## today

- What is Android?
- Getting Started
- The Emulator
- Hello World
- ADB
- Text to Speech
- Other APIs (camera, bitmap, etc)
- Other: Signing Apps, SVN
- Discussion and Questions

## introduction to android

- mobile operating system running on the Linux kernel
- Java language
- Google-developed Java libraries
- Open source

## introduction to android

- Android applications don't have a single entry point for everything in the application (no main() function, for example)
- Rather, they have essential components that the system can instantiate and run as needed.
- A task is a stack of activities

## components

- Activities (activated by Intents)
  - Presents a visual user interface for one focused endeavor the user can undertake
- Services
  - No visual UI but runs in background for an indefinite period of time
- Broadcast Receivers
  - Receives and reacts to broadcast announcements
- Content Providers
  - Makes a specific set of an application's data available to other applications

## sensors

- Compass
- Accelerometer
- Camera
- GPS
- Microphone

## manifest file

- Every application must have an AndroidManifest.xml file
- Where components are declared
- Set capabilities and permissions
- Include libraries
- Name Java package (unique identifier)

## what you'll need

- Supported Operating Systems
  - Windows XP or Vista
  - Mac OS 10.4.8 or later
  - Linux
- Hardware Requirements
  - > 900 MB

## development environments

- Eclipse IDE
  - Eclipse 3.4 (Ganymede) or 3.5 (Galileo)
    - Recommended Packages: Eclipse IDE for Java EE Developers, Eclipse IDE for Java Developers, Eclipse for RCP/Plugin-Developers, or Eclipse Classic (3.5.1+)
    - Eclipse JDT Plugin (incl. in most Eclipse IDE packages)
  - JDK 5 or JDK 6 (JRE not enough)
  - Android Development Tools Plugin
  - Not compatible with Gnu Compiler for Java (gcj)

## development environments ii

- Apache Ant
  - 1.6.5+ for Linux, Mac; 1.7+ Windows
  - JDK 5 or JDK 6

## using lab computers

- install adt plugin
- set android preferences sdk path
- create android virtual device
- creating own run config

## native development

- Way to build performance-critical portions of your apps in native code
- Android 1.6 NDK, Release 1
  - Must already have Android SDK
- Android runs in Davlik VM. This allows development in C, C++
- Increases app complexity, so make sure you need it!
  - Self-contained, CPU-contained operations
  - e.g., signal processing, physics simulations, vision

## get started tutorial

- You now have the Android SDK
- If using Eclipse, make sure you have the ADT plugin (Android Development Tools)
- Add Android platforms to your SDK (1.6, 2.0)
  - so you can build your app on lowest version you want to support, and test on higher versions
- Get the latest documentation
- SDK Add-Ons

## emulator

- Virtual mobile device
- Supports Android Virtual Device (AVD) configurations
- Debug capabilities
- Window > Android SDK and AVD Manager

## hello world

- Show Example
- Note: Project 'Tester' is missing required source folder: 'gen'  (restart-- or refresh)

## apache ant

android create project \
--target <target_ID> \
--name <your_project_name> \
--path /path/to/your/project \
--activity <your_activity_name> \
--package <your_package_namespace>

## use adb with command line

- with real device
- adb devices
- adb logcat
  - collecting and viewing system debug output
  - can be filtered
- adb install

## android.speech.tts

- initialize
  - private TTS myTts;
- add to last line of onCreate()
  - myTts = new TTS(this, ttsInitListener, true);
- after onCreate()
  - private TTS.InitListener ttsInitListener = new TTS.InitListener() {public void onInit(int version) {myTts.speak("Hello world", 0, null);}};
- Eyes-free
  - http://code.google.com/p/eyes-free/

## camera api

- Remember to set camera permissions!
- After picture taken- need callback
  - public void onPictureTaken(byte[] data, Camera camera)
  - get bitmap from byte[] data (needs decoding)
  - then access pixels (be aware of pixel format!)
  - finally access R,G,B values of pixels
  - do some processing

## signing applications

- Debug mode, done automatically
- When your application is ready for release, you must compile in release mode and then sign the .apk with your private key

## svn

- http://subclipse.tigris.org/
- Subversion within Eclipse IDE
- You can use your own method but be aware of versioning and changes!

## android developer challenge

- Android Developer Challenge 2 (over)
- Real-world users help review and score applications and the overall winner will take away $250,000
- Probably next deadline August 2010

## homework 1

- simple color application
- due Tuesday January 12
- send in .apk file and class files