Android

Creating an Earthquake Widget

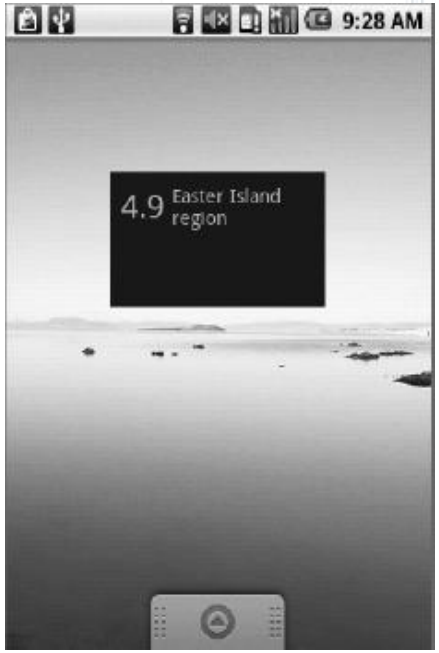
2010.05.28

Database Laboratory
Hanyang Univ.

EARTHQUAKE WIDGET

Creating an Earthquake Widget

- The following instructions show you how to create a new home-screen widget to display details for the latest earthquake detected.
- Once completed and added to the home screen, your widget will appear as below Figure
- This widget listens for broadcast Intents that announce an update has been performed and sets the minimum update rate to ensure it is updated once per day regardless



EARTHQUAKE WIDGET

1. Start by creating the layout for the widget UI as an XML resource Use a Linear Layout to configure Text Views that display the quake magnitude and location

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#F111"
    android:padding="5sp">
    <TextView
        android:id="@+id/widget_magnitude"
        android:layout_width="wrapandroid:layout_height="fill_parent"
        android:textSize="24sp"
        android:padding="3dp"
    />
    <TextView
        android:id="@+id/widget_details"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:textSize="14sp"
        android:padding="3dp"
   />
</LinearLayout>
```

EARTHQUAKE WIDGET

2. Create a stub for a new **EarthquakeWidget** class that extends **AppWidgetProvider**. You'll return to this class to update your widget with the latest quake details.

```
package com.paad.earthquake;

import android.widget.RemoteViews;
import android.appwidget.AppWidgetManager;
import android.appwidget.AppWidgetProvider;
import android.content.ComponentName;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;

public class EarthquakeWidget extends AppWidgetProvider {
}
```

EARTHQUAKE WIDGET

3. Create a new widget definition file, **quake_widget_info.xml**, and place it in the res/xml folder. Set the minimum update rate to 24 hours and set the widget dimensions to two cells wide and one cell high—146dp×74dp. Use the widget layout you created in Step 1 for the initial layout.

```
<?xml version="1.0" encoding="utf-8"?>
<appwidget-provider
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:initialLayout="@layout/quake_widget"
   android:minWidth="146dp"
   android:minHeight="74dp"
   android:label="Last Earthquake"
   android:updatePeriodMillis="86400000"
/>
```

EARTHQUAKE WIDGET

4. Add your widget to the application manifest, including a reference to the widget definition resource you created in Step 3, and registering an Intent Filter for the App Widget update action.

```
<receiver android:name="EarthquakeWidget" android:label="Last
Earthquake">
    <intent-filter>
        <action android:name="android.appwidget.action.APPWIDGET_UPDATE" />
    </intent-filter>
    <meta-data
        android:name="android.appwidget.provider"
        android:resource="@xml/earthquake_widget_info"
    />
</receiver>
```

- Your widget is now configured and will be available to add to the home screen. You now need to update the **EarthquakeWidget** class from Step 2 to update the widget to display the details of the latest quake

EARTHQUAKE WIDGET

5. Start by creating two new **updateQuake** methods within the Earthquake Widget class

① The first should take an App Widget Manager and an array of widget IDs as well as the context. Later you'll extend this second stub to update the widget appearance using Remote Views

```
public void updateQuake(Context context,
                        AppWidgetManager appWidgetManager,
                        int[] appWidgetIds) {
}
```

EARTHQUAKE WIDGET

② The second method stub should take only the context, using that to obtain an instance of the **AppWidgetManager.** Then use the App Widget Manager to find the widget IDs of the active Earthquake widgets, passing both into the method you created in Step 5-1

```
public void updateQuake(Context context) {
    ComponentName thisWidget = new ComponentName(context,
                                                 EarthquakeWidget.class);
    AppWidgetManager appWidgetManager =
        AppWidgetManager.getInstance(context);
    int[] appWidgetIds = appWidgetManager.getAppWidgetIds(thisWidget);
    updateQuake(context, appWidgetManager, appWidgetIds);
}
```

EARTHQUAKE WIDGET

③ **Within the updateQuake stub from Step 5-1 use the Earthquake Content Provider to retrieve the newest quake and extract its magnitude and location:**

