Development for Android Devices
Introduction & Motivation

Alexander K. Seewald

SEEWALD SOLUTIONS

What makes smartphones so compelling?

- (Almost) Always on, instantly available
- (Computationally) Ever more powerful
- Localization: Knows where it is
- Sensors: Knowns how it is held & its environment. Knows wearer movement patterns (walking, running, up/down stairs)
- Mobile internet access anytime, anywhere
- Does *everything*: email, internet, calendar, contacts, remote administration, staying fit, entertainment, ... and telephony.

A Short Timeline of Smartphones

2000: Ericsson R380 Smartphone with touchscreen, marketed as smartphone, but very limited capability

2001: Palm Kyocera 6035: Combination of PDA and mobile phone with limited web browsing

2002: First Blackberry with voice, data, browser, messaging and organizer applications - first true smartphone

2004: HP iPaq h6315: Combination of successful PDA HP iPaq 2215 with cellular capability

2007: Nokia N8: First smartphone with stylus-free capacitive touchscreen & multi-touch capability

Original iPhone w/ multi-touch, only 3rd party web-apps

2008: HTC Dream, first Android (1.0) smartphone

Apple App Store starts *Jul. 2008*

parallel to 2nd Gen. iPhone w/ 3G support

Google Android Market starts *Oct. 2008*

Why develop for Android devices? (1)

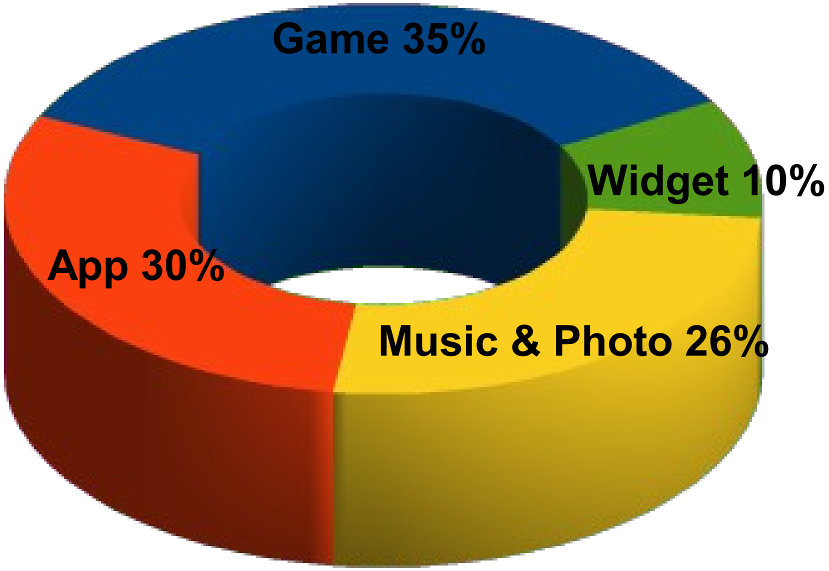
- Large user base - 400 million units sold @ 06/2012
  [iOS: 365 million]
- almost 2,000 device types - choose the perfect phone!
  [vs. iOS: one size fits all]
- 15 billion apps sold @ 05/2012
  [iOS: 25 billion]
- (Computationally) very powerful smartphones
- Very easy app implementation (Language: Java)
  [iOS: objectiveC, C/C++]

Why develop for Android devices? (2)

- Distribute apps world-wide
- One-time fee for developers, any platform
  [iOS: Pay three times as much every year]
- Correct VAT (MWST) processing and reporting
  [iOS: Apple does not care about European taxes]
- SDK works on any platform - Windows, Linux, MacOS
  [iOS: Buy physical Mac or illegally build MacOS VMware]
- New apps or updates you create are available in <15min
  [iOS: Wait several weeks & possibly get rejected]

Top Android Paid Apps (1)

App lifetime(LT) Turnover
= cost * approx.downloads
(@ 05/2012)



Name	Type	AppLT Turnover (€)
DocumentsToGo Full Version Key	App	8,692,500
Draw Something OMGPOP	Game	8,310,000
Beautiful Widgets	Widget	7,140,000
SwiftKey X	App	6,720,000
Camera ZOOM FX	Music & Photo	5,970,000
Minecraft - Pocket Edition	Game	4,680,000
Papier Kamera	Music & Photo	4,470,000
Poweramp Full Version Unlocker	Music & Photo	2,992,500
SoundHound	Music & Photo	2,992,500
Shazam Encore	Music & Photo	2,992,500
Fruit Ninja	Game	2,790,000
Root Explorer (File Manager)	App	2,692,500
Where's My Water	Game	2,280,000
Doodle Jump	Game	2,250,000
Cut the Rope	Game	2,040,000
Titanium Backup PRO Key * root	App	1,497,000
ROM Manager (Premium)	App	1,497,000
Grand Theft Auto III	Game	1,434,000
Star Chart	Game	1,047,000
Endomondo Sports Tracker PRO	App	858,000
Tapatalk Forum App	App	747,000
Osmos HD	Game	747,000
HD Widgets (3,0 Beta)	Widget	597,000
Asphalt 6: Adrenaline	Game	592,500
Smart Tools - Werkzeugkasten	App	570,000
TuneIn Radio Pro	Music & Photo	525,000
Flick Golf!	Game	237,000
Cut the Rope: Experiments	Game	228,000
Angry Birds Space Premium	Game	225,000
NOVA 3 - Near Orbit,,	Game	164,700
Rebuild	Game	57,750

© 2012

Top Android Paid Apps (2)

Caveats

- Fierce Competition: 450,000 apps, 100,000 active developers
- Device Fragmentation
  Hard to ensure good user experience for 2000+ devices!
- Growth in apps bigger than growth in revenue.
  Revenue per app is actually dropping at present!
- Popularity rules! Top lists are frequented often.
  Just uploading won't work. Need marketing as well!

Which apps do I use regularily? (1)

HTC Legend (2010)

- EMail, Calendar, Contacts, SMS (all preinstalled)
  The "killer" apps ;-)

- ConnectBot - SSH client to administer servers everywhere
- Qando (Vienna public transport) & Google Maps to offset my bad sense of orientation
- Mein Einkaufszettel (light weight German app)
- Barcode scanner (to test QR codes)
- World Time (a must for international collaborations!)
- Colorblindness Sim/Correction (a must for colorblind people!)
- Camera (for snapshots)
- Bubble (air level / "Wasserwaage")
- The Whip (for lazy students)

Which apps do I use regularily? (2)

Lenovo Thinkpad Tablet (2011)

- ConnectBot - SSH client
- Colorblindness Sim/Correction
- Writepad stylus (Note app w/o handwriting recognition)
- TabletPresenter (self-written app w/ laserpointer emulation and tiled scaling of very large images, not on market)
- QuickSlides (basic presentation app)
- MXPlayer (the best Android video player on the market)
- JJComics (comic & ebook reader)

Which apps do you use regularily?

iPhone: 4   Android: 3(1)   Neither: 0

Class of 11/2012 DUK

- EMail, Kontakt, SMS... 4
- WhatsApp
- Online Banking
- Spiele: TurboKids (droidGames), Monopoly, Bad Piggies, Fifa am iPad (steuern mit iPhone)
- Taschenlampe 3
- Google Drive
- Facebook App 3
- DerStandard
- SoundHound (MusicID fingerprinting)
- IMDB
- Unified Remote
- Evernote
- Wetter-Widget

Which apps did you pay for?

Myself

• Writepad	0.69 €
• Labyrinth (1,000 levels)	2.39 €
• Contact Lookup Pro	1.83 €

Class of 11/2012 DUK

- Keine 3
- Etwas 1
- Sehr viele 2 (TapTalk Plus, 20 EUR/Monat)

Android vs. iPhone: mixed...

Android / Google Play

- ~ 60% of apps available for free
- A free app cannot be made paid without losing all downloads (thus the importance of using in-app billing from the start...)
- Users are less likely to pay for apps
- May be one reason for popularity of Android!

iOS / Apple App Store

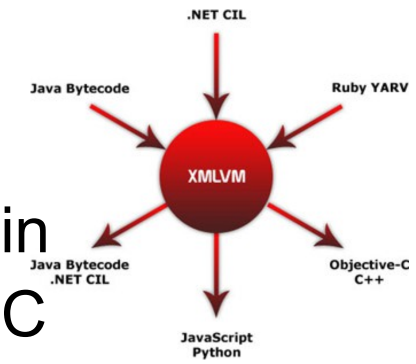
- ~ 30% of apps available for free
- Pricing change at a click, from free to paid and vice versa
- Users are more likely to pay for apps
- May be because they have no choice!

... so it makes sense to do both!

Options for cross-development

- **Just use web-applications!**
  - Most features supported (incl. multi-touch), interfaces vary
  - Easy to support other platforms (BlackBerry, Kindle Fire, ..)
  - New iOS6: You can finally upload files from the iPhone!
  - BUT not feasible for real-time image processing...
- **Proprietary solutions using non-Android toolchain**
  - Appcelerator's Titanium, Rhodes, PhoneGap
  - Program in Javascript/Ruby/Python. Hmm.... webapps? ;-)
  - Vendor lock-in is a real problem
- **Open source solutions using Android toolchain** (Java bytecode to iPhone native app)
  - XMLVM for pure Java apps (no NDK) (this is what we used to port Dancing Guide to iPhone)

XMLVM



Workflow

- Develop your Android app normally
- Compile to Java bytecode using Android toolchain
- XMLVM transforms Java bytecode into objectiveC
- Java classes are (seldom) reimplemented or (often) mapped to existing iOS classes using simple stubs; most is already there (e.g. had to implement audio recording into memory buffer - amazing how similar both platforms are at this level!)
- Output is the objC source code for a native app that needs to be compiled using the normal iPhone toolchain on MacOS
- Need to be well versed in code generator / XML transformation schemes to make this work out of the box. However, this only needs to be done *once*!

