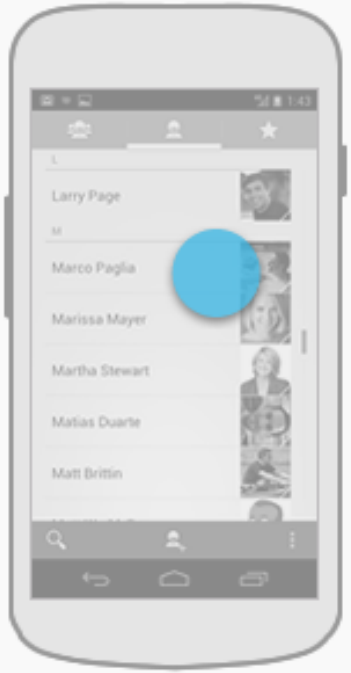
CS371m - Mobile Computing

Gestures

Common Gestures



Touch

Triggers the default functionality for a given item.

Action
Press, lift

Long press

Enters data selection mode. Allows you to select one or more items in a view and act upon the data using a contextual action bar. Avoid using long press for showing contextual menus.

Action
Press, wait, lift

Swipe Or Scroll

Scrolls overflowing content, or navigates between views in the same hierarchy.

Action
Press, move, lift

Common Gestures

Drag

Rearranges data within a view, or moves data into a container (e.g. folders on Home Screen).

Action
Long press, move, lift

Double touch

Zooms into content. Also used as a secondary gesture for text selection.

Action
Two touches in quick succession

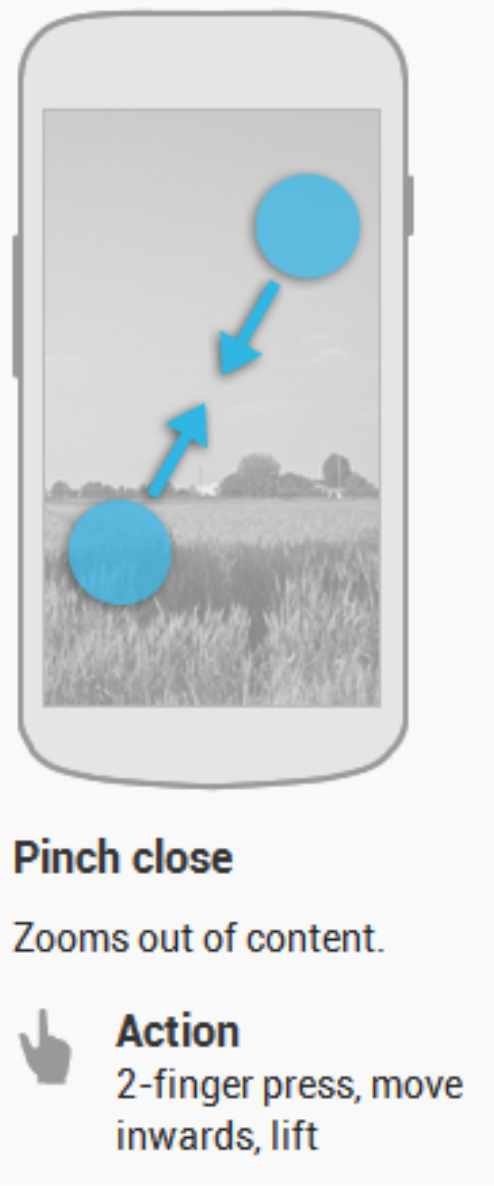
Pinch open

Zooms into content.

