

21779. Algorísmia i Estructures de Dades II

Curs 2023-24

ZenWord: La interfície

11 d'abril de 2024

El ZenWord és un joc de paraules que ens ajuda a entrenar el nostre cervell. Es tracta d'intentar endevinar una sèrie de paraules ocultes a partir de les lletres que es poden seleccionar d'un cercle (veure la Figura 1a). Aquestes paraules ocultes són un subconjunt de totes les possibles paraules que es poden formar **únicament** amb les lletres del cercle.

En aquesta sessió pràctica implementarem una part de la interfície del nostre ZenWord (veure la Figura 1b) i les funcionalitats d'alguns dels seus botons: el *Clear*, el *Random*, el *Bonus* i els botons del cercle.

1 La interfície

Primer de tot, implementarem una part de la interfície de la nostra aplicació (veure Figura 1b). Aquesta interfície consta de:

1. Una *ImageView* amb un cercle de color.
2. Un **TextView**, a la part superior, que mostra les paraules que s'han construït i el número possible de paraules que es poden construir.
3. Un *TextView* que mostra la paraula que es va construït amb el conjunt de lletres (veure els exemples *dia* i *manta* a la Figura 1).
4. Un botó per a cada una de les lletres que hi pot haver al cercle (un màxim de 7 lletres).
5. Un botó *Clear* per esborrar la paraula que s'està construït.
6. Un botó *Send* per comprovar si la paraula entrada és vàlida o no.



Figura 1: (a) L’aplicació ZenWord i (b) la implementació que es demana en aquest enunciati.

7. Un botó *Random*, per reordenar la distribució de les lletres del cercle (veure la Figura 2).
8. Un botó *Ajuda* per mostrar una pista de les paraules ocultes.
9. Un botó *Bonus* per mostrar totes les paraules trobades i dur el compte de quants de bonus temim acumulats (el nombre de bonus acumulats es mostrerà al text del botó).
10. Un botó *Reiniciar* per començar una partida nova.

Fixeu-vos en alguns detalls:



Figura 2: La reordenació de les lletres del cercle o shuffle.

- La complexa distribució dels diferents elements a la pantalla fa necessari introduir noves eines que ens facilitin el seu disseny.
- El fons de la pantalla principal té una imatge que el cobreix completament.
- Quatre dels botons (*Random*, *Ajuda*, *Bonus* i *Reiniciar*) tenen una imatge de fons (*background*).
- El número de lletres que surten al cercle depèn de la longitud de la paraula triada (*wordLength*).

1.1 La utilització de GuideLines per fer el disseny de la pantalla més senzill

Les *GuideLines* o guies són línies que no són visibles per als usuaris, però que ajuden als desenvolupadors a restringir els elements de la interfície de forma fàcil, de manera que el disseny pugui ser més clar i interactiu. Es poden utilitzar *GuideLines* horizontals i verticals, que van d'esquerra a dreta i de dalt a baix, respectivament.

Les guies es poden col·locar de tres maneres diferents:

1. Especificant una distància fixa des de la part superior o esquerra de la pantalla.
2. Especificant una distància fixa des de la part dreta o inferior de la pantalla.
3. Especificant un percentatge de l'amplada o alçada de la pantalla.

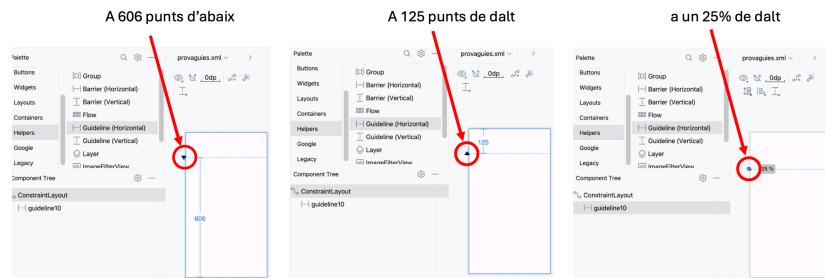


Figura 3: *GuideLines* Horizontals.