Result: One (Java)-Codebase for both Android and iOS!

http://xmlvm.org

App Development 101

Planning for revenue

- Free app & in-app billing (freemium)

  – slightly more complex code, only from 2.3.3 upwards

  + very convenient for user, easy to extend and test

  Needs BILLING permission - cannot be added after 1st upload
- Free app & separate paid app (free/paid)

  – two apps must be managed, different ratings & feedback

  + works for old API levels (<2.3.3)
- Free app (ad-supported)

  – get money only when user *clicks* on an ad...

  + good way to monetize app w/ many downloads (>1mill.)

Never start with just a paid app!

Free apps can never be made paid again!

Anecdotal: Freemium has ~3x revenue of Free/Paid

App Development 101

Planning for... something else

- Building apps for fame instead of fortune (example later)
- Getting more people on your website
  (~ 50% of our website traffic comes from our apps)
- Guerilla Marketing: Build apps that make fun of competition, or just inform your customers for free (e.g. Apo-App, Quando)
- Show off your skills as app developer to get hired / new orders
- Demo apps: test code & get feedback on performance/bugs
- Teach Android development to students ;-)

Standard capabilities of Android Smartphones (1)

- Multi-touch screen

- Measuring gravity & acceleration, magnetic field (compass)

- GSM/WLAN/GPS localization at no cost (+ Google Maps)

- WLAN
- Bluetooth
- NFC

- Microphone
- Speaker

- Camera (back-facing, often also second front-facing)

Standard capabilities of Android Smartphones (2)

- Calendar, Contacts, EMail, ... databases are all integrated as part of the system. It is possible to write new apps for all these functions.
- Write new keyboards / input methodologies (10-finger MT?)
- Write new screen locking apps
- It is even possible to write own dialers and check on numbers which are dialed or received (e.g. to block telemarketers - CIA Anruferkennung by Addafix)

Standard capabilities of Android Smartphones (3)

- Internal sensors are accessible using simple APIs
- Simple and robust localization using available systems
- The Google Maps API integrates global mapping into your app
- Show website or HTML content using WebView
- Many predefined dialogs: Progress, Alert, yes/no/cancel, text,..
- Easy real-time 3D via integrated OpenGL

Standard capabilities of Android Smartphones (4)

- Almost complete Java library. It is very easy to port almost any Java-based code to Android.
- Using native code in C/C++ via Android NDK - moderately difficult to port arbitrary C/C++ libraries and call from Java (JNI is still a bottleneck in some cases)
- Alpha Blending allows to add content in front of a camera preview for Augmented Reality apps. In parallel, preview images can be received via callback and processed (still not easy to completely change image as we will see later)Internal sensors are accessible using simple APIs

Standard capabilities of Android Smartphones (5)

Caveats

- A few capabilities need system/signature permissions, only possible for preinstalled apps. Will get strange error message..
  - Direct screen framebuffer access via SurfaceFlinger
  - Updating phone firmware
  - Backup all apps and settings

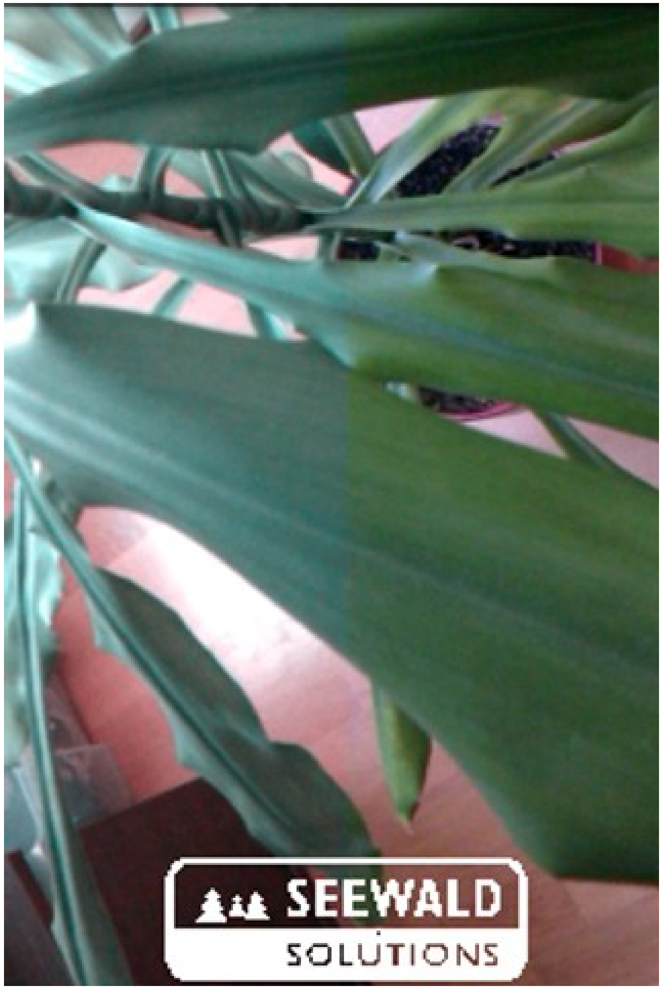
  These can only be done with a rooted phone. Some phones are temporarily rootable (until next reboot) using special apps.

  However you cannot rely on this.

- People may be loath to install apps that can e.g. monitor all keystrokes or know the screen lock combination (at least when they also have permission to access the Internet ;-)

Let's show some apps! (1)

- Colorblindness Simulation/Correction



Let's show some apps! (1)

- Colorblindness Simulation/Correction

Why? I am red/green colorblind and always found it cumbersome to explain this, so simulation mode came first. The correction mode was almost an afterthought but boosted app downloads significantly.

Lessons learned?

- The initial version had a bug which was fixed too late, yielding a lot of 1* ratings. It took almost a year to get over this.
- Changing description text to include synonyms for colorblindness such as daltonize, daltonization, ... was very helpful. In fact CBS is still the most popular free app with correction mode and quite high up in the rankings.

Let's show some apps! (2)

- Best Moves - Dancing Guide

Let's show some apps! (2)

- Best Moves - Dancing Guide

Why? I always found it hard to identify dance music; and for live music and classical music, fingerprinting approaches are not working. So there was an opportunity to do this.

Lessons learned?

- Initially I put in only a paid version. After six months there were few downloads... After making it free, there were thousands of downloads within a few days. I should have made a free version from the beginning.

Let's show some apps! (3)

- Wooden Easter Ratchet

Let's show some apps! (3)

- Wooden Easter Ratchet

Why? Because I could. Also because it is fun!

Lessons learned?

- Rotation works stable only along two axes, but confusing. One axis is enough. Some tablets rotate on the wrong axis...
- Some phones: Camera previews rotated 90°
  Only fixable from 2.3.3 upwards (display orientation available)
- All phones: OpenGL Overlay is incorrectly restored after resume in portrait mode - still not fixed in Android 4!

Let's show some apps! (4)

- Money Maker (~ Markerless Augmented Reality Demo)

Let's show some apps! (4)

- Money Maker (~ Markerless Augmented Reality Demo)

Why? The initial idea was to make an augmented reality system to remove political advertising. As a step towards this, I wanted to build a demo that would use an object that anyone has with them (rather than having to print it out), and the money doubling was just a funny story around that.

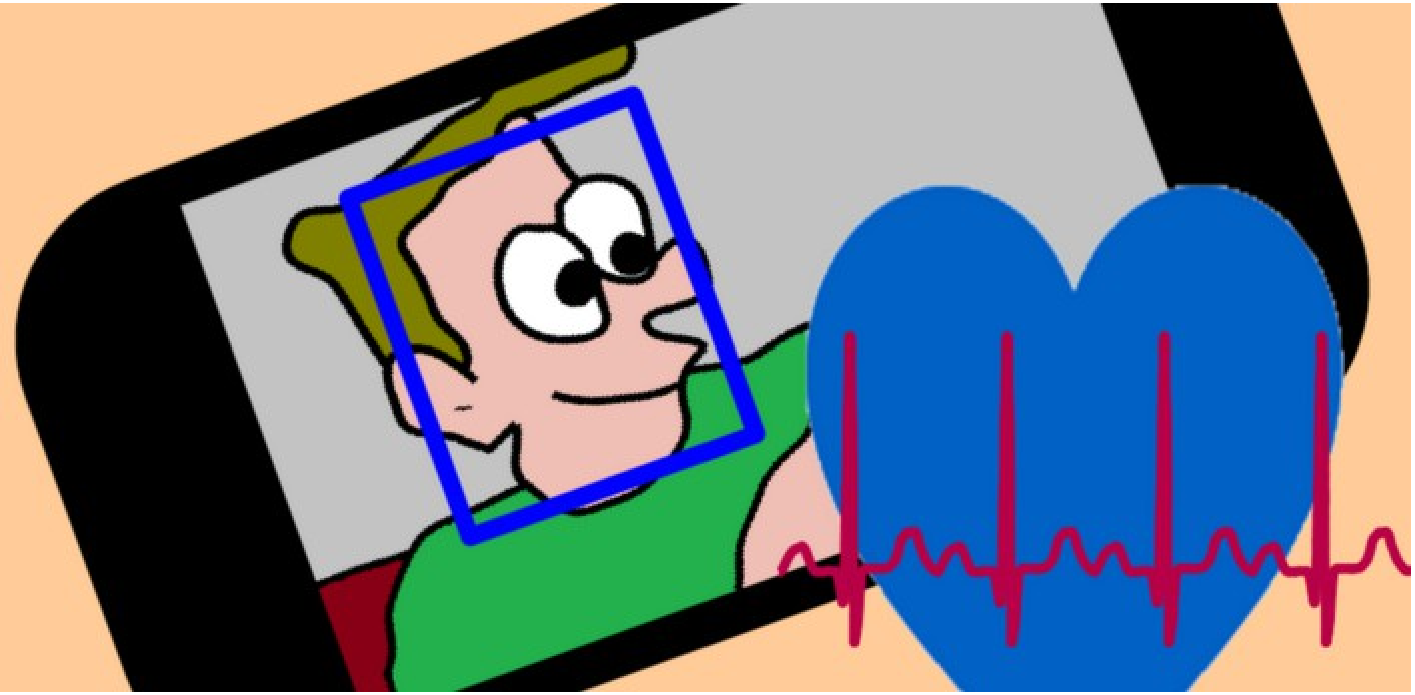
Lessons learned?

