ANDROID/IPHONE BASICS FROM DATA TO DESIGN

21W.789 CLASS 2

Android Fundamentals

Anatomy of an app

Basic APIs

Anatomy of an Android Application:

- Applications: Process (set of screens)
- Activities: Application components (screens)
- Intents: Messages among components (what tasks an activity can perform/events to be notified about)
- Services: Background tasks that can be performed without an application-specific UI visible

Android application model

- One application (apk file) = one process

- Processes are isolated
  - IPC is done through Intents or Services

Components – Activity

- Single, focused thing that a user can do
  - Generally a single screen
  - Consists of a hierarchical collection of Views (UI Components)

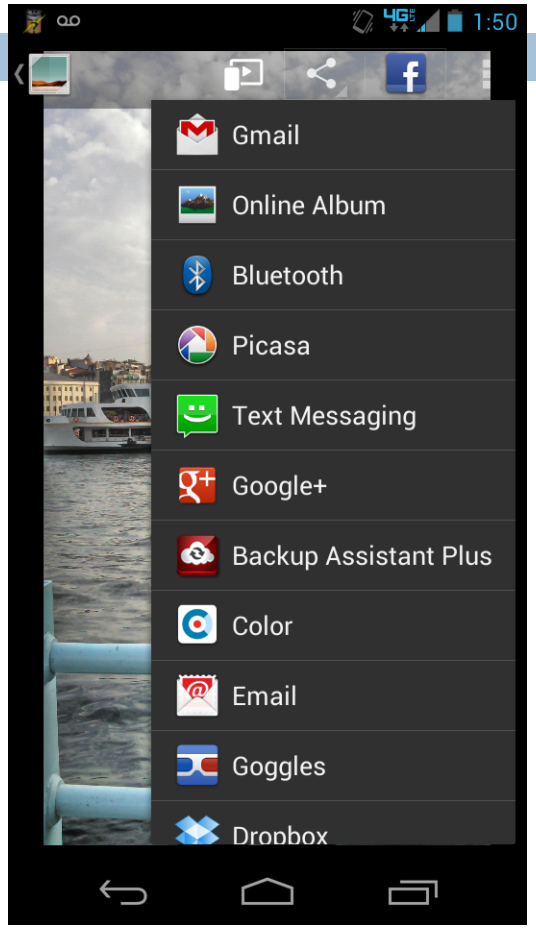
- One activity = one screen in app
  - Current activity starts next one (next screen)
  - One activity marked to be shown at app launch
  - Window does not have to be full screen
    - floating, embedded within another activity

Components – Service

- Used for backgrounds tasks
  - e.g. site polling, data synch, network download
  - CPU intensive (e.g. MP3 playback) or blocking (e.g. networking) services should spawn their own thread
    - In latest Android, networking on main thread throws an exception
    - If spawning a thread, make sure you force device to stay awake, or it might shut off the CPU and go into a power saving mode
  - Can run when application UI is not visible (Start "sticky" if you want them to auto-restart if killed)

Intent

- Forms the glue between Activities
- An abstract description for
  - an operation to be performed
  - something that has happened



- Syntax:

```
startActivity(new Intent(ACTION_DIAL, Uri.parse("tel:6175551212")));
```