Action
2-finger press, move outwards, lift

http://developer.android.com/design/patterns/gestures.html

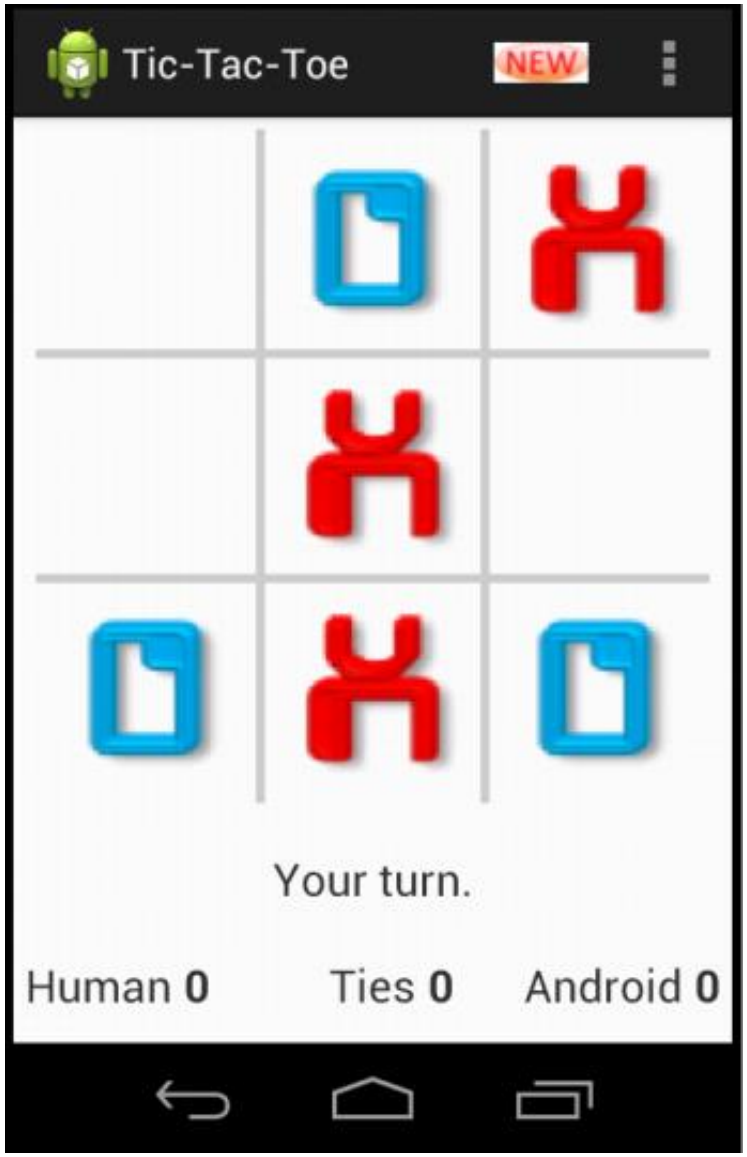
Common Gestures



- Fling or flick gesture: similar to swipe or drag
- scroll/swipe/drag
  - user presses then moves finger in steady motion before lifting finger
- fling or flick
  - user presses then moves finger in an accelerating motion before lifting

Dealing With Gestures

- To handle simple touch events create View.OnTouchListener for view
- Example from tic-tac-toe tutorial, screen press leads to player moving if it is their turn and they touch an open square



onTouchEvent

- passed a MotionEvent object with a **large** amount of data
- in tic tac toe tutorial you only use location of event (x and y)

final float	getHistoricalOrientation (int pos) `getHistoricalOrientation(int, int)` for the first pointer i
final void	getHistoricalPointerCoords (int pointerIndex, int pos, MotionEvent Populates a `MotionEvent.PointerCoords` object with historic
final float	getHistoricalPressure (int pos) `getHistoricalPressure(int, int)` for the first pointer inde
final float	getHistoricalPressure (int pointerIndex, int pos) Returns a historical pressure coordinate, as per `getPressure(i`
final float	getHistoricalSize (int pos) `getHistoricalSize(int, int)` for the first pointer index (ma
final float	getHistoricalSize (int pointerIndex, int pos) Returns a historical size coordinate, as per `getSize(int)`, that
final float	getHistoricalToolMajor (int pointerIndex, int pos) Returns a historical tool major axis coordinate, as per `getToolM`
final float	getHistoricalToolMajor (int pos) `getHistoricalToolMajor(int, int)` for the first pointer ind
final float	getHistoricalToolMinor (int pointerIndex, int pos) Returns a historical tool minor axis coordinate, as per `getToolM`
final float	getHistoricalToolMinor (int pos) `getHistoricalToolMinor(int, int)` for the first pointer ind
final float	getHistoricalTouchMajor (int pointerIndex, int pos) Returns a historical touch major axis coordinate, as per `getTouc`
final float	getHistoricalTouchMajor (int pos) `getHistoricalTouchMajor(int, int)` for the first pointer in
final float	getHistoricalTouchMinor (int pointerIndex, int pos)