- The used system works well but a significant amount of time needs to be invested into optimizing the recognition for each image (1-2 weeks)

Next time when there are elections in Austria, look out for **WahlWerbungWeg** *by Seewald Solutions. ;-)*

Let's show some apps which have not been done by us! (5)

- MedCam - determine heart rate from face blood flow changes
  - Done by MIT on iOS recently but trivial to reimplement...
  - First such app on Android, only 2,000 downloads so far
  - *Desperately needs help in app-design & graphics!*



Which apps do you want to do / would like to see?

Class of 11/2012 DUK

- Spiele
- Parksünder-Empfehlungsapp
- Berggipfel-Identifizierung
- Digitale Wanderstempel
- Innovative NFC-Anwendung

Development for Android Devices
App Development 101

Features, Permissions & Screen Sizes

Alexander K. Seewald

SEEWALD SOLUTIONS

App idea kill list

- No me-too apps unless **significantly** better than competition
  Don't try to copy something which already works!
- Must have sufficiently large number of interested users
  If you need an app just for yourself, make it free.
- Customers must be willing to pay for the app's functionality
  Must not already be available for free or easy to copy.
- Keep development costs low! Think about easiest way to implement basic concept, and show prototype to customers.

What makes a successful app?

Based on a comprehensive analysis of four months tracking of new Android apps, the following patterns emerged. Percentages are successful apps as defined by more than 50,000 downloads at the end of the observed period. Free apps were used for this analysis since the sample of paid apps was too small.

- Upload your app on a Friday or Monday, never on a Thursday

  New apps are shown in a special area of the market, and before the weekend, more people are looking there.

Mon	3.13%
Tue	2.31%
Wed	2.51%
Thu	0.84%
Fri	3.45%
Sat	3.17%
Sun	2.46%

What makes a successful app?

- Get more than 100 downloads within 2h of first upload

Downloads	Prop. Successful
<50	2.27%
50-100	13.04%
100-500	25.00%

- Get a rating of at least 4.5 within 2h of first upload

Ratings Interval	Prop. Successful
[0]	0.71%
(0,4)	5.71%
[4,4.5)	14.71%
[4.5,5)	26.67%
[5]	7.04%

© 2012

What makes a successful app?

- Get more than six people to rate your app within 2h of 1st upl.

RatingsCount Interval	Prop. Successful
[0]	0.71%
[1]	4.05%
[2]	11.50%
[3,6)	16.67%
[6,inf)	50.00%

- Fill out recent changes, contact website, poss. contact phone

	Yes	No
Contact Website	2.60%	2.21%
Contact Phone	2.62%	2.49%
Recent Changes	3.26%	2.32%

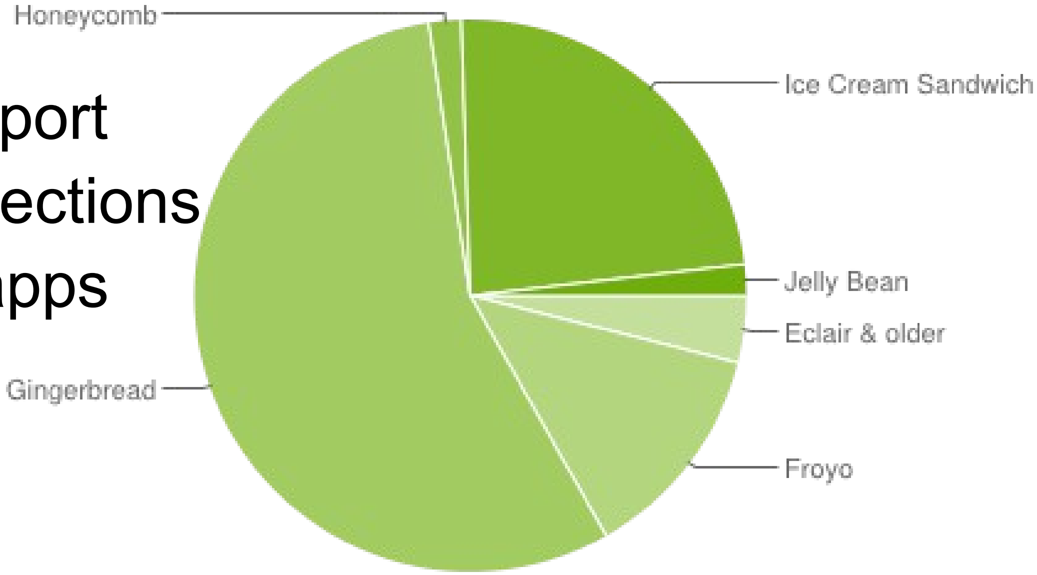
App Development 101

Choose minimum API level to support

(API level = Android version)

- Most common: Gingerbread (2.3.3) = 58.8%
- Older versions (<2.3.3) still at 16.8% (Froyo (2.2) = 12.9%)
- Don't target Honeycomb (3.*) - very buggy! few users anyway...
- Newest version: Ice Cream Sandwich (4.0) = 25% share
  Fixed most bugs of earlier version, quite good & stable

- It's possible but tricky to support multiple API levels using reflections
- Worse alternative: Multiple apps with API level market filter



http://developer.android.com/about/dashboards/index.html

© 2012

SEEWALD SOLUTIONS

alex@seewald.at
www.seewald.at

App Development 101

Which hardware features are *absolutely necessary* for the app?

(used for Market Filtering, depends on API level)

- Microphone
- Camera (Front, Autofocus)
- Touchscreen (gesture only, >2/>5 touches in parallel)
- Sensors (Accelerometer, Gyroscope, Compass, Light, Temperature,...)
- GPS
- Wifi
- Bluetooth

...

Features available on specific device (needs to be connected)

```
adb shell pm list features
```

Full feature list for Lenovo Thinkpad Tablet (Android 4.0.3)

Features	
feature:android.hardware.bluetooth	feature:android.hardware.touchscreen
feature:android.hardware.camera	feature:android.hardware.touchscreen.multitouch
feature:android.hardware.camera.autofocus	feature:android.hardware.touchscreen.multitouch.distinct
feature:android.hardware.camera.front	feature:android.hardware.touchscreen.multitouch.jazzhand
feature:android.hardware.faketouch	feature:android.hardware.usb.accessory
feature:android.hardware.location	feature:android.hardware.usb.host
feature:android.hardware.location.gps	feature:android.hardware.wifi
feature:android.hardware.location.network	feature:android.software.live_wallpaper
feature:android.hardware.microphone	feature:android.software.sip
feature:android.hardware.screen.landscape	feature:android.software.sip.voip
feature:android.hardware.screen.portrait	feature:com.cisco.anyconnect.permissions.patch.lenovo
feature:android.hardware.sensor.accelerometer	feature:reqGlEsVersion=0x20000
feature:android.hardware.sensor.compass	feature:android.hardware.sensor.light

http://developer.android.com/guide/topics/manifest/uses-feature-element.html

App Development 101

Know your hardware multitouch capability!

- `android.hardware.touchscreen`
  - no multitouch capability implied, can process single touches
- `android.hardware.touchscreen.multitouch`
  - two-point multitouch capability (pinch, zoom) but no independent tracking of touch points
- `android.hardware.touchscreen.multitouch.distinct`
  - at least two independently tracked touch points
- `android.hardware.touchscreen.multitouch.jazzhand`
  - at least five independently tracked touch points

Some Android tablets can independently track ten touch points...

© 2012

Camera

- `android.hardware.camera`
  - has one (almost always back-facing) camera
- `android.hardware.camera.autofocus`
  - has at least one camera with autofocus. Missing: only fixed-focus
- `android.hardware.camera.front`
  - has a front facing camera (facing the user, front side of phone)

App Development 101

Localization

- `android.hardware.location`
  - has a way of localizing itself (unspecified)
- `android.hardware.location.network`
  - can locate using GSM network (very inaccurate)
- `android.hardware.location.gps`
  - can locate using GPS (accurate outside, unusable inside)
- `android.hardware.wifi`
  - Every phone with Wifi can roughly localize with it using Google's WLAN access point fingerprinting system. It is currently more accurate than GSM and less accurate than GPS.

