Course Logistics
Welcome to CS 495/595

- **Timings:** Monday 3:00pm to 5:35pm
- **Location:** ECSB 3104
- **Instructor:** Tamer Nadeem
  - Ph.D from Univ. of Maryland, 2006
  - Research in Networks, Dist Sys, Mobile Comp.
  - Email: nadeem@cs.odu.edu
  - Office: ECSB 3204
- **Office Hours:** Mon 1:25pm-2:55pm, 5:40pm-7:10pm, or by appointment
Welcome to CS 495/595

• Teaching Asst.: Mostafa Uddin
  Email: muddin@cs.odu.edu
  Office: ECSB 3106

• Office Hours: Wed 9:30am-12:30pm, or by appointment
Welcome to CS 495/595

• Prerequisites: Comfortable with Java

• Grading:
  • Participation: 10%
  • Midterm: 25%
  • Programming Assignments: 40%
  • Final Project: 25%
Welcome to CS 495/595

• Class Webpage:
  • http://www.cs.odu.edu/~cs495/
  • Please check course website frequently

• Make up classes:
  • Will be occasionally necessary due to travel
  • Fixed schedule versus case by case basis?
Welcome to CS 495/595

• Text:

  • Required:
    • Wei-Meng Lee, "Beginning Android 4 Application Development"

  • Recommended:
    • Reto Meier, "Professional Android 2 Application Development"
    • James Steele, Nelson To, "The Android Developer's Cookbook"

• Android Developers (Dev Guide, Reference, etc.):
Welcome to CS 495/595

• Academic Integrity / Honor Code:

• "I pledge to support the honor system of Old Dominion University. I will refrain from any form of academic dishonesty or deception, such as cheating or plagiarism. I am aware that as a member if the academic community, it is my responsibility to turn in all suspected violators of the honor system. I will report to Honor Council hearings if summoned."

• Please refer to ODU Honor Council’s webpage: http://orgs.odu.edu/hc/
Welcome to CS 495/595

• Course Policy:
  • Grading:
    • 90-100    A
    • 80-89     B
    • 70-79     C
    • 0-69      F
  • Late assignments are not accepted.
  • Attendance
  • Email

Please refer to class webpage for more details.
Course Overview
This Course

- Introduces fundamentals of application development for Android phones

- Goals of this course:
  - Help you learn about mobile app development and best practices
  - Provide you with the tools, knowledge, and excuse to create a novel mobile app that helps solve a serious problem that strengthens your programming portfolio

- Envisions new practical mobile applications/services
Class Responsibilities

• I will lead lectures
  • You present 1-2 paper(s) in entire semester (25 minutes)
  • 2-3 students presentation per class
  • Some classes will include coding

• For every class, read the readings list before the class

• Assignments should be on time
Course Structure

• 1 mid term?, No Final Exam
  • Tentative date of mid-term: Nov 19th

• Semester-long class project
  • In groups of 2 (max 3).
    • Individual projects are allowed by permission
  • Focus on this from early on

• Class ends with a final project presentation & demo
Participation / Presentation

• Ask lots of questions. Period.
  • I strongly encourage you to ask, disagree, debate

• Class presentation
  • You present 1-2 paper (25 minutes)
  • Check class schedule by next week for reading papers
  • Email me any paper you are interested in
  • Pick an open slot (check class schedule)
    • Earlier you pick, more options you have to choose from
    • Deadline is Sep 07, 2012

• Email me your choice of paper (and date)
  • Don’t worry about not knowing the topic of paper
    • By that time, you will know enough
Thoughts on Reading Assignments

• Know why you are reading the paper
  • Reading for absorbing concepts (class assignment)
    • Read fully, think, reread, ask, challenge
  • Reading for excitement (deciding project topic)
    • Read initial parts, don’t try to understand everything, get a feel
Course Term Project

• Initial proposal due Oct 10 after Fall break
  • 1-page progress report every 2 weeks (due Friday night)
  • Final report + demo + presentation

• Projects consist of:
  • Application/Service identification
  • Solution design
  • Implementation

• Discuss your thoughts and ideas with me
  • They need not be cooked, and can have many flaws
  • Statistically, every 18 ideas lead to one decent idea

• If you like an area/direction
  • Search and Read many many related references
More on Projects

• Project ideas take time … think now and then
  • Spending 3 hours for 10 days better than 10 hours for 3 days

• Find a project partner(s) early
  • Search and discuss App/Services ideas

• Everyone in the class will try/critique apps from other teams

• At end of the course we will vote for the Top App

• Possible Application/Service domains:
  • Transportation
  • Education
  • Health
  • Energy
  • Smart Home
Labs/Facilities

• Development Environment:
  • Your laptop
  • ECSB 3104 (Open Research Lab)
  • SmartApp Lab (under construction)

