Special Menus

- Two special application menus
  - options menu
  - context menu

- Options menu replaced by action bar (API 11)
OptionsMenu

• User presses Menu Button
• Activities onCreateOptionsMenu method is called

```java
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    super.onCreateOptionsMenu(menu);
    MenuInflater inflator = getMenuInflater();
    inflator.inflate(R.menu.options_menu, menu);
    return true;
}
```

• In example options_menu.xml in res/menu folder
OptionsMenu

• Alternate creation of OptionsMenu
• add item to menu programmatically

```java
menu.add("Big About")
    .setIcon(R.drawable.about)
    .setIntent(new Intent(this, AboutActivity.class));
```

• chained method calls
SubMenus

• Option on Menu may be creation of a SubMenu

• In XML nest menu inside menu or programmatically by adding SubMenus to Menu in `onCreateOptionsMenu` method

```xml
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item android:id="@+id/file"
       android:title="@string/file" />
  <!-- "file" submenu -->
  <menu>
    <item android:id="@+id/create_new"
          android:title="@string/create_new" />
    <item android:id="@+id/open"
          android:title="@string/open" />
  </menu>
</item>
</menu>
```
Menu Options Selected

• if Menu Option is another Activity it is launched when Menu button pressed
  – The Big About in previous example

• For other items
  – onOptionsItemSelected(MenuItem item)

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    super.onOptionsItemSelected(item);
    switch (item.getItemId()) {
    case R.id.new_game:
        startNewGame();
        return true;
    case R.id.ai_difficulty:
        showDialog(DIALOG_DIFFICULTY_ID);
        return true;
    }
ACTION BAR
ActionBar

• Introduced in Android 3.0
  – Honeycomb, tablet only
• 4.0, Ice Cream Sandwich, tablet and phones
• "The action bar is a window feature that identifies the application and user location, and provides user actions and navigation modes"
• http://developer.android.com/guide/topics/ui/actionbar.html
Purpose of ActionBar

• identification
• navigation
• actions
ActionBar

- ActionBar items declared in menu.xml

```xml
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item
    android:id="@+id/new_game"
    android:icon="@drawable/new_game"
    android:title="New Game"
    android:showAsAction="ifRoom|withText"/>
</menu>
```
ActionBar

• If menu items declared in xml, added to menu in order they appear
• Extra items brought up with overflow button
Navigation Tabs

• Used to switch between fragments

ContextMenu

• pre 3.0, aka Floating Menus
• subtype of Menu
• display when a long press is performed on a View
  – Activity is a descendant of View
  – Activity may be broken up into multiple views
• implement `onCreateContextMenu` method
• must call `registerForContextMenu` method and pass View
ContextMenu

- From Tip Calculator
- Long press on total amount EditText
- Default behavior for EditText
- Nothing added in TipCalculator to create this
Contextual Action Mode

- 3.0 and later

http://developer.android.com/guide/topics/ui/menus.html#CAB
STYLES
Styles

- Defined in XML file
- res/values/style
- similar to a cascading style sheet as used in html
- group layout attributes in a style and apply to various View objects (TextView, EditText, Button)
Sample Styles, in styles.xml

```xml
<style name="sample1">
  <item name="android:textSize">20pt</item>
  <item name="android:textColor">@color/Orange</item>
  <item name="android:textStyle">bold</item>
  <item name="android:gravity">center</item>
  <item name="android:padding">10dp</item>
</style>

<style name="sample2">
  <item name="android:textSize">8pt</item>
  <item name="android:textColor">@color/AliceBlue</item>
  <item name="android:textStyle">italic</item>
  <item name="android:gravity">right</item>
  <item name="android:padding">2dp</item>
</style>
```
Apply Style - in main xml

```xml
<TextView
    android:id="@+id/textView1"
    style="@style/sample2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="field number 1" />

<EditText
    android:id="@+id/editText1"
    style="@style/sample1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:inputType="textCapWords"
    android:text="First Edit Text" />

<TextView
    android:id="@+id/textView2"
    style="@style/sample2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="field number 2" />
```
Result of Styles

• can override elements of style
  – bottom edit text overrides color
• one style can inherit from another
• use UI editor to create view and then extract to style
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, move, lift

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

**Action**
Press, wait, lift
Common Gestures

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

- **Double touch**
  Zooms into content. Also used as a secondary gesture for text selection.

- **Pinch open**
  Zooms into content.

- **Action**
  Long press, move, lift

- **Action**
  Two touches in quick succession

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 a 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 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 used location of event (x and y)
• View also has ability to listener for long clicks and drags

```java
final float getHistoricalOrientation (int pos)
    getHistoricalOrientation (int, int) for the first pointer index (ma
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 index (ma
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 index (ma
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 index
final float getHistoricalTouchMajor (int pointerIndex, int pos)
    Returns a historical touch major axis coordinate, as per getTouch
final float getHistoricalTouchMajor (int pos)
    getHistoricalTouchMajor (int, int) for the first pointer index
final float getHistoricalTouchMinor (int pointerIndex, int pos)
    Returns a historical touch minor axis coordinate, as per getTouch
final float getHistoricalTouchMinor (int pos)
    getHistoricalTouchMinor (int, int) for the first pointer index
```
Handling Common Gestures

• Instead of trying to decode gestures from the MotionEvent passed to onTouch ...
• 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.SimpleOnGestureListener (more gestures) and register it with the GestureDetector

• callback methods for onLongPress, onScroll, onFling, onSingleTapConfirmed, others
Simple Gesture Demo

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

• Recall, Graphics Demo overrode onTouchEvent and displayed new random grid when down event occurred

• Here, simply pass event on to the GestureDetectorCompat object
  – it will call back methods

```java
@Override
public boolean onTouchEvent(MotionEvent event) {
    gestureDetect.onTouchEvent(event);
    return true;
}
```
Callback Methods for OnGestureListener

```java
@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");
}
```
@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

```java
@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
• Handled via MotionEvent
• 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

```java
@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 our out
- out -> scale up
- in -> scale down
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 ScalerGestureDetector with listener
• pass Motion events from onTouch
Scaling Example

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

```java
// 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.getScaleFactor();
        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

LAST GESTURE:

MULTI TOUCH EVENT
Pointer Up
Pointer index: 1

SCALE FACTOR: 0.336
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

• 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
GestureBuilder on Devices

- GestureBuilder app included on emulators
- Possible to put on a real device
- Create project from sample

File -> New Project -> Android
Select Build Target

- Gesture Builder in SDK examples for multiple API levels
- I tried API level 10
Finish Project Create

• After project created, install on real device
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 between entered gesture and those in the library
Animal Sounds App

Draw M to Moo
Draw P to Oink
Draw C to Meow

Draw M to Moo
Draw P to Oink
Draw C to Meow
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: Meow
AnimalSounds  prediction score: 1.0095559070264957, 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: 0.7857167276876791, name: Moo
```java
@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();
}
```
@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: " + prediction.name);

        // 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);
        }
    }
}