Mittuniversitetet
MID SWEDEN UNIVERSITY

Inlämningsuppgift 3

Applikationsutveckling för Android, 7,5 hp

Syfte:	Att visa att du klarar av att skapa egna Custom Views samt kunna använda Animations i ett användargränssnitt.
Antal uppgifter:	2
Att läsa:	Lektion 3
Inlämning:	Inlämningslåda 3 i Moodle

Uppgift 1

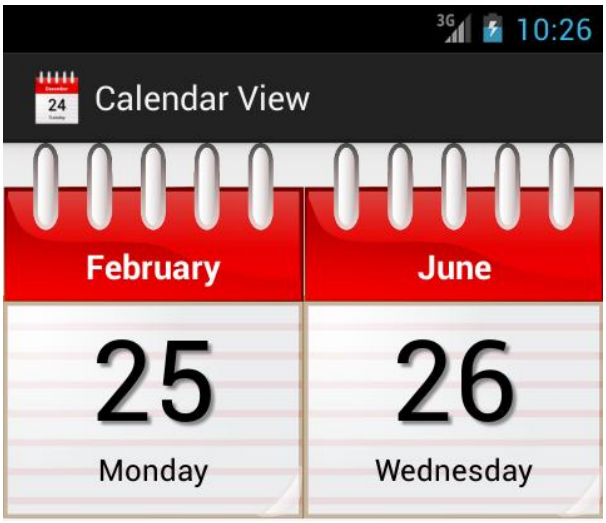
Du ska skapa en egen komponent med namnet `CalendarView` som ärver från klassen `View`. Denna ska visa aktuell månad (Januari, …, December), veckodag (Monday, …, Sunday) och dag (1, …, 31). Du behöver inte använda engelska som språk. När ett nytt objekt av `CalendarView` skapas ska dagens datum användas. Det ska finnas en metod `setDate(date)` som kan användas för att byta datum till det som anges av argumentet av typen `Date`, `Calendar`, eller `String` (välj vilken du vill använda). Det ska även finnas en metod `nextDay()` som sätter aktuellt datum till nästa dag (nuvarande datum + en dag). När någon av dessa metoder anropas ska givetvis komponentens innehåll uppdateras (din `View` ska ritas om).

Till uppgiften följer det med en png-bild (se figur 1) som du ska använda som bakgrund. Ovanpå denna ska sen månad, veckodag och dag på ett tydligt sätt skrivas ut (se figur 2). Använd med fördel någon av dessa två bilder som din ikon för appen.

Figur 1.

Figur 2.

Skapa en `Activity` som använder minst två `CalendarView`. Den ena av dem ska visa aktuellt/dagens datum och de andra kan visa valfria datum (ok att ange detta datum direkt i källkoden med metoden setDate) . Registrera en `OnClickListener` på minst en av dina `CalendarView`. När användaren klickar på kalendern ska metoden `nextDay` anropas. I figur 3 visas två `CalendarView` sida vid sida i en aktivitet.



Figur 3.

Det är inget krav att din `CalendarView` ska skala bilden så att exakt två kalenderblad får plats bredvid varandra i en aktivitet så som figur 3 kan ge intryck av (det är bara en tillfällighet att det blev så i min skärmdump och inget som var planerat). Däremot ska din `CalendarView` fungera (se bra ut) i olika upplösningar. Med detta menas att hela kalenderbladet alltid ska ritas ut (bakgrundsbilden ska inte klippas/beskäras) varken mer eller mindre. Storleken på text och dess position ska även anpassas efter komponentens storlek. Någon del av texten får inte hamna utanför bakgrundsbilden.

Välj själv om du vill att din komponent ska kunna ges olika storlekar eller om den ska använda en fast storlek (= storleken på den bakgrundsbild som används). Om du väljer att tillåta valfri storlek ska givetvis bakgrundsbilden och all text anpassas efter den storlek som anges. Om t.ex. nedanstående storlekar specificeras i xml

```
<my.package.Calendarview
    android:id="@+id/calendarview1"
    android:layout_width="50dp"
    android:layout_height="75dp" />

<my.package.CalendarView
    android:id="@+id/calendarview2"
    android:layout_width="255dp"
    android:layout_height="133dp"/>
```

måste bild och text anpassas enligt exemplet i figur 4:

Figur 4.

Väljer du att din komponent ska använda en fast storlek kommer resultatet att se ut som i figur 3 oavsett vilka värden som anges i android_layout_width/height (eller motsvarande metoder i källkod).

Testa din lösning i minst två olika upplösningar innan lämnar in för att vara säker på att allt fungerar som tänkt.