• Collaboration:
  • Internet
  • BlackBoard Discussion (http://clt.odu.edu/bb/)
  • Friends/Colleagues
Black Board Discussion Forum

http://clt.odu.edu/bb/

My ODU Courses

201110_FALL_CS495_25906: TPC: APP DEV FOR SMART DEVICES

Discussion Board

Forum

Discussion Board

Course Discussions: Dummy Forum

Thread
Remove
Collect
Mark Read
INTRODUCTION
Mobile Computing

• Driven by technology and vision
  • wireless communication technology
  • global infrastructure
  • device miniaturization
  • mobile computing platforms

• The field is moving fast

• “People and their machines should be able to access information and communicate with easily and securely, in any medium each other or combination of media – voice, data, image, video, or multimedia – any time, anywhere, in a timely, cost-effective way.”, Dr. G. H. Heilmeier, Oct 1992

• “The mobile device will be the primary connection tool to the Internet for most people in the world in 2020.”, PEW Internet and American Life Project, Dec. 2008
Smart Devices

- A **smart device** is a device that is digital, active, computer networked, is user reconfigurable and that can operate to some extent autonomously.

- A **smart device** is a ubiquitous computing device: a device that exhibits some properties of ubiquitous computing including artificial intelligence.

- **Mark Weiser** categorized ubiquitous devices:
  - **Tabs**: accompanied or wearable centimeter sized devices, e.g., smartphones, smart cards
  - **Pads**: hand-held decimeter-sized devices, e.g., laptops
  - **Boards**: meter sized interactive display devices, e.g., horizontal surface computers and vertical smart boards.
Mobile Devices

- Multi-purpose devices (e.g., personal office, mobile phone, camera, etc.)
- Mobility: loosely-bound vs. tightly-bound to users
- Personalized
- Operates as a single portal, e.g., a Web portal.
  - Internal application services
  - External services typically accessed local area wireless network
- Intermittent resource access
- A locus of control that resides in the smart device.
- Networked, distributed and transparently accessible.
- Context awareness
Smart Devices at Home/Office

- WiFi
- satellite
- UWB
- Bluetooth
- Cellular
- WiFi 802.11g/n
Smart Devices on Move

Source: http://www.ece.uah.edu/~jovanov/whrms/
Smart Devices on Road

GSM/UMTS, cdmaOne/cdma2000, WLAN, GPS DAB, TETRA, ...

road condition, weather, location-based services, emergency
Smart Devices for Intelligent Transportation

Mobile Millennium Traffic in San Francisco and the Bay Area

Source: http://traffic.berkeley.edu/

CarTel Project at MIT

Source: http://cartel.csail.mit.edu/doku.php
Smart Biomedical Systems

**In-body smart devices**
- sensors/monitoring devices
- drug delivery systems
- medical robots
- neural implants

**Wireless telemedicine**

**Wireless network**

**Recovery from nerve damage**
Habitat monitoring
Supply chain management
Mobile Social Networking
Smart Phone

Trend: everything in one small device
Smart Phone - Overview

• A smartphone is a high-end mobile phone

• Combines the functions of
  • mobile phone
  • personal digital assistant (PDA)
  • portable media players
  • camera phones
  • high-resolution touchscreens
  • web browsers
  • GPS navigation
  • Wi-Fi and mobile broadband access
  • etc.

• Feature phone vs. Smartphone

  • Feature phone
    • proprietary firmware & limited platforms

  • Smartphone
    • open and complete mobile operating system
    • tightly integrate with the user interface and phone features
    • relies on a more powerful application programming interface (API)
Smart Phone - History

Early Years:

IBM Simon (1992)

Nokia 9000 (1996)
Nokia 9110i (1998)
Nokia 9110i (2000)

Ericsson GS88 (1997)

Symbian:

Ericsson R380 (2000)
Ericsson P800 (2002)

Nokia 9210 (2000)
Nokia 9500 (2005)
Nokia E90 (2007)

Nokia N95 (2007)
Nokia N8 (2010)
Smart Phone - History

Palm, Windows, BlackBerry:

- **Palm Kyocera 6035** (2001)
- **Windows CE Pocket PC** (2002)
- **Windows Phones 7** (2007)
- **RIM BlackBerry** (2002)
- **Palm OS Treo** (2002)

iPhone:

- iPhone (2007)
- iPhone 3G (2008)
- iPhone 4 (2010)
- iPhone 4S (2011)

Android:

- **Android G1 – HTC Dream** (2008)
- **Nexus One** (2010)
- **Nexus S** (2011)
## Smart Phones - Statistics