App Development 101

Others (1)

- `android.hardware.bluetooth`
  - Bluetooth support (not yet BT 4.0 like the iPhone 4S/5, sadly)

- `android.hardware.microphone`
  - has a microphone (almost always the case ;-)

- `android.hardware.screen.landscape`
- `android.hardware.screen.portrait`
  - supports portrait / landscape mode (almost always, except Google TV which is fixed in one mode for obvious reasons)

App Development 101

Others (2)

- `android.hardware.usb.accessory`
  - can be attached as USB peripheral device (almost always)
- `android.hardware.usb.host`
  - can be attached as USB host (= can use USB peripherals and USB sticks on its own; seldom)
- `android.hardware.nfc`
  - supports Near-Field-Communication (NFC)
- `android.software.live_wallpaper`
  - support for live wallpapers (android.service.wallpaper)
- `android.software.sip`
  - supports Session-Initiation-Protocol (SIP)
- `android.software.sip.voip`
  - supports SIP and Voice-Over-IP (VOIP)

(you will be able to install Skype anyway ;-)

App Development 101

Sensors (0)

Rather than using camera or microphone and complex processing, many applications work with simple internal sensors. Even augmented reality apps can be made using compass, accelerometer/gyroscope and a camera preview with overlay!

- `android.hardware.sensor.*`
  - corresponds to sensor of TYPE_*

If you app absolutely needs a sensor to work, put it into features. But better is to design a secondary interface and check for sensor presence during runtime, automatically switching to secondary interface (e.g. touchscreen)

App Development 101

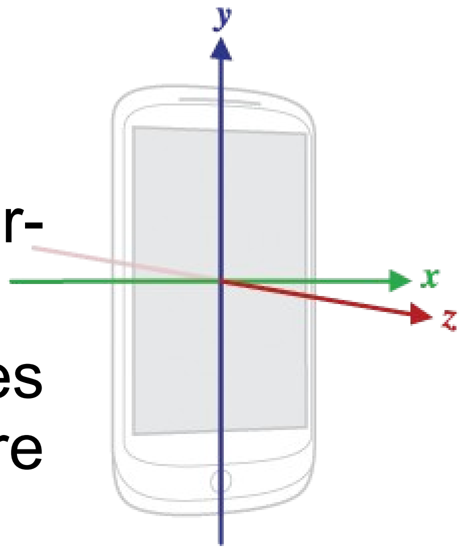
Sensors (1)

- TYPE_PROXIMITY: distance to ear, front side of the phone. Switches off screen when phone is held to ear. Ubiquituous.
- TYPE_LIGHT: measures light in lux, supposedly calibrated. Very uncommon.
- TYPE_AMBIENT_TEMPERATURE: measures temperature in degrees celsius. Never seen.
- TYPE_PRESSURE: measures atmospheric pressure. N.s.
- TYPE_RELATIVE_HUMIDITY: measure humidity. Never seen.

© 2012

App Development 101

Sensors (2)



- TYPE_ACCELEROMETER: measures acceleration via 3D vector. Very common.
- TYPE_GYROSCOPE: measures rotation angles along X/Y/Z axis. Not widely used since more expensive than accelerometer.

  Both can be used to determine position w.r.t. earth plane.
- TYPE_MAGNETIC_FIELD: measures magnetic field strength in X/Y/Z direction (= a 3D compass)

  Needed to measure rotation on a perpendicular axis vs. earth plane, is far less accurate than rotation around other axes.

These are low-level sensors which should not be used. From Gingerbread (2.3.3) we can rely on the following virtual sensors.

© 2012

App Development 101

Sensors (3)

These sensors are computed from either accelerometer or gyroscope (or hopefully both) plus magnetic field sensor.

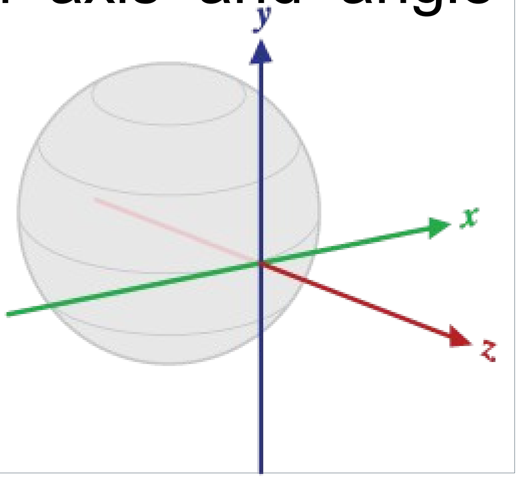
- TYPE_GRAVITY: returns a vector indicating 3D direction and magnitude of gravitiy (i.e. points exactly down)
- TYPE_LINEAR_ACCELERATION: returns a vector indicating direction and magnitude of linear acceleration.

  TYPE_ACCELERATION returns sum of these two sensors

- TYPE_ROTATION_VECTOR: returns rotation axis and angle versus a reference coordinate system:

  Z = straight up, Y = magnetic north, X implied

  Uses either accelerometer or gyroscope plus magnetic field sensor. Most accurate using all three sensors.



© 2012

SEEWALD SOLUTIONS

alex@seewald.at
www.seewald.at

App Development 101

Which permissions are **absolutely necessary** for the app?

(used for Market Filtering, depends on API level)

- Bluetooth (admin or user)
- Camera (implies Autofocus)
- Location (coarse, fine - implies GPS, install own provider)
- Record audio
- Telephony (call, modify state, process outgoing calls, sms...)
- Wifi (access state, change state)
- Sensors (Accelerometer, Gyroscope, Compass, Light, Temperature,...)
- Write to external storage

...

Some permissions imply features! (see link on last page)

Permissions available on specific device (needs to be conn.)

```
adb shell pm list permissions -f
```

App Development 101

Full permissions list for Lenovo Thinkpad Tablet (Android 4.0.3)

permission:cisco.permission.NET_ADMIN
permission:android.intent.category.MASTER_CLEAR.permission.C2D_MESSAGE
permission:android.permission.ACCESS_ALL_DOWNLOADS
permission:android.permission.ACCESS_BLUETOOTH_SHARE
permission:android.permission.ACCESS_CACHE_FILESYSTEM
permission:android.permission.ACCESS_CHECKIN_PROPERTIES
permission:android.permission.ACCESS_DOWNLOAD_MANAGER
permission:android.permission.ACCESS_DOWNLOAD_MANAGER_ADVANCED
permission:android.permission.ACCESS_DRM
permission:android.permission.ACCESS_SURFACE_FLINGER
permission:android.permission.ALLOW_ANY_CODEC_FOR_PLAYBACK
permission:android.permission.BACKUP
permission:android.permission.BATTERY_STATS
permission:android.permission.BIND_DEVICE_ADMIN
permission:android.permission.BIND_INPUT_METHOD
permission:android.permission.BIND_PACKAGE_VERIFIER
permission:android.permission.BIND_REMOTEVIEWS
permission:android.permission.BIND_TEXT_SERVICE
permission:android.permission.BIND_VPN_SERVICE
permission:android.permission.BIND_WALLPAPER
permission:android.permission.BRICK
permission:android.permission.CALL_PRIVILEGED
permission:android.permission.CHANGE_COMPONENT_ENABLED_STATE
permission:android.permission.CLEAR_APP_USER_DATA
permission:android.permission.CONFIRM_FULL_BACKUP
permission:android.permission.CONTROL_LOCATION_UPDATES
permission:android.permission.COPY_PROTECTED_DATA
permission:android.permission.CRYPT_KEEPER
permission:android.permission.DELETE_CACHE_FILES
permission:android.permission.DELETE_PACKAGES
permission:android.permission.DEVICE_POWER
permission:android.permission.DOWNLOAD_CACHE_NON_PURGEABLE
permission:android.permission.FACTORY_TEST
permission:android.permission.FORCE_BACK
permission:android.permission.INJECT_EVENTS
permission:android.permission.INSTALL_DRM
permission:android.permission.INSTALL_LOCATION_PROVIDER
permission:android.permission.INSTALL_PACKAGES
permission:android.permission.INTERNAL_SYSTEM_WINDOW
permission:android.permission.MANAGE_APP_TOKENS
permission:android.permission.MANAGE_NETWORK_POLICY
permission:android.permission.MASTER_CLEAR
permission:android.permission.MODIFY_NETWORK_ACCOUNTING
permission:android.permission.MOVE_PACKAGE
permission:android.permission.NVIDIA_CPU_POWER
permission:android.permission.PACKAGE_USAGE_STATS
permission:android.permission.PACKAGE_VERIFICATION_AGENT
permission:android.permission.PERFORM_CDMA_PROVISIONING
permission:android.permission.READ_FRAME_BUFFER
permission:android.permission.READ_INPUT_STATE
permission:android.permission.READ_NETWORK_USAGE_HISTORY
permission:android.permission.REBOOT
permission:android.permission.SEND_DOWNLOAD_COMPLETED_INTENTS
permission:android.permission.SET_ACTIVITY_WATCHER
permission:android.permission.SET_ORIENTATION
permission:android.permission.SET_POINTER_SPEED
permission:android.permission.SET_TIME
permission:android.permission.SHUTDOWN
permission:android.permission.STATUS_BAR
permission:android.permission.STATUS_BAR_SERVICE
permission:android.permission.STOP_APP_SWITCHES
permission:android.permission.UPDATE_DEVICE_STATS
permission:android.permission.WRITE_GSERVICES
permission:android.permission.WRITE_SECURE_SETTINGS
permission:android.server.checkin.CHECKIN.permission.C2D_MESSAGE
permission:cisco.permission.NET_RAW
permission:cisco.permission.VPN
permission:com.android.browser.permission.PRELOAD
permission:com.android.email.permission.ACCESS_PROVIDER
permission:com.android.vending.billing.ADD_CREDIT_CARD
permission:com.android.vending.billing.BILLING_ACCOUNT_SERVICE
permission:com.android.vending.billing.IN_APP_NOTIFY.permission.C2D_MESSAGE
permission:com.android.vending.INTENT_VENDING_ONLY
permission:com.android.vending.permission.C2D_MESSAGE
permission:com.android.vending.permission.UPDATE_MARKET
permission:com.android.vending.TOS_ACKED
permission:com.google.android.apps.maps.permission.C2D_MESSAGE
permission:com.google.android.gm.permission.READ_ATTACHMENT_PREVIEW
permission:com.google.android.googleapps.permission.ACCESS_GOOGLE_PASSWORD
permission:com.google.android.googleapps.permission.GOOGLE_AUTH.doraemon
permission:com.google.android.googleapps.permission.GOOGLE_AUTH.geowiki
permission:com.google.android.googleapps.permission.GOOGLE_AUTH.goanna_mobile
permission:com.google.android.googleapps.permission.GOOGLE_AUTH.panoramio
permission:com.google.android.googleapps.permission.GOOGLE_AUTH.reader
permission:com.google.android.googleapps.permission.GOOGLE_MAIL_SWITCH
permission:com.google.android.gsf.subscribedfeeds.permission.C2D_MESSAGE
permission:com.google.android.partnersetup.permission.ACCESS_PROVIDER
permission:com.google.android.partnersetup.permission.UPDATE_CLIENT_ID
permission:com.google.android.providers.gsf.permission.WRITE_GSERVICES
permission:com.google.android.providers.settings.permission.READ_GSETTINGS
permission:com.google.android.providers.settings.permission.WRITE_GSETTINGS
permission:com.google.android.talk.permission.RECEIVE_XMPP
permission:com.google.android.voicesearch.AUDIO_FILE_ACCESS
permission:com.google.android.voicesearch.SHORTCUTS_ACCESS
permission:com.humanengines.vortexhd.ACCESS_PROVIDER
permission:com.humanengines.vortexhd.external.email.permission.ACCESS_PROVIDER
permission:com.lenovo.indigo.mailcalendar.permission.ACCESS_AGENT
permission:com.lenovo.indigo.mailcalendar.permission.ACCESS_PROVIDER
permission:com.lenovo.packageinstaller.SYSTEM_INSTALL
permission:com.mcafee.permission.VSM_READ_STATUS
permission:com.oovoo.permission.C2D_MESSAGE
permission:com.wsandroid.suite.permission.C2D_MESSAGE