MotionEvent

Public Methods	
abstract boolean	onTouch (View v, MotionEvent event) Called when a touch event is dispatched to a view.

- Example of the astonishing amount of data packed into the **motionEvent** object

public final float getSize (int pointerIndex) Added in API level 5

Returns a scaled value of the approximate size for the given pointer *index* (use `getPointerId(int)` to find the pointer identifier for this index). This represents some approximation of the area of the screen being pressed; the actual value in pixels corresponding to the touch is normalized with the device specific range of values and scaled to a value between 0 and 1. The value of size can be used to determine fat touch events.

Parameters

pointerIndex	Raw index of pointer to retrieve. Value may be from 0 (the first pointer that is down) to `getPointerCount()`-1.

Other View Listeners

- View also has ability to listen for long clicks and drags
- In addition to View.OnTouchListener
- View.OnLongClickListener
- View.OnDragListener

Handling Common Gestures

- Instead of trying to decode gestures from the MotionEvent passed to the on touch method …
- Use the GestureDetector class
- Add a GestureDetector object to View
- override View.onTouchEvent method to pass MotionEvent on to the GestureDetector.onTouchEvent method

Handling Common Gestures

- create a GestureDetector.OnGestureListener (several gestures)
- or a GestureDetector.DoubleTapListener (more gestures) and register it with the GesturerDetector

GestureDetector.OnGestureListener

Public Methods	
abstract boolean	onDown(MotionEvent e) Notified when a tap occurs with the down `MotionEvent` that triggered it.
abstract boolean	onFling(MotionEvent e1, MotionEvent e2, float velocityX, float velocityY) Notified of a fling event when it occurs with the initial on down `MotionEvent` and the matching up `MotionEvent`.
abstract void	onLongPress(MotionEvent e) Notified when a long press occurs with the initial on down `MotionEvent` that trigged it.
abstract boolean	onScroll(MotionEvent e1, MotionEvent e2, float distanceX, float distanceY) Notified when a scroll occurs with the initial on down `MotionEvent` and the current move `MotionEvent`.
abstract void	onShowPress(MotionEvent e) The user has performed a down `MotionEvent` and not performed a move or up yet.
abstract boolean	onSingleTapUp(MotionEvent e) Notified when a tap occurs with the up `MotionEvent` that triggered it.

GestureDetector.DoubleTapListener

Summary

Public Methods	
abstract boolean	onDoubleTap (MotionEvent e) Notified when a double-tap occurs.
abstract boolean	onDoubleTapEvent (MotionEvent e) Notified when an event within a double-tap gesture occurs, including the down, move, and up events.
abstract boolean	onSingleTapConfirmed (MotionEvent e) Notified when a single-tap occurs.

Clicker Question

- In Java, if a class implements an interface, how many methods declared in the interface does the class have to implement?

A. All of them

B. Some of them

C. None of them

D. It depends

Adapter Classes

- OOP Pattern
- Create a class that implements methods of interface with minimal (or no) functionality
- Standard Java Example
- Interfaces for MouseListener(5), MouseWheelListener(1), and MouseMotionListener(3)
- MouseAdapter class implements all three interfaces with empty methods
- extend MouseAdapter and add functionality for events you care about.

