Wallpaper, Widgets & Wear

Glanceable displays on Android

Elizabeth Mezias

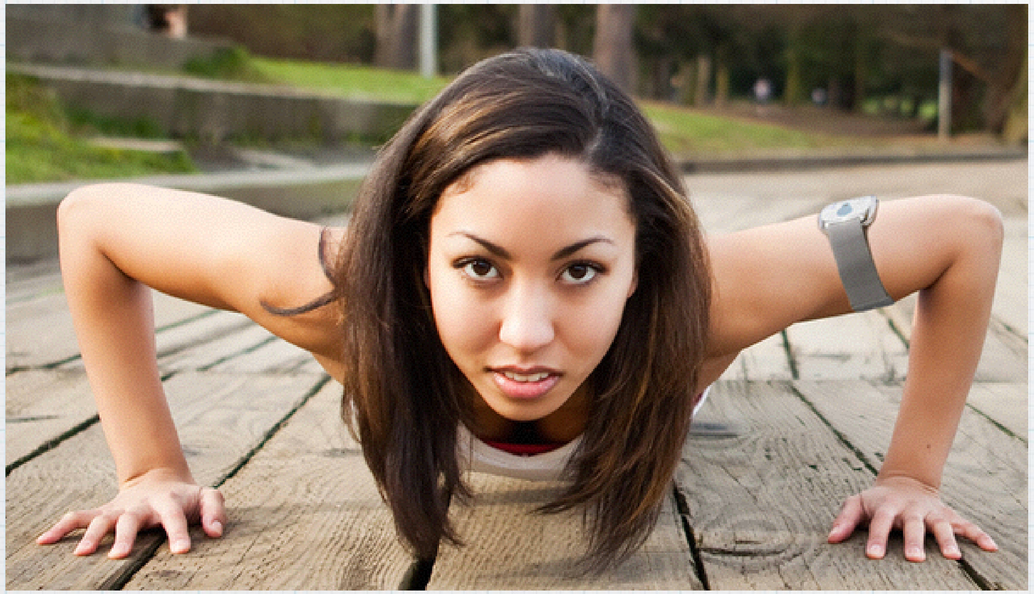
Mezcode means mobile

Glanceable displays on Android

Elizabeth Mezias
Mobile Systems Architect

Mezcode means mobile

Other Wearables? Think Design! new paradigms…



Body Media

necomimi™
Brainwave Cat Ears

Is the phone a wearable?

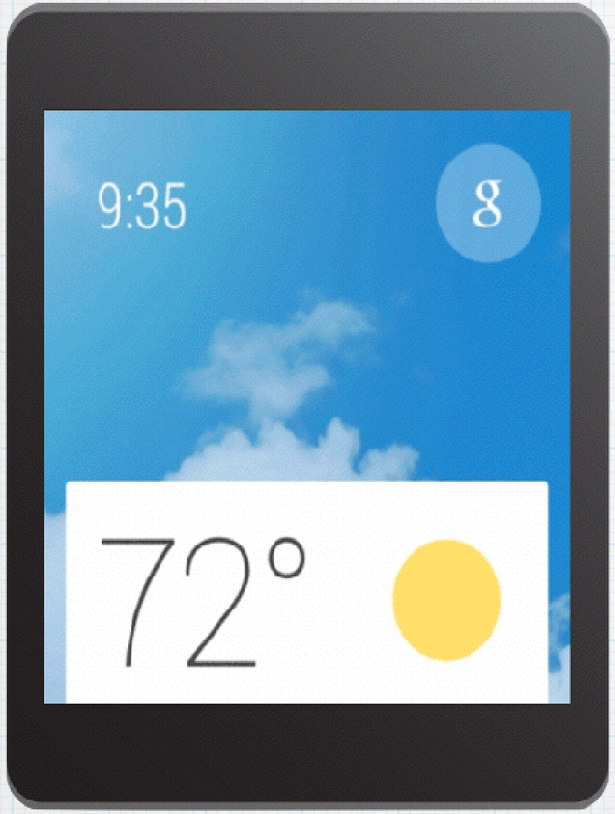
- Lumo Back
  - Worn around the midsection
  - Sensors measure posture, steps, sitting time, and sleep
  - Gentle, haptic input to stop a slouch
- Body Media
  - Armband records temp, heart rate, calories and sleep
  - Proven for weight loss, users win on the portal
- Neurosky
  - Brainwave reader (game input, maybe applied to assist autistics)

