Université Paris Diderot INTERFACES GRAPHIQUES 20 mai 2018 11h45-13h45

Les seuls supports autorisés :
documentation API java `http://docs.oracle.com/javase/8/docs/api/index.html`,
documentation API JavaFX : `http://docs.oracle.com/javase/8/javafx/api/toc.htm`
les tutoriels de javaFX d'Oracle : `http://docs.oracle.com/javase/8/javase-clienttechnologies.htm` et
`https://docs.oracle.com/javase/8/javase-books.htm`
et les transparents et programmes étudiés en cours :
`https://moodlesupd.script.univ-paris-diderot.fr`
L'utilisation de toute autre ressource web (en particulier des forums de développeurs), de moteur de recherche (google et autre), de mail ou d'autre moyens de communication sera traitée comme une fraude. Les documents papier autorisés : vos notes de cours ou TP, les photocopies de transparents de cours, notes personnelles. Livres et photocopies de livres interdits. Vos mobiles doivent être impérativement éteints. L'écran de votre voisin n'est pas une ressource autorisée.
Le sujet comporte 5 pages.

Consignes

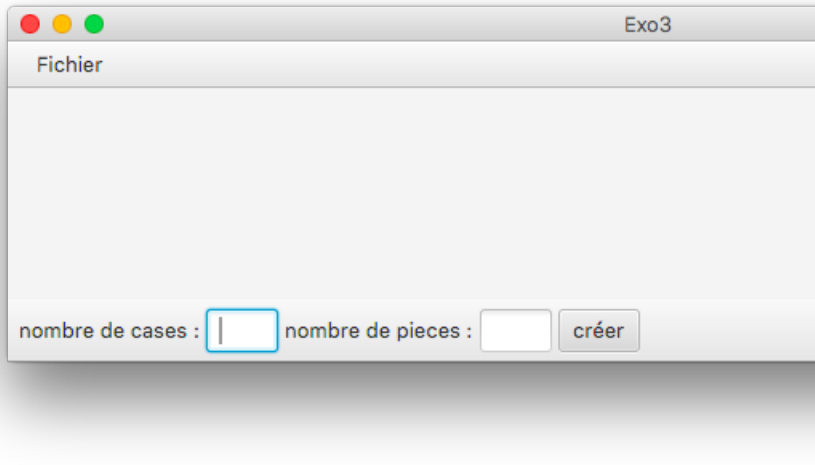
Votre classe principale doit porter votre nom, par exemple pour un dénommé Mathias Dupont la classe principale s'appellera `DupontMathias`.
Chaque exercice est un programme à développer découpé en plusieurs questions. Parfois les questions sont indépendantes, c'est-à-dire on peut faire la question $i+1$ même si on a pas fait la question i.
Dans votre programme mettez en commentaire l'information quelles questions vous avez implémentées.
Il est possible de déposer plusieurs versions numérotées (DupontMathiasExo1a, DupontMathiasExo1b etc.) du même exercice. Cependant c'est la dernière version qui sera examinée pour noter votre travail. Si pour une raison quelconque vous voulez que j'examine aussi d'autres versions faites un fichier README avec un message approprié.

Si vous developpez sur NetBeans le plus simple c'est exporter le programme vers zip (File → Export Projet → to Zip) et déposer le fichier zip avec le projet complet. Sinon vous pouvez aussi faire zip ou tar contenant juste les fichiers sources.
Le fichier d'archive doit être déposé sur moodle, cours Interfaces graphiques, dans le dépôt d'examen groupe 2.
N'attendez pas la fin de l'examen pour déposez vos programme, le dépôt sur moodle ferme à 13h45 pile.

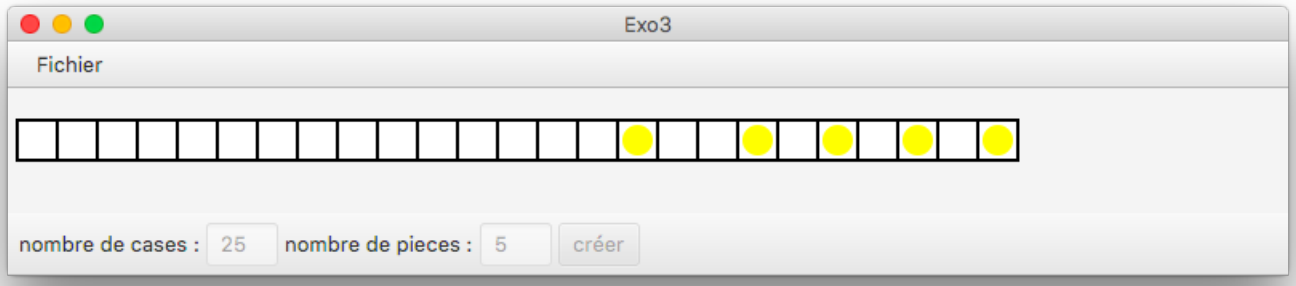
Vos programmes seront examinés par un logiciel de détection de plagiat efficace.
Le dépôt sur moodle est protégé donc c'est moi et vous-même qui auront l'accès à votre programme et personne d'autre.
C'est à vous de protéger votre répertoire de travail sur les machines UFR contre les visites non autorisées.
Dans tous les exercices et question, si vous avez à implémenter un interface avec une seule méthode vous devez utiliser plutôt une lambda expression. L'utilisation de classes anonymes (ou non anonymes) là où une lambda expression est suffisante entraînera la diminution de la note.