http://developer.android.com/reference/android/Manifest.permission.html

App Development 101

Most important permissions (1)

When initiating a sensitive action without having the correct permission, your app will be killed with an appropriate error. Permissions imply necessary hardware features automatically.

- `android.permission.CAMERA`
  - allow Camera access (implies autofocus!)
- `android.permission.INTERNET`
  - allow Internet access
- `android.permission.WAKE_LOCK`
  - allow to keep device awake & screen on
- `android.permission.RECORD_AUDIO`
  - allow to record audio
- `android.permission.READ_EXTERNAL_STORAGE`
- `android.permission.WRITE_EXTERNAL_STORAGE`
  - allow to read/write to external storage (sdcard or USB stick)

App Development 101

Most important permissions (2)

- `android.permission.VIBRATE`
  - allow to activate vibrate mode (low-level control possible)
- `android.permission.READ_CONTACTS`
- `android.permission.WRITE_CONTACTS`
  - allow to read and write contact information
- `android.permission.SET_WALLPAPER`
  - allow to set wallpaper on home screen
- `android.permission.ACCESS_COARSE_LOCATION`
  - access GSM based localization data (~ 100m accuracy)
- `android.permission.ACCESS_FINE_LOCATION`
  - access GPS & WLAN-based localization data (~ 5-10m accuracy, might take some time to become available)

App Development 101

Most important permissions (3)

- `android.permission.GET_TASKS`
  - allow accessing information about running tasks
- `android.permission.CALL_PHONE`
- `android.permission.CALL_PRIVILEGED`
  - allow to initiate calls without user interaction
  - Privileged: also call emergency numbers w/o user interact.
- `android.permission.EXPAND_STATUS_BAR`
  - allow to expand and contract the status bar
- `com.android.vending.BILLING`
  - allow in-app billing. **MUST BE SET ON FIRST UPLOAD!**

App Development 101

Most important permissions (4)

- `android.permission.SEND_SMS`
- `android.permission.RECEIVE_SMS`
  - allow to send and receive SMS
- `android.permission.NFC`
  - allow to use Near-Field-Communcations (NFC)
- `android.permission.ACCESS_WIFI_STATE`
  - allow to access information on wifi networks
- `android.permission.BLUETOOTH`
  - allow to connect to already paired bluetooth devices
- `android.permission.BLUETOOTH_ADMIN`
  - allow to discover, pair & onnect to any bluetooth device

© 2012

App Development 101

Most important permissions (5)

- `android.permission.WRITE_SETTINGS`
- `android.permission.WRITE_SECURE_SETTINGS`
  - allow to write (secure) system settings

- `android.permission.READ_LOGS`
  - allow to read all system logs (needed for own crash report)

- `android.permission.SET_TIME`
  - allow to set time (sadly only for system apps)

App Development 101

Screen sizes

- Not a feature nor a permission: `<supports-screens ...>`
- Large variety of screen sizes (small, normal, large, xlarge) and densities (l/m/h/xhdpi ~ pixel density per inch) are possible
- Need to explicitly declare which sizes are supported!

  (otherwise app is run in compatibility mode and badly scaled)

- Density- & Screen size independence can be achieved...
  - with layouts using density-independent measures (e.g. dp)
  - defining different layout for screen sizes or densities
  - manually scaling bitmap resources
  - providing different bitmap resources for different densities
- Icecream Sandwich (4.0) offers new options to manage screen sizes and densities.

http://developer.android.com/guide/practices/screens_support.html

App Development 101

- Features, Permissions & Screen sizes determine how many Android devices will see your app in their Google Play store.

- For debug builds, market filtering via features is irrelevant. (insufficient permissions will still crash your app) Make sure you test if the app appears on your test phones after upload.

- App must run well on current and last year's Android phones. You can never have enough test phones.
- Fixing bugs as fast as possible is essential for good ratings. However, Google's crash report is not very informative. The ACRA framework is useful to deliver your own crash report. http://acra.ch/

Development for Android Devices
App Development 101

Layouts, Controls, and Lifecycles

Alexander K. Seewald

SEEWALD SOLUTIONS

App Development 101

Layout

- Simple XML format to specify layout of application
- Actual controls are embedded into *Layouts that take care of size and positioning.

```
<?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="@string/hello" />

</LinearLayout>
```

© 2012

Attributes common to all layouts

- `android:layout_width`
- `android:layout_height`
  - controls width & height of the layout object
  - Common value: fill_parent = as large as parent object (root = as large as screen); wrap_content = only as large as necessary after displaying children)
- `android:gravity`
  - placement of child objects when space remains
  - top, bottom, left, right, center, ...
- `android:id`
  - unique id for this layout (usually `@+id/`[unique name])
- `android:visibility`
  - gone (not visible & not part of layout), visible, invisible

© 2012

App Development 101

LinearLayout

- Display children in single column (`android:orientation="vertical"`) or row (default, `android:orientation="horizontal"`)

RelativeLayout

- children are placed in reference to their siblings
- `android:layout_below`
- `android:layout_above`
- `android:layout_toRightOf`
- `android:layout_toLeftOf`

....

- Might be useful for very complex layouts

App Development 101

FrameLayout

- puts children on top of each other
- necessary for multi-layer views (e.g. transparent buttons on top of camera preview or background images)

ScrollView

- Special kind of FrameLayout, scrolls contents (one child!)

TableLayout

- puts children in rows and columns ~ HTML <table>
- `<TableRow ..>`
  - defines rows. Children of rows will be display as columns
- Children outside of rows will be displayed over all columns.

App Development 101

Content from ListAdapter, cannot be statically assigned
Good for showing large amounts of data (e.g. lists, images)

ListView

- 1D scrollable list of children

GridView

- 2D scrollable list of children

Fill via ListAdapter, e.g....

- static: string-array via ArrayAdapter.createFromResource()
- dynamic: write your own adapter subclass

All Layouts can be arbitrarily combined like any other control.

© 2012

Controls

Android offers all usual controls: Textboxes, Listboxes, Text & Image-Buttons, Images... We will only explain the most common.

Attributes common to all controls (widgets)

- All common layout attributes
- `android:layout_weight`
  - higher weight implies more space for this control vs. its siblings (sensible values: 0, 1, 2)
- `android:background`
  - background color, shape or image (drawable)

© 2012

TextView

Non-Input Text field (can still be changed from code)

- `android:text`
  - initial text which is displayed
- `android:textSize`
  - size of text, should be density-independent (e.g. `20sp` = scaled pixel units)
- `android:textColor`
  - color of text, e.g. `#FFFFFFFF`, named color (white, yellow, ..) or ref. to color drawable (`@drawable/seso`)
- `android:typeface`
  - monospace, serif, ...
- `android:textStyle`
  - bold, italics, ...

Many other attributes, check documentation!

EditText

Input Text field. Same attributes as TextView.

Button

Text button. Same attributes as TextView plus...

- `android:label`
  - initial button text (don't use android:text !)

CheckBox

Special button that can be checked or unchecked.

RadioButton

Special button that implements mutual-exclusion.

- Must be grouped within RadioGroup
- Only one RadioButton within RadioGroup can be checked
- All options are visible at one glance

Spinner

- Basically a drop-down listbox
- Must be filled via SpinnerAdapter ~ ListAdapter

ToggleButton

- `android:textOn`
- `android:textOff`
  - shows different text if on or off, click to toggle
- otherwise just like CheckBox

ImageView

Showing a seldom changing image

- `android:src`
  - initial image (must be a drawable)
- `android:scaleType`
  - how is image scaled? (center, fitXY, ...)

