Mittuniversitetet
MID SWEDEN UNIVERSITY

Inlämningsuppgift 6

Applikationsutveckling för Android, 7,5 hp

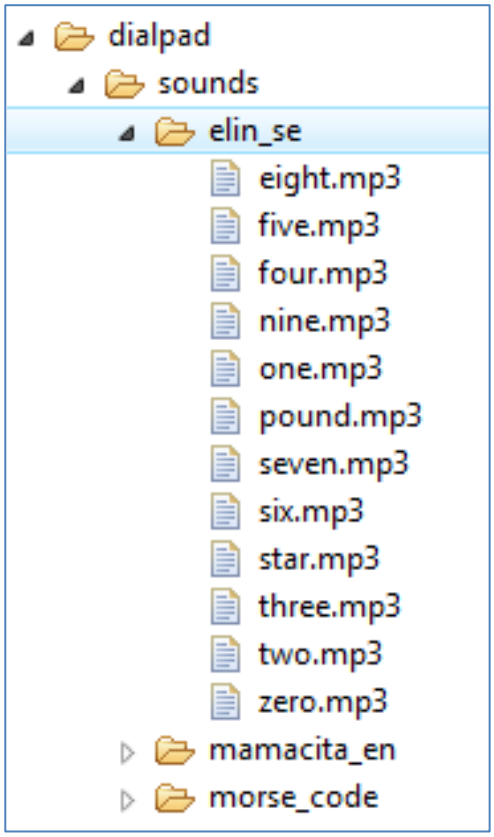
Syfte:	Att visa att du klarar av att använda WebView, och trådar för att i bakgrunden ladda hem resurser från internet och lagra dessa i telefonens externa lagringsutrymme.
Antal uppgifter:	1
Att läsa:	Lektion 6
Inlämning:	Inlämningslåda 6 i Moodle

Uppgift 1

Utgå från din knappsats från inlämningsuppgift 5. I denna uppgift ska du göra det möjligt att starta samtalet direkt i stället för att endast komma till enhetens telefonapp. Använd `Intent.ACTION_CALL` som action när du skapar din `Intent` för att göra samtalet. För att applikationen ska för sätta igång ett samtal direkt måste applikationen tillåta `android.permission.CALL_PHONE` (lägg till en `uses-permission` i applikationens manifest).

Det ska nu även vara möjligt att ladda hem nya röstfiler till telefonen. På adressen http://dt031g.programvaruteknik.nu/dialpad/sounds/ finns ett antal filer som kan laddas hem. Filerna är packade till en zip-fil (.zip), men där filändelsen har ändrats till .dps (DialPad Sound). Varje fil innehåller en mapp, som i sin tur innehåller en ljudfil för varje knapp (zero.mp3, one.mp3, …, pound.mp3). Användaren ska kunna gå till webbsidan ovan och klicka på den röst hon vill ladda hem. Se figur 1 nedan. När användaren klickar på en röstfil ska den laddas hem till enheten, packas upp och sparas i enhetens externa lagringsutrymme.

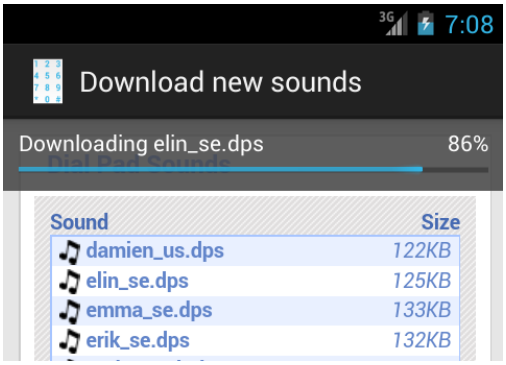
Figur 1. WebView som visar en webbsida med röstfiler att ladda ner.



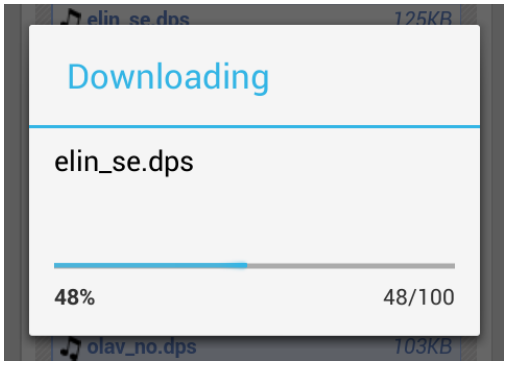
Figur 2. Mapphierarki över nerladdade röstfiler.