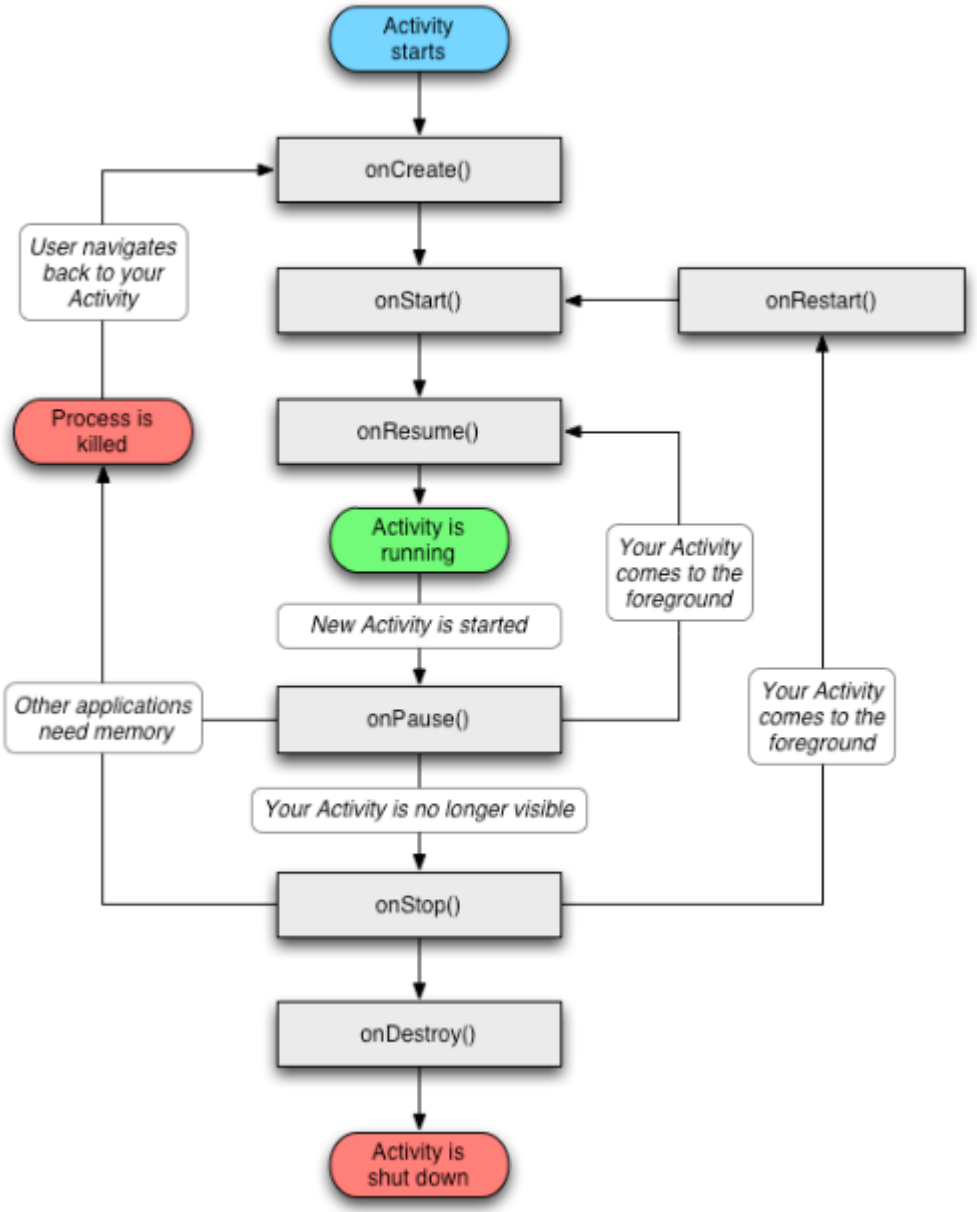
Example Intents

ACTION_VIEW	`content://contacts/people/1`
ACTION_DIAL	`tel:16175551212`
ACTION_SEND	`Extras for subject, text, recipients, data, etc.`

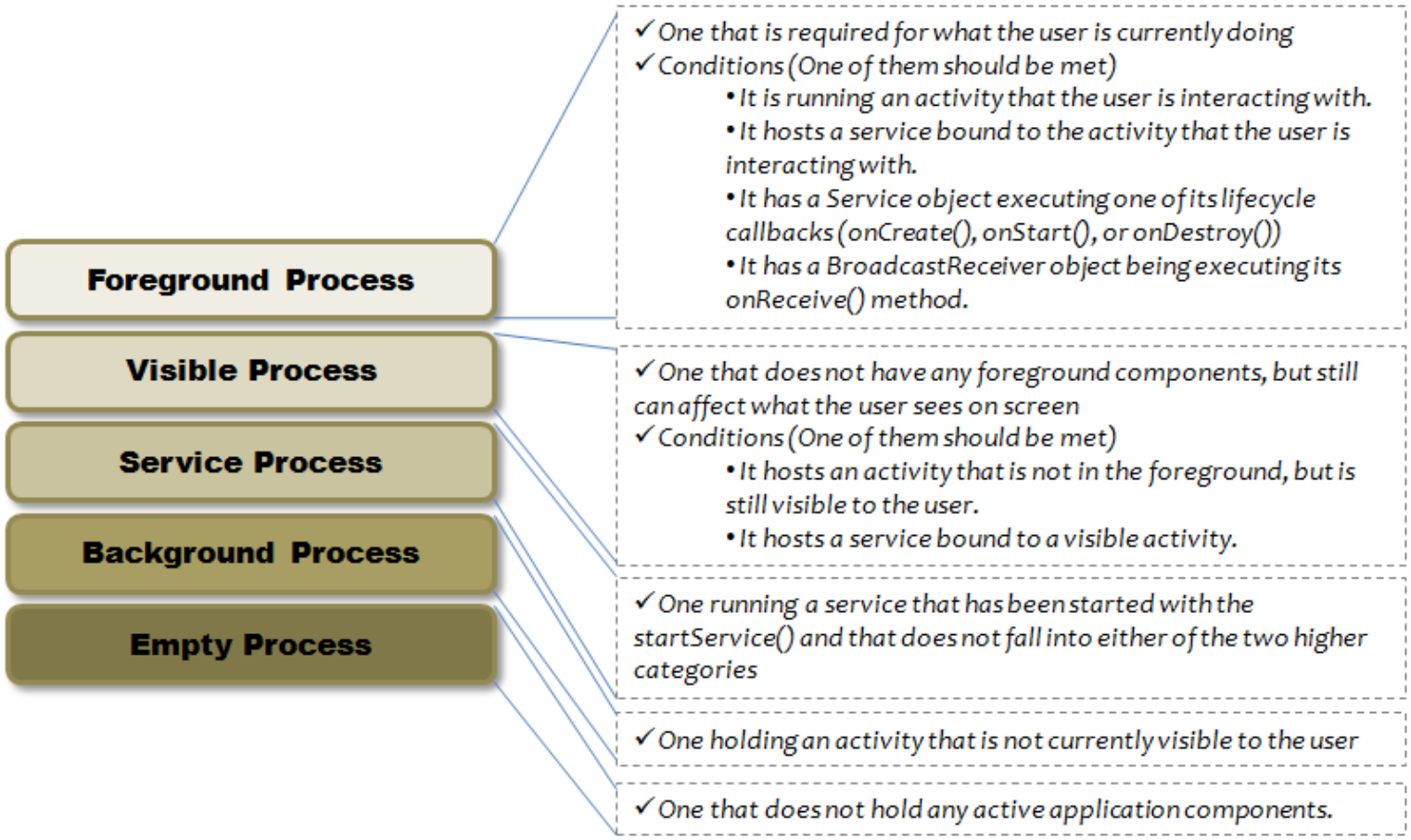
Action Data

Intent = “Show the data pointed to by this URI”

