain the concept of Security Sandboxes in Android. Explain the role of VM and user IDs and how they are essential to application security.

Explain the concept of 'Least Privilege' and how file sharing is achieved. Mention that permissions are granted at the time of installation and how this feature enhances application security.

Application Components

Slide 34

Application Components

- ◆ Activities
- ◆ Services
- ◆ Content Providers
- ◆ Broadcast Receivers
- ◆ Resources
- ◆ Assets
- ◆ Layouts



© Aptech Ltd. Getting Started with Android/Sesson 2 34

Using slide 34, explain various components of an Android application. List the components given in the slide and provide the definition for them. For example, explain that an activity is a screen containing the User Interface (UI) and allows interacting of the user with the application.

A service is a background component that processes lengthy operations or is connected to processes that run remotely.

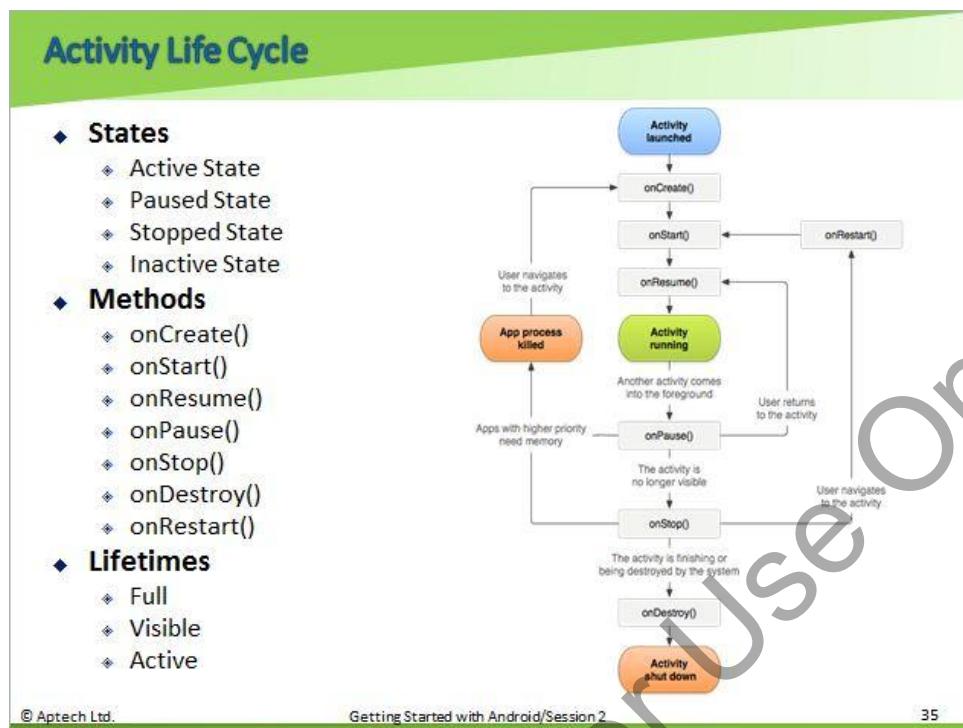
Additional Reference:

You may refer the following link for more information regarding the components:

http://www.tutorialspoint.com/android/android_application_components.htm

Activity Life Cycle

Slide 35



Using slide 35, explain the concepts of Activity life cycle. List and explain the various states of an activity and how they differ from each other. Explain the methods and how they can be used to implement state specific functionality. Explain the importance of `onCreate()` and how it can be used to initialize the activity. The `onCreate()` method is invoked immediately once the activity is started and therefore, it can be used as a 'Constructor' for the activity to initialize it.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Android activities.

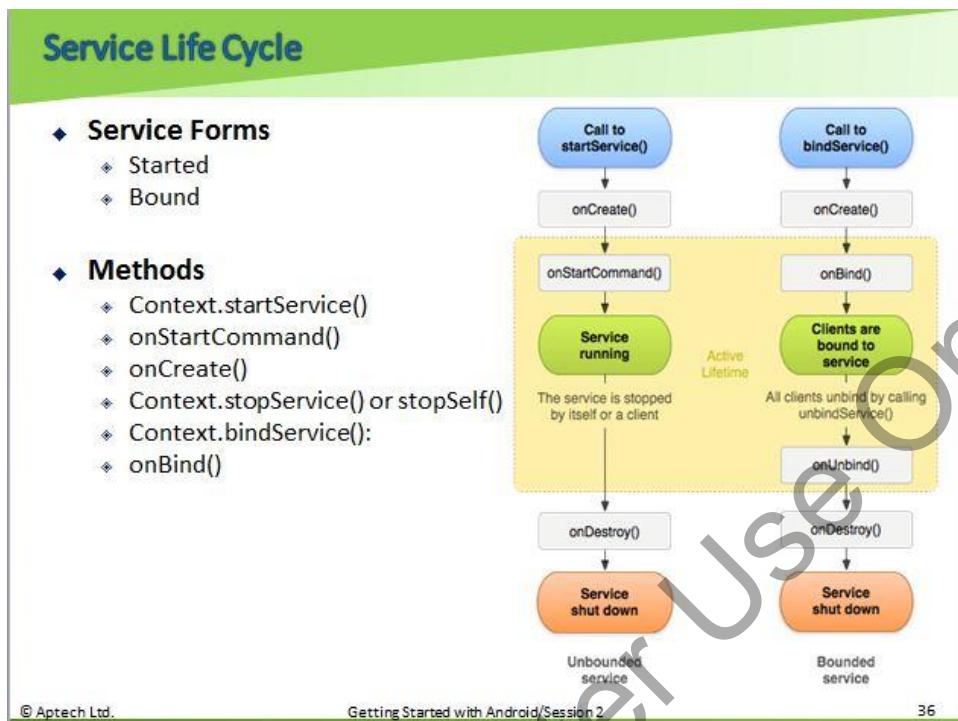
(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/reference/android/app/Activity.html>

Service Life Cycle

Slide 36



Using slide 36, explain the service life cycle. List and explain the various forms of a service and how they differ from each other. Define 'Started Service' and a 'Bound Service'. You need not emphasize on this any further at this stage. Proceed to explain that the diagram given on the slide represents the various stages of a service lifecycle. Each stage has a method that is invoked when the stage is reached. Explain these methods. Detailed explanation on each of these methods can be found in the following reference.

Additional Reference:

You may refer the Android SDK documentation available with the SDK download for more information regarding Android services.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/guide/components/services.html>



In-Class Question: List the various Android application components?

Answer:

- Activities
- Services
- Content Providers
- Broadcast Receivers
- Resources
- Assets
- Layouts

Content Providers

Slide 37

Content Providers

- ◆ **Content Providers**
 - ❖ Provide access to data
 - ❖ Allow CRUD operations
- ◆ **Methods**
 - ❖ onCreate ()
 - ❖ query (Uri, String [], String, String [], String)
 - ❖ insert (Uri, ContentValues)
 - ❖ update (Uri, ContentValues, String, String [])
 - ❖ delete (Uri, ContentValues, String, String [])
 - ❖ getType (Uri)



© Aptech Ltd. Getting Started with Android/Session 2 37

Using slide 37, explain Content Providers and their importance. Explain the term 'CRUD' and how each of the operation differs from each other. CRUD is the abbreviation for 'Create Read/Retrieve Update Delete' operations. List the methods used for accessing Content Provider API and their purpose.

Additional Reference:

You may refer the Android SDK documentation available with the SDK download for more information regarding Content Providers.

(OR)

You can refer to the online Mirror of the same documentation using following link:

<http://developer.android.com/reference/android/content/ContentProvider.html>

Broadcast Receivers

Slide 38

Broadcast Receivers

- ◆ **Broadcast Receiver**
 - ◆ Receives and responds to announcements
 - ◆ Informs applications of events
- ◆ **Methods**
 - ◆ `abortBroadcast()`
 - ◆ `clearAbortBroadcast()`
 - ◆ `getAbortBroadcast()`
 - ◆ `getDebugUnregister()`
 - ◆ `getResultCode()`
 - ◆ `getResultData()`



© Aptech Ltd. Getting Started with Android/Session 2 38

Using slide 38, explain Broadcast Receivers and their importance. Provide a small example of how they can be used in application development such as a service using them to invoke the User Interface components/display a notification when an event occurs. Explain the various methods of broadcast receivers and their functionality.

Additional Reference:

You may refer the Android SDK documentation available with the SDK download for more information regarding Broadcast Receivers and their importance.

(OR)

You can refer to the online Mirror of the same documentation using following link:

<http://developer.android.com/reference/android/content/BroadcastReceiver.html>

Resources and Assets

Slide 39

Resources and Assets

- ◆ **Resources**
 - ◆ Static Content
 - ◆ Structured Data
 - ◆ Several types of resources
 - ◆ Animation, color, drawable, and so on
- ◆ **Assets**
 - ◆ Functionally similar to resources
 - ◆ Non structured data
 - ◆ Behaves similar to a file system



© Aptech Ltd. Getting Started with Android / Session 2 39

Using slide 39, explain how static content differs from dynamic content. Static content does not change based on user input. Examples of static content include images and labels. Dynamic content varies based on user interaction and events. Examples for this include activates, dialogs, and code generated components. Explain the methods used to store static contents such as resources and assets and how they differ from each other. Resources have a limitation in the type of data they can hold whereas assets can hold raw and unstructured data.

Layouts

Slide 40



The slide contains a list of bullet points under two main sections: 'Layouts' and 'Types'. The 'Layouts' section includes points about defining visual structure, being defined in XML, and being editable in code or GUI. The 'Types' section lists various layout types: Linear Layout, Relative Layout, Table Layout, Absolute Layout, Frame Layout, List View, and Grid View.

◆ **Layouts**

- ◆ Define Visual Structure of the UI
- ◆ Defined in XML
- ◆ Can be edited in code or GUI

◆ **Types**

- ◆ Linear Layout
- ◆ Relative Layout
- ◆ Table Layout
- ◆ Absolute Layout
- ◆ Frame Layout
- ◆ List View
- ◆ Grid View

© Aptech Ltd. Getting Started with Android/Sesson 2 40

Using slide 40, explain the basic concepts of layout and how they are essential for the GUI of an application. Using the bulleted points under layouts elaborate the characteristics of a layout file. List the various layout types and provide a brief explanation for each of them.

Additional Reference:

You may refer the Android SDK documentation available with the SDK download for more information regarding Layouts.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/guide/topics/ui/declaring-layout.html>

Application Core Concepts

Slide 41

Application Core Concepts

- ◆ Intent
 - ◆ It Activates Components
- ◆ Intent Filter
 - ◆ Use to select the Intents that can be managed
- ◆ Pre-existing Components
 - ◆ Intent Object
 - ◆ Intent Matching
 - ◆ Intent Resolution
- ◆ Android Manifest File
 - ◆ Contains Configuration Information
 - ◆ Defined in XML
 - ◆ Mandatory for all Applications



© Aptech Ltd.

Getting Started with Android/Session 2

41

Using slide 41, explain the concepts of Intents, Intent Filters, other core-existing components, and the Android manifest file. Using the bulleted points elaborate each topic. As you list each concept, provide the definition to the concept such as Intent is an abstract definition of an operation that is needed to be performed by some other component. Then, provide their uses, which are mentioned as a sub bulleted points under the concepts. Emphasize the importance of the Android manifest file and its functionality.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the Android manifest file.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/guide/topics/manifest/manifest-intro.html>

Recreating HelloWorld in Android Studio

Slides 42 to 47

Recreating HelloWorld in Android Studio 1-6

- Start Android Studio
- Select New Android Project from the Welcome screen as shown in the following figure:

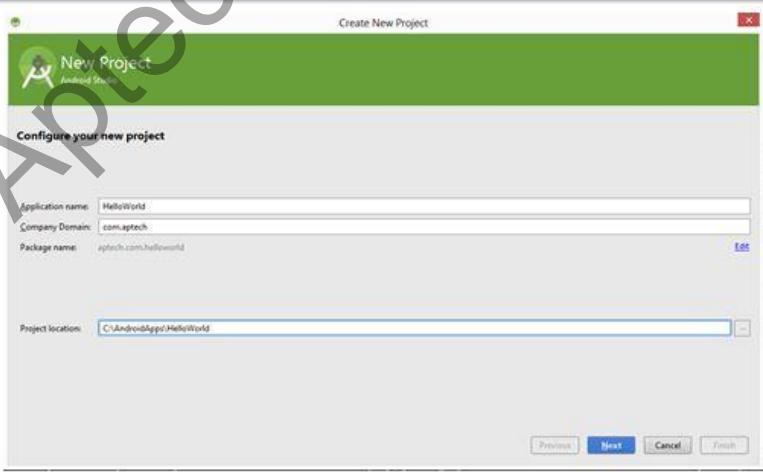


The screenshot shows the 'Welcome to Android Studio' window. On the left, under 'Recent Projects', there is a single entry: 'My Application - <AndroidStudioProjects>/MyApplication'. On the right, under 'Quick Start', there is a list of options: 'Start a new Android Studio project', 'Open an existing Android Studio project', 'Import an Android code sample', 'Check out project from Version Control', 'Import project (Eclipse ADT, Gradle, etc.)', 'Configure', and 'Docs and How-Tos'. A callout box highlights the first option, 'Start a new Android Studio project'. The status bar at the bottom indicates 'Android Studio 1.2 Build 141.1880955. Check for updates now.' and 'Getting Started with Android/Session 2'.

Using slide 42, explain the process of re-creating the same Hello World Project in Android studio. Explain the start screen of the IDE using the figure given on the slide.

Recreating HelloWorld in Android Studio 2-6

- Enter the details and click Next as shown in the following figure:



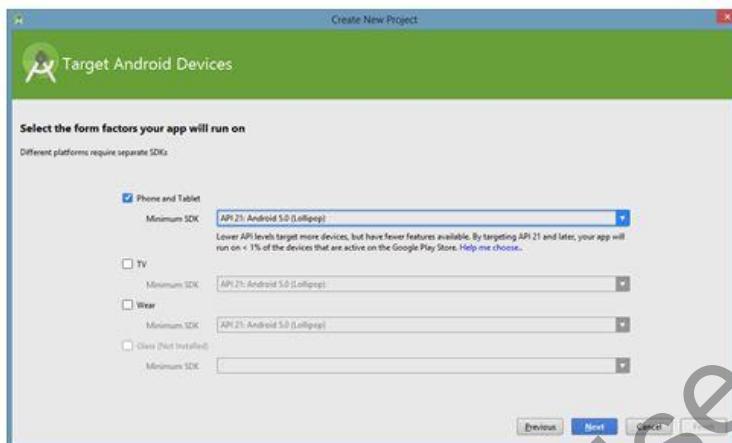
The screenshot shows the 'Create New Project' dialog. It has a green header bar with the title 'Create New Project'. Below it, a section titled 'Configure your new project' contains the following fields:

- Application name:
- Company Domain:
- Package name:
- Project location:

At the bottom of the dialog are four buttons: 'Previous', 'Next', 'Cancel', and 'Finish'. The 'Next' button is highlighted in blue.

Recreating HelloWorld in Android Studio 3-6

- Select API level 21 and click Next as shown in the following figure:



© Aptech Ltd.

Getting Started with Android/Session 2

44

Using slide 44, explain the Target API Version selection screen. Explain that the API version selected here will be stored in multiple configuration files of the project. Hence, it needs to be chosen carefully.

Recreating HelloWorld in Android Studio 4-6

- Select Blank Activity and click Next as shown in the following figure:



© Aptech Ltd.

Getting Started with Android/Session 2

45

Using slide 45, explain the Activity selection screen. Explain that the type of activity chosen will alter the auto-generated code for the activity. It can be changed manually later on.

Recreating HelloWorld in Android Studio 5-6

- Enter the Activity details and click Next as shown in the following figure:

Activity Name:

Layout Name:

Title:

Menu Resource Name:

Blank Activity

The name of the activity class to create

Previous Next Cancel Create

Recreating HelloWorld in Android Studio 6-6

- Click Finish
- The project is created as shown in the following figure:

HelloWorld - [C:\AndroidApp\HelloWorld] - [app] - app\src\main\res\layout\activity_main.xml - Android Studio 1.2

MainActivity.java

activity_main.xml

activity_main.xml

layout

drawable

layout

activity_main.xml

menu

values

Gradle Scripts

Design Test

Properties

layout_height: match_parent

style

accessibilityLiveRegion

accessibilityTraversalOrder

alpha

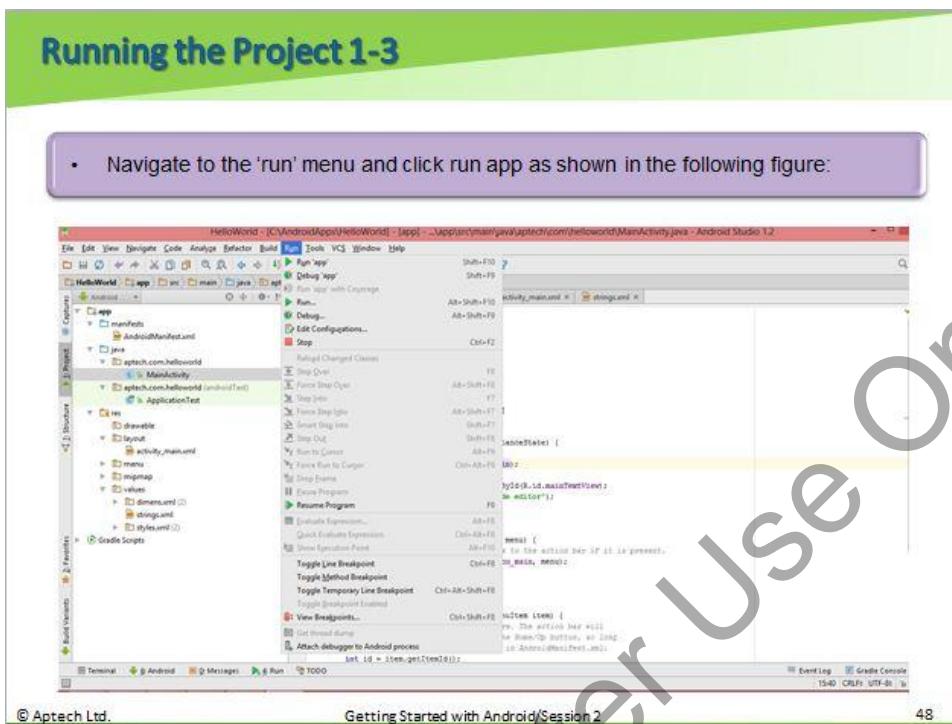
background

backgroundTint

Using slides 42 to 47, demonstrate the process of creating a HelloWorld project in Android studio. You can start by explaining that as the basics are covered, we can now learn the process of creating the first Android project. The steps in most slides are self-explanatory. Read out the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Running the Project

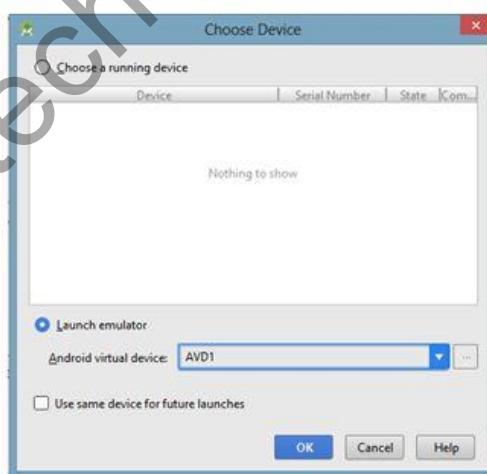
Slides 48 to 50



48

Running the Project 2-3

- From the device selection screen, select AVD1 as shown in the following figure:



49

Running the Project 3-3

- The emulator starts with the application running as shown in the following figure:



© Aptech Ltd.

Getting Started with Android/Session 2

50

Using slides 48 to 50, demonstrate the process of running a project in Android studio. The steps in most slides are self-explanatory. Read out the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Importing Projects

Slides 51 to 55

Importing Projects 1-5

- From the Home Screen select Import Project as shown in the following figure:



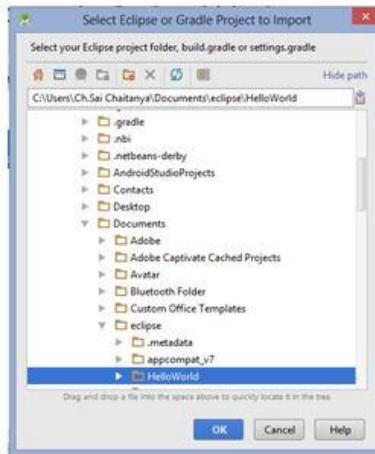
© Aptech Ltd.

Getting Started with Android/Session 2

51

Importing Projects 2-5

- Navigate to the Eclipse project directory and select the folder as shown in the following figure:



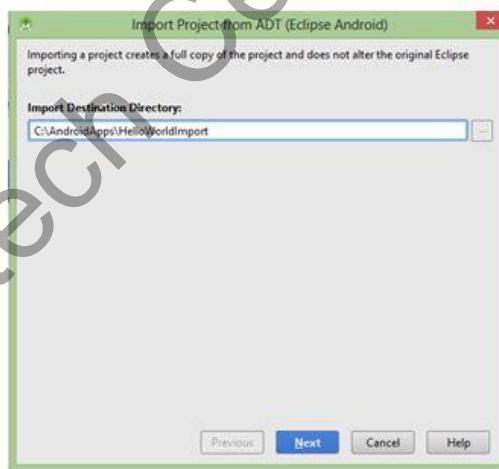
© Aptech Ltd.

Getting Started with Android/Session 2

52

Importing Projects 3-5

- Select the destination directory and click Next as shown in the following figure:



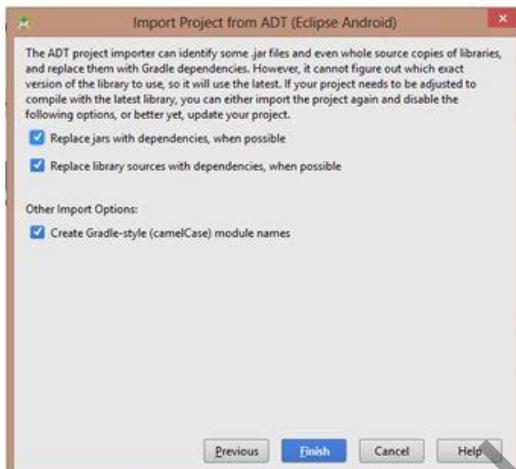
© Aptech Ltd.

Getting Started with Android/Session 2

53

Importing Projects 4-5

- In the confirmation dialog box, click Finish as shown in the following figure:



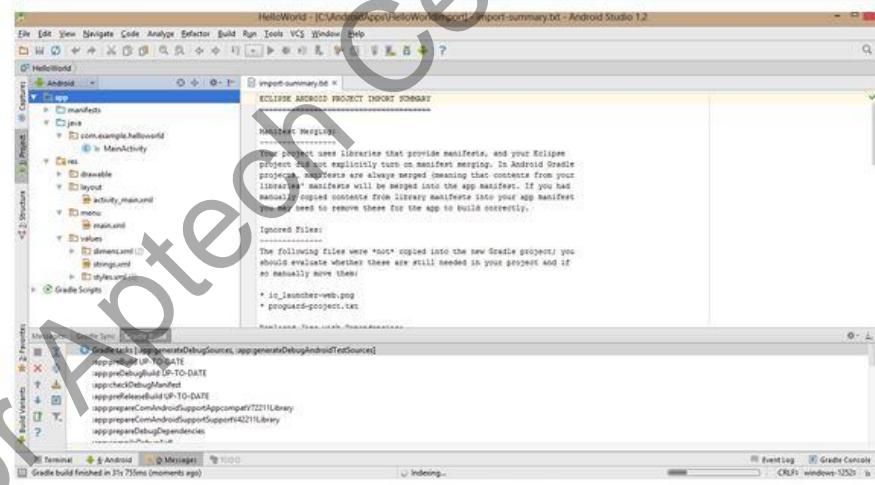
© Aptech Ltd.

Getting Started with Android/Session 2

54

Importing Projects 5-5

- The project is created and displayed as shown in the following figure:



© Aptech Ltd.

Getting Started with Android/Session 2

55

Using slides 51 to 55, demonstrate the process of importing a project from Eclipse into Android Studio. Begin by explaining that Android Studio is the official IDE for Android development and that it provides the means to import Eclipse projects to allow the developers working with Eclipse to transit to Android studio. The steps in most slides are self-explanatory. Using slide 52, explain the process of selecting the Eclipse project directory. Using slide 53, explain the destination directory selection for the project. Mention that this directory needs to be different from the import directory. Explain that the top level directory of the entire project needs to be selected. Read out the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Deploying Applications on a Real Device

Slides 56 and 57

Deploying Applications on a Real Device 1-2**◆ Phone Setup**

- ◆ Navigate to Settings → About Phone
- ◆ Tap on the Build Number Seven Times
- ◆ Return back to the previous screen
- ◆ The Developer Options setting is available
- ◆ Click Developer Options
- ◆ Select the USB Debugging check box

On older versions of
Android prior to 4.2, the
Developer Options
settings is directly
available.



© Aptech Ltd.

Getting Started with Android/Session 2

56

Deploying Applications on a Real Device 2-2**◆ System Setup**

- ◆ Install the adb drivers for Windows
- ◆ Connect the device to your computer
- ◆ Start command prompt and Navigate to SDK directory→ platform-tools
- ◆ Use the command 'adb devices'
- ◆ The connected Android Device should be listed here
- ◆ Start Android Studio, Open the project, and Navigate to the Run menu
- ◆ Run and Click Run 'app'
- ◆ From the device selector screen
- ◆ Choose a running Device
- ◆ Select the connected Android Device
- ◆ Click OK



© Aptech Ltd.

Getting Started with Android/Session 2

57

Using slides 56 and 57, explain the process of deploying an Android application on a real device. Begin by stating that whenever the emulator does not have the required features or if the emulation is found to be too slow, a real device may be used. The steps are self-explanatory. Read out the step and explain it in detail whenever necessary. Use the images to provide an idea of how each step looks.

Summary

Slide 58

Summary

- ◆ An Android project consists of all the files and resources required to build the project into a .apk file for installation
- ◆ Activities in Android are UI Screens. Layouts are used to define the UI of these screens
- ◆ Services in Android are background processes with no UI
- ◆ Content providers are used for managing shared data sets
- ◆ Broadcast Receiver receives or responds to announcements broadcast by the system
- ◆ Resources and assets are used to hold static content of the application
- ◆ Applications can be tested on AVDs or Real Devices

Using slide 58, summarize the session. Make them revise the following points:

- An Android project consists of all the files and resources required to build the project into an .apk file for installation.
- Activities in Android are UI Screens. Layouts are used to define the UI of these screens.
- Services in Android are background processes with no UI.
- Content providers are used for managing shared data sets.
- Broadcast Receiver receives or responds to announcements broadcast by the system.
- Resources and assets are used to hold static content of the application.
- Applications can be tested on AVDs or Real Devices.

2.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes Android System Overview including Preferences, Shared Preference, the Android File System, and Notifications.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 3: Android System Overview

3.1 Pre-Class Activities

You should familiarize yourself with core components of an Android Application. You should also have a good understanding of Preferences, the Android File System, and Notifications. You should review the procedure to create different types of Notifications and the Notification API in general.

Familiarize yourself with the topics of the current session in-depth.

3.1.1 Objectives

After the session, learners will be able to:

- Explain Preferences
- Explain Shared Preferences Storage Structure
- Explain the Android File System
- Explain Notifications

3.1.2 Teaching Skills

You should be familiar with Preferences in Android and storing and retrieving values from Preferences. You should be able to identify the types of File Systems supported by Android and how to store and retrieve data from each of them. Proficiency with the Notifications API is mandatory. You should be able to list the types of Notifications supported by Android and how to create each of them.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

3.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Tell the students that the previous session explained the basics of Android Application its fundamentals, components, and frameworks.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the basics of the Android System such as Preferences, the Android File System, and Notifications.

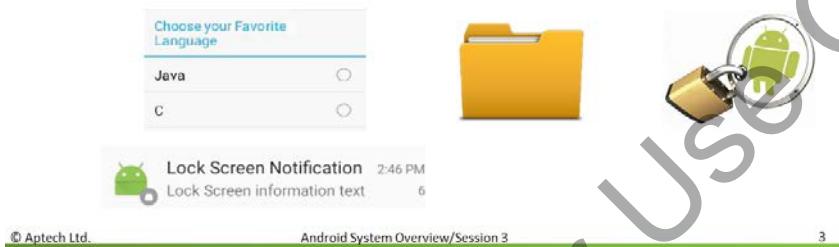
3.2 In-Class Explanations

Introduction

Slide 3

Introduction

- ◆ To create a user friendly app, the developer needs to know –
 - ❖ Setting Preferences
 - ❖ Saving Files
 - ❖ Sending Notifications
 - ❖ Android Security Model
- ◆ Customizability and saving settings can be implemented by using Preferences
- ◆ Notifications are of several types such as simple, progress bar, lock screen, and so on



Using slide 3, introduce Preferences, Files, Notifications, and the Android Security Model. Explain that Preferences can be used to store Application Settings as key value pairs whereas File System can be used to store any type of data. Notifications are messages displayed to the user outside the application. Give a brief description for the types of Notifications using the last bulleted point.

Shared Preferences

Slides 4 and 5

Shared Preferences 1-2

- ◆ Shared Preferences are used to store persistent data on the device
- ◆ The three commonly used preferences are -
 - ◆ CheckBoxPreference
 - ◆ ListPreference
 - ◆ EditTextPreference
- ◆ Preferences can be defined in XML
- ◆ Preference Manager class is used to save Preferences



© Aptech Ltd.

Android System Overview/Session 3

4

Using slide 4, explain Shared Preferences and their uses. Begin by providing the definition of Preference that is, it is a building block for a single setting. Proceed to listing out the types of preferences and providing a brief explanation for each of them. Mention that Preferences are defined in XML using XML tags. Explain the use of PreferenceManager which is the core component for reading preference value and saving changes.

Shared Preferences 2-2

- ◆ Following Code Snippet demonstrates an example for different types of preferences:

```
<?xml version="1.0" encoding="UTF-8"?>
<PreferenceScreen
    xmlns:android="http://schemas.android.
    com/apk/
    res/android">

    <PreferenceCategory
        android:title="Checkbox Preference">
        <CheckBoxPreference
        ...
        />
    </PreferenceCategory>

    <PreferenceCategory
        android:title="EditTextPreference">
        <EditTextPreference
        ...
        />
    </PreferenceCategory>

    <PreferenceCategory
        android:title="ListView Preference">
        <ListPreference ...
        ...
        />
    </PreferenceCategory>
</PreferenceScreen>
```

© Aptech Ltd.

Android System Overview/Session 3

5

Using slide 5, explain the process of adding Shared Preferences in XML. Explain the code given on slide 5. Explain the Tags PreferenceScreen and PreferenceCategory. Specify the use of CheckBoxPreference, EditTextPreference, ListPreference, and the differences between them that is, each of them represents a different type of setting.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Preferences.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/ui/settings.html>

<http://developer.android.com/reference/android/preference/package-summary.html>

<http://developer.android.com/reference/android/content/SharedPreferences.html>

Preference Fragment

Slide 6

Preference Fragment

- Following Code Snippet demonstrates an example for creating a Preference Fragment:

```
package com.example.PreferenceFragment;
Import ...;

public class PrefsFragment extends PreferenceFragment {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        addPreferencesFromResource(R.xml.preferences);
    }
}
```

Using slide 6, explain the process of adding a preference fragment from XML. Explain that the preferences created in XML now need to be loaded onto an activity or a fragment. Explain the code given on the slide in which the `addPreferencesFromResource()` method is used to load the fragment from XML. The resource ID of the XML file is passed as an argument to the method.

Preference Manager

Slide 7

Preference Manager

- Following Code Snippet demonstrates using PreferenceManager class to get preference values and loading them into the activity:

```
...  
SharedPreferences mySharedPreferences =  
PreferenceManager.getDefaultSharedPreferences(this);  
  
boolean my_checkbox_preference = mySharedPreferences.  
getBoolean("checkbox_preferencevalue", false);  
prefCheckBoxvalue.setChecked(my_checkbox_preference);  
  
String my_edittext_preference = mySharedPreferences.  
getString("edittext_preferencevalue", "");  
prefEditTextvalue.setText("Edit Box Preference:" + my_  
edittext_preference);  
  
String my_edittext_preference1 = mySharedPreferences.  
getString("lp_android_choice", "");  
prefEditTextvalueone.setText("ListBox Preference:" + my_  
edittext_preference1);  
...  
...
```

© Aptech Ltd.

Android System Overview/Session 3

7

Using slide 7, explain the process of using the Preference Manager to get preference values. Start by explaining that now we know the process for creating preferences, we will learn the process of accessing these preferences. Explain the code given on the slide. Explain the methods to retrieve preference values and how they differ from each other, that is, each of them deals with a different type of Preference.



In-Class Question: What is the use of Shared Preferences?

Answer: Shared Preferences are used to store persistent data.

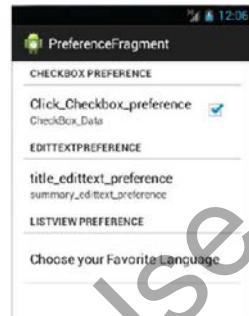
Preferences Example Application

Slides 8 to 10

Preferences Example Application 1-3

- Using the code, an application for demonstrating retrieving Shared Preferences is created as shown in the following figure:

- Click Menu and then Click Preference Example to display the output as shown in the following figure:



© Aptech Ltd.

Android System Overview/Session 3

8

Preferences Example Application 2-3

- Click Choose your Favorite Language to display the output as shown in the following figure:

- Click title_edittext_preference, to display the output as shown in the following figure:



© Aptech Ltd.

Android System Overview/Session 3

9

Preferences Example Application 3-3

- The application logic is as follows:
- Developer creates an XML file and specifies the preferences in the file
- The XML contains a list of preference objects and preference subclasses
- The preferences are loaded into the Activity Preference fragment
- The application allows the user to modify them

© Aptech Ltd.

Android System Overview/Session 3

10

Using slides 8 to 10, demonstrate the steps for creating an application using Shared Preferences. Explain that this application will store user input such that their favorite languages and so on, as preferences. Using slide 10, explain the application logic and how it utilizes the code explained in the earlier slides.

Saving Preferences

Slide 11

Saving Preferences

- Following Code Snippet demonstrates using PreferenceManager class to store custom Preference values:

```
SharedPreferences preferences = PreferenceManager
    .getDefaultSharedPreferences(MainActivity.this);

SharedPreferences.Editor editor = preferences.edit();

editor.putString("data", "value");

editor.commit();
```

© Aptech Ltd.

Android System Overview/Session 3

11

Using slide 11, explain the process of storing Shared preference values. Begin by explaining that, we will now study the process of saving values into these preferences manually. Explain the code given on the slide. A SharedPreferences.Editor object is needed for this purpose. The putXYZ() methods can be used to store values.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding `SharedPreference.Editor`.

(OR)

You can refer to the online Mirror of the same documentation using the following link:
<http://developer.android.com/reference/android/content/SharedPreferences.Editor.html>

Saving Preferences Example Application

Slides 12 and 13

Saving Preferences Example Application 1-2

- Using the explained code, an application for demonstrating saving Shared Preferences is created as shown in the following figure:
- On typing the content in the edit box and click Submit, the output will be as shown in the following figure:

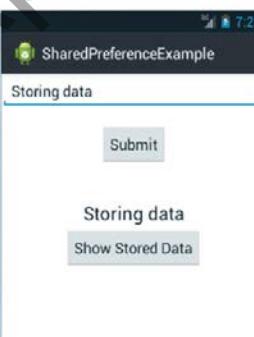


© Aptech Ltd. Android System Overview/Session 3 12

Using slide 12, explain the application created for demonstration. Begin by explaining that this application will take input from an `EditText` and store them manually as a preference.

Saving Preferences Example Application 2-2

- On clicking the Show Stored Data, the output will be as shown in the following figure:
- The application logic is as follows:



- Developer creates a `Edit Text` to accept text input from the user
- The user input is retrieved and stored using the `Shared Preferences. Editor` object
- When the user clicks the button, the value of the preference is retrieved
- The retrieved value is displayed to the user using a `TextView`

© Aptech Ltd. Android System Overview/Session 3 13

Using slide 13, explain the application logic and how it utilizes the code explained in the earlier slides. Explain the output of the application using the figure given on the slide.



In-Class Question: Which class object is used to store Shared Preference data?

Answer: SharedPreferences.Editor

File System

Slide 14

File System

- ◆ Two types of storage options supported -
 - ◆ Internal
 - ◆ External
- ◆ Several types of file systems are supported such as ext4, NTFS, and so on
- ◆ The developer can read and write files using the File APIs
- ◆ Each application has its own internal file system
- ◆ No permissions needed for internal file system
- ◆ Manifest file must explicitly specify permission to write to external storage



© Aptech Ltd.

Android System Overview/Session 3

14

Using slide 14, explain file system and the types of file systems supported by Android. Explain the differences between each file system and when to use an internal file system over an external one. Internal file system is more reliable and it should be used to store data that is critical for the functionality of the application whereas the external file system can be used to store media content and additional BLOB (Binary Large OBject) data. Explain the permission requirements for accessing the file system.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the Android File System.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/training/basics/data-storage/files.html>

<http://developer.android.com/guide/topics/data/data-storage.html>

Internal File System**Slides 15 and 16****Internal File System 1-2**

- ◆ No permissions needed
- ◆ Always available
- ◆ More secure
- ◆ Removed once, the application is deleted
- ◆ Accessed using java.io classes



© Aptech Ltd.

Android System Overview/Session 3

15

Using slide 15, explain the features of the internal file system. Explain that the internal file system is always available and hence, it is more reliable and used to store critical data for the application. Explain that internal file system of an application cannot be accessed from outside the application and hence it is more secure than external storage. Data stored on the internal file system is not necessary to be managed manually and it is automatically deleted along with the application. Procedure to access the internal file system is not different from that of accessing the traditional file system in Java using the java.io package.

Internal File System 2-2

- Following Code Snippet demonstrates an example for creating and deleting a directory on the internal file system:

```
File sdDir=new File(getFilesDir(), "file_name");
sdDir.mkdir();
file.delete();
```

- Following Code Snippet demonstrates an example for creating, reading, writing, and deleting a file on the internal file system:

```
File file=new File(getApplicationContext(), "android.txt");
file.createNewFile();

FileInputStream fis = openFileInput(file);
String text = fis.read();
fis.close();

FileOutputStream fos = openFileOutput(FILENAME, Context.MODE_PRIVATE);
fos.write(string.getBytes());
fos.close();
file.close();
file.delete();
```

© Aptech Ltd.

Android System Overview/Session 3

16

Using slide 16, explain the process of accessing the internal file system. Explain the code given on the slide. The java.io package is used for accessing the internal file system. Explain the 'File' class and its methods.

Internal File System Example Application

Slides 17 to 19

Internal File System Example Application 1-3

- Using the explained code, an application for accessing the Internal File System is created as shown in the following figure:
- Clicking the NEW FILE will create a popup for the file name and contents as shown in the following figure:

© Aptech Ltd. Android System Overview/Session 3 17

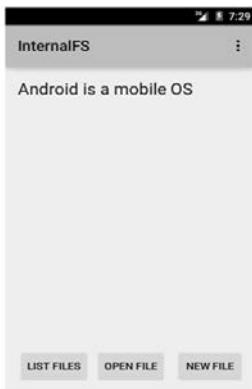
Internal File System Example Application 2-3

- After a file has been created, Clicking the LIST FILES will list the current files in the internal file system as shown in the following figure:
- Clicking the OPEN FILE button will open the dialog Box prompting for the file name to open as shown in the following figure:

© Aptech Ltd. Android System Overview/Session 3 18

Internal File System Example Application 3-3

- Clicking OK will display the file contents as shown in the following figure:



- The application logic is as follows:

- An activity with three button with obvious functionality are created
- Clicking LIST FILES will retrieve the files list and display it to the user using a TextView
- Clicking NEW FILE will create an Alert Box asking for the necessary inputs to create a file
- The File is then created using a File Object
- Clicking OPEN FILE will create an Alert Box asking for the File Name to open
- The file contents are read using a File Objects and displayed using a TextView

© Aptech Ltd.

Android System Overview/Session 3

19

Using slides 17 to 19, demonstrate the steps to create an application to use the internal file system. This application is a simple file management application which is used to store and read data from the internal file system. Explain the application logic and the output using the figure given on slide 19.



In-Class Question: What are the types of file systems supported by Android?

Answer: Internal and External file systems.

External File System**Slides 20 and 21****External File System 1-2**

- ◆ Explicitly permissions needed
- ◆ Not reliable
- ◆ More volume of storage available
- ◆ Can persist even if the app is removed
- ◆ Data can be shared between applications



© Aptech Ltd.

Android System Overview/Session 3

20

Using slide 20, explain the features of the external file system. Begin by explaining that the external file system can hold larger amounts of non-critical data. Due to its nature, it becomes less reliable and the data is not considered secure or persistent if the storage is modified. Explain that this can also be an advantage as the data can be shared among applications.

External File System 2-2

- The code to access files and directories on External File System is similar to that of the Internal File System
- Following Code Snippet demonstrates how to retrieve a File object or a path reference to the external storage:

```
//Path
String path = Environment.getExternalStorageDirectory().getAbsolutePath();

//File Object
File file = Environment.getExternalStorageDirectory();
```

© Aptech Ltd.

Android System Overview/Session 3

21

Using slide 21, explain the process of accessing the external file system. The File object that is returned represents the directory pertaining to the external storage. The rest of the API is identical to that of the internal file system.

External File System Example Application

Slides 22 to 25

External File System Example Application 1-4

- Using the explained code, an application for demonstrating accessing the External File System is created as shown in the following figure:
- Clicking MAKE DIRECTORY will create a prompt asking for directory name as shown in the following figure:

© Aptech Ltd. Android System Overview/Session 3 22

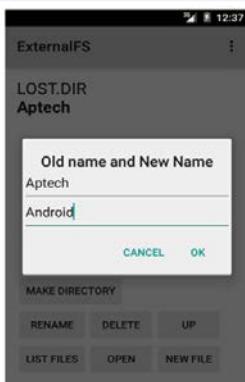
External File System Example Application 2-4

- The directory is created and the files and folders are listed
- The directories are shown in bold as shown in the following figure:
- Clicking OPEN will create prompt asking for the file/directory to open as shown in the following figure:

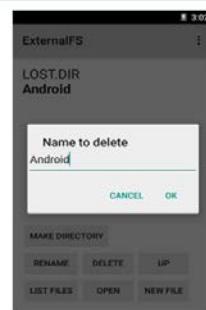
© Aptech Ltd. Android System Overview/Session 3 23

External File System Example Application 3-4

- Clicking RENAME will create prompt asking for necessary details as shown in the following figure:



- The LIST FILES and NEW FILE button function similar to the buttons of Internal File System Example
- Clicking DELETE will create prompt asking for directory to delete as shown in the following figure:



© Aptech Ltd.

Android System Overview/Session 3

24

External File System Example Application 4-4

- The application logic is as follows:

- The functional code is similar to that of the Internal Storage Example
- The External Storage directory is retrieved using Environment.getExternalStorageDirectory() method

© Aptech Ltd.

Android System Overview/Session 3

25

Using slides 22 to 25, demonstrate the steps to create an application that accesses the external file system. The application is a basic file browser which works on the external file system.

Demonstrate and explain the functionality and output of the application using the figures given on slides 22 to 24.

Using slide 25, explain the application logic and the code used to create the application.



In-Class Question: Which package contains the API to access Internal File System?

Answer: java.io

Notifications

Slide 26

Notifications

- ◆ Notifications enable an app to inform the user about events
- ◆ Every notification needs to contain the following information:
 - ◆ Small Icon
 - ◆ Title
 - ◆ Detailed Text
- ◆ Several types of notifications:
 - ◆ Normal/Basic
 - ◆ Expanded
 - ◆ Progress
 - ◆ Heads Up
 - ◆ Lock Screen



Using slide 26, explain the concept of Notifications. List the uses of Notifications and provide a common example of Notification in real life applications (Gmail, Alarm, and so on). List the mandatory information that needs to be supplied for displaying a notification. Also, list any additional information that can be passed on to a Notification. Explain the various types of notifications, providing an example whenever possible such as the Alarm uses Basic Notifications whereas the Music Player uses Progress Notification.

Normal/Basic Notification

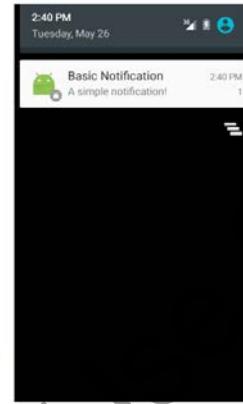
Slide 27

Normal/Basic Notification

- Following Code Snippet demonstrates creating a Basic Notifications:

```
Notification.Builder notification =new
Notification.Builder(this)
.setSmallIcon(R.drawable.ic_notification_small)
.setContentTitle("Basic Notification")
.setContentText("A simple notification!")
.setLargeIcon(BitmapFactory.decodeResource(getResources(), R.mipmap.ic_notification_large));
...
NotificationManager mNotificationManager =
(NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

mNotificationManager.notify(0,
notification.build());
```



© Aptech Ltd.

Android System Overview/Session 3

27

Using slide 27, explain the process of creating a Basic Notification. Explain the code given on the slide. Explain the `Notification.Builder` class and how it is essential to display a Notification. It is responsible for creating the `Notification` object with the required characteristics. You may also refer to the Builder design pattern to provide a better understanding to the students. Explain the `NotificationManager` class and its functionality. Explain the parameters for the `notify()` method. Use the figure given on the slide to provide an idea on what a Basic Notification looks like.

Expanded Notification

Slide 28

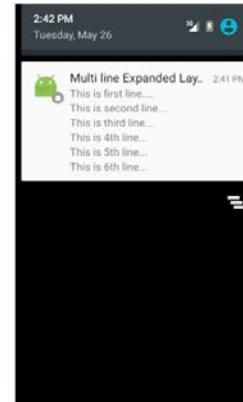
Expanded Notification

- Following Code Snippet demonstrates creating an Expanded Notifications:

```
Notification.Builder notification =new
Notification.Builder(this)
.setSmallIcon(R.drawable.ic_notification_small)
.setContentTitle("Expanded Layout Notification")
.setContentText("This notification can be
expanded")
.setLargeIcon(BitmapFactory.decodeResource(getResources(), R.mipmap.ic_notification_large));

Notification.InboxStyle inboxStyle =new
Notification.InboxStyle();
inboxStyle.setBigContentTitle("Multi line
Expanded Layout ");

for( int i=0;i<5;i++)
{
    inboxStyle.addLine(" ");
}
notification.setStyle(inboxStyle);
mNotificationManager.notify(1,notification.build());
```



© Aptech Ltd.

Android System Overview/Session 3

28

Using slide 28, explain the process of creating a Multi-Line Notification. Explain the code

given on the slide. Explain Notification Style and the process of adding multiple lines using the 'for' loop. A total of six lines are being added in the code. Use the figure given on the slide to provide an idea on what a Multi-Line Notification looks like.

Progress Notification

Slide 29

Progress Notification

- Following Code Snippet demonstrates creating a Progress Notifications:

```
final Notification.Builder notification =new
Notification.Builder(this)
.set...
..
.

notification.setProgress(100, incr, false);
mNotifyManager.notify(2, notification.build());
```



© Aptech Ltd. Android System Overview/Session 3 29

Using slide 29, explain the code and process of creating a Progress Notification. Explain the `setProgress()` method which is used to update the progress value shown in the notification. In order to provide the updates to the notification, you may use another thread which updates the progress value periodically. Use the figure given on the slide to provide an idea on what a Progress Notification looks like.

Heads Up Notification

Slide 30

Heads Up Notification

- Heads Up Notifications are not guaranteed to be displayed Heads Up mode
- Several conditions need to be met
- Following Code Snippet demonstrates creating a Heads Up Notifications:

```
Notification.Builder notification =new
Notification.Builder(this)
.set...
.

.setPriority(Notification.PRIORITY_MAX);
..

NotificationManager mNotificationManager
=(NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

mNotificationManager.notify(3,notification.build());
```



© Aptech Ltd. Android System Overview/Session 3 30

Using slide 30, explain the process of creating a HeadsUp Notification. Explain the use of the setPriority() method and the conditions that need to be met in order for the notification to be displayed as a Heads Up Notification. Emphasize that a Heads Up Notification cannot be guaranteed to be displayed and the Android Systems takes the appropriate decision. Use the figure given on the slide to provide an idea on what a HeadsUp Notification looks like.

Lock Screen Notification

Slides 31 and 32

Lock Screen Notification

- Following Code Snippet demonstrates creating a Lock Screen Notifications:

```
Notification.Builder notification =new
Notification.Builder(this)
.setSmallIcon(R.drawable.ic_notification_small)
.setVisibility(Notification.VISIBILITY_PUBLIC)
.set...;

NotificationManager mNotificationManager =
(NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

mNotificationManager.notify(4,
notification.build());
```

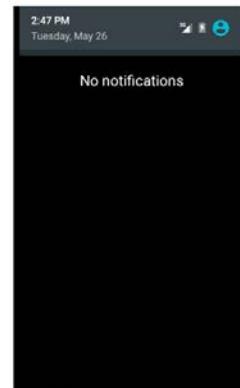
© Aptech Ltd. Android System Overview/Session 3 31

Using slide 31, explain the process of creating a Lock Screen Notification. Explain the setVisibility() method and the arguments it can take stressing on the Visibility level argument. Remind that Lock Screen widgets have been dropped and discontinued, and Lock Screen Notifications is the only way to achieve a similar functionality. Use the figure given on the slide to provide an idea on what a Lock Screen Notification looks like.

Lock Screen Notification

- Following Code Snippet demonstrates clearing all the notifications:

```
NotificationManager mNotificationManager =
(NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);
mNotificationManager.cancelAll();
```



Using slide 32, explain the process of clearing all the notifications. Explain the code given on

the slide. Explain the method `cancelAll()` which removes all the notifications. Show the output using the figure given on the slide.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Android Notifications:

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/ui/notifiers/notifications.html>

<http://developer.android.com/training/notify-user/build-notification.html>

<http://developer.android.com/reference/android/app/Notification.Builder.html>

<http://developer.android.com/reference/android/app/Notification.html>

Notification Example Application

Slides 33 and 34

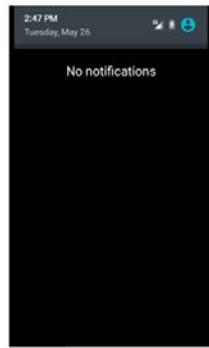
Notification Example Application 1-2

- Using the code, an application for demonstrating accessing the External File System is created as shown in the following figure:
- By clicking the Notification button, the appropriate notification is displayed as shown in the following figure:

Using slide 33, demonstrate creation of an application which displays the different types of notification based on whichever option the user selects. Give a brief explanation on each functionality.

Notification Example Application 2-2

- By clicking the CLEAR NOTIFICATIONS button, all the notifications are cleared as shown in the following figure:



- The application logic is as follows:
 - The developer creates an activity with the necessary buttons
 - Whenever a button is clicked, the onClickListener will execute the code to create the appropriate notification
 - When the CLEAR NOTIFICATIONS button is clicked, all the notifications are cleared using NotificationManager's cancelAll() method

© Aptech Ltd.

Android System Overview/Session 3

34

Using slide 34, explain the output and application logic used to implement the Notifications functionality.



In-Class Question: What are the types of Notifications supported by Android?

Answer: The Notifications supported by Android are as follows:

- Basic
- Expanded
- Progress
- Heads Up
- Lock Screen

Android Security Model

Slides 35 to 37

Android Security Model 1-3

- ◆ Core of Security is the Linux Kernel
- ◆ **Features -**
 - ◆ Permissions tailored to user needs
 - ◆ Separating processes
 - ◆ Securing Inter-Process Communication (IPC)
 - ◆ Eliminating parts of the Kernel that are not secure
 - ◆ Separating resources of one user from another
- ◆ **Permissions -**
 - ◆ Apps can only access resources they have explicit permission for
 - ◆ Permissions are displayed during installation
 - ◆ Requires User consent
 - ◆ Each application gets a different User ID
 - ◆ Application resources are protected to its User ID
 - ◆ OS data cannot be modified without root access

© Aptech Ltd. Android System Overview/Session 3 35



Using slide 35, explain the concept of Security and its importance. You may begin by specifying that Security is essential whenever there is data involved. Explain Android's implementation of Application Security and User Security. Use the bulleted points to list the features of the Android Security Model and provide a brief explanation. Explain the concept of Permission and how it helps protect Application resources. You can use the example from Linux and Windows UAC to help the students understand better. Use the bulleted points to highlight the essential facts regarding Permissions and their implementation in Android.

Additional Reference:

You may refer to the following links for more information regarding Linux Security model and Windows UAC:

<https://www.linux.com/learn/docs/727873-overview-of-linux-kernel-security-features/>
<http://windows.microsoft.com/en-in/windows/what-is-user-account-control#1TC=windows-7>

Android Security Model 2-3

- ◆ **Securing Processes & Resources**
 - ❖ Processes and resources of every application are isolated
 - ❖ Applications cannot have resources that are restricted from the OS
 - ❖ OS keeps tracks of the Process, Resources, and the respective Permissions
- ◆ **User Security Options**
 - ❖ Device Management
 - ❖ Secure Password
 - ❖ Data Encryption
 - ❖ Certification Based System
 - ❖ VPN
 - ❖ Protection Relating to Third-Party Applications



© Aptech Ltd.

Android System Overview/Session 3

36

Using slide 36, explain Android's implementation of Securing Processes and Resources. Explain the concept of isolation and permissions using the bulleted points. Also, explain the user security features provided by Android listed in the slide. You can list the bulleted points providing a brief explanation for each of them. For example, for Secure Password you can explain the various locking/password mechanisms and for Data Encryption, you can mention the Disk Encryption and lock system provided by Android, and so on.

Android Security Model 3-3

- ◆ **Security Features in Lollipop**
 - ❖ Full Disk Encryption Enabled by Default
 - ❖ Improved Encryption Procedures
 - ❖ SELinux Access Control
 - ❖ Updates to Cryptography of HTTPS and SSL
 - ❖ Position Independent Execution Removed
 - ❖ Multi User Profiles
 - ❖ Additional Locking/Unlocking Options



© Aptech Ltd.

Android System Overview/Session 3

37

Using slide 37, list the new security features introduced in Android Lollipop. Provide a brief explanation for each of them such as improved Encryption procedures use stronger encryption algorithms which take exponentially larger time to brute force.

Additional Reference:

You may refer to the following link for more information regarding Lollipop's new security features:

<https://www.androidpit.com/android-5-0-lollipop-security>

Summary

Slide 38

Summary

- Preferences are a subclass that helps users to customize their applications. A developer can specify preferences using an XML file or using code
- Title, Intents, and sub screens are the different ways that helps to group settings
- The Android file system offers internal and external storage. Internal Storage is removed along with the application. External Storage can persist
- Android's notifications enable an app to inform the user about events
- Starting from Android Lollipop, Notifications can be made to be intractable on lock screen and act as substitutes for lock screen widgets
- The Android security model provides several security features, such as, permissions, the Application Sandbox, several user security options, Full Disk Encryption, and protection from third-party applications

© Aptech Ltd.

Android System Overview/Session 3

38

Using slide 38, summarize the session. Make them revise the following points:

- Preferences are a subclass that helps users to customize their applications. A developer can specify preferences using an XML file or using code.
- Title, Intents, and sub screens are the different ways that helps to group settings.
- The Android file system offers internal and external storage. Internal Storage is removed along with the application. External Storage can persist.
- Android's notifications enable an app to inform the user about events.
- Starting from Android Lollipop, Notifications can be made to be intractable on lock screen and act as substitutes for lock screen widgets.
- The Android security model provides several security features, such as, permissions, the Application Sandbox, several user security options, Full Disk Encryption, and protection from third-party applications.

3.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes UI creating Activities, View, Styles, Themes, and handling User Input.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 4: Android User Interface (UI)

4.1 Pre-Class Activities

You should familiarize yourself with core components of an Android User Interface. You should also acquire a good understanding of UI elements and components. You should review the concepts of events and event listeners.

Familiarize yourself with the topics of the current session in-depth.

4.1.1 Objectives

After the session, learners will be able to:

- Explain the process to create the UI for Android applications
- Describe the views, layouts, UI components, styles, and themes
- Explain the procedure for handling user events

4.1.2 Teaching Skills

You should be familiar with GUI concepts of Android. You should be able to identify the types of views and layouts. Proficiency with the graphic and textual GUI editor is mandatory. You should be able to work with events and event listeners.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

4.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Tell the students that the previous session explained the basics of with core components of an Android application that is, Preferences, the Android file system, and Notifications.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the UI components of Android including views, layouts, styles, themes, events, and event listeners.

4.2 In-Class Explanations

Introduction

Slide 3

Introduction

- ◆ UI is a space that enables interaction between the user and the machine
- ◆ UI Enables the following actions:
 - ◆ Input
 - ◆ Output
- ◆ Touch screen, Graphical User Interface (GUI) and Text-based User Interface (TUI)



© Aptech Ltd. 4 Android User Interface/Session 4 3

Using slide 3, introduce GUI programming in Android. Define UI and explain that UI of an application plays an important role in its success. Explain that the GUI should be easy to understand and should be user-friendly.

Android User Interface

Slide 4

Android User Interface

- ◆ User Interface is what the user will see and interact with to perform operations
- ◆ Android comes with many friendly UI elements and layouts which help to build interactive applications
- ◆ Basic concepts of UI are:
 - ◆ Views
 - ◆ Layouts
 - ◆ UI Components



© Aptech Ltd. 4 Android User Interface/Session 4 4

Using slide 4, explain the concepts of Android's implementation of UI widgets. Explain views, layouts, and UI components. Explain that Android comes with native support for widgets known as 'Views'.

Android UI Components

Slide 5

Android UI Components

- ◆ Input Controls
- ◆ Input Events
- ◆ Menus
- ◆ Action Bar
- ◆ Settings
- ◆ Dialogs
- ◆ Additional Views
- ◆ Status Notifications
- ◆ Toasts
- ◆ Search
- ◆ Drag and Drop

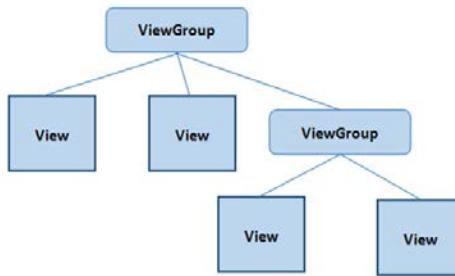
Using slide 5, explain the various UI components of Android. List them and provide a small description for each of them. For example, Input Controls accept input from the user for the application. EditText and spinner are examples of input controls.

Views

Slide 6

Views

- ◆ All UI components in Android are built using View and ViewGroup objects
- ◆ Each item in the Android UI belongs to the View class
- ◆ Views are also known as widgets
- ◆ A set of several views make a ViewGroup
- ◆ Each ViewGroup can have other ViewGroup and View within. These are called child Views



Using slide 6, explain views and their implementation. Mention that views are the widgets of Android. Explain that a ViewGroup can have multiple views or ViewGroups within.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding views and ViewGroups.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/reference/android/view/View.html>

<http://developer.android.com/reference/android/view/package-summary.html>

<http://developer.android.com/training/custom-views/create-view.html>

<http://developer.android.com/reference/android/view/ViewGroup.html>

<http://stackoverflow.com/questions/27352476/difference-between-view-and-viewgroup-in-android>

Concepts of Views

Slide 7

Concepts of Views

- ◆ Focus
- ◆ Listeners
- ◆ Attributes
 - ◆ ID
 - ◆ Setting the Padding
 - ◆ Size of a View
 - ◆ Setting View Location

Using slide 7, explain the attributes and other components of view. List them and provide brief descriptions for each of them. For example, listeners respond to events and handle them appropriately.



In-Class Question: What is a View?

Answer: View is a UI component in Android. Views are considered the widgets of Android.

Overview of Layouts

Slide 8

Overview of Layouts

- ◆ A layout is an extension of ViewGroup class
- ◆ It defines the visual structure of the UI
- ◆ It mainly comprises interconnected child views
- ◆ It helps to define the architecture of the UI present in an Activity
- ◆ The developer can either write code for specifying a layout or create an XML file
- ◆ Every element in the XML file is either a View or a ViewGroup object
- ◆ The Android framework gives the developer flexibility to use either or both methods to declare and control the UI layout

© Aptech Ltd. Android User Interface/Session 4 8

Using slide 8, explain the concept of layouts. Explain that layout inherits from the view group class. Explain that views are used to structure the UI code present in the activity. Explain that layouts can be added or modified using xml code or the graphical editor.

Advantages of Declaring UI in XML

Slide 9

Advantages of Declaring UI in XML

- ◆ It enables the developer to separate the presentation of your application from the application logic that controls its behavior
- ◆ The developer's UI descriptions are external to the application code, so that the developer can make changes without having to modify the source code and recompile
- ◆ Declaring the layout in XML also helps in easy visualization of the structure of the UI
- ◆ XML declarations are reflected in the GUI editor of the IDE
- ◆ UI declared in XML can easily be reused in other projects. As a result, it is easier to debug problems

© Aptech Ltd. Android User Interface/Session 4 9

Using slide 9, explain the advantages of defining the UI in XML. Explain that this type of implementation helps decouple the presentation layer and the application logic. Explain that all changes that are made in XML are reflected in GUI editor.

**In-Class Question:** What is a layout?

Answer: A layout is an extension of the ViewGroup class. It defines the distribution of elements on the activity.

Rules for writing Layout in XML**Slide 10****Rules for writing Layout in XML**

- ◆ Declaring UI elements in XML follows the same structure that one follows while naming classes and methods
- ◆ The direct relation between the class name and methods with that of UI element name and its attributes helps the developer to understand easily
- ◆ Vocabulary is same except for some instances where there are naming differences
- ◆ Each layout file must contain exactly one root element, which must be a View or ViewGroup object
- ◆ The developer needs to define the root element and then add additional layout objects or widgets as child elements
- ◆ Once the developer declares the default layout in a XML file, it is necessary to save it with the .xml extension in Android projects res/layout/ directory to compile it properly

Using slide 10, explain the rules for defining the layout in XML. Explain that the UI elements are represented as XML nodes. Explain that the layout files also follow the basics of XML and therefore, they can only contain one root node.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information about writing layouts in XML.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/ui/declaring-layout.html>

http://www.tutorialspoint.com/android/android_user_interface_layouts.htm

Working with XML UI

Slide 11

Working with XML UI

- ◆ **Load the XML Resource**
 - ❖ The XML file in an application is compiled into a View resource
 - ❖ The developer must load the View resource in the application code using the `onCreate()` callback method of the Activity class
 - ❖ A reference to the View resource must be provided in the `setContentView()` call back feature as well

- ◆ **Reference the Widget Programmatically**
 - ❖ Define a View or widget in the layout file, `activity_main.xml` and assign a unique ID
 - ❖ Create an instance for the View object and obtain its reference programmatically from the layout using the `findViewById()` method

Using slide 11, explain the process of working with views and objects defined in XML. Explain that the activity needs to load it's XML layout within the `onCreate()` method. Explain the process of referencing a widget programmatically. Explain that each resource is assigned a specific ID which can be used to get an object reference using the `findViewById()` method.

Loading an XML Layout/Referencing an XML View

Slide 12

Loading an XML Layout/Referencing an XML View

- ◆ The process for loading an XML layout is shown in the following Code Snippet:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}
```

- ◆ The process for referencing an XML view is shown in the following Code Snippet:

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="New Button"
    android:id="@+id/button"
    android:layout_below="@+id/seekBar"
    android:layout_alignEnd="@+id/ratingBar" />
...
Button buttonName = (Button) findViewById(R.id.button);
```

Using slide 12, explain the process of referencing an XML view in code. Explain the code given on the slide. Explain that the activity sets its content to the layout resource file. Explain that a button with the id 'button' is created and referenced within the code using the `findViewById()` method.

Working with Layouts

Slides 13 and 14

Working with Layouts 1-2

- ◆ It is preferable that the developers work using XML from external resources
- ◆ The XML layout will comprise a root node
- ◆ This root node can have multiple nested layouts and views for designing the UI
- ◆ Layouts can be defined in two views:
 - ◆ Graphical View
 - ◆ Text View

© Aptech Ltd.

Android User Interface/Session 4

13

Using slide 13, explain the steps to work with layouts and utilize them properly. Using the bulleted points, explain the key points that the students need to keep in mind when working with layouts.

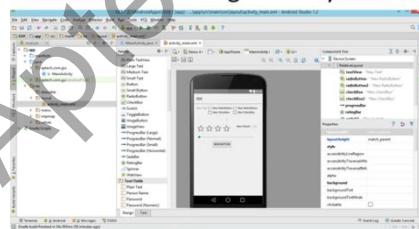


In-Class Question: Which method is used to set the layout of the activity?

Answer: The `setContentView()` method is used to set the layout of the activity.

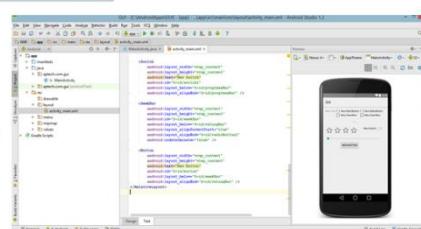
Working with Layouts 2-2

- ◆ The Views for editing XML Layouts is shown in the following figures:



Graphical View

Text View



© Aptech Ltd.

Android User Interface/Session 4

14

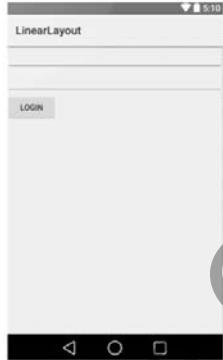
Using slide 14, explain the difference between the Text View and the Graphical View. Using the figures, explain that the Graphical View allows to directly drag and drop widgets, whereas Text View requires the developer to write the XML code which is then reflected in the GUI editor.

Linear Layout

Slide 15

Linear Layout

- ◆ It is the simplest type of layout
- ◆ Linear Layout aligns all its child nodes in a single direction, either vertically or horizontally
- ◆ The android:orientation attribute helps the developer to specify the layout direction
- ◆ A scrollbar appears if the window length exceeds the length of the screen



© Aptech Ltd. Android User Interface/Session 4 15

Using slide 15, explain Linear layout. Explain that the linear layout displays its views and ViewGroups in a linear manner. However, the developer can specify if they need to be displayed horizontally or vertically.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding different types of layouts.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/ui/layout/linear.html>
<http://developer.android.com/guide/topics/ui/layout/relative.html>
<http://developer.android.com/guide/topics/ui/layout/listview.html>
<http://developer.android.com/guide/topics/ui/layout/gridview.html>

Linear Layout Example

Slide 16

Linear Layout Example

- The process creating a Linear layout is shown in the following Code Snippet:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" >

    <EditText
        android:id="@+id/userName" />

    <EditText
        android:id="@+id/passWord" />

    <Button
        android:text="Login"
        android:id="@+id/button" />

</LinearLayout>
```

Using slide 16, explain the process of declaring a Linear Layout. Explain the code given on the slide. A Liner Layout is created which encompasses two EditTexts and a button.

Relative Layout

Slide 17

Relative Layout

- The developer can specify the location of child objects relative to each other or to the parent
- It can remove the need for nested view groups keeping the layout tree simple
- If a developer has several nested linear layout groups, the groups could be replaced with a single relative layout instead



Using slide 17, explain relative layout. Explain that unlike linear layout, relative layout has no concrete distribution of internal views/ViewGroups. Each object is placed at a relative distance from another.

Relative Layout Example

Slide 18

Relative Layout Example

- The process creating a Relative layout is shown in the following Code Snippet:

```
<RelativeLayout android:layout_width="match_parent"
    android:layout_height="match_parent"    android:id="@+id/relativeLayout">

    <TextView
        android:layout_alignParentStart="true" />

    <TextView
        android:layout_toStartOf="@+id/editText" />

    <EditText
        android:layout_toEndOf="@+id/editText" />

    <EditText
        android:layout_toEndOf="@+id/userName" />

    <Button
        android:layout_alignStart="@+id/editText2" />
</RelativeLayout>
```

© Aptech Ltd.

Android User Interface/Session 4

18

Using slide 18, explain the process of creating a relative layout. Explain the code given on the slide. Explain the attributes `layout_align` and `layout_to` attributes which specify the position of the view with respect to another view.

Table Layout

Slide 19

Table Layout

- The screen area is divided in a tabular format
- The developer can specify the row and column position for each view inside the layout
- It is one of the most commonly used layouts in UI design



© Aptech Ltd.

Android User Interface/Session 4

19

Using slide 19, explain the Table layout. Explain that Table layout divides the area into rows and columns. Explain that each view occupies a specific cell which can be expressed as the row and column number.

List View**Slide 20**

- ◆ ListView is a view group that displays a list of scrollable items
- ◆ An Adapter that pulls content from a source, such as a query or an array, helps to insert the list items automatically
- ◆ Each item result is converted into a View and added to the list by the Adapter

Using slide 20, explain the ListView. Explain that ListView is used in almost every application. Explain that ListView uses an adapter to pull data. Explain the output of using a list view using the figure given on the slide.

List View Example**Slides 21 and 22****List View Example 1-2**

- ◆ The process of creating a List View is shown in the following Code Snippet:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".ListActivity">

    <ListView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/listView"
        android:layout_alignParentTop="true"
        android:layout_alignParentStart="true" />
</RelativeLayout>
```

Using slide 21, explain the process of adding a ListView to an activity. Explain the xml code given on the slide. Explain that the ListView tag is added. Explain the commonly used attributes of the ListView.

