21779. Algorísmia i Estructures de Dades II
Curs 2023-24
ZenWord: La interfície revisada

18 d'abril de 2024

Durant les sessió anterior de pràctiques ens vam encarregar de la construcció de la part estàtica de la pantalla principal del joc, a més de les funcionalitats bàsiques associades als botons per entrar les lletres i als botons associats a les diferents opcions del programa. Durant aquesta sessió de pràctiques ens encarregarem de:

- Afegir a la interfície els *TextView* corresponents a les paraules amagades, per programa en lloc d'interactivament.
- Implementar la funció necessària per determinar si una paraula es pot formar a partir de les lletres de la paraula triada, és a dir, si és una possible solució.

1 Les paraules amagades

Una part important de la nostra aplicació és indicar a l'usuari la composició de les paraules amagades, és a dir, s'ha de mostrar a la pantalla un *TextView* per a cada una de les lletres que formen cada una de les paraules amagades. Per exemple, a la Figura 1 es poden veure 5 paraules amagades, 1 de longitud 5, 1 de longitud 4 i 3 de longitud 1.

Aquesta part de la interfície s'ha de crear programàticament, ja que en el temps de disseny de la interfície no es coneixen ni el número ni la longitud de les paraules amagades.

Fonamentalment, el que farem serà escriure una funció que crei una fila de camps de text (corresponent a una paraula oculta), a partir d'una guia

Figura 1: Exemple de paraules amagades de diferent longitud.

de disseny (*GuideLine*) i d'un número de lletres, centrada horitzontalment a la pantalla.

La declaració de la funció tendrà el següent aspecte:

```
public TextView[] crearFilaTextViews(int guia, int lletres)
```

on *guia* correspon a l'identificador d'una guia creada a través de l'eina de disseny (en temps de disseny no coneixem les paraules ocultes, però sí la posició on volem que apareguin). Aquests identificadors són constants a les que podem accedir (per exemple *R.id.nomdelid*).

Per a poder fer aquesta funció, haurem de:

- Generar un *id* per a cada *TextView*: recordau que tots els elements de la interfície tenen un identificador únic. Per crear-lo hem de cridar a la funció de la classe *View* anomenada *generateViewId*:

```
    View.generateViewId();
```

- Crear un *TextView* per a cada lletra de la paraula amb la instrucció:

```
    TextView textView = new TextView(this);
```

- Posar tots els atributs que necessiteu al *TextView*. Com a mínim vos farà falta inicialitzar:

    1. el seu *id* amb el identificador creat abans (*setId*).
    2. el seu text a buit (*setText*).
    3. i tots els aquells atributs (color, alineació, background,...) que considereu necessaris.

Fins aquí hem creat tots els elements que necessitau, ara ja només queda posar al *textView* de la lletra les restriccions (*constraints*) necessàries amb les altres lletres de la paraula. Per això, abans de res haureu de recuperar el *ConstraintLayout* de la vostra aplicació igual que ho fèiem amb els altres elements (*findViewById*) i posar el *TextView* a n'aquest Layout (*addView*).

Programàticament, les restriccions es creen a partir d'una instància de la classe *ConstraintSet*:

```
1    ConstraintSet constraintSet = new ConstraintSet();
```

A n'aquest objecte *constraintSet* li podeu afegir totes les restriccions que vulgueu amb el mètode *connect*:

```
1    void connect (int startID,   // id del View inicial
2                  int startSide, // on va la restricció
3                  int endID,     // id del segon View
4                  int endSide,   // on va la restricció
5                  int margin)    // marge en punts
```

Els elements s'accedeixen amb el seu *id*: els dels *TextView* són els generats anteriorment amb la funció *generateViewId*, el del parent seria la constant *ConstraintSet.PARENT_ID* i el de la guia s'obtendria amb la constant que referencia el seu id (*R.id.nomdelaguia*).

Per exemple, per posar una restricció del *TextView id*, que indiqui que comenci al mateix lloc que comença el seu parent (la pantalla), escriuríem:

```
1    constraintSet.connect(
2        id,
3        ConstraintSet.START,
4        ConstraintSet.PARENT_ID,
5        ConstraintSet.START,
6        margin);
```

Es llegiria: "el START del id va al START del parent i han d'estar separats per un número de punts equivalent a *margin*".

A més de la *ConstraintSet.START*, es poden utilitzar altres constants com *ConstraintSet.TOP*, *ConstrainSet.END* o *ContraintSet.BOTTOM*.

Es poden posar altres restriccions al vostre *TextView*, per exemple si vos interessés tenir una grandària determinada, podríeu utilitzar el *constraintSet.constrainWidth* i el *constraintSet.constrainHeight*:

```
    // Establir les dimensions del TextView
    constraintSet.constrainWidth(id, width);
    constraintSet.constrainHeight(id, height);
```

Quan hagueu posat totes les restriccions necessàries perquè les lletres es vegin correctament com voleu a la pantalla, heu d'aplicar aquestes restriccions al vostre *constraintSet*:

```
    constraintSet.applyTo(constraintLayout);
```

Potser necessitau conèixer les dimensions de la pantalla del vostre dispositiu per calcular els marges necessaris a cada costat de les vostres paraules ocultes. Això es pot fer utilitzant la classe *DisplayMetrics* al vostre mètode *onCreate*:

```
        //object to store display information
        DisplayMetrics metrics = new DisplayMetrics();
        //get display information
        getWindowManager().getDefaultDisplay().getMetrics(
    metrics);
        widthDisplay = metrics.widthPixels;
        heightDisplay = metrics.heightPixels;
```

Ja podeu crear totes les línies de paraules ocultes que vulgueu, cridant a la funció *crearFilaTextViews* amb els paràmetres adequats.

2 Paraula solució

En aquest apartat heu de crear un mètode *esParaulaSolucio* que tengui dos arguments de tipus *String*, paraula1 i paraula2, i que ens retorni un booleà que ens indiqui si la paraula2 es pot generar a partir de les lletres contingudes a la paraula1.

Recordau que per construir la paraula2 només es poden utilitzar les lletres que apareixen a la paraula1 i que només es poden repetir a la paraula2 les lletres que també estan repetides a la paraula1.

Recordau que, tal com s'indica a l'enunciat global de la pràctica, per a determinar si la paraula2 es pot generar a partir de les lletres de la paraula1, **és necessari emmagatzemar les lletres de la paraula1 dins el catàleg de les *lletres disponibles***: número d'aparicions de cada lletra a la *paraula1* (o *paraula triada*).

Exemples d'execució:

```
    System.out.println(esParaulaSolucio("puresa","apres");
    true
    System.out.println(esParaulaSolucio("puresa","apresa");
    false   // falta una a
    System.out.println(esParaulaSolucio("copta","coa");
    true
    System.out.println(esParaulaSolucio("saca","casa");
    true
    System.out.println(esParaulaSolucio("feiner","ene");
    true
    System.out.println(esParaulaSolucio("nado","dona");
    true
    System.out.println(esParaulaSolucio("nimfa","fama");
    false   // falta una a
```