<table>
<thead>
<tr>
<th>OS</th>
<th>2010</th>
<th>2011</th>
<th>2012</th>
<th>2015</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbian</td>
<td>111,577</td>
<td>89,930</td>
<td>32,666</td>
<td>661</td>
</tr>
<tr>
<td>Market Share (%)</td>
<td>37.6</td>
<td>19.2</td>
<td>5.2</td>
<td>0.1</td>
</tr>
<tr>
<td>Android</td>
<td>67,225</td>
<td>179,873</td>
<td>310,088</td>
<td>539,318</td>
</tr>
<tr>
<td>Market Share (%)</td>
<td>22.7</td>
<td>38.5</td>
<td>49.2</td>
<td>48.8</td>
</tr>
<tr>
<td>Research in Motion</td>
<td>47,452</td>
<td>62,600</td>
<td>79,335</td>
<td>122,864</td>
</tr>
<tr>
<td>Market Share (%)</td>
<td>16.0</td>
<td>13.4</td>
<td>12.6</td>
<td>11.1</td>
</tr>
<tr>
<td>iOS</td>
<td>46,598</td>
<td>90,560</td>
<td>118,848</td>
<td>189,924</td>
</tr>
<tr>
<td>Market Share (%)</td>
<td>15.7</td>
<td>19.4</td>
<td>18.9</td>
<td>17.2</td>
</tr>
<tr>
<td>Microsoft</td>
<td>12,378</td>
<td>26,346</td>
<td>68,156</td>
<td>215,998</td>
</tr>
<tr>
<td>Market Share (%)</td>
<td>4.2</td>
<td>5.6</td>
<td>10.8</td>
<td>19.5</td>
</tr>
<tr>
<td>Other Operating Systems</td>
<td>11,417.4</td>
<td>18,392.3</td>
<td>21,383.7</td>
<td>36,133.9</td>
</tr>
<tr>
<td>Market Share (%)</td>
<td>3.8</td>
<td>3.9</td>
<td>3.4</td>
<td>3.3</td>
</tr>
<tr>
<td>Total Market</td>
<td>296,647</td>
<td>467,701</td>
<td>630,476</td>
<td>1,104,898</td>
</tr>
</tbody>
</table>
Smart Phones - Statistics

Chart by The Mac Observer, from Gartner data
Mobile Applications

• What are they?
  • Any application software that is developed for small low-power handheld devices such as personal digital assistants, enterprise digital assistants or mobile phones.

• Users on mobile phone’s
  • Typically check the news, weather, email, or their social networks
  • Often have a choice between the mobile web version or a specially- created mobile app.

• Mobile App Types
  • Web apps: run in a web browser
    • HTML, JavaScript, Flash, server-side components, etc.
  • Native: compiled binaries for the device
    • Not cross-platform, but more interesting options
Web App vs. Native App

The average Android user spends almost an hour per day interacting with web and apps

Proportion of Time Spent on Web vs. Apps
Nielsen Smartphone Analytics, June 2011

56 Minutes per Day

Source: Nielsen

Apps 67%
Web 33%
Native Development Environments

• Options
  • Java ME
  • .NET Compact Framework (C++, C#, VB.NET) for Windows Mobile
  • Qualcomm’s BREW (C or C++)
  • Symbian (C++)
  • BlackBerry (Java)
  • Android (Java)
  • iPhone (Objective-C)

Is having so many choices and so much industry turmoil/competition a good thing?
Development Environments

- Most platforms have an SDK that you can download and build against
- Every platform has an emulator that you can use to test your apps
- Most emulators are configurable to match a variety of mobile devices
  - Various screen sizes, memory limitations, tablets, etc.
  - In practice, emulators quite limited

IDE - integrated development environment that provides tools to allow a developer to write, test and deploy applications into the target platform environment.
xC ode IDE & iPhone Emulator
Eclipse and Android Emulator
Smart Phone – the good

- Always with the user
- Increasingly powerful devices
- Typically GPS capable
- Typically have accelerometer
- Designed for communication
- 2+ types of wireless connections
- Many apps are free or low-cost
Smart Phone – the not-so-good

• Limited battery life
• Limited processor speed
• Limited RAM
• Limited, unreliable, and slow network access
• Limited screen size
• Limited permanent storage capacity
• Limited or awkward input
  • (none great: soft keyboard, phone keypad, touch screen, stylus, speech)
• Inconsistent platforms across devices
• High costs associated with data transfer
Android

• Android, Inc. founded in Palo Alto, California in October 2003

• Google acquired Android Inc. in August 2005

• Developed a mobile device platform powered by the Linux kernel

• Google marketed the platform to handset makers and carriers on the premise of providing a flexible, upgradable system

• On November 2007, the Open Handset Alliance, a consortium of several companies (e.g., Broadcom, Google, HTC, Intel, etc. unveiled itself). The goal is to develop open standards for mobile devices.