List View Example 2-2

- The process of adding elements to a List View is shown in the following Code Snippet:

```
ArrayList contactsList = new ArrayList<Map<String, Object>>();
Map<String, Object> contact = new HashMap<String, Object>();
contact.put("name", "John");
contact.put("number", "+1 248 242 311");
contactsList.add(contact);
...
SimpleAdapter adapter = new SimpleAdapter(this, contactsList, android.R.layout.simple_list_item_2, new String[]{"name", "number"}, new int[]{android.R.id.text1, android.R.id.text2});
listView.setAdapter(adapter);
...
...
```

© Aptech Ltd.

Android User Interface/Session 4

22

Using slide 22, explain the steps to add the elements to a ListView using a SimpleAdapter. Explain the code given on the slide. Explain that SimpleAdapter object is created and data is loaded from a HashMap.

Grid View

Slide 23

Grid View

- This layout displays a scrolling grid consisting of rows and columns
- GridView is a ViewGroup that displays items in a scrollable grid
- The grid items are automatically inserted to the layout using a ListAdapter



© Aptech Ltd.

Android User Interface/Session 4

23

Using slide 23, explain the concept of GridView. Explain that similar to ListView, GridView also uses an adapter to load data. Provide an example of an application using GridViews such as the Gallery app.

Grid View Example**Slides 24 and 25****Grid View Example 1-2**

- The process creating a Grid View is shown in the following Code Snippet:

```
<?xmlversion="1.0"encoding="utf-8"?>
<GridViewxmlns:android="http://schemas.android.com/apk/res/
    android"
    android:id="@+id/gridView1"
    android:numColumns="auto_fit"
    android:gravity="center"
    android:columnWidth="100dp"
    android:stretchMode="columnWidth"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
</GridView>
```

© Aptech Ltd.

Android User Interface/Session 4

24

Using slide 24, explain the process of adding a GridView into a layout. Explain the code given on the slide. Explain that the GridView tag is added and then, describe its commonly used attributes.

Grid View Example 2-2

- The process of adding elements to a Grid View is shown in the following Code Snippet:

```
staticfinal String[] IMAGES = new String[] { "Image 1",
    "Image 2", "Image 3", "Image 4" };
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    gridView = (GridView) findViewById(R.id.gridView1);
    gridView.setAdapter(new ImageAdapter(this, IMAGES));
    gridView.setOnItemClickListener(new
        OnItemClickListener() {
            public void onItemClick(AdapterView<?> parent,
                View vw,
                int position, long id) {
                Toast.makeText(getApplicationContext(),
                    "Click ListItem Number "+ position,Toast.LENGTH_LONG)
                    .show();
            }
        });
}
```

© Aptech Ltd.

Android User Interface/Session 4

25

Using slide 25, explain the process of loading data into a GridView. Explain the code given on the slide. Explain that an ImageAdapter is used to push data into the GridView.

UI Components

Slide 26

UI Components

- UI components are interactive controls in an app's UI that enables the user to perform a wide range of actions
- Through UI components, the user can interact with the application
- Some of the basic UI components are Button, TextView, DatePicker, ProgressBar, and so on



© Aptech Ltd. Android User Interface/Session 4 26

Using slide 26, explain the concept of UI components. Explain that this includes input and output widgets such as buttons, TextViews, and progress bars.

Input Controls

Slide 27

Input Controls

- Android provides a number of input controls
Following table lists the different input controls:

Input Control	Syntax
Button	<Button ... />
Check box	<CheckBox ... />
Radio button	<RadioButton ... />
Spinner	<Spinner ... />
Picker	<DatePicker ... />
Switch	<Switch ... />
SeekBar	<SeekBar ... />
Toggle Button	<ToggleButton ... />
Text Input	<EditText ... />

© Aptech Ltd. Android User Interface/Session 4 27

Using slide 27, explain the various input controls supported by Android. List the controls and provide the syntax to add to the layout file.

Display Views

Slide 28

Display Views

- Android provides a number of Display Views
- Following table lists the different Display Views:

Display View	Syntax
Text View	<TextView ... />
Progress Bar	<ProgressBar ... />
Image View	<ImageView ... />
Video View	<VideoView ... />

© Aptech Ltd.

Android User Interface/Session 4

28

Using slide 28, explain the various display view supported by Android. Explain that display views are used to show the output to the user. List the display views and provide the syntax to add them to a layout.



In-Class Question: What is the difference between input controls and display views?

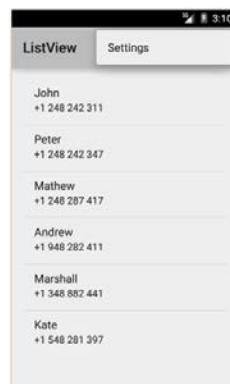
Answer: Input Controls are used to accept input from the user whereas, display views are used to display the output of the application to the user.

Menus

Slide 29

Menus

- ◆ Menu is an UI component which holds several items that provide navigation or settings or more functionality to an application
- ◆ It is a common interface component seen in Android phones and appears when menu buttons on the device are clicked
- ◆ The menu displays all available options
- ◆ Android 4.0 onwards, the standard menu button on device has been replaced with the Action bar menu button



© Aptech Ltd.

Android User Interface/Session 4

29

Using slide 29, explain the concept of menu. Explain that menus are used to display settings and options which are not used quite frequent. Explain that starting from Android 4.0, the standard menu button has been replaced with an action bar. Show the structure of a menu using the figure given on the slide.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding menus and the action bar.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/ui/menus.html>

<https://developer.android.com/training/basics/actionbar/index.html>

<http://developer.android.com/guide/topics/ui/actionbar.html>

Action Bar

Slide 30

Action Bar

- ◆ Action bar is always present at the top of the Android screen
- ◆ Using it, the user can navigate or perform an action
- ◆ It provides user actions and navigation modes
- ◆ Commonly used actions can directly be taken from options menu and placed in the action bar
- ◆ Other options are available in 'overflow menu' button in the action bar

App/ActionBar/Actio... 8:314
Selected Item: Edit

© Aptech Ltd. Android User Interface/Session 4 30

Using slide 30, explain the concept of an action bar. Explain that the action bar functions both as a tool bar and an option menu bar. Using the figure given on the slide, provide an example of an action bar. Explain that the action bar can be used to provide navigation functionality.

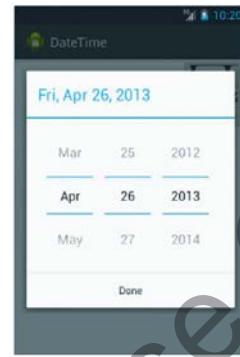
Dialogs

Slide 31

Dialogs

- ◆ Dialogs are prompt or alert displayed to the user to take a decision or to input any information
- ◆ The dialogs are also used to notify user when a task has been completed
- ◆ It does not fill the entire screen and usually appears when a user has to take a particular action before proceeding
- ◆ The different types of Dialogs are:
 - ◆ AlertDialog
 - ◆ Toast
 - ◆ TimePicker
 - ◆ DatePicker

© Aptech Ltd. Android User Interface/Session 4 31



Using slide 31, explain the concept of dialogs. Explain that dialogs are prompts or alerts that are displayed on top of an existing activity. Explain that dialogs are commonly used for notifying the user of simple information or to accept small amounts of input from the user.

Additional Views

Slide 32

Additional Views

- ◆ Android contains some important Views such as:
 - ◆ **TabHost** – Is used to maintain tabs in an application such as in default contacts application. Each tab contains child layout to navigate within
 - ◆ **WebView** – Is used to load URL. WebView is like a browser which will display the Web content within
 - ◆ **SearchView** – Is used to provide search capability with a search provider within the application

© Aptech Ltd. Android User Interface/Session 4 32



Using slide 32, explain the remaining additional views supported by Android. Explain the purpose of TabHost, WebView, and SearchView using the bulleted points provided on the slide. Explain that TabHost is commonly used in application when there is a need to rapidly navigate between multiple activities.

**In-Class Question:** What is a dialog?

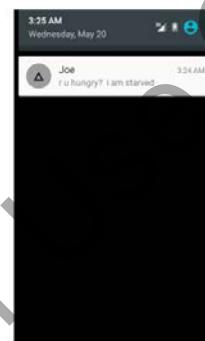
Answer: Dialogs are prompts and alerts which display information or accept input without transitioning away from the current activity.

Notifications

Slide 33

Notifications

- ◆ Notifications are used to notify or provide alerts to the user when a message or notification arrives
- ◆ Notification contains icon, title, body, and the notification arrival time
- ◆ A status bar notification displays an icon on the status bar along with a message
- ◆ When the notification is chosen, an Intent is sent by Android to launch the Activity
- ◆ The status bar notification can be initiated by an Activity or Service class
- ◆ The user can view the details by opening the 'Notifications drawer'



Using slide 33, explain the concept of notifications. Explain that notifications are used to provide brief alerts to user about events. Explain the mandatory information that needs to be provided to display a notification. Explain that notifications first appear as a small icon on the status bar and the user can then have a detailed view by using the notification drawer.

Toasts

Slide 34

Toasts

- ◆ These are simple messages that provide feedback about a user's action in a popup window
- ◆ It is displayed in a small window and does not interfere with the user's ongoing activity



Using slide 34, explain the concept of toasts. Explain that toasts are the MessageBox of Android and they are used to display simple one or two lined information. Explain that displaying a toast does not load a different activity.

Search, Drag and Drop, and Accessibility

Slide 35

Search, Drag and Drop, and Accessibility

- ◆ **Search**
 - ◆ It is an important feature in Android and enables user to search for an item in the gadget or the Internet by entering a keyword
- ◆ **Drag and Drop**
 - ◆ This UI component allows user to move data from one View to another using a graphical drag and drop gesture
 - ◆ The framework consists of drag event class, drag listeners, helper methods, and classes
- ◆ **Accessibility**
 - ◆ Android has accessibility features to cater to the needs of users with special challenges like visual impairment or hearing difficulty
 - ◆ This includes 'text-to-speech' converter, audio prompting, gesture navigation, trackball, and directional pad navigation



The screenshot shows a smartphone screen with a search bar at the top containing the text "Restau". Below the search bar is a list of search suggestions: "restaurants", "restaurants near me", "restaurant week", "restaurant depot", "restaurant week nyc", "restaurant impossible", "restaurant week chicago", and "restaurant supply". Each suggestion has a small edit icon next to it.

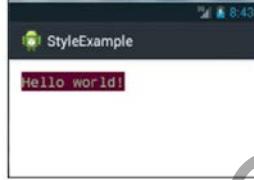
Using slide 35, explain the functionality of search, drag and drop, and accessibility. Explain that search helps locate application on the device as well as retrieve results from the Web. Explain the output of using search with the help of the figure given on the slide. Explain the concept of drag and drop and mention that it makes the application's UI more natural and user friendly. Provide an example of application that uses drag and drop functionality such as Google drive. Explain that accessibility includes features which cater to the needs of user with lack of vision or hearing disability.

Styles and Themes

Slide 36

Styles and Themes

- ◆ Presentation and formatting of an application can be improved using Styles and Themes
- ◆ Android does possess default styles and themes
- ◆ A style can be referred to as a collection of properties that specify the look and format for a View or window in UI
- ◆ All of the attributes related to style can be removed from the XML layout file and incorporated into a style definition file
- ◆ A theme is a style applied to an entire Activity or application, rather than an individual View



© Aptech Ltd. Android User Interface/Session 4 36

Using slide 36, explain styles and themes in Android. Explain that styles in Android are similar to CSS for HTML elements. Explain that Android comes with default styles and themes which can be replaced or customized to suit the needs. Explain that theme is a style applied to an entire activity.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding styles and themes.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/ui/themes.html>

<http://developer.android.com/guide/topics/resources/style-resource.html>

http://www.tutorialspoint.com/android/android_styles_and_themes.htm

Style Properties and Themes Application

Slide 37

Style Properties and Themes Application

- ◆ **Style Properties**
 - ◆ Some style properties are not supported by any View element and can only be applied as a theme
 - ◆ These style properties apply to the entire window and not to any type of View
 - ◆ This kind of style properties do not belong to any View object
 - ◆ To find out theme-only style properties, the user can look at the 'R.attr' reference for attributes that begin with window
- ◆ **Application of Themes**
 - ◆ Adding the 'style' attribute to a View element in the XML for the layout. This is added to an individual View
 - ◆ Adding the android:theme attribute to the <activity> or <application> in the Android manifest file that sets the style to the entire application

© Aptech Ltd. Android User Interface/Session 4 37

Using slide 37, explain style properties and themes. Explain that each view has its own set of properties that it supports. Explain that the user can define theme specific properties such as classes in CSS. Explain that themes are aggregations of styles which apply styles to all the elements of an activity.

Styles and Themes Example

Slides 38 and 39

Styles and Themes Example 1-2

- The process of defining and applying a style is shown in the following Code Snippet:

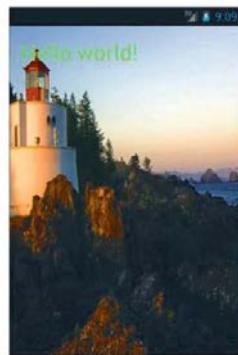
```
<resources>
<style name="NewStyle" parent="@android:style/TextAppearance.Medium">
<item name="android:layout_width">fill_parent
</item>
<item name="android:layout_height">wrap_content
</item>
<item name="android:background">#551033
</item>
<item name="android:textColor">#86C67C
</item> </style>
</resources>
...
<TextView
    style="@style/NewStyle"
    android:text="@string/hello_world"/>
...
<application
    android:theme="@style/NewStyle" >
```

© Aptech Ltd. Android User Interface/Session 4 38

Using slide 38, explain the process of defining and applying styles and themes. Explain the code given on the slide. Explain the attributes to define the styles such as background and textColor.

Styles and Themes Example 2-2

- Using the code, an application for demonstrating Styles and Themes is created as shown in the following figure:



Using slide 39, demonstrate the application that utilizes styles and themes. Explain that the text color and background are added using themes and loaded into the application. Explain the output of the application using the figure given on the slide.

Handling User Events

Slide 40

Handling User Events

- ◆ An event refers to a response generated for an external action
- ◆ The Android framework maintains an Event Queue in which the events are arranged as they occur
- ◆ The events are removed on First-In-First-Out (FIFO) basis
- ◆ The two actions that need to be performed by the developer to inform the user about the user input events are as follows:
 - + Defining an event listener and registering it with the view
 - + Overriding an existing callback method for the view

© Aptech Ltd. Android User Interface/Session 4 40



Using slide 40, explain user events and event handlers. Define event and explain that Android implements the event handling functionality from Java. Explain that events are added to queue and are cleared on First-In-First-Out bases. Explain the steps that should be taken by the developer to handle user events.

Event Listeners

Slide 41

Event Listeners

- Each of the event listeners consist of a single callback method which are invoked when the view to which the listener is registered generates an event due to user interaction
- Following table lists the different Event Listeners:

Event Listener	Method
<code>View.OnClickListener</code>	<code>onClick()</code>
<code>View.OnLongClickListener</code>	<code>onLongClick()</code>
<code>View.OnFocusChangeListener</code>	<code>onFocusChange()</code>
<code>View.OnKeyListener</code>	<code>onKey()</code>
<code>View.OnTouchListener</code>	<code>onTouch()</code>
<code>View.OnCreateContextMenuListener</code>	<code>onCreateContextMenu()</code>

Using slide 41, explain the most commonly used event listeners. List the Even Listeners and the corresponding method that needs to be implemented.



In-Class Question: What is an event?

Answer: An event refers to a response generated to an external action.

Event Listener Callback Methods

Slide 42

Event Listener Callback Methods

- ◆ **onKeyDown(int, KeyEvent)** - Invoked when a key has been pressed and was not handled by the Views within an Activity
- ◆ **onKeyUp(int, KeyEvent)** - Invoked when a key was released and was not handled by the Views within an Activity
- ◆ **onTrackballEvent(MotionEvent)** - Invoked when a trackball motion event occurs
- ◆ **onTouchEvent(MotionEvent)** - Invoked when a touch screen motion event occurs
- ◆ **onFocusChanged(boolean, int, Rect)** - Invoked when the View gains or loses focus

© Aptech Ltd.

Android User Interface/Session 4

42

Using slide 42, explain the event listener callback methods. Explain that these methods are invoked whenever the appropriate event is triggered.

Event Listener Example

Slides 43 to 48

Event Listener Example 1-6

- ◆ The process for creating and registering an Event Listener is shown in the following Code Snippet:

```

private View.OnClickListener addButtonListener = new View.OnClickListener()
{
    public void onClick(View v) {
        ...
    }
};
```

© Aptech Ltd.

Android User Interface/Session 4

43

Event Listener Example 2-6

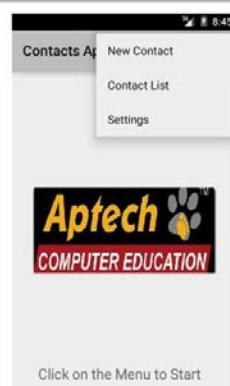
- Using the code, an application for demonstrating Event Listeners is created as shown in the following figure:



© Aptech Ltd.

Android User Interface/Session 4

- Open the Menu and select New Contact as shown in the following figure:



44

Event Listener Example 3-6

- The New Contact Screen is displayed. Enter the details as shown in the following figure:

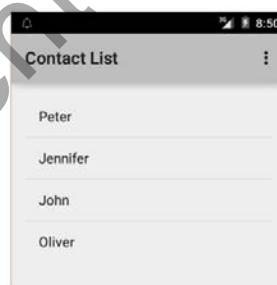
A screenshot of the "New Contact" screen. It has fields for Name (Peter), Phone Number (+1 15 812 694), Nick Name (Pete), Address (31, Wilson Street, NY), and Notes (Family Friend). At the bottom is a button labeled "ADD CONTACT".

© Aptech Ltd.

Android User Interface/Session 4

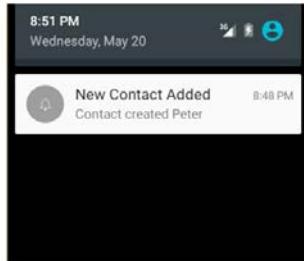
45

- Repeat these steps to add multiple contacts as shown in the following figure:

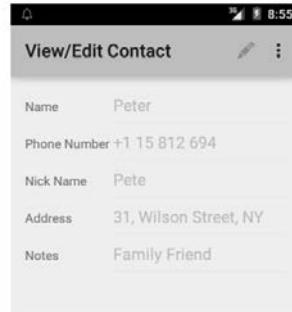


Event Listener Example 4-6

- A notification is displayed when a contact is added. This can be viewed within the notification drawer as shown in the following figure:



- Click any of the Contacts to view contact details as shown in the following figure:



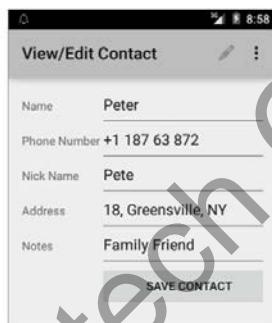
© Aptech Ltd.

Android User Interface/Session 4

46

Event Listener Example 5-6

- Click the Edit Action button to enable editing the contact. Edit the details as shown in the following figure:



- Click Save Contact. The contact details are saved. A toast is displayed to notify this information as shown in the following figure:



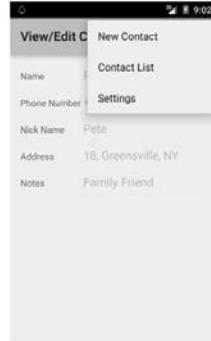
© Aptech Ltd.

Android User Interface/Session 4

47

Event Listener Example 6-6

- The user may return to the New Contact screen or the Contact List using the menu as shown in the following figure:



© Aptech Ltd.

Android User Interface/Session 4

48

Using slides 43 to 48, demonstrate the creation of an application utilizing event listeners. Using slide 43, explain the code to register an event listener for button clicks. Explain that the `onClick()` method of the `View.OnClickListener` class is implemented to handle the button clicks. The application is a contacts application, which stores user's contact information such as mobile number, name, and address. Explain the functional logic and the output of the application using the figures given on the slides.

Summary

Slide 49

Summary

- All UI components in Android are built using `View` and `ViewGroup` objects. `View` draws something on the screen that the user can interact with
- Layout defines the visual structure of UI as in the case of an activity or an app widget. There are different types of layouts such as Linear layout, Relative layout, and Grid layout to name a few
- In Android, UI components are interactive controls in an app's UI that enables the user to perform a wide range of actions. Some common GUI components are buttons, picker, and so on
 - A style is a collection of properties that specify the look and format for a `View` or window
 - A theme is a style applied to an entire Activity or application, rather than an individual View
- In order to handle a particular event, it is necessary that the Event Listener implements the corresponding callback or Event Handler in response

© Aptech Ltd.

Android User Interface/Session 4

49

Using slide 49, summarize the session. Make them revise the following points:

- All UI components in Android are built using `view` and `ViewGroup` objects. `View` draws something on the screen that the user can interact with.
- Layout defines the visual structure of UI as in the case of an activity or an app widget.

- There are different types of layouts such as Linear layout, Relative layout, and Grid layout to name a few.
- In Android, UI components are interactive controls in an app's UI that enables the user to perform a wide range of actions. Some common GUI components are buttons, picker, and so on.
- A style is a collection of properties that specify the look and format for a View or window.
- A theme is a style applied to an entire Activity or application, rather than an individual View.
- In order to handle a particular event, it is necessary that the Event Listener implements the corresponding callback or Event Handler in response.

4.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes advanced UI components including adapters, dialogs, app widgets, and concepts of material design philosophy.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 5: More UI Elements

5.1 Pre-Class Activities

You should familiarize yourself with advanced concepts of an Android UI. You should also acquire a good understanding of Adapters and Views that use adapters. You should review the different types of dialogs supported by Android and how to utilize them. You should also familiarize yourself with Application widgets and the concepts of material design.

Familiarize yourself with the topics of the current session in-depth.

5.1.1 Objectives

After the session, learners will be able to:

- Explain the use of adapters
- Identify the different types of adapters
- Describe the advanced and complex UI components
- Explain the use of custom dialogs
- Explain and create custom widgets
- Explain and use Material Design Philosophy

5.1.2 Teaching Skills

You should be familiar with Adapters in Android, utilizing existing Adapters, and creating new Adapters. You should be able to identify the various dialogs supported in Android and how to utilize them. You should be able to create new application widgets. You should be able to list and explain the Material design philosophy and its importance.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

5.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Tell the students that the previous session explained the basics of Android UI development such as views, layouts, UI components, styles, and themes.

Overview of the Session

Here, give the students an overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the basics of the Android system such as Adapters, Dialogs, Application widgets, and Material design.

5.2 In-Class Explanations

Introduction

Slide 3

Introduction

- An Adapter is an object that acts as a bridge between the UI Components and the underlying data source
- Dialogs are interactive popup windows
- App Widgets are miniature application views that can be embedded in other applications and receive periodic updates
- Material Design is Google's new UI Design philosophy



© Aptech Ltd. More UI Elements/Session 5 3

Using slide 3, introduce Adapters, Dialogs, App widgets, and Material design. Explain that Adapter acts as a bridge between the data and UI, and it is used to push data to containers such as ListView and GridView. Describe the concept of dialogs. Define application widgets and provide an example such as weather widgets and clock widgets. Mention that Material design is Google's new optional design philosophy.

Adapters

Slide 4

Adapters

- An Adapter is an object that acts as a bridge between the UI Components such as List view, Grid view, and so on and the underlying data source
- The underlying data source can be an Array, database, and so on that fills data to the UI component
- Adapter is responsible for providing view for every element of the data source



© Aptech Ltd. More UI Elements/Session 5 4

Using slide 4, explain the concept of adapters. Explain that adapter bridges the gap between the model (data) and the view (GUI).

Explain that adapters provide a consistent API irrespective of the underlying data source.

Types of Adapters

Slides 5 to 9

Types of Adapters 1-5

- ◆ **BaseAdapter**
 - ❖ BaseAdapter is a common implementation for the adapter which can be used for both ListView and Spinner
 - ❖ It is an abstract base class for the adapter interface
 - ❖ The developer can implement his/her own adapters using BaseAdapter

- ◆ **SimpleAdapter**
 - ❖ A SimpleAdapter helps to map static data to the views defined in the XML
 - ❖ The data can be mapped to the view by using ArrayList of Maps and each element in the array will be represented as a separate row in the view

Types of Adapters 2-5

- ◆ **ArrayAdapter**
 - ❖ ArrayAdapter is backed by an array of objects to load the data to the UI view
 - ❖ In other words, it is used to bind an array of data to the view
 - ❖ It overrides the `getView()` method to inflate, populate, and return a custom view for the provided array data

- ◆ **SimpleCursorAdapter**
 - ❖ An adapter which maps columns from a cursor to TextViews or ImageViews defined in an XML file
 - ❖ The developer can specify which columns to be displayed, in which views the developer wants to display the columns, and the XML file that defines the appearance of these views

Types of Adapters 3-5

- ◆ **CursorAdapter**

- ◆ This Adapter that is used for exposing the data from a Cursor to a ListView object using the column named `_id`

- ◆ **ResourceCursorAdapter**

- ◆ ResourceCursorAdapter is very similar to the CursorAdapter, which doesn't have a new View method and is used for simple views
 - ◆ It is used to create views defined in an XML file. This Adapter is deprecated in API Level 11

Types of Adapters 4-5

- ◆ **SpinnerAdapter**

- ◆ SpinnerAdapter acts as a bridge between the spinner component and the data source
 - ◆ SpinnerAdapter allows displaying of data in the following two ways:
 - ◆ Displays data in the spinner view
 - ◆ Displays data in the drop-down list when the spinner is pressed

- ◆ **SimpleCursorTreeAdapter**

- ◆ This Adapter can be used to map column data from the Cursor to the TextView or ImageView as defined in the XML file
 - ◆ The developer can specify the required columns and separate child views to display the specified content. Binding is done using the `setViewValue()` method of the SimpleCursorTreeAdapter, `ViewBinder`
 - ◆ The method returns a boolean value which can be used to determine whether binding has been successful

Types of Adapters 5-5

- ◆ **CursorTreeAdapter**
 - ❖ This Adapter can be used to display data in an expandable list view by using the data from the cursor
 - ❖ In this, the top-level Cursor exposes the group and the `getChildrenCursor(Cursor)` method returns cursors which exposes the child elements within the particular group
- ◆ **HeaderViewListAdapter**
 - ❖ `HeaderViewListAdapter` can be used to implement a `ListView` having a header at the top
- ◆ **WrapperListAdapter**
 - ❖ `WrapperListAdapter` can be used to wrap another `ListAdapter`
 - ❖ The method, `getWrappedAdapter()` is invoked for retrieving the wrapped adapter

Using slides 5 to 9, explain the various adapter classes provided by Android. Explain that all of these inherit from the same base class and provide the same interface. List the classes and explain their purpose using the bulleted points given in the slides. Explain that the cursor adapters can be used to provide data from cursor objects. Mention that each adapter has its own uses.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the Adapter classes supported by Android.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/reference/android/widget/Adapter.html>
<http://developer.android.com/reference/android/widget/SimpleAdapter.html>
<http://developer.android.com/reference/android/widget/ArrayAdapter.html>
<http://developer.android.com/reference/android/widget/CursorAdapter.html>
<http://developer.android.com/reference/android/widget/SpinnerAdapter.html>



In-Class Question: Which class needs to be extended to create custom adapters?

Answer: The `BaseAdapter` class needs to be extended to create custom adapters.

Implementation Using BaseAdapter

Slides 10 and 11

Implementation Using BaseAdapter 1-2

- The process for implementing a simple adapter using the BaseAdapter class is shown in the following Code Snippet:

```

public class MultiSelectionAdapter<T>
    extends BaseAdapter {
    ...
    public MultiSelectionAdapter(Context context,
        ArrayList<T> list) { ... }
    ...
    public ArrayList<T> getCheckedItems() { ... }
    ...
    public int getCount() { return mList.size(); }
    ...
    public Object getItem(int position) { return mList.get(position); }
    ...
    public long getItemId(int position) { return position; }
}

@Override
public View getView(int position, View convertView,
    ViewGroup parent) { ... }

mCheckbox.setOnCheckedChangeListener(new
    OnCheckedChangeListener() {
        ...
        @Override
        public void onCheckedChanged(CompoundButton buttonView,
            boolean isChecked) {
            ...
        }
    }
)

```

© Aptech Ltd. More UI Elements/Session 5 10

Using slide 10, explain the process of implementing a BaseAdapter. Explain the code given in the slide. Explain that a new adapter class called MultiSelectionAdapter is implemented extending the BaseAdapter class. The class stores its data in an ArrayList and implements the required methods.

Implementation Using BaseAdapter 2-2

- Using the code, an application for demonstrating Adapters is created as shown in the following figure:

© Aptech Ltd. More UI Elements/Session 5 11

Using slide 11, demonstrate the steps to create an application that displays a list of multi-selectable options. The application features a list of devices and it displays the selected options as a toast when the view selected button is clicked.

Advanced UI Components

Slide 12

Advanced UI Components

- ◆ UI components act as an interface between the user and the application
- ◆ Advanced Android UI components:
 - ◆ Listview
 - ◆ ScrollView
 - ◆ TabBar
 - ◆ WebView
 - ◆ ViewFlipper
 - ◆ VideoView



© Aptech Ltd. More UI Elements/Session 5 12

Using slide 12, explain the concepts of UI components such as WebView, ViewFlipper, and VideoView. List the components and provide a brief description about their purpose such as the ListView which is used to display a list of elements to the user.

ListView

Slide 13

ListView

- ◆ ListView is used for displaying a list of scrollable data
- ◆ Data is displayed by using an object of the Adapter class
- ◆ The Adapter class acts as a bridge between view and the data source
- ◆ The Adapter is used for inserting data into the list that is retrieved from an array or a database using a query




© Aptech Ltd. More UI Elements/Session 5 13

Using slide 13, explain the concepts of ListView. Explain that the ListView pulls its data from an adapter and the adapter class acts as a bridge between the view and the data source.

ScrollView

Slide 14

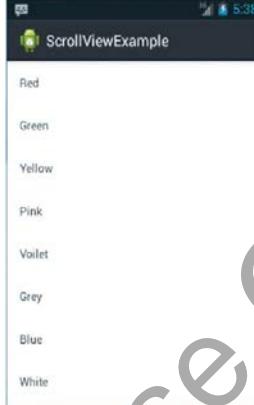
ScrollView

- ScrollView is a container that holds child views
- It has a property of scrolling when the child items size in the container exceeds the screen size
- A ScrollView is a FrameLayout, meaning that the developer should place one child in it containing the entire contents to scroll
- ScrollView should not be used with ListView

Following Code Snippet demonstrates an example for creating a ScrollView:

```
<ScrollView
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <LinearLayout ... >
        ...
    </LinearLayout>
</ScrollView>
```



© Aptech Ltd. More UI Elements/Session 5 14

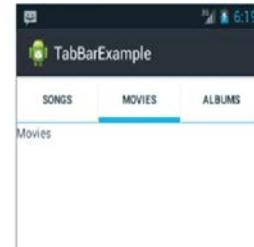
Using slide 14, explain the ScrollView. Explain that the ScrollView is a container which is visually similar to the ListView. In terms of functionality, the ScrollView is different from the ListView in that it does not treat its content as a list of elements. The ScrollView considers all of its content as a single entity. Explain the code to create a ScrollView given in the slide. Using the figure given on the slide, explain the output of a ScrollView.

TabBar

Slide 15

TabBar

- TabBar is used to display data in Tab format as is present in Android's contacts view
- It helps the developer to develop application for users where browsing for categorized data sets are required
- It can be scrollable, fixed, or stacked tabs
- TabHost is a container for a tabbed window view
- This object holds two children, a set of tab labels that the user clicks to select a specific tab and a FrameLayout object that displays the contents of that page



© Aptech Ltd. More UI Elements/Session 5 15

Using slide 15, explain the concept of TabBar. Explain that the TabBar is commonly used in applications where there is frequent navigation in between multiple activities. Explain that the TabHost is a container whereas, the TabBar labels the tabs. Explain that whenever a specific tab from the TabBar is selected, the appropriate tab is loaded.



In-Class Question: What is the difference between a ListView and ScrollView?

Answer: The ListView class identifies each element inside it individually. The ScrollView considers all of its content as a single entity.

TabBar Example

Slide 16

TabBar Example

Following Code Snippet demonstrates an example for creating a TabBar:

```
<TabHost
    android:layout_width="match_parent" android:layout_height="match_parent" >
    <LinearLayout>
        <TabWidget
            android:id="@+android:id/tabs"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" >

            ...
            </TabWidget>

            <FrameLayout >
                ...
                </FrameLayout>
        </LinearLayout>
    </TabHost>
```

© Aptech Ltd.

More UI Elements/Session 5

16

Using slide 16, explain code to add a TabHost to an activity. Explain the tags and explain that each tab is defined using a TabWidget.

WebView

Slide 17

WebView

- WebView is used to display Web pages as a part of the activity layout

- It does not include any feature of Web browser
- WebView loads Web pages and renders raw HTML data

Following Code Snippet demonstrates an example for creating a WebView:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/
    res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <WebView
        android:id="@+id/webview01"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1" >
    </WebView>
</LinearLayout>
```



© Aptech Ltd.

More UI Elements/Session 5

17

Using slide 17, explain the concept of a WebView. Explain that the WebView is used to

display Web pages as part of the activity. Explain the code to add a WebView.

ViewFlipper

Slide 18

ViewFlipper

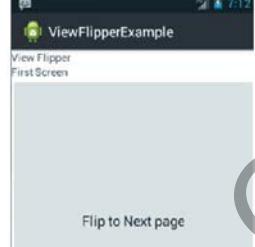
- The ViewFlipper can be used to animate between two or more added views
- Here, only one child-view can be visible at a time
- User can navigate between child views. In ViewFlipper, flip-in and flip-out animations are used while navigating between child views
- The developer can also set automatic flipping between each child views at regular interval

Following Code Snippet demonstrates an example for creating a ViewFlipper:

```

<LinearLayout ... >
<ViewFlipper android:id="@+id/viewflipper" android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <LinearLayout >
        ...
    </LinearLayout>
</ViewFlipper>
</LinearLayout>
```

© Aptech Ltd. More UI Elements/Session 5 18



Using slide 18, explain the concept of a ViewFlipper. Explain that the ViewFlipper is used to provide a 'flip' animation between two or more views. Explain the code to add a ViewFlipper as shown in the slide.

VideoView

Slide 19

VideoView

- VideoView is used to play a video file
- Android 4.2 can support large extent of video formats including .aac format
- Lower versions cannot play all the video files of different formats



© Aptech Ltd. More UI Elements/Session 5 19

Using slide 19, explain the concept of a VideoView. Explain that the VideoView is used to display video formats as a part of the activity. Explain that the earlier versions of Android do not support all the formats for video playback.

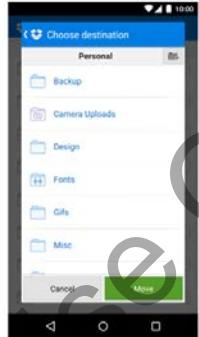
Dialogs

Slide 20

Dialogs

- ◆ A dialog is displayed in front of an activity as a small window
- ◆ When dialogs are active, the activity loses the focus
- ◆ A dialog can be created by extending the Dialog class
- ◆ It should be done by using any one of the following subclasses:
 - ◆ DatePicker Dialog
 - ◆ TimePicker Dialog
 - ◆ Progress Dialog
 - ◆ Alert Dialog
 - ◆ Toasts
 - ◆ Custom Dialog

© Aptech Ltd. More UI Elements/Session 5 20



Using slide 20, explain dialogs. Explain that dialogs are small windows which do not occupy the entire screen. Explain that dialogs do not cause a change in activity. Mention that custom dialogs can be created by extending the dialog classes. List the available subclasses of dialogs provided by Android and provide their purpose. For example, the DatePicker dialog is used to allow the user to select date and the time picker is used to select time, and so on.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding dialogs in Android.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/ui/dialogs.html>

<http://developer.android.com/reference/android/app/Dialog.html>

DatePicker/TimePicker Dialog Example

Slide 21

DatePicker/TimePicker Dialog Example

- Following Code Snippet demonstrates an example for creating a DatePicker/TimePicker dialog:

```
public class TimePickerFragment extends DialogFragment implements
    TimePickerDialog.OnTimeSetListener{
    private TimePickedListener mListener;
    @Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        ...
    }
    @Override
    public void onAttach(Activity activity) {
        ...
    }
    public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
        ...
    }
    public static interface TimePickedListener{
        public void onTimePicked(Calendar time);
    }
}
```

© Aptech Ltd.

More UI Elements/Session 5

21

Using slide 21, demonstrate the process of displaying a DatePicker/TimePicker dialog and retrieving the entered value. Explain the code given in the slide. Explain the methods that need to be implemented for the listener and the purpose for each of them.



In-Class Question: What is a dialog?

Answer: A dialog is a view used to display information without changing the activity.

DatePicker/TimePicker Listener Example

Slide 22

DatePicker/TimePicker Listener Example

- Following Code Snippet demonstrates an example for creating a DatePicker/TimePicker listener:

```
public class Timepicker extends FragmentActivity implements TimePickedListener
{
    @Override
    public void onCreate(Bundle savedInstanceState) {
        ...
    }

    mPickTimeButton.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {
            DialogFragment newFragment = new TimePickerFragment();
            new Fragment.show(getSupportFragmentManager(), "timePicker");
        }
    });

    public void onTimePicked(Calendar time) {
        ...
    }
}
```

© Aptech Ltd.

More UI Elements/Session 5

22

Using slide 22, demonstrate the steps to create a TimePicker listener. Explain the code given

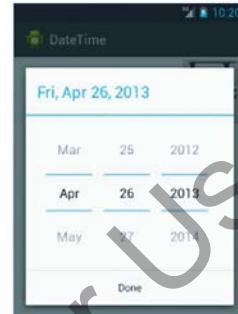
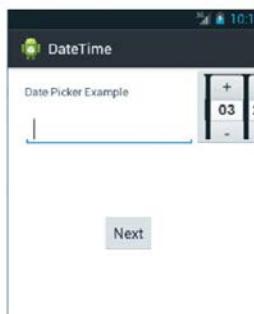
in the slide. Explain that the date picker fragment is displayed to the user once the button is clicked.

DatePicker/TimePicker Application

Slides 23 to 25

DatePicker/TimePicker Application 1-3

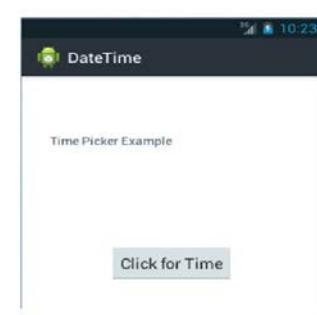
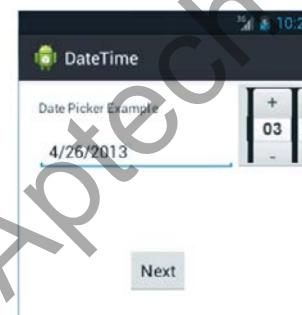
- Using the code, an application for demonstrating DatePicker/TimePicker Dialog is created as shown in the following figure:
- Once the date icon is clicked, the output will be as shown in the following figure:



© Aptech Ltd. More UI Elements/Session 5 23

DatePicker/TimePicker Application 2-3

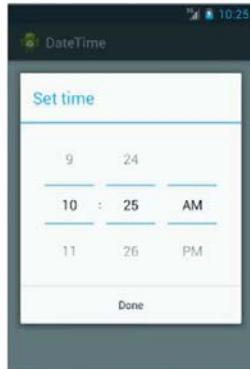
- Click Done and the output will be as shown in the following figure:
- Click Next and the output will be as shown in the following figure:



© Aptech Ltd. More UI Elements/Session 5 24

DatePicker/TimePicker Application 3-3

- Click the button 'Click for Time' and the output will be as shown in the following figure:



© Aptech Ltd.

More UI Elements/Session 5

25

Using slides 23 to 25, demonstrate the steps to create an application which displays a date and time picker dialog to the user. The application accepts date and time input and displays the selected value back to the user. Explain the functional logic of the application using the code explained earlier. Explain the output and functionality of the application using the figures given on the slides.

ProgressDialog Example

Slides 26 and 27

ProgressDialog Example 1-2

- Following Code Snippet demonstrates an example for creating a ProgressDialog:

```
final ProgressDialog myPd_ring = ProgressDialog.show(MainActivity.this,
    "Please wait", "Loading please wait...", true);
myPd_ring.setCancelable(true);
...
myPd_bar = new ProgressDialog(MainActivity.this);
myPd_bar.setMessage("Loading....");
myPd_bar.setTitle("Please Wait..");
myPd_bar.setProgressStyle(myPd_bar.STYLE_HORIZONTAL);
myPd_bar.setProgress(0);
myPd_bar.setMax(30);
myPd_bar.show();
...
myPd_bar.incrementProgressBy(5);
```

© Aptech Ltd.

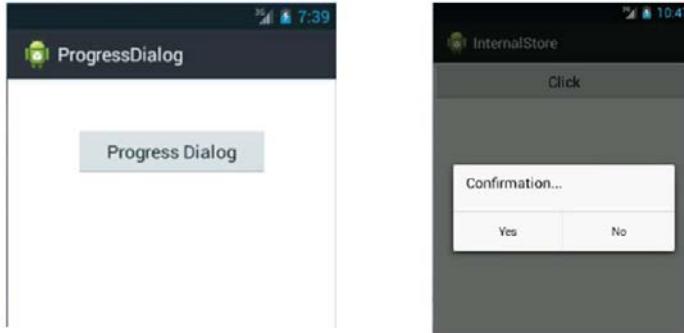
More UI Elements/Session 5

26

Using slide 26, demonstrate the steps to display a ProgressDialog to the user. Explain the code given in the slide. Explain that a ProgressDialog object is created and displayed to the user. Explain that the value is incremented by 5 later on. Explain the purpose of the methods of the ProgressDialog such as the setTitle(), which is used to set the title of the dialog.

ProgressDialog Example 2-2

- Using the explained code, an application for demonstrating ProgressDialog is created as shown in the following figure:
- Clicking the button will display the output as shown in the following figure:



© Aptech Ltd.

More UI Elements/Session 5

27

Using slide 27, demonstrate the steps to create an application which displays a progress dialog. The application displays a progress dialog and later on updates the progress value. Explain the functional logic and the output of the application using the figures given on slide 27.

AlertDialog Example

Slides 28 and 29

AlertDialog Example 1-2

- Following Code Snippet demonstrates an example for creating a AlertDialog:

```
AlertDialog.Builder alertDialogBuilder = new
AlertDialog.Builder(context);
alertDialogBuilder.setTitle("Confirmation..");
alertDialogBuilder.setMessage("Stay in this activity!");
.setCancelable(false)
.setPositiveButton("Yes",
new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog,int id) {
dialog.cancel();
}
})
.setNegativeButton("No", new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog,int id) {
...
>MainActivity.this.finish();
}
});
AlertDialog alertDialog = alertDialogBuilder.create();
alertDialog.show();
```

© Aptech Ltd.

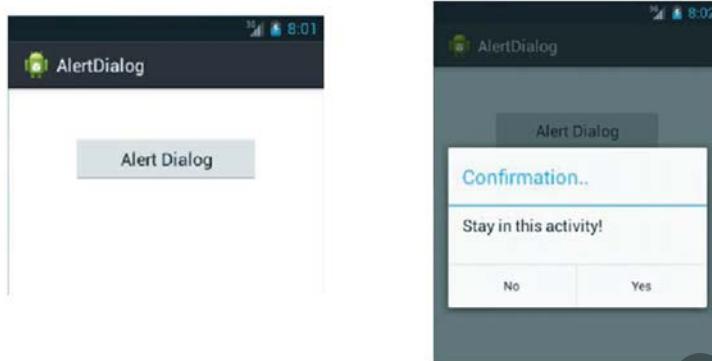
More UI Elements/Session 5

28

Using slide 28, demonstrate the process of displaying an AlertDialog. Explain the code given in the slide. Explain that the AlertDialog.Builder class is used to set the desired properties and retrieve a reference to an AlertDialog.

AlertDialog Example 2-2

- Using the explained code, an application for demonstrating AlertDialog is created as shown in the following figure:
- Clicking the button will display the output as shown in the following figure:



© Aptech Ltd.

More UI Elements/Session 5

29

Using slide 29, demonstrate the steps to create an application which displays an AlertDialog. If the user selects 'Yes', he/she is redirected to the same activity. If 'No' is selected, the application loads another activity. Explain the functional logic and the output of the application using the figures given on slide 29.

PopupDialog Example

Slides 30 and 31

PopupDialog Example 1-2

- Following Code Snippet demonstrates an example for creating a PopupDialog:

```
AlertDialog.Builder alert_dialog = new
AlertDialog.Builder(context);
alert_dialog.setTitle("Confirmation...");
alert_dialog.setNegativeButton("Cancel",
new DialogInterface.OnClickListener() {
@Override
public void onClick(DialogInterface dialog, int which) {
}
}),
alert_dialog.setPositiveButton("Leave current",
new DialogInterface.OnClickListener() {
@Override
public void onClick(DialogInterface dialog, int which) {
finish();
}
});
alert_dialog.show();
```

© Aptech Ltd.

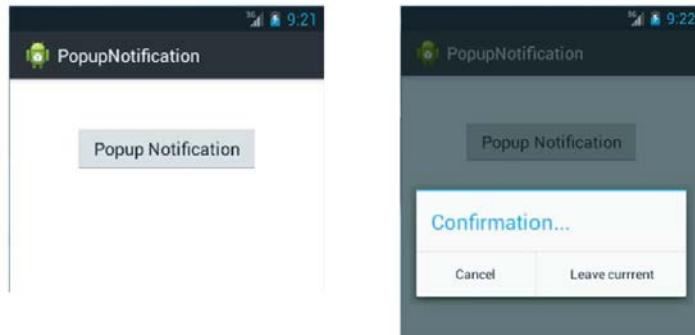
More UI Elements/Session 5

30

Using slide 30, demonstrate the steps to display a popup dialog to the user. Explain the code given in the slide. Explain that the dialog is displayed when the button is clicked using an onClick() listener.

PopupDialog Example 2-2

- Using the explained code, an application for demonstrating PopupDialog is created as shown in the following figure:
- Clicking the button will display the output as shown in the following figure:



© Aptech Ltd.

More UI Elements/Session 5

31

Using slide 31, demonstrate the steps to create an application which displays a popup dialog. The application displays a popup notification which confirms if the user wants to navigate away from the activity. Explain the functional logic and the output of the application using the figures given on slide 31.

Toast Example

Slide 32

Toast Example

- Following Code Snippet demonstrates an example for creating a Toast:

```
Toast.makeText(getApplicationContext(),
    "This is a Toast Notification", Toast.LENGTH_LONG).show();
```

- Using the code, an application for demonstrating Toast is created as shown in the following figure:

© Aptech Ltd.

More UI Elements/Session 5

32

Using slide 32, demonstrate the steps to display a toast. Explain the code given in the slide to display a simple toast. Explain the functionality and output of the application using the figure given on the slide.

CustomDialog Example

Slides 33 and 34

CustomDialog Example 1-2

- Following Code Snippet demonstrates an example for creating a CustomDialog with an EditText and an input check:

```
AlertDialog.Builder alert = new AlertDialog.Builder(MainActivity.this);
userData = new EditText(MainActivity.this);
alert.setTitle("Enter Data..");
alert.setView(userData);
alert.setPositiveButton("Enter Data", new OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        if (!userData.getText().toString().equals("")) {
            Toast.makeText(getApplicationContext(),"Data Entered is : " +
                userData.getText().toString(), Toast.LENGTH_LONG).show();
        } else {
            Toast.makeText(getApplicationContext(),
                "Field should not be empty", Toast.LENGTH_LONG).show();
        }
    }
});
alert.setNegativeButton("Cancel", new OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
    }
});
alert.show();
```

© Aptech Ltd. More UI Elements/Session 5 33

Using slide 33, demonstrate the steps to display a custom dialog box. Explain the code given in the slide. Explain that the AlertDialog.Builder is used to construct an appropriate dialog box.

CustomDialog Example 2-2

- Using the explained code, an application for demonstrating CustomDialog is created as shown in the following figure:
- When the user clicks Enter Data in the dialog, the output will be as shown in the following figure:

© Aptech Ltd. More UI Elements/Session 5 34

Using slide 34, demonstrate the steps to create an application which displays a custom dialog box. Explain the functional logic and the output of the application using the figures given on the slide.

Widgets

Slide 35

Widgets

- ◆ App widgets are miniature application views that can be embedded in other applications and receive periodic updates
- ◆ Application widgets are useful to provide control to the application without using the entire screen
- ◆ Android comes with several pre-installed widgets such as the Analog clock, Settings, and so on
- ◆ Widgets can be displayed on the home screen on any of the screens selected by the user
- ◆ The developer has no control over where the widget is displayed or the position of the widget
- ◆ Lock Screen widget support was introduced in API level 17 (Android 4.2) and dropped in API level 22 (Android 5.1)

© Aptech Ltd. More UI Elements/Session 5 35

Using slide 35, explain the concept of widgets. Explain that widgets in typical GUI programming are not the same as Android widgets. Explain that in Android, application widgets are miniature views which provide a subset of the functionality of the application without having to access the application. Provide a few examples of application widgets such as the clock, settings, and so on. Explain that widgets can also be customized and moved around home screens.

Widget Manifest and Metadata Example

Slide 36

Widget Manifest and Metadata Example

- ◆ Following Code Snippet demonstrates the manifest file changes that are required to be made for widgets:

```
<appwidget-provider xmlns:android="http://schemas.android.com/apk/res/android"
    ...
    android:widgetCategory="keyguard|home_screen">
</appwidget-provider>

<meta-data android:name="android.appwidget.provider"
    android:resource="@xml/quote_appwidget_info" />
```

- ◆ Following Code Snippet demonstrates the meta data file contents that are required to be made for widgets:

```
<appwidget-provider xmlns:android="http://schemas.android.com/apk/res/android"
    android:minWidth="40dp"
    android:minHeight="40dp"
    android:updatePeriodMillis="86400000"
    android:previewImage="@drawable/preview"
    android:initialLayout="@layout/quote_appwidget"
    android:resizeMode="horizontal|vertical"
    android:widgetCategory="home_screen|keyguard"
    android:initialKeyguardLayout="@layout/quote_keyguard">
</appwidget-provider>
```

© Aptech Ltd. More UI Elements/Session 5 36

Using slide 36, demonstrate the process to declare an application widget in the manifest file. Explain the code given in the slide. Demonstrate the important tags and attributes to declare an app widget.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding application widget API.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<https://developer.android.com/guide/topics/appwidgets/index.html>

<http://developer.android.com/reference/android/appwidget/AppWidgetProvider.html>

Determining Display Location/Creating Custom Widget

Slide 37

Determining Display Location/Creating Custom Widget

- Following Code Snippet demonstrates the process for detecting where the widget is being displayed:

```
Bundle myOptions = appWidgetManager.getAppWidgetOptions (appWidgetId);
int category = myOptions.getInt (AppWidgetManager.OPTION_APPWIDGET_HOST_CATEGORY, -1);
boolean isKeyguard = (category == AppWidgetProviderInfo.WIDGET_CATEGORY_KEYGUARD);
int widgetLayout = isKeyguard ? R.layout.quote_keyguard : R.layout.quote_appwidget;
```

- Following Code Snippet demonstrates the code for creating an Application Widget by extending the AppWidgetProvider class:

```
public class QuoteAppWidgetProvider extends AppWidgetProvider {
    public void onUpdate(Context context, AppWidgetManager appWidgetManager, int[] appWidgetIds) {
        ...
    }
    @Override
    public void onReceive(Context context, Intent intent) {
        super.onReceive(context, intent);
        ...
    }
}
```

© Aptech Ltd. More UI Elements/Session 5 37

Using slide 37, demonstrate the steps to display an application widget and handle its events. Explain the code given in the slide. Explain that it is possible to detect the position, space allocated, and the orientation of the widget to set the layout appropriately. Explain the methods that need to be overridden to handle events.



In-Class Question: Which XML tag is used to describe the widgets provided by the application?

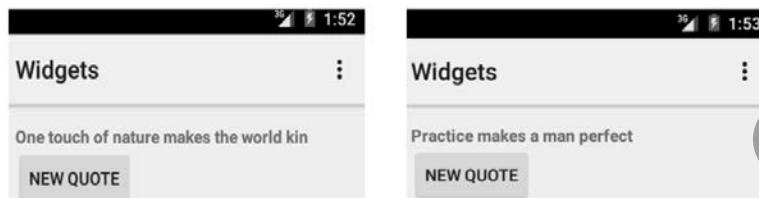
Answer: The <appwidget-provider> tag is used to describe the widgets provided by the application.

Application Widgets Example

Slides 38 to 41

Application Widgets Example 1-4

- Using the explained code, an application for demonstrating Widgets is created as shown in the following figure:
- Clicking on New Quote will show a new quote as shown in the following figure:



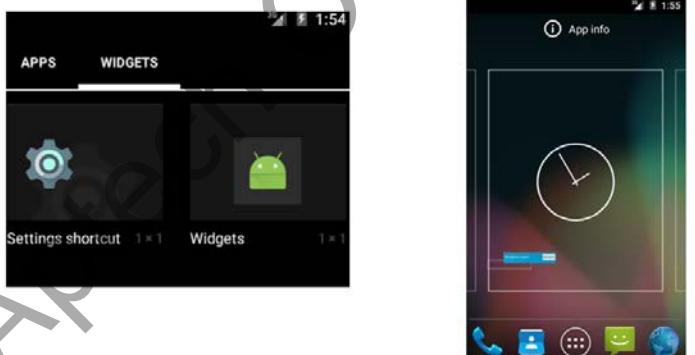
© Aptech Ltd.

More UI Elements/Session 5

38

Application Widgets Example 2-4

- The user can exit the application and return to the app drawer to add the new widget as shown in the following figure:
- The widget can be added to the home screen as shown in the following figure:



© Aptech Ltd.

More UI Elements/Session 5

39

Application Widgets Example 3-4

- The widget is displayed on the home screen as shown in the following figure:

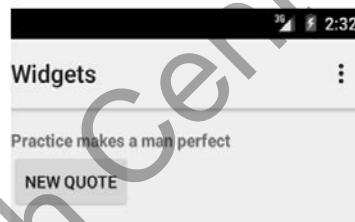


- By clicking the Refresh button a new quote is displayed as shown in the following figure:



Application Widgets Example 4-4

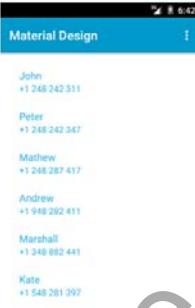
- By clicking the Quote itself, the application will be loaded as shown in the following figure:



Using slides 38 to 41, demonstrate the steps to create an application which comes with application widgets. Explain the functionality of the application using the figures given on the slides. Explain that the application widget can be added to the home screen and it displays a random quote and responds to user clicks. Explain that clicking the widget itself will load up the main activity of the application. Explain that it is also possible to create and install widgets alone without an accompanying application/activity.

Material Design Philosophy

Slide 42



Material Design Philosophy

- ◆ With the release of Android Lollipop, Google has introduced a new UI design philosophy called **Material Design**
- ◆ The material design specifies a set of design guidelines to maintain a look, consistent with the base operating system
- ◆ It is not mandatory to follow the Material Design guidelines
- ◆ Google strongly recommends doing so, at least at a base level

© Aptech Ltd.

More UI Elements/Session 5

42

Using slide 42, explain the material design philosophy. Explain that material design is an optional set of guidelines to customize the look and feel of the application. Explain that implementing material design will provide a consistent look and feel across the OS.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Material Design Philosophy.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

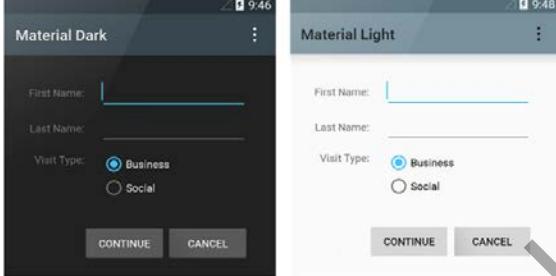
<https://developer.android.com/design/material/index.html>

<https://www.google.co.in/design/spec/material-design/introduction.html>

Material Theme**Slide 43**

Material Theme

- The simplest guideline towards a material look and feel is utilizing the system themes provided by Google
- API level 20 (Android 5.0) comes with Theme.Material (Dark Version) and Theme.Material.Light (Light Version) themes
- The developer can set the theme by navigating to res → values → styles.xml



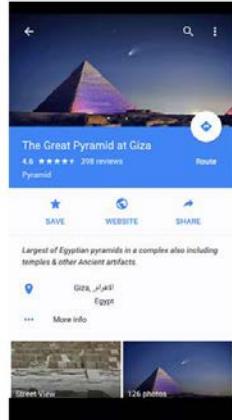
© Aptech Ltd. More UI Elements/Session 5 43

Using slide 43, explain the default material themes provided by Android. Explain that starting from Android 5.0, the two default material themes can be readily used within applications. Using the figures given on the slide, show the output of an application in material dark and material light themes.

Transparent Activity Bars and Full Screen**Slide 44**

Transparent Activity Bars and Full Screen

- Android Lollipop brings the ability to use transparent Activity bars and to use the entire screen without the notification and the navigation bar
- The material design philosophy recommends utilizing this feature when the activity content is static and prominently multimedia information



© Aptech Ltd. More UI Elements/Session 5 44

Using slide 44, explain the material design features introduced in Android. Explain that it is possible to use transparent activity bars starting from Android 5.0.



In-Class Question: What are the two base material themes supported by Android?

Answer: The two base material themes supported by Android are:

- Material dark
- Material light

Lists and Cards

Slide 45

Lists and Cards

- Google recommends the developers to use the RecyclerView instead of the standard ListView
- The RecyclerView comes with a more flexible layout and fully integrates with the Material Design philosophy
- The API to use a RecyclerView is similar to that of a ListView
- The official Gmail app utilizes the RecyclerView

Using slide 45, explain the Material design views. Explain that the RecyclerView acts as a replacement to ListView. Using the figures given on the slide, explain the output of using a RecyclerView. Provide an example of an application using RecyclerViews and cards such as Gmail and Google.

Uniform Layouts

Slide 46

Uniform Layouts

- Google recommends designing the application layout as multiples of a unit measurement
- Dividers of uniform size are recommended whenever a side panel or a navigational menu needs to be displayed
- This is contradictory to the philosophy of the metro theme of windows, where each 'tile' is deliberately made of different dimensions

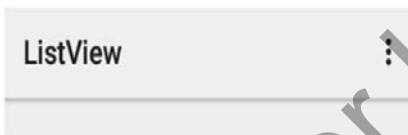
Using slide 46, explain the concept of Uniform layouts. Explain that the material design specification recommends partitioning the application layout in a uniform manner. If the students are familiar with the metro theme of windows, you can provide a comparison with it.

Shadows

Slide 47

Shadows

- ◆ Shadows and elevations are recommended to be used wherever applicable
- ◆ The android:elevation property can be used to set the elevation value of a view
- ◆ The property makes the view to appear elevated or hovering above the rest of the layout



© Aptech Ltd. More UI Elements/Session 5 47

Using slide 47, explain the use of shadows in material design. Explain that the android:elevation property can be used to set the depth and length of the shadow. Explain that this effect makes the view appear elevated or hovering over its background.

Animations

Slide 48

Animations

- ◆ The material design specification recommends using animations to transition between activities appear more natural
- ◆ A fade out animation can be used if a fresh unrelated activity is being loaded. Touch feedback is recommended on clickable objects
- ◆ Circular reveals are recommended to be used whenever the Activity being displayed is changing the screen orientation



© Aptech Ltd. More UI Elements/Session 5 48

Using slide 48, explain the use of animations in material design. Explain that transition animations are a core part of material design. Effects such as circular reveal and fade out are used to transition between animation and changes in orientations.

Color Schemes

Slide 49



Color Schemes

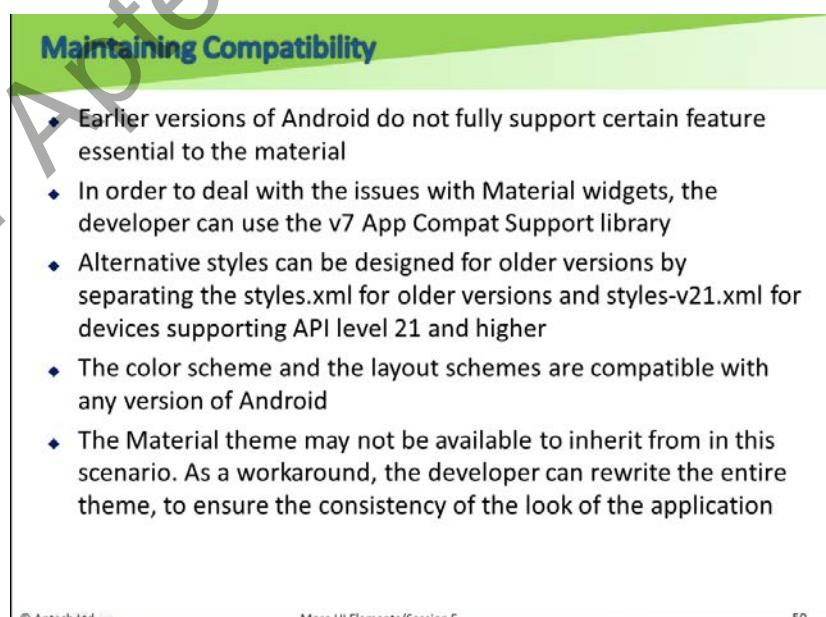
- Material design utilizes Bright color schemes throughout the application
- The developer is strongly recommended to style the app as per the brand requirement
- The ability to color the Action and the notification bar allows for full customization of look and feel
- The developer can use the holo car palette using '@android:color/holo_'
- The color palette can be used to determine the right theme to go along with the application interface

© Aptech Ltd. More UI Elements/Session 5 49

Using slide 49, explain the color schemes used for material design. Explain that utilizing the default color palette provided by Android ensures consistency of look with the rest of the OS. Explain that Android supports the holo color palette natively and that it can be directly used to style the applications.

Maintaining Compatibility

Slide 50



Maintaining Compatibility

- Earlier versions of Android do not fully support certain features essential to the material
- In order to deal with the issues with Material widgets, the developer can use the v7 App Compat Support library
- Alternative styles can be designed for older versions by separating the styles.xml for older versions and styles-v21.xml for devices supporting API level 21 and higher
- The color scheme and the layout schemes are compatible with any version of Android
- The Material theme may not be available to inherit from in this scenario. As a workaround, the developer can rewrite the entire theme, to ensure the consistency of the look of the application

© Aptech Ltd. More UI Elements/Session 5 50

Using slide 50, explain backward compatibility with Android devices running versions prior

to Lollipop.

Explain that the compatibility support library ensures backwards compatibility with older versions without the developer having to worry about it. Explain that the color scheme is supported across all versions of the platform and it should be ported. Explain that writing a fallback theme is recommended in case the material light and material dark themes are not supported.

Summary

Slide 51

Summary

- ◆ Adapter is an object that acts as a bridge between the UI Components such as List view, Grid view, and so on and the underlying data sources
- ◆ User Interface components act as an interface between the user and the application to input data and display the expected result
- ◆ A dialog is displayed in front of an activity as a small window and can be a DatePickerDialog, TimePickerDialog, ProgressDialog, or AlertDialogs
- ◆ Popups are smaller versions of dialogs which are designed in such a way so that users can make a selection to move forward or click outside the popup
- ◆ Toast is a popup that displays feedback for an activity
- ◆ Custom Widgets can be created by extending the AppWidgetProvider class. They can be placed on the home screen or the lock screen
- ◆ Lock screen widget support was dropped in API level 22
- ◆ Material Design is Google's new optional design philosophy for creating application UI

Using slide 51, summarize the session. Make them revise the following points:

- Adapter is an object that acts as a bridge between the UI Components such as List view, Grid view, and so on and the underlying data sources.
- User Interface components act as an interface between the user and the application to input data and display the expected result.
- A dialog is displayed in front of an activity as a small window and can be a DatePickerDialog, TimePickerDialog, ProgressDialog, or AlertDialogs.
- Popups are smaller versions of dialogs which are designed in such a way so that users can make a selection to move forward or click outside the popup.
- Toast is a popup that displays feedback for an activity.
- Custom Widgets can be created by extending the AppWidgetProvider class. They can be placed on the home screen or the lock screen.
- Lock screen widget support was dropped in API level 22.
- Material Design is Google's new optional design philosophy for creating application UI.

5.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes Media Handling in Android including Graphics, Animations, OpenGL, and Audio/Video capture and playback.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

For Aptech Center Use Only

Session 6: Media Handling

6.1 Pre-Class Activities

You should familiarize yourself with Media Handling techniques in Android. You should also review the Media Playback and recording APIs of Android. You should study the OpenGL framework basics of Android. You should also familiarize yourself with the newer Camera2 API and how it differs from the older API.

Familiarize yourself with the topics of the current session in-depth.

6.1.1 Objectives

After the session, learners will be able to:

- Explain the use of graphics in Android
- Explain animation
- Explain OpenGL and OpenGL rendering
- Explain playing of audio and video with Media Player
- Explain the process of capturing image and video using camera
- Explain the process of creating a live wallpaper

6.1.2 Teaching Skills

You should be familiar with Media Handling techniques in Android. You should be familiar with OpenGL and its usage in Android. You should also know the process for recording Audio/Video in Android using the respective API. In addition to this, you must have knowledge about the newer Camera2 API and how it differs from the older API. Go through the requirements and the process for creating live wallpaper.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

6.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Tell the students that the previous session explained in detail the components of Android User Interface (UI). It described the advanced (UI) components, Adapters, dialogs, widgets, and the Material Design philosophy.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the tools and APIs for Media Handling supported by Android and Media Capture.

6.2 In-Class Explanations

Introduction

Slide 3

Introduction

- ◆ Android supports multimedia capability
- ◆ Multimedia capabilities play a significant role for customers
- ◆ Android's Multimedia API provides the necessary functionality
- ◆ Camera API provides functionality to interact with the camera
- ◆ OpenGL support is available for making games and multimedia applications



© Aptech Ltd.

Media Handling/Session 6

3

Using Slide 3, introduce multimedia. Explain the term 'Multimedia' and the types of Multimedia content supported by Android. Explain the importance of Multimedia that is, it can be used to make feature rich applications such as animations and sound notifications. You can provide a few examples of well-established application to help the students understand better such as MX Player or VLC. Provide a brief description of the Media Framework which is the collection of APIs for accessing various multimedia capabilities of Android such as Audio/Video/OpenGL.

Graphics

Slide 4

Graphics

- ◆ Android provides basic level of graphics tools
- ◆ Basic Graphical tools are:
 - ◆ Canvases
 - ◆ Colors filters
 - ◆ Points
 - ◆ Rectangles
- ◆ The technique varies depending upon a 2D and 3D, static and dynamic, and so on



© Aptech Ltd.

Media Handling/Session 6

4

Using slide 4, explain Graphics and its use in application development. Begin by specifying

graphics are the important component of any GUI and that we will now study the Graphics implementation provided by Android. List the basic graphic tools provided by Android given in bulleted points. Provide a brief description and an example for each of them. For example, you can refer to a real life canvas and a color mixer to explain the concepts of Canvases and Color Filters. You can refer to the Graph system to provide a basic idea of how the co-ordinate system works in Android. Explain the differences between 2D and 3D.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Graphics and Graphics Framework.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/graphics/2d-graphics.html>

<http://developer.android.com/training/building-graphics.html>

<http://developer.android.com/reference/android/graphics/package-summary.html>

<http://developer.android.com/reference/android/graphics/drawable/package-summary.html>

Basic Graphics Concepts

Slide 5

Basic Graphics Concepts

- ◆ Views are known to be the visible elements in an Android UI
- ◆ Each View is associated with the Canvas
- ◆ When the View is displayed, its `onDraw()` method is automatically invoked by Android
- ◆ The developer can be creating their own View with the `onDraw()` method to display basic objects using the Canvas
- ◆ Canvas has methods for drawing Arcs, Circles, Lines, and so on
- ◆ Paint has the method for setting the alpha, color, and so on

Using slide 5, explain the components of the Graphics Framework. Define Views, Canvases, Arcs, Lines, and Circle. Explain the importance of the `onDraw()` method which is called whenever the View needs to be displayed or updated. Remind the concept of method overriding. Also, explain the co-ordinate system of Android and the concept of 'Alpha' in colors which represents the transparency component of the color.



In-Class Question: Which class is used to draw objects on?

Answer: The 'Canvas' class.

onDraw() and onMouseEvent() Methods

Slide 6

onDraw() and onMouseEvent() Methods

- Following Code Snippet demonstrates an example overriding the onDraw() method to create custom graphic output:

```
protected void onDraw(Canvas canvas) {
    Paint paint = new Paint();
    paint.setColor(Color.YELLOW);
    canvas.drawRect(40, 20, 90, 80, paint);
}
```

- Following Code Snippet demonstrates an example overriding the onTouchEvent() method to respond to user touch events:

```
public boolean onTouchEvent(MotionEvent event) {
    // if it's an up ("release") action
    if (event.getAction() == MotionEvent.ACTION_UP) {
        // Handle Click Event
        if (x >= 40 && x <= 90 && y >= 20 && y <= 80) {
            // redraw the View... this calls onDraw again!
            invalidate();
        }
    }
}
```

© Aptech Ltd. Media Handling/Session 6 6

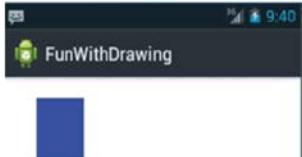
Using slide 6, explain the process of drawing objects and handling mouse (Touch) events for painted objects. Begin by explaining that we will now study the procedure to create and use the components explained so far. Explain the code given on the slide. The onDraw() method is implemented. A new 'Paint' 'Paint' object is created with 'Yellow' color. A rectangle is then drawn using the Paint object. The onTouchEvent() method is implemented to first recognize where the touch event has occurred. If it has occurred within the bounds of the drawn rectangle, then the rectangle is redrawn.

Graphics Example Application

Slide 7

Graphics Example Application

- Using the code, an application for demonstrating graphics is created as shown in the following figure:



Using slide 7 demonstrate the steps for creating an application using Graphics API. The application features a Canvas with a Blue Rectangle drawn on it. Provide a brief description of how the application functions using the code described. Explain that the rectangle in the

application activity is redrawn when touched.

Animation

Slide 8

Animation

- ◆ Android provides a set of API for applying animation to UI controls
- ◆ Android 3.0 introduces the Properties Animation API
- ◆ There are multiple types of animations:
 - ◆ Property Animation
 - ◆ View Animation
 - ◆ Drawable Animation
- ◆ The super class of the animation API is the Animator class
- ◆ The object of Animator class is used to modify the attributes of an object



© Aptech Ltd. Media Handling/Session 6 8

Using slide 8, explain Animations and their importance. Mention that Android provides a separate API for animation.

Give an example of usage of Animations in real life applications such as transitions in applications and games. Explain the types of Animations given in the slide. Using the bulleted points provide a brief description of the Animation API such as the Animator class.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Animations and the Transitions API.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/training/transitions/overview.html>

<http://developer.android.com/training/transitions/scenes.html>

<http://developer.android.com/training/transitions/transitions.html>

<http://developer.android.com/training/transitions/custom-transitions.html>

Base Animation State and Transition Animation State

Slide 9

Base Animation State and Transition Animation State

- Following Code Snippet demonstrates an example of creating the base state for the animation:

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/ android" >
<translate
    android:duration="500"
    android:fromXDelta="100%p"
    android:fromYDelta="0%p"
    android:toXDelta="0%p"
    android:toYDelta="0" />
</set>
```

- Following Code Snippet demonstrates an example of creating the Transition state for the animation:

```
<?xml version="1.0" encoding="utf-8"?>
<set>
<translate
    xmlns:android="http://schemas.android.com/apk/res/ android"
    android:duration="500"
    android:fromXDelta="0"
    android:interpolator="@android:anim/accelerate_interpolator"
    android:toXDelta="0" />
</set>
```

Using slide 9, explain the concept of Animation states and the difference between base and transition states. The base state is the initialization state of the Animation whereas the transition state represents the dynamic animation. Explain the code and the xml tags used in the code from the slide.