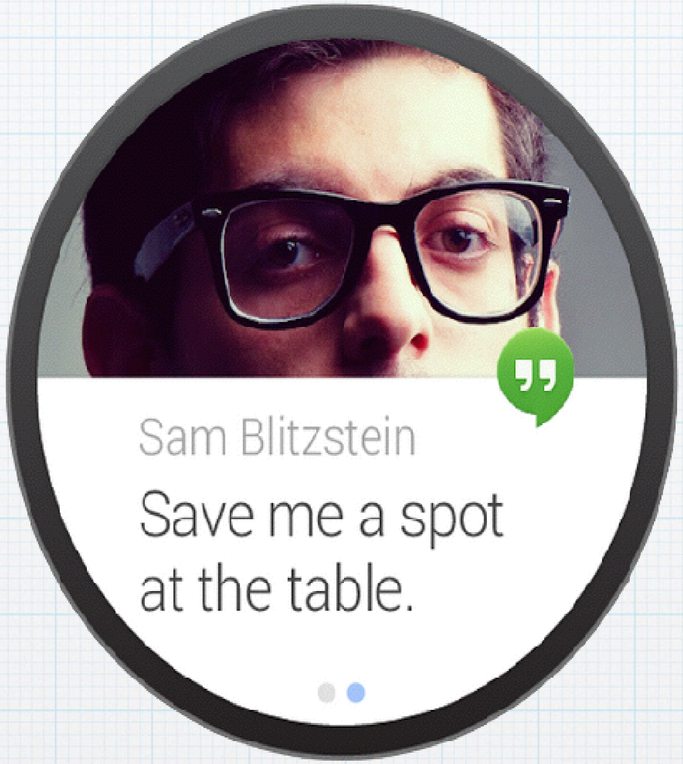
Big names making watches
What's next?

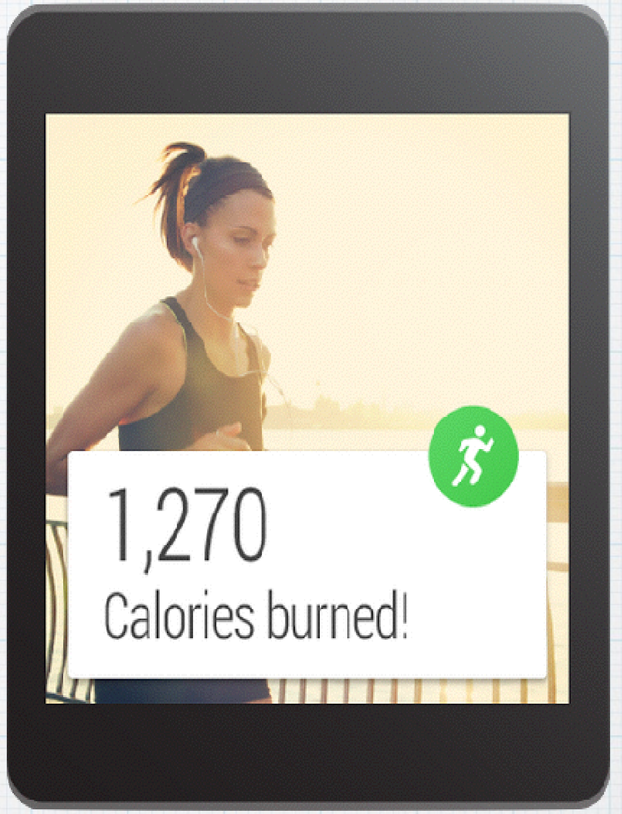
Brillo announced at Google I/O

Be practical, Be considerate

* Bridged notifications –> target Android 4.3, done
* Wear adds-on to apps, complements the phone
* Stay relevant and specific with time and location triggers
* Glanceable is short, sharp, immediate
* Zero to 2 touch interactions are simple, and fast
* Be helpful – anticipate, be efficient, respectful, & responsive







What can it do?