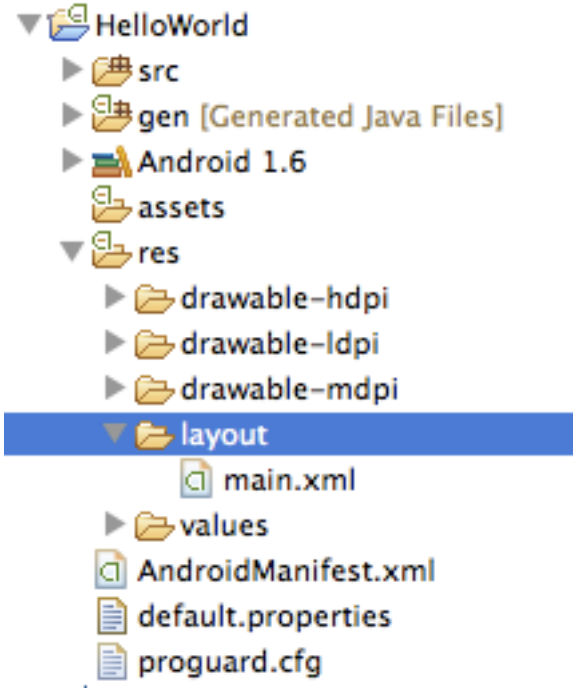
The activity lifecycle

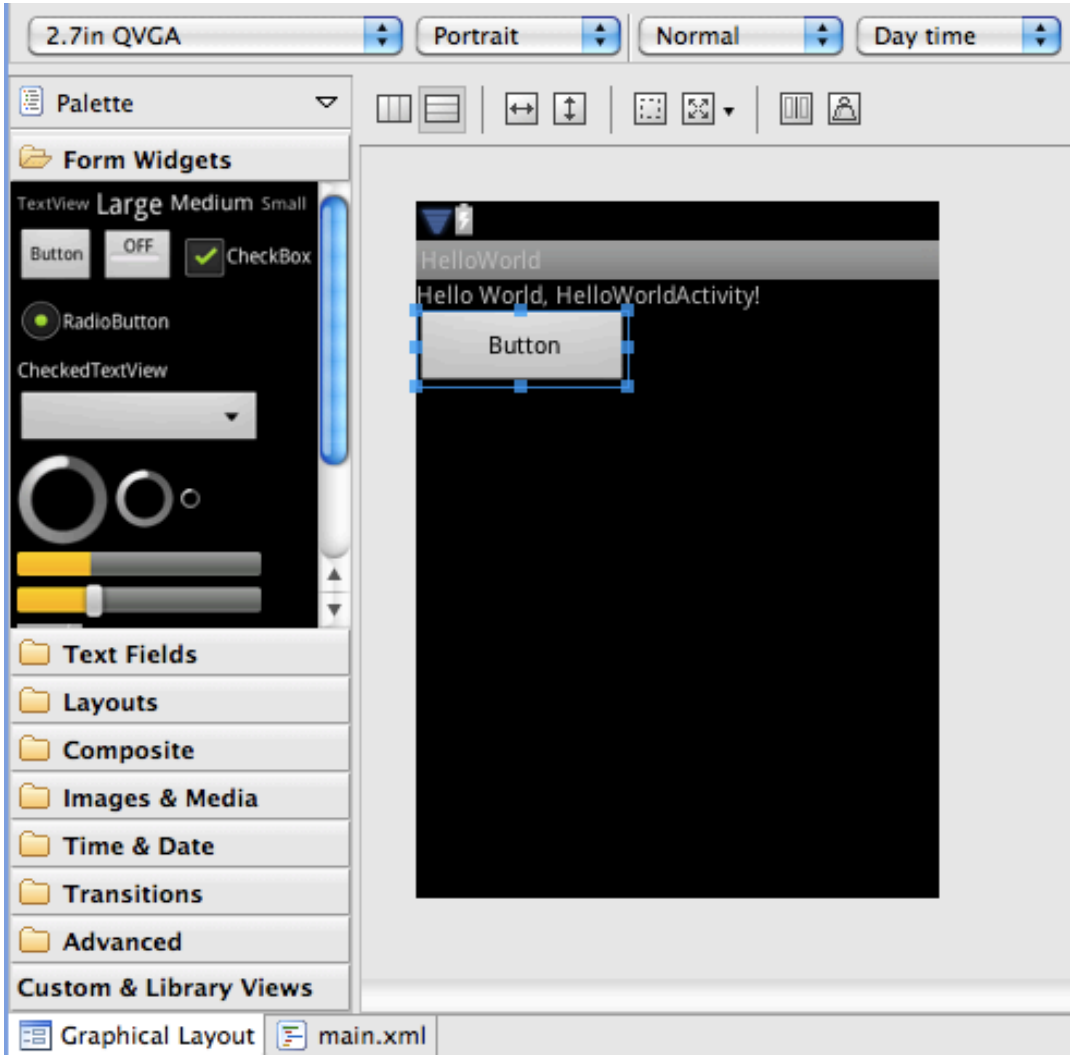


Activity priority list



Creating Layouts

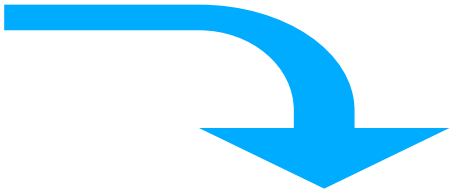




Attaching Views to code

```
…
<Button android:id="@+id/test_button"
android:text="Button"
android:layout_height="wrap_content"
android:layout_width="130dp" />
…
```

main.xml



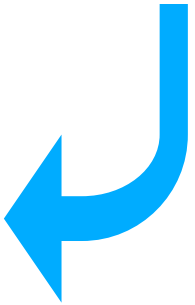
```
public static final class id {
  public static final int
      test_button = 0x7f050001;
}
```

R.java

myactivity.java

```
setContentView(R.layout.main);

Button testButton =
   (Button) findViewById(R.id.test_button);
```