• Open Handset Alliance unveiled their first product, Android, a mobile device platform built on the Linux kernel version 2.6

• Android OS (open source) released in October 2008
Why Android

• Simple and powerful SDK
• No licensing fees
• Excellent documentation, and a thriving developer community

• From commercial perspective
  • Requires no certification for becoming an Android developer
  • Provides the Android Market for distribution and monetization of your application
  • Has no approval process for application distribution
  • Gives you total control over your brand and access to the user’s home screen
Android Overview

• Open source OS and development platform
  • In theory, you can change anything
  • In practice....

• Hardware reference design

• Linux OS kernel

• Open-source libraries for app development
  • E.g., SQLite, Webkit, OpenGL, media managerMobile Application Design
    and D

• SDK and tools

• Preinstalled apps

• Wild west of app stores: the Market
Android Version

• **Initial**: 1.5 (Cupcake) (Apr 2009), 1.6 (Donut) (Sep’09)

• **2.0/2.1 (Eclair)** (Oct’09/Jan’10): new web browser, new user interface, support for HTML5, Geolocation API, enhanced camera features / voice controls, 5 homescreens, animated backgrounds.

• **2.2 (Froyo)** (May’10): speed improvement, Chrome v8 JavaScript engine, Wi-Fi tethering, Adobe Flash support

• **2.3 (Gingerbread)** (Dec’10): Near Field Communication

• **3.0 (Honeycomb)** (Feb’11): tablet-oriented release, supports multicore processors, hardware acceleration for graphics

• **3.1 (Honeycomb)** (May’11): directly transfer content from USB devices

• **3.2 (Honeycomb)** (July’11): adds several new capabilities for users and developer (e.g., providing developers with more precise control over the UI)

• **4.0 (Ice Cream Sandwich)** (Oct’11): combination of Gingerbread and Honeycomb

• **4.1 (Jelly Bean)** (Jul’12): improve user interface

http://en.wikipedia.org/wiki/Android_version_history
Features and Specifications I

- Platform is adaptable to larger, VGA, 2D graphics library, 3D OpenGL graphics library
- **Storage** - SQLite, a lightweight relational database
- **Connectivity** - supports connectivity technologies including GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, NFC and WiMAX.
- **Messaging** – SMS, MMS, threaded text messaging, Push Messaging service.
- **Multiple language support**
- **Web browser** - based on the open-source WebKit layout engine, coupled with Chrome's V8 JavaScript engine.
- **Java support** – no Java Virtual Machine, Dalvik executables and run on Dalvik
Features and Specifications II

- **Media support** - audio/video/still media formats: WebM, H.263, H.264, MPEG-4 SP, WAV, JPEG, PNG, GIF, BMP, etc.

- **Streaming media support** - RTP/RTSP streaming (3GPP PSS, ISMA), HTML5 `<video>` tag, Adobe Flash Streaming (RTMP), HTTP Dynamic Streaming, Apple HTTP Live Streaming

- **Additional hardware support** - video/still cameras, touchscreens, GPS, accelerometers, gyroscopes, magnetometers, dedicated gaming controls, proximity and pressure sensors, thermometers, accelerated 2D bit blits and accelerated 3D graphics

- **Multi-touch**

- **Bluetooth** - supports A2DP, AVRCP, sending files (OPP), accessing the phone book (PBAP), voice dialing and sending contacts between phones. Keyboard, mouse and joystick (HID)
Features and Specifications III

- **Video calling** – no native video calling, but some handsets have a customized version of the operating system that supports it. Video calling through Google Talk is available in Android 2.3.4 and later. Skype 2.1 offers video calling in Android 2.3, including front camera support.

- **Multitasking**

- **Voice based features** - Google search through voice and voice actions for calling, texting, navigation, etc.

- **Tethering**

*Google describes Android as: “The first truly open and comprehensive platform for mobile devices, all of the software to run a mobile phone but without the proprietary obstacles that have hindered mobile innovation.”*
Android Architecture
Android SDK Features

- No licensing, distribution, or development fees
- Wi-Fi hardware access
- GSM, EDGE, and 3G networks for telephony or data transfer, allowing you to make or receive calls or SMS messages, or to send and retrieve data across networks
- Comprehensive APIs for location-based services such as GPS
- Full multimedia hardware control including playback and recording using the camera and microphone
- APIs for accelerometer and compass hardware
- IPC message passing
- Shared data stores
- An integrated open source WebKit-based browser
- Full support for app that integrate Map controls as part of their user interface
- **Peer-to-peer (P2P) support using Google Talk**
- Mobile-optimized hardware-accelerated graphics including a path-based 2D graphics library and support for 3D graphics using OpenGL ES
- Media libraries for playing and recording a variety of audio/video or image formats
- **An application framework that encourages reuse of application components and the replacement of native applications**
Android Development I