GestureDetector.SimpleOnGestureListener

- Implements all methods of GestureDetector.OnGestureListener and GestureDetector.DoubleTapListener
- Does nothing but return false for all the methods
- Extend this class and add more meaningful behavior

Simple Gesture Demo

- App that listens for simple gestures
- update lower TextView in call back methods

Gesture Demo

```
public class GesturesDemo extends Activity
        implements GestureDetector.OnGestureListener,
        GestureDetector.OnDoubleTapListener  {

    private TextView gestureType;
    private GestureDetectorCompat gestureDetect;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_gestures_demo);
        gestureType = (TextView) findViewById(R.id.gesture_type);
        gestureDetect = new GestureDetectorCompat(this, this); //
        gestureDetect.setIsLongpressEnabled(true);
    }
```

Gesture Demo

- Simply pass event on to the GestureDetectorCompat object
  - it will call back methods

```
@Override
public boolean onTouchEvent(MotionEvent event) {
    gestureDetect.onTouchEvent(event);
    return true;
}
```

Callback Methods for OnGestureListener

```
@Override
public boolean onDown(MotionEvent e) {
    gestureType.setText("DOWN");
    return true;
}

@Override
public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX,
        float velocityY) {
    gestureType.setText("FLING");
    return true;
}

@Override
public void onLongPress(MotionEvent e) {
    gestureType.setText("LONG PRESS");

}
```

Callback Methods for OnGestureListener

```
@Override
public boolean onScroll (MotionEvent e1, MotionEvent e2,
        float distanceX, float distanceY) {
        gestureType.setText("SCROLL");
        return true;
}

@Override
public void onShowPress(MotionEvent e) {
    gestureType.setText("SHOW PRESS");
}

@Override
public boolean onSingleTapUp(MotionEvent e) {
    gestureType.setText("SINGLE TAP UP");
    return true;
}
```

Callback Methods for DoubleTapListener

```
@Override
public boolean onDoubleTap(MotionEvent arg0) {
    gestureType.setText("DOUBLE TAP");
    return true;
}

@Override
public boolean onDoubleTapEvent(MotionEvent arg0) {
    gestureType.setText("DOUBLE TAP");
    return true;
}

@Override
public boolean onSingleTapConfirmed(MotionEvent arg0) {
    gestureType.setText("SINGLE TAP CONFIRMED");
    return true;
}
```

Multi Touch Gestures

- Multiple fingers (pointers) touch screen at same time
- Still handled via MotionEvents
- each pointer (finger) has a MotionEvent
- track via index (in array of MotionEvents) or ID
- MotionEvent object sent to onTouch contains number of "pointers" involved

Displaying Multitouch data

- static methods from MotionEventCompat class

```
@Override
public boolean onTouchEvent(MotionEvent event) {
    if(event.getPointerCount() > 1) {
        gestureType.setText("MULTI TOUCH\nEVENT");
        int action = MotionEventCompat.getActionMasked(event);
        gestureType.append("\n" + actionToString(action));
        int index = MotionEventCompat.getActionIndex(event);
        gestureType.append("\nPointer index: "+ index);
    }
    else
        gestureDetect.onTouchEvent(event);

    return true;
}
```

Scale Gestures

- ScaleGestureDetector class from Api level 8 (API 2.2)
- pinch to zoom in or out
- out -> scale up
- in -> scale down

Pinch open

Zooms into content.

Action
2-finger press, move outwards, lift

Scale Gestures

- Create class that implements ScaleGestureDetector.OnScaleGestureListener
- OR create class that extends ScaleGestureDetector.SimpleOnScaleGestureListener
  - adapter class
  - implements methods from OnScaleGestureListener with dummy methods
  - override only the methods you care about
- Create a ScaleGestureDetector with listener
- pass Motion events from onTouch