The Manifest

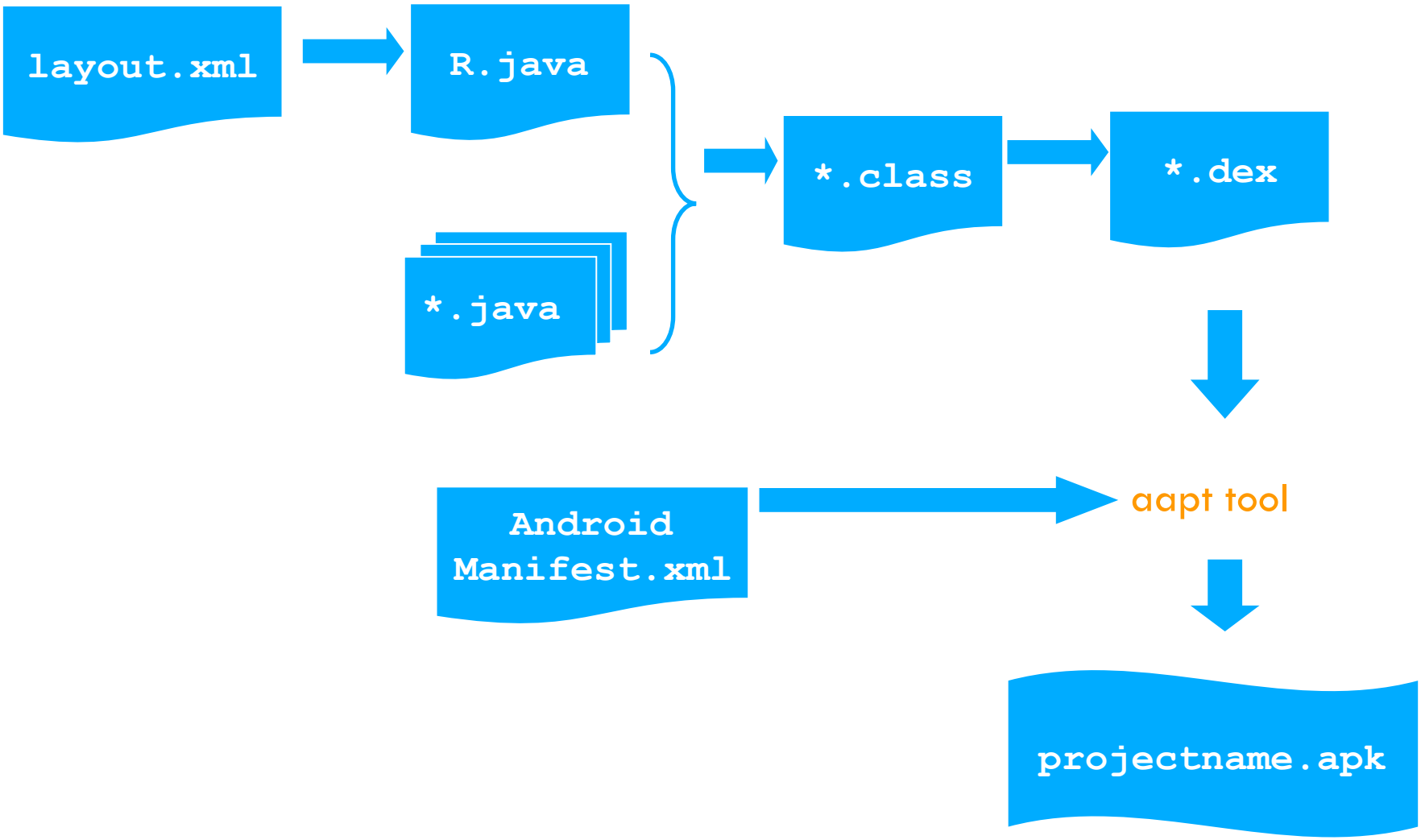
- All new Activities and Permissions need to be declared in the Android manifest file for your application
- Example:

```
<?xml version="1.0" encoding="utf-8"?>
        <manifest xmlns:android=http://schemas.android.com/apk/res/android
          package="com.motorola.health.mashups.mit.amarino"
          android:versionCode="1"
          android:versionName="1.0" >
        <uses-sdk android:minSdkVersion="15" />
        <uses-permission android:name="android.permission.INTERNET"></uses-permission>
        <application
          android:icon="@drawable/ic_launcher"
          android:label="@string/app_name" >
                  <activity
                    android:name=".BeaconAmarinoActivity"
                    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>
```

Common Permissions

- <uses-permission android:name="android.permission.INTERNET" />
- <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
- <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
- <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
- <uses-permission android:name="android.permission.SEND_SMS" />
- <uses-permission android:name="android.permission.CALL_PHONE" />
- <uses-permission android:name="android.permission.VIBRATE" />
- <uses-permission android:name="android.permission.SET_ORIENTATION" />
- <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
- <uses-permission android:name="android.permission.RECORD_AUDIO" />
- <uses-permission android:name="android.permission.CAMERA" />
- <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

Putting everything together



Loading an APK on your device

- Generate APK file in eclipse
- Use “adb” tool in the Android SDK
- Enable debugging tools in settings on device
- Plug in phone with USB cable
  - Charging only mode (not USB mass storage)
- Run “adb install nameofapk.apk” / or just “Run” from Eclipse
- NOTE: Emulators and phones behave quite differently!!