Tips

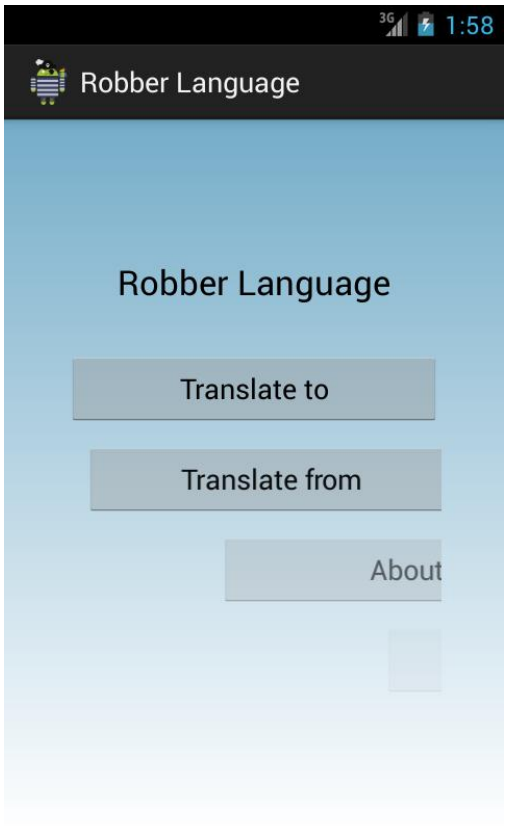
- Ta en titt på klasserna i paketet `android.text.format` för hjälp med att formatera datum.
- För öka datumet med en dag kan du använda `java.util.Calendar`.
- Tänk på att använda ”density independent pixel” i din lösning och spara eventuella värden som en resurs (res/values/dimens.xml i taggen <dimen>). Om du behöver konvertera mellan dpi och vanliga pixlar se följande länk: http://developer.android.com/guide/practices/screens_support.html#dips-pels.

- Med följande kod skapar du en `Bitmap` vilken du kan använda i någon av `Canvas` `drawBitmap`-metoder:
```
Bitmap myBitmap = BitmapFactory.decodeResource
(context.getResources(), R.drawable.mypngfile);
```
- Om din komponent ska använda en fast storlek sätt storleken förslagsvis till samma storlek som din `Bitmap` för bakgrundsbilden. Gör detta genom att överskugga metoden `onMeasure` och anropa `setMeasuredDimension`.

Uppgift 2

Denna uppgift utgår från rövarspråksöversättaren du skapade i inlämningsuppgift 2. Du ska nu förändra bakgrunden så att den tonas från en färg till en annan. Välj själv mellan vilka färger bakgrunden ska tonas samt hur toningen ska se ut. Se exempel i figurerna nedan.

Figur 4. Android 2.3.3 (Theme.Light)



Figur 5. Android 4.2 (Theme.Holo.Light.DarkActionBar)

Du ska även animera vissa grafiska komponenter. När applikationen startar ska de fyra knapparna svepa in från höger en efter en, samtidigt som de tonas från fullständigt genomskinliga till ogenomskinliga. Knapparna ska svepa in en efter en med en viss fördröjning. Se exempel i figur 5 ovan. Välj själv om du vill använda dig av Property Animation eller View Animation. Välj även om du vill deklarera animationen i XML eller direkt i kod.

Om aktiviteten befunnit sig i ett stoppat tillstånd, t.ex. om en annan aktivitet helt har dolt den, ska knapparna animeras igen när aktiviteten startar.

Tips

- Använd en `Shape Drawable` för att skapa toningen. http://developer.android.com/guide/topics/resources/drawable-resource.html#Shape

- Använd följande mall och spara den i res/drawable:

```
<?xml version="1.0" encoding="utf-8"?>
<shape
xmlns:android="http://schemas.android.com/apk/res/android
">
  <gradient
  />
</shape>
```

- För att animera en hel layout (`ViewGroup`) skapa först en animation (Animation eller Animation Set) som definierar hur layouten ska animeras. Sparas som t.ex. `anim/fade_from_right.xml`.

  Skapa därefter en `layoutAnimation` som refererar till den animation som ska användas. Sparas som t.ex. `anim/button_animation.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<layoutAnimation
xmlns:android="http://schemas.android.com/apk/res/android
"
  android:animation="@anim/fade_from_right" />
```

  För att knyta animationen till en `ViewGroup` (t.ex. en `LinearLayout`) i XML använd följande kod:

```
android:layoutAnimation="@anim/button_animation">
```

- Med `android:layoutAnimation` körs animationen när layouten ritas ut. Detta händer till exempel när `setContentView` anropas i kod. Ett annat sätt är att anropa metoden `startLayoutAnimation` på ett `ViewGroup`-objekt. Detta anrop leder till att aktuell view group ritas om.

Svar: Lämna in båda projekten från Eclipse.