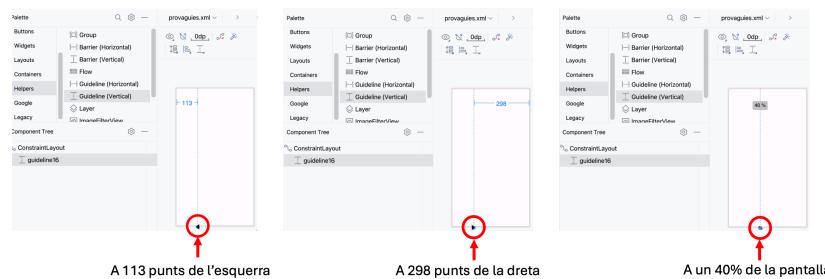


Figura 4: *GuideLines* Verticals.

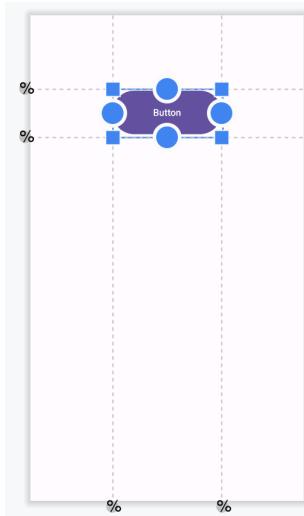


Figura 5: Un botó situat exactament entre un 15% i un 25% d'altura de la pantalla i entre un 30% i un 60% de l'amplada de la pantalla.

Per utilitzar les guies, ho podem fer interactivament (igual que amb els *TextView* o els botons): cal anar a l'apartat de *Helpers* de la paleta de disseny y arrossegar una guia horitzontal o vertical (veure la Figura 3 i la Figura 4). Heu de posar la guia on vulgueu arrossegant-la.

Una vegada situada la guia, veurem que té una fletxa negra que, si hi pitjam a sobre, canvia de direcció (amunt/avall o dreta/esquerra, dependent de si és horitzontal o vertical). Les fletxes ens permeten situar la guia a una distància fixa de les vores. Si tornam a pitjar, apareix un percentatge (%) que permet posicionar la guia a una distància percentual de la pantalla (veure la Figura 3 i la Figura 4).

Quan tenguem totes les guies posades, les podem utilitzar com a referències per determinar les restriccions de posició dels nostres elements (bots, imatges, camps de text,...).

A la Figura 5 es pot veure un botó col·locat definint les seves restriccions (*constraints*) entre quatre guies.

Utilitzant aquestes guies heu de situar els elements de la interfície a les seves posicions per aconseguir un disseny similar al que podeu veure a la Figura 1b. Heu de seguir les següents indicacions:

- El *TextView* que mostra les solucions ha d'estar situat en el 20% superior de la pantalla de la nostra aplicació.
- La imatge del cercle, les lletres del seu interior i els botons dels costats

han d'estar situats a l'àrea del 30% inferior de la pantalla.

- El *TextView* on introduïm la paraula, el boto *Clear* i el botó *Send* han d'estar situats entre el 55% i el 70% de l'alçada de la pantalla.
- Heu de definir interactivament les guies necessàries per situar els botons del cercle i els quatre botons que l'envolten.
- Heu de posar 7 botons al cercle (encara que, durant l'execució, s'hauran d'amagar les lletres que no s'utilitzin). Per amagar i mostrar un botó, o qualsevol element, podeu modificar l'atribut *visibility*:

```
1  Button b;  
2  ...  
3  b.setVisibility(View.GONE);      // l'amaga  
4  ...  
5  b.setVisibility(View.VISIBLE); // el mostra
```

- Heu de posar una imatge com a fons de la pantalla (al *background* del vostre *ConstraintLayout*).

1.2 Com podem fer botons que tenen una imatge com a background?

Primer de tot, hem d'afegir les imatges dels botons (i totes les imatges necessàries) al projecte; per això, hem de posar-les dins la carpeta **res/drawable** (podeu arrossegar les imatges a la carpeta corresponent des del panell *Project*). Com a referència, les imatges del projecte de la captura de la Figura 1 tenen una resolució de 250×250 píxels (la imatge del bonus és una mica més alta).

Una vegada afegides les imatges al projecte, podem seleccionar el botó corresponent des de la vista de disseny i modificar-ne l'atribut **background** seleccionant la imatge que volem utilitzar com a background del botó.

Les noves versions d'Android Studio, per a moltes de les seves plantilles, fan que el projecte utilitzi la llibreria d'Android *Android Components*, configurant el tema predeterminat per basar-se en *Theme.MaterialComponents*. Un efecte secundari d'això és que qualsevol element `<Button>` es converteix en un widget *MaterialButton* enllot d'un widget *Button* normal, fent que s'ignori l'atribut **Android:background**. Per solucionar-ho, podeu canviar l'element XML del disseny perquè sigui `<android.widget.Button>` en comptes de `<Button>` (per modificar l'xml del disseny, heu de seleccionar l'opció *Code* enllot de *Design*).