Om användaren väljer att ladda hem `elin_se.dps` ska innehållet packas upp till `<ExternalStorageDirectory>/dialpad/sounds`. <ExternalStorageDirectory> motsvarar sökvägen till enhetens externa lagringsplats, se figur 2 ovan.

Skapa en ny aktivitet för nerladdning av ljudfilerna. När aktiviteten startas ska du med `Intent.putExtra()` ange från vilken internetadress röstfilerna kan laddas hem samt till vilken destinationsmapp den nerladdade filen ska packas upp. Medan nerladdningen pågår ska en progress bar av något slag visas. Nerladdningen och uppackningen ska ske i en separat tråd. Se exempel i figur 3 och 4.

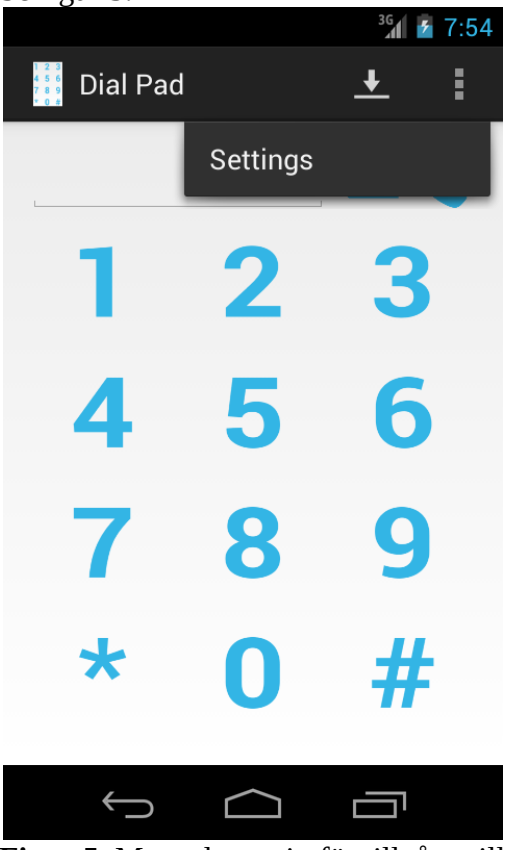


Figur 3. Status på nerladdning i en egen progress bar.

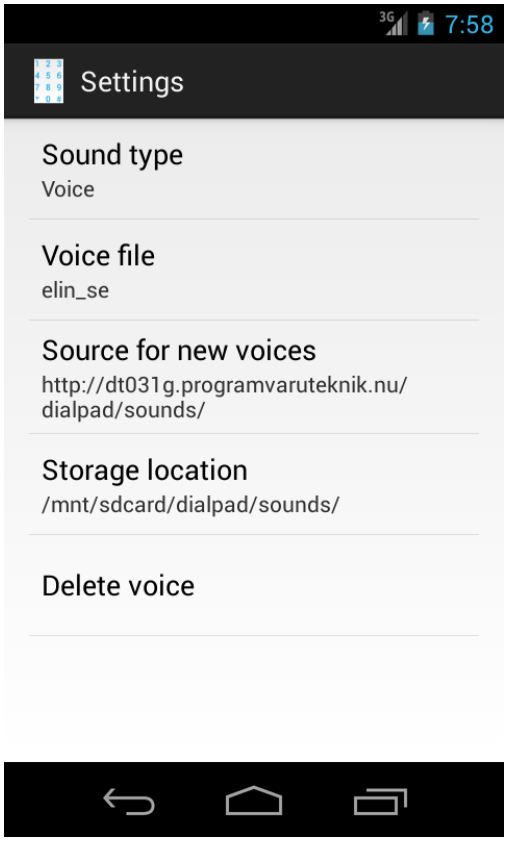


Figur 4. Status på nerladdning i en ProgressDialog.

