Mittuniversitetet
MID SWEDEN UNIVERSITY

Inlämningsuppgift 4

Applikationsutveckling för Android, 7,5 hp

Syfte:	Att visa att du klarar av att skapa mer avancerade egna Custom Views samt kunna spela upp ljud i Android.
Antal uppgifter:	2
Att läsa:	Lektion 4
Inlämning:	Inlämningslåda 4 i Moodle

Uppgift 1

Du ska skapa en egen komponent som ska efterlikna knappsatsen på en telefon. I komponenten ska du på lämpligt sätt rita ut siffrorna 0 till 9 samt tecknen * och #. Du får själv välja hur detta ska göras (exempelvis genom att använda befintliga komponenter som `Button`/`ImageButton` eller överlagra `onDraw` och rita allt själv).

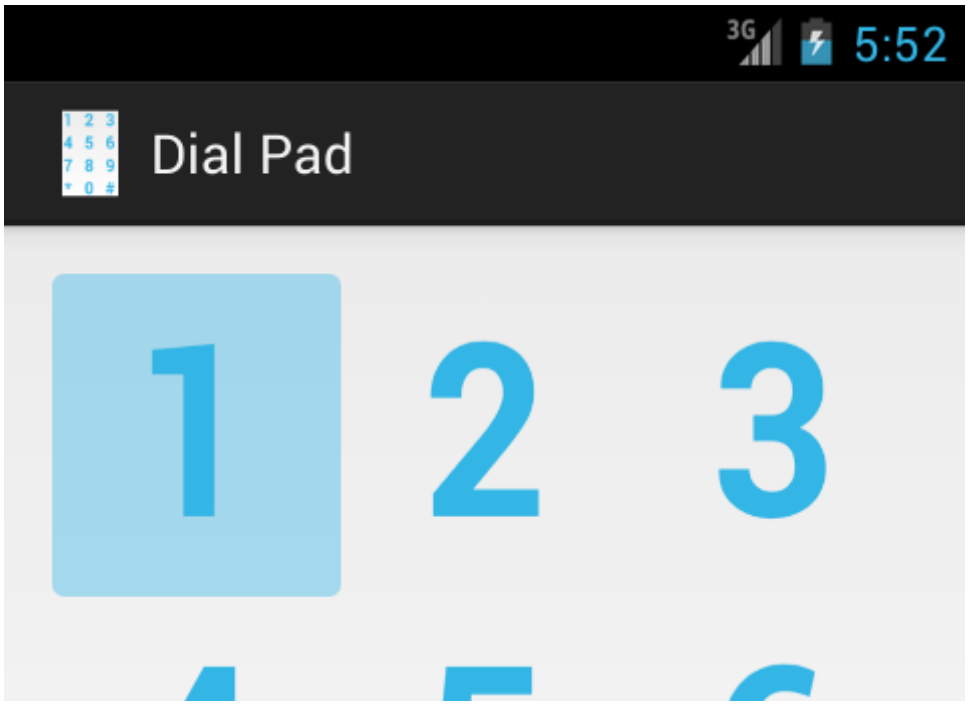
Till inlämningsuppgiften medföljer det zip-filer som innehåller bilder på sifferknappar. Dessa kan du använda dig av (packa i så fall upp dem i mappen `res/drawable` i ditt projekt). Se exempel i figur 1 och 2 nedan där bilder från filen dialpad3.zip används.

Figur 1. Android 2.3.3 (Theme.Light)

Figur 2. Android 4.2 (Theme.Holo.Light.DarkActionBar)

Komponenten ska fungera i både porträtt- och landskapsläge och ska uppta så stor yta som möjligt i båda dessa lägen. Alla knappar i komponenten ska uppta lika stor yta i komponenten (åtminstone sifferknapparna, * och # får vara en annan storlek).