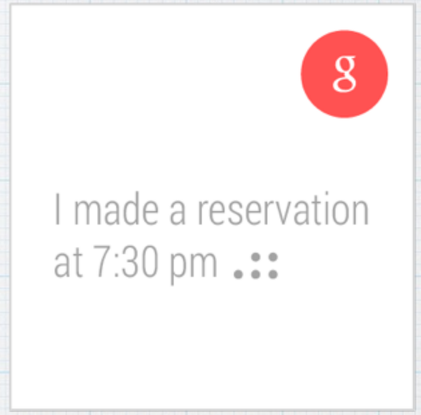
OK Google Dick Tracy hangouts, sensor accesss – GPS & heart rate

More rules…

- Use context, meaning location and time
- Sensor data from the watch OR the phone can show at a glance
- Consider QS, what can be measured?
  Give meaning to that measurement
- Don’t interrupt, assume the watch is muted
- A watch face without time - not much of the user's time...

3 Types of App (for Wear)

1. Notifications, in-line responses (speech, tap)



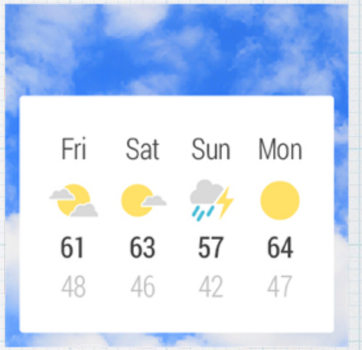
2. On-Board Apps, local sensors, speech, usually end in a request to the phone

   2.a. Watch Faces

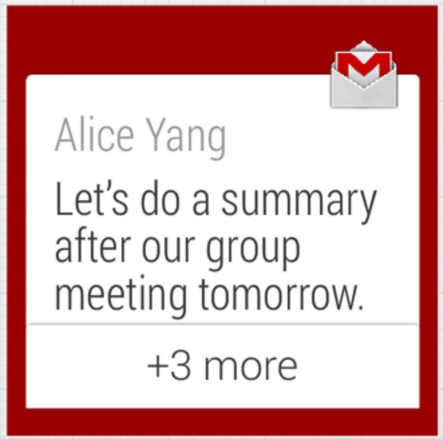
3. Data sync, leverage on-board sensors
   ...sync with the phone and or a desktop portal

Bridged Notifications

* NotificationCompat & NotificationManagerCompat
* WearableExtender for Action Cards (3)
* A “BigView” is a 2 parter, peek with pull up
* Hide the icon (make more room for text)
* RemoteInput & RemoteInput.Builder classes get speech
  * Notification responses get one interpretation
  * On-board apps get an array of possible text
* Stacking aggregates -> setGroup creates a scroll



☹ more text?



Voice Actions

1. Declare “Start…” activity in the manifest
2. Get a ride ”OK Google, get me a taxi" (also call me a car)
   com.google.android.gms.actions.RESERVE_TAXI_RESERVATION
3. Take a note “OK Google, take a note"
   android.intent.action.SEND This action has a Category, text is sent in as Intent Extras
   com.google.android.voicesearch.SELF_NOTE, android.content.Intent.EXTRA_TEXT
4. Set alarm "OK Google, set an alarm for 8 AM"
   android.intent.action.SET_ALARM
   Extras for this action are not required, hour and minute integers
5. Set timer "Ok Google, set a timer for 10 minutes"
   android.intent.action.SET_TIMER
   An integer to tell the duration of the timer android.provider.AlarmClock.EXTRA_LENGTH

Wear with Google Fit
(more Voice Actions)