Starting the Animation

Slide 10

Starting the Animation

- Following Code Snippet demonstrates an example of starting the animation using the xml resources:

```
...
Intent intent = new Intent(getApplicationContext(), FirstActivity.class);
startActivity(intent);
overridePendingTransition(R.anim.anim_in, R.anim. hold);
...
```

- Using the code, an application for demonstrating animation is created as shown in the following figure:



- By clicking the Animation1 button, the title animation is displayed

Using slide 10, explain the application created for displaying a simple animation. The application features a Push button which will display the animation defined in XML when it is clicked. Explain the application logic and the output using the figure given on the slide.



In-Class Question: List the types of animations supported by Android?

Answer: The types of animations are:

- Property Animation
- View Animation
- Drawable Animation

OpenGL

Slide 11

OpenGL

- OpenGL is a cross platform 2D and 3D graphics rendering library
- Android supports the OpenGL for Embedded Systems (OpenGL ES)
- The OpenGL ES library is used to develop multimedia applications and games



© Aptech Ltd.

Media Handling/Session 6

11

Using slide 11, explain the API used to develop Mobile Games and multimedia applications that is, OpenGL. Begin by defining Library and explain the wide spread use of OpenGL beyond Android. You can mention the fact that OpenGL is used for PC and Console Game development as an example. Explain the concept of the Embedded Systems standard and provide a basic idea of how the ES version is different from the base version.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding OpenGL ES and Android's OpenGL API.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/graphics/opengl.html>

<http://developer.android.com/training/graphics/opengl/index.html>

<http://developer.android.com/training/graphics/opengl/environment.html>

<http://developer.android.com/reference/android/opengl/package-summary.html>

OpenGL History and Support

Slide 12

OpenGL History and Support

- Following table shows support for OpenGL ES version by different versions of Android:

OpenGL Version	Android Version
OpenGL ES 1.0	Android 1.0
OpenGL ES 2.0	Android 2.2
OpenGL ES 3.0	Android 4.3
OpenGL ES 3.1	Android 5.0

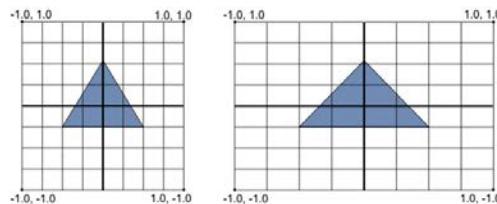
Using slide 12, list the OpenGL support history of Android. Mention the current version of OpenGL ES supported by the latest version of Android that is, OpenGL ES 3.1.

OpenGL Co-Ordinate System

Slide 13

OpenGL Co-Ordinate System

- OpenGL assumes the center to be at the center of the screen
- The co-ordinates are mentioned in fractions varying from 0 to 1
- The extreme end of the screen is specified by the value of 1f and the center by 0f
- A point half way in between is specified by 0.5f
- OpenGL does not account for varying screen sizes and aspect ratio
- On rectangular displays, the canvas is 'stretched', as shown in the following figure:



Using slide 13, explain the implementation of the OpenGL's co-ordinate system. Begin by explaining that, in order to draw anything on the screen, we need its location and the co-ordinate system is important for this.

You can use the basic co-ordinate system on a Graph Paper as an example to explain it better. Explain the use of the 'f' letter in the end. It signifies that the value is a float. Mention that OpenGL does not account for varying screen sizes and aspect ratios and how it affects objects drawn on the canvas. You can use the figure given on the slide to explain this

concept where the same equilateral triangle looks different in different screens or orientations.



In-Class Question: What is the OpenGL ES version supported by Android Lollipop?

Answer: OpenGL ES 3.1.

OpenGL Views

Slide 14

OpenGL Views

- ◆ Views allow the object to be 'perceived' in a different way
- ◆ **Projection View**
 - ◆ Allows the object to be 'projected' according to the size and aspect ratio of the screen
 - ◆ Responsible for transforming the co-ordinates from a square system to the rectangular varying screen sizes
 - ◆ Projection Matrix is used for this
- ◆ **Camera View**
 - ◆ It is where the object appears to be perceived from
 - ◆ In order to move the object on the screen, we can move the camera away from the object
 - ◆ Transformation are of two types Translation and Rotation

Using slide 14, explain the concept of Views. Begin by explaining that along with the location specifying 'where' it is displayed, we also need to specify 'how' the object is being displayed on the screen. Using bulleted points, explain the types of views and how they differ from each other. Explain the concept of Projection Matrix and its importance. Explain transformation and the difference between translation and rotation. Translation is a linear movement whereas rotation is an angular movement.

OpenGL Shaders, OpenGL Vertices, and Draw Order

Slide 15

OpenGL Shaders, OpenGL Vertices, and Draw Order

- ◆ A Shader is a user defined code that runs at some stage in the rendering process
- ◆ The Shader code is specified using a String literal
- ◆ The code to achieve this is shown in the following Code Snippet:

```
private final String vertexShaderCode =
    "uniform mat4 uMVPMatrix;" +
    "attribute vec4 vPosition;" +
    "void main() {" +
        "_gl_Position = uMVPMatrix * vPosition;" +
    "};"
```

- ◆ Colors in OpenGL are represented as a matrix of four units
Programmer is responsible for specifying the order in which the vertices are drawn
- ◆ A square can be specified as drawing two triangles
- ◆ A → B → C → A → C → D

Using slide 15, explain the concept of OpenGL Shaders. Begin by explaining that a shader defines 'what' is being displayed (vs the 'where' and 'how'). Define a shader and how it should be specified. Explain the given Code Snippet. The string vertexShaderCode holds vertex shader code for OpenGL. Explain the concept of vertices and how draw order is essential when drawing a figure.

OpenGL Colors

Slide 16

OpenGL Colors

- ◆ Colors in OpenGL are represented as a matrix of four units:
 - ◆ Red
 - ◆ Green
 - ◆ Blue
 - ◆ Alpha
- ◆ Alpha value specifies the transparency of the color
- ◆ All the values are specified in a range of 0f to 1f

Using slide 16, explain how colors are represented in OpenGL. Explain the concept of Alpha. Alpha represents the transparency value.

Provide a few example values and ask the students to try to convert them into OpenGL values.

Steps to Render an OpenGL Object

Slides 17 and 18

Steps to Render an OpenGL Object 1-2

- ◆ **Create a View:**
 - ❖ Extend the class `GLSurfaceView` from the package `android.opengl.GLSurfaceView`
 - ❖ View itself does not render the objects
 - ❖ Purpose of a view is only to display the rendered objects

- ◆ **Creating a Renderer:**
 - ❖ In order to implement a renderer, extend the class `GLSurfaceView.Renderer`
 - ❖ Following methods need to be implemented:
 - ❖ `onSurfaceCreated()`
 - ❖ `onSurfaceChanged()`
 - ❖ `onDrawFrame()`

Using slide 17, explain the steps to create a View and a Renderer. Explain the method to achieve this. Remind the concepts of method overriding and list the methods to be overridden.

Steps to Render an OpenGL Object 2-2

- ◆ **Creating Objects:**
 - ❖ Objects can be rendered directly on the `onDrawFrame()` method
 - ❖ However, this process is not recommended
 - ❖ Hence, we create OpenGL objects
 - ❖ A valid OpenGL object consists of:
 - ❖ Shader Code
 - ❖ Draw Order
 - ❖ Draw Method

• An application demonstrating OpenGL rendering is created as shown in the figure.



Using slide 18, explain the steps to render an object using OpenGL. Explain the requirements for an OpenGL objects that is, Shader Code, Draw Order, and Draw Method. Using the image, demonstrate the steps to display a yellow rectangle which is drawn on a red canvas using OpenGL API.

Audio/Video Playback

Slide 19

Audio/Video Playback

- ◆ Android uses OpenCore as its core component of media framework
- ◆ OpenCore supports different file formats such as MP3, AAC, MPEG-4, and so on
- ◆ Following classes are used to play audio and video in Android framework:
 - ◆ MediaPlayer
 - ◆ AudioManager



© Aptech Ltd.

Media Handling/Session 6

19

Using slide 19, explain the basics of the most widely used Audio/Video Playback. Mention that Android provides basic API for media playback and list the types of media supported by default by Android such as MP3, FLAC, and so on. Provide a brief description of the Audio Video API.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the MediaPlayer API.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/media/mediaplayer.html>

<http://developer.android.com/reference/android/media/MediaPlayer.html>

MediaPlayer Basics**Slides 20 and 21****MediaPlayer Basics 1-2**

- ◆ MediaPlayer class is the most important component of media framework
- ◆ Media can be played from:
 - ◆ Resource folder
 - ◆ File System path
 - ◆ URL
- ◆ States of MediaPlayer:
 - ◆ Creation and initialization
 - ◆ Preparation
 - ◆ Start of playback
 - ◆ Pause or Stop
 - ◆ Termination

© Aptech Ltd.

Media Handling/Session 6

20

Using slide 20, explain the basics of the Media Playback API which is used for Audio/Video Playback. Explain the importance of the MediaPlayer class and the various data sources for Media. List the states of the MediaPlayer object and provide a brief description for each of them.

MediaPlayer Basics 2-2

- ◆ Permissions may be required based on the source of the media
- ◆ Commonly used permissions are:
 - ◆ Internet Permission
 - ◆ Wake Lock Permission



© Aptech Ltd.

Media Handling/Session 6

21

Using slide 21, list the permissions required by the application to provide Media playback functionality. Explain why each of those permissions are essential. For example, a Wake Lock Permission is commonly used when Video is being played on screen to keep the screen awake.

MediaPlayer Initialization and Playback

Slide 22

MediaPlayer Initialization and Playback

- Following Code Snippet demonstrates an example for initializing the MediaPlayer Object:

```
...  
MediaPlayer = new MediaPlayer();  
MediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);  
MediaPlayer.setDataSource(url);  
MediaPlayer.setOnPreparedListener(this);  
MediaPlayer.setWakeMode(this, PowerManager.PARTIAL_WAKE_LOCK);  
MediaPlayer.prepareAsync();  
...
```

- Following Code Snippet demonstrates an example playing audio using a resource id:

```
...  
MediaPlayer = new MediaPlayer();  
MediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);  
MediaPlayer.setDataSource(fileDesc.getFileDescriptor(),  
fileDesc.getStartOffset(), fileDesc.getLength());  
fileDesc.close();  
MediaPlayer.prepare();  
MediaPlayer.start();  
...
```

© Aptech Ltd.

Media Handling/Session 6

22

Using slide 22, explain the process of MediaPlayer initialization and playback. Explain the code given on the slide. Explain the methods to initialize the object in the first Code Snippet and the methods that are used to play a file shown in the second Code Snippet.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the methods used in the Code Snippet.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/reference/android/media/MediaPlayer.html>

Audio Playback Example Application

Slide 23



- Using the code, an application for demonstrating Audio Playback is created as shown in the following figure:



Using slide 23, demonstrate the steps to create a basic Media Player Application. The application provides a basic interface with four buttons which control Audio Playback. Demonstrate and explain the functionality and output of the application. A MediaPlayer object is created and initialized. When the Start Player button is clicked, the media starts to play.



In-Class Question: Which class is used for starting Media Playback?

Answer: 'MediaPlayer' class.

Video Playback

Slide 24

Video Playback

- Following Code Snippet demonstrates an example playing video using a VideoView:

```
...  
VideoView myVideoView = (VideoView) findViewById(R. id.myvideoview);  
myVideoView.setVideoPath(SrcPath);  
myVideoView.setMediaController(new MediaController(this));  
myVideoView.requestFocus();  
myVideoView.start();  
...
```
- Using the code, an application for demonstrating Video Playback is created as shown in the following figure:

© Aptech Ltd.

Media Handling/Session 6

24

Using slide 24, demonstrate the steps to create a VideoView to display a video to the user. Explain the code given on the slide. Demonstrate and explain the functionality and output of the application using the figure given on the slide. A VideoView is created and a movie is loaded from external storage.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding VideoView and Video Playback API.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/reference/android/widget/VideoView.html>

MediaStore API

Slide 25

MediaStore API

- ◆ Android provides the MediaScanner Service which searches and maintains a record of all the media available on the device
- ◆ The scanned information is available through the MediaStore API
- ◆ MediaStore is a content provider and uses the same API



Using slide 25, explain the importance of the MediaStore Content Provider. Begin by explaining that so far we were able to play media whose location is known. However, the developer would like to discover media present on the device and for this purpose they can use MediaStore API. Define Content Provider. Explain the MediaScanner service and its importance.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the MediaStore API.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/reference/android/provider/MediaStore.html>

<http://developer.android.com/reference/android/provider/MediaStore.Audio.Media.html>

<http://developer.android.com/reference/android/media/MediaScannerConnection.html>

MediaStore Example Application

Slides 26 to 28

MediaStore Example Application 1-3

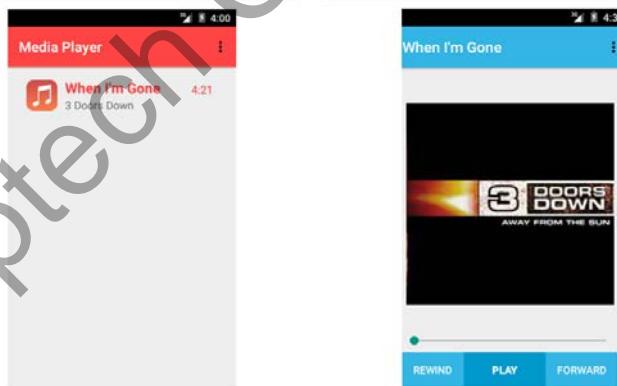
- Following Code Snippet demonstrates an example of using the MediaStore API to retrieve audio and video files:

```
...
Cursor videoCursor = getContentResolver().query(
    MediaStore.Video.Media.EXTERNAL_CONTENT_URI, projection, selection, null,
    null);
...
...
Cursor = getContentResolver().query(
    MediaStore.Audio.Media.EXTERNAL_CONTENT_URI, projection,
    selection, null, null);
```

Using slide 26, explain the process of retrieving Audio and Video content from MediaStore. Explain the code given in the slide. A cursor object is returned for each query containing the result set.

MediaStore Example Application 2-3

- Using the code, an application for demonstrating MediaStore API is created as shown in the following figure:
- By selecting a track, the audio player is displayed as shown in the following figure:



MediaStore Example Application 3-3

- The album art is extracted from the track and displayed to the user as shown in the following figure:



© Aptech Ltd.

Media Handling/Session 6

28

Using slides 27 and 28, demonstrate the steps to create an application that provides complete media player functionality. Using the Media Store API, the available audio files are queried and listed in the home screen. Once a song is selected, another activity is loaded which plays the selected song. Seek functionality is provided. The album art is extracted from the audio file and displayed at the center of the screen. Demonstrate and explain the functionality and output of the application using the figures given on slides 27 and 28.



In-Class Question: Which Content Provider holds the information from Media Scanner?

Answer: The 'MediaStore' content provider.

Camera

Slide 29

Camera

- Android framework provides support for various cameras and features
- The relevant classes for working with the Camera are:
 - Camera
 - SurfaceView
 - MediaRecorder



© Aptech Ltd.

Media Handling/Session 6

29

Using slide 29, explain the procedure for capturing media using the Camera API. List the essential classes and provide a brief description for each of them.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Camera API.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/training/camera/photobasics.html>

Camera Capture

Slide 30

Camera Capture

- Following Code Snippet demonstrates an example capturing the camera output and displaying it:
- Using the code, an application for demonstrating Camera functionality is created as shown in the following figure:

```
Intent captureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
//we will handle the returned data in
//onActivityResult
startActivityForResult(captureIntent,
CAMERA_CAPTURE);
```

```
...
Bitmap thePic =
extras.getParcelable("data");
//retrieve a reference to the ImageView
ImageView picView =
(ImageView) findViewById(R.id.picture);
//display the returned cropped image
picView.setImageBitmap(thePic);
...
```



© Aptech Ltd.

Media Handling / Session 6

30

Using slide 30, explain the process of capturing a still image using the camera API. Explain the code given in the slide. Demonstrate the steps to create an application that uses the Camera API to capture still images. Demonstrate and explain the functionality and output of the application. Explain the application logic and the code used to create the application.

Camera2 API

Slide 31

Camera2 API

- ◆ New API introduced in Lollipop
- ◆ New features include multiple cameras, Color Correction, Auto focus, and RAW Image capture
- ◆ Camera2 API uses a CameraManager to interact with the physical hardware
- ◆ The Camera objects are replaced with CameraDevice objects, and the functionality of the object is reduced to just representing the camera device



© Aptech Ltd.

Media Handling/Session 6

31

Using slide 31, explain the features of Camera2 API and the ideology behind replacing existing framework. Begin by explaining that the current Camera API is very limited and there are a lot of things it cannot do such as RAW image capture. Explain the changes made to the API and how it differs from the old API. Mention the fact that the old API will still function normally and that many Play Store applications still use the old API for compatibility reasons.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the Camera2 API and the ideology behind it.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<https://developer.android.com/reference/android/hardware/camera2/package-summary.html>

<http://spectrastudy.com/camera2-api-on-mwc-2015-devices/>

Capture Process**Slides 32 and 33****Capture Process 1-2**

- ◆ Get a reference to the camera manager instance using the `getSystemService(Context.CAMERA_SERVICE)` method
- ◆ Use the `getCameraIdList()` method to retrieve an id list of connected camera devices of the camera manager
- ◆ Retrieve the camera characteristics using `getCameraCharacteristics()` method of the camera manager
- ◆ Save the supported capture sizes and initialize the preview resolution
- ◆ Open the camera using the `openCamera()` method of the camera manager
- ◆ Get a reference to the `CameraDevice` object from the `onOpened()` callback once the camera is opened
- ◆ Create a new `MediaRecorder` object
- ◆ Initialize the media recorder object with the appropriate settings
- ◆ Retrieve the `MediaRecorder` surface using the `getSurface()` method

Capture Process 2-2

- ◆ (Optional) Get the surface of preview target so that the user can know what is being captured
- ◆ Create a capture session request by passing the list of surfaces containing the surfaces of the media recorder and the preview target. This is done using the `createCaptureSession()` method of the camera device. The camera data will be sent to these surfaces
- ◆ Start recording video when needed using the `MediaRecorder's start()` method
- ◆ The recording can be stopped using the `stop()` method
- ◆ (Optional) A background thread and handler can be used to not block the UI while waiting for IO and the camera to open, and so on

Using slides 32 and 33, explain the process of capturing a still image or a video using the Camera2 API. List the steps as shown in the slides. Stress on the importance of the `MediaRecorder` object which is the core component for capturing content from a Video Source.



In-Class Question: What class serves as a replacement to the `Camera` class in Camera2 API?

Answer: The '`CameraDevice`' class.

Camera2 API Example Application

Slides 34 to 36

Camera2 API Example Application 1-3

- Using the steps, an application for demonstrating Camera2API functionality is created as shown in the following figure:
- By clicking the capture button, the recording starts as shown in the following figure:

© Aptech Ltd. Media Handling/Session 6 34

Camera2 API Example Application 2-3

- To stop the recording, the Stop Capture button can be clicked
- The recorded file can be accessed from DDMS as shown in the following figure:

© Aptech Ltd. Media Handling/Session 6 35

Camera2 API Example Application 3-3

- The file can be pulled from the device to view on the PC as shown in the following figure:



© Aptech Ltd.

Media Handling/Session 6

36

Using slides 34 to 36, demonstrate the steps to create an application that accesses/utilizes the Camera2 API for media capture. The application is a Video Recorder which provides functionality similar to the stock Camera application, but utilizes the Camera 2 API instead.

Demonstrate and explain the functionality and output of the application using the figures given on slides 34 and 36. Explain the application logic and the code used to create the application.

Live Wallpapers

Slide 37

Live Wallpapers

- Live wallpapers are animated and sometimes interactive wallpapers
- Live wallpaper can be created using:
 - Drawing and Animations
 - OpenGL
 - Animated GIFs



© Aptech Ltd.

Media Handling/Session 6

37

Using slide 37, explain the concept of Live Wallpapers and how they differ from traditional wallpapers. Provide examples of pre-installed Live wallpapers such as the Nexus Rain, clock and so on. Explain the processes for creating Live Wallpapers and how they differ from each other.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information about the process of creating Live Wallpapers.

(OR)

You can refer to the online Mirror of the same documentation using the following link:
<http://developer.android.com/reference/android/service/wallpaper/WallpaperService.html>

Steps to Create a Live Wallpaper

Slides 38 and 39

Steps to Create a Live Wallpaper 1-2

- ◆ **Manifest and Permissions:**
 - ❖ The manifest file needs to contain a service extending the Wallpaper service with the permission android.permission.BIND_WALLPAPER
- ◆ **Settings Activity:**
 - ❖ A settings activity needs to be specified which saves the preferences and settings to configure the live wallpaper.onSurfaceChanged()
- ◆ **Metadata Description:**
 - ❖ An xml file which holds the details of the live wallpaper such as the name of the wallpaper, a thumbnail, description, and so on

Steps to Create a Live Wallpaper 2-2

- ◆ **Wallpaper Service:**
 - ❖ A class extending the WallpaperService class needs to exist in the project
 - ❖ The onCreateEngine() method needs to be overridden to return an engine object
- ◆ **Wallpaper Engine:**
 - ❖ The rendering of the wallpaper is handled by the Wallpaper engine
 - ❖ In order to create an engine, a new inner class extending the class WallpaperService.Engine needs to be created
 - ❖ Following methods need to be implemented:
 - ❖ onSurfaceCreated()
 - ❖ onSurfaceChanged()
 - ❖ onDrawFrame()

Using slides 38 and 39, explain the steps needed to create a Live wallpaper. List the methods that need to be overridden and the functionality of each of them. Emphasize on the Wallpaper Engine which is responsible for rendering the Live Wallpaper.

Live Wallpaper Example Application

Slides 40 to 43

Live Wallpaper Example Application 1-4

- Following Code Snippet demonstrates an example for creating a Live Wallpaper:

```
public class AnimatedWallpaperService extends WallpaperService {
    private class WallpaperEngine extends WallpaperService.Engine {
        ...
        public WallpaperEngine() {
            ...
        }
        ...
        private void draw() {
            ...
        }
        @Override
        public void onTouchEvent(MotionEvent event) { ... }
        ...
        public Engine onCreateEngine() { ... }
    }
}
```

© Aptech Ltd.

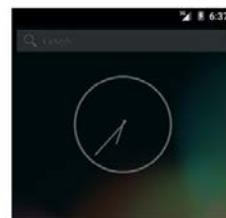
Media Handling/Session 6

40

Using slide 40, explain the process of using the `WallpaperService.Engine` to implement a Live Wallpaper. Explain the code given on the slide. Explain that the `draw()` and `onCreateEngine()` methods need to be implemented. Explain that the `onTouchEvent()` may be implemented if the Wallpaper intends to handle user input.

Live Wallpaper Example Application 2-4

- Using the code and steps, an application for demonstrating Live Wallpapers is created as shown in the following figure:
- In order to set the wallpaper, click and hold on the main screen as shown in the following figure:



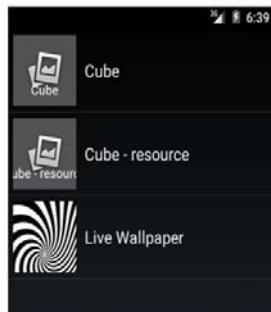
© Aptech Ltd.

Media Handling/Session 6

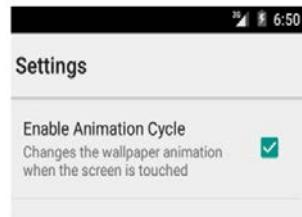
41

Live Wallpaper Example Application 3-4

- Select Live Wallpaper from the list as shown in the following figure:



- Click Settings to display the wallpaper setting as shown in the following figure:



Live Wallpaper Example Application 4-4

- If the Cycle setting is enabled, the animation changes whenever the user touches the screen as shown in the following figure:



Using slides 41 to 43, demonstrate the steps to create a Live wallpaper. The application utilizes a GIF for displaying the animation effects. Demonstrate and explain the functionality and output of the wallpaper using the figures given on the slides.

Summary**Slide 44****Summary**

- ◆ Android provides basic level of graphics tools such as canvases, colors filters, points, and rectangles that allows the developer to handle drawing on the screen directly
- ◆ Android framework provides two animation types namely property animation and view animation. Property Animation creates an animation by modifying an object's property values over a set period of time with an Animator. View Animation creates an animation by performing a series of transformations on the contents of a View object
- ◆ OpenGL is the 2D/3D rendering library supported by Android. It can be used to develop rich multimedia applications and games
- ◆ Android multimedia framework includes support for playing variety of common media types, so that one can easily integrate audio, video and images into the applications. The MediaStore API allows searching for media content
- ◆ Android framework supports capturing images and video through the Camera API or camera Intent
- ◆ Lollipop introduced the new Camera2 API which allows low level access to camera functionality with more features

Using slide 44, summarize the session. Make them revise the following points:

- Android provides basic level of graphics tools such as canvases, colors filters, points, and rectangles that allows the developer to handle drawing on the screen directly.
- Android framework provides two animation types namely property animation and view animation. Property Animation creates an animation by modifying an object's property values over a set period of time with an Animator. View Animation creates an animation by performing a series of transformations on the contents of a View object.
- OpenGL is the 2D/3D rendering library supported by Android. It can be used to develop rich multimedia applications and games.
- Android multimedia framework includes support for playing variety of common media types, so that one can easily integrate audio, video and images into the applications. The MediaStore API allows searching for media content.
- Android framework supports capturing images and video through the Camera API or camera Intent.
- Lollipop introduced the new Camera2 API which allows low level access to camera functionality with more features.

6.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes Data Handling in Android including SQLite Databases, Content Providers, Resources and Assets, and Internal and External Storage.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 7: Data Handling and Content Providers

7.1 Pre-Class Activities

You should familiarize yourself with concepts of data handling in Android. You should also acquire a good understanding of SQLite databases, Content Providers, Resources and Assets, and Internal and External Storage. You should review the procedure to create Content Providers and use existing ones.

Familiarize yourself with the topics of the current session in-depth.

7.1.1 Objectives

After the session, learners will be able to:

- Explain the process of saving and loading preferences
- Explain persistence of data to file
- Explain external storage
- Explain internal storage
- Explain SQLite Database
- Explain the process of storing and retrieving data from the database
- Explain Content Providers
- Make and use content providers
- Explain Resources and Assets

7.1.2 Teaching Skills

You should be familiar with previously covered topics including Shared Preferences and Internal and External Storage. You should know the process of creating databases and storing and retrieving data from it. In addition to this, you should be able to manage data from Content Providers and create new ones. You should also know how to work with resources and assets.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

7.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Revise the concepts of Media Handling techniques in Android including Graphics, OpenGL, Animations, Audio Video Capture and Playback, and Live Wallpaper creation. Emphasize on the important classes and the procedure to achieve the above tasks.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the concepts of data handling supported by Android, saving and retrieving data and the concept of Content Providers, and creating custom Content Providers.

7.2 In-Class Explanations

Introduction

Slide 3

Introduction

- ◆ Android provides different ways of saving persistent application data
- ◆ The appropriate option can be chosen based on the need of the developer
- ◆ Some of the data storage options are as follows:
 - ❖ Shared Preferences
 - ❖ Internal Storage
 - ❖ External Storage
 - ❖ SQLite Databases



© Aptech Ltd.

Data Handling in Android and Content Providers/Session 7

3

Using slide 3, introduce the term 'Persistent Data' and how it is important for applications. Persistent data is information that needs to be saved for long term and it will continue to exist even after restarting the application. Explain the types of storage options for persistent data.

Provide a brief description for each of them and explain when they will be used in comparison to other options.

Persisting Data to Shared Preference

Slide 4

Persisting Data to Shared Preference

- ◆ Shared Preferences are used to save simple application data
- ◆ Data can be saved and retrieved through the use of key/value pairs
- ◆ It enables the developer to save primitive data types such as boolean, float, long, string, and so on
- ◆ The SharedPreferences class is present in the android.content package



© Aptech Ltd. Data Handling in Android and Content Providers/Session 7 4

Using slide 4, explain the first and most basic storage option that is, Shared Preference. List the data types that can be stored using Shared Preferences. Provide a recap to previous concepts of shared preferences.

Creating and Retrieving Shared Preference Object

Slide 5

Creating and Retrieving SharedPreference Object

- ◆ Following Code Snippet demonstrates an example for creating a SharedPreference variable:

```
public class PreferencesActivity extends Activity {
    SharedPreferences prefs;
    String prefName = "MyPref";
    ...
}
```

- ◆ Following Code Snippet demonstrates an example for retrieving the SharedPreference object:

```
prefs = getSharedPreferences(prefName, MODE_PRIVATE);
```

© Aptech Ltd. Data Handling in Android and Content Providers/Session 7 5

Using slide 5, explain the code for creating and retrieving Shared Preference objects. Explain the use of the getSharedPreferences() method and the arguments it receives.

The first argument is the Preference name supplied as a string. The second argument is an integer which specifies the mode. It returns a SharedPreferences object.



In-Class Question: List the storage options provided by Android?

Answer: They are:

- Shared Preferences
- Internal Storage
- External Storage
- SQLite Databases

Saving and Committing Changes Using SharedPreference

Slide 6

Saving and Committing Changes Using SharedPreference

- ◆ Following Code Snippet demonstrates an example for retrieving the SharedPreferences Editor:

```
prefs = getSharedPreferences(prefName, MODE_PRIVATE);
SharedPreferences.Editor editor = prefs.edit();
```

- ◆ Following Code Snippet demonstrates an example for saving data using the SharedPreferences Editor:

```
...
//---save some values using the SharedPreferences object---
editor.putFloat("temperature", 85);
editor.putBoolean("authenticated", true);
editor.putString("username", "Wei-Meng Lee");
...
```

- ◆ Following Code Snippet demonstrates an example for committing changes using the SharedPreferences Editor:

```
...
//---saves the values---
editor.commit();
...
```

© Aptech Ltd.

Data Handling in Android and Content Providers/Session 7

6

Using slide 6, explain the Code Snippets to Store/Modify data stored in a Shared Preference. Explain the use of the SharedPreferences.Editor class and the putXXX() method. Explain the importance of commit() method. The commit() method is used to 'commit' or confirm the changes made to the data.

Retrieving Data Using SharedPreference Code Snippet

Slide 7

Retrieving Data Using SharedPreference Code Snippet

- Following Code Snippet demonstrates an example for retrieving data using the SharedPreferences Editor:

```
...
prefs = getSharedPreferences(prefName, MODE_PRIVATE);
float temperature = prefs.getFloat("temperature", 50);
boolean authenticated = prefs.getBoolean("authenticated", false);
String username = prefs.getString("username", "");
...
...
```

© Aptech Ltd.

Data Handling in Android and Content Providers/Session 7

7

Using slide 7, explain the code for retrieving data stored in shared preferences. Explain the use of the getXXX() methods to retrieve different types of values.

External Storage

Slide 8

External Storage

- External signifies that it is outside
- All Android-compatible devices support a shared 'external storage' which can be used to save files
- This can be a removable storage media (SD card) or USB storage
- Files which are saved to the external storage are readable by everyone and can be modified by the user



© Aptech Ltd.

Data Handling in Android and Content Providers/Session 7

8

Using slide 8, explain the widely used storage option that is, External Storage.

Make sure that students understand the point that 'Android always supports the feature of external storage irrespective of whether an external card slot is provided or not'. The system will still recognize external storage that is connected via USB using an extension cable or the USB OTG standard.

Advantages and Disadvantages of External Storage

Slide 9

Advantages and Disadvantages of External Storage

- ◆ **Advantages**
 - ❖ External storage will be extending the internal storage space of the Android device so that more apps can be stored
 - ❖ External Storage stores all the data in the SD card of a particular Android device
 - ❖ The file which is stored in the SD card can be retrieved and the data which is stored in that particular SD card can be used in some other device
- ◆ **Disadvantages**
 - ❖ If the developer has put a few set of apps on SD card of the Android device, then on removal of the SD card, the apps are no longer available for use until and unless the developer replaces back the SD card
 - ❖ It will take long time for your device to boot-up or shut down
 - ❖ The apps placed on the SD card require frequent update to support the functionality

© Aptech Ltd. Data Handling in Android and Content Providers/Session 7 9

Using slide 9, explain the advantages and disadvantages of utilizing external storage when compared to other storage options. List them using the bulleted points given in the slide. Provide a small description or example of data that will be stored on external storage such as game model/texture data and large media files.

Internal Storage

Slide 10

Internal Storage

- ◆ Files are saved directly on to the device's internal storage
- ◆ Files which are saved to internal storage are private to that particular application
- ◆ Once the user uninstalls the application, the files are removed
- ◆ Accessed using java.io classes



© Aptech Ltd. Data Handling in Android and Content Providers/Session 7 10

Using slide 10, explain the concept of internal storage and its applications. Begin by

explaining that even though external storage is very useful it is not entirely reliable. This brings us to the next storage option that is, internal storage. Provide a few classes from the standard java.io package and their uses. If the students have an understanding of standard of File I/O, you can relate to that.

Writing Data to Internal Storage

Slide 11

Writing Data to Internal Storage

- ◆ Invoke openFileOutput() method with the file name and the operating mode. This returns an object of FileOutputStream class
- ◆ Write to the file by invoking the write() method
- ◆ Close the stream invoking the close() method
- ◆ The code to achieve this is shown in the following Code Snippet:

```
...
String FILENAME = "android_file_save"; // saving into file
String string = "Hello Internal Storage";
FileOutputStream fos = context.openFileOutput(FILENAME, Context.MODE_PRIVATE);
// write I/O using write() and close()
fos.write(string.getBytes());
fos.close();
...
```

Using slide 11, explain the process of writing to internal storage.

With the help of the bulleted points, list the steps to write the data to internal storage. Explain the Code Snippet as given in the slide. Point out the main classes and methods being used that is, FileOutputStream class and the write and close methods.

Advantages and Disadvantages of Internal Storage

Slide 12

Advantages and Disadvantages of Internal Storage

◆ Advantages

- ◆ In this type of storage, the developer will be storing the apps or the data directly on to your device
- ◆ All the data and apps saved in the device will be present in the device only

◆ Disadvantages

- ◆ When the device is very low on internal storage space, then Android might delete the cache files to recover space
- ◆ Once the internal memory is full then the device will send an alert informing about memory full and incoming SMS will be rejected

Using slide 12, explain the advantages and disadvantages of utilizing internal storage when compared to other storage options. List them using the bulleted points. Provide an example of the data that will be stored on internal storage such as critical configuration files and in app media.



In-Class Question: Which storage option is used to store media files?

Answer: External Storage is used for storing media files.

SQLite Databases

Slide 13

SQLite Databases

- ◆ Android uses SQLite database to store relational data
- ◆ SQLite is a program provided by Android to execute services, such as creating tables and databases, amending rows, executing queries, and so on
- ◆ The SQLite database created for an Android application is private and cannot be accessed by other applications
- ◆ To create a database for your Android App, the developer needs to use the package android.database.sqlite



© Aptech Ltd. Data Handling in Android and Content Providers/Session 7 13

Using slide 13, explain the use of SQLite databases. Begin by explaining that so far, we studied the options to store small configuration data and unstructured data, now we will study the options to store structured data that is, databases. Explain about databases, if the students are not familiar with the concept of a database. Define library and explain how the SQLite implementation is being used as a library, how it differs from other database implementations such as MSSQL or MySQL. Mention that databases are private to the applications and the package holding the SQLite classes.



In-Class Question: Which kind of data can be stored in a database?

Answer: A database is used to store structured data.

Creating an SQLite Database

Slide 14

Creating an SQLite Database

- ◆ The developer needs to create a class which is a subclass of android.database.sqlite.SQLiteOpenHelper class
- ◆ This is because this class will enable the developer to create, open, close, and upgrade the database
- ◆ The developer needs to import android.database.sqliteSQLiteDatabase class
- ◆ Using this class instance and its properties the developer can execute the SQL statements to perform the CRUD operations such as create(insert), retrieve, update, and delete

Using slide 14, explain the process of creating a SQLite database. Begin by explaining that we will develop a database to perform the CRUD operations, as a first step we will create a class which is a subclass of android.database.sqlite.SQLiteOpenHelper class. List the steps to create a SQLite database using the bulleted points given in the slide.

Storing and Retrieving Data

Slide 15

Storing and Retrieving Data

- ◆ SQLite queries can be executed by using the query() method present in the SQLite database
- ◆ The query() method will return a Cursor object that points to all the retrieved rows
 - ◆ It helps by caching the result of query
 - ◆ It also provides functions with the help of which one can navigate to the desired row and obtain the data from the specified rows and columns

Using slide 15, explain the process of storing and retrieving data from SQLite database. List the steps using the bulleted points given in the slide.

Adding, Modifying, Searching, and Removing Databases

Slide 16

Adding, Modifying, Searching, and Removing Databases

- ◆ For Android, SQLite is 'baked into' the Android runtime
- ◆ Every Android database based application can create tables in SQLite databases
- ◆ Android provides classes for Database management in android.database and android.database.sqlite packages
- ◆ In this package the most important classes are as follows:
 - ◆ SQLiteOpenHelper
 - ◆ SQLiteDatabase
 - ◆ Cursor
 - ◆ SQLException

Using slide 16, explain the process of managing data in a SQLite database. List the steps using the bulleted points. Provide a brief description of the classes. Detailed explanations are not necessary at this stage as they will be covered later in the session.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the classes for SQLite API.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper.html>

<http://developer.android.com/reference/android/database/sqlite/SQLiteDatabase.html>

<http://developer.android.com/reference/android/database/Cursor.html>

<http://developer.android.com/reference/android/database/SQLException.html>

SQLiteOpenHelper

Slides 17 and 18

SQLiteOpenHelper 1-2

- ◆ The main functionality of the class is:
 - ❖ Open the database if it exists
 - ❖ Create it if it does not
 - ❖ Upgrade the version as required
- ◆ It provides a constructor to construct a helper class by subclassing this class and overriding the methods named `onCreate()` and `onUpgrade()`



© Aptech Ltd. Data Handling in Android and Content Providers/Session 7 17

Using slide 17, explain the functionality of SQLiteOpenHelper. With the help of bullet points explain its role in managing data in a SQLite database.

SQLiteOpenHelper 2-2

Syntax:

```
SQLiteOpenHelper (Context context, String name,
                SQLiteDatabase.CursorFactory factory, int version)
context – represent the context to create or open the database
name – represents the name of the database
factory – represents the factory class used for creating the cursor object
version – represent the number of the database
```

Methods

- **onCreate (SQLiteDatabase db)**
The method is invoked when the database is created for the first time. This is where the table is created and populated. The database name is passed as an argument
- **onUpgrade (SQLiteDatabase db, int oldVersion, int newVersion)**
The method is invoked when the database needs to be upgraded. The implementation should use this method to drop tables, add tables, or perform any other operations such as the need to upgrade to the new schema version

© Aptech Ltd. Data Handling in Android and Content Providers/Session 7 18

Using slide 18, explain the syntax for instantiating the SQLiteOpenHelper class. Explain the arguments for the constructor and the important methods using the accompanying description.

SQLiteDatabase

Slides 19 and 20

SQLiteDatabase 1-2

- The class has methods to create, delete, execute SQL commands, and other database management tasks

Methods

- **execSQL (String sql)**
 - Used to execute a single SQL statement that is not a SELECT statement or any other SQL statement that returns data and for creating tables. The sql statement to be executed is passed as an argument
- **long insert (String table, String nullColumnHack, ContentValues values)**
 - It is a convenience method used for inserting a row into the database
- **int update (String table, ContentValues values, String whereClause, String[] whereArgs)**
 - It is a convenience method used for updating rows in the database. The method returns the number of rows affected

SQLiteDatabase 2-2

Methods

- **int delete (String table, String whereClause, String[] whereArgs)**
 - It is a convenience method used for deleting rows in the database. The method returns the number of rows affected if a where clause is passed
- **Cursor query (String table, String[] columns, String selection, String[] selectionArgs, String groupBy, String having, String orderBy)**
 - The method queries the given table and returns a Cursor over the result set

Using slides 19 and 20, explain the syntax for SQLiteDatabase class. Describe the important methods using the accompanying description.

Cursor

Slides 21 to 23

Cursor 1-3

- This interface provides random read-write access methods to the result set returned by a database query
- Following table lists some of the methods present in Cursor class:

Methods	Description
<code>public int getColumnIndex(String colName)</code>	Returns the index of the indicated column as the result
<code>public String getColumnName(int colIndex)</code>	Returns the name of the specified column as the result
<code>public abstract String[] getColumnNames()</code>	Returns the column names as a string array
<code>public int getColumnIndexOrThrow (String colName)</code>	Returns the index of a column. The concept of throwing an exception arises when there is no existence of the column with the indicated name
<code>public abstract int getCount()</code>	Returns the count of number of rows
<code>public final int getPosition()</code>	Returns the present cursor position in the result set

Cursor 2-3

Methods	Description
<code>public final boolean moveToPosition (int pos)</code>	Moves the cursor towards the desired row in the result set and returns true, if the required record exists
<code>public final boolean moveToFirst()</code>	Moves the cursor to the first row in the queried result set and returns false, if there are no records in the result set
<code>public final boolean moveToNext()</code>	Moves the cursor to the subsequent rows in the result set and returns false, if the cursor is after the last record
<code>public final boolean moveToPrevious()</code>	Moves the cursor to the previous rows in the result set and returns false if the cursor is before the first record

Cursor 3-3

- ◆ The Cursor interface also has getXXX() style of methods to retrieve XXX type of values
- ◆ Some of the commonly used methods of this type are as follows:
 - ◆ `getString(int columnindex)`
 - ◆ `getLong(int columnindex)`
 - ◆ `getInt(int columnindex)`
 - ◆ `getFloat(int columnindex)`
 - ◆ `getDouble(int columnindex)`



© Aptech Ltd.

Data Handling in Android and Content Providers/Session 7

23

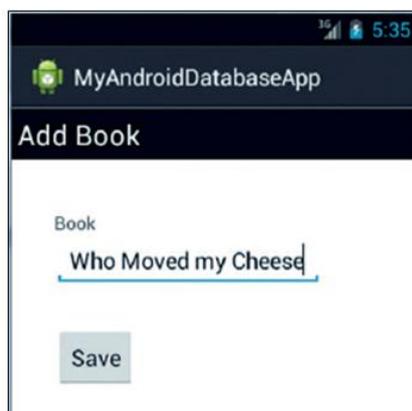
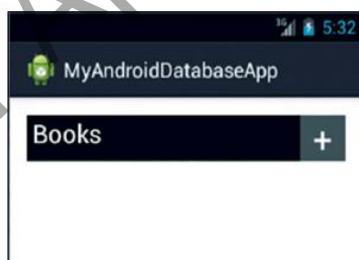
Using slides 21 to 23, explain the Cursor class API. Provide descriptions using the tables on slides 21 and 22. Slide 23 describes the getXXX() methods used to retrieve data from a Cursor. The Cursor class is extremely important and its application extends beyond the databases. Ensure the students are comfortable with the concepts of Cursor before proceeding.

SQLite Example Application

Slides 24 to 26

SQLite Example Application 1-3

- Using the methods, an application for demonstrating usage of SQLite Databases is created as shown in the following figure:
- Output when the plus symbol on the upper right corner is clicked as shown in the following figure:



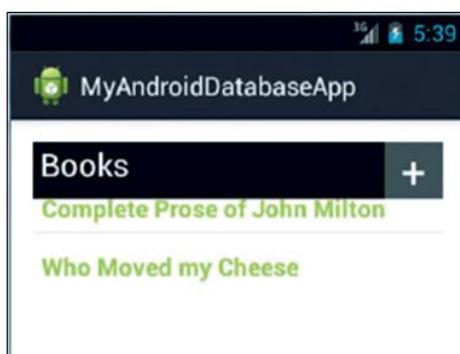
© Aptech Ltd.

Data Handling in Android and Content Providers/Session 7

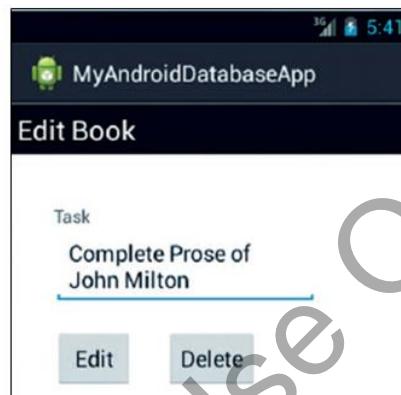
24

SQLite Example Application 2-3

- Output when the user clicks Save and the book name gets added to the Book database as shown in the following figure:

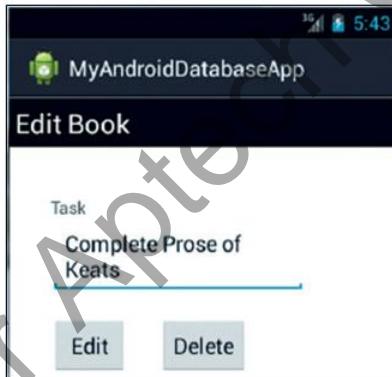


- Output when the book is selected to display the Edit Book pane as shown in the following figure:

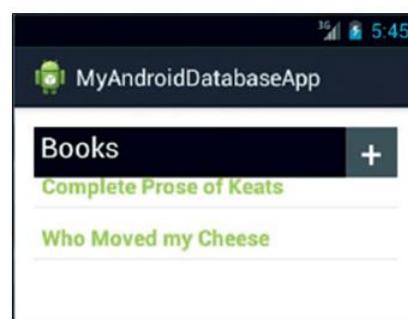


SQLite Example Application 3-3

- Output after the book name has been edited in the Edit Book pane as shown in the following figure:



- Output when the user has clicked Edit as shown in the following figure:



Using slides 24 to 26, demonstrate the steps to create an application which utilizes databases. The application stores the list of books on SQLite Database.

Explain the application logic and output of the application as shown in the figures given on the slides. Data about the books is being stored in databases and retrieved whenever requested.



In-Class Question: Which class is used to read the result from a database query?

Answer: The Cursor class is used to read the result from a database query.

Content Providers

Slide 27

Content Providers

- ◆ Content Providers provide access to data using a unified API
- ◆ Acts as a common gateway to applications for accessing the data
- ◆ Responsible for access management to the data
- ◆ Irrespective of the method the data is stored, data is almost always presented to the application in a table format
- ◆ A cursor is returned to navigate the data



© Aptech Ltd. Data Handling in Android and Content Providers/Session 7 27

Using slide 27, explain the concept of Content Providers. You may begin by explaining that data storage methods discussed so far are only limited to access within the application. Content Providers allow sharing data across applications using a common API. Explain the application of Content Providers and how they are different from other storage options. Mention that data stored in content providers can be accessed outside the application.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Content Providers.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/providers/content-provider-basics.html>

<http://developer.android.com/guide/topics/providers/content-providers.html>

Content URI

Slide 28

ContentURI

- ◆ Each content provider has a unique Content URI for the data associated with it
- ◆ This URI can be used by the applications to query data from the content provider
- ◆ An example content URI is shown in the following Code Snippet:

```
Content: //Domain/subdomain
```

Using slide 28, explain the importance of a Content URI and the way it works with content providers. A content URI acts as a primary key to uniquely identify the data and its source. You can relate to Web URLs as examples.

Searching and Accessing Content Providers

Slide 29

Searching and Accessing Content Providers

- ◆ The first argument to the content provider is a URI to the content
- ◆ The second argument is the projection specification. It contains the list of the columns to retrieve from the table format
- ◆ The third argument is the selection statement. It specifies the selection criteria of the data. It is similar to the 'WHERE' clause in SQL or the 'IF' condition in Java
- ◆ The fourth optional argument is the arguments for the selection statement. This is used if placeholders are used in the selection statement. It can be null otherwise
- ◆ The final optional argument is sort order specification. It decides how the retrieved content is sorted before being presented to the application

Using slide 29, explain the process of searching and accessing data from a content provider. List the steps using the bulleted points. Explain the arguments that are being supplied to specify the search criteria.

Accessing Content Providers Example

Slide 30

Accessing Content Providers Example

- ◆ Following Code Snippet demonstrates an example for Searching and Accessing data from the MediaStore Content Provider as an example:

```

String selection = MediaStore.Audio.Media.IS_MUSIC + " != 0";

String[] projection = {
    MediaStore.Audio.Media._ID,
    MediaStore.Audio.Media.ARTIST,
    MediaStore.Audio.Media.TITLE,
    MediaStore.Audio.Media.DATA,
    MediaStore.Audio.Media.DISPLAY_NAME,
    MediaStore.Audio.Media.DURATION,
    MediaStore.Audio.Media.ALBUM_ID,
};

Cursor videoCursor = getContentResolver().query(
    MediaStore.Video.Media.EXTERNAL_CONTENT_URI, projection, selection, null,
    null);

```

© Aptech Ltd. Data Handling in Android and Content Providers/Session 7 30

Using slide 30, explain the code to retrieve data from content providers. Explain the projection string array used to specify the information. Demonstrate the use of the selection string. Selection string is used to mention the selection criteria.

Adding, Modifying, and Removing Content

Slides 31 and 32

Adding, Modifying, and Removing Content

- ◆ **Inserting Content:**
 - ◆ The insert() method of the content resolver can be used
 - ◆ Data needs to be added as a key value pair
- ◆ **Updating Content:**
 - ◆ The Update() method of the content resolver can be used
 - ◆ For updating data a ContentValues object is created with the new values of the data
- ◆ **Removing Content:**
 - ◆ The delete() method of the content resolver can be used
 - ◆ The method accepts a URI along with the selection criteria of the entries to be deleted and the placeholder's values if required

© Aptech Ltd. Data Handling in Android and Content Providers/Session 7 31

Using slide 31, explain the steps to insert, update, and delete content from content providers.

Adding, Modifying, and Removing Content

- Following Code Snippet demonstrates an example for adding, modifying, and removing content:

```
//Insert  
ContentValues insertValues = new ContentValues();  
insertValues.put(CUSTOM.Domain.Column_name,"column value");  
...  
Uri contentURI = uriBuilder.build();  
getContentResolver().insert(uri,insertValues);  
...  
  
//Update  
...  
ContentValues updatedValues = new ContentValues();  
updatedValues.put(CUSTOM.Domain.Column_name,"column value");  
...  
  
String selectionCriteria = CUSTOM.Domain.ID + " = 3 AND "+CUSTOM.Domain.NAME+" = ?";  
String[] placeHolderValues = {"customer 3"};  
getContentResolver().update(uri,updatedValues,selectionCriteria,placeHolderValues);  
  
  
//Delete  
String selectionCriteria = CUSTOM.Domain.ID + " = 4 ";  
getContentResolver().delete(uri,selectionCriteria,null);  
// No placeholder values. Hence last argument is null
```

Using slide 32, explain the code to manage data from content providers. Explain the use of selection criteria and the methods used to insert, update, and delete data. Describe the arguments of all the methods.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the Content Provider data manipulation.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/reference/android/content/ContentValues.html>

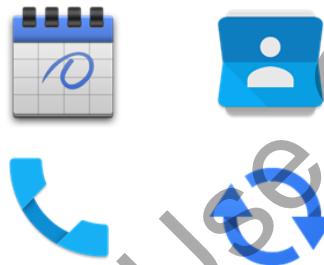
<http://developer.android.com/reference/android/content/ContentResolver.html>

Using Native Content Providers

Slide 33

Using Native Content Providers

- ◆ Android provides several content providers to access from and add content to
- ◆ The native content providers of Android can be utilized by importing the package android.provider
- ◆ The set of native content providers are as follows:
 - ◆ Calendar
 - ◆ Contact
 - ◆ Media Store
 - ◆ Openable Columns
 - ◆ Sync States
 - ◆ Telephony



Using slide 33, explain the various content providers existent in Android and their advantages. List the native content providers and the type of data they hold.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the native content providers.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/providers/calendar-provider.html>

<http://developer.android.com/guide/topics/providers/contacts-provider.html>

When to Create Content Providers?

Slide 34

When to Create Content Providers?

- ◆ A content provider can be created when:
 - ❖ You want to provide complex data or files to other applications
 - ❖ To provide search suggestions from the application to the search feature of Android

- ◆ A content provider should NOT be created when:
 - ❖ Providing access to databases
 - ❖ To act as a mode of communication between applications

© Aptech Ltd.

Data Handling in Android and Content Providers/Session 7

34

Using slide 34, explain the circumstances under which the content providers need be created and when they are not required to be created. Make sure the students understand the concept.

Considerations When Creating Content Providers

Slide 35

Considerations When Creating Content Providers

- ◆ Data Storage
 - ❖ Files
 - ❖ SQLite Databases

- ◆ Content URI
 - ❖ Authority
 - ❖ Path Structure
 - ❖ Content Id

- ◆ URI Patterns

- ◆ Manifest Changes

© Aptech Ltd.

Data Handling in Android and Content Providers/Session 7

35

Using slide 35, explain the various factors that must be considered to create a custom content provider. List and provide a brief description for each of them. For example, data storage decision impacts how the data provided by the content provider is stored internally.

Implementing Custom Content Providers

Slide 36

Implementing Custom Content Providers

- ◆ A concrete class extending the ContentProvider class needs to be created
- ◆ The following methods need to be overridden and implemented:
 - ◆ onCreate()
 - ◆ query()
 - ◆ update()
 - ◆ delete()
 - ◆ insert()
 - ◆ getType()

Using slide 36, explain the process of creating custom content providers and when it is necessary to create them. Explain the various methods as given in the slide, providing a brief description about their functionality. A reference to the ContentResolver instance is retrieved using the query() method.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the ContentProvider methods.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/reference/android/content/ContentProvider.html>

Content Provider MIME Types

Slide 37

Content Provider MIME Types

- ◆ MIME is the abbreviation for Multipurpose Internet Mail Extensions
- ◆ MIME was added as an extension to the e-mail protocol
- ◆ Android supports the ability to return MIME media types or MIME message content from content providers
- ◆ Allows for handling data which cannot be handled using the table format
- ◆ Android also provides the ability to define custom MIME types

Using slide 37, explain the term 'MIME' and provide a brief description of its history. Explain their use in Android and content providers. Explain MIME was added as an extension to the e-mail protocol to support transfer of custom types of data, such as images, video and application, and so on that were not initially supported. Describe more on MIME with the help of bulleted points providing brief descriptions when necessary.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding MIME.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<https://en.wikipedia.org/wiki/MIME>

Resources

Slide 38

Resources

- ◆ **Resource Storage:**
 - ❖ Resources are not stored on the Android File System
 - ❖ They are packaged as part of the application itself
 - ❖ They cannot be accessed with the standard File class
 - ❖ Any sensitive or copyrighted content is advised to be encrypted
- ◆ **Locating Resources:**
 - ❖ The standard format for a resource identifier is shown in the following Code Snippet:

```
[<package_name>.]R.<resource_type>.<resource_name>
```

© Aptech Ltd. Data Handling in Android and Content Providers/Session 7 38

Using slide 38, explain the concepts of resources. Begin by explaining about the storage options that are used to store static content. Explain that Resources and Assets fit this criterion. Using the bulleted points, explain how they are implemented in Android and where they are stored. Also, explain how resources are identified in Android. Using the Code Snippet, explain the default method of identifying resources. Mention that generally resources belong to the same package as the application code hence, the package name can be omitted.

Procedure for Locating Resources

Slide 39

Procedure for Locating Resources

- ◆ Android follows the following procedure to locate Resources:
 - ❖ Locate the package that is being referred. If it is not mentioned, the application name is used such as com.aptech.app_name to locate the apk holding the resource
 - ❖ Identify the resource type being referred to
 - ❖ List the directories capable of holding the resource type
 - ❖ Eliminate the types not supported by the device. For example, depending on the resolution, the drawable-mdpi could be eliminated
 - ❖ Repeat steps 2-4 until all the qualifiers are eliminated, such as language, key support, and so on until a maximum of one directory remains
 - ❖ Search the directory for a file name matching the resource name without the extension
 - ❖ Return an integer identifier to the resource

Using slide 39, explain how Android internally locates resources when a resource is referred. The bulleted points provide a brief description on this process.

Handling Runtime Configuration Changes

Slide 40

Handling Runtime Configuration Changes

- ◆ Certain device types have a changing configuration during run time such as media centre devices running Android
- ◆ The developer may have intended the application to use a different set of resources for this changed configuration
- ◆ Developer can either handle the changes manually in code or let Android auto configure the application automatically
- ◆ To manually handle Configuration changes is shown in the following Code Snippet:

```
@Override
public void onConfigurationChanged(Configuration newConfig) {
    super.onConfigurationChanged(newConfig);

    if (newConfig.orientation == Configuration.ORIENTATION_LANDSCAPE) {
        setContentView(R.layout.horizontal_layout);
    }
    else{
        setContentView(R.layout.vertical_layout);
    }
}
```

Using slide 40, provide some examples of dynamic configuration changes.

Begin by explaining that resources play a very important role in handling changes in configurations. Explain the methods to handle configuration changes. Explain the code given in the slide and specifically the setContentView() method.

Reading Raw Data from Resources

Slide 41

Reading Raw Data from Resources

- ◆ It is possible to read the raw data from a resource
- ◆ Assets are a better alternative for this purpose
- ◆ If resources are being used, then the developer is recommended to create a new directory called 'raw' in the /res directory
- ◆ The code to retrieve raw data from a resource with the file name picture.jpg stored in the raw directory is shown in the following Code Snippet:

```
InputStream is = getResources().openRawResource("picture");
```

Using slide 41, explain the process of reading raw data from resources such that the files were part of a File system. Emphasize that assets are a better alternative for this purpose and it should only be used when absolutely necessary. Specify the arguments for the openRawResource() method, that is, it accepts the file name.

Assets

Slides 42 and 43

Assets 1-2

- ◆ Assets are functionally similar to resources, but serve a different purpose
- ◆ Assets can hold any type of data
- ◆ Assets are not given a resource id
- ◆ They are directly identified by their file names
- ◆ Assets are stored in their own directory inside the package apk
- ◆ No changes are made and the files in the asset folder are stored as is
- ◆ The AssetManager class is used to access the assets
- ◆ The API is similar to the File class API

Assets 2-2

- ◆ Application of Assets in Realtime Applications:
 - ❖ Store textures
 - ❖ Store models
 - ❖ Store blob data
 - ❖ Other pre-compiled data in games
- ◆ If the size exceeds beyond 100 MB, it is preferable to use external storage to store these assets
- ◆ The code to open an asset called image.png is shown in the following Code Snippet:

```
InputStream is = context.getAssets().open("image.png");
```

Using slides 42 and 43, explain the importance of other storage options for static application content that is, assets and how they differ from resources. Explain the type of data that can be stored in assets and when it is preferable to use external storage instead. Also, explain the process to access data stored in Assets.



In-Class Question: Differentiate between Resources and Asset.

Answer:

Resources:

- Resources stores only known data types.
- Each resource gets a unique resource ID.
- Resources are not suitable for raw data access.

Assets:

- Assets can store any type of data.
- No ID is assigned to the data, which is being stored.
- Assets access using java.io File API.

Summary

Slide 44

Summary

- ◆ The different types of storages available in Android are Shared Preferences, Internal Storage, External Storage, and SQLite Database
- ◆ Android uses SQLite database to store relational data. SQLite queries can be executed by using the query() method present in the SQLiteDatabase. The query() method will return a Cursor object that points to all the retrieved rows
- ◆ Content Providers in Android act as resources to information
- ◆ Content Providers can be accessed using unique URLs
- ◆ Custom Content Providers can be created by extending the ContentProvider class
- ◆ Resources and assets are used to store static content for the application
- ◆ Assets do not have a resource id generated and they are packaged as-is

Using slide 44, summarize the session. Make them revise the following points:

- The different types of storages available in Android are Shared Preferences, Internal Storage, External Storage, and SQLite Database.
- Android uses SQLite database to store relational data. SQLite queries can be executed by using the query() method present in the SQLiteDatabase. The query() method will return a Cursor object that points to all the retrieved rows.
- Content Providers in Android act as resources to information.
- Content Providers can be accessed using unique URLs.
- Custom Content Providers can be created by extending the ContentProvider class.
- Resources and assets are used to store static content for the application.
- Assets do not have a resource ID generated and they are packaged as-is.

7.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes services in Android, creating and utilizing Broadcast Receivers and Intent Filters, and their applications.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 8: Services, Broadcast Receivers, and Intent Filters

8.1 Pre-Class Activities

You should familiarize yourself with the concept of services in Android. You should also acquire a good understanding of how to create services and explain the service lifecycle. You should review how to create and utilize Broadcast Receivers and understand how they work. You should also acquire the skill to explain Intents and Intent filters.

Familiarize yourself with the topics of the current session in-depth.

8.1.1 Objectives

After the session, learners will be able to:

- Explain services
- Explain service lifecycle
- Describe broadcast receiver and its working
- Explain filters
- Explain intent matching and its rules
- Explain filters in manifest file and broadcast receivers

8.1.2 Teaching Skills

You should be familiar with services in Android, how they can be created and their uses. You should also be able to identify and explain each and every phase of a Service lifecycle. You should be able to explain broadcast receivers and how they work. In addition to this, you should also be able to explain Intents and Intent filters, how they are created, and their uses.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

8.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Revise the concepts of data handling in Android including SQLite databases, Content Providers, Resources and Assets, and Internal and External Storage. Emphasize on the important classes and the API of the previous session.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the concepts of Service, Broadcast Receiver, and Intent Filters. Specify that they are very

important components of an Android application and each of them will be explained in detail in the session.

8.2 In-Class Explanations

Introduction

Slide 3

Introduction

- ◆ The main components that play an important role are:
 - ◆ Activities
 - ◆ Services
 - ◆ Content Providers
 - ◆ Broadcast receivers
- ◆ Services are executed in the background to perform long-running operations
- ◆ Broadcast receivers respond to announcements
- ◆ Intents are used for activating components



© Aptech Ltd. Services, Broadcast Receivers, and Intent Filters/Session 8 3

Using slide 3, introduce the topics that will be covered in this session. List the components and provide a brief description for each of them such as activities are the GUI components of the application. Use the bulleted points to emphasize the important aspects for all of them.

Services

Slide 4

Services

- ◆ Services can perform background tasks without providing a user interface
- ◆ Service class creates application components that are specifically suited to handle functions that should run at the background
- ◆ Even if the user switches to another application, a service will continue to run in the background
- ◆ Service has higher priority when compared with an inactive activity



© Aptech Ltd.

Services, Broadcast Receivers, and Intent Filters/Session 8

4

Using slide 4, explain the importance of services and their applications. Explain the uses of services in applications and provide an example of real time application which uses services (Google Play Store, Google Maps, and so on). Provide a brief idea on the steps to create a service. Mention that services continue to run in the background irrespective of activity status and that services have a higher priority for resources.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Services.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/guide/components/services.html>

Implementing Services

Slide 5

Implementing Services

- ◆ Services are declared in the application's manifest file
- ◆ To declare a service in the AndroidManifest.xml file, add a `<service>` element as a child of the `<application>` element as shown in the following Code Snippet:

```
<manifest ... >
...
<application ... >
<service android:name=".SampleService"/>
...
</application>
</manifest>
```

Using slide 5, explain the process to implement services. Specify that the application needs to declare its services in the Manifest file. Explain the importance of the service tag. It is used by the Android System to identify which applications have services and to identify the Service class.

Creating a Service

Slide 6

Creating a Service

- ◆ A component starts a service by invoking the startService() method
- ◆ It results in a call to the Service class onStartCommand() method
- ◆ To create a started service, following are the two classes from which the service class can extend from:
 - ◆ IntentService
 - ◆ Service
- ◆ IntentService class is implemented to avoid multi-threading
- ◆ It is started as a normal service, performs the tasks within a worker thread, and terminates when the task is performed

Using slide 6, explain the process to create a service that is declared in the Manifest file. List the steps. Specify the methods that need to be implemented namely, startService(), and onStartCommand().

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding creating Services.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<https://developer.android.com/training/run-background-service/create-service.html>

Extending IntentService

Slide 7

Extending IntentService

- Following Code Snippet demonstrates an example creating services by extending the IntentService class:

```
public class MyIntentService extends IntentService {
    public MyIntentService() {
        super("MyIntentService");
    }

    @Override
    protected void onHandleIntent(Intent Intent) {
        ...
    }

    @Override
    public void onCreate() {
        super.onCreate();
        ...
    }
}
```

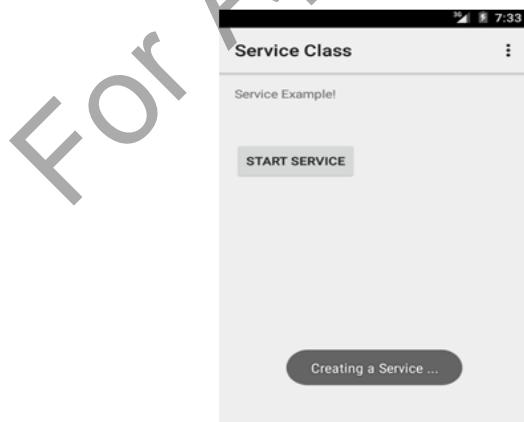
Using slide 7, explain the process of creating a service by extending the IntentService class. Explain the code given in the slide. The IntentService class is extended and the onHandleIntent() and onCreate() methods are implemented.

Services Example Application

Slide 8

Services Example Application

- Using the code, an application for demonstrating Services is created as shown in the following figure:



- The application logic is as follows:

- Developer creates a Service by extending the Intent Service class
- The Service is started when the START SERVICE button is clicked
- The Service creates a Toast once it is started
- The Service creates another toast when an Intent is handled

Using slide 8, demonstrate the steps for creating an application using services by extending IntentService. The application features a TextView showing the status of the service and a button to Start/Stop service. Explain the functional logic of the application and the output of the application as shown in the figure given on the slide.

Extending Service Class

Slide 9

Extending Service Class

- Following Code Snippet demonstrates an example for creating a Service by extending the Service class:

```
public class ServiceExample extends Service {  
  
    @Override  
    public IBinder onBind(Intent intent) { ... }  
    @Override  
    public void onCreate() {  
        ...  
    }  
    @Override  
    public void onDestroy() {  
        ...  
    }  
    @Override  
    public void onStart(Intent intent, int startId) {  
        ...  
    }  
}
```

© Aptech Ltd. Services, Broadcast Receivers, and Intent Filters/Session 8 9

Using slide 9, explain the process of creating a service by extending the Service class. Explain the code given on the slide. The Service class is extended and the onStart(), onCreate(), and onStart() methods are implemented.