- To take a screenshot of app run “ddms” / or on Android 4.0+ press “volume up” and power simultaneously
  - Useful for posters, final reports

Other useful APIs

- HTTP
  - Uses Apache Commons library

```
String url = "http://web.mit.edu/");
HttpClient client = new DefaultHttpClient();
HttpGet request = new HttpGet(url);
Try
{
        HttpResponse response = client.execute(request);
}
```

- Accelerometer
  - Good example code here: http://mobilehealth.posterous.com/example-for-accessing-the-accelerometer-with

iPhone Development

- Need to have a paid Apple Developer account to launch app on a real device
- Generate certificate with UDIDs of devices

- Distribution
  - Debug load directly on phone
  - AdHoc distribution (.mobileprovision file + app bundle)
  - iTunes store (requires Apple approval)

Objective C

- Superset of C
  - Can Mix C/C++ and Objective C
  - Single Inheritance
  - Loosely typed (treat compiler warnings seriously!)
- Syntax:
  - [instance method];
  - [instance method:arg1 arg2name:arg2];

Strings, Logs, and Arrays

- Strings
  - NSString *myString = @"my string";
  - [NSString stringWithFormat:@"with number: %d",5];
- Logging
  - NSLog(@"debug info here");
- Arrays
  - NSArray *array = [NSArray arrayWithObjects:@"One", @"Two", @"Three", nil];
  - If any of your objects is nil, array will not be full!!

View Controllers

- Application contains a NavigationController
- Each screen is a UIViewController
- New screens appear with a push of a View Controller onto Navigation Controller:

```
[[self navigationController] pushViewController:targetViewController animated:YES];
```

Application Lifecycle

- Applications suspended when phone sleeps or when interrupted (e.g. incoming call)
  - On wake-up, -(void)applicationDidBecomeActive called on AppDelegate
  - All state maintained, but no execution occurs while application is inactive
- Newer versions of iOS allow some background events (notifications, location triggers, etc.) but not the running of arbitrary background code

iPhone resources

http://www.stanford.edu/class/cs193p/cgi-bin/index.php Stanford iPhone Class

http://ericasadun.com/ Erica Sadun's iPhone Coo kbook

http://www.cocoabuilder.com/archive/bydate Cocoabuilder

http://cocoadevcentral.com/articles/000082.php CocoaDevCentral: Cocoa Style for Objective-C: Part I

http://www.iphonesdkarticles.com/ iPhone SDK Articles

http://cocoadevcentral.com/ Cocoa Dev Central

http://icodeblog.com/ iCodeBlog

http://theocacao.com/document.page/510 Theocacao

http://www.v2ex.com/tag/uitableviewcell/ UITableViewCell | V2EX

http://idevkit.com/forums/tutorials-code-samples-sdk/30-custom-uitableviewcell.html Custom UITableViewCell - iDevKit

http://pegolon.wordpress.com/2008/11/15/using-uitableviewcell-with-interfacebuilder/ Building UITableViewCell with IB

http://discussions.apple.com/thread.jspa?threadID=3D1579070&tstart=3D43 Loading views in landscape orientation

http://discussions.apple.com/thread.jspa?threadID=3D1603141&tstart=3D27 Half-curl transitions

http://cocoawithlove.com/2008/12/heterogeneous-cells-in.html Heterogeneous cells in a UITableViewController

https://www.nearinfinity.com/blogs/scott_leberknight/iphone_bootcamp_blogs.html Phone bootcamp blogs

http://www.sleberknight.com/blog/sleberkn/entry/iphone_bootcamp_day_4 iPhone bootcamp blogs 2

http://www.iphonedevsdk.com/forum/iphone-sdk-development/4879-uitableview-cell-deletion-methods.html UITableViewCell deletion methods

http://savoysoftware.com/blog/ enhancing performance iPhone

http://stackoverflow.com/questions/328391/last-indexed-cell-in-uitableview-is-taking-on-wrong-font Cell Identifiers

http://stackoverflow.com/questions/tagged/iphone StackOverflow

http://www.cocoadev.com/index.pl?NSUserDefaults NSUserDefaults

http://knol.google.com/k/usman-ismail/iphone-sdk-application-preferences/34oprzanmpe7q/8# Application Preferences tutorial

http://icodeblog.com/2009/02/02/great-resource-for-all-iphone-developers-ibetatestcom/ iBetaText.com

http://blog.coriolis.ch/2008/11/09/add-an-uiprogressview-or-uiactivityindicatorview-to-your-uialertview/ progressView

http://idevkit.com/forums/general-sdk/299-nsurlconnection-nshttpcookie.html NSURLConnection, NSHTTPCookie - iDevKit

http://stackoverflow.com/questions/576265/convert-nsdate-to-nsstring Convert NSDate to NSString - Stack Overflow

http://www.cocoadev.com/index.pl?DescriptionWithCalendarFormat CocoaDev: DescriptionWithCalendarFormat

http://www.planetcocoa.org/ Planet Cocoa

http://www.oiledmachine.com/posts/2009/01/04/managing-concurrent-asynchronous-url-requests-in-cocoa.html Managing concurrent asynchronous URL requests in Cocoa

From Data to Design

Data Analysis

- Flow Models
- Grounded Theory Affinity Analysis

Generating Design Ideas from Data

From Data to Design:

- Ultimate goal is successful concept idea grounded in data from users

1. Make sense of data
   - Find Patterns
   - Develop deep understanding of current practices and desires in your domain
2. Create design guidelines
3. Create design ideas grounded in data

Data Analysis

- Qualitative methods generate a LOT of data
  - Typical full study ~1500 notes
- Most data very descriptive in nature
- Analysis used to build models of use and inspire new ideas