ImageButton

Image button. Same attributes as ImageView.

- `android:src`
  - XML drawable w/ different images for button states, e.g.

```
<?xml version="1.0" encoding="utf-8"?>
 <selector xmlns:android="http://schemas.android.com/apk/res/android">
     <item android:state_pressed="true"
           android:drawable="@drawable/btn_news_sel" />
     <item android:state_focused="true"
           android:drawable="@drawable/btn_news_sel" />
     <item android:drawable="@drawable/btn_news_normal" />
</selector>
```

App Development 101

SurfaceView

For fast-moving images and camera previews

- Much faster rendering than ImageViews & normal views
- Can be drawn on via background threads (normal views throw an Exception when doing this, need RunOnUiThread())
- Not necessary to use main user interface thread for drawing

Tricky to have two SurfaceViews in one FrameLayout (e.g. camera preview and OpenGL alpha surface)

- first OpenGL alpha surface (with alpha channel & "holes")
- then camera preview

Even in Android 4.0 (very seldom) combined in wrong order!

Find the (non-fixable) bug in wooden easter rattle!

Understanding Android application directory structure (1)

- **Essential**

  AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.SeewaldSolutions.HelloWorld"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk android:minSdkVersion="10" />
    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <activity
            android:name=".HelloWorldActivity"
            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>
```

Put permissions, features & screen sizes here like this:

```
<uses-permission android:name="perm.." />

<uses-feature android:name="feature:.."
android:required=["true"|"false"] />

<supports-screens android:resizeable=
["true"|"false"] android:(small|normal|large|
xlarge)Screens=["true"|"false"] ... />
```

Understanding Android application directory structure (2)

- **Essential**

  src/ - contains all source code

  (at least the activity mentioned in AndroidManifest.xml)

```
└── com
    └── SeewaldSolutions
        └── HelloWorld
            └── HelloWorldActivity.java:
```

```
package com.SeewaldSolutions.HelloWorld;

import android.app.Activity;
import android.os.Bundle;

public class HelloWorldActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

Displays XML layout

Can also create everything with code.

Understanding Android application directory structure (3)

- **Essential**

  res/ - contains all string, image and xml resources. Resolutions not found will be interpolated.

```
├── drawable-hdpi / ..-ldpi / ..-mdpi / ..-xhdpi
|    └── ic_launcher.png
├── layout
|    └── main.xml:
```

App icon in all resolutions

36x36 (ldpi), 48x48 (mdpi)
72x72 (hdpi), 96x96 (xhdpi)

XML Layout w/ Textview

```
<?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="@string/hello" />

</LinearLayout>
    |
    └── menu
        └── menu.xml
```

XML menu resources (optional)
Load like this:

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
  super.onCreateOptionsMenu(menu);

  MenuInflater inflater = getMenuInflater();
  inflater.inflate(R.menu.menu, menu);

  return true;
}
```

© 2012

Understanding Android application directory structure (4)

- **Essential**

res/ - (continued)

```
└── values
    └── strings.xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="hello">Hello World, HelloWorldActivity!</string>
  <string name="app_name">HelloWorld</string>
</resources>
└── raw
```

XML string resources

/values = default language
/values-[languageCode] =
other languages(de,en,fr,..)

Must have same entries in all languages.

"Raw" resources, open with:

InputStream ins = getResources().openRawResource(R.raw.file);

Use resources in XML & code

```
<android:text="@string/hello" /><android:label="@string/app_name" />
<android:icon="@drawable/ic_launcher" />
String S = getString(R.string.hello); S = getString(R.string.app_name);
Drawable D = getResources().getDrawable(R.drawable.ic_launcher);
```

Automatically chooses the language for string resources, and resolution for drawable resources when running on device.

Understanding Android application directory structure (5)

- **Optional**

  assets/ - alternative way to store any resources (no res. ID)
- res/drawable-*/* (all except drawable-nodpi) is scaled on each viewing - small memory leak, unfixed, ~10x restart kills most apps. Drawable memory for resources restricted. Solved by using assets and loading resources ourselves.
- Cannot be used in XML, just in Code:

```
AssetManager as = getAssets(); InputStream i = as.open("test.txt");
Bitmap bitmap = BitmapFactory.decodeStream(
    new BufferedInputStream(assets.open("drawable/test.jpg"))
);
Drawable d = new BitmapDrawable(bitmap);
d.setBounds(0,0,d.getIntrinsicWidth(),d.getIntrinsicHeight());
```

- Need to release all asset-based bitmap drawables manually!

```
d.getBitmap().recycle(); d=null;
System.gc();
```

(optional)

© 2012

App Development 101

Understanding Android application directory structure (6)

- **Optional**

  project.properties - generated from AndroidManifest.xml, can be used to switch on obfuscation via proguard

  proguard-project.txt - proguard settings

  jni/ - contains C/C++ code (compiled using Android NDK), can be set up to automatically compile together with SDK toolchain

- **Internal, can be deleted (in case of compilation problems)**

  bin/ - final apk files and intermediate compilation

  gen/ - resource ids and other generated code

Toolchain (1)

- Android toolchain is based on the Javac build tool *Apache Ant*
- Simple command line interface (used by ADT plugin)

  `ant clean, ant build, ant install ...`
- Java source plus generated code (resource IDs, external libraries, etc..) is compiled into Java bytecode and put into a simple jar (~ zip) file.
- Other resources are added right after the code. Direct readonly access just by computing the right pointer (done internally). This also allows to put apps into read-only memory.

App Development 101

Toolchain (2)

- The whole package (apk file) is public-key signed by one of...
  - Debug key: can be put on any device if install of non-market applications is allowed (otherwise fails). No feature checking vs. device! Can install any app anywhere.

    To install this app on a device, you can...
    - send it via EMail as attachment
    - put it on a sdcard or on the device, start with file manager (e.g. Linda Manager)
    - switch on USB-Debugging and use SDK's adb tool (most convenient): `adb install [-r] bin/debug-app.apk`
  - Release key: available only to developers, part of Market account. Can upload apk to the Market via webinterface. Will be available in Market after short delay. Market filtering in place - only devices with the right features will see app!

Toolchain (3)

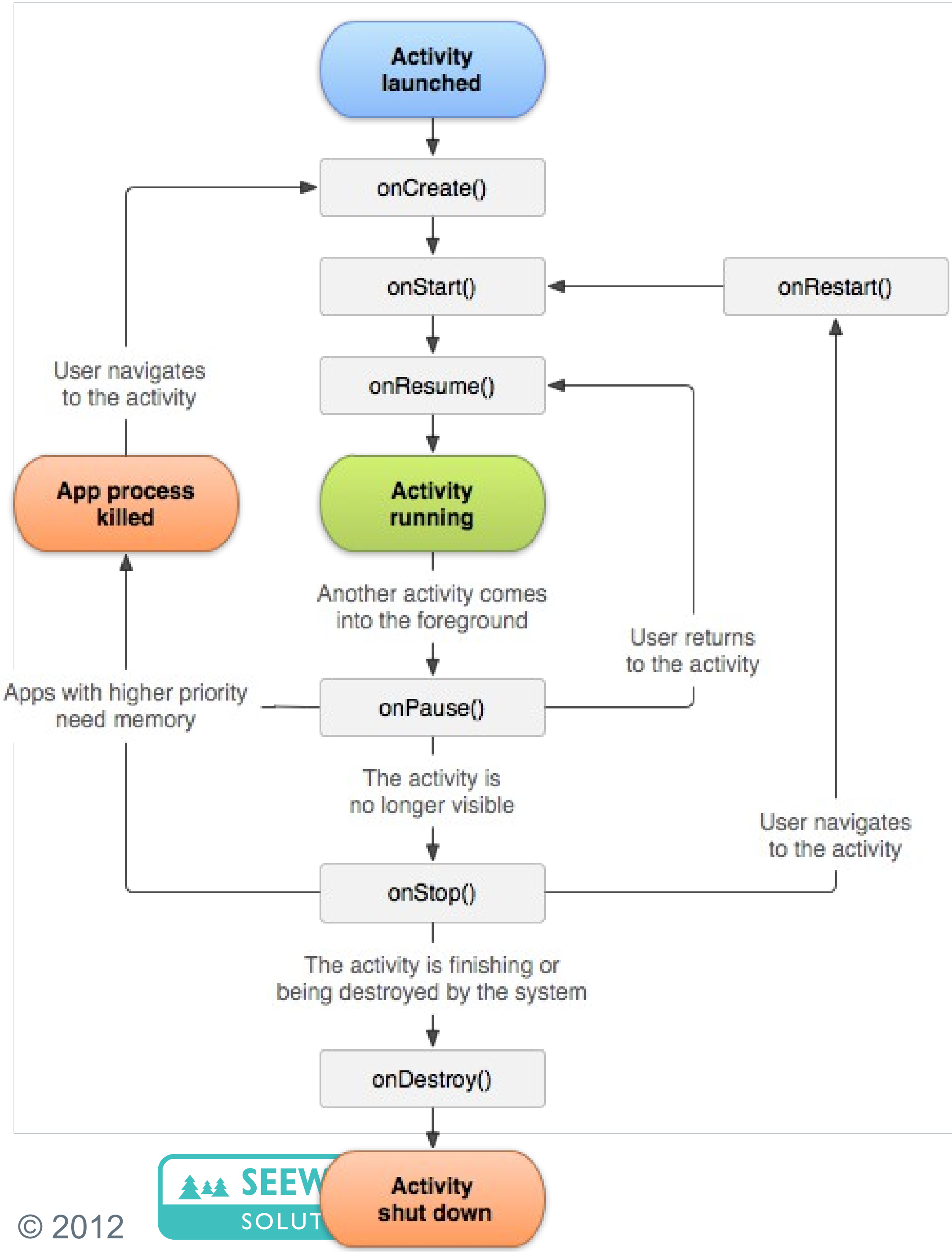
- On the device, a modified java just-in-time engine interprets the Java byte code (= Dalvik).
- Newer devices interpret byte code faster and more efficiently.
- Garbage collection has also seen major improvement from Android 1.0 to 1.5 to 2.1 to 2.3.3.
- Now very efficient also for complex code!

App Development 101

Activity

- The most important class of your app - the one essential!
- Can be killed by Android anytime...
  Need to restore settings otherwise user will be confused!
- **When to save and when to restore? Lifecycle!**

App Development 101


Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

Only one safe place!

- save in onPause()
- restore in onResume()

Temporarily save settings by onSaveInstanceState() &onRestoreInstanceState() Will be destroyed when app is finally killed.

Permanently save settings using SharedPreferences. Will remain when restarting app.

App Development 101

Temporarily save settings using onSave/RestoreInstanceState()

- One argument: `Bundle savedInstanceState`
- Save named values via putInt/Float/Boolean/String()