Example Application Extending Service 1-2

Slides 10 and 11

Example Application Extending Service 1-2

- Using the code, an application for demonstrating Services using the Service class is created as shown in the following figure:
- By clicking the 'Start Music by Service', the music starts to play as shown in the following figure:

© Aptech Ltd. Services, Broadcast Receivers, and Intent Filters/Session 8 10

Example Application Extending Service 2-2

- The application logic is as follows:
 - Developer creates a Service by extending the Service class
 - The Service is started when the Start Music by Service button is clicked
 - Once started, the Service creates a MediaPlayer object to play an audio file
 - The Service is stopped when the Stop Music by Service is clicked

© Aptech Ltd. Services, Broadcast Receivers, and Intent Filters/Session 8 11

Using slides 10 and 11, demonstrate the steps to create a service by extending the Service class. This application uses the service to play music in the background. Explain the application logic and the output as shown in the figures given on slide 10.



In-Class Question: What are the ways to create a Service in an Android application?

Answer: They can be done in two ways:

- Extending the IntentService class
- Extending the Service class

Bound Services

Slide 12

Bound Services

- ◆ When components of an application bind to a service by calling bindService(), it is called a bound service
- ◆ Binding to a service helps in creating long-standing connection
- ◆ A service can be bound to:
 - ◆ Activity
 - ◆ Services
 - ◆ Content Providers
- ◆ To use a bound service, the onBind() callback method must be implemented
- ◆ It returns an IBinder that specifies the interface for communication with the service



© Aptech Ltd. Services, Broadcast Receivers, and Intent Filters/Session 8 12

Using slide 12, explain the concept of bound services and how they differ from regular services. Explain the uses of creating a bound service along with an example if necessary. List the various components that a service can be bound to. Provide a brief description of the API used to implement a bound service that is, the Messenger class, IBinder class, and the onBind() method.

Bound Service Example

Slide 13

Bound Service Example

- Following Code Snippet demonstrates how to create a Bound Service:

```
public class MyBoundService extends Service {  
    ...  
    final Messenger mMessenger = new Messenger (new IncomingHandler());  
    ...  
    @Override  
    public IBinder onBind(Intent intent) {  
        return mMessenger.getBinder();  
    }  
    ...  
}
```

© Aptech Ltd.

Services, Broadcast Receivers, and Intent Filters/Session 8

13

Using slide 13, explain the process of creating a bound service by extending the Service class. Explain the code given on the slide. The service class is extended and the onBind() method is implemented in addition to the regular methods to create a service.



In-Class Question: Which method should be called to bind to a service?

Answer: The bindService() method is called to bind to a service.

Bound Services Example Application

Slides 14 and 15

Bound Services Example Application 1-2

- Using the explained code, an application for demonstrating Bound Services as shown in the following figure:
- Click START SERVICE, then click ADD BY 100 and finally, click BIND SERVICE to get the output as shown in the following figure:

© Aptech Ltd. Services, Broadcast Receivers, and Intent Filters/Session 8 14

Bound Services Example Application 2-2

- The STOP SERVICE button will stop the service and unbind the service. The output of stopping the service will be as shown in the following figure:

- The application logic is as follows:

- Developer creates a Service by extending the Service class
- The Service is started and stopped by clicking the Start and Stop buttons respectively
- Once the Bind Service button is clicked, the bindService() method is called
- When the UNBIND SERVICE button is clicked, the service is unbound
- Clicking on the ADD BY 100 button will send a message to the service which will increment the count by a value of 100

Using slides 14 and 15, demonstrate the steps to create an application utilizing a bound service. This application is a simple counter application. Explain the application logic and the output as shown in the figures given on the slides.

Broadcast Receivers and Intents

Slide 16

Broadcast Receivers and Intents

- ◆ Responsible for responding to system-wide broadcast announcements
- ◆ Intents are responsible for binding the different individual components to each other at runtime
- ◆ A broadcast receiver is an Android component which allows the application to register for system or application events
- ◆ Broadcast receivers do not display a user interface



© Aptech Ltd. Services, Broadcast Receivers, and Intent Filters/Session 8 16

Using slide 16, explain the concept of broadcast receivers. Provide the definition and explain their uses. Define Intents and how they are related to broadcast receivers. Mention that broadcast receivers are components of an application which have no co-relation with the UI.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding BroadcastReceiver.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/reference/android/content/BroadcastReceiver.html>

Implementing Broadcast Receivers

Slide 17

Implementing Broadcast Receivers

- ◆ The steps to create a Broadcast Receiver are:
 - ❖ You have to create a subclass of Android's Broadcast Receiver
 - ❖ You have to implement the onReceive() method
- ◆ Android calls the onReceive() method on all registered broadcast receivers, when a matching broadcast Intent is identified
- ◆ The onReceive() method accepts following two arguments:
 - ❖ Context
 - ❖ Intent
- ◆ Following Code Snippet demonstrates how to create a Broadcast Receiver:

```
public class MyBroadcast extends BroadcastReceiver{  
    @Override  
    public void onReceive(Context context, Intent Intent) {  
        "  
    }  
}
```

© Aptech Ltd.

Services, Broadcast Receivers, and Intent Filters/Session 8

17

Using slide 17, explain the process to create a broadcast receiver by extending the BroadcastReceiver class. Explain the methods that should be implemented and the arguments that they accept. Provide the description for each of those methods. List the steps.

Broadcast Receiver Example Application

Slides 18 and 19

Broadcast Receiver Example Application 1-2

- Using the code, an application for demonstrating Broadcast Receiver is as shown in the following figure:

© Aptech Ltd. Services, Broadcast Receivers, and Intent Filters/Session 8 18

Broadcast Receiver Example Application 2-2

- After three seconds the message will be displayed as shown in the following figure:

- The application logic is as follows:
 - Developer creates an Activity with an EditText and a Button
 - The user enters an interval to send the Broadcast Message
 - When the timeout occurs, a Broadcast Message is sent
 - A toast is displayed signifying the same

© Aptech Ltd. Services, Broadcast Receivers, and Intent Filters/Session 8 19

Using slides 18 and 19, demonstrate the steps to create an application that uses broadcast receivers. This application notifies itself using a broadcast receiver. Explain the application logic and the output as shown in the figures given on the slides.



In-Class Question: Which class should be extended in order to create a broadcast receiver and which method should be implemented?

Answer: The BroadcastReciever class should be extended to create a broadcast receiver. The onRecieve() method should be implemented by the broadcast receiver.

System Broadcast

Slides 20 and 21

System Broadcast 1-2

- In the API, there are many classes that have specific broadcast events
- Following table lists some of the available events:

Event	Usage
Intent.ACTION_BATTERY_LOW	The battery level has dropped below a threshold
Intent.ACTION_BATTERY_OKAY	The battery level has increased again
Intent.ACTION_BOOT_COMPLETED	Android is up and running
Intent.ACTION_DEVICE_STORAGE_LOW	Storage space on the device is becoming less
Intent.ACTION_DEVICE_STORAGE_OK	The storage situation has improved again
Intent.ACTION_HEADSET_PLUG	A headset was plugged in or a previously plugged headset was removed
Intent.ACTION_LOCALE_CHANGED	The language of the device has been changed by the user

System Broadcast 2-2

Event	Usage
Intent.ACTION_MY_PACKAGE_REPLACED	The application has been updated
Intent.ACTION_PACKAGE_ADDED	A new application has been installed
Intent.ACTION_POWER_CONNECTED	The device has been plugged in
Intent.ACTION_POWER_DISCONNECTED	The device has been disconnected again
KeyChain.ACTION_STORAGE_CHANGED	The key store has changed
BluetoothDevice.ACTION_ACL_CONNECTED	A Bluetooth ACL connection has been established
AudioManager.ACTION_AUDIO_BECOMING_NOISY	The internal audio speaker is about to be used instead of other output means (like a headset)

Using slides 20 and 21, explain the various broadcast events. Explain their usage using the description provided in the table given in the slides. Inform the students to memorize as many of them as possible.

Role of Filters

Slide 22

Role of Filters

- ◆ When Intent is sent to the Android system, it determines suitable applications for this Intent
- ◆ If several components have been registered for this type of Intent, Android offers the user the choice to open one of them
- ◆ This decision is based on IntentFilters
- ◆ An Intent filter is an instance of the IntentFilter class



Using slide 22, explain the role of Intent filters in applications. Provide an example in real time applications such as, multiple applications that support viewing PDF in devices. A choice is displayed in these situations. Provide a brief idea on the process of creating an Intent filter.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding IntentFilters.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/guide/components/intents-filters.html>

Intent Filter Example

Slide 23

Intent Filter Example

- Following Code Snippet demonstrates how to create an Intent Filter to trigger an activity:

```
<activity android:name=".BrowserActivity"
    android:label="@string/app_name">
    <Intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <category android:name="android.intent.category.DEFAULT" />
        <data android:scheme="http"/>
    </Intent-filter>
</activity>
```

© Aptech Ltd.

Services, Broadcast Receivers, and Intent Filters/Session 8

23

Using slide 23, explain the process of creating an Intent filter. Explain the code given on the slide. Mention that Intent filters should be declared in the application Manifest file. Explain the Intent-filter tag and the information that should be specified in order to create an Intent filter.



In-Class Question: What is the main purpose of an Intent filter?

Answer: Intent Filters are used to determine the type of application receiving the intent.

Intent Matching

Slide 24

Intent Matching

- ◆ Matching of intents with intent filters is performed to discover a target component to activate
- ◆ Also used to get information about the set of components available on the device
- ◆ Android system populates the application launcher, by finding all the activities with intent filters that specify the `android.Intent.action.MAIN` action and `android.Intent.category.LAUNCHER` category

© Aptech Ltd. Services, Broadcast Receivers, and Intent Filters/Session 8 24

Using slide 24, explain the concept of Intent matching and how Android decides a target to handle Intent. Explain the importance and how Intent matching is used to discover applications which have a main activity. Intent matching is used by Android in discovering application and wallpapers.

Intent Matching Rules

Slide 25

Intent Matching Rules

- ◆ To match an IntentFilter with an Intent, three conditions must be fulfilled - the action, the category, and the data (both the data type and data scheme+authority+path, if specified) must match
- ◆ The Data characteristic is divided into four attributes:
 - ◆ Types
 - ◆ Schemes
 - ◆ Authority
 - ◆ Path
- ◆ Data Scheme matches when any of the given values match the Intent data's scheme
- ◆ Data Authority matches the given values with Intent data's authority
- ◆ Data Path matches any given values with the Intent's data path and when both a scheme

Using slide 25, explain the process Android follows internally to find the Intent target. List the data characteristic attributes and provide a brief description for each of them. List the rules and provide a brief description if necessary.

Filters in Manifest

Slide 26

Filters in Manifest

- ◆ By using intent filters, components specify their capability that is, the kinds of Intents they can respond to
- ◆ Intent filters are specified in the manifest as <Intent-filter> elements
- ◆ A component can have any number of filters, every filter describing a diverse capability
- ◆ When an intent explicitly names a target component, it activates that component; the filter does not play any role
- ◆ When Intent does not specify a target by name, it activates a component only if it can pass through one of the component's filters
- ◆ Following Code Snippet demonstrates how to create a Intent Filter in the AndroidManifest File:

```
<intent-filter android:icon="resource path from drawables"  
    android:label="resource path of string" android:priority="integer">  
</intent-filter>
```

Using slide 26, explain the process of specification of intent filters in the Android manifest file. Explain the information that needs to be provided by an intent filter and the priority property of the intent filter. Explain the process that Android follows when a target name is specified. Explain the code given on the slide.

Filters in Broadcast Receivers

Slide 27

Filters in Broadcast Receivers

- ◆ When the user registers a BroadcastReceiver to be executed in the main activity thread, the receiver is called in the main application thread
- ◆ The system can broadcast Intents that are 'sticky'
- ◆ There might be times when multiple sticky Intents may match the filter
- ◆ Only one of these Intents can be returned directly by the function and that is randomly decided by the system
- ◆ When the user knows that the registered Intent is sticky, then null can be supplied for the receiver
- ◆ Following Code Snippet demonstrates how to create an Intent Filter in Broadcast Receivers:

```
...
<receiver android:name=".BCastReceiver">
<intent-filter>
<action android:name="android.intent.action.MAIN"/>
<category android:name="android.intent.category.LAUNCHER"/>
<action android:name="android.intent.action.ACTION_POWER_CONNECTED"/>
</intent-filter>
</receiver>
...
```

Using slide 27, explain how intent filter can be used in conjunction with broadcast receivers. Explain the concepts of 'Sticky' intents and how Android deals in case there are multiple sticky intents. Explain the code given on the slide. It demonstrates the use of intent filters in broadcast receivers.

Summary**Slide 28**

Summary

- ◆ A service is an application component that is able to carry out long-running operations in the background. It does not provide a user interface
- ◆ In order to interact with the components, a service can let other components bind to it and perform Inter Process Communication (IPC). A service runs in the main thread of the application that hosts it
- ◆ A Broadcast Receiver is a component that responds to system-wide broadcast announcements
- ◆ Activities, Services, and Broadcast Receivers have intent filters to inform the system which intents they can handle
- ◆ Intent filters can handle two types of intents: Implicit and Explicit
- ◆ Matching of intents with intent filters is done to discover a target component to be activated, and also to get information about the set of components available on the device
- ◆ To match an intent filter with an Intent, three conditions must be fulfilled: the action, category, and the data

© Aptech Ltd. Services, Broadcast Receivers, and Intent Filters/Session 8 28

Using slide 28, summarize the session. Make them revise the following points:

- A service is an application component that is able to carry out long-running operations in the background. It does not provide a user interface.
- In order to interact with the components, a service can let other components bind to it and perform Inter Process Communication (IPC). A service runs in the main thread of the application that hosts it.
- A Broadcast Receiver is a component that responds to system-wide broadcast announcements.
- Activities, Services, and Broadcast Receivers have intent filters to inform the system which intents they can handle.
- Intent filters can handle two types of intents: implicit and explicit.
- Matching of intents with intent filters is done to discover a target component to be activated, and also to get information about the set of components available on the device.
- To match an intent filter with an Intent, three conditions must be fulfilled: the action, category, and the data.

8.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes Google API. Study the process of using the Google Maps API and location based service. Revise the process of signing applications and generating application keys. Familiarize with the concepts of play services and using them to provide Google+ authentication for applications.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 9: Google API

9.1 Pre-Class Activities

You should familiarize yourself with Google API. You should review Play Services, API keys, and signing applications. You should acquire expertise in the procedure to create and retrieve an API key for Google applications.

Familiarize yourself with the topics of the current session in-depth.

9.1.1 Objectives

After the session, learners will be able to:

- Explain API and its uses
- Explain Google API
- Explain location based service

9.1.2 Teaching Skills

You should be familiar with the concept of API and its application and the Play Service client library with its uses. Knowledge about location based services and their use is mandatory.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

9.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Tell the students that the previous session explained the basics of Services in Android Application, Intent Filters, and Broadcast Receivers.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the use of Google API, providing Google login authentication in the application and location based services.

9.2 In-Class Explanations

Introduction

Slide 3

Introduction

- ◆ An API can be explained as follows:
 - ◆ API stands for Application Programming Interface
 - ◆ It is a specification used by software components to communicate with each other
 - ◆ It is a library containing specifications for objects, classes, variables, and so on
 - ◆ It describes the ways in which a particular task is performed
- ◆ Google API's basically consist of specialized Web services, programs, and specialized scripts
- ◆ Google APIs are used as an added resource in their applications

© Aptech Ltd.

Google API/Session 9

3

Using slide 3, introduce the students to the concept of Google API. Explain that Google provides several Web services such as Maps, News, and so on. Explain that all of these can be utilized by the applications by using the Google Play Services library.

Google API

Slide 4

Google API

- ◆ Google as an organization provides several services such as Google Drive, Google App Engine, Google Translate, Google Maps, and so on
- ◆ Developers can integrate these services provided by Google into their application free of cost
- ◆ The developer needs to keep in mind that by using Google API, the application will rely on a network connection in order to function properly



© Aptech Ltd.

Google API/Session 9

4

Using slide 4, explain the various services provided by Google API. Mention that all of them are provided free of cost, even though they come at the cost of requiring a constant network connection to function properly.

Additional Reference:

You may refer to the following link for more information regarding Google API:

https://en.wikipedia.org/wiki/Google_APIs

Working with Location Based Services

Slide 5

Working with Location Based Services

- ◆ Location Based Service (LBS) is an information service and has a number of uses in social networking
- ◆ With the incorporation of Global Positioning System (GPS) devices in Smartphones, LBS have become important in the past few years
- ◆ Most important classes and interface that are to be used are as follows:
 - ◆ LocationManager
 - ◆ LocationProvider
 - ◆ Location
 - ◆ LocationListener



© Aptech Ltd. Google API/Session 9 5

Using slide 5, explain the concept of Location Based Services (LBS) and its uses. Explain that LBS require GPS functionality to provide accurate results. Mention that whenever GPS location is not available, alternate methods can be used to derive the location such as network registrations, IP addresses, and Wi-Fi connections.

Retrieving Current Location Example

Slide 6

Retrieving Current Location Example

- ◆ The user analytics on the map are feasible in one of the following ways:
 - ◆ Using the GPS device that comes with the mobile
 - ◆ Using the ID of the cell that the user is currently being served by
- ◆ The current user location can be retrieved as shown in the following Code Snippet:

```

LocationManager locationManager =  

(LocationManager) getSystemService(Context.LOCATION_SERVICE);  
  

locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER,  

MINIMUM_TIME_BETWEEN_UPDATES, MINIMUM_DISTANCE_CHANGE_FOR_UPDATES, new  

MyLocationListener());  
  

Location location = locationManager.getLastKnownLocation(LocationManager.GPS_  

PROVIDER);
  
```

© Aptech Ltd. Google API/Session 9 6

Using slide 6, explain the process of retrieving the GPS location of the device.

Explain the code given on the slide. Explain that the LocationManager class is used to read the last known location of the device and register for updates in the location values.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the LocationManager API:

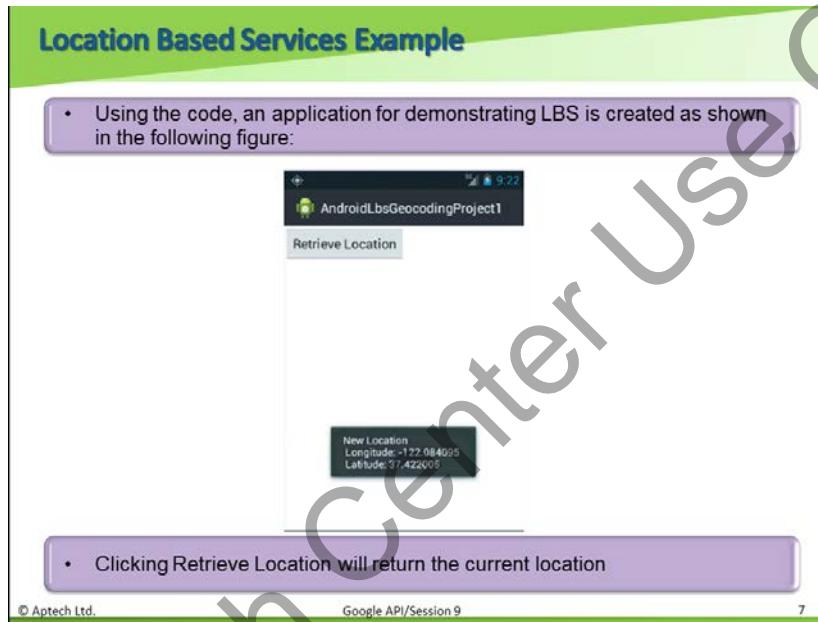
(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/reference/android/location/LocationManager.html>

Location Based Services Example

Slide 7



Using slide 7, demonstrate the steps to create an application that can display the GPS co-ordinates of the device. The application features a single Button which will retrieve the GPS co-ordinates of the device and display them as a Toast.



In-Class Question: Which class is used to register listener for GPS updates?

Answer: The LocationManager class is used for this purpose.

Setting Up Android SDK for Google API

Slide 8

Setting up Android SDK for Google API

- ◆ Navigate to Tools → Android and select SDK Manager
- ◆ Select Google APIs under the Android API being used. In this case it is Android 5.0.1
- ◆ Select Google Play services under Extras
- ◆ Click Install. The selected packages will be installed as shown in the given figure

The screenshot shows the Android SDK Manager interface. The 'Packages' tab is selected. Under the 'Extras' section, 'Google APIs' is checked. Other options like 'Android Support Repository' and 'Google Play services' are also checked. The 'Status' column indicates which packages are installed (e.g., 'Installed'), available for update (e.g., 'Update available rev. 5'), or have no updates (e.g., 'Not updated'). The bottom status bar shows 'Google API/Session 9'.

Using slide 8 explain the process of setting up support for Google API from the SDK Manager. Explain the steps given on the slide. Explain that the developer needs to manually install the Google Play Services packages to utilize Google API in the applications.

Working with Google Maps

Slide 9

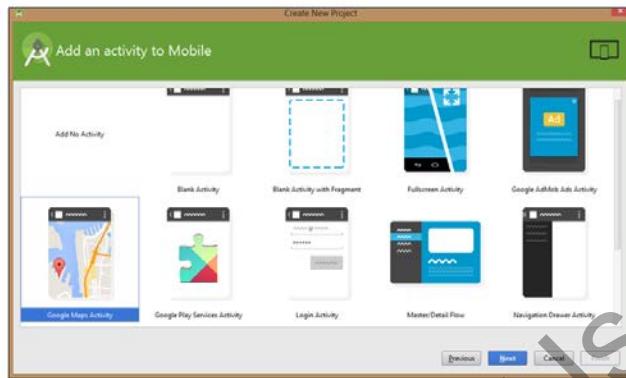
Working with Google Maps

- ◆ Google Maps allows the users to explore the world with rich maps provided by Google
- ◆ It helps to identify locations with custom markers, augment the map data with image overlays, embed one or more maps as fragments, and much more
- ◆ In order to use Google Maps the needs to host an Activity Fragment
- ◆ The Steps are as follows:
 - ◆ The Android SDK should have libraries for Google Maps installed
 - ◆ The Activity class that will be responsible for showing the map needs to extend from `ActivityFragment` class
 - ◆ The application manifest file needs to be setup with the '`android.permission.INTERNET`' permission
 - ◆ The application manifest file is also required to be setup with the Maps key

Using slide 9, explain the process of utilizing Google Maps in an application. Explain the benefits of implementing Google Maps support within the application. Explain the steps of achieving this using the bulleted points given on the slide. Explain that Internet permission is mandatory for utilizing Google Maps.

Adding Dependencies for the Application**Slides 10 to 12****Adding Dependencies for the Application 1-3**

- Create a project named GoogleProject in Android Studio
- During the activity selection screen, select Google Maps Activity as shown in the following figure:



© Aptech Ltd.

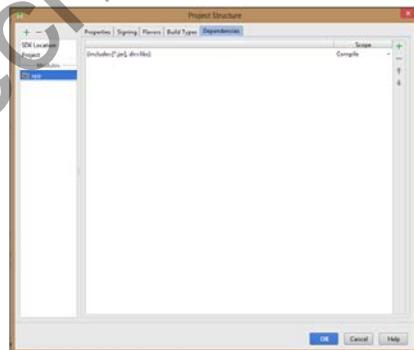
Google API/Session 9

10

Using slide 10, explain the first step to creating an application using the Google Maps API. Explain that selecting the Google Maps Activity will create different auto generated code with a Maps Fragment included.

Adding Dependencies for the Application 2-3

- If the project is created using a Blank Activity, right-click app and select Open Module Settings
- From the Project Structure dialog box, select the Dependencies tab as shown in the following figure:



© Aptech Ltd.

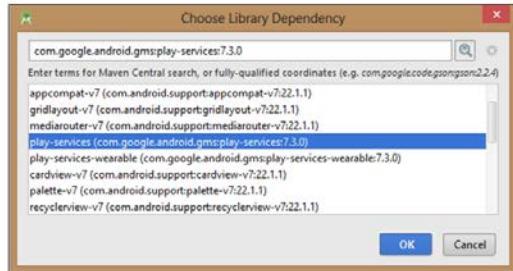
Google API/Session 9

11

Using slide 11, explain the process of adding dependencies if the Blank Activity was selected instead. Explain that the dependencies tab can be accessed from the Project Structure dialog box.

Adding Dependencies for the Application 3-3

- Click the '+' sign and select Library Dependency to open Library Selection dialog box as shown in the following figure:



- Select the play-services library and click OK
- Click OK again to confirm the dependencies
- Navigate to app → java → com.example.googleproject and add the import for the package com.google.android.gms.maps.GoogleMap

Using slide 12, explain the process of adding the Play Services dependencies to the application. Explain that Android Studio treats play-services as an external library and it should be added manually. List the steps and explain the process using the figure given on the slide.

Generating a Key for the Application

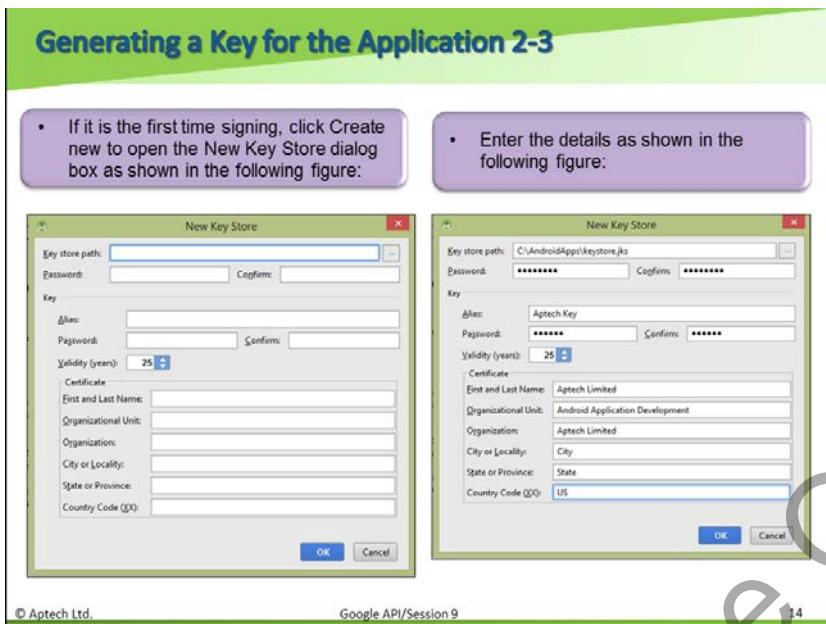
Slides 13 to 15

Generating a Key for the Application 1-3

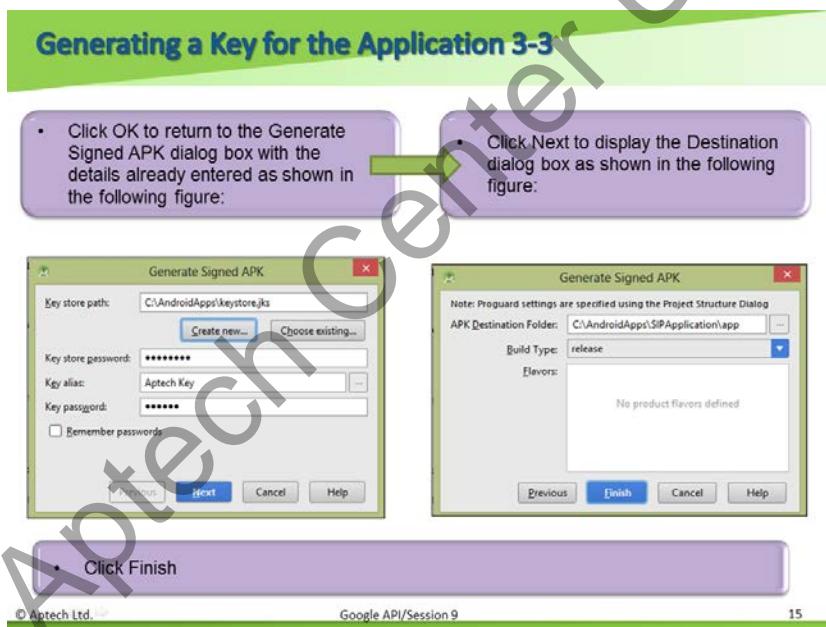
- Start Android Studio and open the desired project
- Navigate to the Build Menu and select Generate Signed APK
- The Generate Signed APK dialog box is displayed as shown in the following figure:



Using slide 13, explain the process of generating a signed apk. Explain that the first step for generating a signed application is to generate a unique key for the apk.



Using slide 14, explain the process of generating a key from the keystore. Explain that the developer needs to specify the details for the key which is stored inside a keystore.



Using slide 15, explain the steps to generate an application key for signing the APK. Explain that in order to use Google Play Services, the application needs to be signed. List the steps and explain the process using the figures given on the slide.



In-Class Question: Which dependency is mandatory to utilize Google API?

Answer: The play-services dependency is used for this purpose.

Retrieving a Google Maps API Key

Slides 16 to 24

Retrieving a Google Maps API Key 1-9

- The keytool.exe is present in the C:\Program Files\Java<JDK_version_number>\bin folder
- Type the following command to extract the SHA1 fingerprint: keytool -list -v -keystore C:\AndroidApps\keystore.jks. When prompted, enter the password as shown in the following figure:

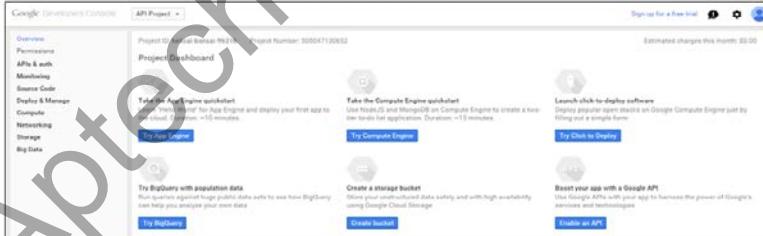


© Aptech Ltd. Google API/Session 9 16

Using slide 16, explain the process of extracting SHA1 fingerprint of the application key. Explain that this can be done using the command line and SDK tools.

Retrieving a Google Maps API Key 2-9

- Enter <https://console.developers.google.com/> in the address bar of a browser
- Sign-in with the Gmail ID of the developer. If you are using the Google APIs Console for the first time, then click Create Project to create a new project named API Project as shown in the following figure:

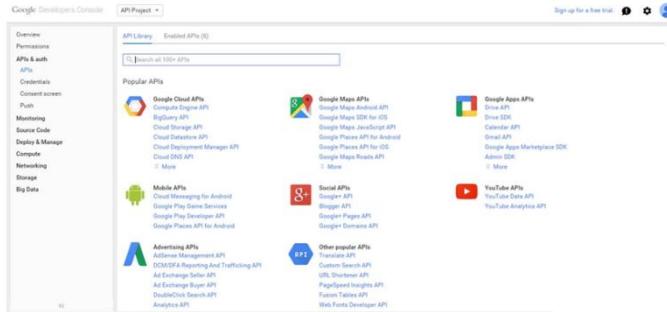


© Aptech Ltd. Google API/Session 9 17

Using slide 17, explain the process of accessing the Google Developer Console. Explain that the users need to access the developer console online and sign in using their Google account.

Retrieving a Google Maps API Key 3-9

- Click APIs & auth and select APIs from the left pane as shown in the following figure:



© Aptech Ltd.

Google API/Session 9

18

Retrieving a Google Maps API Key 4-9

- Select Google Maps Android API under Google Maps APIs and click Enable API. The output will be as shown in the following figure:



© Aptech Ltd.

Google API/Session 9

19

Retrieving a Google Maps API Key 5-9

- Select Google Maps Android API under Google Maps APIs and click Enable API. The output will be as shown in the following figure:



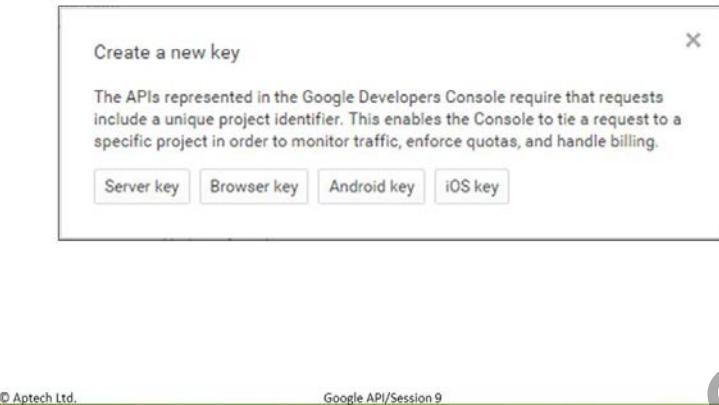
Retrieving a Google Maps API Key 6-9

- From the left pane, click Credentials as shown in the following figure:



Retrieving a Google Maps API Key 7-9

- Click Create new Key
- Click Android key to obtain an API key from the popup as shown in the following figure:



© Aptech Ltd.

Google API/Session 9

22

Using slides 18 to 22, explain the process of selecting the key that needs to be created. Each type of application requires its own set of details. Explain that we will select Android Key for our example.

Retrieving a Google Maps API Key 8-9

- Enter the application key followed by a semicolon and the package name as shown in the following figure:

Create an Android key and configure allowed Android applications

This key can be deployed in your Android application.

API requests are sent directly to Google from your client Android device. Google verifies that each request originates from an Android application that matches one of the certificate SHA1 fingerprints and package names listed below. You can discover the SHA1 fingerprint of your developer certificate using the following command:

```
keytool -list -v -keystore mystore.keystore
```

[Learn more](#)

Accept requests from an Android application with one of the certificate fingerprints and package names listed below

SHA1 Certificate Fingerprint and package name (separated by a semicolon) per line. Example:
45:31:84:6F:36:AD:0A:98:94:B4:02:66:2B:12:17:F2:56:26:AD:ED:com.example
Or leave this blank, requests will be accepted from any Android application. Be sure to add SHA1 certificates before using this key in production.

5B:98:E9:FB:92:5F:F3:F7:D4:AD:8E:26:B3:0E:B2:74:2A:90:83:08.aptech.com

Create **Cancel**

© Aptech Ltd.

Google API/Session 9

23

Retrieving a Google Maps API Key 9-9

- Click Create to generate an API key. The API key is created and displayed on the Credentials page as shown in the following figure:

© Aptech Ltd. Google API/Session 9 24

Using slides 23 and 24, explain the process of retrieving a Google Maps API key from Google Developer console. Explain that this key will be used by the application to communicate with the Google API servers. Explain that the key is used as a form of authentication and for keeping track of analytics. Point out the structure of the key generated using the figures given on the slides.

Google API Manifest Changes

Slides 25 and 26

Google API Manifest Changes 1-2

- The `AndroidManifest.xml` needs to be modified to add the key and set the permissions for your application to access Android features and Google Maps servers
- An example is shown in the following Code Snippet:

```

<permission android:name="com.trafficapp.permission.MAPS_RECEIVE"
    android:protectionLevel="signature"></permission>
<uses-permission android:name="com.trafficapp.permission.MAPS_RECEIVE"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-feature android:name="android.hardware.location" android:required="false" />
<uses-feature android:name="android.hardware.location.network" android:required="false" />
<uses-feature android:name="android.hardware.location.gps" />
<uses-feature android:name="android.hardware.wifi" android:required="false" />
<uses-feature android:glEsVersion="0x00020000" android:required="true"/>
<meta-data
    android:name="com.google.android.maps.v2.API_KEY"
    android:value="AIzaSyDr-IOLYKwNOjd8xh5NpNb4_KBckQVM2uy"/>
```

© Aptech Ltd. Google API/Session 9 25

Using slide 25, explain the changes that need to be made in the manifest file to utilize Google API. Explain that the api key can be supplied as part of the metadata.

Google API Manifest Changes 2-2

- If the project is made using the Google Maps activity, a string resource is used instead
- In this case, navigate to res → values and open google_maps_api.xml for editing and modify the code as shown in the following Code Snippet:

```
<resources>
<string name="google_maps_key" translatable="false"
    templateMergeStrategy="preserve">
    AIzaSyDr-IOLYKWN0jD8xh5NpNb4_KBckQVM2uY
</string>
</resources>
```

Using slide 26, explain the changes that need to be made to google_maps_api.xml file if the Google Maps Activity is selected. Explain that this file is a string resource file which contains the string with the name google_maps_key which is used within the auto generated code.

Map Fragment Example

Slide 27

Map Fragment Example

- The code to add a Map Fragment is shown in the following Code Snippet:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/ apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" >
        <fragment
            android:id="@+id/map"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            class="com.google.android.gms.maps.SupportMapFragment" />
    </LinearLayout>
</RelativeLayout>
```

Using slide 27, explain the xml code to include a Map fragment in the activity. Explain that the 'class' attribute of the fragment tag is used to indicate that the fragment is a Google Maps fragment.

Google Maps Example Application

Slide 28

Google Maps Example Application

- Using the code, an application for demonstrating Google Maps is created as shown in the following figure:

© Aptech Ltd. Google API/Session 9 28

Using slide 28, demonstrate the steps to create an application using Google Maps API. The application contains a map fragment which displays data retrieved from Google Maps. Explain the functional logic of the application and the output using the figure given on the slide.

Google Play Services

Slide 29

Google Play Services

- Play Services is a client library available for use by other applications to access the Google API
- Play Services is installed as an APK on devices by default. It is automatically updated on a regular basis
- On older devices, where play services is not pre-installed, the Play Store will make an automatic install
- The developer can assume that almost all devices have Play Services installed and freely use the services provided by the library

© Aptech Ltd. Google API/Session 9 29

Using slide 29, explain the importance of Play Services. Explain that Play Services is a client library which provides access to Google API. Explain that Play Services is automatically installed on all devices by the Play Store.



In-Class Question: What is Google Play Services?

Answer: Google Play Services is a client side library that provides access to Google API.

Additional Reference:

You may refer the following link for more information regarding Play Services:

https://en.wikipedia.org/wiki/Google_Play_Services

Google Play Services Classes

Slides 30 and 31

Google Play Services Classes 1-2

- ◆ **GoogleApiClient.Builder**

- ◆ The builder class is used to retrieve a reference to a GoogleApiClient object
- ◆ The API that will be utilized by the application can be specified using the addAPI() method
- ◆ The addConnectionCallbacks() method is used to set the callback listener
- ◆ The addOnConnectionFailedListener() method is used to set a call back listener if the connection to the service fails
- ◆ The build() method is used to retrieve an instance of GoogleApiClient

- ◆ **GoogleApiClient**

- ◆ This class is the main entry point for Google Play Services integration
- ◆ The blockingConnect() and connect() methods are used to make a blocking and a non-blocking connect respectively
- ◆ The disconnect() and reconnect() methods are used to disconnect and reconnect respectively
- ◆ The getSessionId() method is used to get the session id for the connection

Google Play Services Classes 2-2

- ◆ **GoogleApiClient**

- ◆ The registerConnectionCallbacks() is used to register callback listeners for non-blocking connect and unregisterConnectionCallbacks() method is used to unregister the listener

- ◆ **GooglePlayServicesUtil**

- ◆ A utility class which can be used to verify that Google Play Services are available on the devices and to check the version to ensure that they are up to date
- ◆ The static method isGooglePlayServicesAvailable() is used to check if Google Play Services is installed

Using slides 30 and 31, explain the important classes of the Google Play Services API. List the classes and explain their purpose using the bulleted points. Explain that the GooglePlayServicesUtil class is used to check support for Play Services in the device. Explain

that the GoogleApiClient class is used to create requests and register listeners.



In-Class Question: Which class is the entry point for Google Play Services?

Answer: The GoogleApiClient class is considered the entry point for Google Play Services.

Adding Dependencies for the Application

Slides 32 to 35

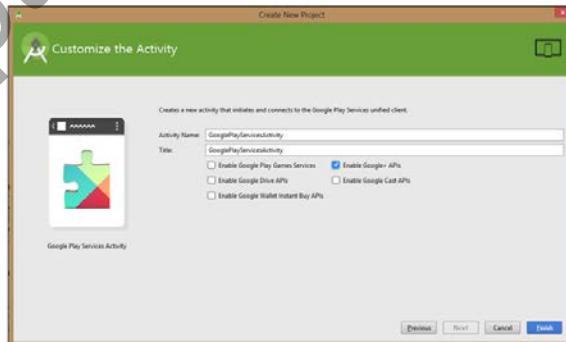
Adding Dependencies for the Application 1-4

- Create a project named PlayServices in Android Studio
- During the activity selection screen, select Google Play Services Activity as shown in the following figure:



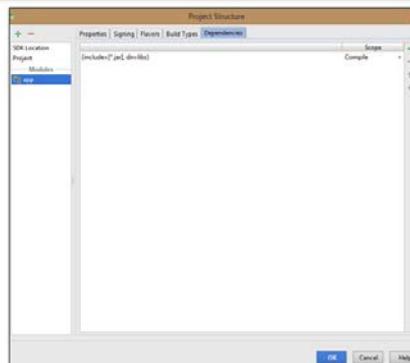
Adding Dependencies for the Application 2-4

- Click Next
- Select the Enable Google + APIs check box as shown in the following figure:



Adding Dependencies for the Application 3-4

- If the project is created using a Blank Activity, right-click app and select Open Module Settings
- From the Project Structure dialog box, select the Dependencies tab as shown in the following figure:



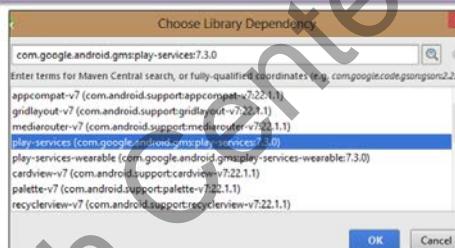
© Aptech Ltd.

Google API/Session 9

34

Adding Dependencies for the Application 4-4

- Click the '+' sign and select Library Dependency to open Choose Library Dependency dialog box as shown in the following figure.



- Select the play-services library and click OK
- Click OK again to confirm the dependencies
- Navigate to app → java → aptech.com.playservices and add the imports
- Generate a Key for the Application

© Aptech Ltd.

Google API/Session 9

35

Using slides 32 to 35, explain the process of adding dependencies for Google+ login API. The process is identical to that of Google Maps API.

Retrieve a Google+ API Key

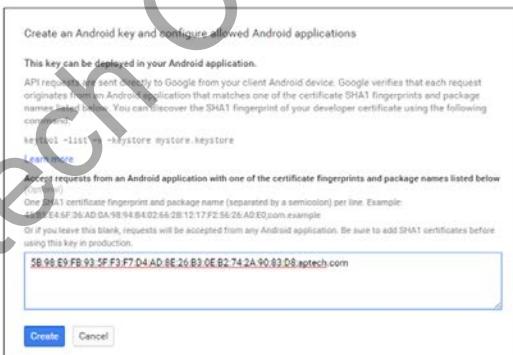
Slides 36 to 38

Retrieve a Google+ API Key 1-3

- ◆ The keytool.exe is present in the C:\Program Files\Java\<JDK_version_number>\ bin folder
- ◆ Type the following command to extract the SHA1 fingerprint:
keytool -list -v -keystore C:\AndroidApps\keysotre.jks .When prompted, enter the Password
- ◆ Enter <https://console.developers.google.com/> in the Address bar of a browser. This opens the Google console to register your application for the Maps API
- ◆ Sign-in with the Gmail id of the developer. If you are using the Google APIs Console for the first time, then click Create Project to create a new project named API Project
- ◆ Click API & auth and select APIs
- ◆ Select Google+ API and click Enable API
- ◆ From the Left pane, click Credentials

Retrieve a Google+ API Key 2-3

- Click Create new key
- Click Create new Android Key to obtain an API key from the popup
- Enter the application key followed by a semicolon and the package name as shown in the following figure:



Retrieve a Google+ API Key 3-3

- Click Create to generate an API key. The API key is created and displayed on the Credentials page as shown in the following figure:



Using slides 36 to 38, explain the process of retrieving an API key for Google+ authentication. Explain the steps using the figures given on the slides. The process is similar to retrieving an API key for Google Maps.

Google+ Login Button Example

Slide 39

Google+ Login Button Example

- The code to add a Google+ Sign in Button is shown in the following Code Snippet:

```
<LinearLayout>
<com.google.android.gms.common.SignInButton
    android:id="@+id/gpluslogin"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:onClick="onGPlusClicked"
    android:layout_marginBottom="20dp"/>
</LinearLayout>
```

Using slide 39, explain the process of adding the button for Google+ Login. Explain the code given on the slide. Explain that the Play Services dependency needs to be added prior to this.

Retrieving User Information Example

Slide 40

Retrieving User Information Example

- The code to retrieve logged in user information is shown in the following Code Snippet:

```

try {
    if (Plus.PeopleApi.getCurrentPerson(mGoogleApiClient) != null) {
        Person user = Plus.PeopleApi.getCurrentPerson(mGoogleApiClient);

        String userName =
        Plus.AccountApi.getAccountName(mGoogleApiClient);
        String userRealName = user.getDisplayName();
        String userPhotoUrl = user.getImage().getUrl();
        String userCoverUrl = user.getCover().getCoverPhoto().getUrl();
    }
}

```

© Aptech Ltd. Google API/Session 9 40

Using slide 40, explain the code to retrieve user information after the login. Explain that the code should be placed inside the onClick event listener. The 'Person' object of the logged in user is retrieved to get user details.



In-Class Question: Which tag is used to create a Google+ Sign in button in an Activity?

Answer: The com.google.android.gms.common.SignInButton is used for this purpose.

Google API Example

Slides 41 to 43

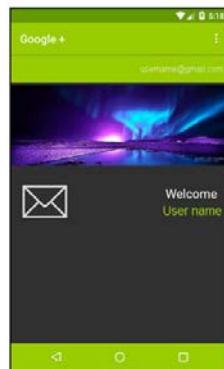
Google API Example 1-3

- Using the code, an application for demonstrating Google API is created as shown in the following figure:

© Aptech Ltd. Google API/Session 9 41

Google API Example 2-3

- Click the Red 'g+' sign in the SIGN IN button and sign in to your Google+ account
- Once the user signs in, the Google+ Home page is displayed as shown in the following figure:



© Aptech Ltd.

Google API/Session 9

42

Google API Example 3-3

- The users can logout from the action menu and return to the login page as shown in the following figure:



© Aptech Ltd.

Google API/Session 9

43

Using slides 41 to 43, demonstrate the steps for creating an application using Google Play Services. Explain that this provides authentication using Google+ credentials. Explain the functionality and output of the application using the figures given on the slides.

Google API Application Logic

Slide 44

Google API Application Logic

- ◆ A Google+ Sign in Button is added for the Main Activity
- ◆ An onClick() listener is registered for the Login Button
- ◆ Once the login process is completed, the login listener is triggered
- ◆ The user details are then retrieved from the Login Listener
- ◆ The user details are added as extras and the login Home Page is loaded



© Aptech Ltd.

Google API/Session 9

44

Using slide 44, explain the application logic and how it utilizes the code explained earlier. Explain that the Google+ Sign in Button is part of the Play Services library and the user details can be retrieved from the person object after the login process is completed. Explain that this can be done by using an onClick() listener.

Summary

Slide 45

Summary

- ◆ Google provides API for developers to access their product services such as search, drive, App Engine and so on
- ◆ Location-based Services (LBS) refer to a set of applications that exploit the knowledge of the geographical position of a mobile device in order to provide services based on that information
 - ◆ Google Maps allow the developers to develop applications of world with rich maps provided by Google
 - ◆ Google API Add on is an extension to Android SDK development environment that lets the developer develop applications for devices that include Google's set of custom applications, libraries, and services
- ◆ The GoogleAPIClient and GooglePlayServicesUtil classes are used to access Google API
- ◆ Google Play Services is a client side library for accessing Google services

© Aptech Ltd.

Google API/Session 9

45

Using slide 45, summarize the session. Make them revise the following points:

- Google provides API for developers to access their product services such as search, drive, App Engine, and so on.
- Location-based Services (LBS) refer to a set of applications that exploit the knowledge of the geographical position of a mobile device in order to provide

services based on that information.

- Google Maps allow the developers to develop applications of world with rich maps provided by Google.
- Google API Add on is an extension to Android SDK development environment that lets the developer develop applications for devices that include Google's set of custom applications, libraries, and services.
- The GoogleAPIClient and GooglePlayServicesUtil classes are used to access Google API.
- Google Play Services is a client side library for accessing Google services.

9.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes Web Services and their implementation, Text to Speech, and Voice Recognition.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 10: Web Services in Android

10.1 Pre-Class Activities

You should familiarize yourself with the concept of Web Services. You should also acquire a good understanding of creating Web Services and using third Party API. You should review the procedure to create and retrieve an API key from third party providers. You should acquire knowledge about Text-to-Speech (TTS) and Voice Recognition.

Familiarize yourself with the topics of the current session in-depth.

10.1.1 Objectives

After the session, learners will be able to:

- Explain and use of Web Services
- Explain standards for Web Services Communication
- Use third party API
- Create Web Services
- Understand the correct use of Web Service
- Explain and use Text-to-Speech and Voice Recognition

10.1.2 Teaching Skills

You should be familiar with the concept of Web Services and its applications. You should be able to explain when to and when not to use Web Service. Knowledge about creating custom Web Service is mandatory. You should also be familiar with the API for Text-to-Speech and Voice Recognition.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

10.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Tell the students that the previous session explained the basics of Google API, Play Services, API keys, and signing applications.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the use of third Party API, Web Services, Text-to-Speech, and Voice Recognition.

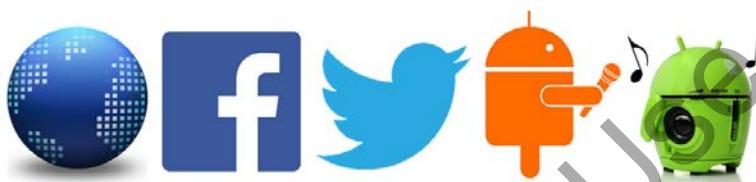
10.2 In-Class Explanations

Introduction

Slide 3

Introduction

- ◆ Web Services can enhance the application by providing computational offloading and information services
- ◆ Text to Speech and Voice Recognition provide unique and efficient ways for interacting with the application
- ◆ Coupled together, the developer can make an extremely user friendly and easy to use applications



© Aptech Ltd.

Web Services/Session 10

3

Using slide 3, introduce Web Services, TTS, and Voice Recognition. Explain that these features make the application more interactive and easy to use. Provide examples for application using Web services (such as Wisepilot), TTS (such as Google Chrome), and Voice Recognition (such as Google Search).

Web Services

Slide 4

Web Services

- ◆ Web Service is an application component which allows communication between devices over a network
- ◆ There are several platforms which provide useful resources which can be utilized in developing Android Applications
- ◆ This section describes the process of utilizing third party Web services, along with the various standards



© Aptech Ltd.

Web Services/Session 10

4

Using slide 4, explain the concept of Web Services. Define Web Services and provide a few examples for Web services such as the Wisepilot application.

Types of Web Services

Slides 5 and 6

Types of Web Services 1-2

- ◆ **ReSTful**
 - ◆ ReST is the abbreviation for Representational State Transfer
 - ◆ There are no concrete standards for implementing a Web service using ReST
 - ◆ Requests are sent as standard HTTP 'GET' requests and the reply can be in any format that is mutually understandable among the communicating parties
- ◆ **SOAP**
 - ◆ SOAP is an abbreviation for Simple Object Access Protocol
 - ◆ SOAP is an XML based Remote Procedure Call protocol. It relies on application layer protocols such as HTTP and SMTP
 - ◆ The client executes a remote method (procedure) on the Web server using an XML envelope to send the request
 - ◆ The Web service generates a response and sends it back to the client in an XML envelope

Types of Web Services 2-2

- ◆ **Custom/Proprietary**
 - ◆ Besides the two open standards, several Web services provide custom formats for communication with the Web service
 - ◆ These Web services usually come with extensive documentation on using these services or with libraries for use with the platforms

Using slides 5 and 6, explain the standard communication protocols for Web Services using the bulleted points. Explain the difference between SOAP and ReSTful. Explain that they are open standards and they can be used to utilize most of the available Web Services in the market. Using slide 6, explain that custom/proprietary implementations can be used when the functionality of the Web Services needs to be extended.

Additional Reference:

You may refer the following links for more information regarding SOAP and ReSTful:

<http://www.drdobbs.com/Web-development/restful-Web-services-a-tutorial/240169069>

https://en.wikipedia.org/wiki/Representational_state_transfer

<https://en.wikipedia.org/wiki/SOAP>

<http://searchsoa.techtarget.com/definition/SOAP>

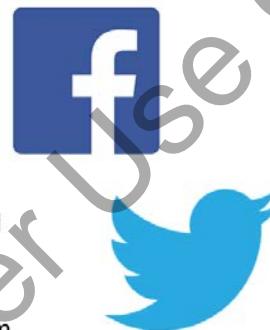
http://www.w3schools.com/webservices/ws_soap_intro.asp

Using Third Party API

Slide 7

Using Third Party API

- ◆ There are other API providers besides Google who provide similar or better functionality
- ◆ If the application is using authentication, then it will be a wise decision to integrate Facebook login into applications
- ◆ This reduces the hassle for registration and provides a trustable authentication system for the application
- ◆ It is also possible to use social features from Facebook such as friend's lists and post to wall



Using slide 7, explain the importance of third party API. You can use examples of Facebook and Twitter who provide API for authentication and information retrieval.



In-Class Question: What are the primary standards for communications with Web Services?

Answer: The two commonly used standards are:

- ReSTful
- SOAP

Generating a Hash Key

Slides 8 and 9

Generating a Hash Key 1-2

- ◆ Generate an Application key for signing the application
- ◆ Download OpenSSL for windows from the URL
https://code.google.com/p/openssl-for-windows/downloads/detail?name=openssl-0.9.8k_WIN32.zip&can=2&q
- ◆ Extract the contents of the zip file to C:\openssl
- ◆ Navigate to the keytool
- ◆ Enter the following command shown in the following Code Snippet:

```
keytool -exportcert -alias "apttech key" -keystore "C:\AndroidApps\keystore.jks"
| "C:\openssl\bin\openssl" sha1 -binary | "C:\openssl\bin\openssl" base64
```

© Aptech Ltd.

Web Services/Session 10

8

Using slide 8, explain the steps to generate a Hash key for use with Facebook. Explain that this key is different from the key used to sign in to the application.

Generating a Hash Key 2-2

- ◆ Enter the password
- ◆ The key is displayed as shown in the following figure:

```
C:\Program Files\Java\jdk1.8.0_40\bin>keytool -exportcert -alias "apttech key"
-keystore "C:\AndroidApps\keystore.jks" | "C:\openssl\bin\openssl" sha1 -binary
| "C:\openssl\bin\openssl" base64
Enter keystore password: pressure
6L5/p10Qzr8=
```

- ◆ Store the key value

© Aptech Ltd.

Web Services/Session 10

9

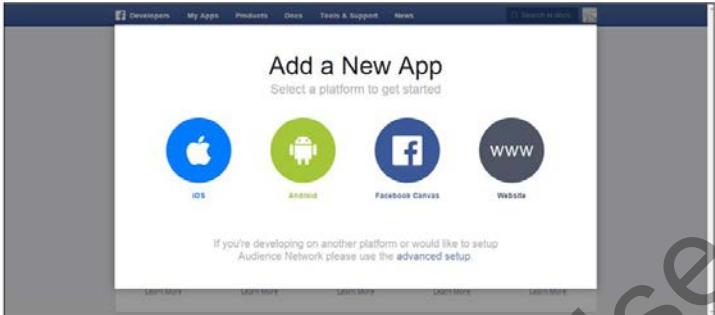
Using slide 9, explain the output of generating the key. Explain that the key is displayed on the console once it is generated. Mention that the developer needs to store this for use with Facebook.

Generating an API Key

Slides 10 to 17

Generating an API Key 1-8

- Open <https://developers.facebook.com> and enable developer account
- From the My Apps menu, select Add a New App. The app type selection screen is displayed as shown in the following figure:



© Aptech Ltd. Web Services/Session 10 10

Generating an API Key 2-8

- Select Android and enter the app name as Aptech Auth and select Create New Facebook App ID as shown in the following figure:



© Aptech Ltd. Web Services/Session 10 11

Generating an API Key 3-8

- Select a category and click Create a New App ID as shown in the following figure:



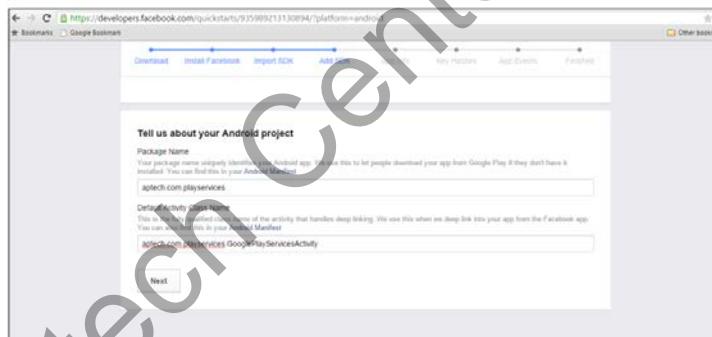
© Aptech Ltd.

Web Services/Session 10

12

Generating an API Key 4-8

- Download the SDK (If using the Eclipse IDE) and the APK file (If the Facebook App is not installed on your device) and navigate to the bottom of the page
- Enter the app details as shown in the following figure:



© Aptech Ltd.

Web Services/Session 10

13

Generating an API Key 5-8

- Click Next. If a warning pops up, click Use this package name as shown in the following figure.



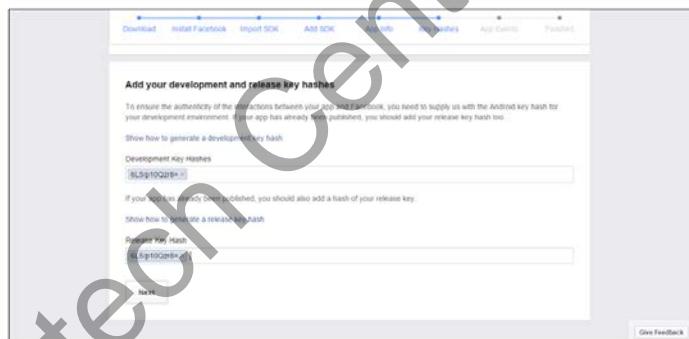
© Aptech Ltd.

Web Services/Session 10

14

Generating an API Key 6-8

- Enter the hash key generated earlier as shown in the following figure:



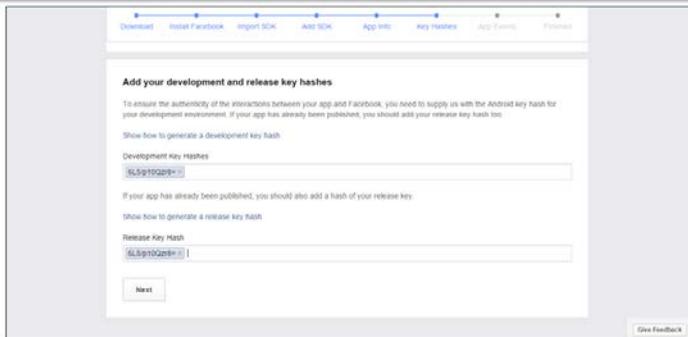
© Aptech Ltd.

Web Services/Session 10

15

Generating an API Key 7-8

- Enter the hash key generated earlier as shown in the following figure:



- Click Next
- Scroll down to the bottom of the page to see the finished image

© Aptech Ltd.

Web Services/Session 10

16

Generating an API Key 8-8

- Navigate to My Apps menu and select the created application Aptech Auth
- Copy the App id displayed as shown in the following figure:



© Aptech Ltd.

Web Services/Session 10

17

Using slides 10 to 17, explain the steps to generate a Facebook API key. Explain that this key will be used by the application to identify itself. Using slide 10, explain the process of accessing Facebook developer tools. Explain that this can be done using the developers.facebook.com URL.

Using slide 13, explain the steps to add the description of the application. Mention that the developers need to provide the default class name which acts as the main activity of the application.

Using slide 15, explain that the developers need to provide the Hash key used by the application, which was generated earlier.

Explain the steps and the output of the process using the figures given on the slides. Explain that the API key can be retrieved under the App ID field.

Additional Reference:

You may refer the following link for more information about the process to generate the Facebook API Key:

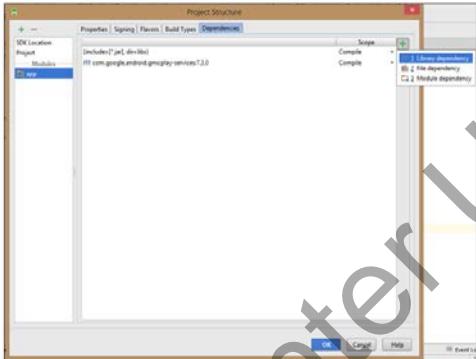
<https://developers.facebook.com/>

Setting up Android Studio

Slides 18 and 19

Setting up Android Studio 1-2

- Open the project PlayServices
- Right-click app and select Open Module Settings
- Navigate to the Dependencies tab and click the '+' sign
- Select Library dependency as shown in the following figure:



© Aptech Ltd.

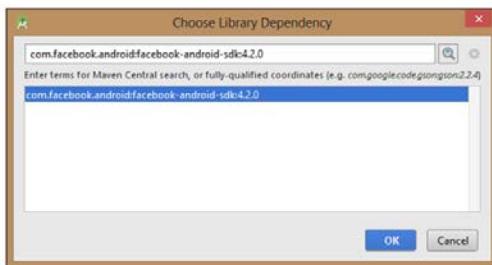
Web Services/Session 10

18

Using slide 18, explain the process of setting up Android Studio support for Facebook API. Explain that the Facebook SDK is an external library and it should be added as a dependency for use within the application.

Setting up Android Studio 2-2

- In the popup box, enter the name com.facebook.android:facebook-android-sdk: and select the Facebook library as shown in the following figure:



- Click OK
- Click OK again in the Dependencies tab

© Aptech Ltd.

Web Services/Session 10

19

Using slide 19, explain that the developer should add the downloaded Facebook SDK as a dependency to the application. Explain that the user needs to search for com.facebook.android. Mention that the version number should not be provided during searches as it changes over a period of time.

Setting up the Target Device

Slide 20

Setting up the Target Device

- ◆ Navigate to the Play Store and search for the Facebook application and click Install
- ◆ The app will be downloaded and installed on the device
- ◆ Download the Facebook application from the Developer Page as described in the previous section
- ◆ Connect the target device using USB with adb
- ◆ Execute the command as shown in in the following Code Snippet:

```
adb install C:\Path_here\Facebook-{version_here}.apk
```

© Aptech Ltd. Web Services/Session 10 20

Using slide 20, explain the process of setting up the device. Explain that the device needs to have the Facebook application installed to utilize Facebook's API within the application.



In-Class Question: Which dependency needs to be added to include support for Facebook SDK?

Answer: The dependency com.facebook.android:facebook-android-sdk needs to be added.

Facebook Login Button Code Snippet

Slide 21

Facebook Login Button Code Snippet

- ◆ The code to add a Facebook Sign in Button is shown in the following Code Snippet:

```
<LinearLayout>
<com.facebook.login.widget.LoginButton
    android:id="@+id/fbLoginButton"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_weight="0.11"
    android:minHeight="50dp"
    android:padding="10dp" />
</LinearLayout>
```

© Aptech Ltd. Web Services/Session 10 21

Using slide 21, explain the process of adding a Facebook login button.

Explain the code given on the slide.

A login button is added to the layout. Explain that this can only be done if the Facebook SDK is added as a dependency.

Additional Reference:

You may refer to the following link for more information about the Facebook SDK and sign in API:

<https://developers.facebook.com/docs/facebook-login/android>

Requesting User Information Code Snippet

Slide 22

Requesting User Information Code Snippet

- ♦ The code to request Logged in user information is shown in the following Code Snippet:

```
FacebookSdk.sdkInitialize(getApplicationContext());
        callbackManager = CallbackManager.Factory.create();
        fbPermissions = new ArrayList<String>();
        fbPermissions.add("public_profile");
        fbPermissions.add("email");
        fbPermissions.add("user_friends");
        fbLoginButton.setReadPermissions(fbPermissions);
        fbLoginButton.registerCallback(callbackManager, new
FacebookCallback<LoginResult>() {
    @Override
    public void onSuccess(LoginResult loginResult) accessToken =
loginResult.getAccessToken();
    loadFbInfo();
})
```

Using slide 22, explain the code to retrieve the details of the logged in user. Explain that the information being queried needs to be added prior to generating the login request. Explain that the listener onSuccess() can be used to receive notification on a successful login.

Retrieving User Information Code Snippet

Slide 23

Retrieving User Information Code Snippet

- The code to retrieve Logged in user information is shown in the following Code Snippet:

```
Profile userProfile = Profile.getCurrentProfile();

String userId = userProfile.getId();
String userRealName = userProfile.getName();
String userPhotoUrl =
userProfile.getProfilePictureUri(100,100).toString();
```

© Aptech Ltd. Web Services/Session 10 23

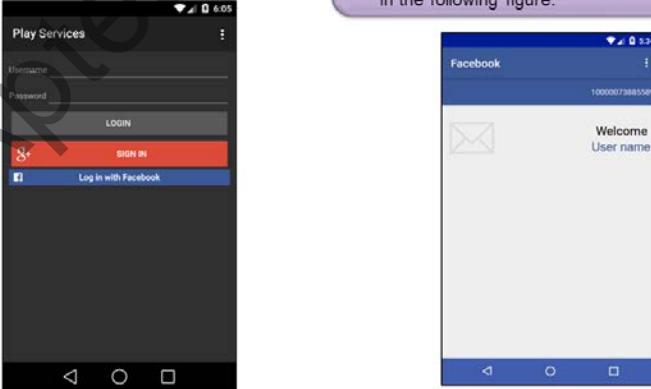
Using slide 23, explain the code to retrieve user information of a logged in user. Explain that the information can be retrieved from a Profile object using the getCurrentProfile() method.

Third Party API Example Application

Slides 24 and 25

Third Party API Application 1-2

- Using the code, an application for demonstrating third Party API is created as shown in the following figure:
- Click the Log in with Facebook button and login to your Facebook account
- Once logged in, the user is redirected to the Facebook login homepage as shown in the following figure:



© Aptech Ltd. Web Services/Session 10 24

Third Party API Application 2-2

- The users can logout from the action menu and return to the login page as shown in the following figure:



© Aptech Ltd.

Web Services/Session 10

25

Using slides 24 and 25, demonstrate the steps for creating an application implementing Facebook authentication. Explain that this application will use Facebook's API to provide authentication feature within the application. Explain the functionality and output of the application using the figures given on the slides.



In-Class Question: Which class is used to retrieve user details of a logged in Facebook user?

Answer: The Profile class is used to retrieve user details of a logged in Facebook user.

Third Party API Application Logic

Slide 26

Third Party API Application Logic

- A Facebook Sign in Button is added for the Main Activity
- An `onSuccess()` listener is registered for the Login Button
- Once the login process is completed, the login listener is triggered
- The user details are then retrieved from the Login Listener
- The user details are added as extras and the login Home Page is loaded



© Aptech Ltd.

Web Services/Session 10

26

Using slide 26, explain the functional logic of the application. Explain that once the users click the login button, they are redirected to login to their Facebook account.

The details of the logged in user can be retrieved after the login process is completed using the onSuccess() listener.

Creating Web Services

Slide 27

Creating Web Services

- ◆ Before we create an application using the Web service, the Web service itself needs to be created
- ◆ For demonstration purposes, a simple PHP Web service is created running on an Apache Server
- ◆ The PHP code is shown in the following Code Snippet:

```
<?php

$username = $_GET['uname'];
$password = $_GET['pass'];

if($username == "admin" && $password == "adminpass")
{ echo "success"; }
else
{ echo "failure"; }

?>
```

© Aptech Ltd.

Web Services/Session 10

27

Using slide 27, explain the process of creating a basic authentication Web Service. Explain the code given on the slide. The PHP code checks for the username and password to match 'admin' and 'adminpass' respectively and returns the appropriate response.

Authenticating with the Web Services

Slide 28

Authenticating with the Web Services

- ◆ The code to authenticate with the created Web Service is shown in the following Code Snippet:

```
Uri.Builder builder = new Uri.Builder();
builder.scheme("http")
.authority("192.168.1.19")
.appendPath("webservice")
.appendPath("index.php")
.appendQueryParameter("uname", userName)
.appendQueryParameter("pass", passWord);
Uri webService = builder.build();

try {
HttpClient = new DefaultHttpClient();
HttpResponse response = httpclient.execute(new HttpGet(webService.toString()));
checkResponse(stringBuffer.toString());
}
catch (IOException e) {
e.printStackTrace();
}
```

© Aptech Ltd.

Web Services/Session 10

28

Using slide 28, explain the code to generate a simple ReSTful request for authentication. The user details are passed as get parameters. The UriBuilder class is used to formulate the appropriate URL with the get parameters. The request is then sent to the server using HttpClient.

Custom Web Service Example

Slides 29 and 30

Custom Web Service Example 1-2

- Using the code, an application for demonstrating Custom Web Services is created as shown in the following figure:
- Enter the Username as admin and Password as adminpass and click LOGIN as shown in the following figure:

© Aptech Ltd. Web Services/Session 10 29

Custom Web Service Example 2-2

- Once logged in, the user is redirected to the Web Service home screen as shown in the following figure:
- Users can logout from the Action menu to return to the home screen as shown in the following figure:

© Aptech Ltd. Web Services/Session 10 30

Using slides 29 and 30, demonstrate the steps for creating an application capable of authenticating users with a custom Web Service. Explain that this application will receive a request and use the RESTful protocol. Explain the functionality and the output of the application using the figures given on the slides.

Custom Web Service Example Application Logic

Slide 31

Custom Web Service Application Logic

- ◆ A Custom Web Service is created for the application
- ◆ Once the user enters the login details and clicks the Login button, the details are retrieved and passed to the service using a get request
- ◆ The Service checks the username and password and if they match, returns "success". Else it returns "failure"
- ◆ The application checks the server response and loads the activity if log in was successful



© Aptech Ltd.

Web Services/Session 10

31

Using slide 31, explain the functional logic of the application. Explain that a server is setup with a simple authentication Web Service. The application communicates with the Web Service using ReSTful and redirects the user to the appropriate activity.

Web Service Implementation

Slide 32

Web Service Implementation

- ◆ There are service providers for almost all kinds of applications. There are also storage providers such as Google Drive, Computation providers such as Amazon EC2, and so on
- ◆ The first step is to decide the communication standard that will be used for the Web Service
 - ◆ ReSTful is an architectural style whereas SOAP is a protocol implementation
 - ◆ All the requirements need to be communicated beforehand
 - ◆ Once the development is complete, the Android Developer is provided with the API and the associated documentation of the Web Service

© Aptech Ltd.

Web Services/Session 10

32

Using slide 32, explain the methods for implementing Web services. Explain the various choices that need to be made when creating a Web service such as the host, communication standard, and other requirements.

When to NOT Implement Web Services?

Slide 33

When to Not Implement Web Services?

- ◆ If the solution can be implemented locally on the device
- ◆ If battery usage is a concern
- ◆ If application is expected to respond quickly
- ◆ If data access is required
- ◆ When Sensitive data is involved
- ◆ If the number of requests is expected to be low
- ◆ If the application is expected to function offline

Using slide 33, explain the situations where utilizing the Web Services is not recommended. List the bulleted points and explain that it is not financially or logically beneficial in these circumstances.

When to Implement Web Services?

Slide 34

When to Implement Web Services?

- ◆ If a platform is already established
- ◆ If cross-platform services need to be provided
- ◆ When scaling is a concern. Web Services can scale up very easily
- ◆ Data Encapsulation is a concern. The mode and structure of storage of information is not revealed to the user
- ◆ When fault tolerance is desired. Web Services can quickly recover from major errors/crashes without revealing it to the user
- ◆ If services are planned to be changed/tuned. Web Services provide the option to completely rewrite the implementation without affecting the service
- ◆ If a task is intensive and offloading it can save energy and time on the device



Using slide 34, explain the circumstances where utilizing the Web Services is favorable using the points given on the slide.

Text-to-Speech and Voice Recognition

Slide 35

Text-to-Speech and Voice Recognition

- ◆ Text-to-Speech is process of narrating text to the user in audio form
- ◆ Voice Recognition is process of taking the user's audio input and generating text
- ◆ Android readily provides API for both these operations



© Aptech Ltd.

Web Services/Session 10

35

Using slide 35, explain the concepts of Text-to-Speech and Voice Recognition. Explain that Android provides native support for both of these features.

Relevant Classes

Slides 36 and 37

Relevant Classes 1-2

- ◆ **Voice Recognition**
 - ◆ **RecognizerIntent:** A broadcast intent that starts an activity for accepting user voice input. The extras for this Intent can be used for setting additional settings
- ◆ **Text-to-Speech**
 - ◆ **TextToSpeech:**
 - ◆ The central class used in text to speech systems
 - ◆ It is responsible for generating speech from input text
 - ◆ The `setPitch()` method is used to set the pitch of the voice and the `setSpeechRate()` method is used to set the speed of the voice
 - ◆ The `setVoice()` method can be used to set the voice of the speech if multiple voice packs are available
 - ◆ The `synthesizeToFile()` method can be used to save the speech output to a file

© Aptech Ltd.

Web Services/Session 10

36

Relevant Classes 2-2

- ◆ **TextToSpeech.OnInitListener:**
 - ❖ The event listener class for TextToSpeech
 - ❖ The `onInit()` method is called once TextToSpeech has been initialized
 - ❖ The method can be implemented to set the base settings for TTS
- ◆ **Voice:**
 - ❖ The class representing a voice pack installed on the device
 - ❖ The name of the voice can be retrieved by using `getName()` method
 - ❖ The `getQuality()`, `getLatency()`, and `getLocale()` methods are used to retrieve the quality, latency, and locale information respectively

Using slides 36 and 37, explain the API for Text-to-Speech and Voice Recognition. List the important classes and explain their functionality using the bulleted points.



In-Class Question: Which Intent is used to register a BroadcastListener for Voice Recognition?

Answer: The RecognizerIntent is used to register a BroadcastListener for Voice Recognition.

Additional Reference:

You may refer to the following links for more information regarding Text-to-Speech and Voice Recognition API:

<http://developer.android.com/reference/android/speech/RecognizerIntent.html>

<http://developer.android.com/reference/android/speech/tts/TextToSpeech.html>

<http://developer.android.com/reference/android/speech/tts/Voice.html>

Voice Recognition Example

Slide 38

Voice Recognition Example

- The code to receive voice input from the user is shown in the following Code Snippet:

```
Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);

intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,RecognizerIntent.LANGUAGE_
MODEL_FREE_FORM);

intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE, "en-US");
intent.putExtra(RecognizerIntent.EXTRA_PROMPT, "Speak a few words for
TTS");
try {
    startActivityForResult(intent,1);
} catch (ActivityNotFoundException a) {
    Toast.makeText(getApplicationContext(),"Voice Recognition not
supported on this Device / Currently Unavailable",Toast.LENGTH_SHORT).show();
}
...
ArrayList<String> result =
data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
```

© Aptech Ltd.

Web Services/Session 10

38

Using slide 38, explain the process of voice/speech recognition. Explain the code given on the slide. An intent is created for initiating voice recognition and the activity is started. In the event where voice recognition support is not available, a toast informing about it is displayed.

Text-to-Speech Example

Slide 39

Text-to-Speech Example

- The code convert text to speech is shown in the following Code Snippet:

```
TextToSpeech textToSpeech = new TextToSpeech(this, new
TextToSpeech.OnInitListener() {
    @Override
    public void onInit(int status) {
        textToSpeech.setLanguage(Locale.US);
        List<String> voiceList = new ArrayList<String>();
        try {
            Set<Voice> voices = textToSpeech.getVoices();
            if (voices.size() > 0) {
                Iterator<Voice> iterator = voices.iterator();
                while (iterator.hasNext()) {
                    voiceList.add(iterator.next().getName());
                }
            }
        } catch(NullPointerException e) {
            voiceList.add("Default");
        }
    }
});
```

© Aptech Ltd.

Web Services/Session 10

39

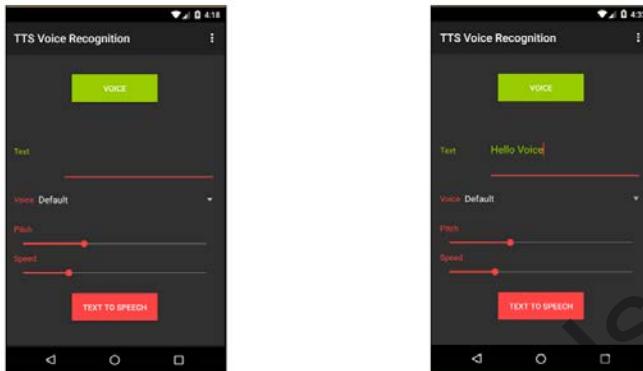
Using slide 39, explain the process of converting text to speech programmatically. Explain the code given on the slide. Explain that the TextToSpeech class is the core component of the TTS framework. Explain that the onInit() method should be implemented to initialize the Text- to-Speech settings.

TTS and Voice Recognition Application

Slide 40

TTS and Voice Recognition Application

- Using the explained code, an application for demonstrating TTS and Voice Recognition is created as shown in the following figure:



- Click the voice button and speak few words. Once completed, the voice is converted into text and the output is displayed in the text box as shown in the following figure:

Using slide 40, demonstrate the steps for creating an application capable of Text-to-Speech conversion and voice recognition. Explain that the application can convert voice input from the user to text. The user can then edit the text and use TTS to narrate the text. Explain the functionality and the output of the application using the figures given on the slides.

TTS and Voice Recognition Application Logic

Slide 41

TTS and Voice Recognition Application Logic

- The Main Activity consist of a button to receive Voice Input, and another button to Convert the text to Voice
- An EditText is provided to display the input text and to allow the user to enter his own input text manually
- Controls are provided for Voice Settings
- Once the Text to Speech button is clicked, the text is converted to voice with the settings selected by the user
- The TextToSpeech's speak() method is used for this purpose



Using slide 41, explain the functional logic of the application using the bulleted points. Explain that the application consists of a single activity with controls for Voice Recognition and Text-to-Speech.

Summary

Slide 42

Summary

- Web Services are application components that allow communicating devices to use features that are not available on the device locally
- An application can use both Google API as well as third party Web Services
- ReSTful and SOAP are well established communication standards for Web Services
- Unnecessary use of Web Services can hurt the application performance and usability
- Text-to-Speech (TTS) is the process of narration of text to the user. Android has built-in support for TTS which can be accessed via the TextToSpeech class
- Voice Recognition is the converse of TTS. It is the process of converting user audio input into text

Using slide 42, summarize the session. Make them revise the following points:

- Web Services are application components that allow communicating devices to use features that are not available on the device locally.
- An application can use both Google API as well as third party Web Services.
- ReSTful and SOAP are well established communication standards for Web Services.
- Unnecessary use of Web Services can hurt the application performance and usability.
- Text-to-Speech (TTS) is the process of narration of text to the user. Android has built-in support for TTS which can be accessed via the TextToSpeech class.
- Voice Recognition is the converse of TTS. It is the process of converting user audio input into text.

10.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes wireless and networks including Wi-Fi, Bluetooth, Network Connections and NFC.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 11: Wireless and Networking

11.1 Pre-Class Activities

You should familiarize yourself with various modes of wireless communication and networking supported by Android. You should study the API for managing Wi-Fi connections, Bluetooth data transfer, data connections. In addition to this, familiarize yourself with the concepts of NFC and NFC supported by Android.

Familiarize yourself with the topics of the current session in-depth.

11.1.1 Objectives

After the session, learners will be able to:

- Explain Bluetooth, Network, Wi-Fi, and NFC
- Explain the process of using Bluetooth
- Explain network management and Internet connectivity
- Explain working with Wi-Fi
- Explain working with NFC

11.1.2 Teaching Skills

You should be familiar with various wireless communication standards supported by Android and the API to utilize them. You should also study the concepts of Near Field Communication, its uses, and the API to use them.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

11.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Revise the concepts of Web services and the process to create custom Web services. Also, revise the ‘Text to Speech’ and ‘Voice Recognition’ concepts and their respective APIs.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. List the wireless standards supported by Android, providing a brief description for each of them. Specify that these will be explained in detail over the course of this session.

11.2 In-Class Explanations

Introduction

Slide 3

Introduction

- ◆ High speed data transfer is essential for Smart Phones
- ◆ Wireless was the chosen medium as mobile devices cannot be tied down with a wired connection
- ◆ The Wireless Data Transfer standards currently supported by Android are:
 - ◆ Bluetooth
 - ◆ Network Connections
 - ◆ Wi-Fi
 - ◆ NFC



© Aptech Ltd. Wireless and Networking / Session 11 3

Using slide 3, explain the importance of wireless communication and how it has changed the mobile industry as a whole. High speed data transfers are now considered a norm. List the wireless standards supported by Android and provide a brief description for each of them.

Bluetooth

Slide 4

Bluetooth

- ◆ Bluetooth operates in the 2.4 GHz wireless spectrum
- ◆ It provides a means for short distance data transfer at reasonable speeds
- ◆ Android provides Bluetooth APIs, with the following functionalities:
 - ◆ Scan for Bluetooth devices which are present nearby
 - ◆ Query for paired Bluetooth devices using the local Bluetooth adapter
 - ◆ Establishing the RFCOMM channels
 - ◆ Connect through service discovery to other devices
 - ◆ Exchange data to and from other devices
 - ◆ Manage multiple connections



© Aptech Ltd. Wireless and Networking/Session 11 4

Using slide 4, explain the Bluetooth standard and its application. Provide a brief description of the functionality provided by the Android Bluetooth API using the bulleted points.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Bluetooth support in Android.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/guide/topics/connectivity/bluetooth.html>

Bluetooth API Classes

Slides 5 to 9

Bluetooth API Classes 1-5

◆ **BluetoothAdapter:**

Is a representation of the local Bluetooth adapter and helps to perform the following functionalities:

- ◆ Discover all Bluetooth devices
- ◆ Query a list of devices
- ◆ Create a Bluetooth device instance using the MAC address
- ◆ Create a BluetoothServerSocket which will enable it to listen for communications from other Bluetooth connected devices

◆ **BluetoothDevice:**

Is a representation of a remote Bluetooth device and performs the following functionalities:

- ◆ Request a connection with remote device using BluetoothSocket
- ◆ Retrieve information about the device such as name, address, and state of the device

Bluetooth API Classes 2-5

◆ **BluetoothSocket:**

Is a representation of the interface like TCP socket and performs the following functionalities:

- ◆ Allows data exchange between the Bluetooth connected devices via InputStream and OutputStream can occur through BluetoothSocket

◆ **BluetoothServerSocket:**

Is a representation of an open server socket and performs the following functionalities:

- ◆ Listens for the incoming requests
- ◆ Responds to the received requests by returning a Bluetooth socket when the connection is accepted

Bluetooth API Classes 3-5

- ◆ **Bluetooth Class:**

Is a read only set of properties that describe the characteristics and capabilities of a Bluetooth device and all Bluetooth profiles and services supported by the particular device

- ◆ **Bluetooth Profile:**

Is a wireless interface specification for communication between Bluetooth based devices

- ◆ **Bluetooth Handset:**

It provides support for Bluetooth headsets that are connected with the mobile device

Bluetooth API Classes 4-5

- ◆ **BluetoothA2dp:**

It stands for Advanced Audio Distribution Profile and performs the following functionality:

- ◆ Defines the way of streaming a high quality audio from one Bluetooth connected device to another

- ◆ **Bluetooth Health:**

It represents a Health Device profile proxy and is used for controlling the Bluetooth service

- ◆ **BluetoothHealthCallback:**

Is an abstract class that the callback methods must be implemented for the following functionality:

- ◆ Receive updates about the changes in the Bluetooth channel state and application registration

Bluetooth API Classes 5-5

◆ **BluetoothHealthAppConfiguration:**

It represents a configuration of the Bluetooth Health third party application that is used for communicating with a remote Bluetooth health device

◆ **BluetoothProfile.ServiceListener:**

Is an interface that performs the following functionality:

- ◆ Notifies the IPC clients whenever they are connected or disconnected from the service

Using slides 5 to 9, explain the Bluetooth API provided by Android. List the classes, providing a brief description for each of them. Using slide 5, explain the classes used for device discovery. Using slide 6, explain the classes for initiating connections. Using slides 7 and 8, explain the support classes which provide additional information regarding the device and connection. Using slide 9, explain the additional configuration classes.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Bluetooth support in Android.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/reference/android/bluetooth/package-summary.html>



In-Class Question: What is the spectrum that Bluetooth operates in?

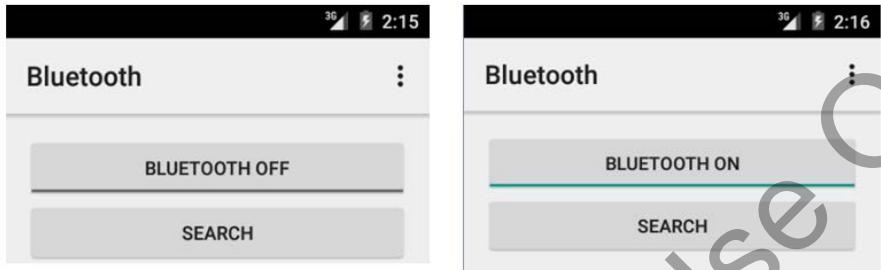
Answer: Bluetooth operates at 2.4 GHz wireless spectrum.

Bluetooth Example Application

Slides 10 and 11

Bluetooth Example Application 1-2

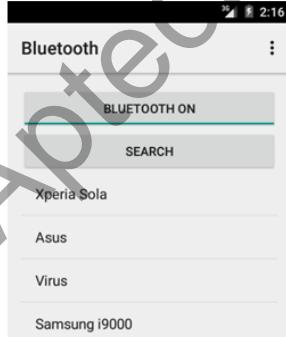
- Using the classes, an application for demonstrating Bluetooth functionality is created as shown in the following figure:



© Aptech Ltd. Wireless and Networking/Session 11 10

Bluetooth Example Application 2-2

- When SEARCH button is clicked the output will be as shown in the following figure:



- The application logic is as follows:

- ◆ Developer creates an Activity with a Toggle Button and a normal Button
- ◆ The Toggle Button is used to Turn BLUETOOTH ON and OFF
- ◆ When the SEARCHButton is clicked, the search() method is called
- ◆ The method retrieved the Bonded devices using the BluetoothAdapter's getBondedDevices() method
- ◆ The retrieved list is displayed to the user

© Aptech Ltd. Wireless and Networking/Session 11 11

Using slides 10 and 11, demonstrate the steps to create an application that utilizes the Bluetooth services. The application searches and lists Bluetooth devices. Demonstrate and explain the functionality and output of the application using the figures given on the slides. Using slide 11, explain the application logic and the code used to create the application.

Network**Slide 12****Network**

- ◆ In Android, the type of the connection is irrelevant and hidden from the application
- ◆ As long as a network connection is established, it will be available to access using the classes HttpClient and HttpURLConnection
- ◆ The Network Connection could be:
 - ◆ Mobile data connection
 - ◆ Wi-Fi network connection
 - ◆ Wired connection
 - ◆ Bluetooth tether



© Aptech Ltd.

Wireless and Networking/Session 11

12

Using slide 12, explain the basics of a network connection in Android. Explain that the mode of connection is considered as irrelevant. List various modes of establishing a network connection. Provide a brief idea of the network API.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Network Connections in Android.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/reference/java/net/HttpURLConnection.html>

<http://developer.android.com/preview/behavior-changes.html#behavior-apache-http-client>

HttpClient

Slide 13

HttpClient

- ◆ HttpClient encapsulates a smorgasbord objects which are required to execute HTTP requests to send and receive the data over the network
- ◆ HttpClient simplifies the handling of Http Requests
- ◆ The thread safety of HttpClient depends on the way the configuration of a specific client is being made

© Aptech Ltd. Wireless and Networking/Session 11 13

Using slide 13, explain the purpose of the HttpClient class. It is responsible for sending and receiving data across a network.

HttpURLConnection

Slide 14

HttpURLConnection

- ◆ HttpURLConnection is used to make a single request to the Http Server
- ◆ Connection for the particular URL can be established by invoking the openConnection() method and the data from the URL can be read by invoking the method getInputStream()
- ◆ It is used for sending and receiving data over the Web
- ◆ The class uses the GET and POST method to send and receive data
- ◆ By default it uses the GET method
- ◆ The other HTTP methods are used with setRequest() method

© Aptech Ltd. Wireless and Networking/Session 11 14

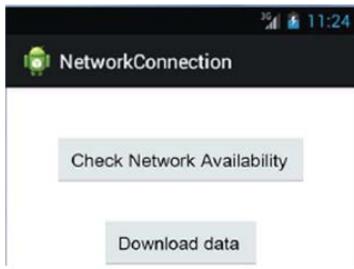
Using slide 14, explain the purpose of the HttpURLConnection and its methods. Explain the difference between GET and POST requests and specify the default method used by HttpURLConnection. Explain the important methods of the class HttpURLConnection.

Network Example Application

Slides 15 and 16

Network Example Application 1-2

- Using the classes, an application for demonstrating Network functionality is created as shown in the following figure:



- When Check Network Availability button is clicked the output will be as shown in the following figure:



© Aptech Ltd. Wireless and Networking/Session 11 15

Network Example Application 2-2

- When Download data button is clicked the output will be as shown in the following figure:



- Downloaded data displays the final output as shown in the following figure:



© Aptech Ltd. Wireless and Networking/Session 11 16

Using slides 15 and 16, demonstrate the steps to create an application that checks and utilizes a network connection. The application checks the network connection availability and downloads basic weather data. Demonstrate and explain the functionality and output of the application using the figures given on the slides.

Network Example Application Logic

Slide 17

Network Example Application Logic

- The application logic is as follows:

- ◆ Developer creates an Activity with two Buttons
- ◆ When the Check Network Availability button is clicked the Network availability is checked using the ConnectivityManager API
- ◆ A Toast is displayed to the user with the Network state
- ◆ When the Download data button is clicked, the Remote server is contacted using an HttpClient Object
- ◆ The data transfer takes place using JSON
- ◆ The returned data is displayed to the user

© Aptech Ltd. Wireless and Networking/Session 11 17

Using slide 17, provide a brief description of the functional logic of the application created. Explain that the data transfer occurs using JSON.

Wi-Fi

Slide 18

Wi-Fi

- ◆ Wi-Fi is the most commonly used standard for wireless communication for high bandwidth transfer
- ◆ Wi-Fi runs in the same 2.4 Ghz spectrum as Bluetooth
- ◆ It has a reasonably long range with high transfer rates reaching upto 300 Mbps and higher
- ◆ It is also possible to share a mobile data connection with other devices using Android's 'tether' feature



© Aptech Ltd. Wireless and Networking/Session 11 18

Using slide 18, explain the features of Wi-Fi and its characteristics. Explain how it differs from short range low bandwidth standard such as Bluetooth. Explain the tether feature in Android and its applications.



In-Class Question: Which wireless standard shares the same spectrum with Wi-Fi?

Answer: Wi-Fi operates in the same spectrum as Bluetooth.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Wi-Fi Support in Android.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<http://developer.android.com/reference/android/net/wifi/package-summary.html>

Wi-Fi API Classes

Slide 19

Wi-Fi API Classes

- ◆ **WifiManager:**
 - ❖ This class is used to manage the Wi-Fi connectivity by invoking the method, Context.getSystemService(Context.WIFI_SERVICE)
 - ❖ The Wifi Manager can deal with the following:
 - ❖ List of configured networks
 - ❖ Currently available active networks
 - ❖ Provide enough information to the developer regarding which network one has to choose and connect to
- ◆ **ConnectivityManager:**
 - ❖ ConnectivityManager notifies the application whenever any network changes happen
 - ❖ Some of the tasks performed by the ConnectivityManager are as follows:
 - ❖ Scans Wi-Fi, GPRS and UMTS, and other network connections
 - ❖ Sends broadcast intents whenever the connectivity changes
 - ❖ Queries the state of available network using an API

Using slide 19, explain the Wi-Fi API. List the classes and explain their functionality using the bulleted points.

Wi-Fi Example Application

Slide 20

Wi-Fi Example Application

- Using the classes, an application for demonstrating Wi-Fi functionality is created as shown in the following figure:
- When the Turn Wifi On button is clicked, the output will be as shown in the following figure:

The slide contains two screenshots of an Android application named "WifiExample". The first screenshot shows a screen titled "Wifi Manager" with two buttons: "Turn Wifi On" and "Turn Wifi Off". The second screenshot shows the same screen after a button has been clicked, with the text "WIFI STATE ENABLING" displayed below the buttons.

Using slide 20, demonstrate the steps to create an application that controls Wi-Fi state of the device. The application allows the user to control the Wi-Fi connectivity of the device. Demonstrate and explain the functionality and output of the application using the figures given on the slide.



In-Class Question: Which class is used to manage Wi-Fi connectivity?

Answer: The WiFiManager class is used to manage Wi-Fi connectivity.

Wi-Fi Example Application Logic

Slide 21

Wi-Fi Example Application Logic

- The application logic is as follows:

- ◆ Developer creates an Activity with two Buttons
- ◆ When the Turn Wifi On button is clicked, Wi-Fi is enabled on the device
- ◆ The converse happens when the Turn Wifi Off button is clicked
- ◆ The status is displayed to the user

© Aptech Ltd. Wireless and Networking/Session 11 21

Using slide 21, provide a brief description of the functional logic of the application created above. Explain that the Wi-Fi manager class is used to switch ON/OFF the Wi-Fi adapter.

NFC

Slide 22

NFC

- ◆ NFC is the abbreviation for Near Field Communication
- ◆ It is a new wireless communication standard for short range communication
- ◆ It operates at 13.56 Mhz frequency and can achieve speeds up to 424 Kbit/s
- ◆ NFC devices are activated on proximity using a touch or tap gesture among the devices
- ◆ NFC primarily consists of three standards/utilities:
 - ◆ Card emulation mode
 - ◆ Read/Write mode
 - ◆ Peer-to-Peer mode



© Aptech Ltd. Wireless and Networking/Session 11 22

Using slide 22, explain the importance of NFC standard. Explain how it differs from Wi-Fi and how it adds non-data transfer features to the system. You may provide real life examples such as Google Wallet, NFC tags for products in super markets, and Android Beam.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding NFC Support in Android.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

<https://developer.android.com/guide/topics/connectivity/nfc/index.html>

NFC API

Slide 23

The screenshot shows a slide titled "NFC API". The content is organized into two main sections: "NDEF Content Type" and "Relevant Classes".

- ◆ **NDEF Content Type**
 - ◆ NDEF is the data format used by Android for NFC messages and records. The data formats are implemented in the classes NdefMessage and NdefRecord
 - ◆ An NDEF object can be retrieved using the get(Tag) method. The tag type needs to be mentioned before the NDEF object is retrieved. There are currently four supported tag types:
 - ◆ NFC_FORUM_TYPE_1
 - ◆ NFC_FORUM_TYPE_2
 - ◆ NFC_FORUM_TYPE_3
 - ◆ NFC_FORUM_TYPE_4
- ◆ **Relevant Classes**
 - ◆ This section describes the relevant classes needed to use the NFC API:
 - ◆ NdefMessage
 - ◆ NdefRecord
 - ◆ NfcAdapter
 - ◆ NfcEvent
 - ◆ NfcManager

Using slide 23, explain the NFC API. Explain the importance of the NDEF content type. NDEF is the encapsulating data type for all NFC messages and records. Use the accompanying text to help the students understand the concept. List the relevant classes, providing a brief description for each of them.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding NDEF Data type and associated classes.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/reference/android/nfc/tech/Ndef.html>

<http://developer.android.com/reference/android/nfc/NdefMessage.html>

<http://developer.android.com/reference/android/nfc/NdefRecord.html>

NFC Permissions

Slide 24

NFC Permissions

- ◆ The application needs to explicitly request permission to access NFC hardware
- ◆ The manifest code is shown in the following Code Snippet:

```
<uses-permission android:name="android.permission.NFC" />
```

Using slide 24, explain the permissions that need to be specified in the manifest file for the application to use NFC hardware.



In-Class Question: Expand the term NFC.

Answer: NFC is the abbreviation for Near Field Communication.

Verifying NFC Support

Slide 25

Verifying NFC Support

- ◆ It is a good practice to ensure that the device has the hardware and API support to perform NFC operation
- ◆ This can be done using one of the following methods:
 - ◆ Manifest Code
 - ◆ System Feature
 - ◆ API Support
- ◆ The code to check the System Feature and API Support is shown in the following Code Snippet:

```
//System Feature  
boolean systemSupport =  
getPackageManager.getSystemFeature(PackageManager.FEATURE_NFC);  
  
//API Support  
int level = Build.VERSION.SDK_INT;
```

Using slide 25, explain the importance of verifying the support for NFC hardware within the application. NFC is an emerging standard and its functionality can be easily substituted by other wireless standards. List the various methods that can be used for this purpose providing the merits and demerits for each of them. Explain the code given in the slide. The first line of code retrieves the system features to ensure that the device has NFC features enabled. The second line ensures that the API level is high enough to support NFC.

NFC Example Application

Slide 26

NFC Example Application

- Using the classes, an application for demonstrating NFC functionality is created as shown in the following figure:
- Once the START TRANSFER button is clicked, the transferred file can be viewed as shown in the following figure:

© Aptech Ltd. Wireless and Networking/Session 11 26

Using slide 26, demonstrate the steps to create an application that facilitates file transfer using NFC. The application implements NFC Beam API to transfer the image nfcimage.jpg to another device. Demonstrate and explain the functionality and output of the application using the figures given on the slide.

NFC Example Application Logic

Slide 27

NFC Example Application Logic

- The application logic is as follows:
 - Developer creates an Activity with a Button
 - When the START TRANSFER button is clicked, the file nfcimage.jpg is transferred from one device to another
 - This is done using the NFCAdapter's setBeamPushUris() method
 - The Transferred file can be viewed on the other device using any File Manager Application

© Aptech Ltd. Wireless and Networking/Session 11 27

Using slide 27, provide a brief description of the functional logic of the application. Explain that the application uses the NFC beam standard to transfer the nfcimage.jpg file to another device.

Summary

Slide 28

Summary

- ◆ Bluetooth is a network stack that transfers data between devices wirelessly. Android provides all the Bluetooth APIs' under android.bluetooth package
- ◆ Bluetooth Adapter plays the role of discovering all the Bluetooth enabled devices and querying for request
- ◆ Android applications performing network operations uses HTTP to send and receive data
- ◆ HttpClient can be used to send and receive data from the server by creating a DefaultHttpClient() which would help in sending and receiving of data
- ◆ Android provides Wi-Fi APIs, through which applications can communicate with the wireless stack that provides Wi-Fi network access
- ◆ NFC is a short range wireless data transfer protocol
- ◆ The API for NFC is encapsulated in the android.nfc package

Using slide 28, summarize the session. Make them revise the following points:

- Bluetooth is a network stack that transfers data between devices wirelessly. Android provides all the Bluetooth APIs' under android.bluetooth package.
- Bluetooth Adapter plays the role of discovering all the Bluetooth enabled devices and querying for request.
- Android applications performing network operations uses HTTP to send and receive data.
- HttpClient can be used to send and receive data from the server by creating a DefaultHttpClient() which would help in sending and receiving of data.
- Android provides Wi-Fi APIs, through which applications can communicate with the wireless stack that provides Wi-Fi network access.
- NFC is a short range wireless data transfer protocol.
- The API for NFC is encapsulated in the android.nfc package.

11.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes Telephony, SMS, and VoIP.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 12: Telephony, SMS, and VoIP

12.1 Pre-Class Activities

You should familiarize yourself with core components of the telephony framework. You should also have a good understanding of concepts of VoIP. You should review the procedure to send and receive SMS from the device.

Familiarize yourself with the topics of the current session in-depth.

12.1.1 Objectives

After the session, learners will be able to:

- Explain Telephony
- Explain Telephony Manager
- Explain SMS
- Explain the process of sending SMS messages programmatically from within your application
- Explain the process of receiving incoming SMS messages
- Explain SIP and VoIP
- Use the SIP Framework to make a VoIP application

12.1.2 Teaching Skills

You should be familiar with the telephony framework in Android. You should be able to identify the important classes of the framework. You should also have the knowledge to programmatically send and receive SMS. You should be familiar with SIP framework and the functionality of a VoIP application.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

12.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Tell the students that the previous session explained the basics of Wireless and Networking support in Android including Bluetooth, Wi-Fi, and NFC.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the basics of the Telephony framework in Android and its functionality. The session also explains the process of sending and receiving messages and SIP framework for VoIP applications.

12.2 In-Class Explanations

Introduction

Slide 3

Introduction

- ◆ Communication is essential to interactive applications
- ◆ Android provides easy-to-use API for SMS, SIP, and Telephony
- ◆ The RIL is the core component of the Telephony framework
- ◆ SIP is the standard framework for developing VoIP Applications



© Aptech Ltd.

Telephony, SMS, and VoIP/Session 12

3

Using slide 3, introduce the telephony framework and its importance. Explain that Android provides easy-to-use API for SMS, SIP, and Telephony.

Radio Hardware

Slide 4

Radio Hardware

- ◆ Android devices supporting telephony services have a modem installed within them either as a separate module or as part of System on Chip (SoC)
- ◆ Android always views the modem as an autonomous device
- ◆ The communication with the modem is done with a serial port or an emulated serial port using Hayes AT Command Set
- ◆ Voice implementation varies from one modem to another



© Aptech Ltd.

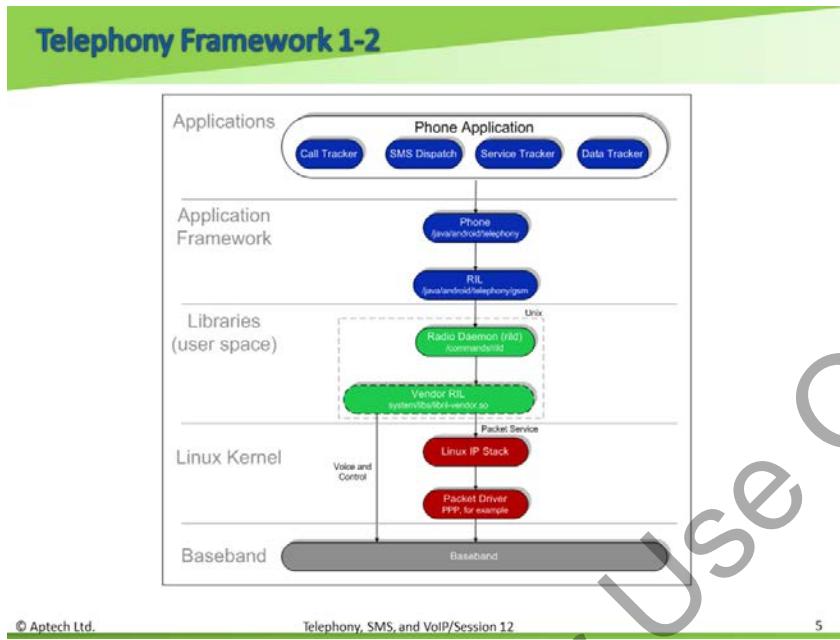
Telephony, SMS, and VoIP/Session 12

4

Using slide 4, explain the physical hardware implementation behind telephony support in Android. Explain that Android views the Radio hardware as a separate entity irrespective of its implementation or as part of the SoC (System on Chip). Explain that Hayes AT commands are used to communicate with the modem.

Telephony Framework

Slides 5 and 6



© Aptech Ltd.

Telephony, SMS, and VoIP/Session 12

5

Using slide 5, explain the telephony framework structure. Explain that the framework is implemented in layers with each layer providing functionality to the higher layers. Mention that the top two layers that is, Application and Application Framework are provided by Android and do not need to be implemented by the vendor. The bottom three layers need to be implemented by the vendor and they are device specific.

Telephony Framework 2-2

- ◆ The telephony framework provides services to the telephony application similar to the services provided by the Wi-Fi Manager
- ◆ This is the API that is exposed to the developer for creating applications
- ◆ The API is same across all the devices irrespective of the modem type or the AT commands supported by the device



© Aptech Ltd.

Telephony, SMS, and VoIP/Session 12

6

Using slide 6, explain the basics of telephony framework. Explain that this is the second layer in the implementation. It is responsible for providing services to the application layer.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding the Telephony Architecture.

(OR)

You can refer to the online Mirror of the same documentation using the following link:

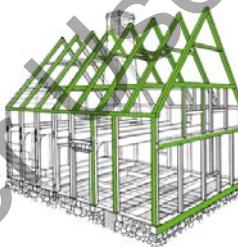
<http://www.netmite.com/android/mydroid/development/pdk/docs/telephony.html>

RIL Layer

Slide 7

RIL Layer

- ◆ The telephony manager uses the API provided by the RIL Layer
- ◆ The RIL layer is where the AT commands and the modem interaction is executed
- ◆ It consists of the RIL Daemon which listens for requests and the RIL library which has the implementation to handle the requests
- ◆ The RIL implementation varies from one device to another
- ◆ The RIL layer translates the application requests into AT commands understandable by the modem



Using slide 7, explain the RIL layer of the framework. Explain that it acts as the interface between the application framework and the Linux Kernel. Explain that it translates the requests from the higher layers into AT commands understandable by the modem.

Additional Reference:

You may refer to the following links for more information regarding Hayes command set:

https://en.wikipedia.org/wiki/Hayes_command_set

<http://home.intekom.com/option/hayesat.htm>

Linux Kernel Stack

Slide 8

Linux Kernel Stack

- ◆ The Kernel stack is responsible for the drivers which support the modem device
- ◆ It is also responsible to provide a channel of communication for the RIL layer to send the AT commands
- ◆ In modern devices with SoCs, the Kernel stack is also responsible for initializing the modem hardware on startup

© Aptech Ltd. Telephony, SMS, and VoIP/Session 12 8



Using slide 8, explain the importance of the Linux Kernel Stack. The Linux Kernel Stack establishes a connection with the modem and transports the AT commands from the RIL layer to the modem.



In-Class Question: What is the functionality of the RIL layer?

Answer: The RIL layer translates commands from higher layers to AT commands understandable by the modem.

Flow of Commands

Slide 9

Flow of Commands

- ◆ A request is initiated from the caller application or the SMS application or a custom application using the telephony manager
- ◆ The request is received by the telephony framework. The telephony framework then creates a request for the RIL layer using a JNI interface
- ◆ RIL layer resolves the type of the request and executes the necessary Hayes commands
- ◆ The commands are sent over a serial port or an emulated port
- ◆ The Kernel layer translates the data from the emulated channel into the mode of communication used by the Modem (such as USB)

© Aptech Ltd. Telephony, SMS, and VoIP/Session 12 9

Using slide 9, explain the flow of commands in the Android System. Explain how the interaction transfers from one layer to another and the function of each layer in the process.

Telephony

Slide 10

Telephony

- ◆ Applications can use the methods present in this class to determine the telephony services and states, as well as have access to some types of subscriber information
- ◆ Applications can also register to a listener to receive notification of telephony state changes
- ◆ Apart from this, the telephony API is also used for activities such as monitoring phone information including the current states of the phone, connections, and network



© Aptech Ltd.

Telephony, SMS, and VoIP/Session 12

10

Using slide 10, explain the importance of the telephony framework in Android from a developer's perspective. Explain that custom dialer application can be developed which can replace the Android's default 'Phone' application.

Telephony Manager Example

Slides 11 and 12

Telephony Manager Example 1-2

- ◆ It provides information about the available telephony services and states on the device
- ◆ The application should have the required permission specified in the AndroidManifest.xml
- ◆ The code to create a Phone State Listener is shown in the following Code Snippet:

```
...
TelephonyManager telephonyManager;
...
telephonyManager = (TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE);

phoneStateListener = new PhoneStateListener() {
    @Override
    public void onCallStateChanged(int state, String
        incomingNumber) {
    }
}
telephonyManager.listen(phoneStateListener, PhoneStateListener.LISTEN_CALL_STATE);
```

© Aptech Ltd.

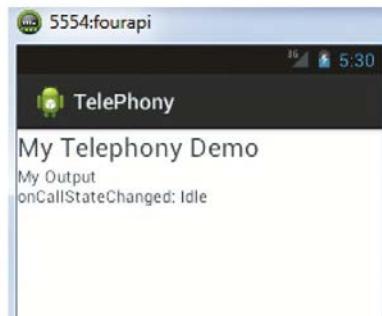
Telephony, SMS, and VoIP/Session 12

11

Using slide 11, explain the use of TelephonyManager class. Explain that it is the core component of the telephony framework and it is used to register listeners for events.

Telephony Manager Example 2-2

- Using the code, an application for demonstrating Telephony Manager functionality is created as shown in the following figure:



© Aptech Ltd.

Telephony, SMS, and VoIP/Session 12

12

Using slide 12, demonstrate the steps to create an application using the telephony framework. The application monitors for changes in the network status and updates them on the Main Activity. Explain the functional logic of the application that is, it uses a listener for the call status and displays it on the main activity. Explain the output of the application using the figure given on the slide.

SMS

Slide 13

SMS

- ◆ SMS stands for Short Messaging Service
- ◆ SMS is one of the most important and frequently used applications that are executed on mobile devices
- ◆ Android comes with the built-in SMS application that enables you to send and receive SMS messages
- ◆ The SMSManager manages the SMS operations, such as sending text and data messages



© Aptech Ltd.

Telephony, SMS, and VoIP/Session 12

13

Using slide 13, explain the concept of SMS. Explain that SMS is the abbreviation for Short Messaging Service. Explain that SMS functionality is built in to the Android eco system. However, the developers can replace it with substitute applications having improved functionality.

Sending SMS Example

Slides 14 and 15

Sending SMS Example 1-2

- The application can send SMS to another phone, when an event takes place
- In an Android application, the developer can access the SMS application using the SmsManager
- The sendTextMessage() function is used to send an SMS message as shown in the following Code Snippet:

```
...
SmsManager smsMgr = SmsManager.getDefault();
smsMgr.sendTextMessage("phone_num", null, "msg_text", null, null);
...

/*
public void sendTextMessage (String destinationAddress, String scAddress, String
text, PendingIntent sentIntent, PendingIntent deliveryIntent);
*/
```

© Aptech Ltd. Telephony, SMS, and VoIP/Session 12 14

Using slide 14, explain the process of sending an SMS from the device programmatically. Explain the code on the slide. Explain the arguments accepted by the sendTextMessage() method. Mention that the last two arguments are optional and they are used to receive updates on the delivery status.

Sending SMS Example 2-2

- Using the code, an application for demonstrating SMS functionality is created as shown in the following figure:
- When the SMS is sent and received by another device, the output will be as shown in the following figure:

The first screenshot shows the 'SMSApp' application's main screen with a 'Send SMS' button. The second screenshot shows the device's home screen with a received SMS from '1 555-521-5554' containing the text 'Hello World!!' twice.

© Aptech Ltd. Telephony, SMS, and VoIP/Session 12 15

Using slide 15, demonstrate the steps for creating an application capable of sending SMS. Mention that this application will send an SMS with the contents 'Hello World!'. Explain the application logic and the output of the application using the figures given on the slide.

Receiving SMS Example

Slides 16 and 17

Receiving SMS Example 1-2

- The developer can also receive incoming SMS messages from within the application using BroadCastReceiver object
- The code for this is shown in the following Code Snippet:

```

<receiver android:name=".SMSReceiver" >
<intent-filter>
<action android:name="android.provider.Telephony.SMS_RECEIVED" />
</intent-filter>
</receiver>
""

public class SMSReceiver extends BroadcastReceiver {
@Override
public void onReceive(Context , Intent intent) {
}
}

```

© Aptech Ltd.  Telephony, SMS, and VoIP/Session 12 16

Using slide 16, explain the process of receiving SMS Notifications. Explain the code on the slide and the use of intent-filter for receiving updates on SMS status. The notifications are received by the use of a Broadcast Receiver.

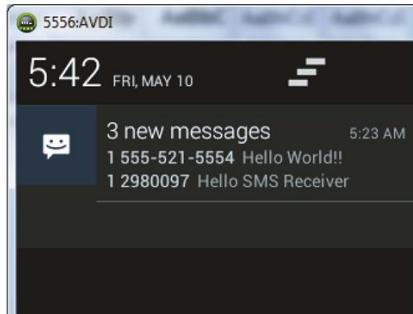
Additional Reference:

You may refer to the following link for more information regarding sending and receiving SMS:

<http://codetheory.in/android-sms/>

Receiving SMS Example 2-2

- Using the code, an application for demonstrating receiving SMS is created as shown in the following figure:



- The application creates a notification whenever an SMS is received

© Aptech Ltd.  Telephony, SMS, and VoIP/Session 12 17

Using slide 17, demonstrate the steps for creating an application capable of receiving SMS. Explain that this application will receive SMS updates using a broadcast receiver and display a notification each time this happens.

Explain the application logic and the output of the application using the figure given on the slide.



In-Class Question: Which intent filter is used to receive SMS updates?

Answer: The android.provider.Telephony.SMS_RECEIVED intent filter is used for receiving SMS updates.

Sending SMS and Tracking It

Slide 18

Sending SMS and Tracking it

- ◆ For tracking SMS, use broadcast receiver which checks the status of the message
- ◆ You have to develop your own BroadcastReceiver and pass the string using an object of the PendingIntent class
- ◆ Also, ContentObserver listening on the content://sms to track SMS
- ◆ This class will invoke the onChange() method when the SMS database changed



Using slide 18, explain the process of tracking a sent SMS. Explain that the delivery status can be received by passing the optional PendingIntent argument and broadcast receiver.

VoIP and SIP

Slide 19

VoIP and SIP

- ◆ **VoIP**
 - ❖ VoIP is the abbreviation for Voice over Internet Protocol
 - ❖ Transfer of voice over network has proven to be cheaper than the traditional telephone networks
 - ❖ Most providers have a free service for VoIP as long as you have a data connection to their servers along with Mobile Clients
 - ❖ These clients, in most cases, utilize the SIP protocol for signaling and data transfer
- ◆ **SIP**
 - ❖ SIP is the abbreviation for Session Initiation Protocol
 - ❖ SIP uses message based stateless protocol similar to HTTP and SMTP for session control
 - ❖ Streaming protocols such as Real-time Streaming Protocol (RTP) and Secure Real-time Streaming Protocol (SRTP) are used for data transfer such as voice and video
 - ❖ On a network level, SIP can utilize TCP, UDP, or SCTP



© Aptech Ltd. Telephony, SMS, and VoIP/Session 12 19

Using slide 19, explain the concepts of SIP and VoIP. Provide abbreviations for both the terms and explain the difference between them. Explain that SIP is the most commonly used protocol for implementing VoIP.

Additional Reference:

You may refer the following links for more information regarding SIP and VoIP:

https://en.wikipedia.org/wiki/Session_Initiation_Protocol

https://en.wikipedia.org/wiki/Voice_over_IP

SIP Architecture

Slide 20

SIP Architecture

- ◆ The SIP model is similar to the SMTP mail transfer architecture
- ◆ It consists of:
 - ❖ SIP User Agent Client
 - ❖ SIP User Agent Server
 - ❖ Proxy Servers
 - ❖ Redirection Servers
 - ❖ Gateways
 - ❖ Registrars



© Aptech Ltd. Telephony, SMS, and VoIP/Session 12 20

Using slide 20, explain the architecture of the SIP framework. If the students are familiar

with the architecture of SMTP, you can provide that as an example. Explain that SIP has architecture of interconnected nodes. List the components of the architecture and explain them. For example, SIP User Agent Client is the client application and SIP User Agent Server receives information from SIP User Agent client. It transfers the information to the proxy/redirection server based on the architecture, and so on.



In-Class Question: Expand the terms SIP and VoIP.

Answer: SIP is the abbreviation for Session Initiation Protocol and VoIP is the abbreviation for Voice over Internet Protocol.

SIP Requests

Slides 21 and 22

SIP Requests 1-2

- SIP uses commands to generate requests
- SIP requests are listed in the following table:

Request	Description
INVITE	Indicates a client is being invited to participate in a call session
ACK	Confirms that the client has received a final response to an INVITE request
BYE	Terminates a call and can be sent by either the caller or the callee
CANCEL	Cancels any pending request
OPTIONS	Queries the capabilities of servers
REGISTER	Registers the address listed in the To header field with a SIP server
PRACK	Provisional acknowledgement

SIP Requests 2-2

Request	Description
SUBSCRIBE	Subscribes for an Event of Notification from the Notifier
NOTIFY	Notify the subscriber of a new Event
PUBLISH	Publishes an event to the Server
INFO	Sends mid-session information that does not modify the session state
REFER	Asks recipient to issue SIP request (call transfer)
MESSAGE	Transports instant messages using SIP
UPDATE	Modifies the state of a session without changing the state of the dialog

Using slides 21 and 22, explain the requests supported by the SIP protocol. Explain that SIP uses request response model. List the request and provide a description using the table given in the slides.



In-Class Question: What are the contents of the SIP protocol architecture?

Answer: The contents are:

- SIP User Agent Client
- SIP User Agent Server
- Proxy Servers
- Redirection Servers
- Gateways
- Registrars

SIP Responses

Slide 23

SIP Responses

- SIP uses numeric response codes as replies similar to the response codes used in HTTP
- The code families are in the following table:

Response	Description
1xx	Status Response codes
2xx	Success Response code
3xx	Redirection Response code
4xx	Client Side Error
5xx	Server Side Error
6xx	Protocol not supported
1xx	Status Response codes

© Aptech Ltd. Telephony, SMS, and VoIP/Session 12 23

Using slide 23, explain the response codes supported by SIP. Explain that SIP uses numerical response codes similar to http. List them and provide a description for each of them using the table given on the slide.

SIP Framework

Slide 24

SIP Framework

- ◆ Relevant Classes
 - ◆ **SipAudioCall:** This class is used to handle a voice call
 - ◆ **SipSession:** This class is used to handle a SIP session
 - ◆ **SipErrorCode:** This class is used to contain the error data
 - ◆ **SipProfile:** This class holds connection information such as the server, user name, and password for the account
 - ◆ **SipManager:** This class is responsible for establishing connection and provides access to SIP services
- ◆ Support Classes
 - ◆ **SipAudioCall.Listener:** The listener class for and SipAudioCall. The methods for handling on incoming and on outgoing call events can be handled here. These events are also received by the SipSession.Listener class
 - ◆ **SipSession.Listener:** The listener class for an SIP session
 - ◆ **SipProfile.Builder:** The builder class for creating SipProfile objects
 - ◆ **SipSession.State:** An encapsulation class holding integer values for call states

Using slide 24, explain the core components of the SIP framework. Explain the important classes along with their purpose using the bulleted points.

Explain that the core classes are sufficient for SIP session management and the support classes can help with code modularization.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding SIP classes.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/guide/topics/connectivity/sip.html>

<http://developer.android.com/reference/android/net/sip/SipManager.html>

<http://developer.android.com/reference/android/net/sip/SipProfile.html>

<http://developer.android.com/reference/android/net/sip/SipAudioCall.html>

<http://developer.android.com/reference/android/net/sip/SipAudioCall.Listener.html>

Permission and Perquisites

Slide 25

Permission and Perquisites

- ◆ Permissions
 - ◆ In order to utilize SIP, the INTERNET and the USE_SIP permissions are mandatory
 - ◆ Both these permissions need to be acquired by the application using the Manifest File
- ◆ Pre-requisites
 - ◆ A device running Android Lollipop with a data connection. The SIP application needs to be deployed on a real device as AVD does not allow network connections
 - ◆ A registered SIP account for use with the application
 - ◆ Access to a SIP server

Using slide 25, explain the permissions that are required by an application to use the SIP framework. Explain the pre-requisites for creating an application using SIP. Explain that although there are SIP access providers, it usually comes with a financial burden for the developer. Explain that it is therefore better to have dedicated SIP servers for the application.

SIP Login**Slide 26****SIP Login**

- Following Code Snippet demonstrates an example for logging into an SIP Server:

```

SipProfile.Builder builder = null;
try {
    builder = new SipProfile.Builder(userName, domain);
    builder.setPassword(passWord);
    builder.setAutoRegistration(true);
    profile = builder.build();

    // If AutoRegistration is set to false, uncomment the //below line of
    code
    //sipManager.register(profile, 9999, listener);

    sipManager.open(profile, pendingIntent, listener);
}

```

Using slide 26, explain the process to provide login functionality to the application. Explain the code given on the slide. An SIP profile is created which contains all the information regarding the connection. The session can be logged in by using the open() method of SipManager.

SIP Incoming Call**Slide 27****SIP Incoming Call**

- Following Code Snippet demonstrates an example for receiving updates for incoming SIP calls:

```

receiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context , Intent intent) {
        setStatus(" Incoming Call ");
    }
};

filter = new IntentFilter();
filter.addAction("sipApp.intent.IncomingCall");

this.registerReceiver(receiver, filter);
}

```

Using slide 27, explain the process to receive notifications on incoming calls. Explain the code given on the slide. Mention that notifications of change in SIP call states can be received by the use of Intent filters and BroadcastReceivers.

SIP Example Application

Slides 28 and 29

SIP Example Application 1-2

- Using the code, an application for demonstrating SIP framework is created as shown in the following figure:
- Once the connect button is clicked, the application registers with the SIP registrar the status is changed to connected as shown in the following figure:

© Aptech Ltd. Telephony, SMS, and VoIP/Session 12 28

SIP Example Application 2-2

- If the account receives an incoming call, the status is set to Incoming call as shown in the following figure:



Using slide 28, demonstrate the steps for creating an application using the SIP framework. Explain that this application will allow Login functionality to any SIP server that the user chooses. The status notifications are also displayed at the top of the application. Using slide 29, explain the application logic and how it utilizes the code explained in the earlier slides.

Alternatives to SIP

Slide 30

Alternatives to SIP

- ◆ Alternatives to SIP protocols are available such as Extensible Messaging and Presence Protocol (XMPP) and Standard Interface for Multiple Platform Link Evaluation (SIMPLE)
- ◆ XMPP is an xml based instant messaging protocol that also supports voice and video data
- ◆ SIMPLE is an emerging standard closely based on the SIP standard
- ◆ Neither of these two protocols is natively supported by Android
- ◆ However, third party libraries are available to make the process of development much easier
- ◆ XMPP, in particular, has gained a huge market share

© Aptech Ltd.

Telephony, SMS, and VoIP/Session 12

30

Using slide 30, explain various alternatives available to SIP. Explain that even though some of these protocols have advantages over SIP, SIP's universal support across all platforms makes it a much more viable option.

Summary

Slide 31

Summary

- ◆ Telephony provides access to information about the telephony services on the devices
- ◆ The Android TelephonyManager provides information about the Android telephony system. In other words, it provides information about the available telephony services and states on the device
- ◆ SMS stands for Short Messaging Service. SMS messaging is one of the most important and frequently used applications that is executed on the mobile devices
- ◆ The SmsManager manages the SMS operations, such as sending text and data messages. Similarly, ConnectivityManager manages the Internet connectivity in a mobile phone
- ◆ Apart from sending SMS messages from Android applications, the developer can also receive incoming SMS messages from within the application using BroadcastReceiver object
- ◆ VoIP is the process of utilizing network access to provide telephony services such as voice and video
- ◆ SIP is the protocol natively supported by Android for VoIP and similar applications

© Aptech Ltd.

Telephony, SMS, and VoIP/Session 12

31

Using slide 31, summarize the session. Make them revise the following points:

- Telephony provides access to information about the telephony services on the devices.
- The Android TelephonyManager provides information about the Android telephony system. In other words, it provides information about the available telephony services and states on the device.

- SMS stands for Short Messaging Service. SMS messaging is one of the most important and frequently used applications that is executed on the mobile devices.
- The SmsManager manages the SMS operations, such as sending text and data messages. Similarly, ConnectivityManager manages the Internet connectivity in a mobile phone.
- Apart from sending SMS messages from Android applications, the developer can also receive incoming SMS messages from within the application using BroadCastReceiver object.
- VoIP is the process of utilizing network access to provide telephony services such as voice and video.
- SIP is the protocol natively supported by Android for VoIP and similar applications.

12.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes UI creating Activities, View, Styles, Themes, and handling User Input.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 13: Sensors

13.1 Pre-Class Activities

You should familiarize yourself with the concepts of Sensors and Android's Sensor framework. You should acquire a good understanding on types of sensors and the standard API to access them. You should also revise the methods to extract meta-data from sensors. This data can be derived from processing of raw data.

Familiarize yourself with the topics of the current session in-depth.

13.1.1 Objectives

After the session, learners will be able to:

- Explain various types of Sensors
- Use Motion Sensors
- Use Position Sensors
- Use Status Sensors
- Explain Context aware services based on sensor information

13.1.2 Teaching Skills

You should be familiar with Sensors in Android. You should be able to identify the types of sensors supported by Android and how to retrieve the information from each of them. You should also be able to create applications that rely upon sensors for real world applications.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

13.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Tell the students that the previous session explained the basics of Telephony Framework in Android, sending and receiving SMS, and VoIP.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains the basics of the sensor support in Android and the API to access information from them.

13.2 In-Class Explanations

Introduction

Slide 3

Introduction

- ◆ Sensors are hardware devices that record and measure physical data
- ◆ Sensors in Android play a key role in the core operating system as well as the platform as a whole
- ◆ Sensors in Android can be broadly classified into following three types:
 - ◆ Motion Sensors
 - ◆ Environmental Sensors
 - ◆ Position Sensors



© Aptech Ltd. Sensors/Session 15 3

Using slide 3, introduce the student to the concept of sensors. Define sensors and explain their role in the Android system. Sensors make the entire OS more fluid and 'natural'. Give an example for applications utilizing sensors such as games and multimedia applications.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding sensor framework in Android.

(OR)

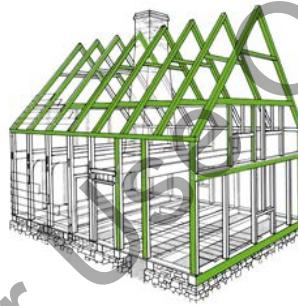
You can refer to the online Mirror of the same documentation using the following link:
http://developer.android.com/guide/topics/sensors/sensors_overview.html

Sensor Framework

Slide 4

Sensor Framework

- ◆ Sensor Framework is responsible for translating the input received from the hardware devices to readings usable by the applications
- ◆ Raw data received from the sensors can also be accessed using the Sensor Framework
- ◆ Certain sensors can be implemented by software emulation using data received from other sensors



© Aptech Ltd.

Sensors/Session 13

4

Using slide 4, explain the Sensor framework in Android. Explain that all sensors have a unified API which can be accessed by using the Sensor framework. Mention that sensor data is usually calibrated, although raw reading can be read.



In-Class Question: What are Sensors?

Answer: Sensors are hardware devices used to measure physical data.

Sensor API

Slides 5 and 6

Sensor API 1-2

◆ SensorManager

- ❖ This class is the central component of the sensor framework
- ❖ It is responsible for retrieving sensor instances and setting event listeners for the sensor
- ❖ It is also responsible for identifying the sensor support on the device

◆ Sensor

- ❖ This class is used to reference a specific sensor on the device
- ❖ An instance of a sensor class can be retrieved by using the getDefaultSensor(int type) or the getSensorList(int type) method of SensorManager

Sensor API 2-2

◆ SensorEvent

- ❖ This class stores the event information whenever there is a change in status of the sensor

◆ SensorEventListener

- ❖ This class provides an interface to implement event listeners for the sensors
- ❖ The methods onSensorChanged() and onAccuracyChanged() need to be implemented

Using slides 5 and 6, explain the important classes of the Sensor framework.

Explain the uses of each of the classes using the bulleted points. Mention that the SensorEvent class is used to hold sensor related information and the SensorEventListener is used to register a listener for specific sensors.

Additional Reference:

You may refer the Android SDK documentation available with the SDK Download for more information regarding Sensor API classes.

(OR)

You can refer to the online Mirror of the same documentation using the following links:

<http://developer.android.com/reference/android/hardware/Sensor.html>

<http://developer.android.com/reference/android/hardware/SensorEvent.html>

<http://developer.android.com/reference/android/hardware/SensorManager.html>

<http://developer.android.com/reference/android/hardware/SensorEventListener.html>

Working with Sensor Data

Slide 7

Working with Sensor Data

- ◆ Obtain a reference to the SensorManager instance
- ◆ Ensure that the sensor type is supported by the device
- ◆ Get a reference to the specific type of the sensor
- ◆ Create an event listener for the sensor(s) by implementing the SensorEventListener interface
- ◆ Implement the onSensorChanged() method to handle changes in sensor data
- ◆ Implement the onAccuracyChanged() method to handle accuracy changes
- ◆ Register the listener with the sensor manager for the type of the sensor
- ◆ Implement the onPause() method of the activity and unregister the listener
- ◆ Implement the onResume() method of the activity to re-register the listener. This saves CPU cycles that are wasted unnecessarily in processing the sensor changes

Using slide 7, explain the steps to register a listener and read sensor related data. Explain that the SensorManager class is the core of the Sensor framework and it is used to get a reference to a specific sensor. Explain the methods that need to be implemented and their purpose.



In-Class Question: Which class holds sensor data?

Answer: The SensorEvent class holds the sensor information.

Working with Sensor Data Example

Slides 8 and 9

Working with Sensor Data Example 1-2

- Following Code Snippet demonstrates reading Sensor Data:

```
public class MainActivity extends Activity {

    SensorManager sm;

    private SensorEventListener listener = new SensorEventListener() {

        @Override
        public void onSensorChanged(SensorEvent event) { ... }

        @Override
        public void onAccuracyChanged(Sensor sensor, int accuracy) {
            // Code here
        }
    };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
}
```

Working with Sensor Data Example 2-2

- It is strongly recommended to unregister the listener when the activity is paused as shown in the following Code Snippet:

```
...
@Override
protected void onResume() {
    super.onResume();

    sensorManager.registerListener(listener, sensorManager.getDefaultSensor(Sensor.
TYPE_PROXIMITY), SensorManager.SENSOR_DELAY_NORMAL);
}

@Override
protected void onPause() {
    super.onPause();
    sensorManager.unregisterListener(listener);
}
...
```

Using slides 8 and 9, explain the process of reading data from Sensors. Explain the code given in the slide. A SensorEventListener is created with the required methods implemented. The listener is registered to monitor changes on the Proximity sensor. Mention that the listener is unregistered when the Activity is paused, to save CPU cycles.

Reading Sensor Event Data

Slide 10

Reading Sensor Event Data

- ◆ The `onSensorChanged(SensorEvent event)` and `onAccuracyChanged(SensorEvent event)` methods are passed an event object reference
- ◆ The event float matrix holds the sensor reading values. The values represent the following data:
 - ◆ For most sensors, the values of the indices 0, 1, and 2 measure the values along the X, Y, and Z axis respectively
 - ◆ Uncalibrated sensors have three additional fields in the indices 3, 4, and 5 which measure the drift values along the X, Y, and Z axis respectively
 - ◆ Environment sensors have only one value in the index 0, which measures the reading of the sensor

© Aptech Ltd.

Sensors/Session 13

10

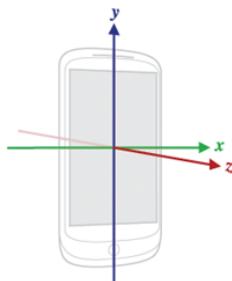
Using slide 10, explain the process to extract sensor data from the `SensorEvent` object. Explain that the values represent different information for different types of Sensors.

Motion Sensors

Slide 11

Motion Sensors

- ◆ Motion Sensors help monitor the movement of the device
- ◆ Some motion sensors are hardware only and the remaining may be implemented in software
- ◆ Motion sensors are primarily used in games and gimmicky multimedia applications



© Aptech Ltd.

Sensors/Session 13

11

Using slide 11, explain Motion Sensors and their uses. Explain that they are used to read the movement data. Explain their uses and give an example such as Asphalt.



In-Class Question: Which methods are invoked when there is a change in Sensor information?

Answer: The `onSensorChanged()` and `onAccuracyChanged()` methods are invoked whenever there is change in sensor information.

Motion Sensor Types

Slide 12

Motion Sensor Types		
Sensor Type	Implementation	Description
TYPE_ACCELEROMETER	Hardware	Accelerometer measures the acceleration or movement of the device in a direction
TYPE_GRAVITY	Hardware or Software	Gravity Sensor measure the gravitation force experienced by the device
TYPE_GYROSCOPE	Hardware	Gyroscope measures the rotational force of the device in a direction
TYPE_GYROSCOPE_UNCALIBRATED	Software	The software sensor provides raw uncalibrated data from the Gyroscope
TYPE_LINEAR_ACCELERATION	Hardware or Software	Linear Acceleration scope provides the acceleration force experienced by the device without the Gravitation force
TYPE_ROTATION_VECTOR	Hardware or Software	Provides the angle and inclination of the device
TYPE_STEP_COUNTER	Hardware or Software	Calculates the number of steps measured by the step sensor since the time of activation

© Aptech Ltd.

Sensors/Session 13

12

Using slide 12, explain the various motion sensors supported by Android. List the sensors and explain their purpose. Mention that some sensors can only be implanted in hardware, whereas other can be emulated in software as well. Android System does not interfere with the particulars of the implementation as long as it adheres to the standards.

Motion Sensor Example Application

Slide 13

Motion Sensor Example Application

- Using the code for Sensor Framework, an application for demonstrating Motion Sensors is created as shown in the following figure:
- Select a sensor and move the device in any direction to view the data as shown in the following figure:

© Aptech Ltd.

Sensors/Session 13

13

Using slide 13, explain the steps to create an application that utilizes motion sensors. The application lists the motion sensors and displays the sensor related information for the selected sensor.

Detecting Shake Motion

Slide 14

Detecting Shake Motion

- ◆ A shake or jerk motion has been staple to mobile phone even before Android came into existence
- ◆ A shake motion is used to signify a 'Next' gesture
- ◆ **Reading Sensor Input**
 - ◆ The main sensor that is responsible for detecting a shake motion is the accelerometer
 - ◆ The Linear Acceleration sensor is the ideal sensor for this purpose, but not all devices have this sensor
 - ◆ Hence, the accelerometer will be used as it is guaranteed to be supported on all devices
 - ◆ A shake motion in terms of sensor reading is characterized by a sudden increase in the acceleration experienced by the device in a short period of time

© Aptech Ltd.

Sensors/Session 13

14

Using slide 14, explain how motion sensors can be used to detect a shake motion in Android. Explain the uses of the shake motion and provide an example such as Music

Players. Explain the steps to detect the shake motion.

Steps to Detect Shake Motion

Slide 15

Steps to Detect Shake Motion

- ◆ Create a SensorEventListener object for receiving changes in the accelerometer readings
- ◆ Save the initial time of measuring the acceleration
- ◆ Calculate the Root Mean Square (RMS) values of the initial acceleration of the device, that is, Square root of $X^2+Y^2+Z^2$
- ◆ Store the initial acceleration value
- ◆ Whenever new acceleration values are experienced, check the time difference from the time of the initial measurement and time of changed values
- ◆ If the time difference is greater than a threshold (in this case, 300 milliseconds), then calculate the new RMS acceleration value
- ◆ Calculate the change in acceleration using the RMS value
- ◆ If the change in acceleration is greater than a threshold (in this case, 6), execute the code to handle a shake event. This implies a rapid change in acceleration in a short period time

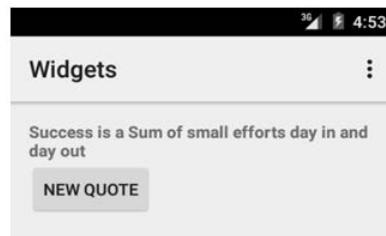
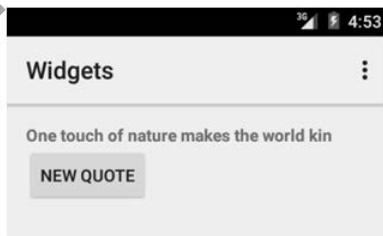
Using slide 15, explain the procedure for detecting the shake motion. Explain the theory behind the process using the bulleted points.

Shake Motion Example Application

Slide 16

Shake Motion Example Application

- Using the logic, an application for demonstrating Shake Motion is created as shown in the following figure:
- By shaking the device, a new quote is displayed as shown in the following figure:



Using slide 16, demonstrate the steps to create an application that uses the shake motion.

The application is a simple quote display application which detects a shake motion and displays the next quote when detected. Explain the output of the application using the figures given on the slide.



In-Class Question: List the types of sensors supported by Android.

Answer: The different types of sensors supported by Android are:

- Motion Sensors
- Position Sensors
- Environment Sensors

Shake Motion Example Application Logic

Slide 17

Shake Motion Example Application Logic

- The application logic is as follows:
 - ◆ Developer creates an Activity which displays a random Quote to the user
 - ◆ The Quote is generated using the QuoteGenerator class which contains a list of Quotes of which one is returned randomly
 - ◆ The app detects a shake motion using the change in acceleration using the accelerometer
 - ◆ If a shake motion is detected, a new Quote is retrieved and displayed to the user
 - ◆ A new quote is also displayed by clicking the New Quote button

Using slide 17, explain the application logic. The application uses the theory described earlier to detect a shake motion and displays a new quote to the user.

Position Sensors

Slide 18

Position Sensors

- ◆ Position Sensors help monitor the position and orientation of the device with respect to its surroundings
- ◆ The Geomagnetic Sensor and the Proximity Sensor are implemented in almost all Android Devices



© Aptech Ltd. Sensors/Session 13 18

Using slide 18, explain position sensors. Explain that they are used to read position related information of the device with respect to its surroundings. Provide a couple of examples.

Position Sensor Types

Slide 19

Position Sensor Types

- Following table lists the types of position sensors supported by Android along with their implementation option and utility:

Sensor Type	Implementation	Description
TYPE_GAME_ROTATION_VECTOR	Software	A specialized rotation vector that is useful for developing games. It is the rotational vector value excluding the geomagnetic readings. This sensor is used to implement 'Steering Wheel' or 'Tilt' functionality in games
TYPE_GEOmAGNETIC_ROTATION_VECTOR	Hardware or Software	It provides a rotation angle and inclination information of the device similar to the rotation vector. The key difference being that it uses the Geomagnetic Sensor instead of the Gyroscope
TYPE_MAGNETIC_FIELD	Hardware or Software	Geomagnetic Sensor helps monitor changes in the Earth's magnetic field. The readings are reported in terms of field strength along the x, y, and z axis
TYPE_MAGNETIC_FIELD_UNCALIBRATED	Software	Provides Raw uncalibrated data from the Geomagnetic sensor
TYPE_PROXIMITY	Hardware or Software	The proximity sensor measures the distance of the device from a nearby object or the device user. This information can be in the form of exact distance measured in centimetres (cm) or just a binary value indicating far or near

© Aptech Ltd. Sensors/Session 13 19

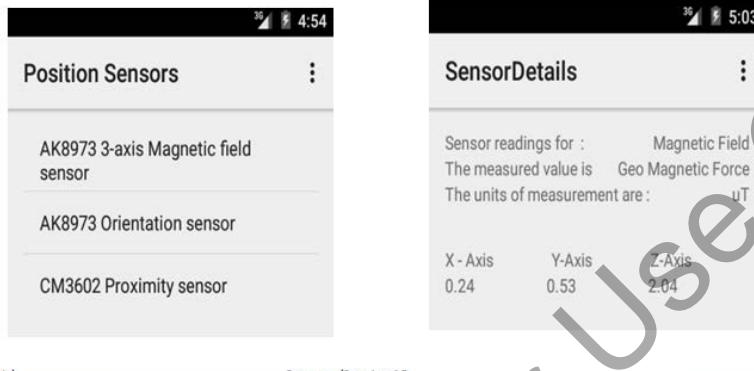
Using slide 19, explain the various Position sensors supported by Android. List the sensors and explain their purpose.

Position Sensor Example Application

Slide 20

Position Sensor Example Application

- Using the code explained for Sensor Framework, an application for demonstrating Position Sensors is created as shown in the following figure:
- Select a sensor and move the device in any direction to view the data as shown in the following figure:



© Aptech Ltd.

Sensors/Session 13

20

Using slide 20, explain the steps to create an application that utilizes position sensors. The application lists the position sensors and displays sensor related information for the selected sensor.

Turning OFF the Device Screen

Slides 21 and 22

Turning OFF the Device Screen 1-2

- There is no standard procedure in Android for Turning OFF the Screen
- Some of the workaround to achieve this behavior is explained as follows:
 - Device Policy Administrator**
 - This process involves adding the application as a Device Administrator and then, using the device policy manager to lock the screen of the device essentially switching OFF the screen
 - This is similar to emulating a Power button press
 - Power Manager Sleep Mode**
 - The Power Manager's goToSleep() method can be used to put the device in sleep mode

© Aptech Ltd.

Sensors/Session 13

21

Turning OFF the Device Screen 2-2

- ◆ **Using a Wake Lock**

- ◆ Another implementation using the power manager involves getting hold of a wake lock momentarily and then releasing it, to put the device to sleep

- ◆ **Setting the Brightness to Extremely Low**

- ◆ Most Android devices turn off the screen entirely if the brightness value of the screen is set to an impossibly low value

- ◆ **Reducing the Screen Timeout Value**

- ◆ The timeout value for the display to power OFF when there is no interaction can be set to a value low enough that the display powers OFF immediately

Using slides 21 and 22, explain the process of turning OFF the device screen from an Android application. Explain that this functionality is similar to that provided by Android's default 'Phone' application using the proximity sensor. Explain the theory behind the process.

Turning OFF the Screen Example

Slides 23 and 24

Turning OFF the Screen Example 1-2

- ◆ Following Code Snippet demonstrates various methods for turning OFF the device screen:

```
DevicePolicyManager dpm =
(DevicePolicyManager) getSystemService(Context.DEVICE_POLICY_SERVICE);
dpm.lockNow();

//Alternate Method 1
/*
PowerManager manager = (PowerManager)
getSystemService(Context.POWER_SERVICE);
manager.goToSleep(1000);
*/

//Alternate Method 2
/*
PowerManager.WakeLock wl =
manager.newWakeLock(PowerManager.PARTIAL_WAKE_LOCK, "Your Tag");
wl.acquire();
wl.release();
*/
```

Turning OFF the Screen Example 2-2

- Following Code Snippet demonstrates various methods for turning OFF the device screen:

```
//Alternate Method 3
/*
WindowManager.LayoutParams params = getWindow().getAttributes();
params.flags |= WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON;
params.screenBrightness = 0;
getWindow().setAttributes(params);
*/

//Alternate Method
/*
Settings.System.putInt(getContentResolver(),
Settings.System.SCREEN_OFF_TIMEOUT, 10);
*/
}
```

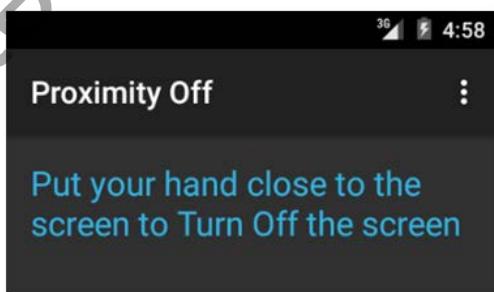
Using slides 23 and 24, explain the example of turning OFF the screen. Explain the code given in the slide. Mention that any one of the methods is sufficient to achieve this and what works on one device may not work on the other.

Proximity OFF Example

Slide 25

Proximity OFF Example

- Using the explained code, an application for automatically turning OFF the screen on proximity is created as shown in the following figure:



- The screen will be turned OFF if a hand is placed near the screen or if the device is held close to the face in an in-call position

Using slide 25, demonstrate the steps to create an application that turns OFF the screen when the phone is in proximity to the user. This application detects when an object is in close distance with the device and turns off the screen to prevent accidental touch inputs.

Proximity Example Application Logic

Slide 26

Proximity Example Application Logic

- The application logic is as follows:
- ◆ Developer creates an Activity which displays the usage information to the user
 - ◆ A listener is registered for the Proximity Sensor
 - ◆ Whenever there is change in the proximity values, the values are checked to be lesser than a threshold
 - ◆ If the values are found to be lesser then the screen is turned off
 - ◆ This is the same functionality provided by the Android's "Phone" application and other VoIP application such as skype

© Aptech Ltd.

Sensors/Session 13

26

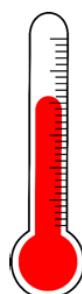
Using slide 26, explain the application logic and functionality. Explain that the application uses the theory of Proximity sensors explained earlier. Mention its real world applications such as receiving a call and VoIP applications.

Environment Sensors

Slide 27

Environment Sensors

- ◆ Environment sensors help monitor the properties of the surroundings of the device
- ◆ All environment sensors are completely optional to implement by the device vendor
- ◆ All the Environment sensors are hardware based only



© Aptech Ltd.

Sensors/Session 13

27

Using slide 27, explain Environment sensors. Explain that they are used to get information regarding the surroundings of a device. Provide a couple of examples such as the Temperature sensor. Mention that all Environment sensors are hardware based only.

Environment Sensor Types

Slide 28

Environment Sensor Types

- Following table lists the types of Environment sensors supported by Android along with their implementation option and utility:

Sensor Type	Implementation	Description
TYPE_AMBIENT_TEMPERATURE	Hardware	Temperature sensor measures the room temperature that the device is placed in
TYPE_LIGHT	Hardware	Light sensor measures the ambient light falling on the device. This information is used in devices to adjust the brightness of the screen automatically
TYPE_PRESSURE	Hardware	Pressure sensor measures the air pressure around the device
TYPE_RELATIVE_HUMIDITY	Hardware	Relative Humidity sensor measures the relative humidity in the atmosphere

Using slide 28, explain the various Environment sensors supported by Android. List the sensors and explain their purpose. Mention that they can only be implemented in hardware.

Calculating Additional Information

Slide 29

Calculating Additional Information

♦ Absolute Humidity

- Absolute humidity is the total mass of water vapour present in a given volume of air
- Absolute humidity is measured in grams/meter³
- It can be calculated using the following equation:

$$d(t,RH) = \frac{216.7 \cdot (RH/100\%) \cdot A \cdot \exp(m \cdot t / (T_n + t))}{273.15 + t}$$

♦ Dew Point

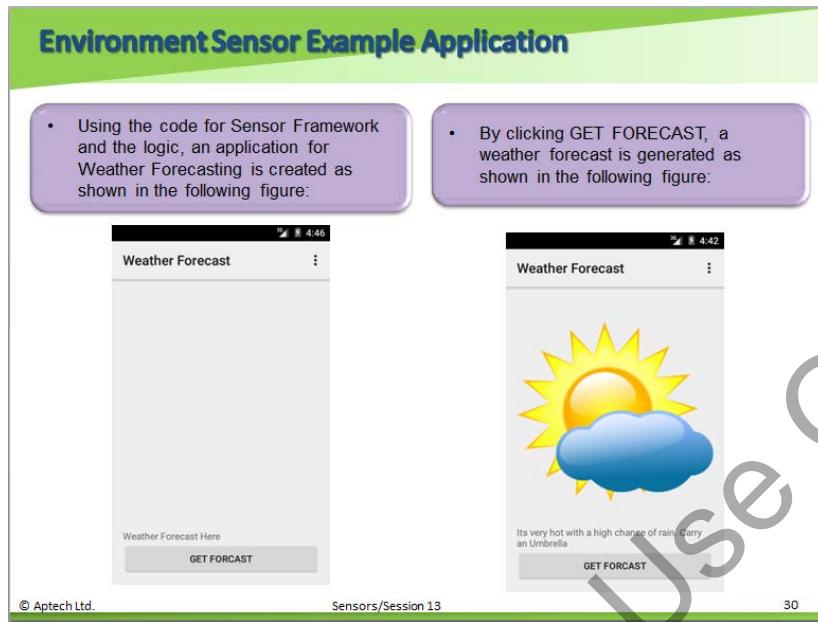
- The dew point is the temperature at which the water vapour in a sample of air at constant barometric pressure condenses into liquid water at the same rate at which it evaporates
- It can be calculated using the following equation:

$$t_d(t,RH) = \frac{T_n \cdot \ln(RH/100\%) + m \cdot t / (T_n + t)}{m - [\ln(RH/100\%) + m \cdot t / (T_n + t)]}$$

Using slide 29, explain the process of calculating metadata from Environment sensors. Explain the formulas for calculating the Absolute humidity and the dew point, both of which can be used for calculating weather forecasts.

Environment Sensor Example Application

Slide 30



Using slide 30, demonstrate the steps to create an application that uses Environment sensors. The application generates basic weather forecasts using sensor data. Explain the functionality and output of the application using the figures given on the slide.

Weather Forecast Application Logic

Slide 31

Weather Forecast Application Logic

- The application logic is as follows:
- ◆ Developer creates an Activity which displays an image summing up the weather condition and a tag line
- ◆ Once the Get Forecast button is clicked the generateForecast() method is called
- ◆ The method retrieves Environment information such as temperature, humidity and also calculates further information based on the values
- ◆ Based on the inputs, a weather prediction is made and the appropriate Icon and tag lines are displayed to the user

Using slide 31, explain the application logic of the weather forecast application.

The application reads sensor data of the Environment sensors and generates metadata to create a basic weather prediction. The weather prediction is then displayed to the user in the form of large icon appropriate to the prediction.

Other Status Related Sensors

Slide 32

Other Status Related Sensors

- ◆ Besides the three major types of sensors, there are additional sensors in Android devices which can be accessed by other means or optional and specific to certain device types.
- ◆ Some of them are:
 - ◆ Battery Sensor
 - ◆ Heart Rate Sensor



Using slide 32, explain the miscellaneous sensors which do not fall into the three categories of sensors mentioned earlier. Explain that some of these sensors need to be accessed separately and they are not part of the sensor framework. List couple of examples.

Additional Reference:

You may refer the following links for more information regarding Battery and Heart rate sensors:

<http://mobiledevtuts.com/android/android-sdk-get-device-battery-information/>

https://source.android.com/devices/sensors/sensor-types.html#heart_rate

Battery Sensor

Slide 33

Battery Sensor

- ◆ Battery status can be retrieved using a broadcast receiver with an intent filter for the intent Intent.ACTION_BATTERY_CHANGED
- ◆ The receiver will receive an update whenever there is a change in the status of the battery (that is, charging, discharging, and so on.)
- ◆ The receiver will also receive and update when there is a change in the battery level



© Aptech Ltd.

Sensors/Session 13

33

Using slide 33, explain the importance of the Battery sensor. Mention that it is not part of the standard sensor framework and the developer needs to use an Intent filter for receiving updates on battery levels.

Heart Rate Sensor

Slide 34

Heart Rate Sensor

- ◆ The Heart Rate sensor is a heart rate monitoring sensor that is unique to the Android Wear devices
- ◆ The application using this sensor needs to be targeted specifically for a wear device
- ◆ The requirements for this are small screens with circular layouts and limited resolution
- ◆ The information reported by the sensor is a single event value at the index 0
- ◆ The data represents the heart rate in beats per minute
- ◆ Heart Rate Monitor Sensor API is newly introduced and subject to change

© Aptech Ltd.

Sensors/Session 13

34

Using slide 34, explain the importance of the Heart Rate sensor. Explain that it is not supported in the base version of Android and it is unique to wearable devices.

Context Sensing and Location Based Services

Slide 35

Context Sensing and Location Based Services 1-2

- ◆ Sensors contain a lot of information which can be further processed to generate even more information
- ◆ Location also plays a key role in identifying the context
- ◆ Marketing is a prime area of use for location based services, followed by news and infotainment
- ◆ In order to implement context sensing, the developer can either write a specialized sensing framework from scratch that fits their needs



© Aptech Ltd.

Sensors/Session 13

35

Using slide 35, explain the concept of Context Aware Sensing. Explain the concept of Location Based Services (LBS). Explain that LBS adapt to the present location of the user. Explain that services such as restaurant recommendations utilize the user location to generate relevant results. Provide a few examples for its uses such as marketing and infotainment.

Context Sensing and Location Based Services

Slide 36

Context Sensing and Location Based Services 2-2

- ◆ An alternative would be to use existing context sensing services such as Intel's Context Sensing SDK
- ◆ The SDK framework automatically detects the available sensors and generates context data over time
- ◆ Google's own applications such as Google Now and products such as Google Ads implement context sensing to generate suggestions
- ◆ Before implementing Context Sensing, the developer needs to take the privacy concerns and the performance hit into account

© Aptech Ltd.

Sensors/Session 13

36

Using slide 36, explain the methods for implementing context aware services in applications. Explain that there are existing SDKs which fully implement context aware services such as Intel's Context Sensing SDK. Provide a real life example such as Google Ads.



In-Class Question: What are the applications of Location Based Services (LBS) and Context Sensing?

Answer: LBS and Context Sensing can be used in personalized Web searches, location based services, news, and infotainment applications.

Additional Reference:

You may refer the following links for more information regarding Context Sensing SDKs:

https://software.intel.com/sites/default/files/managed/88/e9/Context_ReleaseNotes_v1.5.57.pdf

<https://software.intel.com/en-us/context-sensing-sdk>

Summary

Slide 37

Summary

- ◆ Sensors are devices that record physical data
- ◆ Android supports primarily three types of sensors
- ◆ Motion Sensors monitor the movement of the device
- ◆ Position Sensors monitor the position and orientation of the device
- ◆ Environment Sensors monitor the surroundings of the device
- ◆ Other miscellaneous or vendor specific sensors are also supported by Android
- ◆ Sensors can be used for context sensing in applications

© Aptech Ltd.

Sensors/Session 13

37

Using slide 37, summarize the session. Make them revise the following points:

- Sensors are devices that record physical data
- Android supports primarily three types of sensors
- Motion Sensors monitor the movement of the device
- Position Sensors monitor the position and orientation of the device
- Environment Sensors monitor the surroundings of the device
- Other miscellaneous or vendor specific sensors are also supported by Android
- Sensors can be used for context sensing in applications

13.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session. You should also explore the next session which describes Google Play Store, publishing applications, best practices, and maximizing revenue.

Tips:

You can also check the **Articles/Blogs/Expert Videos** uploaded on the Onlinevarsity site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the Onlinevarsity site to ask queries related to the sessions.

Session 14: Google Play Store

14.1 Pre-Class Activities

You should familiarize yourself with the Play Store. You should also acquire a good understanding of market shares and best practices to ensure the success of the application. You should review the procedure to publish an application on the Play Store.

Familiarize yourself with the topics of the current session in-depth.

14.1.1 Objectives

After the session, learners will be able to:

- Explain the requirement for Google Play Store
- Explain the different versions and feature set
- Explain the share of Android in the market
- Explain the different kinds of devices available in the market
- Describes the making of an .apk file
- Explain the process of publishing the .apk file in Google Play Store
- Explain the best practices to be followed
- Explain good marketing and promotion strategies

14.1.2 Teaching Skills

You should be familiar with Play Stores and the current Android store eco system. You should be able to suggest and implement best practices when publishing applications. Proficiency with the procedure of publishing applications is mandatory. You should be able to explain the market shares and the necessity of targeting the right version.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of the students by asking questions in between the class.

14.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Review of Previous Session

You should begin with a brief recap or review of previous session. Tell the students that the previous session explained the basics of Sensors including Motion, Position, and Environment Sensors, and Location and context aware services.

Overview of the Session

Here, give the students the overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell the students that this session explains basics of the Play Store, best practices, and steps to ensure the success of the application.

14.2 In-Class Explanations

Introduction

Slide 3

Introduction

- ◆ Google Play Store is Google's market place for Android Applications, music, movies, and books
- ◆ Google Play Store comes pre-installed in all Android Devices
- ◆ It serves as the main source for application for more than 70% of Android users
- ◆ Hence, it is in the best interest of a developer to publish the app on Google Play



© Aptech Ltd.

Google Play Store/Session 14

3

Using slide 3, introduce Google Play Store and its importance. Explain that Google Play Store is the central store application for all Android Devices and it comes pre-installed in all Android certified devices.

What is the Play Store?

Slide 4

What is the Play Store?

- ◆ Google Play Store (formerly, Android Market) is an online electronic store or digital application distribution platform for Android powered devices
- ◆ It is a robust publishing platform
- ◆ The services offered by Play Store to the users includes downloading free and paid applications, books, movies, music, magazines, and so on
- ◆ Users can also purchase mobile devices such as Google-Nexus, Chromebook, and so on through Play Store
- ◆ The developer will have access to revenue generating tools, such as in-app-billing and application-licensing
- ◆ The developer will also know the sale trends and understand the end users

© Aptech Ltd.

Google Play Store/Session 14

4

Using slide 4, explain the importance of the Play Store. Explain that it is Google's official application distribution platform. Explain that it is important to publish on the Play Store for the application's success.

Additional Reference:

You may refer to the following links for more information regarding Play Store:

<http://www.androidcentral.com/google-play-store>

https://en.wikipedia.org/wiki/Google_Play

Market Shares and Targeting the Right Versions

Slide 5

Market Shares and Targeting the Right Versions

- ◆ The Android market is fragmented. The hardware and the OS version vary from one device to another
- ◆ This enables innovation and rapid progress on the platform
- ◆ However, at the same time, it adds the burden on the developer to maintain compatibility
- ◆ Targeting the latest and greatest may give you better quality application, but the potential market of the application will be reduced drastically



© Aptech Ltd.

Google Play Store/Session 14

5

Using slide 5, explain the concept of Market Shares and the importance of targeting the right version. Explain that revenue can only be as large as the target audience and therefore, choosing a small subset of audience for features can severely affect the returns. Explain that selecting the least common denominator in terms of OS support is usually recommended.



In-Class Question: What is the previous title of the Play Store?

Answer: Play Store was formerly called as Android Market.

Hardware Requirements

Slide 6

Hardware Requirements

- ◆ The developer needs to try to utilize the least common denominator in terms of hardware
- ◆ Support alternatives when the hardware is not available
- ◆ If the application is using location based services and GPS hardware is not available, the application uses the network location to generate an estimate
- ◆ If GPU acceleration is being used, try to make sure that the application runs properly in a device from two generations earlier (Nexus 4 if Nexus 6 is the current generation)



© Aptech Ltd.

Google Play Store/Session 14

6

Using slide 6, explain the hardware requirements that need to be set for the application. Explain that relying on device specific hardware can hinder the app's 'reachability' immensely. Explain that choosing a flagship device from two generations earlier is a good benchmark for a minimum hardware requirement.

Operating System Updates

Slide 7

Operating System Updates

- ◆ Operating System upgrades on the Android platform are much slower than any other platform
- ◆ The user cannot decide when/if he can upgrade to a new OS version. The device will receive the update only if the vendor pushes an update
- ◆ The vendor may also choose to discontinue updates for the device leaving the device forever at the same OS version
- ◆ Only the Nexus devices receive the update on day one
- ◆ Hence, the developer needs to make sure that by targeting the latest version, he is not alienating potential customers
- ◆ Most applications have no particular use of the latest API
- ◆ Unless it is absolutely mandatory to make the switch, try to use the old API

© Aptech Ltd.

Google Play Store/Session 14

7

Using slide 7, explain operating system updates in Android. Mention that migrations in Mobile devices are much slower than other platforms. Explain that the main reason behind this is that the users do not have control of when they have the facility to update. Explain that Nexus devices are the only line of devices which receive update on the first day of release.

Market Shares

Slide 8

Market Shares

- As of May 2015, the number of devices executing particular version of Android are displayed in the following table:

Android Version	Version No.	API	Share
Lollipop	5.1	22	0.7 %
Lollipop	5.0	21	9 %
Kitkat	4.4	19	39.8 %
JellyBean	4.2	18, 17, 16	39.2 %
Ice Cream Sandwich	4.0	15	5.3 %
Gingerbread	2.3	9	5.7 %
Froyo	2.2	8	0.3 %

Market Share

- It must be ensured that the application supports atleast 80% of the market share for it to be successful. In that case, the minimum SDK version suitable is JellyBean.

© Aptech Ltd. Google Play Store/Session 14 8

Using slide 8, explain the market share of the versions of Android. Using the Pie chart given on the slide, explain that KitKat is the most widely used version, whereas choosing JellyBean will ensure compatibility with greater than 80% of the devices.

Additional Reference:

You may refer to the following link for more information regarding market shares:

[https://en.wikipedia.org/wiki/Android_\(operating_system\)#Platform_usage](https://en.wikipedia.org/wiki/Android_(operating_system)#Platform_usage)

Support Library

Slide 9

Support Library

- In order to ease the issue with variation of API, Android has introduced Support Libraries to use in applications
- By using these libraries, the developer can continue to use the newer API and still make sure that the application will be compatible with older devices
 - The Library chooses the appropriate implementation for the developer
 - The packages android.support.v4 and android.support.v7 should be imported for the v4 and v7 support libraries respectively
 - The classes in this library have a prefix of AppCompat with the same API

© Aptech Ltd. Google Play Store/Session 14 9

Using slide 9, explain the importance of the support library. Explain that support library ensures compatibility across API changes. Explain that the library chooses the appropriate implementation for the developer and the developer need not worry about the implementation code. The code will remain the same. Explain that support library classes have the prefix of AppCompat before their class names.



In-Class Question: What is the minimum recommended platform support percentage for the API Version?

Answer: It is recommended to choose a version which ensures support with at least 80% of the user base or higher.

Additional Reference:

You may refer to the following link for more information regarding Google Support Library:
<https://developer.android.com/tools/support-library/index.html>

Adding Support Libraries

Slide 10

Adding Support Libraries

- Support Libraries can be added from the SDK Manager as shown in the following figure:

Name	API	Rev.	Status
Android 2.2 (API 8)	14	Update available: rev. 15	
Android Support Repository	22.1.1	Update available: rev. 22.2	
Android Support Library	24	Update available: rev. 25	
Google Play services	17	Update available: rev. 19	
Google Repository	3	Not installed	
Google Play ARCore Expansion Library	5	Not installed	
Google Play Billing Library	2	Not installed	
Google Play Licensing Library	1	Not installed	
Android Auto API Simulators	11	Installed	
Google USB Driver	2	Not installed	
Google Web Driver	5.3	Installed	
Intel x86 Emulator Accelerator (HAXM installer)			

Show: Updates/New Installed Select New or Updates
 Obsolete Delete All Install 8 packages... Delete 7 packages...

Done loading packages.

Using slide 10, explain the steps to add support libraries to Android. Explain that this can be done from the SDK Manager. Instruct the students to select the Android Support Library and Android Support Repository, and click Install X packages.

Perquisites to Publishing on the Play Store

Slide 11

Perquisites to Publishing on the Play Store

- ◆ Certain steps need to be taken before publishing the application to End users
- ◆ Before uploading the application in Android market, the basic points to be noted are as follows:
 - ◆ Testing the application
 - ◆ Checking the application performance
 - ◆ SDK Compatibility

© Aptech Ltd.

Google Play Store/Session 14

11

Using slide 11, explain the steps that need to be performed before publishing the application on the Play Store. List the steps using the bulleted points given on the slide.

Testing the Application

Slide 12

Testing the Application

- ◆ The application developed must be tested well before uploading it to the Play Store so that the uploaded application is not only error/bug free but also does not crash at users' end
- ◆ A stable application and good reviews are key to the success of the application
- ◆ The output varies from one device to another and also from one version of Android to another
- ◆ The application is required to be tested on real devices of different types before being uploaded to the Play Store for the users to download
- ◆ The resolutions supported and the layouts for the supported resolutions need to be thoroughly tested

© Aptech Ltd.

Google Play Store/Session 14

12

Using slide 12, explain the importance of testing the application before publishing it on the Play Store. Explain that the developer needs to ensure that the application gets published bug free to receive favorable reviews. Explain that the developer also needs to ensure that the application works properly across all devices.



In-Class Question: What is the purpose of the Support libraries?

Answer: The support libraries ensure backward compatibility of the code with older versions of Android.

Preparations

Slides 13 and 14

Preparations 1-2

- ◆ **Collect Material for Release:**

- ◆ This includes preparation of End User License Agreement (EULA) for the application
- ◆ This will help to protect not only the developer and the organization but also the intellectual property rights
- ◆ It also includes the process of obtaining an encrypted key for digitally signing the application and creation of an icon for the application

- ◆ **Arrange Application for Resource:**

- ◆ This includes gathering of all the materials required for configuring the applications which includes configuration changes made to the source code, resource files, and the manifest file of the application
- ◆ Besides this the developer has to clean up the project, update the manifest file setting, update the URLs for servers and services, and so on

Using slide 13, explain the preparations that need to be made before publishing the application on Play Store. Explain that the developer needs to prepare the EULA and privacy policy before publishing the application. Mention that the application also needs to be signed prior to publishing. Explain that the developer also needs to clean up the project and update the Test Settings environments to real time values.

Preparations 2-2

- ◆ **Build Application for Resource:**

- ◆ This includes signing the application and building the application for release

- ◆ **Prepare Remote Server:**

- ◆ This includes the process of ensuring that if a remote server is used it is secure and configured for use

- ◆ **Test Application for Resource:**

- ◆ This includes testing the application to ensure that the application works properly under real device and varying conditions

Using slide 14, explain the remaining steps that need to be performed before publishing the application. List and explain them using the bulleted points given on the slide.

Application Performance and Pre-Publish Tasks

Slide 15

Application Performance and Pre-Publish Tasks◆ **Application Performance**

- ◆ The performance of the application is one of the most important concerns to the developer
- ◆ A sluggish application is destined to receive bad reviews followed by less downloads
- ◆ Code optimization tools such as lint and code clean up tools (part of the IDE) can be used to improve the performance of the application

◆ **Pre-Publish Tasks**

- ◆ Set permissions
- ◆ Establish version, set icon, and application label
- ◆ Set compatibility options
- ◆ Remove log data
- ◆ Export the project and create the key using Eclipse

Using slide 15, explain the importance of checking the application performance and the pre-publish tasks that need to be performed. Explain that code optimization is very important and it can also be automated using tools such as lint. List the pre-publish tasks and explain them briefly.

Setting Permissions

Slide 16

Setting Permissions◆ **<uses-permission> element**

- ◆ In this tag, specify only those permissions that are relevant and required for executing the application

◆ **android:icon and android:label attributes**

- ◆ In the <application> element, specify values for these attributes as these are displayed to the user. The default icon needs to be replaced with a proper icon representing the application

◆ **android:versionCode and android:versionName attributes**

- ◆ In the <manifest> element, specify values for these attributes

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS" />
<uses-permission android:name="android.permission.WAKE_LOCK"/>
```

Using slide 16, explain the importance of setting the appropriate permissions. Explain that the developer specifies only the required permissions. Mention that adding unnecessary permissions can raise a warning.

Version

Slide 17

Version

- The version names have to be specified in the AndroidManifest file that is written within the <manifest> tag
- The version code starts from 1 and is required to be incremented by 1 for each upload. Version name specifies the release version and always should be in ascending order

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.helloandroid"
    android:versionCode="1"
    android:versionName="1.0" >
```

© Aptech Ltd.

Google Play Store/Session 14

17

Using slide 17, explain that version tag and version control is mandatory for publishing applications on the Play Store. Explain that Android uses this to distinguish updates for the application.

Specifying Icon, Application Label, and SDK Version

Slide 18

Specifying Icon, Application Label, and SDK Version

- The icon to be set for the application and the name of the application have to be specified inside the <application> tag in AndroidManifest file
- The minSdkVersion, specifies the minimum sdk version the application will support
- The targetSdkVersion, specifies the version to which the application is built for
- The maxSdkVersion, specifies the maximum sdk version the application can support

```
<uses-sdk android:minSdkVersion="8"
    android:targetSdkVersion="10"
    android:maxSdkVersion="16"/>
```

© Aptech Ltd.

Google Play Store/Session 14

18

Using slide 18, explain the importance of the application icon, the application label, and the required API version. Explain that a unique icon is required to distinguish the application from others.

Other Pre-Publish Tasks

Slide 19

Other Pre-Publish Tasks

- ◆ **Feature Restriction**
 - ❖ Features that are essential to the functionality of the application can be specified using the `uses-feature` tag
 - ❖ This ensures that the application is not installed on devices that do not have the feature
- ◆ **Compatibility Options**
 - ❖ The Android system will decide on which platform the application will be installed depending on the target specified in the manifest file
 - ❖ The play store automatically identifies the compatible devices and prevents unsupported devices from installing the application
 - ❖ The publisher can change this from the Google Play Developer Console
- ◆ **Remove Log Data**
 - ❖ Remove the calls to logs, as it will display output data in LogCat always
 - ❖ Unnecessary resource files are required to be removed as it will lead to increase in size of the application
 - ❖ This can also lead to security vulnerabilities and leak key information about the source code of the application

Using slide 19, explain the rest of the pre-publish tasks that need to be completed. Explain that feature restriction needs to be used to ensure the application only runs on devices that are capable of running it. Explain that the compatibility options need to be used to enforce the minimum version requirement on the application. Explain that the developer also needs to remove log data from the application.

Signing an Application

Slides 20 to 24

Signing an Application 1-5

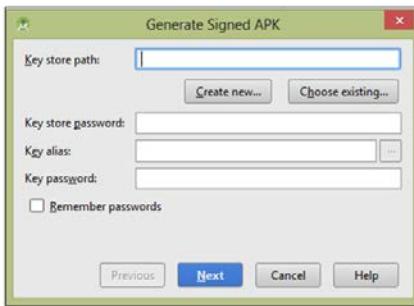
- ◆ Before an application can be uploaded to any application store, the application needs to be signed by the developer
- ◆ There is no need for a Certification Authority
- ◆ The developers can generate their own certificates for signing the application
- ◆ The purpose of signing the application is to verify the authenticity of the application and to ensure that it is not a counterfeit or tampered with



Using slide 20, explain the importance of signing the application. Explain that a Certification Authority is not mandatory for signing the application and the developers can generate their own certificates for this purpose. Explain that signing the application ensures that the application is not tampered with or modified with a malicious intent.

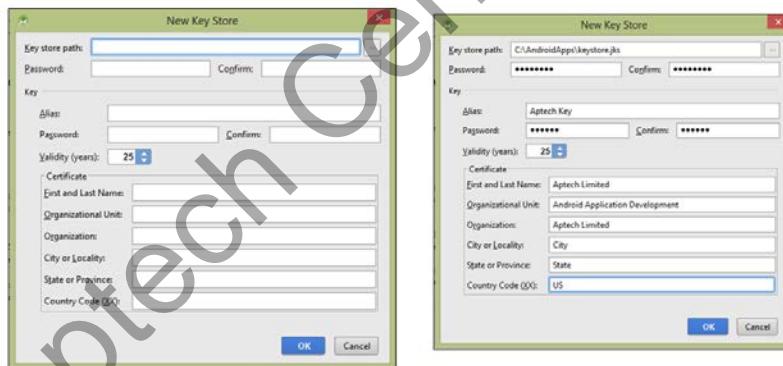
Signing an Application 2-5

- Start Android Studio and open the desired project
- Navigate to the Build Menu and select Generate Signed APK
- The Generate Signed APK dialog box is displayed as shown in the following figure:



Signing an Application 3-5

- If it is the first time signing, click Create new to open the New Key Store dialog box as shown in the following figure:
- Enter the details as shown in the following figure:



Signing an Application 4-5

- Click OK to return to the Generate Signed APK dialog box with the details already entered as shown in the following figure:



© Aptech Ltd.

Google Play Store/Session 14

23

Signing an Application 5-5

- Click Next to display the Destination dialog box as shown in the following figure:



- Click Finish. A new app-release.apk file is generated by Android Studio in the specified target directory. This .apk file can be renamed and uploaded to the Play Store

© Aptech Ltd.

Google Play Store/Session 14

24

Using slides 21 to 24, demonstrate the process of signing the application. List the steps and explain them using the figures given on the slides.



In-Class Question: What is the purpose of signing an application?

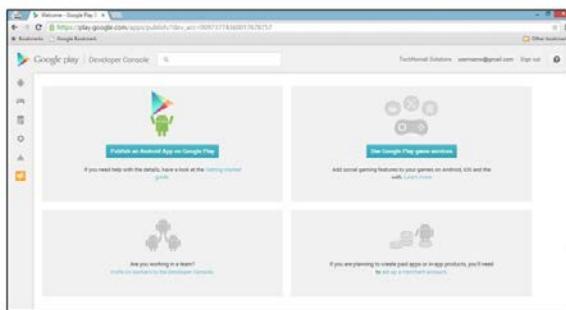
Answer: Signing an application ensures authenticity of the application and certifies that the application has not been tampered with.

Publishing Android Application to the Play Store

Slides 25 to 37

Publishing Android Application to the Play Store 1-13

- Create a Signed .apk file
- Sign in to the Play Store Account as shown in the following figure:



- URL - <https://play.google.com/apps/publish/>

© Aptech Ltd.

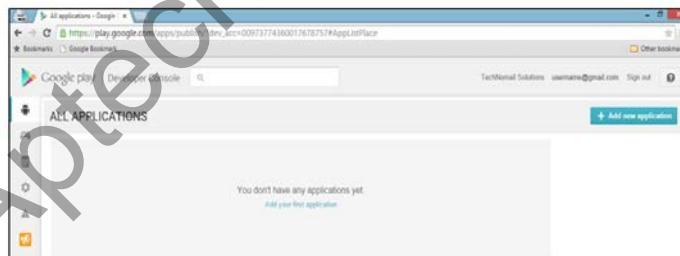
Google Play Store/Session 14

25

Using slide 25, explain that the application can be published by visiting play.google.com/apps/publish. Explain that a developer account is needed to publish an application.

Publishing Android Application to the Play Store 2-13

- Select the 'ALL APPLICATIONS' tab from the left pane. Add your application to play store by clicking Add new application as shown in the following figure:



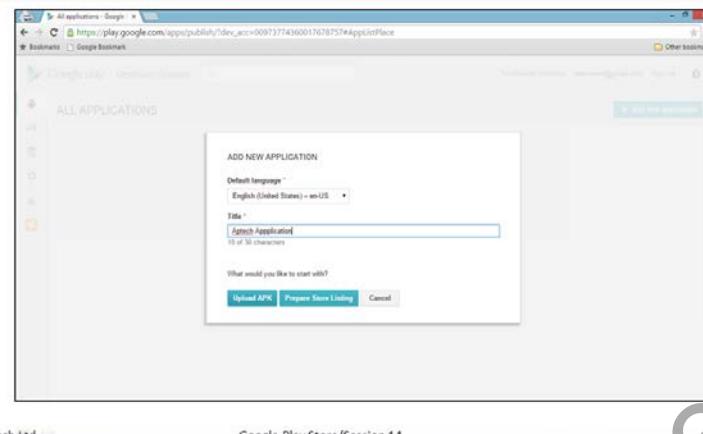
© Aptech Ltd.

Google Play Store/Session 14

26

Publishing Android Application to the Play Store 3-13

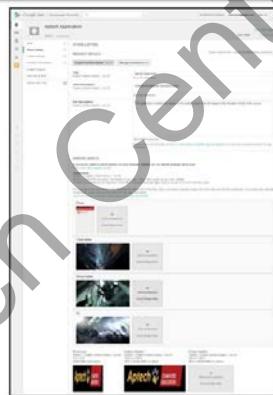
- The Add New Application screen is displayed as shown in the following figure:



© Aptech Ltd. 27 Google Play Store/Session 14

Publishing Android Application to the Play Store 4-13

- Store listing displays a page which allows the developer to enter the description of the application, Promo text, Recent changes, and so on as shown in the following figure:



© Aptech Ltd. 28 Google Play Store/Session 14

Publishing Android Application to the Play Store 5-13

- Screen shots of the application is taken using either Emulator or Real device and uploaded as shown in the following figure:



© Aptech Ltd.

Google Play Store/Session 14

29

Publishing Android Application to the Play Store 6-13

- The Application type and Category helps the developer to specify the type of application that is getting uploaded as shown in the following figure:

The screenshot shows the 'Categorization' section of the Google Play Store. It includes dropdown menus for 'Application type' (set to 'Applications'), 'Category' (set to 'Education'), and 'Content rating' (set to 'Everyone'). There is also a note about filling a rating questionnaire for new content ratings.

© Aptech Ltd.

Google Play Store/Session 14

30

Publishing Android Application to the Play Store 7-13

- Play Store allows the developer to enter the contact information as shown in the following figure:

CONTACT DETAILS

Website: http://www.aptech-education.com

Email: info@aptech-education.com
Please provide an email address where you may be contacted. This address will be publicly displayed with your app.

Phone:

PRIVACY POLICY *

If you wish to provide a privacy policy URL for this application, please enter it below.

Privacy Policy: Not submitting a privacy policy URL at this time. [Learn more](#)

© Aptech Ltd.

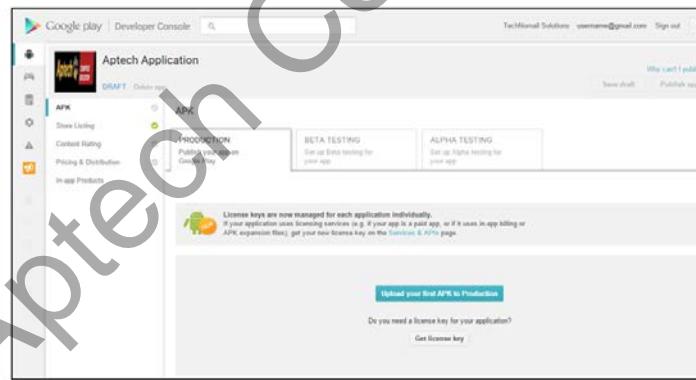
Google Play Store/Session 14

31

Using slides 26 to 31, explain that the developer needs to specify the details required to publish the application. Explain that the screenshots are important when publishing the application.

Publishing Android Application to the Play Store 8-13

- Finally, the developer needs to click 'Upload your first APK to Production' as shown in the following figure:



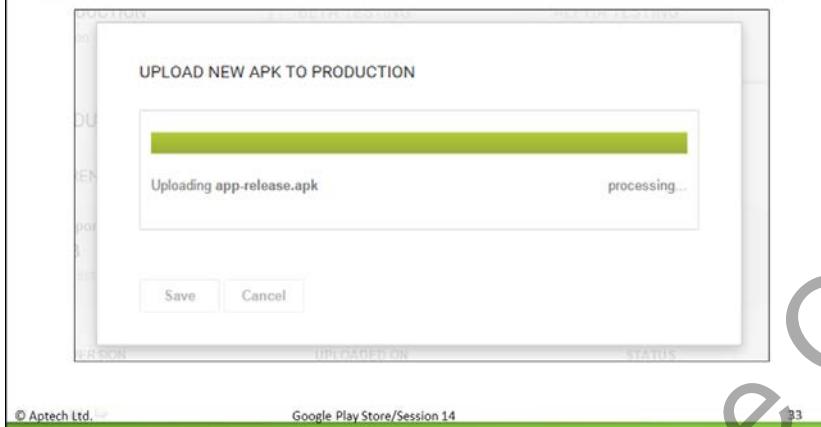
© Aptech Ltd.

Google Play Store/Session 14

32

Publishing Android Application to the Play Store 9-13

- The developer needs to navigate to the desired folder and select the .apk file and the uploading process starts as shown in the following figure:



Publishing Android Application to the Play Store 10-13

- The developer needs to complete a content rating questionnaire before the application is published. Click the Content Rating tab as shown in the following figure:



Publishing Android Application to the Play Store 11-13

- Finish the questionnaire and click Save questionnaire as shown in the following figure:



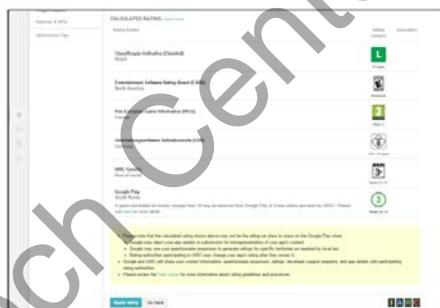
© Aptech Ltd.

Google Play Store/Session 14

35

Publishing Android Application to the Play Store 12-13

- Click Calculate rating. The confirmation screen is displayed as shown in the following figure:



- Click Apply Rating

© Aptech Ltd.

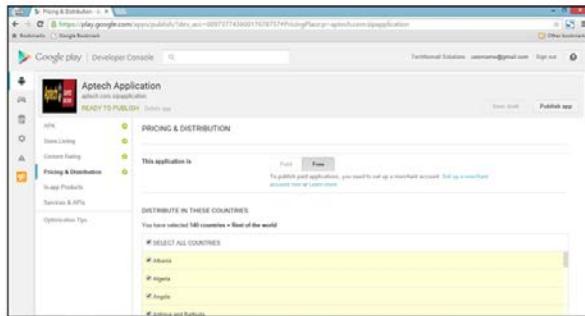
Google Play Store/Session 14

36

Using slides 32 to 36, explain that the questionnaire is used to generate rating for the application. Explain that the rating dictates the suggestions for the application. A rating with mature audience will limit the application only to be suggested for users with 18+ ages.

Publishing Android Application to the Play Store 13-13

- Click Pricing & Distribution tab in the left pane of the screen as shown in the following figure:



- Once all these steps are completed, the developer is ready to publish the application. Click Publish app. The App is published on the Play Store

© Aptech Ltd.

Google Play Store/Session 14

37

Using slide 37, explain the steps to publish an application on the Play Store. Explain that clicking Publish app will publish the application to the store. Explain that an application once published, cannot be actually 'deleted' from the store. It can only be delisted.

Best Practices

Slide 38

Best Practices

- Setup a Developer Page**
 - Play Store allows the developer to set up a developer page which displays developer/publisher information and promotional material
 - This page can help to increase visibility and priority of the application in search results
- Upload and Publish for Testing**
 - Play Store allows the developer/publisher to upload and publish app in Alpha or Beta testing phase before making it available to all users
 - Rating is also disabled during these phases
 - Participants can be added from within the organization during alpha phase
- Revenue Stream**
 - Choose the right price
 - Free Trial and Limited Versions



© Aptech Ltd.

Google Play Store/Session 14

38

Using slide 38, explain the best practices that need to be followed by the developer for best results. Explain the right steps and decisions that need to ensure the application's success. Explain that having a developer's page makes the developer appear more professional and helps in application visibility. Explain that improper testing of an application can severely affect the application's success in the long run. Explain the alternative revenue generation streams such as, free, paid, ad-supported, and 'Freemium' models.

Marketing and Promoting

Slide 39



The slide has a green header bar with the title "Marketing and Promoting". Below the header, there is a list of five items: Quality Assurance, Customer Support, Google Adwords, Setup a Website, and Reputation and Reviews. To the right of the list is a large green Android robot head icon with a diagonal slash through it, indicating that the content is for internal use only. The footer of the slide includes the copyright notice "© Aptech Ltd.", the source "Google Play Store/Session 14", and the slide number "39". A large watermark "For Aptech Internal Use Only" is diagonally across the slide.

- ◆ Quality Assurance
- ◆ Customer Support
- ◆ Google Adwords
- ◆ Setup a Website
- ◆ Reputation and Reviews

© Aptech Ltd. Google Play Store/Session 14 39

Using slide 39, explain the importance of marketing and promotion. Explain that without the right marketing strategies even the best of the ideas are doomed to fail. Explain the various strategies that can be used to market the application to the widest possible audience. Explain that Quality Assurance, reputation, reviews, and good customer support will ensure good publicity. Explain that Google AdWords can be used to target specific potential audience. Explain that setting up a Website will make the organization look more professional and build trust among the user base. This can be extremely important if the application is paid or implements micro transactions.



In-Class Question: Which are the commonly used revenue models for Android applications?

Answer: The most commonly used revenue models for Android applications are:

- Free
- Ad supported
- Paid
- Freemium (Micro-transactions)

Summary

Slide 40

Summary

- ◆ The Google Play Store (formerly Android market) is known to be the very own repository of Google for Android applications
- ◆ Play Store also sells music, movies, e-Books, and audio books
- ◆ In order to publish applications on the play store, a Google Play Developer Console account is needed. A payment of USD 25 is required for this
- ◆ The application needs to be signed and tested before uploading to the Play Store
- ◆ Marketing and good reputation are key factors for the success of an application
- ◆ In order to maximize the audience, the market shares and distributions need to be researched thoroughly

Using slide 40, summarize the session. Make them revise the following points:

- The Google Play Store (formerly Android market) is known to be the very own repository of Google for Android applications.
- Play Store also sells music, movies, e-Books, and audio books.
- In order to publish applications on the play store, a Google Play Developer Console account is needed. A payment of 25 USD is required for this.
- The application needs to be signed and tested before uploading to the Play Store.
- Marketing and good reputation are key factors for the success of an application.
- In order to maximize the audience, the market shares and distributions need to be researched thoroughly.

14.3 Post Class Activities for Faculty

Since this is the last session, you can skip doing this.

Session 15: Android Studio 2.1 and Android 6.0 Marshmallow

15.1 Pre-Class Activities

Before you commence the session, you should familiarize yourself with the topics of this session in-depth. Prepare a question or two that will be a key point to relate the current session objectives.

15.1.1 Objectives

By the end of this session, learners will be able to:

- Identify the updated features of Android Studio 2.1
- Explain the updated and new features of Android 6.0 Marshmallow

15.1.2 Teaching Skills

To teach this session, you should be well versed with Android 6.0 Marshmallow along with its history, features, uses, and benefits. You should be familiar with the differences between Android syntax and rules. You should also be aware of how Android Application is created.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of students by asking questions in between the class.

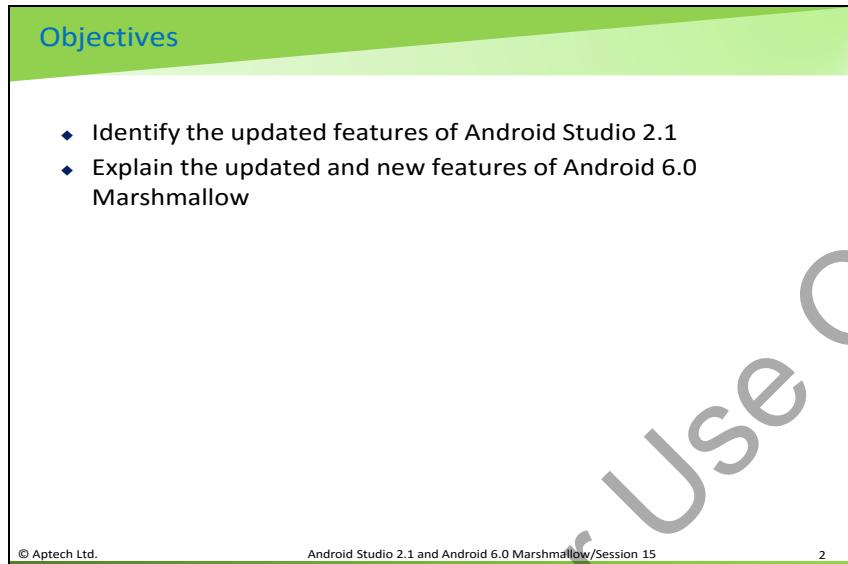
15.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Overview of the Session

Give the students an overview of the current session in the form of session objectives. Show students slide 2 of the presentation.

Slide 2



The slide has a green header bar with the word "Objectives". Below the header is a white content area containing two bullet points:

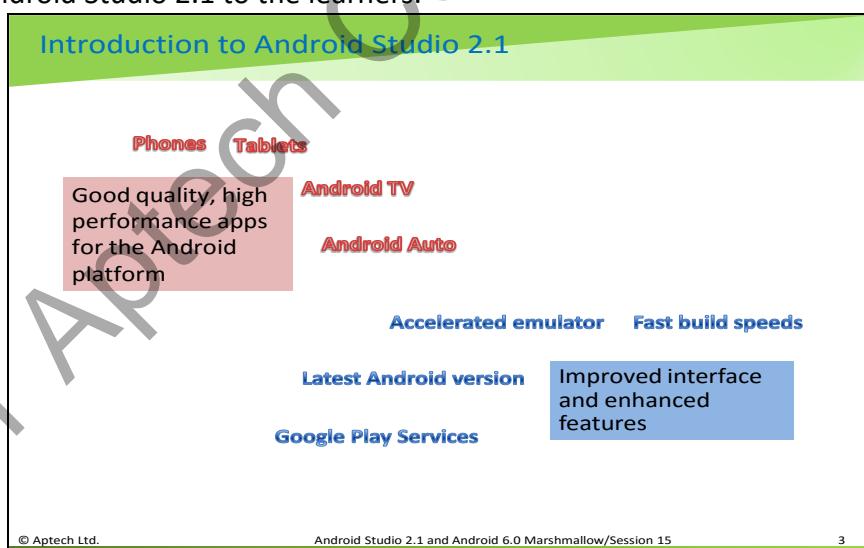
- ♦ Identify the updated features of Android Studio 2.1
- ♦ Explain the updated and new features of Android 6.0 Marshmallow

At the bottom of the slide, there is a thin green footer bar with the text "© Aptech Ltd.", "Android Studio 2.1 and Android 6.0 Marshmallow/Session 15", and the number "2".

15.2 In-Class Explanations

Slide 3

Introduce Android Studio 2.1 to the learners.



The slide has a green header bar with the title "Introduction to Android Studio 2.1". Below the header is a white content area with several sections:

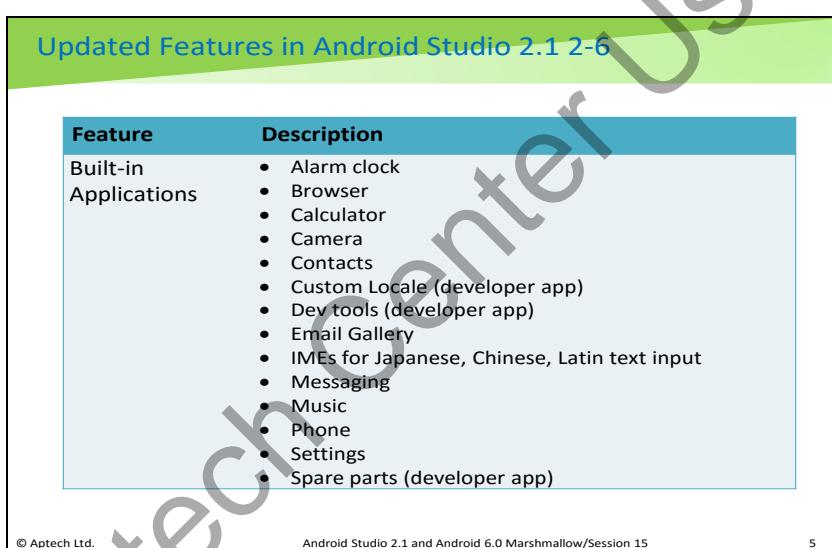
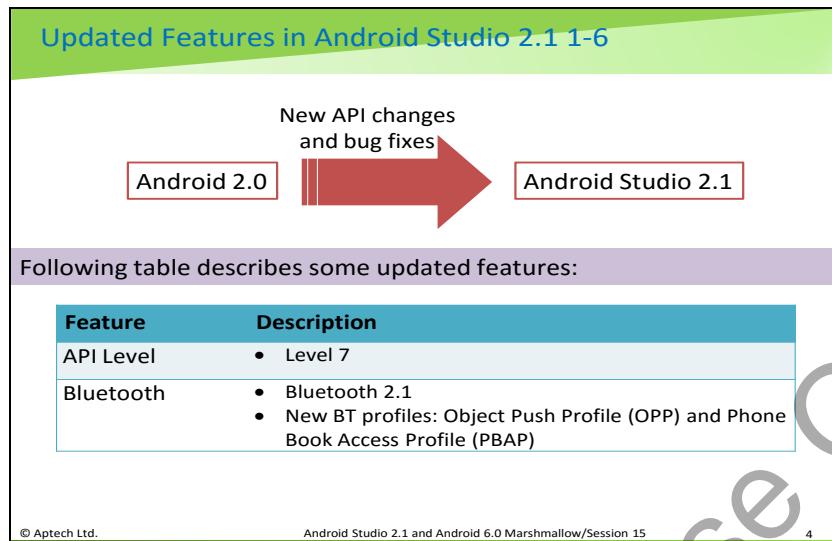
- Phones** and **Tablets**: "Good quality, high performance apps for the Android platform"
- Android TV** and **Android Auto**
- Accelerated emulator** and **Fast build speeds**
- Latest Android version** and **Improved interface and enhanced features**
- Google Play Services**

At the bottom of the slide, there is a thin green footer bar with the text "© Aptech Ltd.", "Android Studio 2.1 and Android 6.0 Marshmallow/Session 15", and the number "3".

Explain that Android Studio 2.1 is the latest release of Android Studio that enables developers to build good quality, high performance apps for the Android platform, including phones, tablets, Android Auto, and Android TV. This new and stable version of Android Studio has an improved interface and many enhanced features, such as fast build speeds and an accelerated emulator with support for the latest Android version and Google Play Services.

Slides 4 to 8

Explain that the table describes the updated features of Android Studio 2.1. Read out the contents of the table.



Updated Features in Android Studio 2.1 4-6

Feature	Description
Views	Android Studio 2.1 provides New: <ul style="list-style-type: none">• View methods: <code>isOpaque()</code> and <code>onDrawScrollBars()</code>• RemoteViews methods: <code>addView()</code> and <code>removeAllViews()</code>• ViewGroup methods: <code>isChildrenDrawingOrderEnabled()</code> and <code>setChildrenDrawingOrderEnabled()</code>

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

7

Updated Features in Android Studio 2.1 5-6

Feature	Description
Webkit	Android Studio 2.1 provides New: <ul style="list-style-type: none">• WebStorage methods to manipulate Web storage databases.• GeolocationPermissions methods to get and set Geolocation permissions.• WebSettings methods to manage settings for app cache, Web storage, and zooming.• WebChromeClient methods for handling video, browsing history, custom Views and app cache limits.

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

8

Slide 9

Talk about additional features of Android Studio 2.1.

Updated Features in Android Studio 2.1 6-6

Additional Features

- Instant Run
- Android Emulator
- Cloud Test Lab Integration
- App Indexing Code Generation and Test
- GPU Debugger Preview
- IntelliJ 15 Update

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

9

Explain that, in addition the platform provides new features that Android developers can use to develop and build applications faster and more efficiently.

Let us look at some features now. Instant Run can apply the changes instantly in live and running app. This ensures faster builds for developers.

The new Android emulator is at least three times faster than the previous emulator. The enhanced Android Debug Bridge (ADB) ensures that the developers can deploy apps and data to the emulator at a much faster rate. In addition, the built-in Google Play Services enable the developer to test out more API functionality. The new emulator has rich new features to manage calls, battery, network, GPS, and more.

Cloud Test Lab Integration is used to run and test the app on a wide range of physical Android devices in the Cloud Test Lab from within Android Studio. This helps to make the apps more robust and fool-proof.

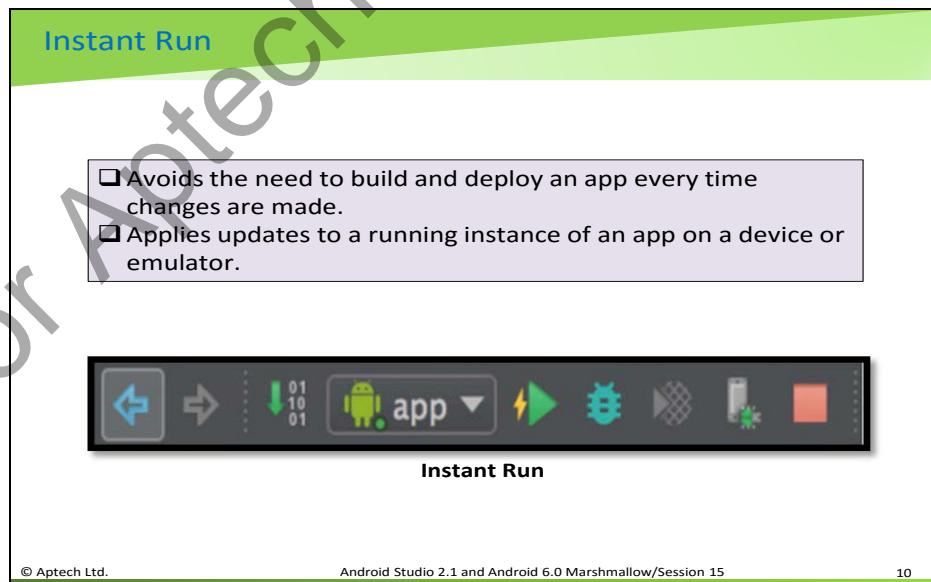
After the app is out in the market, developers would want their app to be visible in Google Search for users. The new App Indexing feature in Android Studio can help developers to add indexable URL links that they can test all within the IDE.

While developing OpenGL ES-based games or apps, the new GPU debugger displays each frame and the GL state. This helps a great deal in diagnosing GL rendering issues by capturing and analyzing the GPU stream from the Android device.

Android Studio 2.1 comes with new features from IntelliJ, a world class coding platform packed with useful tools for code navigation, code inspection, and refactoring.

Slide 10

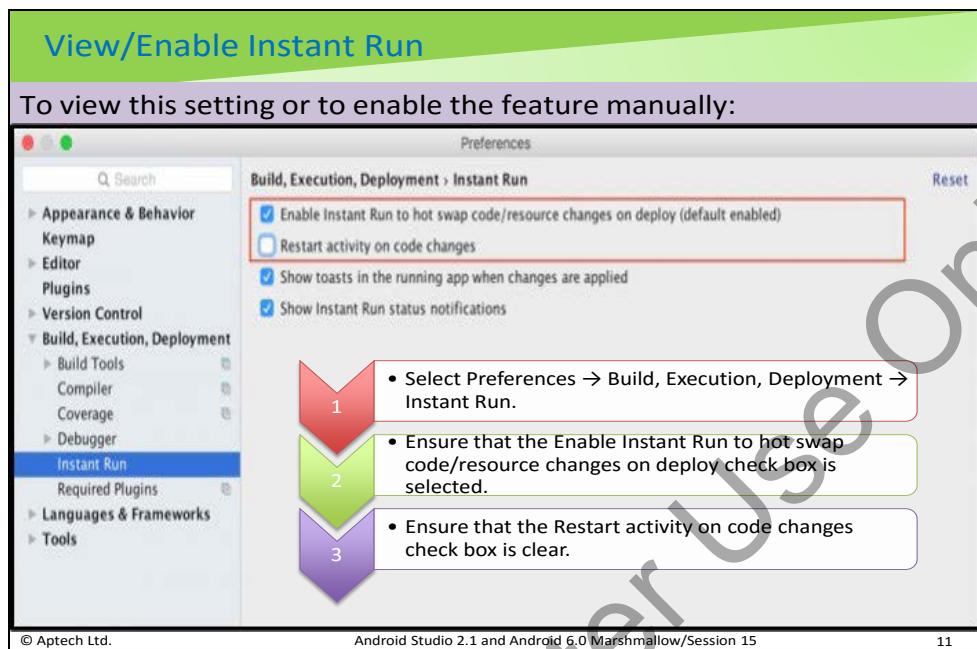
Explain each feature of Android Studio 2.1 in brief. Talk about Instant Run.



Explain that this feature of Android Studio 2.1 is a leap towards reducing the need to build and deploy an app every time changes are made. Instant Run provides pushing of updates, including code and resources, to a running instance of an app on a device or emulator without requiring a full reinstall.

Slide 11

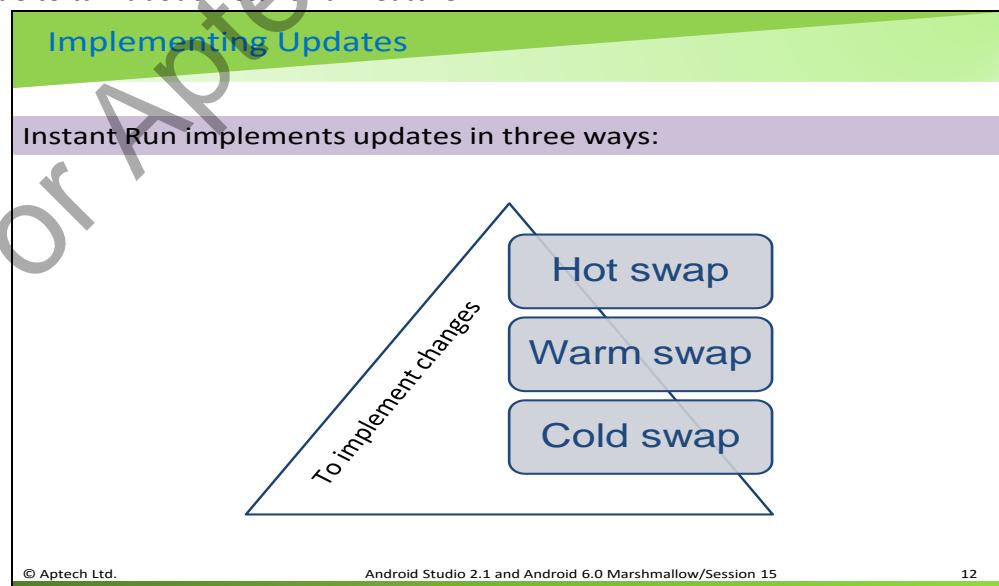
Talk about enabling the Instant Run feature manually. Tell students that the screenshot shows the Instant Run feature. It also shows that Restart activity on code changes check box is clear.



Explain that the Instant Run feature is enabled by default. However, to view this setting or to enable the feature manually, select Build, Execution, Deployment under Preferences and choose Instant Run. This screenshot appears after Instant Run is selected. Here, ensure that the Enable Instant Run to hot swap code/resource changes on deploy is selected. And, ensure that the Restart activity on code changes is clear.

Slide 12

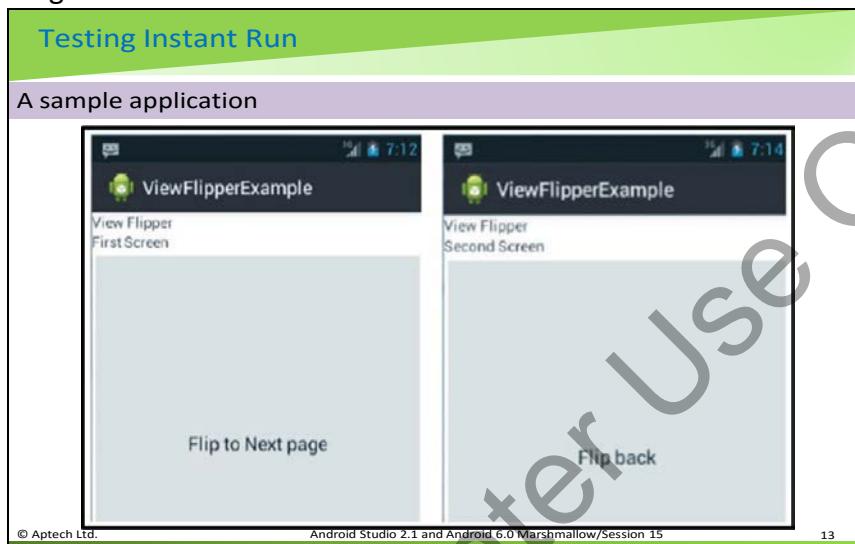
Continue to talk about Instant Run feature.



Explain that depending upon the extent and type of changes made to the app, Instant Run implements updates in three ways. In case changes are made to a method, the app will continue to run and use the changes next time. This process is called a hot swap. If the resources are modified, the app continues to run, but restarts the current activity. This is called warm swap. In case of structural code changes, such as to inheritance or method signatures, the app will restart but will not reinstall. This is called cold swap.

Slide 13

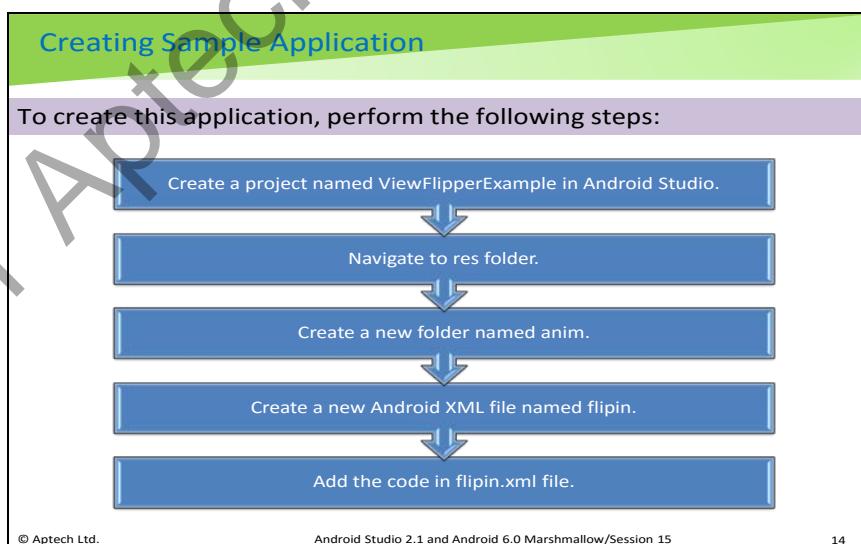
Talk about testing the Instant Run feature.



Explain that to test the Instant Run feature, let us create an application named ViewFlipperExample that has two screens as you see on the slide.

Slide 14

Explain that the application can be created in the steps shown on the slide.



Explain that first, create a project named ViewFlipperExample in Android Studio. Then, navigate to res folder. Next, create a new folder named anim. Then, create a new Android XML file named flipin. Finally, add the code in flipin.xml file.

Slide 15

Talk about adding the code in flipin.xml file using this code snippet.

Adding Code

Following code snippet shows how to add the code in flipin.xml file:

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:interpolator="@android:anim/decelerate_interpolator" >
    <translate
        android:duration="500"
        android:fromXDelta="-100%"
        android:toXDelta="0%" />
</set>
```

Slide 16

Continue with the steps to create an application. Say that the code snippet shows how to modify the code in flipout.xml file.

Modifying Code

To create this application, perform the following steps:

Create another new Android XML file named flipout under res → xml.

Modify the code in flipout.xml file.

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:interpolator="@android:anim/decelerate_interpolator" >
    <translate
        android:duration="500"
        android:fromXDelta="0%"
        android:toXDelta="100%" />
</set>
```

Tell the students that we will continue with the steps to create the sample application. Once we have added the code in flipin.xml file, we have to create another new Android XML file. For this demonstration, let us create a file called flipout. Next, we should modify the code in that xml file.

Slides 17 and 18

Continue with the steps to create an application. Say that the code snippet shows how to modify the code in activity_main.xml file.

Modifying code in activity_main.xml File 1-2

```

    graph TD
        A[Navigate to res → layout folder.] --> B[Modify the code in activity_main.xml file.]
    
```

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
    <TextView android:layout_width="fill_parent"
        android:layout_height="wrap_content" android:text="View Flipper"
        />
    <ViewFlipper android:id="@+id/viewflipper"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" >
        <LinearLayout android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:orientation="vertical" >
            <TextView android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="First Screen" />
        </LinearLayout>
    </ViewFlipper>
</LinearLayout>

```

© Aptech Ltd. Android Studio 2.1 and Android 6.0 Marshmallow/Session 15 17

Modifying code in activity_main.xml File 2-2

```

    graph TD
        A[Navigate to res → layout folder.] --> B[Modify the code in activity_main.xml file.]
    
```

```

        <?xml version="1.0" encoding="utf-8"?>
        <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:orientation="vertical" >
            <TextView android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Second Screen" /><Button
                android:id="@+id/button2"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:text="Flipback" />
        </LinearLayout>
    </ViewFlipper>
</LinearLayout>

```

© Aptech Ltd. Android Studio 2.1 and Android 6.0 Marshmallow/Session 15 18

Tell the students that we will continue with the steps to create the sample application. After modifying the code in flipout.xml file, we have to modify it in the activity_main.xml file. This code snippet shows how to modify code in activity_main.xml file. The code uses ViewFlipper widget.

Slides 19 to 21

Continue with the steps to create an application.

Adding Code in MainActivity.java File 1-3

Add the code into the MainActivity.java file.

```
package com.example.viewflipperexample;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.Button;
import android.widget.ViewFlipper;
public class MainActivity extends Activity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final ViewFlipper FlipIt = (ViewFlipper)
            findViewById(R.id.viewflipper);
    }
}
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

19

Adding Code in MainActivity.java File 2-3

```
Button flip1 = (Button) findViewById(R.id.button1);
Button flip2 = (Button) findViewById(R.id.button2);
Animation FlipIn = AnimationUtils.loadAnimation (this, R.anim.
flipin);
Animation FlipOut = AnimationUtils.loadAnimation(this, R.anim.
flipout);
FlipIt.setInAnimation(FlipIn);
FlipIt.setOutAnimation(FlipOut);
flip1.setOnClickListener(new Button. OnClickListener()
{
    @Override
    public void onClick(View arg0)
    {
        // TODO Auto-generated method stub
        FlipIt.showNext();
    }
});
}
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

20

Adding Code in MainActivity.java File 3-3

```

flip2.setOnClickListener(new Button.OnClickListener()
{
    @Override
    public void onClick(View arg0)
    {
        FlipIt.showPrevious();
    }
});
@Override
public boolean onCreateOptionsMenu(Menu menu)
{
    getMenuInflater().inflate(R.menu.main, menu);
    return true;
}
}

```

Tell the students that we will continue with the steps to create the sample application. After modifying code in activity_main.xml file, we have to add some code into the MainActivity.java file. The code to add in that file is displayed here.