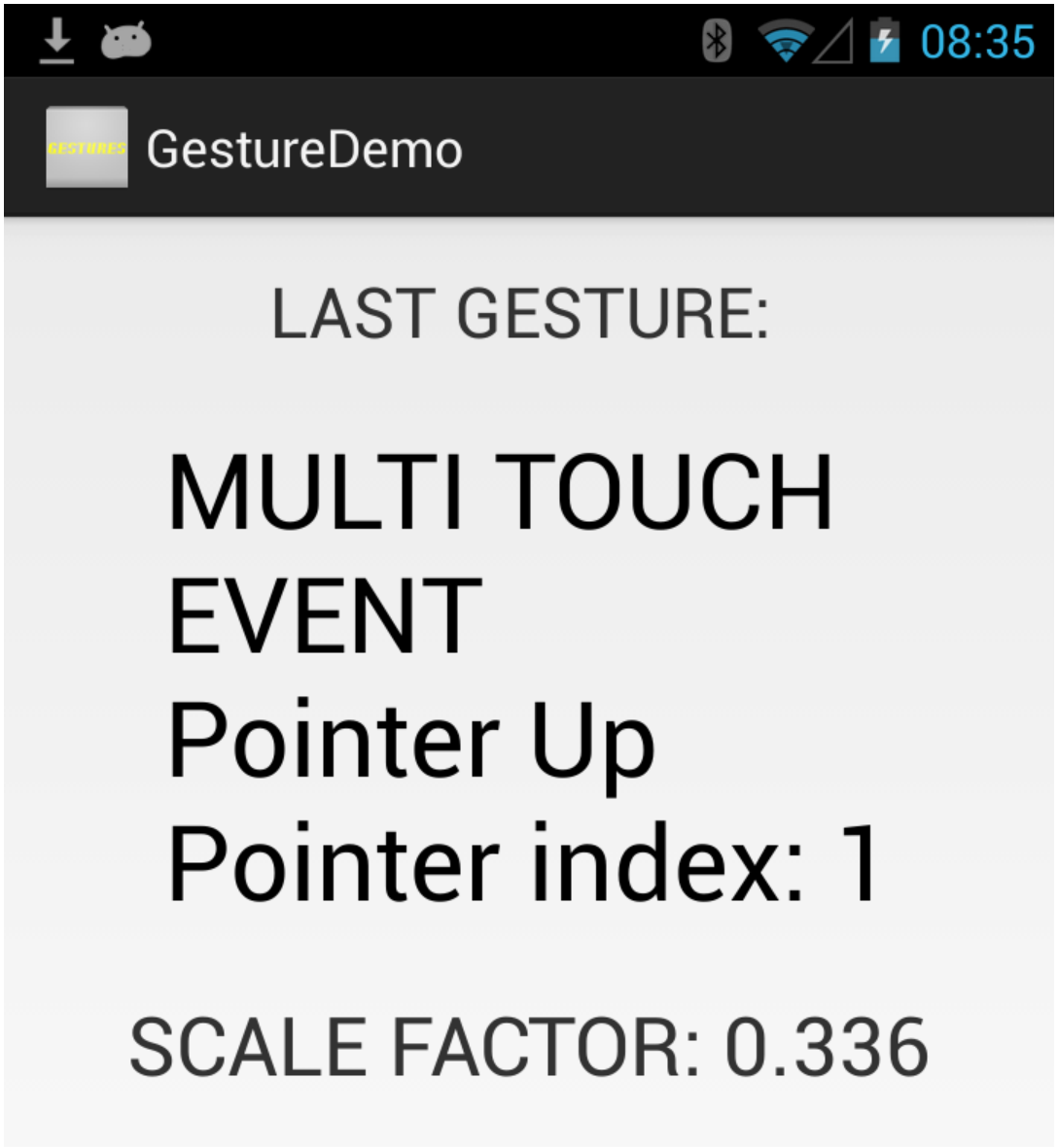
Scaling Example

- listener updates overall scale factor
- shows current scale factor in TextView

```
// from http://developer.android.com/training/gestures/scale.html
private class MyScaleListener
        extends ScaleGestureDetector.SimpleOnScaleGestureListener {
    @Override
    public boolean onScale(ScaleGestureDetector detector) {
        // Log.d("GESTURE DEMO", "Scale factor: " + detector.getScaleFacto
        scaleFactor *= detector.getScaleFactor();
        // Log.d("GESTURE DEMO", "Scale factor calculated: " + scaleFactor
        scaleFactor = Math.max(0.001f, Math.min(scaleFactor, 10.0f));
        // Log.d("GESTURE DEMO", "Scale factor clamped: " + scaleFactor);
        double scaleFactorDisplay = ((int) (scaleFactor * 1000)) / 1000.0