```
savedInstanceState.putString("v",value);
savedInstanceState.putInt("i",10);
```

- Restore named values via getInt/Float/Boolean/String()

```
String v=savedInstanceState.getString("v");
int i = savedInstanceState.getInt("i");
```

- Internally used by Android as well, don't forget to call super.onSave/RestoreInstanceState() before returning!

Permanently save settings using SharedPreferences

- Save in onPause()
- SharedPreferences.Editor = same put functions as Bundle.

```
SharedPreferences sp = getPreferences(MODE_PRIVATE);
SharedPreferences.Editor se = sp.edit();
se.putString("v",value);
se.putInt("i",10);
se.commit();
```

^^^^^^^^^^ *Necessary to actually save the changed data!*

- Restore in onResume().
- SharedPreferences = same get functions as Bundle

```
SharedPreferences sp = getPreferences(MODE_PRIVATE);
String v=sp.getString("v","");
int i =  sp.getInt("i");
```

(no sp.edit() / editor necessary)

App Development 101

Making a control accessible from code

- add `android:id="@+id/any_name_you_want"` to XML definition of the control (if not already there)
- in java code, use e.g. for an EditText control

```
EditText et = (EditText) findViewById(R.id.any_name_you_want);
```

Saving/Restoring text from TextView & EditText

```
String saved = (et.getText()+"");
et.setText(saved);
```

Development for Android Devices
App Development 101

Multitouch, Sensors and Hello World

Alexander K. Seewald

SEEWALD SOLUTIONS

Processing touch events (1)

- Have to subclass a View (e.g. ImageView)
- To process multi-touch events in order, work like this!

```
public boolean onTouchEvent(MotionEvent ev) {
  final int historySize = ev.getHistorySize();
  final int pointerCount = ev.getPointerCount();
  for (int h = 0; h < historySize; h++) {
    System.out.printf("At time %d:", ev.getHistoricalEventTime(h));
    for (int p = 0; p < pointerCount; p++) {
      System.out.printf("  pointer %d: (%f,%f)",ev.getPointerId(p),
ev.getHistoricalX(p, h), ev.getHistoricalY(p, h));
    }
  }
  System.out.printf("At time %d:", ev.getEventTime());
  for (int p = 0; p < pointerCount; p++) {
    System.out.printf("  pointer %d: (%f,%f)",ev.getPointerId(p),
ev.getX(p), ev.getY(p));
  }
  System.out.printf("  action %d",ev.getAction());
}
```

Processing touch events (2)

- `ev.getPointerId(p)`
  - gives a unique pointer id for touch number #p
  - unique as long as touch is not removed
  - assigned the first time this pointer goes down
- `ev.getAction()`
  - *ACTION_DOWN* when first pointer goes down
  - *ACTION_POINTER_DOWN* - pointers being added (plus shifted pointer index, use getActionMasked()/Index())
  - *ACTION_POINTER_UP* - pointers being removed (plus shifted pointer index, use getActionMasked()/Index())
  - *ACTION_MOVE* - pointers move w/o going up/down
  - *ACTION_UP* - last pointer goes up. Interpret gesture now!
  - *ACTION_CANCEL* - gesture canceled, do not interpret!

Processing touch events (3)

Simple single/multitouch processing (drag & pinch/zoom)

- Store number of pointers down (maximum) and the positions of the first two pointers at beginning b0,b1 (when they first appear) and in each ACTION_MOVE c0, c1 (b0,b1,c0,c1 are 2D positions)

In ACTION_MOVE:

- If max.number of pointers down = 1: single touch, use first position b0 to current position c0 as drag gesture
- If max. number of pointers down = 2: multi-touch, use ratio of distance(b0,b1) to distance(c0,c1) to pinch/zoom

In ACTION_UP: make sure changes are made permanent (no jumps when removing all pointers)

Drawing on ImageView

Only possible for subclassed image view

- override `protected void onDraw(Canvas canvas)`
- call `super.onDraw()` in first line
- use Canvas draw functions:
  `Canvas.drawLines(), drawCircles(), ...`
  http://developer.android.com/reference/android/graphics/Canvas.html

Subclassed views are referenced in XML layout as full class (e.g. `com.SeewaldSolutions.HelloWorld.MyImageView`)

Processing sensor data (1)

- Must implement interface `SensorEventListener`

- Register a listener like this (preferably onCreate())

```
SensorManager        sm        =        (SensorManager)(this.getContext().
getSystemService(Context.SENSOR_SERVICE));
List<Sensor> l = sm.getSensorList(Sensor.TYPE_GRAVITY);
if (l.size()>0) { Sensor mSensor = (Sensor)(l.get(0));
  sm.registerListener(this,mSensor,SensorManager.SENSOR_DELAY_GAME);   }
```

- Can register an arbitrary number of sensors to same routine. Need to distinguish which sensor sent data!
- Sensor update speed depends on 3rd par. in registerListener()

Processing sensor data (2)

- `public void onSensorChanged(SensorEvent event)`
  - Called only when sensor values change
  - `int accuracy` - accuracy of event (mostly useless)
  - `Sensor sensor` - sensor who was responsible for this event
  - `long timestamp` - time in nanoseconds for this event
  - `float event.values[]` - values, depends on sensor type
- When accuracy of sensor changes, this routine is called. Part of *SensorEventListener* interface, must be implemented

```
public void onAccuracyChanged(Sensor sensor, int accuracy)
```

Getting a camera preview (1)

Camera preview needs a subclassed SurfaceView (extends ..) that at least implements SurfaceHolder.Callback.

New private member variables for Holder & Camera:

```
SurfaceHolder mHolder; Camera mCamera;
```

New private member variables for camera preview size and surface size (may be different):

```
int mWidth, mHeight, mSFWidth, mSFHeight;
```

Set up holder in constructor:

```
mHolder = getHolder();
mHolder.addCallback(this);
mHolder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
```

Getting a camera preview (2)

Open camera in surfaceCreated():

```
public void surfaceCreated(SurfaceHolder holder) {
  mCamera = Camera.open();
  mHolder = holder;
  mCamera.setPreviewDisplay(mHolder);
}
```

Close & release camera in surfaceDestroyed():

```
public void surfaceDestroyed(SurfaceHolder holder) {
  if (mCamera!=null) {
    mCamera.stopPreview();
    mCamera.release();
    mCamera=null;
  }
}
```

© 2012

App Development 101

Getting a camera preview (3)

Start preview in surfaceChanged()

```
public void surfaceChanged(SurfaceHolder h, int format, int w, int h) {
  mSFWidth=w; mSFHeight=h; mWidth=w; mHeight=h;
  Camera.Parameters p = mCamera.getParameters();
  p.setPreviewFormat(PixelFormat.YCbCr_420_SP);
  p.setPreviewSize(w,h);
  mCamera.setParameters(p);
  mCamera.startPreview();
}
```

May not always work, may be slow on older devices with large screen; fails on some buggy devices... better to compute best preview size that fits to screen (possible upscaled)!

Getting a camera preview (4)

Given w,h - find best fit preview size!