Slide 22

Initiate the process of updating the code in the sample app.

Replacing Code Using Hot Swap

Replacing code for onClick () method

```

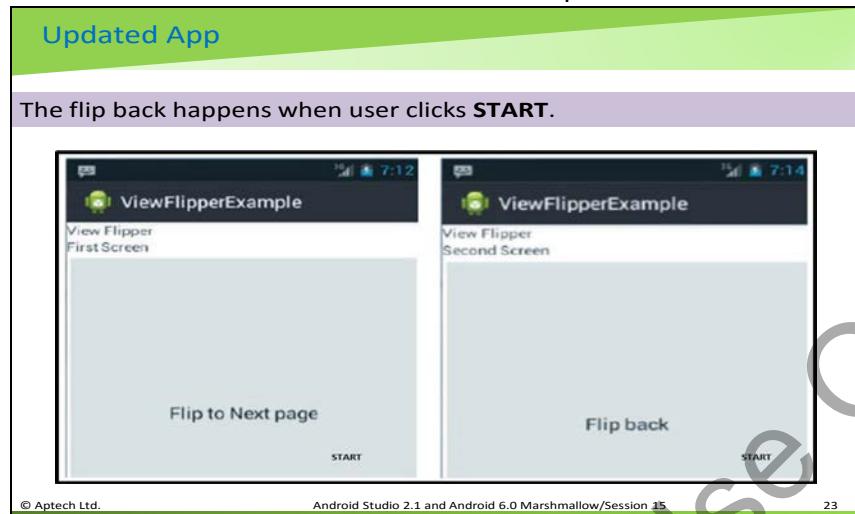
public void onClick(View v)
{
    //add code to display new message is displayed on selecting
    // the action button
    ...
}

```

Tell the students that let us test a hot swap update first. The code for the onClick() method is replaced with the code given on the slide.

Slide 23

Explain that adding the code will ensure that the flip takes place when the user clicks the START button. Inform that the flip back also happens when the user clicks the START button on the Flip back screen. Ask students to observe the output.



Explain that when the developer clicks the Run button, a toast will appear to notify that Instant Run applied code changes without activity restart. Instant Run helps to make the app layout and styling more efficient, because the developer can instantly see how the code changes result in UI changes. Let us look at an example using the same ViewFlipperExample app.

Slide 24

Give another example of updates that can be done to the app.

Changing App Background

To change the gray app background to pale yellow using warm swap method:

```
import android.view.View.OnClickListener;
import android.widget.Button;
import android.graphics.Color;
@Override
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_new_contact);
    Button button = (Button) findViewById(R.id.button);
    button.setOnClickListener(buttonListener);
    RelativeLayout colorLayout = (RelativeLayout) findViewById(R.id.colorLayout);
    colorLayout.setBackgroundColor(Color.YELLOW);
}
```

Explain that suppose you want to change the gray color of the app background to pale yellow, you can use this code snippet. After adding this code and clicking the Run button, the screen will flicker, and a toast will be displayed notifying the developer that the activity has restarted. The background color of the app will change to yellow without needing the full build and installation. Remember, as the activity has restarted, we will use warm swap process for updating.

Slide 25

Explain replacing code using cold swap update. Explain the code snippet.

Replacing Code Using Cold Swap

Following code shows the onCreate () method code replaced for cold swap update.

```
/ A complete restart is triggered by a cold swap.  
//Just add a new method after onCreate()  
//for a complete restart of currently running application  
public void demoMethod()  
{  
}  
/
```

Explain that for a cold swap update, the onCreate() method has to be replaced with different code.

Slide 26

Proceed to the next feature - Android Emulator.

Android Emulator Features

New Interface Features

- Resizable emulator
- Possibility of multi-touch actions
- Toolbar to provide easy-access controls
- Quick installation of APKs
- New features in GUI version

Tell the students that let us move on to learn about the next feature of Android Studio 2.1. The new Android Emulator provides the Instant Run feature. This feature helps to instantly view the appearance of the app as changes are made to the code. It also populates builds much faster as compared to previous versions. The apps can be developed at a faster rate on the official Android Emulator as compared to a real device.

When compared with the previous versions of android emulator, the latest Android Emulator is up to three times faster in terms of I/O, RAM and CPU, thereby, providing enhanced performance. In addition, ADB push speeds are at least 10 times faster, resulting in app being built at a much rapid rate.

The Android Emulator has an enhanced user interface. We will now learn the features of the new interface.

The emulator can be resized by dragging the corner of the emulator window.

The new emulator allows the developers to easily use the various multi-touch actions, such as zoom, pinch, pan, tilt, and rotate.

A new toolbar is added to the emulator which provides easy-access controls for the emulator, a screenshot button and a screen rotation button.

APKs can be quickly installed by simply dragging and dropping them.

Some extra features are now available in GUI version. For example, SMS Sending, Fingerprint, Phone Calling, and so on.

Add that one of the new Extended Control Features in the new Android emulator is the battery status and charging state.

Slide 27

Take an example of knowing battery status and how that can be displayed using this code snippet.

Knowing Battery Status

Following code snippet displays the battery status in percentage on screen:

```
package com.example.test;
import java.io.IOException;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.app.Activity;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.res.AssetFileDescriptor;
import android.widget.ProgressBar;
setContentView(R.layout.activity_main);
registerReceiver(mbcr,new IntentFilter(Intent.ACTION_BATTERY_CHANGED));
}
```

© Aptech Ltd. Android Studio 2.1 and Android 6.0 Marshmallow/Session 15 27

Explain that if we want to display the battery status in percentage on the app screen, we can use this code.

Slides 28 and 29

Explain how to display the battery status on screen by adding this code in XML file.

Adding Code to XML File 1-2

Following code is added to the XML file to display the battery status in percentage on screen:

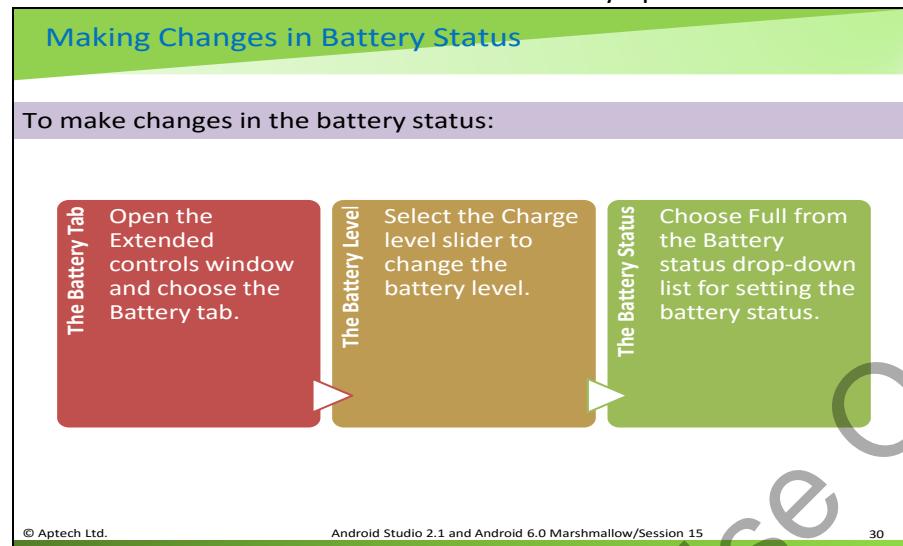
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:background="#abc" >
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text=""
        android:textSize="25sp"
        android:layout_gravity="center" />
```

Adding Code to XML File 2-2

```
<ProgressBar
    android:id="@+id/progressBar1"
    style="?android:attr/progressBarStyleHorizontal"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:minHeight="100dp"
    android:minWidth="200dp" />
</LinearLayout>
```

Slide 30

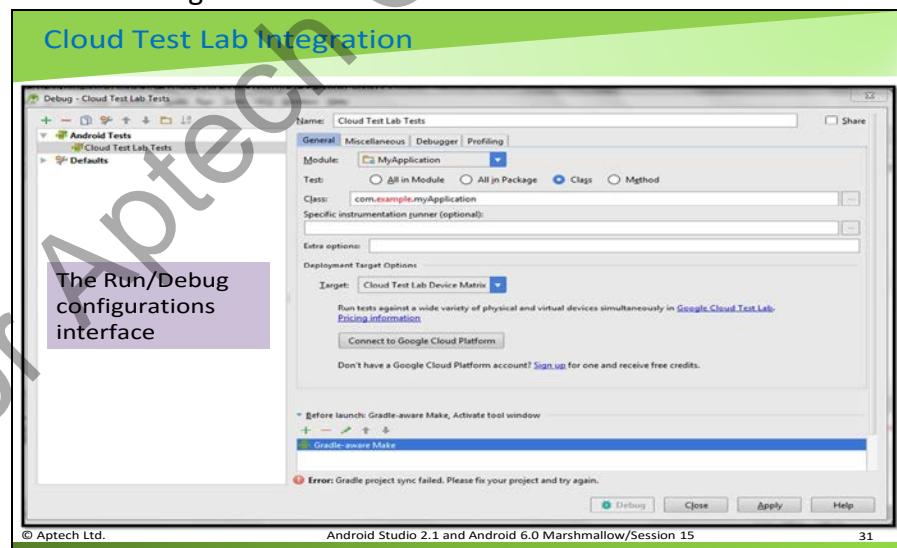
Explain the steps to make changes to the battery status. Say that developers can use the Location tab in the Extended controls window to manually update the device location.



Explain that to make changes in the battery status, first open the Extended controls window and choose the Battery tab. This tab includes various options for changing the battery status. Then, select the Charge level slider to change the battery level. There is an immediate reflection of the new battery status change in the status bar of the emulator. Finally, choose Full from the Battery status drop-down list for setting the battery status explicitly. The app detects the change and displays the new status immediately.

Slide 31

Explain cloud test lab integration.

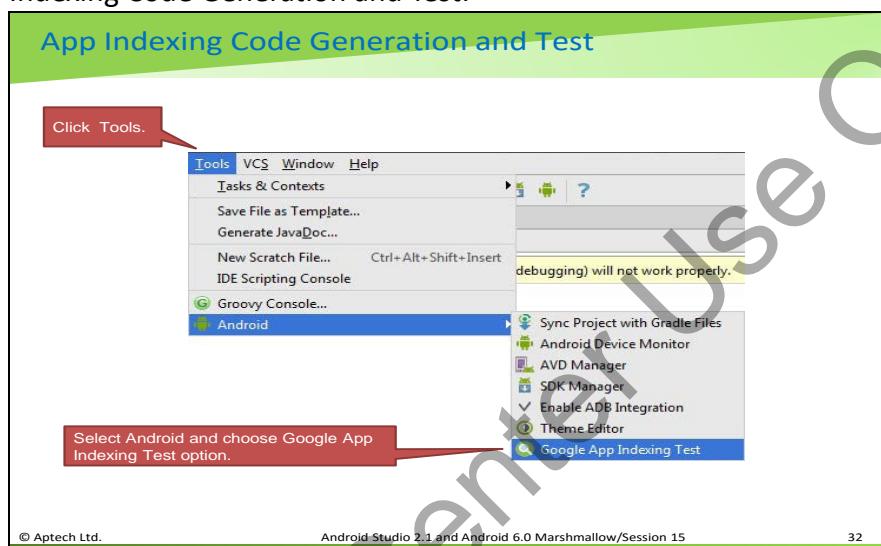


Explain that Google data centers host a wide range of device configurations and physical devices. A new service called Cloud Test Lab helps developers to test their apps across those device configurations and physical devices. It enables the developers to conduct a basic set of crash tests, directly from the Android Studio, even if they have not written tests explicitly for it.

The Run/Debug Configuration interface is a new interface that can be used to configure the portfolio of tests when the developers want to run it in the Cloud Test Lab, in Android. Ask students to observe the figure. Say that this interface also shows the results of the tests.

Slide 32

Explain App Indexing Code Generation and Test.



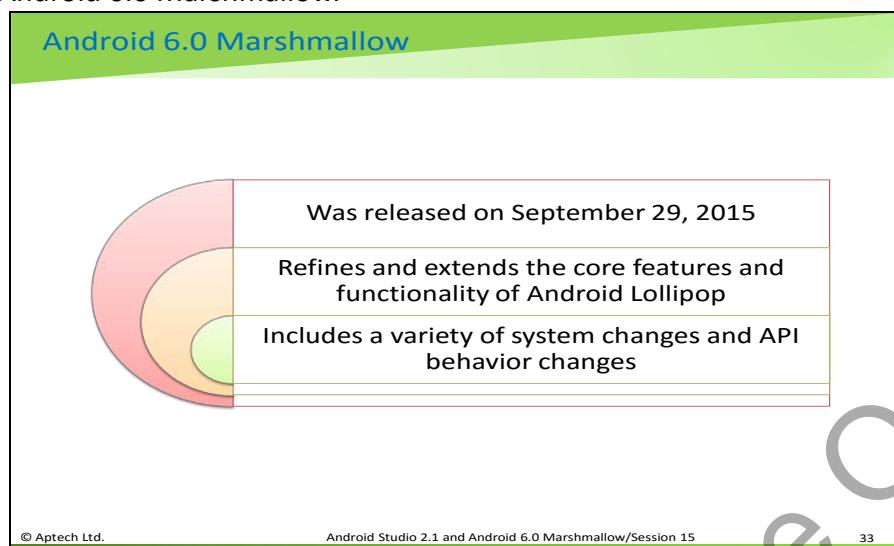
Tell the students that let us move on to the app indexing code generation and test feature of Android. An app can be made visible to users in Google Search with the help of the App Indexing API. To do this, you should add attributes to the `AndroidManifest.xml` file. The Google App Indexing service helps to list the app in the search results. This service directly works with the `AndroidManifest.xml` file.

Explain that developers can add the correct URL structure in their app code.

To validate the app indexing code, developers first need to click Tools. Next, they should Select Android and choose Google App Indexing Test option, as shown on the slide.

Slide 33

Talk about Android 6.0 Marshmallow.



The infographic has a green header bar with the text "Android 6.0 Marshmallow". Below it is a large stylized Android logo composed of overlapping red, yellow, and green circles. To the right of the logo is a vertical column of text in three sections:

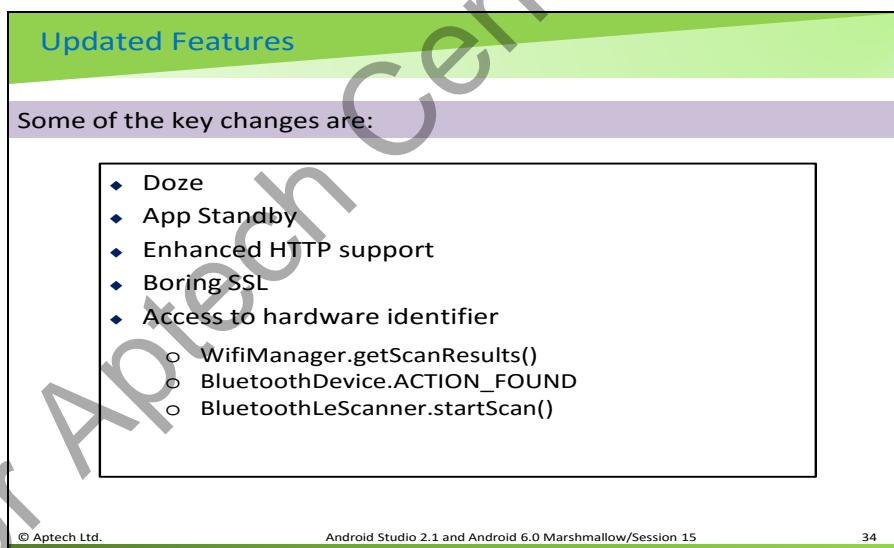
- Was released on September 29, 2015
- Refines and extends the core features and functionality of Android Lollipop
- Includes a variety of system changes and API behavior changes

At the bottom left is the copyright notice "© Aptech Ltd." and at the bottom center is "Android Studio 2.1 and Android 6.0 Marshmallow/Session 15". On the far right is the slide number "33".

Explain that Android 6.0 Marshmallow was released on September 29, 2015. It refines and extends the core features and functionality of Android Lollipop. However, it also includes a variety of system changes and API behavior changes.

Slide 34

Talk about the updated features of Android Marshmallow 6.0. Explain the features one by one.



The slide has a green header bar with the text "Updated Features". Below it is a purple bar with the text "Some of the key changes are:". To the right is a list of features in a box:

- Doze
- App Standby
- Enhanced HTTP support
- Boring SSL
- Access to hardware identifier
 - WifiManager.getScanResults()
 - BluetoothDevice.ACTION_FOUND
 - BluetoothLeScanner.startScan()

At the bottom left is the copyright notice "© Aptech Ltd." and at the bottom center is "Android Studio 2.1 and Android 6.0 Marshmallow/Session 15". On the far right is the slide number "34".

Explain that if a user leaves the device stationary, unplugged, and with its screen off, for a period of time, the device goes into Doze or sleep mode. Thus, power is saved during idle time. While in the doze mode, the device allows only app syncing and any pending system operations to take place.

App standby: The system disables network access and suspends syncs and jobs for any apps that the user has not used for some time. This helps to save battery as well as network bandwidth.

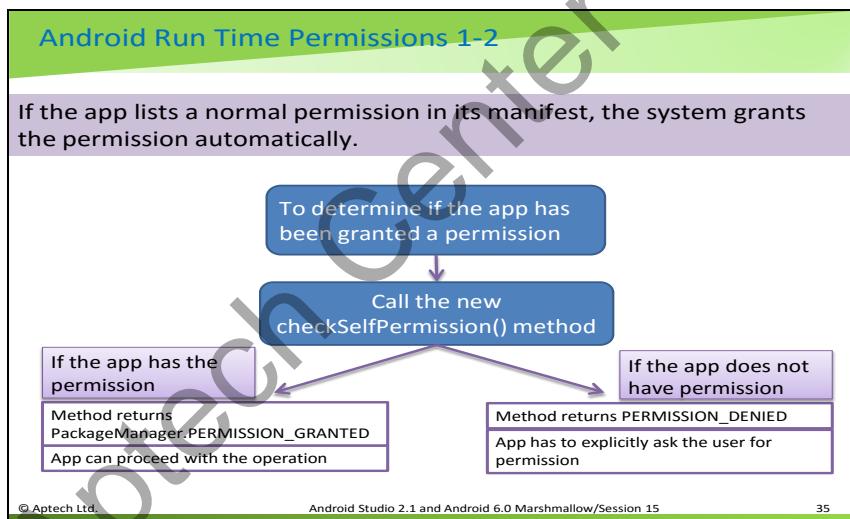
Enhanced HTTP support: Android 6.0 release replaces the support for the Apache HTTP client with a more efficient API using the HttpURLConnection class. This API reduces network use through transparent compression and response caching, and minimizes power consumption.

BoringSSL: Android 6.0 Marshmallow uses BoringSSL library instead of OpenSSL. The developer must either statically link against a cryptography library or modify the native code to call the Java cryptography APIs via JNI. This will avoid linking to libraries that may have changed or been removed and any security vulnerabilities that the device could be exposed to.

Access to hardware identifier: Marshmallow does not allow programmatic access to the device's local hardware identifier for apps using the Wi-Fi and Bluetooth APIs any longer. This protects the data in the device. If the app needs to access the hardware identifiers of other devices, it needs to have the ACCESS_FINE_LOCATION or ACCESS_COARSE_LOCATION permissions: WifiManager.getScanResults(), BluetoothDevice.ACTION_FOUND, and BluetoothLeScanner.startScan().

Slide 35

Talk about the new features of Android Marshmallow 6.0.



Explain that, in addition to the updates discussed so far, Android 6.0 Marshmallow has come with a variety of new features. Let us learn about few important ones here.

Talk about Android Run Time Permissions.

Explain that Android 6.0 Marshmallow enables the users to directly manage app permissions at runtime. Users can now grant or revoke permissions individually for installed apps. Apps that target Android 6.0 or higher should be tested under the new permissions model.

Explain the specifications as per the new permission model.

Slide 36

Continue to talk about the new features of Android Marshmallow 6.0.

Android Run Time Permissions 2-2

- ◆ Users should give explicit approval to the app for access to confidential data.
- ◆ The new requestPermissions() method can be used to request permissions for the app at runtime.

Explain that if the app asks for dangerous permission, that is, access to the user's confidential data, the user has to explicitly give approval to the app. To request permissions for the app at runtime, the developer must call the new requestPermissions() method.

Slides 37 and 38

Continue to talk about the new features of Android Marshmallow 6.0. Android Run Time Permissions.

Checking App's Permission to Data Access 1-2

Following code checks if the app has permission to read user's contacts and requests the permission:

```
// Here, thisActivity is the current activity
if (ContextCompat.checkSelfPermission(thisActivity, Manifest.permission.READ_CONTACTS) != PackageManager.PERMISSION_GRANTED)
{
    /* Check whether explanation needs to be shown */
    if(ActivityCompat.shouldShowRequestPermissionRationale(thisActivity, Manifest.permission.READ_CONTACTS))
    {
        /* Show an explanation to the user and wait for the user's
        response
        After user has noticed the explanation, request permission
        again.*/
    }
}
```

Checking App's Permission to Data Access 2-2

Following code checks if the app has permission to read user's contacts and requests the permission:

```
else
{
    // Request permission directly without explanation.
    ActivityCompat.requestPermissions(thisActivity, new
        String[]{Manifest.permission.READ_CONTACTS}, GRANT_REQUESTS_TO_
        READ_CONTACTS);
    /* GRANT_REQUESTS_TO_READ_CONTACTS is an app-defined int
    constant.
    The callback method gets the result of the request. */
}
/* When user responds, the system passes response to the app's
onRequestPermissionsResult() method. */
@Override
public void onRequestPermissionsResult(int requestCode, String
    accessRights[], int[] accessResults)
{
    switch (requestCode)
    {
        case GRANT_REQUESTS_TO_READ_CONTACTS:
            if (accessResults[0] == PackageManager.PERMISSION_GRANTED)
                ...
    }
}
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

38

Explain that this code snippet checks if the app has permission to read user's contacts and requests the permission if necessary. Note that the app cannot configure or alter the dialog box that is shown to the user when the app calls `requestPermissions()` method. If the developers need to provide any information or explanation to the user, they should do that before calling the `requestPermissions()` method.

Slide 39

Talk about the next new feature of Android Marshmallow 6.0 - Android Data Binding Library.

Android Data Binding Library

To configure the app to use data binding:

Download the library from the Support repository in the Android SDK manager.

Add the `dataBinding` element to the `build.gradle` file in the app module.

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

39

Explain that data Binding Library is a support library that helps developers to build rich and responsive user layouts and experiences with minimal effort. These libraries have pre-defined sets of boilerplate code for common layouts and functionality, which an app developer can use to plug those features in their apps with minimal code. Developers can use the Data Binding Library with all Android platform versions back to Android 2.1 (API level 7+). To use data binding, the developers will require Android Plugin for Gradle 1.5.0-alpha1 or higher. Inform that to configure the app to use data binding, the developer must perform these steps. Read out the steps.

Slide 40

This is the third step of the process covered on slide 39. Explain with code snippet configuring data binding.

Configuring Data Binding

To configure the app to use data binding:

Use the code snippet to configure data binding.

```
android
{
    ...
    dataBinding
    {
        enabled = true
    }
}
```

© Aptech Ltd. Android Studio 2.1 and Android 6.0 Marshmallow/Session 15 40

Tell the students that let us see the next step to configure the app to use data binding.

Slide 41

Talk about how to show the layout file without data binding.

Layout File without Data Binding

Following code snippet shows layout file without data binding.

```
<LinearLayout . . . . >
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="DataBound"
    android:id="@+id/dataBoundText"
    android:layout_centerVertical="true"
    android:layout_centerHorizontal="true"/>
</LinearLayout>
```

© Aptech Ltd. Android Studio 2.1 and Android 6.0 Marshmallow/Session 15 41

Explain that this code snippet can be used to show layout file without data binding.

Slide 42

Continue with code snippet explanation.

Adding DataBindingHelper Class

Following code snippet shows adding of DataBindingHelper class to the project:

```
package com.databindingexample;
public class DataBindingHelper
{
    private String information;
    public static DataBindingHelper get(String information) {
        return new DataBindingHelper(information);
    }
    public DataBindingHelper(String information)
    {
        this.information = information;
    }
    public String getMessage()
    {
        return String.format("Welcome %s", information);
    }
}
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

42

Explain that to configure data binding, the developer needs to add a new class to the project called DataBindingHelper, as shown in the code snippet. This class will bind elements with data.

Slide 43

Talk about the data binding library and the layout files.

Layout Files 1-2

Following code snippet shows the layout files will have the root tag of layout:

```
<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/
    android"
    android:id="@+id/Layout01"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
<data>
<variable
    name="databindingidentifier"
    type="com.databindingexample"/>
</data>
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@{databindingidentifier.message}"
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

43

Explain that the Data binding library is used. The layout files will have the root tag of layout and this is accompanied by a view root element and data element, as shown in code snippet.

Slide 44

Continue with code snippet explanation.

Layout Files 2-2

```
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
    />
</LinearLayout>
```

© Aptech Ltd. Android Studio 2.1 and Android 6.0 Marshmallow/Session 15 44

Explain that the androidmanifest.xml must declare the activities. Data Binding allows a developer to write expressions for handling events that are initiated from the layout; for example, one such event is onClick().

Slide 45

Explain the ways to handle an event. Read the first way, Method References.

Method References

Following code snippet shows how method references can be used:

```
<Button ...
    android:onClick="@{isTrainer ? handler.trainerClick :
    handler.studentClick}" />
// First search the setOnClickListener() method whenever event is fired by
// clicking on the button.
@BindingMethods
({@BindingMethod(type = View.class, attribute = "android:onClick",
method = "setOnClickListener")})
// Search the OnClickListener for abstract method
void onClick(View v);
// Search the View for setOnClickListener
void setOnClickListener(View.OnClickListener l)
```

© Aptech Ltd. Android Studio 2.1 and Android 6.0 Marshmallow/Session 15 45

Explain that in expressions, a developer can reference methods that conform to the signature of the listener method. For example, android:onClick() can be assigned to a method in an Activity. The expression is processed at compile time, so if the method does not exist or its signature is not correct, a compile-time error is thrown. The parameters of the method must match the parameters of the event listener. When an expression evaluates to a method reference, Data Binding wraps the method reference and owner object in a listener, and sets that listener on the target view. If the expression evaluates to null, Data Binding sets a null listener.

This code snippet shows how method references can be used. The code needs to be added to the XML file of the activity where the button is present.

Slide 46

Explain the ways to handle an event. Read the second way, Listener Bindings.

Listener Bindings

Following code snippet shows how to modify the code to use `onClickListener` to handle events:

```
static class Implementation1OnClickListener implements
    OnClickListener
{
    public Handler mouseHandler;
    @Override
    public void onClick(android.view.View arg0)
    {
        mouseHandler.trainerClick(arg0);
    }
}
static class Implementation2OnClickListener implements
    OnClickListener
{
    public Handler mouseHandler;
    @Override
    public void onClick(android.view.View arg0)
    {
        mouseHandler.studentClick(arg0);
    }
}
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

46

Explain that unlike in case of method references, Data Binding always creates a listener and sets on the view. When the event is dispatched, the listener evaluates the binding or lambda expression. The return value must match the expected return value of the listener, unless it is expecting void. This code snippet shows how to modify the code to use `onClickListener` to handle events.

Slide 47

Explain Finger Print and Payments.

Fingerprint Authentication for Payments

To implement fingerprint scan-based authentication feature:

Ensure that the app is running with a fingerprint sensor.

Include the standard Android fingerprint icon in the UI.

Add the `USE_FINGERPRINT` permission in the manifest.

```
<uses-permission
    android:name="android.permission.USE_FINGERPRINT" />
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

47

Explain that these APIs help developers to incorporate fingerprint scan-based authentication and accelerate payments and sign-in operations in their app.

Let us assume that developers want to include the Fingerprint feature in the app so that users can use fingerprint authentication for making an online payment. For this, they have to ensure that the app is running on a compatible device with a fingerprint sensor. Secondly, they should include the standard Android fingerprint icon in the UI. And finally, they should add the `USE_FINGERPRINT` permission in the manifest, as shown in the code snippet.

Slides 48 to 63

Explain with code snippet fingerprint authentication for making online purchases.

Fingerprint Authentication for Online Purchases 1-16

Following code snippet shows how to implement fingerprint authentication for making online purchases:

```
package com.example.android.fingerprintdialog;
import android.app.Activity;
import android.app.DialogFragment;
import android.content.SharedPreferences;
import android.hardware.fingerprint.FingerprintManager;
import android.os.Bundle;
import android.view.KeyEvent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.view.inputmethod.EditorInfo;
import android.view.inputmethod.InputMethodManager;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.ImageView;
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

48

Fingerprint Authentication for Online Purchases 2-16

```
import android.widget.TextView;
import javax.inject.Inject;
public class FingerprintAuthenticationDialogFragment extends
DialogFragment
implements TextView.OnEditorActionListener, FingerprintUiHelper.
Callback {
private Button mCancelButton;
private Button mSecondDialogButton;
private View mFingerprintContent;
private View mBackupContent;
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

49

Fingerprint Authentication for Online Purchases 3-16

```
private EditText mPassword;
private CheckBox mUserfingerprintFutureCheckBox;
private TextView mPasswordDescriptionTextView;
private TextView mNewFingerprintEnrolledTextView;
private Stage mStage = Stage.FINGERPRINT;
private FingerprintManager.CryptoObject mCryptoObject;
private FingerprintUiHelper mFingerprintUiHelper;
private MainActivity mActivity;
@Inject FingerprintUiHelper.FingerprintUiHelperBuilder
mFingerprintUiHelperBuilder;
@Inject InputMethodManager mInputMethodManager;
@Inject SharedPreferences mSharedPreferences;
@Inject
public FingerprintAuthenticationDialogFragment() {}
@Override
public void onCreate(Bundle savedInstanceState) {
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

50

Fingerprint Authentication for Online Purchases 4-16

```
// During re-instantiated of activity do not a create a new
fragment
setRetainInstance(true);
super.onCreate(savedInstanceState);
setStyle(DialogFragment.STYLE_NORMAL, android.R.style.
Theme_Material_Light_Dialog);
}
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup
container,
Bundle savedInstanceState) {
getDialog().setTitle(getString(R.string.sign_in));
View v = inflater.inflate(R.layout.fingerprint_dialog_container,
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

51

Fingerprint Authentication for Online Purchases 5-16

```
container, false);
mCancelButton = (Button) v.findViewById(R.id.cancel_
button);
mCancelButton.setOnClickListener(new View.
OnClickListener() {
@Override
public void onClick(View view) {
dismiss();
}
});
mSecondDialogButton = (Button) v.findViewById(R.id.second_
dialog_button);
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

52

Fingerprint Authentication for Online Purchases 6-16

```
mSecondDialogButton.setOnClickListener(new View.
OnClickListener() {
@Override
public void onClick(View view) {
if (mStage == Stage.FINGERPRINT) {
goToBackup();
} else {
verifyPassword();
}
});
mFingerprintContent = v.findViewById(R.id.fingerprint_
container);
mBackupContent = v.findViewById(R.id.backup_container);
mPassword = (EditText) v.findViewById(R.id.password);
mPassword.setOnEditorActionListener(this);
mPasswordDescriptionTextView = (TextView) v.findViewById(R.
id.password_description);
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

53

Fingerprint Authentication for Online Purchases 7-16

```
mUseFingerprintFutureCheckBox = (CheckBox)
v.findViewById(R.id.use_fingerprint_in_future_check);
mNewFingerprintEnrolledTextView = (TextView)
v.findViewById(R.id.new_fingerprint_enrolled_
description);
mFingerprintUiHelper = mFingerprintUiHelperBuilder.
build(
(ImageView) v.findViewById(R.id.fingerprint_icon),
(TextView) v.findViewById(R.id.fingerprint_
status), this);
updateStage();
// If fingerprint authentication is not available, switch
immediately to the backup
// (password) screen.
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

54

Fingerprint Authentication for Online Purchases 8-16

```
if (!mFingerprintUiHelper.isFingerprintAuthAvailable())
{
    goToBackup();
}
return v;
}
@Override
public void onResume() {
    super.onResume();
    If (mStage == Stage.FINGERPRINT) {
        mFingerprintUiHelper.startListening(mCryptoObject);
    }
}
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

55

Fingerprint Authentication for Online Purchases 9-16

```
public void setStage(Stage stage) {
    mStage = stage;
}
@Override
public void onPause() {
    super.onPause();
    mFingerprintUiHelper.stopListening();
}
@Override
public void onAttach(Activity activity) {
    super.onAttach(activity);
    mActivity = (MainActivity) activity;
}
/***
 * Sets the crypto object to be passed in when authenticating
with fingerprint.
*/
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

56

Fingerprint Authentication for Online Purchases 10-16

```
public void setCryptoObject(FingerprintManager.CryptoObject
cryptoObject) {
    mCryptoObject = cryptoObject;
}
// After Unsuccessful attempt for authenticating lets switch to
the other authentication method
private void goToBackup() {
    mStage = Stage.PASSWORD;
    updateStage();
    mPassword.requestFocus();
    // Enable keyboard.
    mPassword.postDelayed(mShowKeyboardRunnable, 500);
    // Fingerprint is not used anymore. Stop listening for it.
    mFingerprintUiHelper.stopListening();
}
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

57

Fingerprint Authentication for Online Purchases 11-16

```
//After Successful authentication dismisses the dialog
private void verifyPassword() {
    if (!checkPassword(mPassword.getText().toString())) {
        return;
    }
    if (mStage == Stage.NEW_FINGERPRINT_ENROLLED) {
        Sharedpreferences.Editor editor = mSharedpreferences.
        edit();
        editor.putBoolean(getString(R.string.use_fingerprint_
        to_authenticate_key),
        mUseFingerprintFutureCheckBox.isChecked());
        editor.apply();
        if (mUseFingerprintFutureCheckBox.isChecked()) {
            // Re-create the key so that fingerprints including
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

58

Fingerprint Authentication for Online Purchases 12-16

```
new ones are validated.
mActivity.createKey();
mStage = Stage.FINGERPRINT;
}
}
mPassword.setText("");
mActivity.onPurchased(false /* without Fingerprint */);
dismiss();
}
/**
 * @return true if {@code password} is correct, false otherwise
 */
private boolean checkPassword(String password) {
    // Authentication process needs to be verified at server side.
    return password.length() > 0;
}
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

59

Fingerprint Authentication for Online Purchases 13-16

```
private final Runnable mShowKeyboardRunnable = new Runnable() {
@Override
public void run() {
mInputMethodManager.showSoftInput(mPassword, 0);
}
};
private void updateStage() {
switch (mStage) {
case FINGERPRINT:
mCancelButton.setText(R.string.cancel);
mSecondDialogButton.setText(R.string.use_password);
mFingerprintContent.setVisibility(View.VISIBLE);
mBackupContent.setVisibility(View.GONE);
break;
case NEW_FINGERPRINT_ENROLLED:
// Intentional fall through
case PASSWORD:
```

Fingerprint Authentication for Online Purchases 14-16

```
mCancelButton.setText(R.string.cancel);
mSecondDialogButton.setText(R.string.ok);
mFingerprintContent.setVisibility(View.GONE);
mBackupContent.
setVisibility(View.VISIBLE);
if (mStage == Stage.NEW_FINGERPRINT_ENROLLED) {
mPasswordDescriptionTextView.
setVisibility(View.GONE);
mNewFingerprintEnrolledTextView.
setVisibility(View.VISIBLE);
mUseFingerprintFutureCheckBox.
setVisibility(View.VISIBLE);
}
break;
}
}
```

Fingerprint Authentication for Online Purchases 15-16

```
@Override
public boolean onEditorAction(TextView v, int actionId,
KeyEvent event) {
    if (actionId == EditorInfo.IME_ACTION_GO) {
        verifyPassword();
        return true;
    }
    return false;
}
@Override
public void onAuthenticated() {
    // After the authentication was successfull, call the
    UiHelper
    mActivity.onPurchased(true /* withFingerprint */);
    dismiss();
}
```

Fingerprint Authentication for Online Purchases 16-16

```
@Override
public void onError() {
    goToBackup();
}
//Types of Authentication Methods which are used to authenticate
the user.
public enum Stage {
    FINGERPRINT,
    NEW_FINGERPRINT_ENROLLED,
    PASSWORD
}
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

63

Explain that developers can use this code to implement fingerprint authentication during online purchases. This code will be covered in the further few slides.

When allowing the users to make payments through the app using Fingerprint authentication, the developers may want to include an additional security feature. The developers can have their apps authenticating users based on how recently they last unlocked their device by implementing a public or secret key for user authentication. To set the timeout duration for which the same key can be re-used after a user is successfully authenticated, call the new setUserAuthenticationValidityDurationSeconds() method when setting up a KeyGenerator or KeyPairGenerator.

After the timeout expires, use the createConfirmDeviceCredentialIntent() method to re-authenticate the user within the app.

Slide 64

Explain the confirm credential functionality.

Confirm Credential Functionality

Following code snippet shows inclusion of Confirm Credential Functionality:

```
private void showAuthenticationScreen()
{
    // Description and Title are customizable, this
    // creates the Confirm Credentials screen Or if it is left
    null,
    // a generic one shall be provided
    Intent i = mKeyguardManager.createConfirmDeviceCredentialIntent(null, null);
    if (i != null)
    { startActivityForResult(i, REQUEST_CODE_CONFIRM_DEVICE_
    CREDENTIALS);
    }
}
```

© Aptech Ltd.

Android Studio 2.1 and Android 6.0 Marshmallow/Session 15

64

Explain that this code snippet shows the inclusion of Confirm Credential Functionality.

Slide 65

Explain the next feature Direct Share.

Direct Share 1-2

To enable direct share targets, developers need to perform following steps:

1. Define a class that extends the ChooserTargetService class. Declare this service in the manifest, as shown in code snippet.

```
<service android:name=".ChooserTargetService" android:label="@string/service_name" android:permission="android.permission.BIND_CHOOSER_TARGET_SERVICE">
<intent-filter> <action android:name="android.service.chooser.ChooserTargetService" />
</intent-filter>
</service>
```

© Aptech Ltd. Android Studio 2.1 and Android 6.0 Marshmallow/Session 15 65

Explain that Marshmallow provides the ability to define direct share targets, such as contacts within other apps. This feature allows users to share content with a friend or community defined in another app, such as a social network app. Read the steps and explain code snippet that shows how to define a class that extends the ChooserTargetService class.

Slide 66

Explain the next step.

Direct Share 2-2

2. For each activity that needs to be exposed to ChooserTargetService, add a <metadata& gt; element with the name "android.service.chooser.chooser_target_service" in the app manifest, as shown in the code snippet.

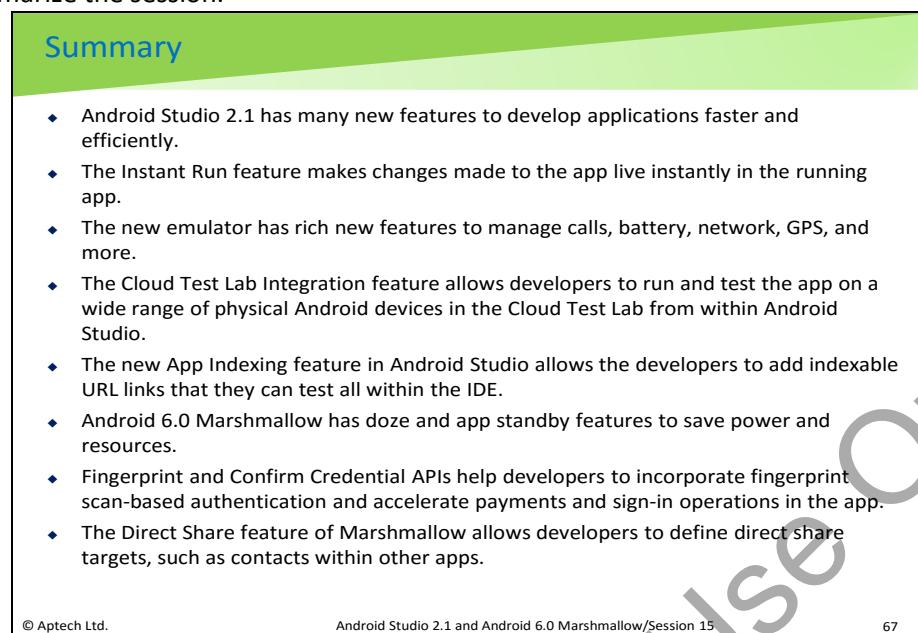
```
<activity android:name=".MyShareActivity"
    android:label="@string/share_activity_label">
    <intent-filter>
        <action android:name="android.intent.action.SEND" />
    </intent-filter>
    <meta-data
        android:name="android.service.chooser.chooser_target_
        service"
        android:value=".ChooserTargetService" />
</activity>
```

© Aptech Ltd. Android Studio 2.1 and Android 6.0 Marshmallow/Session 15 66

Explain that for each activity that needs to be exposed to ChooserTargetService, add a <metadata& gt; element with the name "android.service.chooser.chooser_target_service" in the app manifest, as shown in the code snippet.

Slide 67

Let us summarize the session.



The slide has a green header bar with the word "Summary" in white. The main content area contains a bulleted list of features for Android Studio 2.1 and Android 6.0 Marshmallow. At the bottom, there is a footer bar with the text "© Aptech Ltd.", "Android Studio 2.1 and Android 6.0 Marshmallow/Session 15", and the number "67". A large watermark "FOR APTECH CENTER USE ONLY" is diagonally across the slide.

- ◆ Android Studio 2.1 has many new features to develop applications faster and efficiently.
- ◆ The Instant Run feature makes changes made to the app live instantly in the running app.
- ◆ The new emulator has rich new features to manage calls, battery, network, GPS, and more.
- ◆ The Cloud Test Lab Integration feature allows developers to run and test the app on a wide range of physical Android devices in the Cloud Test Lab from within Android Studio.
- ◆ The new App Indexing feature in Android Studio allows the developers to add indexable URL links that they can test all within the IDE.
- ◆ Android 6.0 Marshmallow has doze and app standby features to save power and resources.
- ◆ Fingerprint and Confirm Credential APIs help developers to incorporate fingerprint scan-based authentication and accelerate payments and sign-in operations in the app.
- ◆ The Direct Share feature of Marshmallow allows developers to define direct share targets, such as contacts within other apps.

Using this slide, summarize the key points of this session. Explain the following points in brief:

- Android Studio 2.1 has many new features to develop applications faster and more efficiently.
- The Instant Run feature makes changes made to the app live instantly in the running app.
- The new emulator has rich new features to manage calls, battery, network, GPS, and more.
- The Cloud Test Lab Integration feature allows developers to run and test the app on a wide range of physical Android devices in the Cloud Test Lab from within Android Studio.
- The new App Indexing feature in Android Studio allows the developers to add indexable URL links that they can test all within the IDE.
- Android 6.0 Marshmallow has doze and app standby features to save power and resources.
- Fingerprint and Confirm Credential APIs help developers to incorporate fingerprint scan-based authentication and accelerate payments and sign-in operations in the app.
- The Direct Share feature of Marshmallow allows developers to define direct share targets, such as contacts within other apps.

Next, let us check your understanding of the topics covered so far in this session.

15.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session.

Session 16: Material Design**16.1. Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of this session in-depth. Prepare a question or two that will be a key point to relate the current session objectives.

16.1.1 Objectives

By the end of this session, learners will be able to:

- Explain material design
- Describe the material design environment elements
- Describe various material design elements

16.1.2 Teaching Skills

To teach this session, you should be well versed with the concepts of Android application development, material design. Say that this session explains the basic concepts of material design. It describes the various material design elements such as themes, list and cards, view shadows, and animations.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of students by asking questions in between the class.

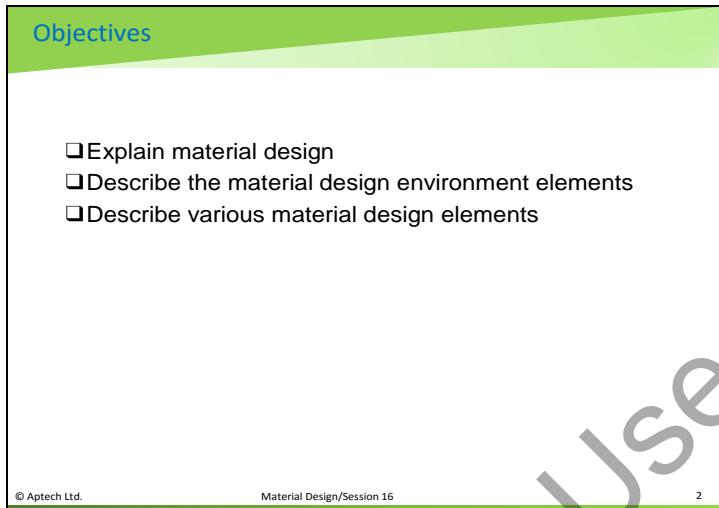
16.1.3 In-Class Activities

Follow the order given here during In-Class activities.

Overview of the Session

Give the students an overview of the current session in the form of session objectives. Show the students slide 2 of the presentation.

Slide 2



The slide has a green header bar with the word "Objectives" in white. The main content area contains a bulleted list of three items:

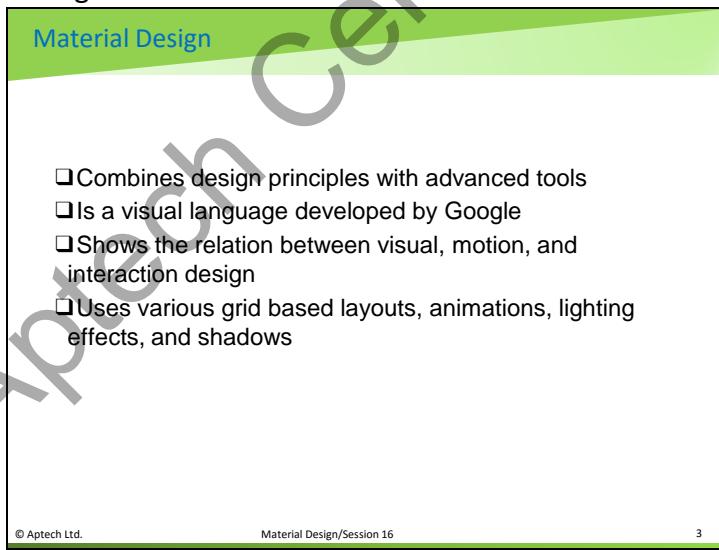
- ❑ Explain material design
- ❑ Describe the material design environment elements
- ❑ Describe various material design elements

At the bottom left is the copyright notice "© Aptech Ltd.", in the center is "Material Design/Session 16", and at the bottom right is the number "2".

16.2 In-Class Explanations

Slide 3

Talk about material design.



The slide has a green header bar with the word "Material Design" in blue. The main content area contains a bulleted list of five items:

- ❑ Combines design principles with advanced tools
- ❑ Is a visual language developed by Google
- ❑ Shows the relation between visual, motion, and interaction design
- ❑ Uses various grid based layouts, animations, lighting effects, and shadows

At the bottom left is the copyright notice "© Aptech Ltd.", in the center is "Material Design/Session 16", and at the bottom right is the number "3".

Explain that material design combines good and classic design principles with some advanced tools to create a single underlying system across various platforms and mobile devices. It is also considered as a visual language or design language developed by Google for the users. It explains the relation between visual, motion, and interaction design. Material design uses various grid based layouts, animations, lighting effects, and shadows.

Slide 4

Read out the table given on the slide that lists the material design environment elements.

Introduction to Material and Material Environment					
❑ Material design	<ul style="list-style-type: none"> ◦ Has a 3-D environment ◦ Contains light, material, and cast shadows 				
❑ Material design environment elements include:					
<table border="1"> <thead> <tr> <th>Objects</th><th>Shadows</th></tr> </thead> <tbody> <tr> <td> <ul style="list-style-type: none"> • Have three dimensions - x,y, and z. • Material sheets within the objects have a standard thickness of 1dp. </td><td> <ul style="list-style-type: none"> • Virtual lights illuminate the material environment. • Difference in elevations between overlapping material objects creates shadows. • The key lights create directional shadows; ambient lights create soft shadows. </td></tr> </tbody> </table>		Objects	Shadows	<ul style="list-style-type: none"> • Have three dimensions - x,y, and z. • Material sheets within the objects have a standard thickness of 1dp. 	<ul style="list-style-type: none"> • Virtual lights illuminate the material environment. • Difference in elevations between overlapping material objects creates shadows. • The key lights create directional shadows; ambient lights create soft shadows.
Objects	Shadows				
<ul style="list-style-type: none"> • Have three dimensions - x,y, and z. • Material sheets within the objects have a standard thickness of 1dp. 	<ul style="list-style-type: none"> • Virtual lights illuminate the material environment. • Difference in elevations between overlapping material objects creates shadows. • The key lights create directional shadows; ambient lights create soft shadows. 				
<small>© Aptech Ltd. Material Design/Session 16 4</small>					

Let us learn about the basic concepts of material design. Let us learn about material and material environment in material design. Material design has a three-dimensional environment. It contains light, material, and cast shadows.

Slide 5

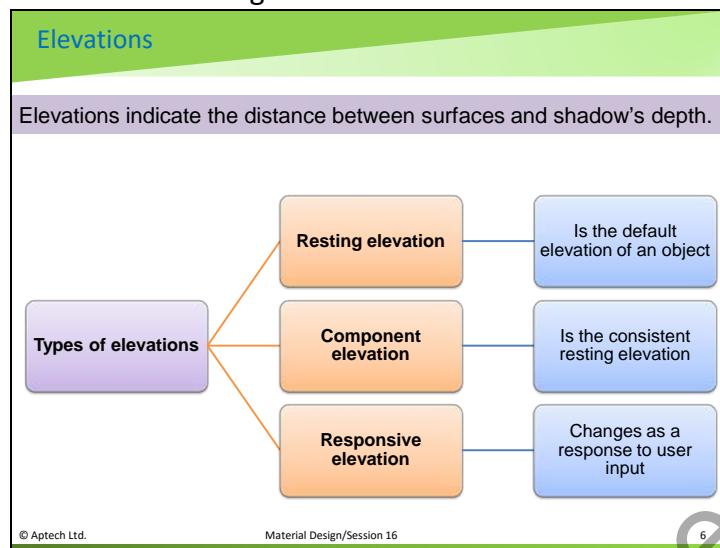
Let us understand built-in characteristics of materials.

Built-in Characteristics of Materials			
Solid by nature	Cannot be penetrated	Can easily join with other materials in the environment	
Move along any axis in the plane	Have varying x and y dimensions		Can be created and destroyed
Never bend or fold	Have uniform thickness of 1 dp		
Can change their shape	Can cast shadows		

Explain that materials have certain built-in characteristics. They are solid by nature. They move along any axis in the plane. They never bend or fold. They can change their shape. They can cast shadows. They have uniform thickness of 1 dp. They have varying x and y dimensions. They cannot be penetrated. They can easily join with other materials in the environment. They can be created and destroyed.

Slide 6

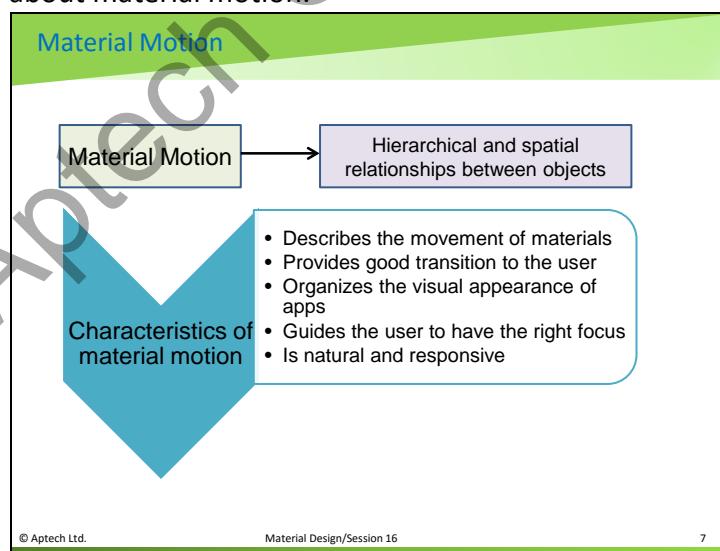
Talk about elevations in material design.



Explain that material design objects or elements possess elevations. Elevations indicate the distance between surfaces and shadow's depth. There are different types of elevations in material objects. The first type of elevation is the resting elevation. It is the default elevation of an object that does not change under any circumstance. The second type of elevation is the component elevation. It is the consistent resting elevation maintained by the components. Finally, the third type of elevation is the responsive elevation. This type of elevation changes as a response to user input.

Slide 7

Let us understand about material motion.



Explain that material motion describes hierarchical and spatial relationships between objects. Explain that material motion describes the movement of materials. It provides a good transition to the user. It organizes the visual appearance of apps. It guides the user to have the right focus. Also, it is natural and responsive. Explain that the developers can use easing curves such as standard curve, deceleration curve, acceleration curve, and sharp curve to create smooth and consistent motion in materials.

Slides 8 to 11

Proceed to talk about the style elements that can be used in material design to customize the apps.

Style 1-4

Following table lists the style elements that can be used in material design to customize the apps:

Element	Content	Description
Color	• Color schemes	<ul style="list-style-type: none"> • Primary color • Secondary color • Accent color • Text and background colors
	• Themes	<ul style="list-style-type: none"> • Light theme • Dark theme

© Aptech Ltd. Material Design/Session 16 8

Style 2-4

Element	Content	Description
Icons	• Product icons	<ul style="list-style-type: none"> • Design approach • Product icon grid • Keyline shapes • DP unit grid • Geometry
	• System icons	<ul style="list-style-type: none"> • Design principles • Grid, proportion, and size • Icon grid • Content area • Dense layouts • Keyline shapes

© Aptech Ltd. Material Design/Session 16 9

Style 3-4

Element	Content	Description
Imagery	• Principles	<ul style="list-style-type: none"> • Personal relevance • Information • Delight • Dynamic and context-relevant
	• Best Practices	<ul style="list-style-type: none"> • Multiple mediums • No stock photos or clipart • Iconic point of focus in imagery • Immersive story
	• UI Integration	<ul style="list-style-type: none"> • Resolution • Scale • Text protection • Avatars and thumbnails • Hero Images • Gallery Images

© Aptech Ltd. Material Design/Session 16 10

Style 4-4

Element	Content	Description
Typography	<ul style="list-style-type: none"> • Roboto • Nato 	<ul style="list-style-type: none"> • App bar • Buttons • Subheading • Body 1 • Text contrast ratios
Writing	<ul style="list-style-type: none"> • Clear • Easy to understand • Concise • Accurate text 	<ul style="list-style-type: none"> • Tone • Capitalization • Punctuation • UI button text • Guidelines

© Aptech Ltd.

Material Design/Session 16

11

Read out the table given on the slides 8 to 11 for students and explain.

Slides 12 and 13

Talk about the layout elements that can be used in material design.

Layout 1-2

Following table lists the layout elements that can be used in material design:

Element	Description
Seam	Two sheets of material sharing a common edge is called a seam.
Step	Two overlapping sheets of material with different depths is called a step.
Floating action button	Circular sheet that is separate from a toolbar is a floating action button.
Pixel density	Number of pixels that can accommodate in an inch is pixel density.
Density-independent pixels(dp)	Uniform display of user interface components on screen having different densities is termed as density-independent pixels.

© Aptech Ltd.

Material Design/Session 16

12

Layout 2-2

Element	Description
Metrics and keylines	Includes the baseline grids, ratio keylines, sizing by increments, and touch target size for the screen elements.
Structure	Includes the UI regions, toolbars, app bar, system bars, slide nav, and white frames.
Responsive UI	Determines the consistency, adaptability, and scalability of apps on different screen sizes. Includes the breakpoints, grid, surface behaviors, and patterns.
Split screen	Allows the user to view two tasks or activities on the screen simultaneously.

© Aptech Ltd.

Material Design/Session 16

13

Explain that the layout simplifies the process of creating scalable apps by using elements from print-based design. Read out the table given on the slides 12 and 13 for students.

Slides 14 and 15

Let us understand in brief about the components

Components 1-2	
Following table lists the specific components:	
Component	Description
Bottom navigation	Provides quick navigation between top-level views in an app
Bottom sheet	Reveals more screen content by sliding up from the bottom of the screen
Card	Displays information composed of different elements such as photo or text
Chip	Contains photos, text, or icons
Divider	Helps to organize the page into different tiles and thus separate the content

© Aptech Ltd. Material Design/Session 16 14

Components 2-2	
Component	Description
Picker	Helps to pick a single value from a pre-determined set of values
Snackbar	Provides a feedback on the performed operation at the bottom of the screen
Toast	Provides system messages at the bottom of the screen
Stepper	Displays the progress of an operation with the help of logical and numbered steps

© Aptech Ltd. Material Design/Session 16 15

Explain in brief about the components. Tell them that there are various user interface components in material design. Apart from the common components such as buttons, dialog boxes, grid lists, lists, menus, sliders, progress bars, tabs, text fields, toolbars, and tooltips, there are specific components in material design. Explain that the table lists the specific components. Read out the table given on the slides for the 14 and 15 students.

Slide 16

Let us understand about patterns.

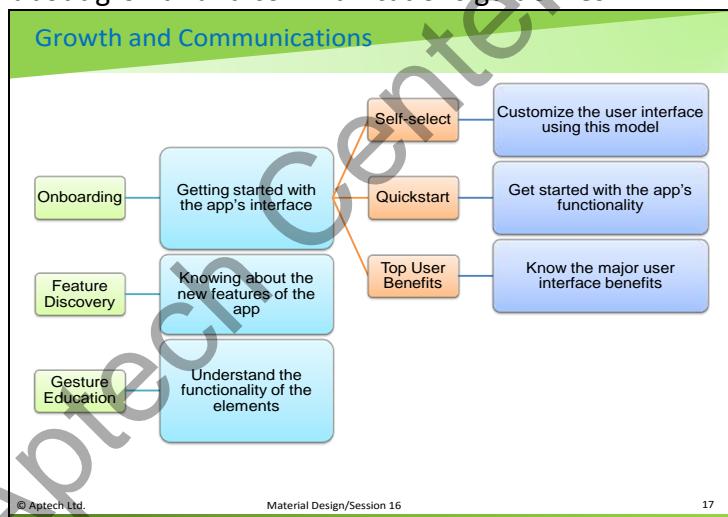
The slide has a green header bar with the word 'Patterns'. Below it is a purple bar containing the text 'Following table lists some of the patterns:'. A table follows, divided into two columns: 'Pattern' and 'Description'. The table contains six rows, each with a pattern name and its description. At the bottom of the slide are three small text elements: '© Aptech Ltd.', 'Material Design/Session 16', and '16'.

Pattern	Description
Empty states	Occurs when an action or operation does not display any content
Fingerprint	Unlocks the device or sign into apps
Gestures	Includes touch mechanics and touch activities
Launch screens	Indicates the initial screen that is launched in an app
Permissions	Explains the need for each permission request
Swipe to refresh	Refreshes the screen content based on user's action

Explain that there are different types of patterns in material design. Say that the table given on the slide lists some of the patterns. Read it out for students.

Slide 17

Let us understand about growth and communications guidelines.



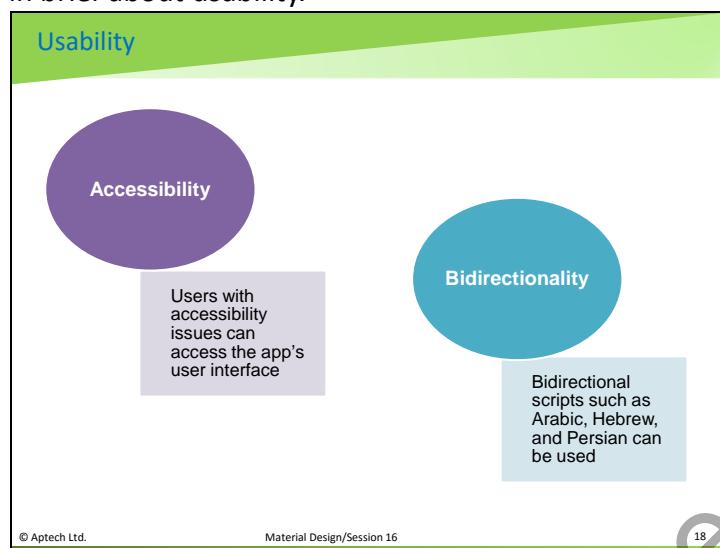
Explain that the growth and communications guidelines define the best practices that the user can adopt for the apps. The guidelines explain the features of onboarding, feature discovery, and gesture education. Let us learn in brief about the features.

Onboarding helps the users to get started with the app's user interface. There are three onboarding models in material design. These are self-select, quickstart, and top user benefits.

In the self-select model, the user can customize the user interface. In quickstart model, the user can quickly get started with the app's functionality. In top user benefits model, the user can know the major user interface by using the app. Feature discovery facilitates the user to gain awareness about new app features in the right context. Gesture education feature supports the users to understand the functionality of the elements, thereby enabling maximum interaction with the interface.

Slide 18

Let us understand in brief about usability.



Explain that there are two aspects of usability in material design. They are accessible and bidirectionality. The accessibility aspect allows all kinds of users, even those with accessibility issues, to access the app's user interface successfully. Some examples of such users, are those with visual and hearing challenges. There are different elements such as color and contrast, sound, motion, layout, and writing that provides effective access to the user interface for those users.

Material design is considered a feature of bidirectionality. Apps can include bidirectional scripts such as Arabic, Hebrew, and Persian. In bidirectional scripts, text is read and written from right-to-left unlike in the English language.

Slide 19

Let us understand about resources.

The table has a green header row with the title "Resources". The first column is labeled "Resource" and the second column is labeled "Description". The table lists eight resources: Color palette, Devices, Layout templates, Roboto, Noto, Sticker sheets, and Icons. Each resource has a corresponding description in the adjacent column. The table is framed by a light blue border. At the bottom left is the copyright notice "© Aptech Ltd.", and at the bottom center is "Material Design/Session 16". A small number "19" is in the bottom right corner.

Resources	
Following table lists the resources:	
Resource	Description
Color palette	Includes the Adobe Photoshop and Adobe Illustrator color swatches
Devices	Include the device metrics
Layout templates	Include the mobile layout, tablet layout, desktop layout, and whiteframes
Roboto	Is designed for mobile and Web usage
Noto	Is the standard typeface that covers the languages not dealt by Roboto
Sticker sheets	Are used by components and include elements that make up layouts
Icons	Indicate system and product icons

Explain that there are various downloadable resources available in material design. Say that the table given on the slide lists some of the resources. Read out the table for students.

Slide 20

Let us understand about themes and how to define the material theme.

Themes

What are Themes?

- The themes provide new styles for the apps.
- They offer uniform or logical styling of apps through surface shades, shadow depth, and ink opacity.

User can set:

- Various themes for the apps
- The color palette and touch animations for the system widgets using material themes
- Themes for the action bars and status bars

Material theme is defined as:

- @android:style/Theme.Material (dark version)
- @android:style/Theme.Material.Light (light version)
- @android:style/Theme.Material.Light.DarkActionBar

© Aptech Ltd. Material Design/Session 16 20

Explain about the material design elements. Tell them that, one of the primary design elements is themes. The themes provide new styles for the apps. They offer uniform or logical styling of apps through surface shades, shadow depth, and ink opacity. Users can set various themes for the apps. They can set the color palette and touch animations for the system widgets using material themes. They can also set themes for the action bars and status bars.

Slides 21 and 22

Let us understand about applying material themes.

Applying Material Themes 1-2

Following code snippet demonstrates how to customize the theme's base colors or define custom colors:

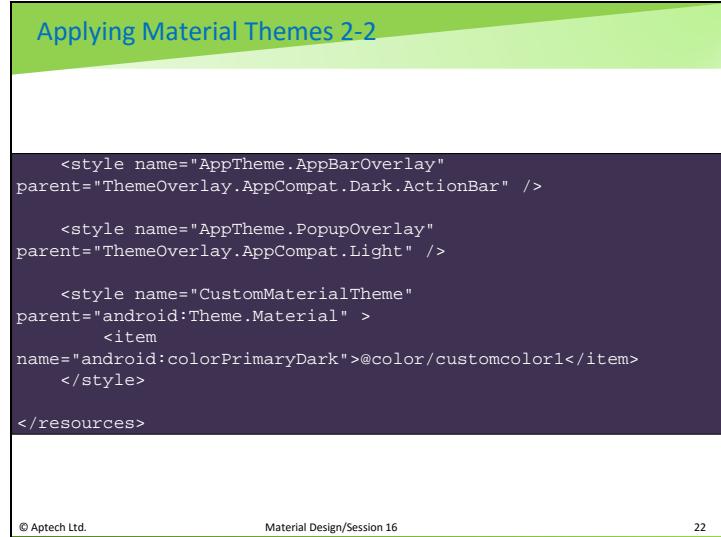
```

<resources>
<!-- Base application theme. -->
<style name="AppTheme"
parent="Theme.AppCompat.Light.DarkActionBar">
    <!-- Customize your theme here. -->
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent</item>
</style>

<style name="AppTheme.NoActionBar">
    <item name="windowActionBar">false</item>
    <item name="windowNoTitle">true</item>
</style>

```

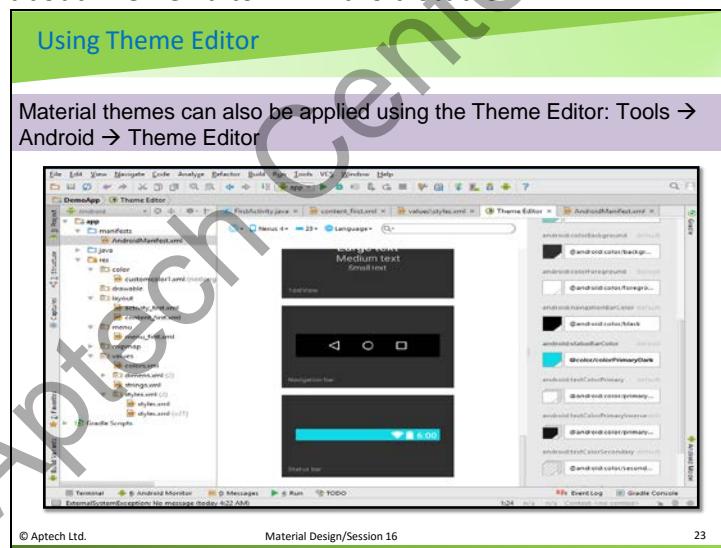
© Aptech Ltd. Material Design/Session 16 21



Explain that the material theme is only available in Android 5.0 and above versions. The user can use the material theme API to customize the color palette. Explain the code snippet to customize the theme's base colors or define custom colors. Using slide 21 and 22, explain the important lines.

Slide 23

Let us understand about Theme Editor in Android Studio.



Explain that to access the Theme Editor, click **Tools → Android → Theme Editor**. In the left pane, the theme options can be set and the effect of the settings selected can be viewed in the central pane or the work area.

Slide 24

Let us understand how to customize the status bar.

Customizing Status Bar

Following code snippet demonstrates how to customize the status bar:

```
<item name="android:statusBarColor">@color/colorPrimaryDark
</item>
```

© Aptech Ltd. Material Design/Session 16 24

Explain that the user can draw behind the status bar, set custom animations, and set custom colors. Explain in brief the code snippet to customize the status bar.

Slide 25

Let us understand about list and cards.

Lists and Cards

- Lists display multiple line items in a vertical manner as a single continuous element
- Cards are material sheets that function as entry points to display detailed information

RecyclerView widget

- Is a pluggable version of ListView
- Supports different types of layouts
- Improves and enhances the app's performance

CardView widget

- Allows the user to display important information inside cards

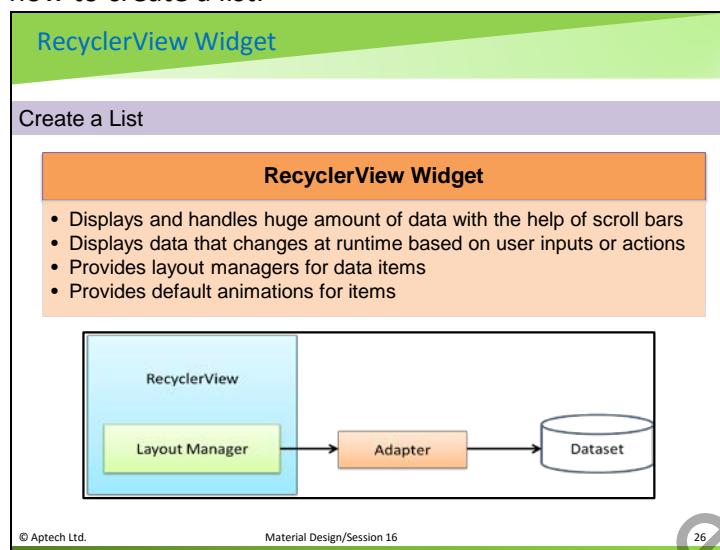
© Aptech Ltd. Material Design/Session 16 25

Explain that list and cards are the design elements available in material design. Lists display multiple line items in a vertical manner as a single continuous element. Cards are material sheets that function as entry points to display detailed information. There are two widgets in Android to display a list and cards with styles and animations. The app developer can create complex lists and cards using RecyclerView widget and CardView widget.

RecyclerView widget is a pluggable version of ListView. It supports different types of layouts. It improves and enhances the app's performance. CardView widget allows the user to display important information inside cards. There is a consistency maintained in the look and feel of the cards.

Slide 26

Let us understand how to create a list.

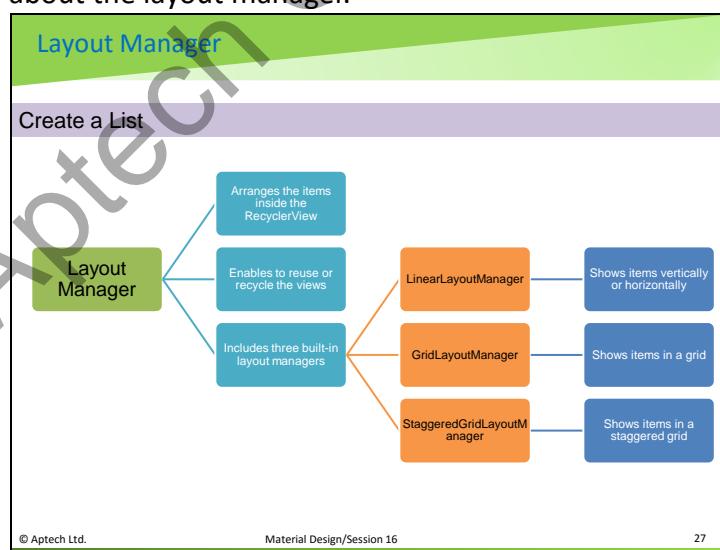


Explain that the developer can use the RecyclerView widget to display and handle huge amount of data with the help of scroll bars. The widget can be used to display data that changes at runtime, based on user inputs or actions. The widget provides layout managers for data items. It also provides default animations for items. Point to the figure given on the slide and explain.

This figure illustrates the RecyclerView widget. The RecyclerView widget has an adapter and a layout manager.

Slide 27

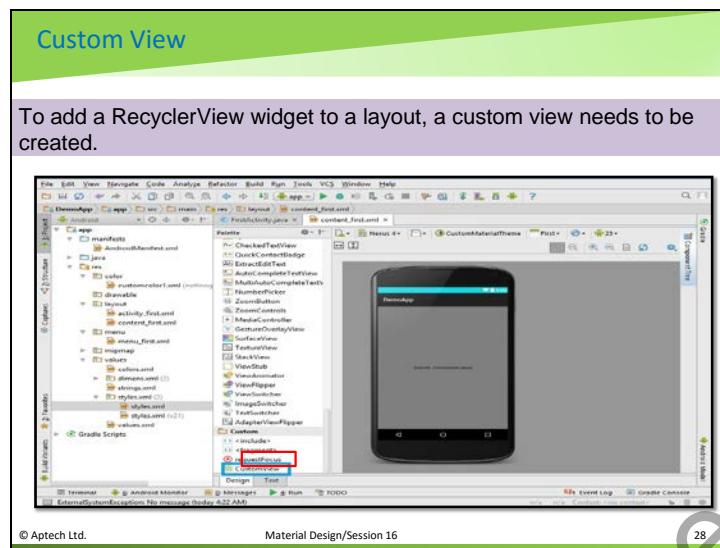
Let us understand about the layout manager.



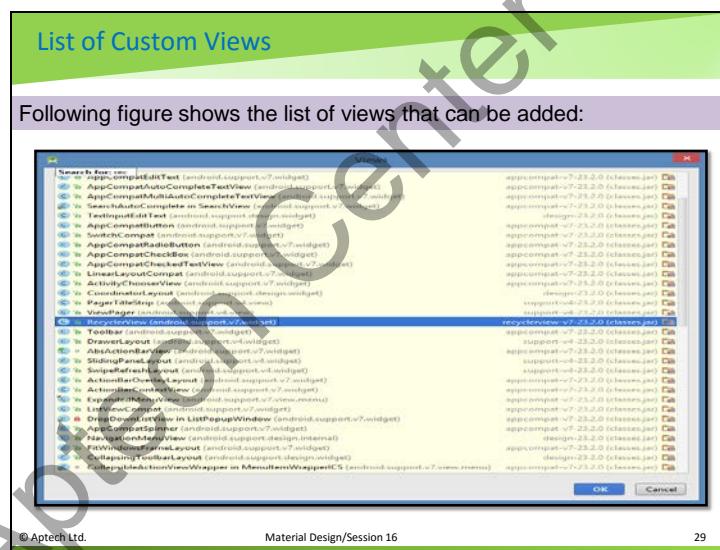
A layout manager arranges the items inside the RecyclerView. It enables to reuse or recycle the views with the help of the adapter. There are three built-in layout managers in RecyclerView widget. The first layout manager is LinearLayoutManager. This layout manager shows items in a vertical or horizontal manner with a scroll list. The second layout manager is GridLayoutManager. This layout manager shows items in a grid. The third layout manager is StaggeredGridLayoutManager. This layout manager shows items in a staggered grid.

Slides 28 and 29

Let us understand about the custom views.



Using slide 28, explain that in Android Studio, a custom view needs to be added to add a RecyclerView widget.



Using slide 29, explain that list of custom views that can be added is displayed. The RecyclerView widget can be selected from this list.

Slide 30

Let us understand how to add the RecyclerView widget to a layout.

Adding RecyclerView to Layout

Following code snippet demonstrates how to add the RecyclerView widget to a layout:

```
<!-- A RecyclerView -->
<android.support.v7.widget.RecyclerView
    android:id="@+id/view"
    android:scrollbars="vertical"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

© Aptech Ltd. Material Design/Session 16 30

Explain the code that shows how to add the RecyclerView widget to a layout by updating the content_first.xml file.