• “Open philosophy”
• Application framework that encourages reuse of app components
• Access to (much) hardware
• Release and fix mentality (or it seems like it, sometimes)
• Background services
• Shared data and inter-process communication
  • Notifications (via UI)
  • Intents
  • Content providers (managed access to app private data)
• All apps created equal
• Native Development Kit (NDK) (C++ tinkering under the hood)
• OS optimization of memory and process management
Android Development II

• Expect...
  • Limited processing power
  • Limited RAM
  • Limited permanent storage capacity
  • Small screen and low resolution
  • High cost of data transfer
  • Slow data transfer rates with high latency
  • Unreliable data connections

• Application framework that encourages reuse of application components
Android Development III

• “Screens” are Activities that are chained with lightweight exchange of data between them

• OS can handle stack of Activities if you want it to (e.g., back)

• OS Manages process lifetime (app assassin)

• You MUST
  • Ensure that your app is ready for swift death
  • Yet, it must remain response and/or restart in the background
  • Must come to the foreground quickly
Application Types

- **Foreground**
  - Useful when being used.
  - Suspended otherwise

- **Background**
  - Apart from when being configured, spends most of lifetime hidden (e.g., call screening app)

- **Intermittent**
  - Some interaction but mostly in the background (e.g., media player)

- **Widget**
  - Home screen status update
Application Good behavior

• Is well behaved
• Switches seamlessly from background to foreground
• Is polite (e.g., stealing focus)
• Presents a consistent user interface
• Is responsive
Behavior police - process assassin

• Two conditions monitored
  • Must respond to any user action (e.g., key press) within 5s
  • A BroadcastReceiver must return from its OnReceive handler within 10s
Android Market (https://market.android.com/apps)

- Online software store developed by Google for Android devices
- As of December 2010 there were about 200,000 games, applications and widgets available on the Android Market.

- At end of June 2011 Google said there had been more than 6 billion Android apps installed
- The operating system itself is installed on 130 million total devices.
- Only devices that comply with Google's compatibility requirements are allowed to preinstall Google's closed-source Android Market app and access the Market.

- The Market filters the list of applications presented by the Market app to those that are compatible with the user's device,
Android Market

Most common SDK versions, August 25, 2011

New Applications & Games in the Android Market by Month (10^3)

AndroLib

Page 62 Fall 2012 CS 495/595 - App Development for Smart Devices
Android Tools
Objective

• Understand Android Tools
• Setup Android Development Environment
• Create HelloWorld Application
• Understand HelloWorld Structure
• Familiarize with Android Application Types and Components
• Understand the Role of Android Activity
• Know how to Build and Run Android Application
• Introduce Debug and Publish
Developing for Android

- Eclipse
- Android SDK
- Android Development Tools (ADT)
- Android Virtual Devices (AVD) & SDK Manager
- The Android Emulator
- Dalvik Debug Monitor Services (DDMS)
- The Android Debug Bridge (ADB)
Eclipse
Eclipse

• Eclipse IDE for Java Developers
  • Open http://www.eclipse.org/
  • Select "Downloads"
  • Select Eclipse IDE for Java Developers, 149MB
  • Select Windows32

• Extract zip file to c:\eclipse

• Installation Tutorial:
  • http://archive.eclipse.org/technology/phoenix/europa/EclipseIdeForJavaDevelopers/

• “Test First Development” Tutorial:
  • http://archive.eclipse.org/technology/phoenix/europa/TestFirstWithEclipse/
Android SDK

- Check Your Development Computer
  - Supported Operating Systems
    - 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 (tested on Ubuntu Linux, Lucid Lynx)
  - Supported Development Environments
    - JDK SE 6.0 – 32bit version (including JRE)
    - Eclipse IDE for Java Developers - Indigo version – 32bit version
    - Android Development Tools (ADT) plugin

- SDK Installation
  - Android SDK starter package
    - installer_r12-windows.zip
  - Android SDK Components
    - Run Android SDK setup
    - From “Available packages” window, select all items and then click install
ADT plugin

- Start Eclipse, then select Help > Install New Software
- Click Add, in the top-right corner.
- In the Add Repository dialog that appears, enter "ADT Plugin" for the Name and the following URL for the Location: https://dl-ssl.google.com/android/eclipse/
- Click OK
- Wait and you should see "Developer Tools" in the Available Software dialog, select the checkbox next to Developer Tools and click Next.
- In the next window, you'll see a list of the tools to be downloaded. Click Next.
- Click Finish.
Modify your ADT preferences in Eclipse to point to the Android SDK directory:

1) Select Window > Preferences... to open the Preferences panel (Mac OS X: Eclipse > Preferences).