6. Start stopwatch "Ok Google, start stopwatch"
   com.google.android.wearable.action.STOPWATCH

7. Record Exercise "OK Google, start/stop cycling", running, cycling
   Action is vnd.google.fitness.TRACK, extra ActiveActionStatus & CompletedActionStatus

   A. (Cycling Mime Type) vnd.google.fitness.activity/biking

   B. (Running, Workout Mime Type) vnd.google.fitness.activity/running ...other

8. Show Heart Rate/Step Count "OK Google, what's my heart rate/step count?"
   Action is vnd.google.fitness.VIEW

   A. vnd.google.fitness.data_type/com.google.heart_rate.bpm (Mime Types*)

   B. vnd.google.fitness.data_type/com.google.step_count.cumulative

Home Screen Widgets

Live Widgets
Collections Lists & Stacks, Shortcuts
Part 2

Mezcode means mobile

Demo?

* A completely new design and UX is required
* Not immersive, not interactive, apps time out
* Apps are using custom gestures for launch
* No webkit, print, backup, appwidget, usb classes

Meter Widgets

Stanford Medical School intervention

Accelerometer data

Minutes of “mvpa” and “sedentary” hours

Resets @midnight

Always-on service

(Notice the beloved calendar next door)

Mezcode means mobile

How to do it:

* Two classes and the methods to override...

1. RemoteViewsFactory - bind Collection data to widgets
    * getViewAt() - inflate a layout
    * onDataSetChanged() - show the collection
2. AppWidgetProvider - respond to the user, timed updates
    * onReceive(), like any Broadcast Receiver
    * onUpdate(), code runs on time, updatePeriodMillis

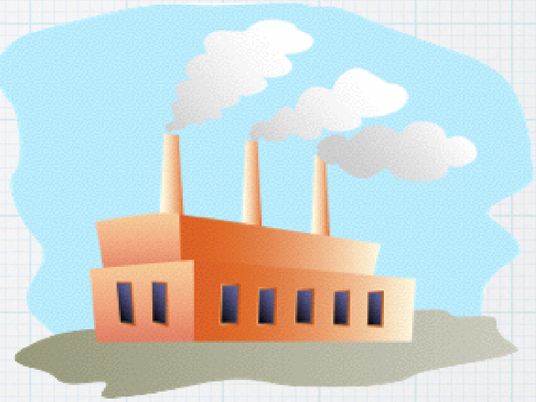
Manifest.xml

* The Homescreen app gets a Widget's views from the Manifest

1. PROVIDER Content provider, e.g. WeatherDataProvider
    * Binds a collection of data to the views of a widget
2. RECEIVER - AppWidgetProvider class declaration, e.g. WeatherWidget
    * Set the intent filter action - APPWIDGET_UPDATE
    * appwidget-provider xml file is meta-data with the content provider
3. SERVICE - RemoteViewsFactory is a service, e.g. WeatherWidgetService
    * Set BIND_REMOTEVIEW for the homescreen to see the data

1
```
<!-- The content provider serving the (fake) weather data -->
<provider android:name=".weatherwidget.WeatherDataProvider"
  android:authorities=".weatherwidget.provider" />
```
2
```
<!-- The widget provider -->
<receiver android:name=“.weatherwidget.WeatherWidget”>
  <intent-filter>
    <action android:name="android.appwidget.action.APPWIDGET_UPDATE" />
  </intent-filter>
  <!-- This specifies the widget provider info -->
  <meta-data android:name="android.appwidget.provider"
    android:resource="@xml/widgetinfo" />
  </receiver>
```
3
```
<!-- The service serving the RemoteViews to the collection widget -->
<service android:name=“.weatherwidget.WeatherWidgetService”
  android:permission="android.permission.BIND_REMOTEVIEWS"
  android:exported="false" />
```

#3 RemoteViewsService creates RemoteViewsFactory



* Override OnCreate to initialize
* Override onDataSetChanged to update the views
* Clean up in OnDestroy
* Examples are WeatherWidgetService, StackWidgetSvc