Types of qualitative data analysis

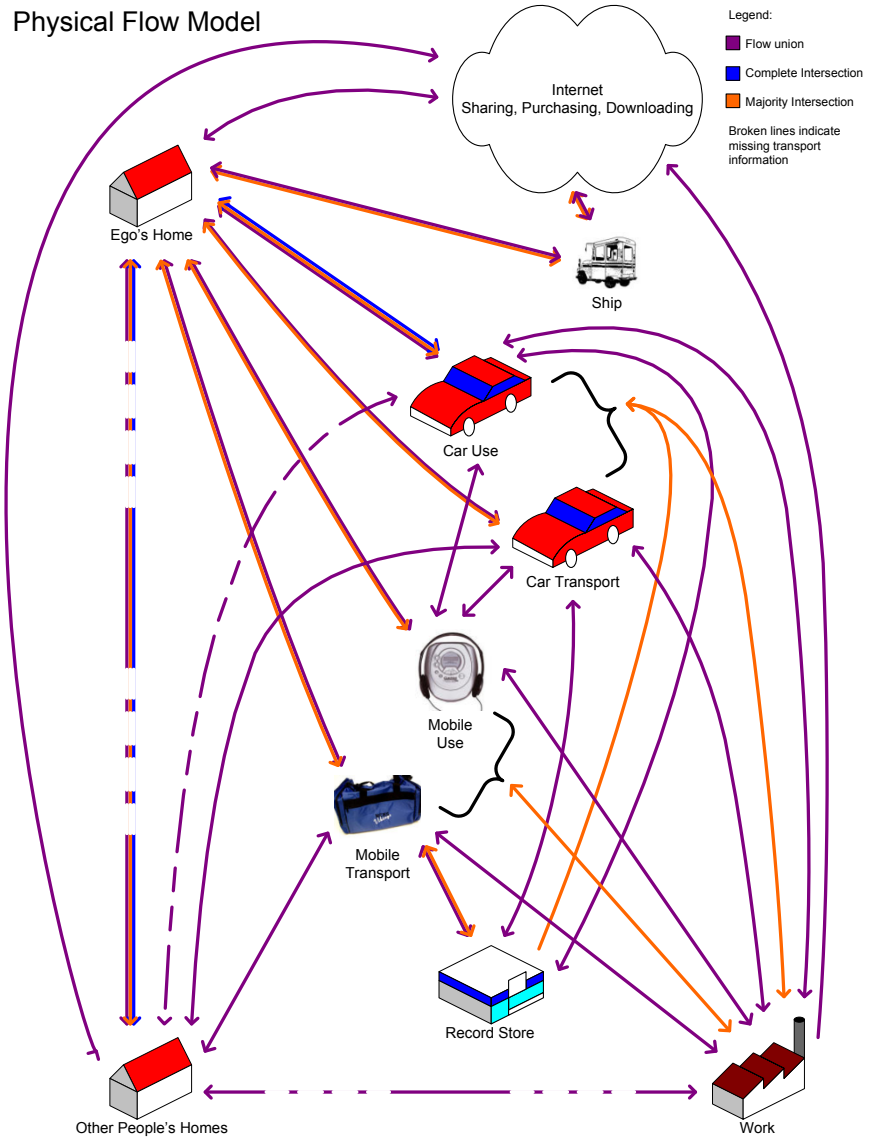
- Flow Models
- Conversation Analysis
- Critical Incident Analysis
- Affinity (Grounded Theory) ***

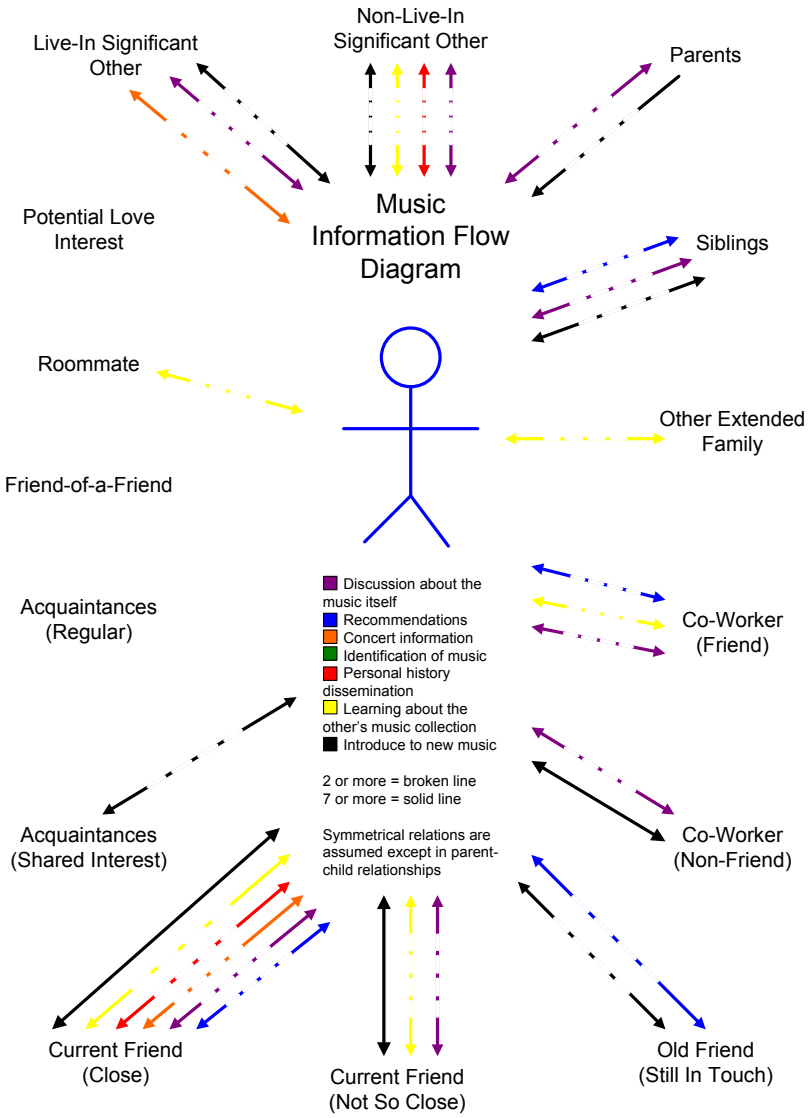
*** focus for this class

Flow Models

- Developed in Contextual Design by Beyer and Holtzblatt
- Builds model of how information, physical objects flow through the environment and between people
- By looking through data, or collecting directly in-situ helps understand bottlenecks, smooth-points in interaction