2) Select Android from the left panel.

3) You may see a dialog asking whether you want to send usage statistics to Google. If so, make your choice and click Proceed. You cannot continue with this procedure until you click Proceed.

4) For the SDK Location, locate your downloaded SDK directory. Click Apply, then OK. (Spaces in path causes an error, if you are using Windows64. For example, `C:\Program Files(x86)\Android\android-sdk` → `C:\PROGRA~2\Android\android-sdk`.)
## Android Emulator Hardware Option

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device ram size</td>
<td>The amount of physical RAM on the device, in megabytes. Default value is &quot;96&quot;.</td>
</tr>
<tr>
<td>Touch-screen support</td>
<td>Whether there is a touch screen or not on the device. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>Trackball support</td>
<td>Whether there is a trackball on the device. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>Keyboard support</td>
<td>Whether the device has a QWERTY keyboard. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>DPad support</td>
<td>Whether the device has DPad keys. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>GSM modem support</td>
<td>Whether there is a GSM modem in the device. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>Camera support</td>
<td>Whether the device has a camera. Default value is &quot;no&quot;.</td>
</tr>
<tr>
<td>Maximum horizontal camera pixels</td>
<td>Default value is &quot;640&quot;.</td>
</tr>
<tr>
<td>Maximum vertical camera pixels</td>
<td>Default value is &quot;480&quot;.</td>
</tr>
<tr>
<td>GPS support</td>
<td>Whether there is a GPS in the device. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>Battery support</td>
<td>Whether the device can run on a battery. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>Accelerometer</td>
<td>Whether there is an accelerometer in the device. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>Audio recording support</td>
<td>Whether the device can record audio. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>Audio playback support</td>
<td>Whether the device can play audio. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>SD Card support</td>
<td>Whether the device supports insertion/removal of virtual SD Cards. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>Cache partition support</td>
<td>Whether we use a /cache partition on the device. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>Cache partition size</td>
<td>Default value is &quot;66MB&quot;.</td>
</tr>
<tr>
<td>Abstracted LCD density</td>
<td>Sets the generalized density characteristic used by the AVD's screen. Default value is &quot;160&quot;.</td>
</tr>
<tr>
<td>Trackball support</td>
<td>Whether there is a trackball present.</td>
</tr>
</tbody>
</table>
## Android Emulator Controls

<table>
<thead>
<tr>
<th>Key</th>
<th>Emulated Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Escape</td>
<td>Back button</td>
</tr>
<tr>
<td>Home</td>
<td>Home button</td>
</tr>
<tr>
<td>F2, PageUp</td>
<td>Menu button</td>
</tr>
<tr>
<td>Shift-F2, PageDown</td>
<td>Start button</td>
</tr>
<tr>
<td>F3</td>
<td>Call/Dial button</td>
</tr>
<tr>
<td>F4</td>
<td>Hangup/EndCall button</td>
</tr>
<tr>
<td>F5</td>
<td>Search button</td>
</tr>
<tr>
<td>F7</td>
<td>Power button</td>
</tr>
<tr>
<td>Ctrl-F3, Ctrl-KEYPAD_5</td>
<td>Camera button</td>
</tr>
<tr>
<td>Ctrl-F5, KEYPAD_PLUS</td>
<td>Volume up button</td>
</tr>
<tr>
<td>Ctrl-F6, KEYPAD_MINUS</td>
<td>Volume down button</td>
</tr>
<tr>
<td>KEYPAD_5</td>
<td>DPAD center</td>
</tr>
<tr>
<td>KEYPAD_4, KEYPAD_6</td>
<td>DPAD left, DPAD right</td>
</tr>
<tr>
<td>KEYPAD_8, KEYPAD_2</td>
<td>DPAD up, DPAD down</td>
</tr>
<tr>
<td>F8</td>
<td>Toggle cell network on/off</td>
</tr>
<tr>
<td>F9</td>
<td>Toggle code profiling (when -trace set)</td>
</tr>
<tr>
<td>Alt-ENTER</td>
<td>Toggle fullscreen mode</td>
</tr>
<tr>
<td>Ctrl-T</td>
<td>Toggle trackball mode</td>
</tr>
<tr>
<td>Ctrl-F11, KEYPAD_7</td>
<td>Rotate screen orientation to previous or next layout</td>
</tr>
</tbody>
</table>
Your First Android Project

• Select File -> New -> Project… -> Android -> Android Project and create the Android project

• right-click the project and select Run As > Android Application
Project Structure