List Adapter vs. Remote Views, getView vs. getViewAt

```
public RemoteViews getViewAt(int position) {
    String city = "Unknown City";  int image_id = -1;
    if(mCursor.moveToPosition(position)) {
        city = mCursor.getString(mCursor.getColumnIndex(Columns.CITY));
        final int imgColIndex = mCursor.getColumnIndex(Columns.IMAGE);
        image_id = mCursor.getInt(imgColIndex);
  }
    RemoteViews rv = new RemoteViews(packageName,
        position % 2 == 0 ? R.layout.item1 : R.layout.item2);
    rv.setTextViewText(R.id.text, city);
    rv.setImageViewResource(R.id.image, image_id);
    rv.setOnClickFillInIntent(R.id.image, (new Intent())
        .putExtra(EXTRA_CITY_ID, city));
    return rv;
}
```

ACTION_APPWIDGET_UPDATE #2

3 parts to onUpdate()

1. Tell the widget to get the views from the adapter
   setRemoteAdapter(RemoteViewsService)
2. Bind an action to the Widget Views
   SetPendingIntentTemplate(), the response to a view on the widget
3. Code a touch response on any collection item
   SetOnClickPendingIntent(), set an activity or broadcast action

Pseudo-Code

```
@Override
public void onUpdate(Context ctx, AppWidgetManager appWidgetMgr, int[] appWidgetIds) {
    //set piece Soccer fans
    for (int i = 0; i < appWidgetIds.length; ++i) {
       appWidgetManager.updateAppWidget(appWidgetIds[i], rv);
    } //end for loop
    super.onUpdate(context, appWidgetManager, appWidgetIds);
} //end onUpdate
```

WidgetProvider - 2 Intents

First, set the RemoteViews Service

```
Intent i = new Intent…
i.putExtra(AppWidgetManager.EXTRA_APPWIDGET_ID, appWidgetIds[i]);
i.setData(Uri.parse(i.toUri(Intent.URI_INTENT_SCHEME)));
RemoteViews rv = new RemoteViews…
rv.setRemoteAdapter(R.id.stack_view, i);
// Set the service and the empty view, if appropriate
rv.setEmptyView(R.id.stack_view, R.id.empty_stack_view);
```

Intent 2 - fill in the blank

```
Intent viewItem = new Intent(context, ImageActivity.class);
rv.setPendingIntentTemplate(R.id.stack_view,
PendingIntent.getActivity(context, 0, viewItem,
    PendingIntent.FLAG_UPDATE_CURRENT));
```

set on the widget

```
rv.setOnClickPendingIntent(R.id.image,
PendingIntent.getActivity(context, 0, viewItem,
    PendingIntent.FLAG_UPDATE_CURRENT));
```

set on the list/stack item

AppWidgetProvider – touch it

```
public void onReceive(Context ctx, Intent intent) {
        final String action = intent.getAction();
        if(action.equals(REFRESH_ACTION)) {
                //Button pressed, update temperatures
        } else if(action.equals(CLICK_ACTION)) {
                //Touch on list, display photo in activity
        }
}
```

Broadcast Receiver

Set a pending intent on itself to fill in the blank


DEMO
PLEASE!