Den nya nerladdningsaktiviteten ska vara åtkomlig från knappsatsaktiviteten via ett menyalternativ. Menyalternativet ska kunna nås genom att trycka på telefonens menyknapp alternativt vara åtkomlig som en action item i Action Bar för enheter utan dedikerad menyknapp. Detta menyalternativ ska både ha en ikon (använd en egen eller den som finns bifogad med inlämningsuppgiften) samt ha en titel (Download). Se figur 5.



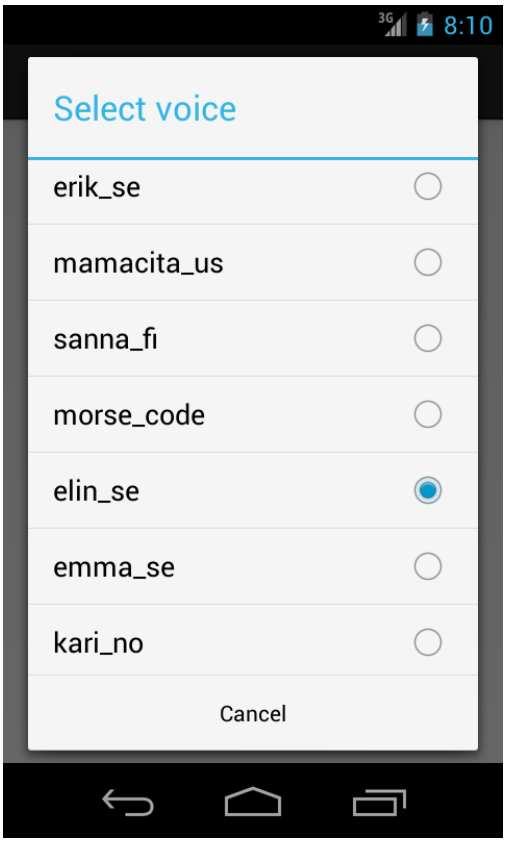
Figur 5. Menyalternativ för tillgång till inställningar på enhet utan menyknapp.



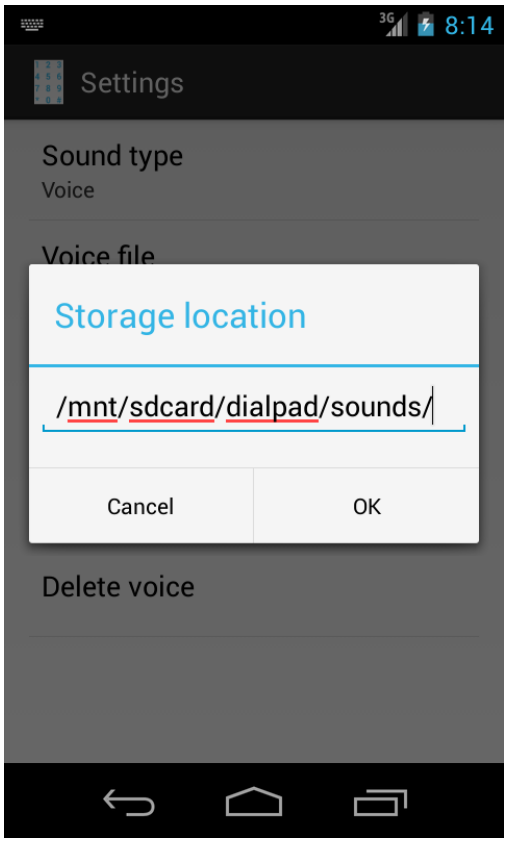
Figur 6. Möjliga inställningar för knappsatsen.

I knappsatsaktiviteten ska det även finnas ett menyalternativ, åtkomlig via enhetens menyknapp alternativt genom overflow menu för enheter utan dedikerad menyknapp, där användaren kan komma åt inställningar för knappsatsen. Se exempel i figur 5 och 6 på föregående sida.

Det ska minst finnas en inställning för att välja vilken röst som ska användas (bland de röster som finns lagrade i <ExternalStorageDirectory>/dialpad/sounds). Se exempel i figur 6 på föregående sida för exempel på ytterligare inställningar du kan ha med. Om du för dina inställningar använder paketet `android.preferences` kan du med fördel använda klassen `ListPreference` för val av ljud. Se figur 7.



Figur 7. Användning av en List-Preference för val av röst.



