Jeu du pendu

Session 2

14 juin 2010 : Durée 2h30

Le but de ce TP est d'écrire un jeu de pendu. Pour rappel : le joueur doit trouver un mot en proposant des lettres. Lorsqu'une lettre proposée apparaît effectivement dans le mot à deviner toutes ses occurences sont affichées. Le joueur a droit à 5 erreurs, si la lettre n'apparaît pas le nombre d'erreurs qu'il lui reste est décrémenté. Le joueur perd quand le nombre d'erreurs restantes est égal à 0.

Le joueur gagne, s'il a trouvé toutes les lettres du mot. (Nous ne vous demandons pas de dessiner un vrai pendu comme c'est l'habitude dans ce jeu.)

Pour accéder au site officiel d'`android developer`, nous vous conseillons d'utiliser `iceweasel` en configurant correctement le proxy (`cache.informatique.univ-paris-diderot.fr`, port 3128).

I) Modalités pratiques

Vous rendrez votre travail à travers l'interface didel, sous la forme d'une archive compressée de vos projets android. Nous vous encourageons à ne pas attendre le dernier moment pour soumettre. Les soumissions ne seront plus acceptées à partir de 12h10. Nous ne tiendrons pas compte des soumissions par mail.

II) Mise en place du *layout*

Exercice 1 :

Mettez en place un *layout* semblable à celui de la figure 1 comprenant :

- un affichage du nombre d'erreurs possibles restantes,
- un affichage des lettres déjà devinées (les lettres non devinées seront représentées par le caractère -),
- un champ de saisie,
- un bouton de soumission.

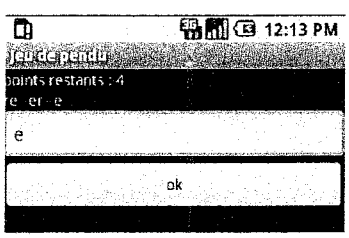

FIG. 1 – mise en page du jeu, ici le mot à chercher est "recherche" et les lettres testées sont 'e', 'a' et 'r', on a donc déjà perdu 1 point

III) Activité principale : mise en place du jeu

Le mot à deviner sera mis en dur dans le programme. Le mot partiellement deviné pourra être une variable dont tous les caractères seront initialement -. Il pourra, par exemple, être stocké sous forme de tableau de caractères et n'être converti qu'au moment de l'écriture.

Exercice 2 :

1. Mettez en place le `onCreate` de votre activité principale : le bouton de soumission doit être associé à un contrôleur.
2. Écrivez une méthode `miseAJour` qui prend en argument une lettre, met à jour le mot deviné s'il y a lieu et renvoie une valeur indiquant si la lettre apparaît ou non dans le mot à deviner.
3. Écrivez le contrôleur associé au bouton de soumission : quand une lettre est proposée :
   - si la lettre proposée apparaît dans le mot à deviner elle est placée où il faut,
   - sinon le nombre d'erreurs restantes est décrémenté.
   - si après décrémentation, le nombre d'erreurs restantes est égal à zéro ou que le mot est entièrement trouvé, l'activité se termine (on dit que *la partie est terminée*). Dans un premier temps on se contentera d'écrire "perdu" ou "gagné" à la place du nombre d'erreurs restantes.

IV) Activité secondaire : on recommence ?

Exercice 3 :

1. Créez une deuxième activité qui est lancée lorsque la partie est terminée.
2. Cette deuxième activité affichera **perdu** ou **gagne** suivant le cas et proposera au joueur de rejouer. Mettez en place ce fonctionnement.
3. En utilisant le code de retour de l'activité secondaire au retour dans l'activité principale, faites en sorte que cette dernière se termine ou propose un autre mot à deviner à l'utilisateur.

V) Jeu : prise en compte des lettres déjà testées

Modifiez l'activité de jeu pour les lettres déjà proposées soient affichées dans l'ordre alphabétique comme dans la figure 2.

De plus, si on propose une lettre pour la deuxième fois, on ignorera cet essai.

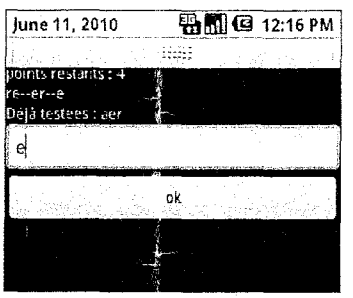

FIG. 2 – le mot à chercher est toujours "recherche", les lettres testées sont 'e', 'a' et 'r', on a donc déjà perdu 1 point