Wallpaper X

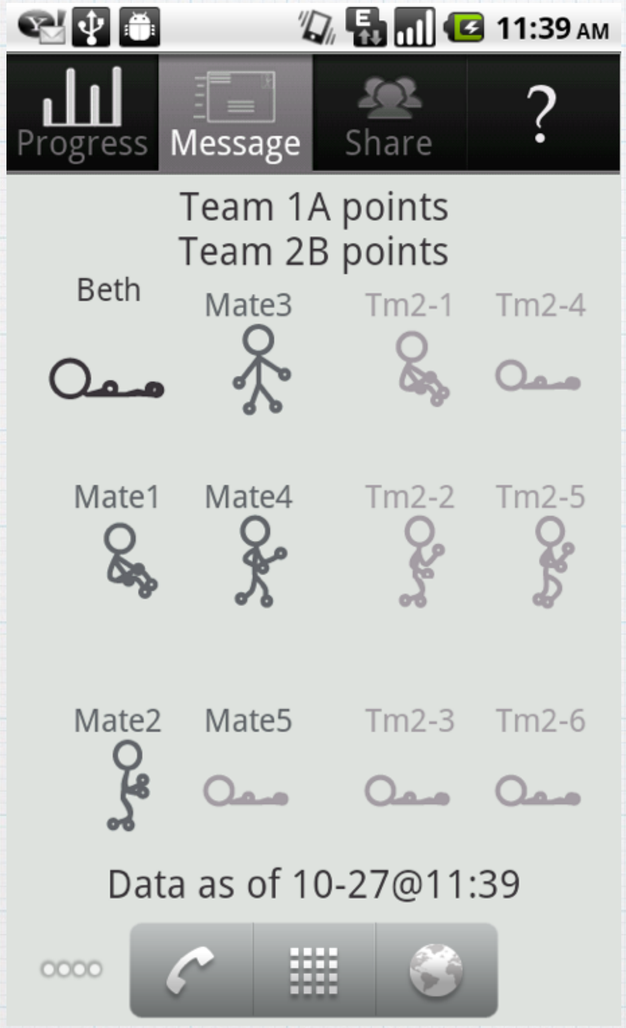
Live Wallpaper
More Glanceable Experiences…
Part 3

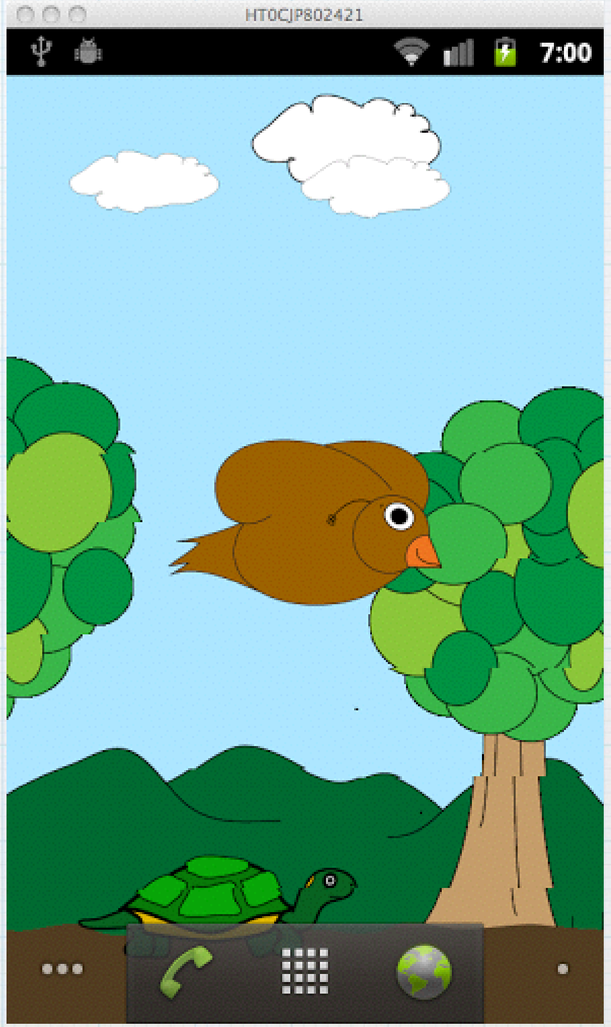
History of the term

* UbiFit
  Intel & University of Washington.
  Low-cost sensing, A.I.
  Bio-feedback - be more active.
  Activity-based
  ubiquitous computing