        scaleFactorTV.setText("SCALE FACTOR: " + scaleFactorDisplay);
        return true;
    }

}
```

Scale Example



Drag Gestures

- Similar to handling Scale gestures
- Implement View.OnDragListener
  - one method, onDrag(View v, DragEvent de)
- Drag event phases:
  - start
  - continuing
  - dropped
  - ended

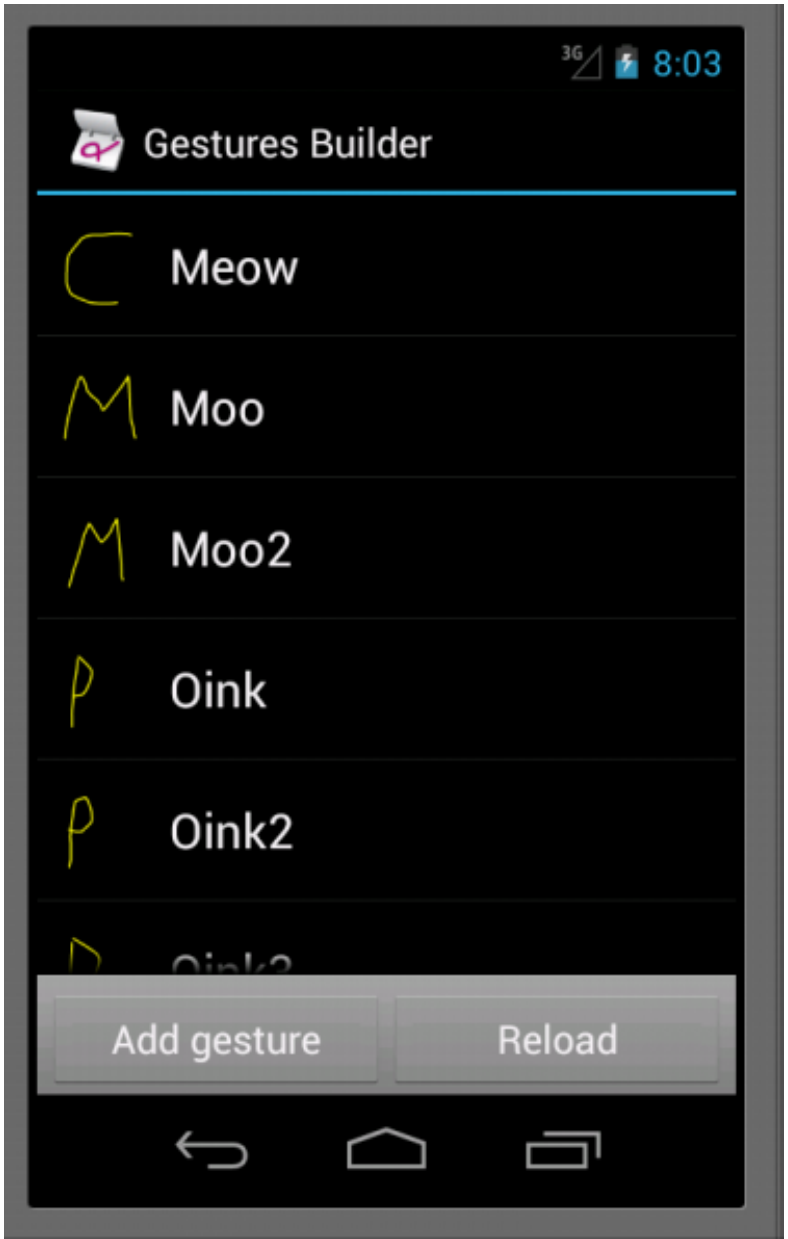
COMPLEX GESTURES

Complex Gestures

- Non standard gestures require lots of code to recognize
- Android 1.6 introduced new APIs to store, load, draw, and recognize gestures
- Gesture Builder app on emulator
  - emulator must include virtual SD card
  - allows creating set of gestures for your application
  - limited success with jelly bean emulators
  - App on class GitHub repo

