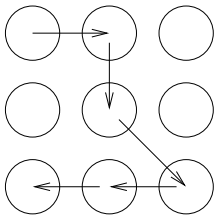
FlowDroid: devoir sur table

UPMC/Licence/Info/2I013

Avril 2015

On veut protéger notre application de jeu par la saisie d'un code. Pour la saisie du code, on se donne un panneau de 9 points. Le code est saisi en reliant les points dans un certain ordre, c'est-à-dire, selon un certain chemin. Les points reliés sont naturellement adjacents selon les axes verticaux horizontaux ou selon les deux diagonales. Les points sont numérotés de 0 à 8, de gauche à droite et de haut en bas.

Dans l'exemple ci-dessous, le chemin saisi correspond au code 014876.



Exercice I.

L'activité de saisie du code s'appellera `CodeActivity`. Elle est activée en premier lorsque l'application est lancée.

Question (I.1) Quelles modifications faut-il apporter au fichier `AndroidManifest.xml` de votre application ?

Afin de saisir le code, la vue de l'activité `CodeActivity` contiendra une *SurfaceView* que l'on appellera `CodeView`. La vue de l'activité sera définie dans le fichier `code.xml`.

Question (I.2) Donnez une définition minimale du fichier `code.xml`.

Question (I.3) Que doit contenir la méthode `onCreate` de la classe `CodeActivity` pour que cette vue soit présentée au lancement de l'application ?

Exercice II.

On se donne une classe `CodeModel` dont le rôle est d'enregistrer les points balayés lors de la saisie et de contrôler la validité du code saisie. On pose que le code attendu est contenu dans la variable privée `private int[] code` de la classe `CodeModel`. On pose également que les points balayés lors de la saisie sont enregistrés dans la variable `ArrayList<Integer> path` de cette classe. On a accès à cette dernière par la méthode `ArrayList<Integer> getPath()`.

Question (II.1) Définir la méthode `void start(int x, int y)` qui définit le début de la saisie (premier point touché). Les coordonnées `x` et `y` correspondent, respectivement au numéro de colonne et au numéro de ligne du point.

Question (II.2) Définir la méthode `void extend(int x, int y)` qui permet d'ajouter un numéro de point à la liste de ceux déjà saisis.

Question (II.3) Définir la méthode `boolean check()` qui donne `true` si et seulement si la liste de points saisis correspond exactement au code mémorisé dans le tableau `code`.

Attention :

- vous devrez veiller à ce qu'un même point ne soit pas enregistré 2 fois de suite ;
- pensez que l'on peut faire plusieurs tentatives de saisie.

Exercice III.

La classe `CodeView` étend `SurfaceView` et implémente `SurfaceHolder.Callback`. La mise à jour du dessin du panneau de saisie est confié à un *thread*. On pose que la classe `CodeView` possède la variable d'instance `CodeModel theCode`.

Question (III.1) Que faudra-t-il faire dans la classe `CodeView` pour lancer l'activité de jeu lorsque le code aura été correctement saisi ?

Question (III.2) Donnez la définition de la méthode `public boolean onTouchEvent(MotionEvent event)` de la classe `CodeView` qui doit réagir aux évènements tactiles `ACTION_DOWN` (début de saisie), `ACTION_MOVE` (ajout d'un point) et `ACTION_UP` (fin de saisie).

Souvenez vous qu'en fin de saisie, on lance ou non l'activité du jeu.

Question (III.3) Donnez une définition minimale de la méthode `public void onDraw(Canvas canvas)` de la classe `CodeView`.

Question (III.4) Où faut-il placer le code de lancement du *thread* de dessin ? Où faut-il placer le code de l'arrêt de ce *thread* ?