Slide 31

Let us understand about CardView widget.

CardView Widget

To add a CardView widget, drag the widget from the list of widgets to the work area in the Design mode.

The screenshot shows the Android Studio interface with the CardView widget selected in the palette on the left. The design view on the right displays a smartphone icon with a card on its screen, indicating where the widget has been placed.

© Aptech Ltd. Material Design/Session 16 31

Explain that you can use the CardView widget to create cards. You can use various properties to customize the appearance of the CardView widget. To add a CardView widget, you need to drag the widget from the list of widgets to the work area in the Design mode.

Slides 32 and 33

Let us understand how to add CardView to Layout.

Adding CardView to Layout 1-2

Following code snippet demonstrates how to add the CardView widget to a layout:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
    ...
    <!-- A CardView with a TextView -->
    <android.support.v7.widget.CardView
        xmlns:card_view="http://schemas.android.com/apk/res-auto"
        android:id="@+id/mycardview"
        android:layout_gravity="center"
        android:layout_width="200dp"
        android:layout_height="200dp"
        mycardview:cardCornerRadius="4dp">
```

© Aptech Ltd. Material Design/Session 16 32

Adding CardView to Layout 2-2

Following code snippet demonstrates how to add the CardView widget to a layout:

```
<TextView
    android:id="@+id/cardtext"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:text="This is a Card" />
</android.support.v7.widget.CardView>
</LinearLayout>
```

© Aptech Ltd. Material Design/Session 16 33

Explain code snippet that demonstrates how to add the CardView widget to a layout in CardViewLayout.xml file.

Slide 34

Let us understand the features and components of Z property.

Z Property

The Z Property:

- Is an addition to the existing X and Y properties
- Represents the view's elevation
- Determines the shadow's size, appearance, and the drawing order of the views
- Is measured in dp

The two components for the Z value of the view:

- Elevation:** It is the static component.
- Translation:** It is the dynamic component used for animations.

© Aptech Ltd. Material Design/Session 16 34

Explain that there is a Z property for views in Android. This is in addition to the existing X and Y properties. The Z property represents the view's elevation. It determines the shadow's size and appearance and the drawing order of the views. It is measured in dp.

There are two components for the Z value of the view. They are elevation and translation. Elevation is the static component. To set the elevation of the view in a layout definition, use the android:elevation attribute. To animate the elevations, use ViewPropertyAnimator.z() and ViewPropertyAnimator.translationZ() methods.

Translation is the dynamic component used for animations. To set the translation of the view, use View.setTranslationZ() method.

Slides 35 and 36

Let us understand how to customize view shadows.

Customizing View Shadows 1-2

Following code snippet demonstrates how to customize view shadows:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context=".com.syskan.demoapp.FirstActivity"
    tools:showIn="@layout/activity_first">
```

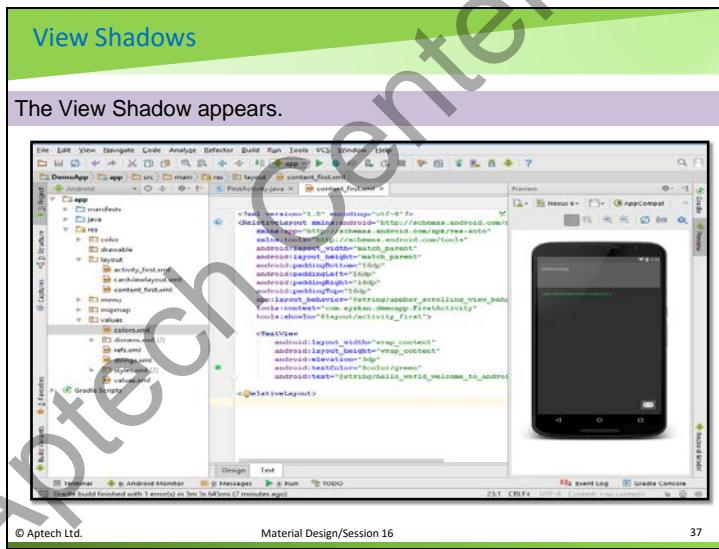
© Aptech Ltd. Material Design/Session 16 35



Explain the code snippet that demonstrates how to customize view shadows using slides 35 and 36.

Slide 37

Let us understand View Shadows.



Explain to the students how view shadow appear using the code given on the slide. Also explain the figure given on the slide that displays View Shadow.

Slide 38

Let us understand about clipping views.

Changing Shape of the View

Clipping Views

- ❑ Change the shape of the views
- ❑ Are applied to outline area
- ❑ Can be applied to rectangle, circle, and round rectangle

View.setClipToOutline() method

android:clipToOutline attribute

To clip view to the outline area

© Aptech Ltd. Material Design/Session 16 38

Explain that clip views are used to change the shape of the views. The developer can clip view to its outline area using the `View.setClipToOutline()` method or the `android:clipToOutline` attribute. All outlines do not support clipping except rectangle, circle, and round rectangle.

Slide 39

Let us understand that how to create touch-based custom animations for components in user interface.

Creating Touch-based Custom Animations for UI Components

Touch feedback	• Provides instant feedback to the users
Circular Reveal	• Provides visual continuity to users during showing or hiding of elements
Activity transitions	• Provide visual connections between different states of views
Curved motion	• Enables to customize the timing curves and curve motion patterns
View state change	• Enables to animate changes to view state in Android

© Aptech Ltd. Material Design/Session 16 39

Explain that the material themes provide default animations for buttons and activity transitions. Android 5.0 and above versions have various animation APIs that let the developers create touch-based custom animations for the user interface components. The custom animations can be added to the custom views. The developer can customize animations such as touch feedback, circular reveal, activity transitions, curved motion, and view state change.

The users can show or hide a group of user interface elements. The reveal animations provide visual continuity to the users during the process of showing or hiding of elements.

Material design provides activity transitions. The developers can specify custom animations for enter and exit transitions and for transitions of shared elements between activities. The enter transition determines the way views enter the scene; the exit transition determines the way views exit the scene; Android 5.0 supports explode, slide, and fade enter and exit transitions. The shared elements transition determines the sharing of views between activities transitions. For example, if two activities have the same image but of different sizes, the shared element transition ensures smooth transition between the activities with respect to the image. Android 5.0 supports shared elements transitions such as changeBounds, changeclipBounds, changeTransform, and changelImageTransform. In Android 5.0, the developers can customize the timing curves and curve motion patterns. The developer can animate changes to view state in Android. The StateListAnimator class defines animators that run when the state of a view changes.

Slides 40 and 41

Let us understand how to use the hide and reveal animation effect on a view.

Use Hide And Reveal Animation Effect On A View 1-2

Following code snippet demonstrates how to use hide and reveal animation effect on a view:

```
// previously invisible view
View myView = findViewById(R.id.mycardview);
// get the center for the clipping circle
int width = myView.getWidth() / 2;
int height = myView.getHeight() / 2;
float finalRadius = (float) Math.hypot(width, height);
Animator anim =
    ViewAnimationUtils.createCircularReveal(myView, width, height, 0,
    finalRadius);
myView.setVisibility(View.VISIBLE);
anim.start();
// previously visible view
final View myView = findViewById(R.id.mycardview);
```

© Aptech Ltd. Material Design/Session 16 40

Use Hide And Reveal Animation Effect On A View 2-2

Following code snippet demonstrates how to use hide and reveal animation effect on a view:

```
int width = myView.getWidth() / 2;
int height = myView.getHeight() / 2;
float initialRadius = (float) Math.hypot(width, height);
// create the animation (the final radius is zero)
Animator anim =
ViewAnimationUtils.createCircularReveal(myView, width, height, initialRadius, 0);
// make the view invisible when the animation is done
anim.addListener(new AnimatorListenerAdapter() {
    @Override
    public void onAnimationEnd(Animator animation) {
        super.onAnimationEnd(animation);
        myView.setVisibility(View.INVISIBLE);
    }
});
// start the animation
anim.start();
```

© Aptech Ltd. Material Design/Session 16 41

Explain that the reveal animations provide visual continuity to the users during the process of showing or hiding of elements.

Slide 42

Let us understand about custom transitions.

Custom Transitions

Following code snippet demonstrates custom transitions:

```
<style name="BaseAppTheme" parent="android:Theme.Material">
    <!-- enable window content transitions -->
    <item name="android:windowActivityTransitions">true</item>
    <item
        name="android:windowEnterTransition">@transition/burst</item>
    <item
        name="android:windowExitTransition">@transition/burst</item>
    <!-- specify shared element transitions -->
    <item name="android:windowSharedElementEnterTransition">
        @transition/transform_image</item>
    <item name="android:windowSharedElementExitTransition">
        @transition/transform_image</item>
</style>
```

© Aptech Ltd. Material Design/Session 16 42

Explain the code snippet given on the slide.

Slide 43

Let us understand the code snippet that demonstrates the usage of curved motion.

Customizing Timing Curves and Curve Motion Patterns

Following code snippet demonstrates the usage of curved motion.

```
<pathInterpolator
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:controlX1="0.4"
    android:controlY1="0"
    android:controlX2="1"
    android:controlY2="1" />
ObjectAnimator myAnimator;
myAnimator = ObjectAnimator.ofFloat(view, View.X, View.Y, path);
...
myAnimator.start();
```

© Aptech Ltd. Material Design/Session 16 43

Slides 44 and 45

Let us understand the code snippet that demonstrates the usage of the StateListAnimator class.

Animating Changes to View State 1-2

Following code snippet demonstrates the usage of the StateListAnimator class:

```
<!-- animate the translationZ property of a view when pressed -->
<selector
    xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_pressed="true">
        <set>
            <objectAnimator android:propertyName="translationZ"
                android:duration="@android:integer/config_shortAnimTime"
                android:valueTo="3dp"
                android:valueType="floatType" />
        </set>
    </item>
```

© Aptech Ltd. Material Design/Session 16 44

Animating Changes to View State 2-2

Following code snippet demonstrates the usage of the StateListAnimator class:

```
<item android:state_enabled="true"
      android:state_pressed="false"
      android:state_focused="true">
    <set>
        <objectAnimator android:propertyName="translationZ"
            android:duration="150"
            android:valueTo="0"
            android:valueType="floatType" />
    </set>
</item>
</selector>
```

© Aptech Ltd. Material Design/Session 16 45

Explain that the StateListAnimator class defines animators that run when the state of a view changes using slides 44 and 45.

Slide 46

Let us Summarize the session.

Summary

- ❑ Material design combines good design principles with advanced tools to create a single underlying system across various platforms and mobile devices.
- ❑ The difference in the elevations between the overlapping material objects creates shadows in the environment.
- ❑ Material motion describes hierarchical and spatial relationships between objects.
- ❑ The layout simplifies the process of creating scalable apps by using elements from print-based design.
- ❑ The growth and communications guidelines define the best practices that can be adopted for developing the apps.
- ❑ RecyclerView widget is a pluggable version of ListView and supports different types of layouts.
- ❑ Activity transitions provide visual connections between different states of views.
- ❑ The shared elements transition determines the sharing of views between activities transitions.

© Aptech Ltd. Material Design/Session 16 46

Using this slide, summarize the key points of this session. Explain the following points in brief.

- Material design combines good design principles with advanced tools to create a single underlying system across various platforms and mobile devices.
- The difference in the elevations between the overlapping material objects creates shadows in the environment.
- Material motion describes hierarchical and spatial relationships between objects.
- The layout simplifies the process of creating scalable apps by using elements from print-based design.
- The growth and communications guidelines define the best practices that can be adopted for developing the apps.
- RecyclerView widget is a pluggable version of ListView and supports different types of layouts.
- Activity transitions provide visual connections between different states of views.
- The shared elements transition determines the sharing of views between activities transitions.

Next, let us check your understanding of the topics covered so far in this session.

16.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session.

Session 17: Android Interface Definition Language

17.1. Pre-Class Activities

Before you commence the session, you should familiarize yourself with the topics of this session in-depth. Prepare a question or two that will be a key point to relate the current session objectives.

17.1.1 Objectives

By the end of this session, learners will be able to:

- Explain the AIDL Interface
- Explain how to create an AIDL file
- Explain how to implement the interface
- Explain the process of communication through IPC
- Explain how the IPC method is invoked

17.1.2 Teaching Skills

To teach this session, you should be well versed with the concepts of Android Application Development is Android Interface Definition Language or AIDL. Say that this session explains the AIDL in detail. Say that it also explains how objects of both the client and the service provider can be shared with the help of AIDL.

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of students by asking questions in between the class.

In-Class Activities

Follow the order given here during In-Class activities.

Overview of the Session

Give the students an overview of the current session in the form of session objectives. Show the students slide 2 of the presentation.

Slide 2

Objectives

- ❑ Explain the AIDL Interface
- ❑ Explain how to create an AIDL file
- ❑ Explain how to implement the interface
- ❑ Explain the process of communication through IPC
- ❑ Explain how the IPC method is invoked

© Aptech Ltd. Android Interface Definition Language/Session 17 2

17.2 In-Class Explanations

Slide 3

Using this slide, give an introduction to AIDL.

Introduction to AIDL

- ❑ App1 processes need to connect to the processes in App2.
- ❑ App1 is the service provider and App2 is the client.
- ❑ AIDL helps in communication between the processes with the help of Interprocess Communication (IPC).

© Aptech Ltd. Android Interface Definition Language/Session 17 3

Ask: What does IPC stand for?

Expected response: Interprocess Communication.

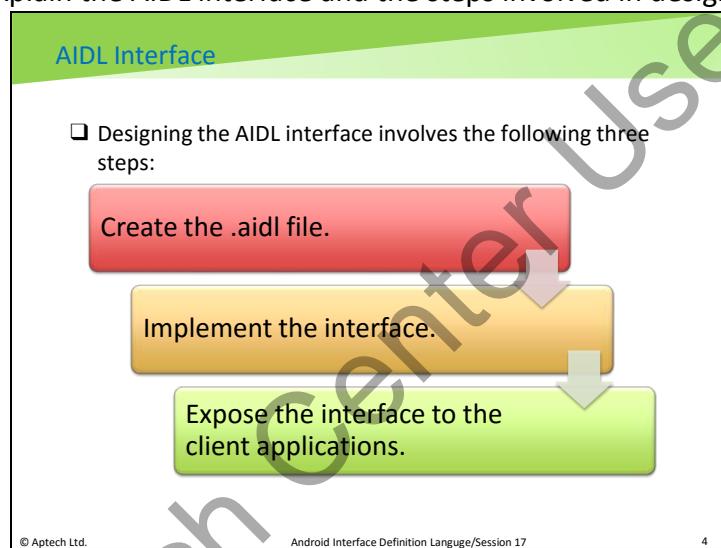
Appreciate the students' responses.

Tell the students that suppose App1 is a school application that provides data related to the schools in an area in a certain city. The data includes information about students studying in different schools—age, personal information, class, stream, and so on—and data related to the teachers—qualification, years of experience, other personal information, and so on. This application can be viewed as the large data bank that provides essential information about the schools, students, and teachers of a particular city.

Now, there is another school-based application App2. This application processes the student information provided by App1 to suggest a suitable course or career choice for the student. In this case, the AIDL interface can be used for communication between the two processes residing in App1 and App2. IPC can be used to implement AIDL.

Slide 4

Using this slide, explain the AIDL interface and the steps involved in designing the interface.



Ask:

What is the first step to design an AIDL interface?

Expected response: Create an .aidl file.

Appreciate the students' responses.

Explain that to enable communication between the client and service provider interfaces, the user has to define the AIDL interface. This interface determines the methods and fields that should be available for client access. The user can define the AIDL interface in the .aidl file using the Java programming language syntax. The user needs to save the file in the source code of the service hosting application and the application that requires the information from the service.

Ask:

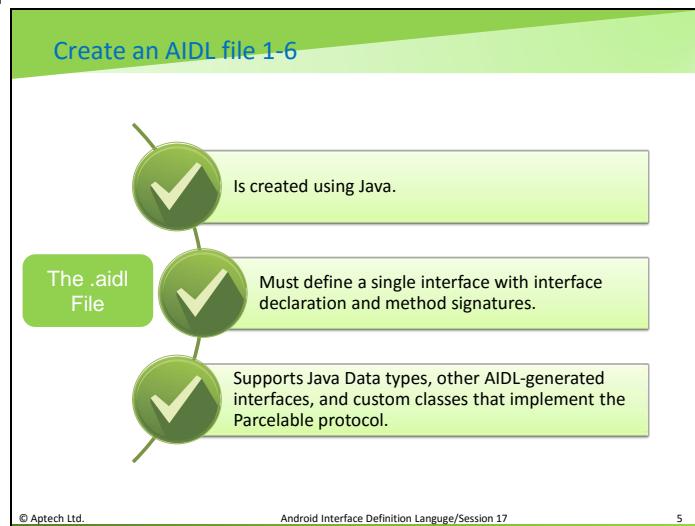
Can someone summarize the steps to design an AIDL interface?

Expected response:

- Create the .aidl file
- Implement the interface
- Expose the interface to the client applications

Slide 5

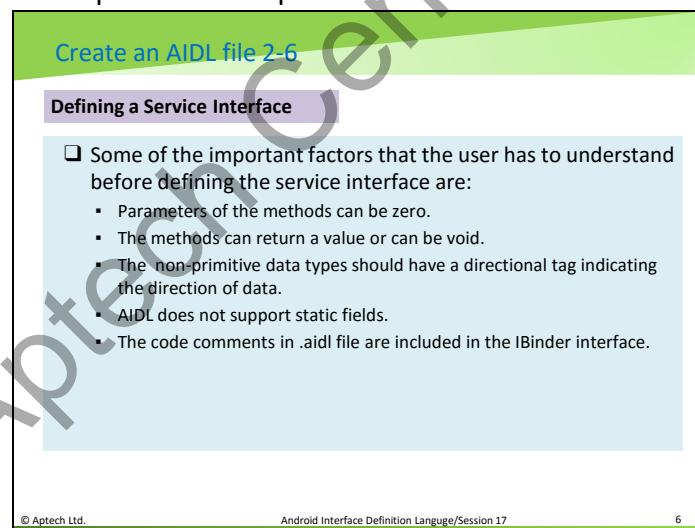
Using this slide, explain how to create an .aidl file.



Explain that the AIDL file includes an interface with one or more methods. These methods can take parameters and return values of any type, including another aidl interface. The data types supported by AIDL by default include all primitive data types offered by the Java programming language, other AIDL-generated interfaces, and custom classes that implement the Parcelable protocol.

Slide 6

Using this slide, further explain some important factors to create an .aidl file.



Explain that some of the important factors to consider when defining the service interface are:

Parameters of the methods can be zero.

The methods can return a value or can be void.

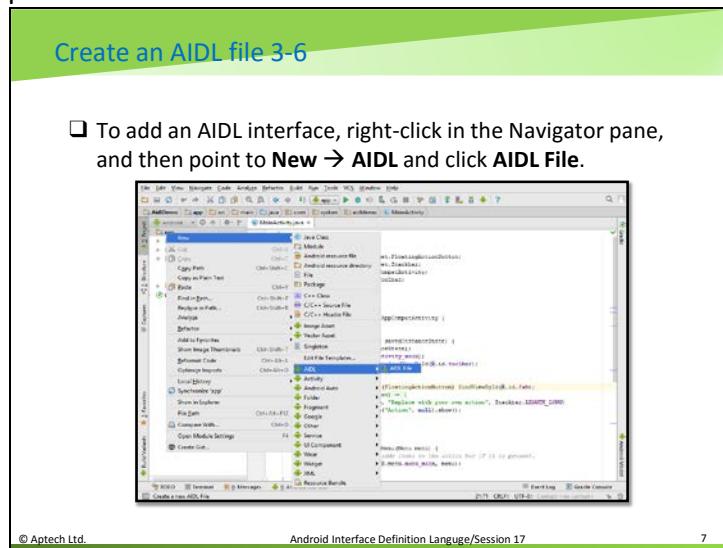
The non-primitive data types should have a directional tag to indicate the direction of flow of the data.

AIDL does not support static fields.

The code comments in the .aidl file is included in the IBinder interface.

Slide 7

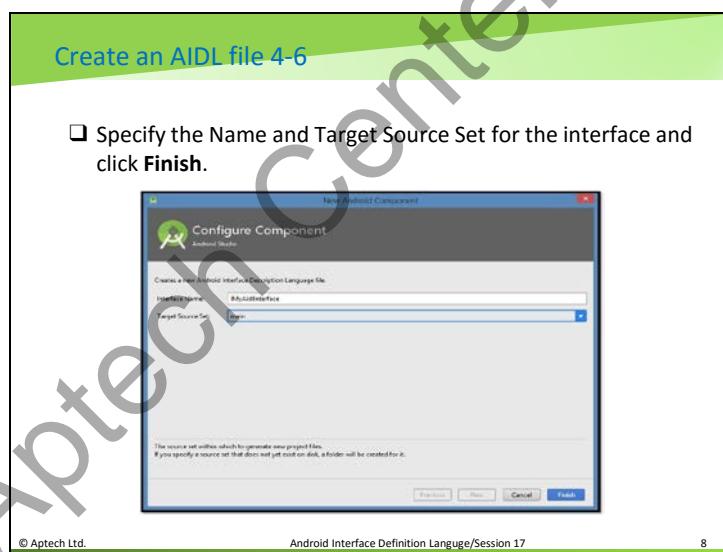
Using this slide, explain how to create .aidl file in Android Studio.



Explain that to create an .aidl file, right-click in the Navigator pane, and then point to **New** → **AIDL** and click **AIDL File**.

Slide 8

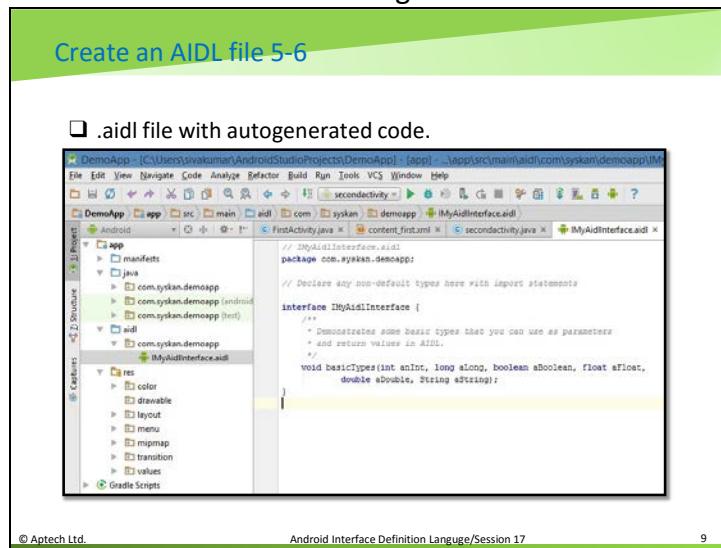
Explain the further steps.



Explain the figure given on the slide where New Android Component dialog box appears. In this dialog box, the name of the interface and the target source set of the interface must be specified. After providing the required details, the Finish button must be clicked.

Slide 9

Understand that an .aidl file is created with autogenerated.



Explain that an .aidl file is created with autogenerated code as shown in figure on the slide.

Slide 10

Let us understand that .aidl file is placed in the src directory of the server and the client applications.



Explain that the .aidl file is placed in the src directory of the server and the client applications. When the application builds, the SDK tools generate the IBinder interface. The IBinder interface is placed in the ‘gen’ or ‘generated’ directory and has the same name as the .aidl file, however with a .java extension.

Slide 11

Using this slide, explain how to implement the AIDL interface.

Implement AIDL Interface 1-2

- ❑ The interface generated by the .aidl file with .java extension includes a subclass Stub that declares all the methods from the .aidl file.
- ❑ The Stub consists of helper methods such as asInterface() which implements the IBinder interface and returns the instance of the interface, which is used to call the RPC methods.

© Aptech Ltd. Android Interface Definition Languge/Session 17 11

Explain that the interface generated by the .aidl file has a subclass named Stub. This subclass declares all the methods in the .aidl file. It includes the helper methods such as asInterface() which implements the IBinder interface and returns the instance of the interface, which is used to call the RPC methods.

Slide 12

Using this slide, explain the code snippet.

Implement AIDL Interface 2-2

- ❑ Following code snippet shows how to extend the Binder interface and implement the interface using an anonymous interface:

```
private final IMyAidlInterface.Stub myBinder = new
IMyAidlInterface.Stub()
{
    public void basicTypes(int IntegerData, long LongData, boolean
BooleanData,
                           float FloatData, double DoubleData, String StringData)
    {
        // Does nothing
    }
},
```

© Aptech Ltd. Android Interface Definition Languge/Session 17 12

Explain that the code shows how to extend the Binder interface and implement the interface using an anonymous instance.

Explain that the code defines an instance of the stub class, which defines the Remote Procedure Call (RPC) for the service. However, the user should consider of some of the important factors before the implementation of the interface.

The service should be thread safe, that is, it should support multithreading.

The RPC calls are synchronous. RPC calls should use separate thread, if the process is time consuming. This avoids forcing shutting down of the application.

The Exceptions thrown are not diverted back to the caller.

Slide 13

Let us understand how to expose the interface to the client.

Expose AIDL Interface to Clients 1-3

- Expose the interface or publish the service to the client, the user needs to inherit Service and implement the Service.onBind () method.
- This method returns an instance of the class that implements the generated Stub.

© Aptech Ltd. Android Interface Definition Languge/Session 17 13

Explain that to expose the interface or to publish the service to the client, inherit the Service and implement the Service.onBind () method. This method returns an instance of the class that implements the generated Stub.

Slide 14

Let us understand the code snippet.

Expose AIDL Interface to Clients 2-3

```
import android.app.Service;
. . . // Other import statements
import com.syskan.aidldemo.IMyAidlInterface;
public class MyFirstRemoteService extends Service {
@Override
public void onCreate() {
super.onCreate();
}
@Override
public IBinder onBind(Intent intent) {
// Return the interface
return myBinder;
}
private final IMyAidlInterface.Stub myBinder = new
IMyAidlInterface.Stub() {
public void basicTypes(int IntegerData, long LongData, boolean
BooleanData, float FloatData, double DoubleData, String StringData) {
// Does nothing
} };
}
```

© Aptech Ltd. Android Interface Definition Languge/Session 17 14

Explain that the code snippet shows how to publish the service to the client by exposing the IRemoteService Interface. Ensure that any other import statements that may be required are added. IMyAidlInterface should be used only for Stub instantiation, not for this class.

Ask the class to consider the scenario discussed earlier about the School application. App2 process calls the bindService () method to connect to the service provided by App1 and the App2 onServiceConnected () callback receives the mBinder instance. This instance is returned by the onBind () method.

The .aidl file should also be copied to the src directory of the client (App2 in this case), if the client and service (App1) are in separate applications.

Slide 15

Let us understand the code snippet.

Expose AIDL Interface to Clients 3-3

```
IMyAidlInterface iMyAidlInterface;
protected ServiceConnection mConnection = new ServiceConnection() {
    // Called when the connection with the service is established
    public void onServiceConnected(ComponentName className, IBinder
        service) {
        // Following the example given earlier for an AIDL interface,
        // this gets an instance of the IRemoteInterface, which
        // we can use to call on the service
        iMyAidlInterface = IMyAidlInterface.Stub.asInterface
            (service);
    }
    String TAG;
    // Called when the connection with the service disconnects
    // unexpectedly
    public void onServiceDisconnected(ComponentName className) {
        Log.e(TAG, "Service has unexpectedly disconnected");
        iMyAidlInterface = null;
    }
};
```

© Aptech Ltd. Android Interface Definition Language/Session 17 15

Explain that code snippet shows how the client calls the **IMyAidlInterface**.Stub.asInterface (service) when it receives the IBinder in the onServiceConnected () method. This code must be added to the activity.java file.

Slide 16

Let us understand that how IPC facilitates communication between processes.

Communication Through IPC 1-4

To facilitate communication through IPC and to create a class that supports parcelable protocol:

- 1 • Implement the Parcelable interface in the class.
- 2 • Write the current state of the object to Parcel, implement writetoParcel.
- 3 • Implement the Parcelable.Creator interface, use a static field CREATOR in the class.
- 4 • Ensure that the .aidl file that is created, declares the Parcelable class.

© Aptech Ltd. Android Interface Definition Language/Session 17 16

Explain that for any class to be shared with another process, the class must support the Parcelable interface. IPC interface facilitates communication between processes, including sending class from one process to another. However, IPC can be implemented only if the class code exists on both sides of the IPC channel. In addition, the class must also support the Parcelable interface. Parcelable interfaces allows the Android system to decompose the objects of a class into primitives. This primitive form of the objects can be easily interfaced across various other processes.

To facilitate communication through IPC and to create a class that supports the Parcelable protocol, the Parcelable interface must be implemented in the class.

To write the current state of the object to Parcel, writetoParcel must be implemented.

To implement the Parcelable.Creator interface, a static field called CREATOR must be used in the class. The .aidl file that is created must declare the Parcelable class.

Slide 17

Let us understand with the help of code snippet how to create a class that supports Parcelable interface.

Communication Through IPC 2-4

- Following code snippet shows how to create a parcelable class in the existing or a new .aidl file:

```
// Declare Triangle so AIDL can find it & knows that it implements
// the parcelable protocol.
parcelable Triangle;
```

© Aptech Ltd.

Android Interface Definition Language/Session 17

17

Slides 18 and 19

Let us understand how to declare a class parcelable in the .aidl file.

Communication Through IPC 3-4

```
import android.os.Parcel;
import android.os.Parcelable;
public final class Triangle implements Parcelable {
    public int leftpos;
    public int toppos;
    public int rightpos;
    public static final Parcelable.Creator<Triangle> CREATOR = new
        Parcelable.Creator<Triangle>() {
            public Triangle createFromParcel(Parcel in) {
                return new Triangle(in);
            }
            public Triangle[] newArray(int size) { return new
                Triangle[size];
            }
        };
}
```

© Aptech Ltd.

Android Interface Definition Language/Session 17

18

Communication Through IPC 4-4

```
public Triangle() { }
private Triangle(Parcel in) {
    readFromParcel(in);
}
public void writeToParcel(Parcel out) {
    out.writeInt(leftpos);
    out.writeInt(toppos);
    out.writeInt(rightpos);
}
public void readFromParcel(Parcel in) {
    leftpos = in.readInt();
    toppos = in.readInt();
    rightpos = in.readInt();
}
```

© Aptech Ltd.

Android Interface Definition Language/Session 17

19

Using slide 18 and 19, explain with the help of code snippet how to declare a class parcelable in the .aidl file. Continue with the code snippet explanation on this slide. Say this code snippet shows the implementation of the Parcelable protocol. This class assists when sending a parcelable object in an IPC call. Declaring the parcelable class name in .aidl file, enables sending object in an IPC. In addition, the parcelable interface allows android system to decompose the object into primitives and assist the marshalling process.

Slide 20

Let us understand the steps to invoke an IPC method.

IPC Method Invocation 1-5

❑ The IPC method is invoked when the calling class performs the following steps:

1. It copies the .aidl file to the src directory of the project.
2. It creates an instance of the IBinder interface.
3. It implements ServiceConnection.
4. It calls context.bindservice () in the ServiceConnection implementation.
5. It calls IMyAidlInterface.Stub.asInterface (IBinder) service to cast the returned parameter to IMyAidlInterface type.
6. It calls the methods defined in the interface.
7. It calls the Context.unbindService () method with the instance of the service to disconnect.

© Aptech Ltd. Android Interface Definition Language/Session 17 20

Using the slide 20, explain the steps to invoke an IPC method.

Say that the IPC method is invoked when the calling class performs certain steps to call the remote AIDL interface. These steps include:

- It copies the .aidl file to the src directory of the project.
- It creates an instance of the IBinder interface.
- It implements ServiceConnection.
- It calls context.bindservice () in the ServiceConnection implementation.
- It calls IMyAidlInterface.Stub.asInterface (IBinder) service to cast the returned parameter to IMyAidlInterface type.
- It calls the methods defined in the interface.
- It calls the Context.unbindService () method with the instance of the service to disconnect.

Slides 21 to 25

Let us understand how to call an AIDL interface.

IPC Method Invocation 2-5

```
import android.content.ComponentName;
import android.content.Intent;
import android.content.ServiceConnection;
import android.os.Bundle;
import android.os.IBinder;
import android.os.RemoteException;
import android.support.v7.app.AppCompatActivity;
import android.widget.TextView;
public class MyFirstBinding extends AppCompatActivity
private IMyAidlInterface mService;
private TextView mLog;
@Override
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    mLog = (TextView) findViewById(R.id.log);
```

© Aptech Ltd.

Android Interface Definition Languge/Session 17

21

IPC Method Invocation 3-5

```
Intent serviceIntent = new Intent()
.setComponent(new ComponentName(
"com.syskan.aidlexamplereceiver",
"com.syskan.aidlexamplereceiver.MainService"));
mLog.setText("Starting service..\n");
startService(serviceIntent);
mLog.append("Binding service..\n");
bindService(serviceIntent, mConnection, BIND_AUTO_CREATE);
}
private ServiceConnection mConnection = new ServiceConnection()
{
@Override
public void onServiceConnected(ComponentName className, IBinder
service)
{
    mLog.append("Service binded!\n");
    mService = IMyAidlInterface.Stub.asInterface(service);
    performListing();
}
```

© Aptech Ltd.

Android Interface Definition Languge/Session 17

22

IPC Method Invocation 3-5

```
@Override
public void onServiceDisconnected(ComponentName className)
{
    mService = null;
    // This method is only invoked when the service quits from the // other end or gets killed
    // Invoking exit() from the AIDL interface makes the Service
    // kill itself, thus invoking this.
    mLog.append("Service disconnected.\n");
}
private void performListing()
{
    mLog.append("Requesting file listing..\n");
    long start = System.currentTimeMillis();
    long end = 0;
```

© Aptech Ltd.

Android Interface Definition Languge/Session 17

23

IPC Method Invocation 4-5

```
try
{
    MainObject[] results = mService.listFiles("/sdcard/testing");
    end = System.currentTimeMillis();
    int index = 0;
    mLog.append("Received " + results.length + " results:\n");
    for (MainObject o : results)
    {
        if (index > 20)
        {
            mLog.append("\t -> Response truncated!\n");
            break;
        }
        mLog.append("\t -> " + o.getPath() + "\n");
        index++;
    }
}
```

© Aptech Ltd.

Android Interface Definition Language/Session 17

24

IPC Method Invocation 5-5

```
catch (RemoteException e)
{
    e.printStackTrace();
}
mLog.append("File listing took " + (((double) end - (double) start) / 1000d) + " seconds, or " + (end - start) + " milliseconds.\n");
try
{
    mService.exit();
}
catch (RemoteException e)
{
    e.printStackTrace();
}
}
```

© Aptech Ltd.

Android Interface Definition Language/Session 17

25

Using the slide 23, explain with the help of code snippet how to kill the service. Using the slide 22, explain with the help of code snippet how the ServiceConnection interface is implemented.

Using the slide 24, explain with the help of code snippet how to use the try block to capture errors. Using the slide 25, explain with the help of code snippet how to use the catch block to handle errors.

Slide 26

Let us summarize the session.

Summary

- Android Interface Definition Language allows communication between the client and service provider through IPC.
- AIDL interface is defined in the .aidl file using Java programming language.
- .aidl file defines the programming interface that includes method signatures.
- The programming interface contains an inner abstract class Stub, that implements methods from the AIDL interface.
- To publish the service to the clients, expose the interface to the clients.
- IPC interface allows a parcelable class to be sent from one process to another.
- The calling class must follow certain steps to call the remote AIDL interface.

© Aptech Ltd. Android Interface Definition Language/Session 17 26

Using the slide 26, summarize the key points of this session. Explain the following points in brief:

- Android Interface Definition Language allows communication between the client and service provider through IPC.
- AIDL interface is defined in the .aidl file using Java programming language.
- .aidl file defines the programming interface that includes method signatures.
- The programming interface contains an inner abstract class Stub, that implements methods from the AIDL interface.
- To publish the service to the clients, expose the interface to the clients.
- IPC interface allows a parcelable class to be sent from one process to another.
- The calling class must follow certain steps to call the remote AIDL interface.

Next, let us check your understanding of the topics covered so far in this session.

17.3 Post Class Activities for Faculty

You should familiarize yourself with the topics of the next session.

Session 18: Android Native Development Kit (NDK)

18.1 Pre-Class Activities

Before you commence the session, you should familiarize yourself with the topics of this session in-depth. Prepare a question or two that will be a key point to relate the current session objectives.

18.1.1 Objectives

By the end of this session, learners will be able to:

- Describe app components
- Explain the process to install the NDK
- Explain the steps to create and test a sample native app

18.1.2 Teaching Skills

To teach this session, you should be well versed with the concepts of Android application development is on Android Native Development Kit (NDK). Say that this session explains the basic concepts of Android Native Development Kit (NDK). It describes the main components that are used in the native app development. It explains the process to install the NDK and create a sample app..

You should teach the concepts in the theory class using the images provided. For teaching in the class, you are expected to use slides and LCD projectors.

Tips:

It is recommended that you test the understanding of students by asking questions in between the class.

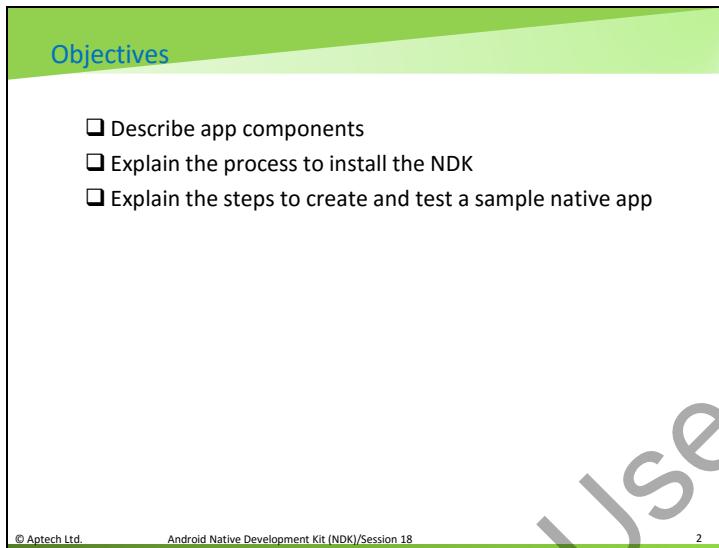
In-Class Activities

Follow the order given here during In-Class activities.

Overview of the Session

Give the students an overview of the current session in the form of session objectives. Show the students slide 2 of the presentation.

Slide 2



The slide has a green header bar with the word "Objectives" in blue. The main content area contains three bullet points:

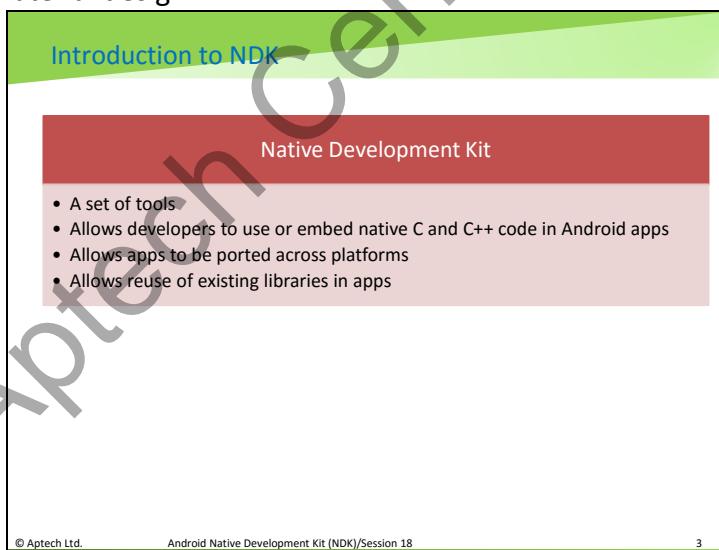
- Describe app components
- Explain the process to install the NDK
- Explain the steps to create and test a sample native app

At the bottom, there is a thin green footer bar with the text "© Aptech Ltd." and "Android Native Development Kit (NDK)/Session 18" on the left, and the number "2" on the right.

18.2 In-Class Explanations

Slide 3

Let us talk about material design.



The slide has a green header bar with the text "Introduction to NDK" in blue. Below it is a red bar with the text "Native Development Kit". The main content area contains a bulleted list:

- A set of tools
- Allows developers to use or embed native C and C++ code in Android apps
- Allows apps to be ported across platforms
- Allows reuse of existing libraries in apps

At the bottom, there is a thin green footer bar with the text "© Aptech Ltd." and "Android Native Development Kit (NDK)/Session 18" on the left, and the number "3" on the right.

Explain that material design helps to create a single underlying system across various platforms and mobile devices by combining the best and classic design principles with some advanced tools. It is also considered as a visual or design language developed by Google. It explains the relation between visual, motion, and interaction design using various grid-based layouts, animations, lighting effects, and shadows.

Slides 4 and 5

Let us understand the components used for building native applications for Android devices.

App Components 1-4

❑ Components that can be used when building native applications for Android devices.

Application Binary Interface (ABI)

- CPU-specific interface between system and app code.
- Facilitates interaction of app code with the system at runtime.
- App to work on systems with different CPUs and instruction sets.

Java

- During build, all Java files in the android app are converted into .dex files.
- If application does not include any Java code, a .dex file is generated for the native component.

Java Native Interface (JNI)

Programming framework that enables Java code and C or C++ code contained in an application to interact with each other.

© Aptech Ltd. Android Native Development Kit (NDK)/Session 18 4

App Components 2-4

Manifest

- NativeActivity class must be declared in the manifest if there is no Java code in the app.

ndk-build

- Shell script which runs the required NDK build scripts
- Generates binary files to be copied to the app's project path
- Requires Application.mk to build apps using the ndk-build file
- Android.mk configuration file added in the jni folder

Native Shared Libraries

- Built from the app's native source code and have an extension of .so

Native Static Libraries

- Can be linked with other existing libraries.
- Have an extension of .a

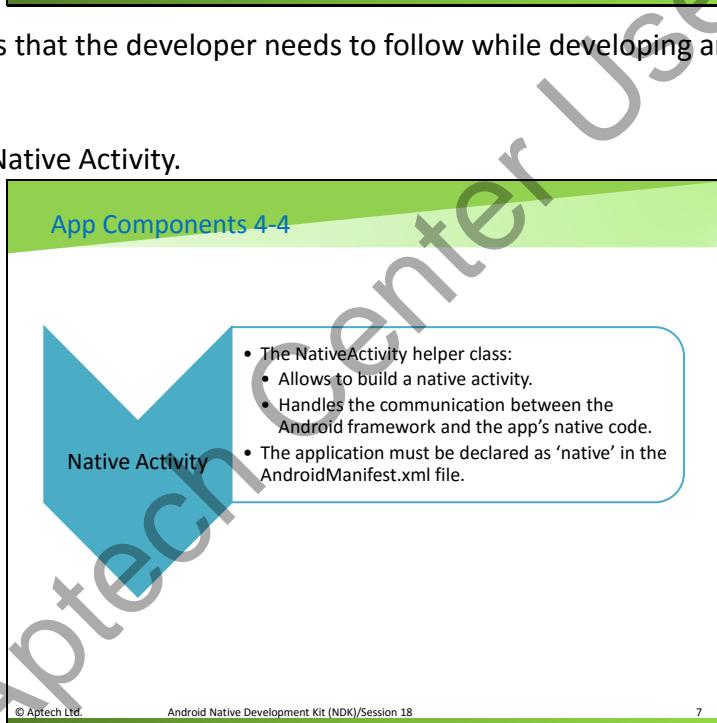
© Aptech Ltd. Android Native Development Kit (NDK)/Session 18 5

Using slides 4 and 5, explain the different components that the developers can use when building native applications for Android devices.

Explain that developers can use various components when building native applications for Android devices. Explain that we will discuss the main components. Read out the points for each of the components and explain.

Slide 6

Let us understand the steps for developing Android app.



App Components 3-4

The steps for developing an Android app:

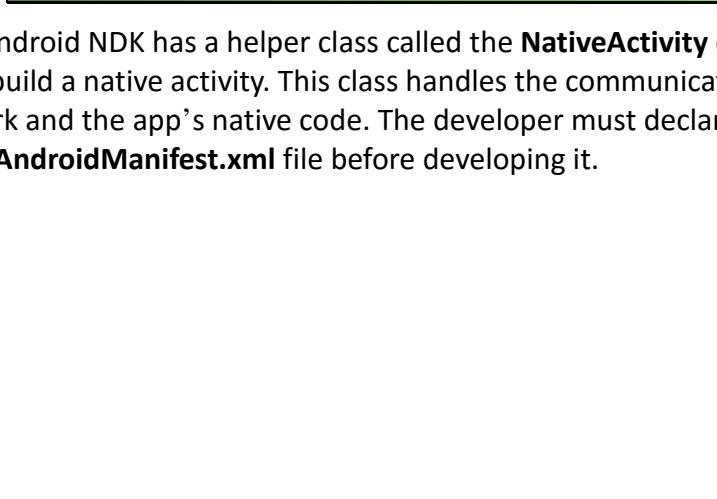
- 1 • Design the app.
- 2 • Decide the Java and native code components that need to be used in the app.
- 3 • Create an Android app project.
- 4 • For a native-only app, declare the **NativeActivity** class in **AndroidManifest.xml**.
- 5 • Create an **Android.mk** file.
- 6 • Add the native source code under the project's **jni** directory.
- 7 • Use **ndk-build** to compile the native libraries.
- 8 • Build the Java component.
- 9 • Package all the components into an **Android Application Package (APK)** file.

© Aptech Ltd. Android Native Development Kit (NDK)/Session 18 6

Read out the steps that the developer needs to follow while developing an Android app.

Slide 7

Let us talk about Native Activity.



App Components 4-4

Native Activity

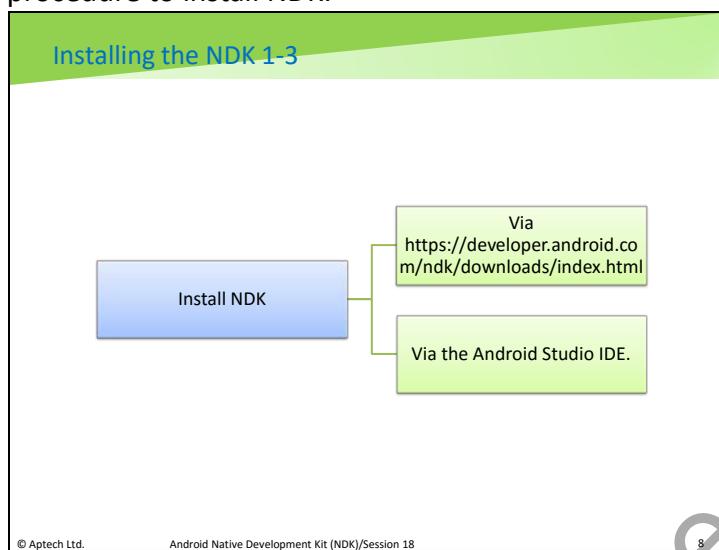
- The **NativeActivity** helper class:
 - Allows to build a native activity.
 - Handles the communication between the Android framework and the app's native code.
 - The application must be declared as 'native' in the **AndroidManifest.xml** file.

© Aptech Ltd. Android Native Development Kit (NDK)/Session 18 7

Explain that the Android NDK has a helper class called the **NativeActivity** class that allows the developer to build a native activity. This class handles the communication between the Android framework and the app's native code. The developer must declare the application as 'native' in the **AndroidManifest.xml** file before developing it.

Slide 8

Let us understand procedure to install NDK.



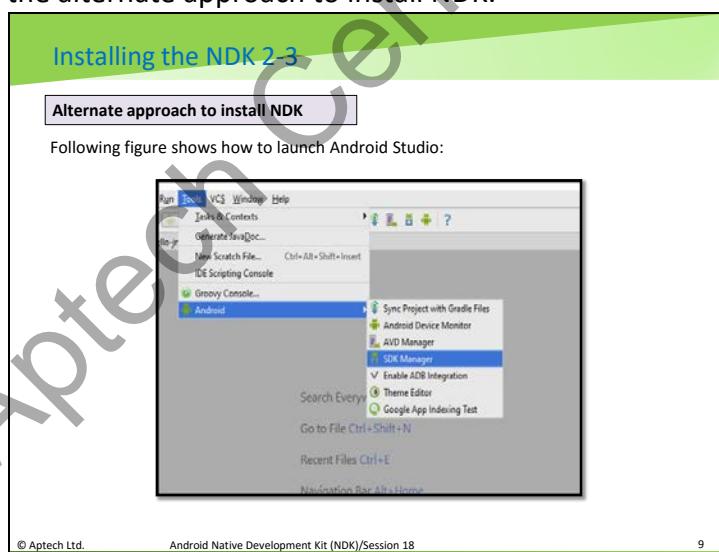
Using slide 8, explain in brief the procedure to install NDK.

Explain that there are two ways to install NDK.

Install via <https://developer.android.com/ndk/downloads/index.html> and then configure the IDE to use this manually or Install it via the Android Studio IDE.

Slide 9

Let us understand the alternate approach to install NDK.



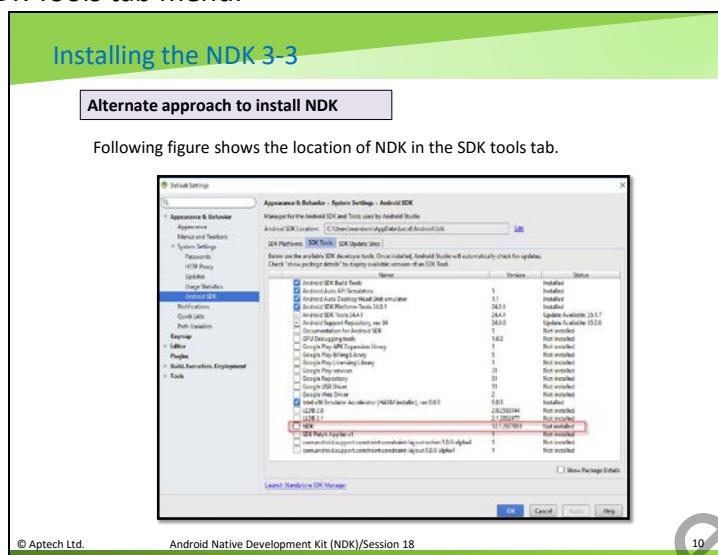
Using slide 9, explain the second approach to install NDK. Explain step by step.

The first step is:

Launch Android Studio and click **Tools** → **Android** → **SDK Manager** as shown in figure.

Slide 10

Understand the SDK Tools tab menu.

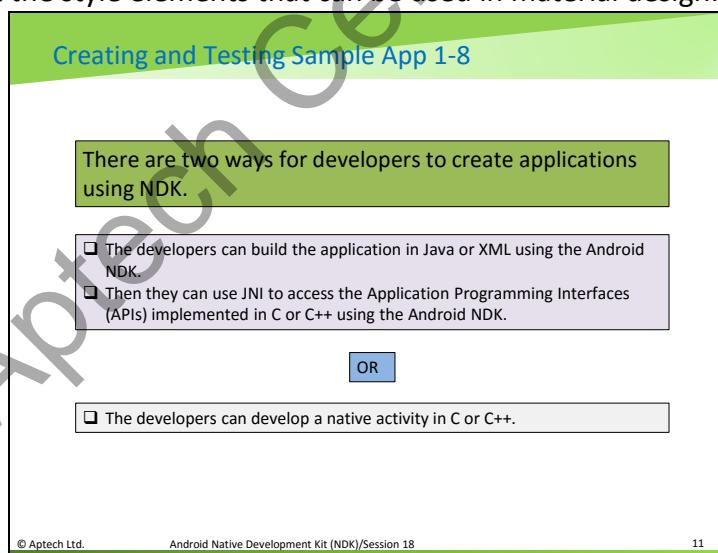


Explain that the next step is to select the **SDK Tools** tab, in the dialog box that is displayed. You will observe NDK as one of the tools with status **Not installed** beside it as shown in figure.

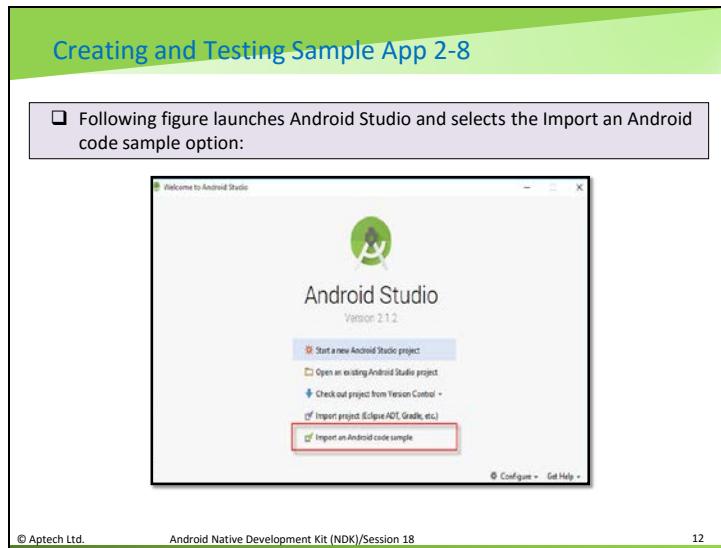
Finally click the check box beside NDK and then click **OK**. Wait until the tools are installed. Say that with this you are ready to create native apps with NDK.

Slides 11 to 15

Understand about the style elements that can be used in material design.



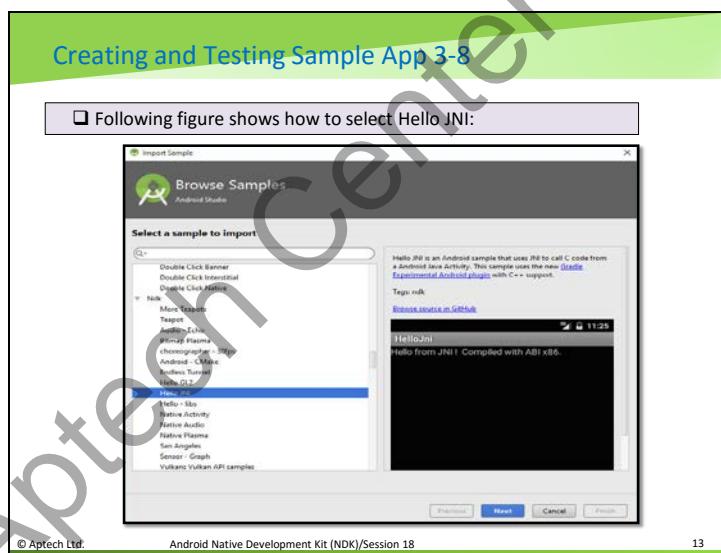
Using slide 11, proceed to talk about the style elements that can be used in material design to customize the apps. Read out the table given on the slide to students and explain.



Using slide 12, tell the students that, let us reuse an existing project sample from the Android Studio to understand native app development.

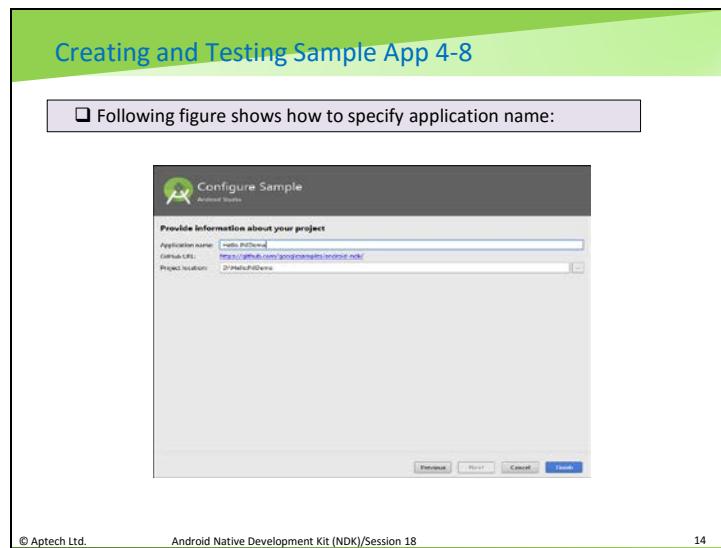
The steps to do this are as follows:

Launch Android Studio and select the **Import an Android code sample** option as shown in figure.

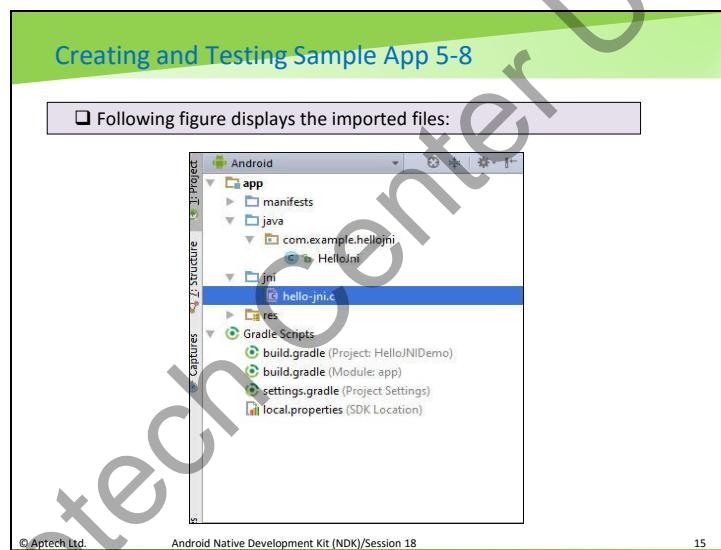


Using slide 13, explain the **Import Sample** dialog box allows you to specify which samples to import.

Now, scroll down until you see **Ndk** and under that, select **Hello JNI**. This a ready made sample application that will call C code within an Android Java activity and Click Next.



Using slide 14, explain that now they need to specify a name for the application and then click **Finish**.



Using slide 15, explain that the application will be created. Tell them that, figure displays the files that are imported as part of the project.

Slides 16 and 17

Let us understand how to create and test sample app..

Creating and Testing Sample App 6-8

Following code snippet shows the code in HelloJni.java:

```
public class HelloJni extends Activity
{
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);

        /* Create a TextView and set its content. the text is retrieved by
         * calling a native function.*/
        TextView tv = new TextView(this);
        tv.setText( stringFromJNI() );
        setContentView(tv);
    }
    /* A native method that is implemented by the 'hello-jni' native library,
     * which is packaged with this application.
    */
}
```

© Aptech Ltd. Android Native Development Kit (NDK)/Session 18 16

Creating and Testing Sample App 7-8

Following code snippet shows the code in HelloJni.java:

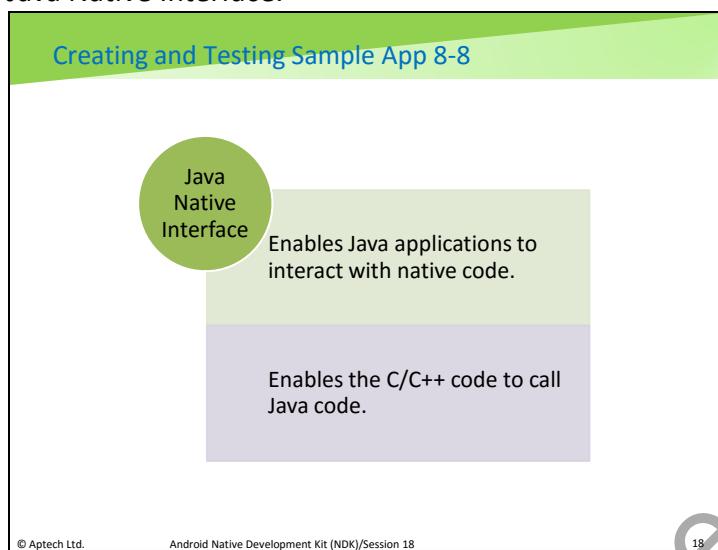
```
public native String stringFromJNI();
    /* This is another native method declaration that is *not*
     * implemented by 'hello-jni'. This is simply to show that
     * you can declare as many native methods in your Java code
     * as you want, their implementation is searched in the
     * currently loaded native libraries only the first time
     * you call them.
    *
    * Trying to call this function will result in a
    * java.lang.UnsatisfiedLinkError exception !
    */
public native String unimplementedStringFromJNI();
    /* this is used to load the 'hello-jni' library on application
     * startup. The library has already been unpacked into
     * /data/data/com.example.hellojni/lib/libhello-jni.so at
     * installation time by the package manager.
    */
    static
    {
        System.loadLibrary("hello-jni");
    }
}
```

© Aptech Ltd. Android Native Development Kit (NDK)/Session 18 17

Explain the code snippet using slides 16 and 17. Say that the code snippet shows the code present in HelloJni.java file.

Slide 18

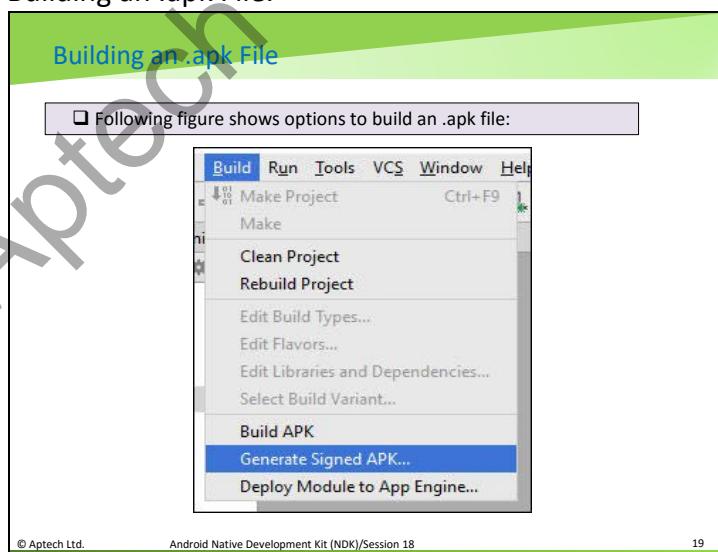
Let us understand Java Native Interface.



Further explain Java Native Interface. Explain that Java Native Interface (JNI) enables Java applications to interact with native code. The JNI and the NDK work together to provide native support to Android apps. NDK is part of the Android framework whereas JNI is not specific to Android and is available to any Java application. Using the JNI, C/C++ code can call Java code, including the standard Android libraries, and the Java code can call native functions defined in C/C++ code. To run the app on an Android device emulator, perform steps similar to any other Android application and execute it. Alternatively, you can create an apk file and test it on an actual phone device.

Slide 19

Let us understand Building an .apk File.



Building an .apk File. Explain an APK file is used to distribute and install the application on Android devices. The APK file has the source code, manifest, assets, resources, and certificates of the app project. To create an APK file for a project, in the Android Studio, click Build menu and select either Build APK or Generate Signed APK.

Ask students to observe the figure.

Explain that once the file is generated, it can be deployed to an actual device.

When an app interacts with native code, it must support different CPU architectures. This means it must be compiled separately for each platform it has to be run on. Each ABI corresponds to a target architecture.

Slide 20

Let us summarize the session.

Summary

- The ndk-build file automatically identifies the project that needs to be built.
- The NDK builds the native shared libraries from the app's native source code.
- The NDK builds static libraries that can be linked with other existing libraries.
- JNI is the programming framework that enables Java code and C or C++ applications to interact with each other.
- Android Studio provides support to install NDK.
- ABI is the interface between two programs that are at different levels.
- The Android NDK has a helper class called the NativeActivity class that allows the developer to build a native activity.

© Aptech Ltd. Android Native Development Kit (NDK)/Session 18 20

Using this slide, summarize the key points of this session. Explain the following points in brief.

- The ndk-build file automatically identifies the project that needs to be built.
- The NDK builds the native shared libraries from the app's native source code.
- The NDK builds static libraries that can be linked with other existing libraries.
- JNI is the programming framework that enables Java code and C or C++ applications to interact with each other.
- Android Studio provides support to install NDK.
- ABI is the interface between two programs that are at different levels.
- The Android NDK has a helper class called the NativeActivity class that allows the developer to build a native activity.

Next, let us check your understanding of the topics covered so far in this session.

18.3 Post Class Activities for Faculty

You can discuss queries of the session here.