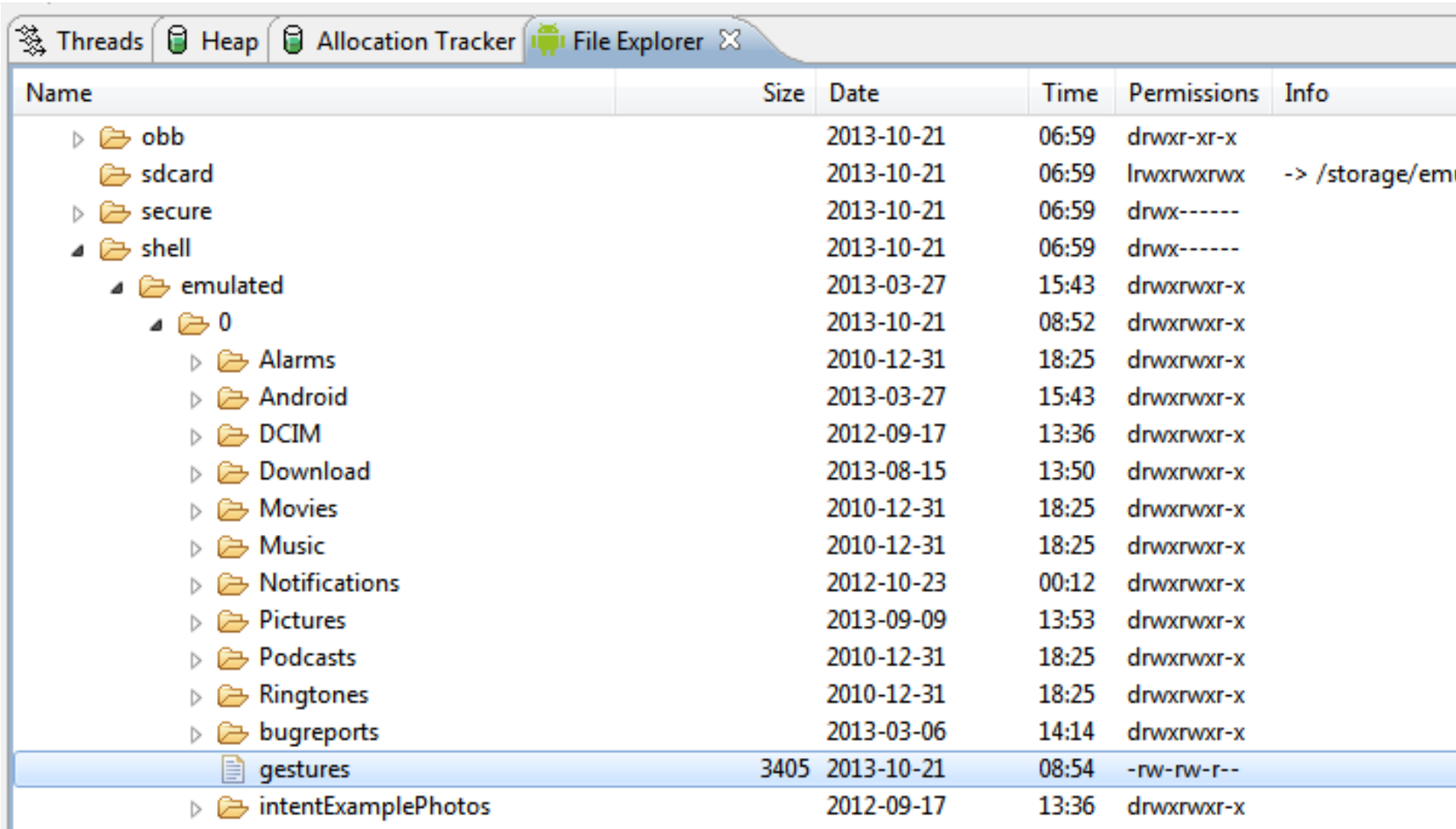
Complex Gestures

- Each gesture associated with name
- Limited to single pointer
- multiple gestures can have same name
  - variations on same gesture, better chance of recognizing
- Move gestures from emulator to application res/raw folder



Gesture Data File

- DDMS file explorer



Complex Gestures

- Recognizing gestures via a GestureOverlayView
- simple drawing board on top of view that shows and records user gestures
- When gesture complete GestureLibrary queried to see if gesture is recognized
- Predictions of entered gesture and those in the library

Animal Sounds App

Predictions

```
AnimalSounds      prediction score: 5.020522997579021, name: Oink2
AnimalSounds      prediction score: 11.698475110815773, name: Meow
AnimalSounds      prediction score: 1.4253241939996129, name: Oink3
AnimalSounds      prediction score: 1.708742452226205, name: Oink
AnimalSounds      prediction score: 1.7788133409813087, name: Oink
Choreographer     Skipped 30 frames!  The application may be doing
AnimalSounds      prediction score: 1.5979739128902553, name: Moo2
AnimalSounds      prediction score: 1.1312601585038455, name: Moo
AnimalSounds      prediction score: 1.733056893468628, name: Meow
AnimalSounds      prediction score: 0.7404827760194891, name: Moo
AnimalSounds      prediction score: 1.009555907026495, name: Moo2
AnimalSounds      prediction score: 1.408645869375701, name: Moo2
AnimalSounds      prediction score: 2.048106505538496, name: Oink3
AnimalSounds      prediction score: 3.078060118728627, name: Meow
AnimalSounds      prediction score: 2.932816689691991, name: Meow
AnimalSounds      prediction score: 1.792527999275177, name: Meow
