Android App Components:
Module Introduction

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA
Introduction to Android

- Android is designed as a layered architecture
Introduction to Android

- Android is designed as a layered architecture

- Variant of the GNU Linux kernel that manages some device hardware & mediates access to shared resources

See [en.wikipedia.org/wiki/Android_(operating_system)#Linux_kernel](en.wikipedia.org/wiki/Android_(operating_system)#Linux_kernel)
Introduction to Android

• Android is designed as a layered architecture

Android’s middleware infrastructure layers enhance app programmer productivity by providing many reusable components & services that extend the hardware-centric mechanisms available from the Android Linux kernel.

See en.wikipedia.org/wiki/Middleware#Other_examples
Introduction to Android

- Android is designed as a layered architecture

Frameworks in the Application Framework layer are written in Java & provide abstractions of the underlying native libraries & Android Runtime capabilities to apps

See www.dre.vanderbilt.edu/~schmidt/frameworks.html
Introduction to Android

• Android is designed as a layered architecture

Most users of Android devices primarily interact with this layer, e.g., to make phone calls, browse the Web, send/receive email, take photos, etc.
Introduction to Android

- Understanding these layers makes you a more effective “full stack” developer
Introduction to Android

- Android has 1,000’s of classes, so it’s important to have a “bird’s eye” view!

See [www.makelinux.net/android/classes](http://www.makelinux.net/android/classes)
This module focuses on concepts, not code.

```java
public void downloadAndDisplay(DownloadContext downloadContext) {
    // Create an AsyncTask to download an image in the background
    // and display it to the user in the UI Thread.
    mDownloader = new AsyncTask<String, Void, Bitmap>() {
        /**
         * Called by the AsyncTask framework in the UI Thread to
         * perform initialization actions.
         */
        protected void onPreExecute() {
            // Show the toast before starting the download in a
            // background thread.
            downloadContext.showToast("downloading via AsyncTask");
        }
        /**
         * Download a bitmap image in a background thread.
         *
         * @param params          The url of a bitmap image
         * @return The Bitmap representation of the downloaded image.
         */
        protected Bitmap doInBackground(String... urls) {
            // Download the image, which can block since we're in
            // a background thread.
            return downloadContext.downloadBitmap(urls[0]);
        }
        /**
         * Called after an operation executing in the background
         * completes to set the bitmap image to an image view and
         * dismiss the progress dialog.
         *
         * @param image            The bitmap image
         */
        protected void onPostExecute(Bitmap image) {
            // Display the downloaded image to the user.
            downloadContext.displayBitmap(image);
        }
    }.execute(downloadContext.getUrl());
}
```
You needn’t know all these details to write Android apps!
End of Module
Introduction