Om användaren trycker på någon av knapparna i komponenten, antingen genom att trycka på skärmen (kontrollera med `onTouchEvent` eller `onClick` beroende på hur din lösning ser ut) eller genom att använda enhetens fysiska tangentbord (kontrollera med `onKeyDown`/`onKeyUp`) ska motsvarande siffra/tecken aktiveras i din knappsats. Med aktivera menas att knappen ska ändra utseende på något sätt (t.ex. at skifta bildfil från blått till grönt, att bakgrundsfärgen ändras, att knappen animeras så att den ser nedtryckt ut). Se exempel i figur 3 nedan där användaren tryckt på sifferknappen 1.



Figur 3. Android 4.2 (Theme.Holo.Light.DarkActionBar)

Utöver att knappen ändrar utseende ska även ett ljud spelas upp. Ljuden som ska spelas upp finner du i filen mamacita_us.zip. Packa upp innehållet till mappen `res/raw` i ditt Eclipse-projekt och använd klassen `SoundPool` för att spela upp ljuden.

Skapa tillsist en `Activity` som använder din knappsats. Denna aktivitet ska i stort sett enbart innehålla kod i `onCreate` för att sätta din komponent som content view.

Extra

Du som vill ha en extra utmaning kan göra följande tillägg:

I paketet `android.media` finns klassen `ToneGenerator`. Denna används för att spela upp DTMF-toner. Följande rader skapar en `ToneGenerator` och spelar en ton som motsvarar siffran 0:

```
ToneGenerator tone =
  new ToneGenerator(AudioManager.STREAM_MUSIC, 100);
tone.startTone(ToneGenerator.TONE_DTMF_0);
```

Tonen spelas med högsta volym (talet 100 anger procent av maxvolymen ljudet spelas upp i). Tonen spelas kontinuerligt till dess att metoden `stopTone()` anropas:

```
tone.stopTone();
```

Du ska lägga till möjligheten för användaren att välja om ljudet ska spelas upp som en röst (ljuden i zip-filen) eller som en ton (`ToneGenerator`).

Tips

Det som menas med att göra en egen komponent i denna uppgift är att knappsatsen ska utformas så att till exempel nya objekt av knappsatsen kan skapas…

```
DialPadView myDialPad = new DialPadView(context);
setContentView(myDialPad);
```

Alternativt att knappsatsen används i en layoutfil (main.xml):