AnimalSounds      prediction score: 1.8169176605869966, name: Oink3
AnimalSounds      prediction score: 0.7143366373124087, name: Moo
AnimalSounds      prediction score: 1.5232821190754195, name: Oink
Choreographer     Skipped 32 frames!  The application may be doing
AnimalSounds      prediction score: 0.785716727687679, name: Moo
```

onCreate

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    mLibrary = GestureLibraries.fromRawResource(this, R.raw.gestures);
    if (!mLibrary.load()) {
        finish();
    }

    GestureOverlayView gestures
            = (GestureOverlayView) findViewById(R.id.gestures);
    gestures.addOnGesturePerformedListener(mGestureListener);

    createSoundPool();
}
```

Listener

```
@Override
public void onGesturePerformed(GestureOverlayView overlay,
        Gesture gesture) {
    // from http://android-developers.blogspot.com/2009/10/gestures-on-android-

    ArrayList<Prediction> predictions = mLibrary.recognize(gesture);

    // We want at least one prediction
    if (predictions.size() > 0) {
        Prediction prediction = predictions.get(0);

        Log.d(TAG, "prediction score: " + prediction.score + ", name: " + predi

        // We want at least some confidence in the result
        if (prediction.score > 3.0) {
            String name = prediction.name;
            if(name.contains("Moo"))
                mSounds.play(mSoundIDMap.get("Moo"), 1, 1, 1, 0, 1);
            else if(name.contains("Oink"))
                mSounds.play(mSoundIDMap.get("Oink"), 1, 1, 1, 0, 1);
            else if(name.contains("Meow"))
                mSounds.play(mSoundIDMap.get("Meow"), 1, 1, 1, 0, 1);
        }
    }
}
```