```
Camera.Parameters p = mCamera.getParameters();
List<Camera.Size> ls = p.getSupportedPreviewSizes();
Camera.Size bAR = null; Camera.Size bOV = null;
for (int i=0; i<ls.size(); i++) {
  if (w*ls.get(i).height == h*ls.get(i).width) { // same aspect ratio
    if (bAR==null || Math.abs(w*h-ls.get(i).height*ls.get(i).width)
<Math.abs(w*h-bAR.width*bAR.height)) { bAR=ls.get(i); }
  } else if (bOV==null || Math.abs(w*h-ls.get(i).height*ls.get(i).width)
<Math.abs(w*h-bOV.width*bOV.height)) { bOV=ls.get(i); }
  }
}
if (bAR!=null && (bOV==null || Math.abs(w*h-bAR.width*bAR.height)
<1.5*Math.abs(w*h-bOV.width*bOV.height))) {
  w=bAR.width; h=bAR.height; // use best aspect ratio preview
} else {
  w=bOV.width; h=bOV.height; // use best overlay preview
}
```

© 2012

Install Android SDK!

- Choose the right version for your operating system
  http://developer.android.com/sdk/index.html

- Start SDK Manager and install recommended packages
  (Win: SDKManager.exe; MacOS/Linux: tools/android sdk)

- Ensure SDK Manager has Android 2.3.3 SDK (Gingerbread, API Level 10) & SDK Tools Rev. 20.0.3 installed. We will work with 2.3.3 SDK (targetSdk=10) throughout.

App Development 101

Install Eclipse for Java! (preferred IDE)

- At least version 3.6.2 (Helios)

- Linux: can probably install from default repository
- MacOS/Windows: install newest version from homepage http://www.eclipse.org/downloads/

- Use AVD manager to create a SDK level 10 emulator device with WVGA800 screen size.

Install ADT Plugin for Eclipse!

- Install instructions: http://developer.android.com/sdk/installing/installing-adt.html

- *Possible to use another IDE or no IDE at all. Toolchain is also available as command line using ant.*

Creating our first app using Eclipse

- Click on "New Android App" in toolbar

  (if it's not there, need to install ADT plugin and/or install SDK)
- **Build SDK:** is used to compile the app, >= Min.Req.SDK
- **Minimum Required SDK:** what API level the app requires at least. If you have Build SDK > Minimum Required SDK, need to check whether features are available using Java reflection.
- **App name:** Needs to be unique on whole market

  Usually: `com.[CompanyName].[AppName]` (all ASCII)
- **No test project, no copying from sample code**

This will create the Android project directories plus some initial code.

App Development 101

Hello World - Our First App

Install the app on your emulator! Start it and see if it works!

- Debugging
  - ADT plugin LogCat view shows the recent log entries
    Check the ... *caused by* (first line of error, scroll up!)
  - From command line: run `adb logcat`
  - `System.out.println` will output to log as well
  - Only when encountering a hard reset will you lose logs
    (can happen e.g. with Camera preview surfaces)
  - All this will work with emulator & device (make sure not both of them are running at the same time!)

© 2012

App Development 101

Hello World - Our First App

Exercise 1: Make this app multi-language (add German!)

Hello World - Our First App

Exercise 2: Add your own app logo instead of the default one. Leave the old one in, give the new logo another name.

Hello World - Our First App

Exercise 3: Add one arbitrary hardware feature and one arbitrary permission to your app, see what happens when running it!

App Development 101

Hello World - Our First App

Exercise 4: Change all application text by modifying res/values/strings.xml and res/values-de/strings.xml . Is there something you cannot change?

App Development 101

Hello World - Our First App

Exercise 5: Add EditText in HelloWorld layout/main.xml, then save/restore its contents using...

1) temporary method (onSave/RestoreInstanceState())

2) permanent method (SharedPreferences in onPause/ onResume)

Also set TextSize to 20 sp for original text and EditText!

Switch away from app (e.g. home button), back again, etc... see which method does what you expect it to do!

App Development 101

Hello World - Our First App

Exercise 6: Add nine TextViews below the existing one. Use these to output sensor values from TYPE_GRAVITY (first three), TYPE_LINEAR_ACCELERATION (middle three) and TYPE_ROTATION (last three). This will only work on an actual device, so stop the emulator and connect your device.

Make sure Settings - Applications - Development options - USB Debug is switched on. Then installing on device via ADT plugin should work.

Tell us your experiences with the values from these sensors! Which are reliable, which are not and why?

Hello World - Our First App

Exercise 7: Comment out all text views in XML layout, add one subclassed image view and implement onTouch() there.

1) use the provided onTouch() code from slide, see if you get touch events and if you understand them

2) visualize touches by overriding onDraw() - e.g. each pointer as drawCircle(), lines between any two pointers as drawLines()

3) (optional) draw anything on Canvas, scale/drag using single & multitouch gestures.

Development for Android Devices
App Development 101

Team Work

Alexander K. Seewald

SEEWALD SOLUTIONS

App Development 101

Teamwork!

- Groups of 1-3 people
- Implement any app you want, provided it is...
  - feasible to finish within 1.5 days
  - uses (mostly) what we have done
- Ideas
  - Presentation app using single touch moves to go left/right, multitouch for pinch/zoom and rotate gestures (use sample bitmaps for slides)
  - Virtual horizon: Visualize the virtual horizon using output from appropriate sensors (optional) over camera preview.
  - Labyrinth: Implement a simple labyrinth using only Canvas draw functions. Ball should be movable using appropriate sensors following gravity (~ normal labyrinth game)

App Development 101

You have until tomorrow afternoon to finish your app.

I will be available for questions throughout!

Development for Android Devices

Team Presentations

Alexander K. Seewald

SEEWALD SOLUTIONS

Team Presentations

- Every group distributes their app to all other groups.
- Install each app on each device
- Short presentation from each group about their app
- Together we will test all apps and see if we can find bugs!
- Afterwards we will discuss our experiences

Development for Android Devices

Outlook

Alexander K. Seewald

SEEWALD
SOLUTIONS

What can be expected in the future?

More Of The Same

- Faster devices: more computing power, faster internet access
- Larger devices: more internal and external memory, larger screens, higher-resolution cameras
- Special purpose devices: adapted to specific users' needs, e.g. for old people = simpler interface with larger buttons

Somewhat Different

- Flexible displays (possibly whole devices): researched for 10+ years, few prototypes. Should be available in 3-5 years
- Alternative displays (e-Ink ~ Kindle, other e-book readers)
  - need special interfaces because screen refresh ~ 1-2s
- Will we always have to work with capacitive touchscreens? Alternative input/output devices!

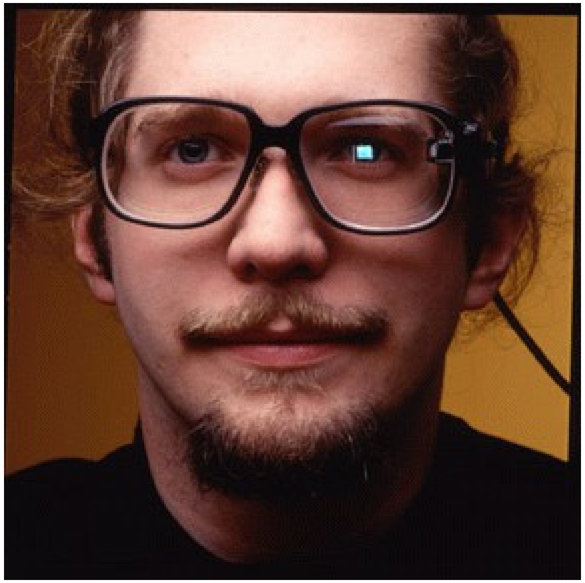
Alternative Output Devices (1)

Project Glass (Google)

- Virtual display & camera integrated into a glasses-like frame
- Not suitable for Augmented Reality
  - Small screen (~ 19" at 2m distance), Only one eye
  - Far less than needed for full immersion
- Not stand-alone usable: no sensible input! Controlled e.g. by smartphone (so you have to lug two devices around...)
- Not a new idea at all!

Alternative Output Devices (2)

Similar devices have been available for decades.



Thad Starner, Wearable Comp. Pioneer =

- This device was offered to me in 1999 at 3000 CHF at a conference (resolution: 320x240, 16 colors, no camera ;-)

Steve Mann, the grandfather of Wearable Computing (Thad's doctoral advisor) =

- **EyeTap:** Several generations of prototype full immersion augmented reality glasses (display + camera), going back to 1981.

Now this I'd like to see in shops!

Alternative Input Devices

Output devices better than Project Glass are already available, how about input devices?

<u>Current Inputs</u>

- Chorded (one hand) keyboard & mouse (e.g. Twiddler2.1)
  - Must be specifically learned (moderate effort)
  - Also available wireless via bluetooth
- Smartphone/Tablet via multitouchscreen, integrated keyboard
- Speech recognition (very susceptible to environment noise)

<u>Future Inputs</u>

- Head movements, Eye Tracking, Hard movements
- EEG analysis (tricky)
- Robust speech recognition

Alternative Hardware Devices (1)

Android runs on other devices as well....

Raspberry Pi

- cheap (~ 25-35US$) ARM device
- low power (~ 75-125mA)
- small (large matchbox)
- can run Android 4.0!

E-book Reader Devices

- Barnes & Nooble Nook *
- enTourage eDGe
- Spring Design Alex eReader
- PocketBook eReader IQ 701

Alternative Hardware Devices (2)

Netbooks

- Acer Aspire One (dual-boot with WinXP/Win7)
- Augen GenBook 108
- Toshiba AC100

Smartwatch

- Motorola MOTOACTV (GPS Fitness tracker)
- Blue Sky I'm Watch
- Sony SmartWatch
  - both are just input/output devices for a smartphone...
  - the first smart watch, however, was...

Alternative Hardware Devices (3)

Timex Datalink 150
Available 1994 - 2010

Features

- Calendar, count-down timer, tasks, two time zones, notes; write your own wristapps via SDK
- 64K ROM, 2K EEPROM, very limited memory
- Import from MS Schedule+ via "blinking lines"
- Completely autonomous
- 3 year runtime out of a single battery

However: No possibility to input calendar entries; tasks could only be marked "done" and otherwise not changed (five buttons would have been enough for simple chorded keyboard ;-)

Challenges

Full immersion augmented Reality (~ EyeTap)

- Realtime (<12ms) alignment of head position with environment overlay (otherwise "motion sickness")
- Overlay must not distract overly from environment (otherwise "lose one eye" effect)
- Overlay may need 3D data on surrounding environment. Very hard even using stereo cameras! 3D cameras ~ Kinect, Xtion? (they work well but need 12W power - too much for mobile!)
- Analysis of image data from camera might be useful for head position estimate but takes too much computing power. Currently not feasible. Best: Integrate 3-axis gyroscope & acceleration sensor into headset!
- Would be very interesting to write apps for this!

© 2012