Their latest app – Sleep…

MILES - Stanford Medical School

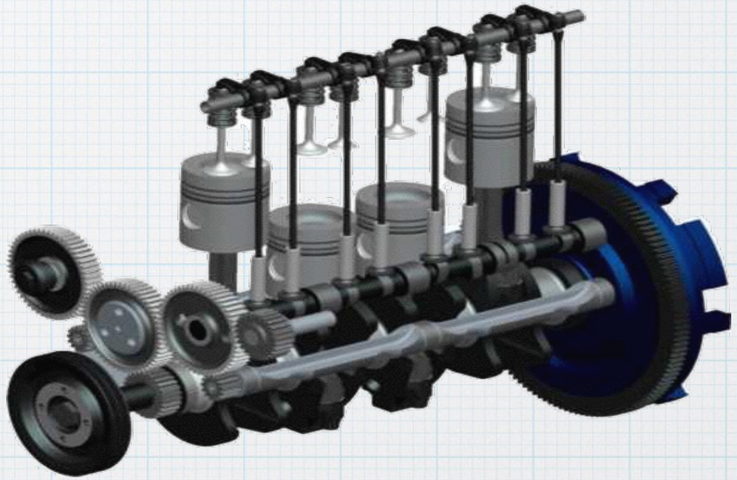




Mezcode means mobile

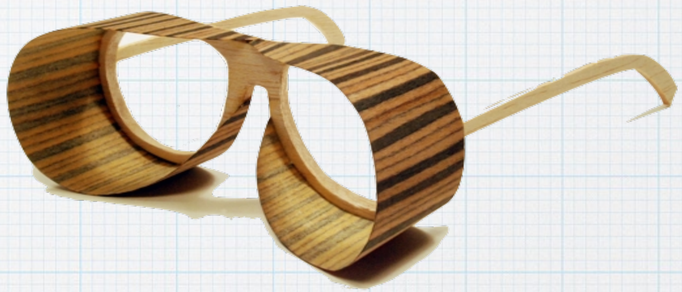
How-To

* WallpaperService nests an Engine with onCreateEngine( )
* Uses postDelayed on a Handler to draw, Canvas or view.draw()
  * onOffsetsChanged( ) - don’t go there
  * onVisibilityChanged( ) - stop the Handler
  * onSurfaceDestroyed( ) - cleanup, exit
  * onSurfaceChanged( ) - first call to draw
  * onTouchEvent( ) - user interaction (careful!)



```
public class MyWallpaperService extends WallpaperService {
    @Override
    public Engine onCreateEngine() {
            return new MyWallpaperEngine();
     }

    class MyWallpaperEngine extends Engine {
            Handler handler = new Handler();
            Runnable drawRunner = new Runnable() {
            @Override
                    public void run() {
                        draw();
                     }
              }; //end drawRunner

     public MyWallpaperEngine() { <constructor> }
              < Canvas drawing logic >
     }  //end MyWallpaperEngine class

  }//end service
```

```
@Override
public void onVisibilityChanged(boolean visible) {

  if(visible) {
    drawRunner.run();
  } else {
    handler.removeCallbacks(drawRunner);
  }
}
```



```
@Override
public void onTouchEvent(MotionEvent event) {
  //Each touch adds a circle
  //after some number, the list is cleared
  super.onTouchEvent(event);
  if(touchEnabled  && event.getAction() ==
            MotionEvent.ACTION_UP) {
    int x = Float.valueOf(event.getX()).intValue();
    int y = Float.valueOf(event.getY()).intValue();
    if(reset) {
      mNumber = 1;
      reset = false;
      touches.clear();
    } else {
      mNumber +=1;
      touches.add(new TouchPoint(x, y));
    Draw();
  }
}
```

MotionEvent.ACTION_UP, ACTION_DOWN, ACTION_MOVE, etc.

Links

* The MILES Study, Stanford University
  http://med.stanford.edu/miles
* MILES Apps, screen shot slide show, Mezcode site
  https://sites.google.com/site/mezcocorp/apps/miles
* Stanford results published
  http://goo.gl/1ntS8n
* Lars Vogel tutorial
  http://goo.gl/TSguD
* Mezcode, my portfolio site
  https://sites.google.com/site/mezcocorp/

Please take a moment to fill out the feedback form
Paper feedback forms are available in back

* eventmobi.com/adcboston