Examples:


Physical Flow Model
Legend:
Flow union
Complete Intersection
Majority Intersection
Broken lines indicate missing transport information
Internet
Sharing, Purchasing, Downloading
Ego's Home
Ship
Car Use
Car Transport
Mobile Use
Mobile Transport
Record Store
Other People's Homes
Work


Live-In Significant Other
Non-Live-In Significant Other
Parents
Potential Love Interest
Music Information Flow Diagram
Siblings
Roommate
Other Extended Family
Friend-of-a-Friend
Acquaintances (Regular)
Discussion about the music itself
Recommendations
Concert information
Identification of music
Personal history dissemination
Learning about the other's music collection
Introduce to new music
2 or more = broken line
7 or more = solid line
Symmetrical relations are assumed except in parent-child relationships
Co-Worker (Friend)
Acquaintances (Shared Interest)
Co-Worker (Non-Friend)
Current Friend (Close)
Current Friend (Not So Close)
Old Friend (Still In Touch)

Conversation Analysis

- When you have logs of recorded conversation / textual logs from digital systems
- Analyze flow of conversation (openings, closings, requests)
- Analyze how medium affords or discourages certain types of conversation / interactions
  - "I can't talk now, I'm in a fitting room" – Weilenmann et al

Critical Incident Analysis

- Analyze in detail specific interactions of note
  - Places where participants got stuck in a task
  - Places where participants got excited
  - Places where complex workarounds exist today
- Focus on tasks that were being accomplished and necessary steps in completing that task
- Great for designing systems that improve on existing solutions, incremental changes

Grounded Theory/Affinity Diagrams

- A tool used to organize large amounts of qualitative data into logical and linked categories based on recognizable relationships

- Helps to generate holistic explanations of interrelated phenomena
- Provides the foundation of inductive explanations
- Accommodates brainstorming for new concepts

- What it does *not* do:
  - Test hypotheses
  - Prove/disprove theories

Thanks to Crysta Metcalf, Motorola for slides on Grounded Theory

What an affinity looks like

Where the Affinity Method Comes From

- Japanese Anthropologist Jiro Kawakita (KJ Method)
  - Crisis of method: standard field techniques weren't working
  - Rejects the imposition of preconceived ideas and hypotheses
  - Inductive method for the "holistic integration of qualitative data" (examining interrelationships between phenomena)
  - Used the method for technological innovation! (ropelines and pipelines for the Nepalese Sikha Valley villagers)
- Hugh Beyer and Karen Holtzblatt
  - Psychologist and anthropologist
  - Adapted the affinity from the KJ method
  - Different from the original KJ method (items are insights)
  - Popularized the affinity method in the design and HCI community in the U.S.

Steps in the Basic Method

- Qualitative Fieldwork and Data Collection
- Creating Post-It Notes (or Data Cards)
  - 1-2 Sentences
  - Try to get a single idea on the note
- Putting up the notes
  - "Memory game" problems
  - Bucketing problems
- Grouping the notes
  - Check the interpretation of the note
  - Group the notes based on their affinity to each other
  - Label the groups
  - Create groups of groups, in a hierarchical tree-like diagram, eventually bringing all the data together to tell a single story

Basics of the GT Affinity

(Part 1)

What is it: Inductive Hypothesis Generation

- Item level (create the post-its)
  - "*As analytic categories emerge, pull all the data (that is, exemplars) from those categories together…*
- Pattern level (create the groupings)
  - "*…and compare them, considering not only what [items belong] in each emerging category but also how the categories are linked together.*
- Constitutive level (create the story)
  - "*Use the relationships among categories to build theoretical models, constantly checking the models against the data…*"

(Quotes from H. Russell Bernard, 1998, Handbook of Methods in Cultural Anthropology, p.608)

Basics of the GT Affinity

(Part 2)

- Identifying Themes (Patterns)
- Look for:
  - Repetitions ("topics that occur and reoccur")
  - "Indigenous categories" (locally specific terms, expressions)
  - Similarities and differences (constant comparison method)
  - Analogies
  - Linguistic connectors (causal such as "because," sequential such as "before," conditional such as "if," etc.)

(From Ryan, Gery W., and H. Russell Bernard, 2003, "Techniques to Identify Themes," Field Methods 15(1) 85-109)

How To: Rules to Work By

- Creating the Team
  - Who?
  - How many?
- Grouping the Items
  - Think about design implications
  - Think about the research questions
  - Think about what the research is meant to inform
  - Think about how your perspective is biasing the interpretation
- Working as a Team
  - Read each note aloud as you put it up
  - Talk about what "goes with" what until the groups make sense (negotiated truth)
  - Feel free to move and modify groups until everyone is happy with the story being told
  - Be open to other people's interpretations

Variations of the Method

- The Beyer and Holtzblatt Method (B-H)
  - Uses researcher insights from the data
  - Rapidly generates descriptions, furthest from the data
  - Better if you want to quickly devise possible solutions for the problem/issue at hand
- The Original KJ Method (KJ)
  - Uses researcher summaries of the data
  - Rapidly generates explanations, closer to the data
  - Better if you want to understand the complexity of the situation being studied
- The Grounded Theory Method (GT)
  - Uses the data itself
  - Semi-rapidly generates hypotheses, closest to the data
  - Better if you want predictive explanations of behavior that can be used for other projects

Affinity Example

"Skip, too fast, skip, too slow, skip, skip … now this is ok."