```
public void updateQuake(Context context,
                        AppWidgetManager appWidgetManager,
                        int[] appWidgetIds) {
  Cursor lastEarthquake;
  ContentResolver cr = context.getContentResolver();
  lastEarthquake = cr.query(EarthquakeProvider.CONTENT_URI,
                            null, null, null, null);
  String magnitude = "--";
  String details = "-- None --";

  if (lastEarthquake != null) {
    try {
      if (lastEarthquake.moveToFirst()) {
          magnitude =
      lastEarthquake.getString(EarthquakeProvider.MAGNITUDE_COLUMN);
          details =
      lastEarthquake.getString(EarthquakeProvider.DETAILS_COLUMN);
       }
  }
  finally {
    lastEarthquake.close();
  }
 }
}
```

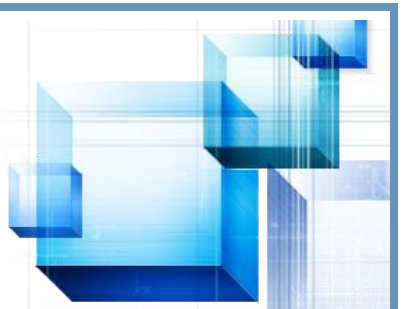
EARTHQUAKE WIDGET

④ **Create a new RemoteViews object to set the text displayed by the widget's Text View elements to show the magnitude and location of the last quake:**

```
public void updateQuake(Context context,
                        AppWidgetManager appWidgetManager,
                        int[] appWidgetIds) {
  Cursor lastEarthquake;
  ContentResolver cr = context.getContentResolver();
  lastEarthquake = cr.query(EarthquakeProvider.CONTENT_URI,
                            null, null, null, null);
  String magnitude = "--";
  String details = "-- None --";

  if (lastEarthquake != null) {
    try {
      if (lastEarthquake.moveToFirst()) {
        magnitude =
lastEarthquake.getString(EarthquakeProvider.MAGNITUDE_COLUMN);
        details =
lastEarthquake.getString(EarthquakeProvider.DETAILS_COLUMN);
      }
    }
    finally {
      lastEarthquake.close();
    }
  }
```

EARTHQUAKE WIDGET



④ Con't...

```
final int N = appWidgetIds.length;
for (int i = 0; i < N; i++) {
   int appWidgetId = appWidgetIds[i];
   RemoteViews views = new RemoteViews(context.getPackageName(),
                                                    R.layout.quake_widget);
   views.setTextViewText(R.id.widget_magnitude, magnitude);
   views.setTextViewText(R.id.widget_details, details);
   appWidgetManager.updateAppWidget(appWidgetId, views);
 }
}
```

EARTHQUAKE WIDGET

6. Override the onUpdate handler to call updateQuake:

```
@Override
public void onUpdate(Context context,
                     AppWidgetManager appWidgetManager,
                     int[] appWidgetIds) {
  updateQuake(context, appWidgetManager, appWidgetIds);
}
```

- Your widget is now ready to be used, and will update with new earthquake details when added to the home screen and once every 24 hours thereafter.

EARTHQUAKE WIDGET

7. Now enhance the widget to update whenever the Earthquake Service has refreshed the earthquake database:

① Start by updating the **doRefreshEarthquakes** method in the Earthquake Service to broadcast an Intent when it has completed.

```
public static String QUAKES_REFRESHED =
    "com.paad.earthquake.QUAKES_REFRESHED";
public void doRefreshEarthquakes() {
    [ ... Existing doRefreshEarthquakes code ... ]
    sendBroadcast(new Intent(QUAKES_REFRESHED));
}
```

EARTHQUAKE WIDGET

② Override the **onReceive** method in the **EarthquakeWidget** class, but be sure to call through to the superclass to ensure that the standard widget event handlers are still triggered:

```
@Override
public void onReceive(Context context, Intent intent){
    super.onReceive(context, intent);
}
```

EARTHQUAKE WIDGET

③ **Add a check for the QUAKES_REFRESHED action you broadcast in Step 7.1, and call updateQuakes when it's received:**

```
@Override
public void onReceive(Context context, Intent intent){
    super.onReceive(context, intent);

    if (intent.getAction().equals(EarthquakeService.QUAKES_REFRESHED))
      updateQuake(context);
}
```

EARTHQUAKE WIDGET

④ **Finally, add an Intent Filter for this Intent action to the widget's manifest entry:**

```
<receiver android:name="EarthquakeWidget" android:label="Last
Earthquake">
    <intent-filter>
      <action
android:name="android.appwidget.action.APPWIDGET_UPDATE" />
    </intent-filter>
    <intent-filter>
      <action
android:name="com.paad.earthquake.QUAKES_REFRESHED" />
    </intent-filter>
    <meta-data
        android:name="android.appwidget.provider"
        android:resource="@xml/earthquake_widget_info"
    />
</receiver>
```

EARTHQUAKE WIDGET

- Your widget will now update once per day, and every time the Earthquake Service performs a lookup.
- To enhance the Earthquake Widget, consider how you could use Layered Drawables within an Image View to indicate the magnitude of the earthquake being shown. Below Figure shows one possibility.