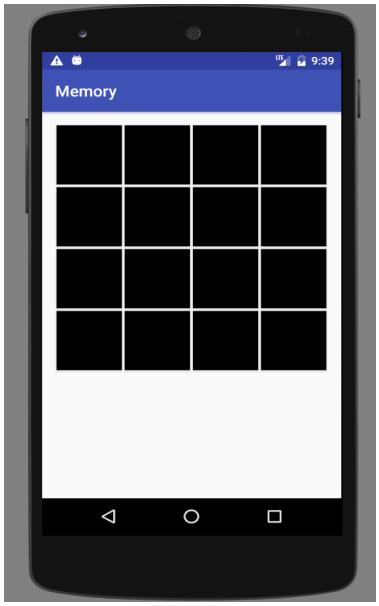
JEU DE MEMORY

L'objectif de ce TP est de développer un jeu de memory constitué de 16 cartes de couleur.



Les 16 cartes seront représentées par 16 objets de la classe **Button** qui seront disposés à l'intérieur d'un **GridLayout** déclaré dans votre fichier de ressources xml. Afin de déterminer la taille des boutons pour qu'ils occupent la largeur de l'écran, vous aurez besoin de connaître cette dimension. La largeur de l'écran en pixels est accessible via :

```
Resources.getSystem().getDisplayMetrics().widthPixels
```

Il vous faudra aussi récupérer la taille en pixels des marges gauches et droites. Utilisez pour cela :

```
getResources().getDimension(R.dimen.activity_vertical_margin)
```

qui vous permet de récupérer la valeur de la variable **activity_vertical_margin** situé dans le fichier ressource **dimen.xml**. Il faudra ensuite diviser cette valeur par

```
getResources().getDisplayMetrics().density
```

afin de prendre en compte la résolution de votre écran. Notons **dp** la valeur obtenue. Pour convertir cette valeur en pixels, exécutez :

```
int px = (int) TypedValue.applyDimension(TypedValue.COMPLEX_UNIT_DIP, dp,
                getResources().getDisplayMetrics())
```

Une fois que vous aurez déterminé la taille de vos boutons, il vous faudra créer un ViewGroup.MarginLayoutParams en spécifiant pour largeur et hauteur la taille que vous aurez préalablement calculée. Utilisez les attributs **leftMargin, rightMargin, topMargin** et **bottomMargin** de l'objet **ViewGroup.MarginLayoutParams** afin de définir une marge de 5 pixels autour des boutons.

Finalement , à chaque création d'un bouton vous devrez via la méthode **setLayoutParams**, appliquer votre objet **ViewGroup.MarginLayoutParams** à votre bouton afin de fixer sa dimension et ses marges.

Les contraintes du jeu sont les suivantes :

- seules 2 cartes peuvent être retournées en même temps,
- lorsque le joueur clique sur une carte, sa couleur est dévoilée (changez la couleur de fond du bouton)
- lorsque le joueur clique sur la deuxième carte, sa couleur est dévoilée puis au bout de une seconde:
    - si les deux cartes dévoilées sont identiques , elles deviennent blanches,
    - sinon les cartes sont à nouveau cachées (fond noir).

Afin d'effectuer une action qu'au bout d'un certain laps de temps, il faut utiliser un Handler :

```
Handler handler = new Handler();

handler.postDelayed(new Runnable() {
    public void run() {
            .. code à exécuter ...
    }
}, 1500);
```

dans l'exemple ci-dessus, le code de la méthode **run()** ne sera exécuter qu'au bout de 1500ms (1,5 secondes).