- **src/** - Java packages. Each package can have multiple .java files representing different classes.
- **res/layout/** - XML files that specify the layout of each screen.
- **res/values/** - XML files used as references by other files.
- **res/drawable-hdpi/**, **res/drawable-mdpi/**, and **res/drawable-ldpi/** - high, medium, and low dots-per-inch resolution pictures.
- **res/color**, **res/menu**, **res/anim**
- **assets/** - additional non-media files.
- **AndroidManifest.xml** specifies the project to the Android OS.
- **Auto-generated files (do not modify):**
  - **gen/** contains auto-generated code. Class **R.java** generated by **Android Asset Packaging Tool (aapt)**.
  - **default.properties** contains project settings.
## Resources, Package, Manifest

<table>
<thead>
<tr>
<th>Resource</th>
<th>Reference in Java</th>
<th>Reference in XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>res/layout/main.xml</td>
<td>R.layout.main</td>
<td>@layout/main</td>
</tr>
<tr>
<td>res/drawable-hdpi/icon.png</td>
<td>R.drawable.icon</td>
<td>@drawable/icon</td>
</tr>
<tr>
<td>@+id/home_button</td>
<td>R.id.home_button</td>
<td>@id/home_button</td>
</tr>
<tr>
<td>&lt;string name=&quot;hello&quot;&gt;</td>
<td>R.string.hello</td>
<td>@string/hello</td>
</tr>
</tbody>
</table>