1.3 Quina funcionalitat ha de tenir el botó *Clear*?

Quan es pitgi el botó *Clear*, s'ha d'esborrar el contingut de la paraula construïda i la seva visualització en el *TextView* (canviant el valor del seu atribut *text* amb la instrucció *setText()*).

1.4 Quina funcionalitat han de tenir els botons del nostre cercle?

Cada vegada que es pitja sobre algun dels botons de les lletres (els que contenen les lletres) s'ha d'afegir la lletra del botó corresponent a la paraula que es va constraint i s'ha de visualitzar dins el *TextView* corresponent (a la Figura 1b, s'ha construït la paraula MANTA). Per tant, veim que tots aquests botons tenen la mateixa funcionalitat i, per això, seria lògic que tenguessin el mateix mètode a l'atribut **onClick()**:

```
1 public void setLletra (View view) {}
```

Aquest mètode hauria de saber quin ha estat el botó que s'ha pitjat per tal de poder afegir a la paraula la lletra del boto pitjat; però, com podem saber quin botó ha estat el que s'ha pitjat si tots tenen el mateix mètode a l'atribut **onClick()**? Aquesta informació la podem obtenir del paràmetre *view* del mètode, d'on podem agafar, per exemple, el text del botó que ha desencadenat l'acció:

```
1 public void setLletra (View view) {
2     // Recuperam el botó que ha desencadenat la crida
3     Button btn = (Button) View;
4     // Podem agafar, per exemple, el text del botó
5     String lletra = btn.getText().toString();
6     ...
7 }
```

Cada vegada que es pitja una lletra, aquesta lletra es desactiva i canvia el seu color per mostrar que no es pot tornar a pitjar (veure la Figura 1a i la Figura 1b, on s'observa la diferència de color de les lletres ja utilitzades per la paraula que s'està constraint).

2 Quina funcionalitat ha de tenir el botó *Random*?

Per a realitzar una reordenació o *shuffle* de les lletres visibles del cercle, podem utilitzar l'algorisme de **Fisher-Yates**, que genera una permutació

aleatòria d'una seqüència finita (en el nostre cas, podem generar un array a partir de les lletres que es mostren al cercle).

L'esquema de l'algorisme és:

- Començant pel darrer element de l'array, per a cada element i :
 1. Seleccionar un índex aleatori, j , entre 0 i l'element anterior a l'element i .
 2. Intercanviar els elements dels índexs i i j .

3 Quina funcionalitat ha de tenir el botó *Bonus*?

Quan es pitgi el botó *Bonus*, s'ha de mostrar una finestra emergent amb la llista de les paraules que s'han trobat (veure Figura 6).

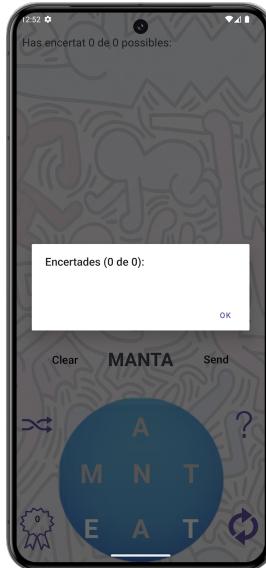


Figura 6: El diàleg dels bonus.

En Android, aquestes finestres emergents es poden implementar amb la classe *AlertDialog*. Hem de crear un *AlertDialog* que mostri un text informatiu amb el nombre de paraules construïdes, el nombre total de totes les paraules que es poden construir i la llista (**ordenada alfàbeticament**) de totes les paraules que s'han construït (veure la Figura 6). De moment, mostarem un text estàtic.

El següent codi mostra el procés de creació d'aquest element:

```
1 ...
2 AlertDialog.Builder builder = new AlertDialog.Builder(this);
3 ...
4 builder.setTitle(<encertades i possibles>);
5 builder.setMessage(<la llista de trobades>);
6 // Un botó OK per tancar la finestra
7 builder.setPositiveButton("OK", null);
8 // Mostrar l'AlertDialog a la pantalla
9 AlertDialog dialog = builder.create();
10 dialog.show();
```