"As soon as a Dylan song comes on, I have to play the whole album."

"I want to listen to something newer than this…I'm tired of listening to 80s music."

"I was playing my music on random and all of a sudden a country song came on. Then I switched to searching by genre to listen to other country music."

Affinity Example (2)

I want to play music that is different from the currently playing song in some way.

"Skip, too fast, skip, too slow, skip, skip … now this is ok."

"I want to listen to something newer than this…I'm tired of listening to 80s music."

I want to listen to music that is like the currently playing song in some way.

"As soon as a Dylan song comes on, I have to play the whole album."

"I was playing my music on random and all of a sudden a country song came on. Then I switched to searching by genre to listen to other country music."

Affinity Example (3)

It should be easy to jump to other music based on the metadata of the currently playing song

I want to play music that is different from the currently playing song in some way.	I want to listen to music that is like the currently playing song in some way.
"Skip, too fast, skip, too slow, skip, skip … now this is ok."	"As soon as a Dylan song comes on, I have to play the whole album."
"I want to listen to something newer than this…I'm tired of listening to 80s music."	"I was playing my music on random and all of a sudden a country song came on. Then I switched to searching by genre to listen to other country music."

Affinity Best Practices

- When you're done, no group should have more than 5 notes – if you have more than 5, there's likely a way to break it down further into subgroups
- Each lowest level group should express observations from multiple participants
- Names for first level groups are often in the first person and summarize the important similarity in the data below (e.g. "I want to play music that is different from the currently playing song in some way.")
- Names for second level groups are often in the 3rd person discussing larger trends (e.g. "It should be easy to jump to other music based on the metadata of the currently playing song")

Ideation

- Design ideas should be:
  - Inspired by data
  - Grounded in real-world observations
- In brainstorming, no idea is a bad idea
- Think beyond what people are doing today

Design Ideas

It should be easy to jump to other music based on the metadata of the currently playing song

I want to play music that is different from the currently playing song in some way.

I want to listen to music that is like the currently playing song in some way.

DI: Metadata Knob, dial knob like a radio to "tune into" music on varying metadata (e.g. year, BPM, last played, etc.)

"Skip, too fast, skip, too slow, skip, skip … now this is ok."

"As soon as a Dylan song comes on, I have to play the whole album."

DI: Playtree, see multiple options for the next song to play based on the metadata for the current song.

"I want to listen to something newer than this…I'm tired of listening to 80s music."

"I was playing my music on random and all of a sudden a country song came on. Then I switched to searching by genre to listen to other country music."

Prioritizing Design Ideas

- Rate concepts based on factors that matter to you
- Suggestions:
  - Are you excited about the idea?
  - Is there a large potential market?
  - Would people pay for it? (or is there an opportunity for alternate business models, e.g. ads)
  - Can you build it in the scope of the class?
  - …

In-Class activity

- In your studio sections:
  - Perform affinity analysis from field observations
  - Identify first and second-level groupings
  - Brainstorm design ideas for applications/services based on data
  - Prioritize ideas, choose one for semester project
  - TAs will help you through this and answer questions, show them your final affinity diagram and top design ideas before leaving

Assignments:

- You have 2 weeks until the next class

- Solidify design idea into application concept
- Write a formal proposal of concept (think of this as a request for funding from a VC or masters thesis proposal)
  - References to related work (see class website and proceedings of CHI and Mobile HCI in the ACM Digital Library/Google Scholar)
- SHORT Oral Proposal in next class

- Implement a simple Hello World app on your development phone

Written proposal for next class…

Length: 10 pages (including figures). Although a single student may be serving as editor and content gatherer, all students in the groups are required to author sections of the proposal related to their chief area of responsibility. Add an appendix stating the page count written by each author.

Title page: name of project, names of team members, group email address, type of report (proposal), and current date.

Abstract: one paragraph, ca. 150 words; state the problem, methods, expected results; no figures or references in abstract; do not use first person pronouns.

Table of contents: With a list of figures, if you have four or more. Figures should be numbered and labeled.

Introduction: background motivation for the project. This section establishes the need for the project; state primary and secondary audience.

Motivation: the motivation for your project based on qualitative research study, description of study and analysis, purpose of the service/ app; its scope.

Description of project: what are the key use cases for your application/service, design strategies you will employ, technical requirements, tools needed and how you will acquire them, any platform/browser dependencies.

Tasks and milestones: show a Gantt chart which divides the life of the project into definable tasks (vertical axis) over time in weeks (horizontal axis). Punctuate the horizontal axis with important milestones you are expected to meet. Roles each team member will perform.

References (Include relevant work from the ACM Digital Library and other sources)

Please Note: All figures are given a caption and a figure number (placed below the figure) and are referenced in the text ("See Figure 1"). Figures should be placed within the text as close as possible to the reference.

Email PDF to ebarrett, bentley and bring one hard copy to class

Oral Presentation for next class…

- Oral Presentation Format
  - Time limit: 3 minutes (max.), followed by 1 minute of Q & A.
  - Domain & Motivation: background motivation for the project. Domain of interest, study conducted, main findings.
  - Project: What are the main use cases of your application/ service? What are the components involved in making it.
  - Project timeline / Gantt chart.
  - List of deliverables: what you can realistically finish by the end of the semester.
  - Project team roles.
- Put slides in a Google presentation and link it from this Google Doc: http://goo.gl/aSrTQQ Set sharing permissions on your slides to "anyone with the link"

Hello World

- Create an application that displays “Hello 21w. 789” and has a button. When this button is pressed, the text should change to “Goodbye 21w. 789”
- Install this application on your group development phone and show it in class at the end of your presentation

Break

- Meet in 10 minutes in your section rooms