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:layout_width="match_parent"
  android:layout_height="match_parent" >
<mitt.paket.DialPadView
  android:id="@+id/dialpad"
  android:layout_width="match_parent"
  android:layout_height="match_parent" />
</LinearLayout>
```

… för att sen göras synlig i en aktivitet med:

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

Ett sätt att skapa en egen komponent är enligt avsnittet Compound Controls i Android developers. Det vill säga skapa en klass som ärver från en `ViewGroup` (en layout-klass) av lämpligt slag. Korta instruktioner finner du på följande sida:

http://developer.android.com/guide/topics/ui/custom-components.html#compound

Ta en titt på exemplen som det hänvisas till. `List4.java` och `List6.java`. Dessa klasser finner du i installationskatalogen för Android SDK (ladda hem Samples for SDK via Android SDK Manager). Kolla i källkoden och scrolla ner till den privata klassen `SpeechView`. Den ärver `LinearLayout` och i konstruktorn finner du exempel på hur du lägger till komponenter av olika slag.

Ett annat (enklare?) sätt att skapa en egen komponent är att specificera innehållet i den egna komponenten med en layoutfil. Därefter skapar du en klass som ärver från samma typ av klass som root-elementet i din layoutfil för den egna komponenten. Använd sen en `LayoutInflater` för att "ladda" innehållet i layoutfilen. Dvs enligt avsnitt 3 och 4 i den tutorial lektionen hänvisade till:

http://www.vogella.com/articles/AndroidCustomViews/article.html#compoundcontrols

I stället för `view_color_options.xml` döper du layoutfilen förslagsvis till `dialpadview.xml` och i denna utformar du knappsatsens utseende. Använd en `LinearLayout` eller `TableLayout` som root-element. I stället för klassen `ColorOptionsView` skapar du en klass `DialPadView` som ärver från `LinearLayout` eller `TableLayout` (beroende på vad du vald i `dialpadview.xml`).

Får du problem med att din knappsats inte kan hantera `KeyEvent` på rätt sätt läs om vad `android:focusable` och `android:focusableInTouchMode` innebär.

Eftersom knapparna ska gå att klicka på utgår du förslagsvis från någon befintlig klickbar komponent i din layoutfil för att slippa implementera all logik själv. För att ändra knappens utseende till att använda någon av de bifogade filerna använder du `android:background`. För att hantera knappens olika tillstånd (nedtryckt, fokus, otryckt) använder du med fördel en State List Drawable (en för varje knapp i din knappsats). Se exempel på Android Developers i avsnittet om Buttons:

http://developer.android.com/guide/topics/ui/controls/button.html#CustomBackground

I figur 3 har jag använt mig av tekniken enligt länken ovan. För tillståndet ”pressed” har jag använt mig av en Layer List (`LayerDrawable`):

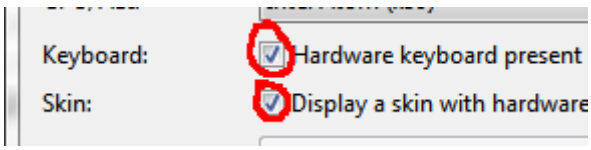
http://developer.android.com/guide/topics/resources/drawable-resource.html#LayerList

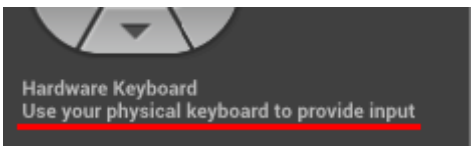
```
<layer-list>
  <item android:drawable="@drawable/dialpad_1"/>
  <item android:drawable="@drawable/button_pressed"/>
</layer-list>
```

Som första item används knappens vanliga bakgrundsbild och som andra item används en Shape Drawable som då ritas ovanpå den första item. I button_pressed finns kod som ”ritar” en halvt genomskinlig blåaktig rundad rektangel (ritas ovanpå bakgrundsbilden tack vare användandet av en layer list).

```
<shape xmlns:android="http://schemas.android.com/apk/res/android" >
    <solid android:color="#6633b5e5" />
    <corners android:radius="4dp" />
</shape>
```

För att i en AVD (emulator) kunna testa med ett fysiskt tangentbord måste du skapa en, eller editera en befintlig, AVD i vilken du kryssar i Hardware keyboard present. Om du dessutom kryssar i Display a skin with hardware controls så ska det i emulatorn visas en text att du kan använda datorns tangentbord för att ge input.





Hör av dig till kursansvarig för ytterligare tips och råd om du kör fast.

Svar: Lämna in hela projektet från Eclipse.

Uppgift 2

Denna uppgift utgår från rövarspråksöversättaren du gjorde i inlämningsuppgift 3. Du ska nu lägga till bakgrundsmusik som spelas när användaren befinner sig på första sidan (den med knapparna). Välj själv din favoritmusik i ett filformat som stöds av Android. När användaren går från startsidan ska musiken sluta spelas. Återvänder användaren till startsidan ska musiken börja spela igen från den plats där musiken stoppades. Även vid skärmrotation ska musiken spelas upp från den position den befann sig på när skärmrotationen gjordes.

Det är ok att använda en vanlig mp3 eller liknande i denna övning och lägga den direkt i res/raw. Detta rekommenderas dock inte i en ”riktig” applikation eftersom denna typ av fil vanligtvis är onödigt stor (4-5 MB inte ovanligt) för en app liknande denna.

Tips

Använd aktivitetens livscykelmetoder för att styra musikspelarens motsvarande tillstånd (spela/pausa/stoppa/frigöra). Överskugga metoderna `onSaveInstanceState` och `onRestoreInstanceState` i klassen `Activity` för att spara och återställa mediespelarens nuvarande position. Exempel:

http://developer.android.com/training/basics/activity-lifecycle/recreating.html

Svar: Lämna in hela projektet från Eclipse.