Exercice 1

Question 1: Écrire une application qui comporte initialement un Menu avec un seul item qui permet de quitter l'application, un `Pane` vide au centre un `ToolBar` en bas. Dans le ToolBar il y a plusieurs éléments comme sur cette image :
L'utilisateur écrit dans les deux champs texte deux nombres entiers positifs (nombre de cases et nombres de pièces) et clique sur le bouton `créer`
Si les deux nombres sont positifs alors votre programme les mémorise (dans des variables) et désactive (disable) les deux champs texte ainsi que le bouton `créer` pour que l'utilisateur ne puisse plus les utiliser.
Si un nombre est négatif ou zéro videz les deux champs texte et recommencez.



Question 2: Implémentez le comportement suivant. Quand l'utilisateur clique sur le bouton `créer` et les deux nombres a, b dans les champs texte sont positifs le programme construira a cases carrés dans lesquelles on place b pièces d'or :



On note que les cases (rectangles) sont des carrés c'est-à-dire

$$\text{largeur} = \text{hauteur} = \text{L}$$

où L une constante de votre choix déclarée dans votre programme.
Le bord de chaque rectangles est noir (de largeur 2 pixels), l'intérieur est blanc.
Les rectangles sont alignés (même valeur fixe de la propriété `y` pour chaque rectangle et la valeur de la propriété `x` augmente de L d'un rectangle au rectangle suivant).
Sur moodle vous trouverez le code de la fonction

```
static int[] choix(int nbC, int nbP)
```

Le paramètre nbC est le nombre de carrés, le paramètres nbP le nombre de pièces d'or à placer. La fonction retourne un tableau trié de int qui contient nbP éléments. Le tableau contient les indices de cases dans lesquelles on mettra les pièces d'or (cercles jeunes).
Exemple. `choix(20,4)` retournera un tableau trié de 4 entiers entre 0 et 19. Par exemple `[2,8,11,14]` indique qu'il faut poser les pièces sur les cases 2,8,11 et 14.

Chaque pièce d'or doit être centrée dans sa case. (Facile de voir que pour placer le centre du cercle `centerX, centerY` au centre de la case il faut que le centre satisfasse

```
centerX = x + width/2, centerY = y + height/2
```

où `x,y,width, height` les propriétés du rectangle qui contient le cercle.
Le rayon de chaque pièce d'or doit être tel que la pièce doit (presque) remplir la case mais sans débordement.

Question 3: Les cases et les pièces servent à jouer le jeu *silver coin*.
L'utilisateur clique sur une case libre. Cela provoque le déplacement de la pièce d'or la plus proche à droite de cette case vers la case sélectionnée par le clic de l'utilisateur.
Implémenter ce comportement.
Si toutes les pièces sont à gauche le jeu est terminé.

Question 4: Ajouter dans le menu un item `rejouer` qui supprime tout le contenu du Pane et on débloque les champs texte et le bouton `créer` dans le ToolBar. On pourra maintenant récréer une nouvelle configuration du jeux.

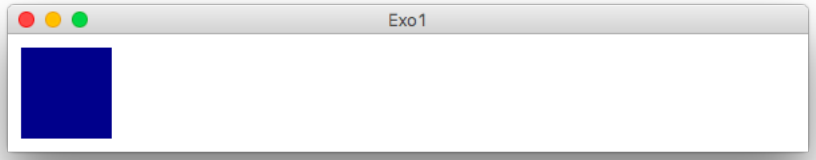
Exercice 2

Cet exercice sert à tester vos connaissances de bindings. Si vous le faites sans bind alors l'exercice vous apportera **très très peu de points**.

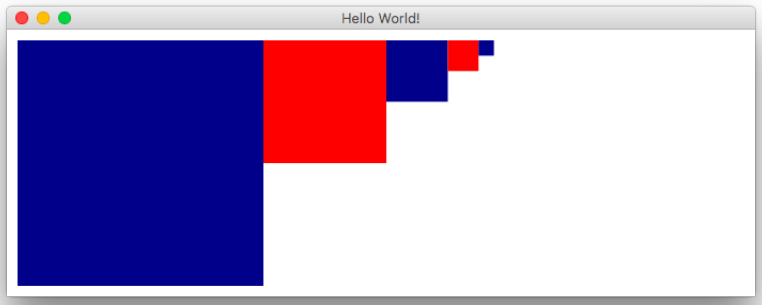
Question 1: Placer un Rectangle dans un Pane. Les propriétés `x` et `y` du rectangle qui donnent sa position dans le Pane (la position du premier sommet de rectangle) dans le Pane sont fixes (par exemple DX et DY pixels où DX, DY constantes de votre choix).
Par contre la hauteur du rectangle doit s'adapter à la hauteur du Pane qui le contient selon la formule :

```
la hauteur du rectangle = la hauteur du Pane - 20
```

La hauteur (height) du rectangle doit automatiquement s'ajuster à la hauteur du Pane, si l'utilisateur agrandit la fenêtre la hauteur du rectangle augmente, si l'utilisateur rétrécit la fenêtre la hauteur du rectangle diminue, la valeur du hauteur du rectangle est à chaque moment donnée par la formule ci-dessus.
La largeur du rectangle doit être toujours égale à la hauteur (le rectangle est toujours un carré).



Question 2: Ajouter d'autres rectangles (au moins 4). Les couleurs de rectangles sont de votre choix mais doivent alterner (un rouge, un bleu, un rouge, un bleu etc. par exemple). La hauteur et la largeur de chaque rectangle doivent être toujours égales à la moitié de la hauteur et la moitié de la largeur du rectangle précédent. (En particulier tous les rectangles changent la taille quand le premier change la taille avec le changement de la taille de la fenêtre).
Les rectangles doivent être toujours alignés comme ici :



donc il faut adapter automatiquement les valeurs de propriétés `x` et `y` de chaque rectangle, sauf le premier, quand la taille de la fenêtre change.