**HelloAndroid Manifest**

```xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.example.hello"
    android:versionCode="1"
    android:versionName="1.0">
    <uses-sdk android:minSdkVersion="8" />

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".Hello"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```
Android Software Stack
Application Framework

- **Activity Manager**
  - Manages the lifecycle of applications

- **Contents Provider**
  - Store and retrieve data and make it accessible to all applications

- **View System**
  - Handles GUI related tasks

- **Package Manager**
  - Retrieve various kinds of info. related to the currently installed application on the device.

- **Resource Manager**
  - Provides access to non-code resources such as icons etc.

- **Location Manager**
  - Location-based and related services.

- **Notification Manager**
  - Executes and manages all Notifications, alerts etc.

...
Android Component Model

Packaging: APK File (Android Package)
- Collection of components
- Components share a set of resources
  - Preferences, Database, File space
- Components share a Linux process
  - By default, one process per APK
- APKs are isolated
  - Communication via Intents or AIDL
- Every component has a managed lifecycle

Task (what users know as applications)
- Collection of related activities
- Capable of spanning multiple processes
- Associated with its own UI history stack
- Processes are started & stopped as needed
- Processes may be killed to reclaim resources
Android Application Components

i) **Activity**: Activity is a visual screen for interaction of user with the application. Depends upon design, an application may consists of one or more activities.

ii) **Views**: The User interface of an Activities is build with widgets.

iii) **Service**: Service do not have a visual interface, it runs in the back ground, like play back music and fetching data from the network.

iv) **Broadcast Receiver**: Broadcast receiver receive broadcast announcements and response to them according to the situation.

v) **Content Provider**: Content provider is a SQLite database, which supports the sharing and accessing of data among applications.

vi) **Intents**: Asynchronous messages which allow the application to request functionality from other services or activities.

vi) **Others parts** are Android widgets/Live Folders and Live Wallpapers.
Example of Components Interactions

- Starting an Activity for a Result
- Communicating with a Service
- Querying a Content Provider
- Receiving an Intent Broadcast
Types of Android Applications

- **Foreground Activity**: An application that’s only useful when it’s in the foreground and is effectively suspended when it’s not visible.

- **Background Service**: An application with limited interaction that, apart from when being configured, spends most of its lifetime hidden. Examples of this include call screening applications or SMS auto-responders.

- **Intermittent Activity**: Expects some interactivity but does most of its work in the background. Often these applications will be set up and then run silently, notifying users when appropriate. A common example would be a media player.

- **Widgets**: Interactive visual components that users can add to their home screens. Usually used to display information such as battery levels, weather forecasts, or the date and time.
Android’s Activity

- An "activity" is an application component that provides a screen with which users can interact.

- **Activity is usually a single screen**
  - Implemented as a single class extending Activity
  - Displays user interface controls (views)
  - Reacts on user input / events

- **An application typically consists of several activities**
  - Each screen is typically implemented by one activity
  - Each activity can then start another activity (new screen)
  - An activity may return a result to the previous activity

- "main" activity is presented to the user when launching the application for the first time.

- Each activity receives callbacks due to a change in its state during its **lifecycle** — whether the system is creating it, stopping it, resuming it, or destroying it.
Building Android Project

• Eclipse outputs an .apk file automatically to the bin folder of the project

• Contains all information to run the application, such as compiled .dex files, a binary version of AndroidManifest.xml, compiled resources (resources.arsc) and uncompiled resource files.

• Eclipse automatically builds application, enables debugging and signs the .apk with a debug key, by default.
Running

• **Running on the emulator**
  - To run (or debug) your application, select **Run** > **Run** (or **Run** > **Debug**) from the Eclipse menu bar.
  - Be certain to create multiple AVDs upon which to test your application.

• **Enable Running on a device**
  - Ensure application is debuggable by setting `android:debuggable="true"` of the `<application>` element in AndroidManifest.xml.
  - Set up the device to allow installation of non-Market applications. On the device, go to **Settings** > **Applications** and enable **Unknown sources**.
  - Enable USB Debugging on the device. On the device, **Settings** > **Applications** > **Development** > **USB debugging**.
  - For Windows, use **Device Manager** to install a **USB driver for adb**. (<android-sdk>\extras\google\usb_driver)
Manual Application Installation

• Manual Installation using `adb`
  • App Package: `<workspace>\<Project Name>\bin\<Project Name>.apk`
  • Need `adb` tool located at `<sdk>/platform-tools/`
  • Modify Device settings as in “Running on a Device”
  • Connect the device to your machine through USB
  • In Command Prompt, type: `adb [-d | -e] install <path>/<file>.apk`

• Manual Uninstall
  • On device, go to Applications Menu -> Settings->Manage Applications.
  • Select the application which you want to Uninstall, and click on the ‘Uninstall’ button.
Debugging

The main components that comprise Android debugging environment are:

- **Device or Android Virtual Device**

- **Delvik Debug Monitoring Service (DDMS)**: graphical program that can capture screenshots, gather thread and stack information, spoof incoming calls and SMS messages, and other features.

- **The Android Debug Bridge (ADB)**: provides various device management capabilities.

- **JDWP debugger**: allows you to step through code, view variable values, and pause execution of an application. A JDWP-compliant debugger is already included in Eclipse.
DDMS  (Window > Open Perspective > Other... > DDMS)

Debug (Window > Open Perspective > Debug)
Putting Application on Market

• Signing Your Application with Eclipse
  • To distribute your application, however, you must build your application in release mode and sign the .apk file with your own private key.
  • right-click the project in the Package Explorer and select Android Tools > Export Unsigned Application Package. Then specify the file location for the unsigned .apk.

• Versioning Your Applications
  • android:versionCode, android:versionName
  • android:minSdkVersion, android:targetSdkVersion, android:maxSdkVersion

• Preparing to Publish
  • Check the following checklist: http://developer.android.com/guide/publishing/preparing.html

• Publishing on Android Market
  • Android Market developer registration: http://market.android.com/publish
Questions?
Assignment #1: Warming UP

• Due Fri Aug 31st, 11:59pm
• Select three high-quality apps from the iPhone or Android Market that are related to one of the following application domains:
  • Increase driving safety
  • Traffic monitoring
  • Enhance education experience
  • Monitor/support personal health
  • Monitor/save energy consumption
  • Support smart environments
• Pick applications that are innovative in some way (e.g., use new phone functionality, easy-to-use, popular, or take an approach that is very different from competitor apps.
• Critically evaluate the three apps
Assignment #1: Warming UP

• For each app, create a one-page pdf write-up that includes the following:
  • Your name and the name of the app evaluated.
  • A summary paragraph that identifies key features of the app.
  • A paragraph or bullet list describing what makes this app better than competitors. Why did you select it?
  • A list of positive characteristics (e.g. high-quality graphics, fun, indispensable tool). Try to be as precise as possible about what makes it good. For example, don’t just say it is addictive; try to explain why it is addictive. Don’t just say that it “looks professional.” Explain what makes it look that way.
  • A list of negative characteristics (e.g. force close, slow, confusing menu titles). What could be done better? Be detailed
  • A paragraph identifying the target audience for the app. Who might use this?
  • A paragraph discussing what additional features and functionalities you would add to enhance it.

• Few samples will be selected “randomly” to be presented and discussed in the next class (be prepared).
Assignment #1: Warming UP

• Send me an ordered list of only 3 reports/articles/papers you mostly liked about phone applications/services/features!
  
  • Again use instincts/guts
  • Your presentation paper could be among this list
  • Where to search:
    • Internet
    • Conferences: ACM PhoneSense, Mobicom, MobiSys, HotMobile, IEEE SECON
    • Magazines

• Submission Format:
  • Send all pages in a single PDF file
  • Have a cover page with your name and your email
  • Send the file as an attachment to me
  • Make the subject line of the email: cs495_assignment_1
Recommended Assignment

• Refresh your Java programming
  • http://docs.oracle.com/javase/tutorial/java/
    • http://docs.oracle.com/javase/tutorial/java/javaOO/index.html

• Have your development environment setup

• Programming: Hello World