Figur 8. Användning av en EditText-Preference för val av lagringsplats.

När nya ljudfiler laddas ner (eller om ljudfiler tas bort eller lagringsplatsen ändras) ska listan i figur 7 givetvis uppdateras.

Välj själv hur pass omfattande felkontroller du vill göra. T.ex. är det inte nödvändigt att varna om ljudfilen som laddas hem redan finns i telefonen. Däremot ska du givetvis undvika så långt det är möjligt att applikationen kraschar.

Tips

När du använder en `WebView` för att visa en webbsida måste du skapa en egen `WebViewClient` och överskugga metoden `shouldOverrideUrlLoading`. Om inte kommer länkar/filer m.m. som användaren klickar på öppnas i enhetens webbläsare i stället för i din `WebView`. När du överlagrat metoden kan du registrera en `DownloadListener` på din `WebView`. I metoden `onDownloadStart` kan du sen kontrollera och hantera nerladdningar som görs.

Klasserna `URL`, `URLConnection` m.fl. fungerar på samma sätt i Android som i J2SE. Använd dessa för att skapa en `InputStream` till den fil som ska laddas ner. Om du får problem med att skriva till det externa lagringsutrymmet, kontrollera så att du använder rätt Manifest.permission.

Klassen `ZIP` har jag använt som ett exempel i en annan kurs. Du kan använda den om du vill för att packa upp innehållet i den nerladdade filen.

Om du sköter inställningar med klassen `PreferenceActivity` och behöver veta när en förändring i en inställning sker, kan du låta klassen implementera gränssnittet `OnSharedPreferenceChangeListener`.

I metoden `onSharedPreferenceChanged (SharedPreferences sharedPreferences, String key)` kan du sen kontrollera vilken inställning som förändrats och reagera därefter (argumentet `key` är samma som du anger med `android:key` i XML-filen). För att registrera lyssnaren använder du följande kod:

```
getPreferenceScreen().getSharedPreferences().registerOnSharedP
referenceChangeListener(this);
```

För att med Javakod ange värden i en `ListPreference` använder du metoden `setEntries` samt `setEntryValues`. Båda tar en array av typen `String` som argument. Ex:

```
ListPreference voice =
(ListPreference)getPreferenceScreen().findPreference("voice");
String entries = {"elin_se", "erik_se"};
String entryValues = {"/mnt/sdcard/dialpad/sounds/elin_se/",
                      "/mnt/sdcard/dialpad/sounds/erik_se/"};
voice.setEntries(entries);
voice.setEntryValues(entryValues);
```

För att få det aktuella valet kan du göra följande anrop:

```
voice.getValue();
```

Enligt de riktlinjer som finns för användning av dialogrutor i Android rekommenderas inte längre användning av `ProgressDialog` (se Avoid ProgressDialog på följande sida http://developer.android.com/guide/topics/ui/dialogs.html). I stället bör man lägg till en `ProgressBar` i sin layout.

När vi använder en `WebView` som upptar hela enhetens yta kan det vara svårt att på ett snyggt sätt få till användandet av en `ProgressBar` varför jag anser det vara motiverat at ändå använda en `ProgressDialog` vid dessa tillfällen (som i figur 4). Med andra ord är det ok om du i din lösning använder en `ProgressDialog`.

I exemplet i figur 3 har jag i layoutfilen lagt en `ViewGroup` (`RelativeLayout`) ovanpå `WebView`. I den har jag sen använt två `TextView` och en `ProgressBar` för att visa vilken fil som laddas ner och hur långt nerladdningen har kommit. Initialt är denna `ViewGroup` satt att vara osynlig (`android:visibility`) för att sen göras synlig när nerladdningen väl startar (döljer den igen när nerladdningen är klar). Bakgrunden på `ViewGroup` har jag satt till en delvis genomskinlig grå färg.

För bakgrundsjobb (som nerladdning av filer) där användargränssnittet samtidigt ska uppdateras är klassen `AsyncTask` väldigt smidig att använda. Skapa en inre klass som ärver `AsyncTask` till den klass som behöver utföra bakgrundsjobbet. På så sätt kommer din `AsyncTask` enkelt åt instansvariabler deklarerade i den yttre klassen.

Svar: Lämna in hela projektet från Eclipse.