Thomas
Nicolle

Vendredi 30 Avril 2021

Rapport final

1. Arborescence

jFrame « Le Jeu Loto » (BORDER_LAYOUT)
- jMenuBar
  - jMenu Ajout
    - jMenuItem AjoutJoueur
    - jMenuItem AjoutLot
  - jMenu Action
    - jMenuItem optionsJeu
    - jMenuItem afficheLots
    - jMenuItem AcheterCartes
    - jMenuItem runJeu
    - jMenuItem resumeJeu
    - jMenuItem retryJeu
- jLabel « Voici les cartes dont vous disposez » (CENTER)
- jPanel CENTRE (BORDER_LAYOUT)
  - jPanel LesCartons (GRID_LAYOUT(2,4))
    - jPanel j1Carton1
    - jPanel j1Carton2
    - jPanel j2Carton1
    - jPanel j2Carton2
    - jPanel j3Carton1
    - jPanel j3Carton2
    - jPanel j4Carton1
    - jPanel j4Carton2
  - jPanel LesJoueurs (GRID_LAYOUT(1,4))
    - jButton j1
    - jButton j2
    - jButton j3
    - jButton j4
  - jLabel Cadeau
- jPanel SUD (GRID_LAYOUT(2,2))
  - jLabel jLabel2« Historique du tirage des boules »
  - jLabel Option « Option choisie : »
  - jScrollPane jScrollPane1
    - JTextArea historique
  - jPanel jPanelTirage (GRID_LAYOUT(3,1))
    - jButton Tirage « Tirage »
    - jLabel NbrTirage
    - jButton STOP « Arrêter »

Thomas
Nicolle

Vendredi 30 Avril 2021

2. Attributs, Constructeurs & Méthodes

Attributs

L'application principale s'organise autour dix variables nécessaires à son bon fonctionnement.

```
private int option; // option
private int nbCol; // nombre
private int nbNum; // nombre
private boolean carteAchetee;
final private int MAX = 90; /
private int boules[]; // poin
LesJoueurs lstJ; // Sert à st
LesJoueurs Gagnants; // liste
LesLots lots; // liste des lo
Lot lot; // lot à gagner
```

Tout d'abord, trois variables de type entier servent à la configuration du jeu: « option » permet de savoir quel est le mode jeu sélectionné par l'utilisateur (1.Quine, 2.Double Quine, 3.Carton Pleins) , « nbCol » et « nbNum » servent à définir la taille des cartes lotos proposées à l'achat. Un booléen appelé « carteachetee » permet d'indiquer si au moins une carte a été acheté par un des joueurs pour permettre de lancer la partie par la suite. En ce qui concerne les tirages de numéros, cela est d'abord fixé par la variable d'entier « MAX » définit à 90 par défaut pour indiquer la valeur maximum pouvant être tiré; un tableau d'entier « boules » permet de servir de pointeur pour ne pas tiré deux fois le même nombre. Aussi, l'application est composé de deux instances LesJoueurs permettant d'avoir une collection d'instance Joueur nécessaire à la partie mais aussi pour lister les gagnants et tiré au sort un unique joueur. Enfin, une instance LesLots similaire à LesJoueurs permet quand à elle de regrouper les lots qui sont à gagnés et aussi une instance Lot pour récupérer le lot choisis aléatoirement à gagné au cours d'une partie.

Constructeur

Premièrement, la fenêtre principale est construite grâce à la méthode initComponents() qui va permettre en autre d'instancier tous les éléments, la disposition ainsi que leur attributs. Ensuite une méthode initJoueurs() est appelée, servant à instancier et définir la liste des joueurs (lstJ) et d'y ajouter deux joueurs de bases (Jack et Lara) via AjoutJouer() et de leur définir une couleur et une image via les méthodes setCouleur() et setPhoto(). Une deuxième méthode initButtons() est ensuite utilisé afin de rafraîchir les boutons associés aux joueurs de la liste pour y afficher leur pseudonyme si non présent en comparant l'attribut du bouton i et le pseudo du joueur i. Puis les trois attributs de configuration (option, nbCol, nbNum) sont initialisés avec des valeurs par défaut afin d'éviter toutes erreurs. Le tableau boules est ensuite instancié avec en paramètre la variable MAX et initialisé avec les valeurs zéro via la méthode initBoules(). Enfin, la collection des joueurs gagnants est aussi instanciée, l'indicateur de carte achetée initialisé à false. Aussi la liste des lots est instances avec trois lots de bases.

3. Déroulement du jeu

Avant le démarrage

Avant de lancer une partie, 3 étapes sont nécessaires:

Premièrement, il faut si besoin ajouter un/deux joueur(s) supplémentaire en cliquant dans le menu « Ajout » puis « joueur » qui affichera une instance AjoutJoueurDlg (jDialog) permettant de remplir les informations du nouveau joueur: pseudo et couleur étant donnée que le solde est fixé à 20 et non modifiable, ce qui modifiera l'attribut Joueur de la boite de dialogue. Si le joueur décide de valider son choix, l'application principal récupérera l'instance Joueur via la méthode getJoueur() et l'ajoutera à la collection. Une intention de rajouter un 5ème joueur affichera un message d'erreur via une instance jOptionPane.

Deuxièmement il faut configurer la partie en cliquant sur le menu « Action » puis « Options Jeu », ce qui va permettre d'afficher une instance OptionDlg (dérivée de jDialog) avec en paramètre les variables « option », « nbCol » et « nbNum ». L'utilisateur pourra choisir à l'aide de bouton radio parmi trois règles du jeu ainsi que le nombre de colonnes (1 à 5) via une ComboBox ainsi que le nombre de numéros et enfin Valider, qui permettra de modifier 3 variables de la boite de dialogue et de leur attribuer les valeurs choisi et de fermer la fenêtre ou alors Annuler qui ne prendra pas en compte les modifications. Si validation, l'application principale utilisera les

méthodes de la jDialog pour récupérer les valeurs de ses attributs et de les appliquer aux attributs de la classe principale.

Troisièmement, le(s) joueur(s) peuvent acheter deux cartes chacun via le menu Action puis « Acheter cartes » ce qui affichera une instance AchatCarteDlg (dérivée de jDialog) avec en paramètre la liste des joueurs (lstJ), la variable pour le nombre de colonne (nbCol), 3 puisque le nombre de lignes est fixé dès le début et non modifiable et enfin le nombre de numéro présent sur la carte (nbNum). Une fois la fenêtre affiché, les variables en paramètre seront récupéré attribué aux variables locales, la liste des joueurs sera affiché dans la ComboBox en récupérant leur pseudo et le solde du premier joueur sera affiché avec un jLabel. Enfin une carte loto sera généré via une méthode initCarte() qui reconstruira dynamiquement une carte loto avec des boutons sur une grille de dimension correspondant à celle récupéré en paramètre et les valeurs seront affichés sur les boutons. Si le joueur achète une carte, un indicateur boolean sera initialisé à true. Enfin, l'application principal récupèrera le booléen de la boite de dialogue et si la carte est achetée (true) la méthode afficheCartes() s'exécutera pour l'afficher dans l'application principale. AfficheCartes() permet entre autre de choisir dynamiquement la position ou dessiner la carte avant de l'effectuer via la méthode dessineCarte() pour l'effectuer sur le jPanel sélectionné avec la couleur du joueur ainsi que les valeurs de cette carte détenu.

Si besoin le joueur peux afficher les lots disponibles en cliquant dans le menu « Action » puis « Afficher lots » ce qui ouvrira une instance VisuLots(dérivé de jDialog) avec en paramètre la collections des lots. Un affichage dynamique des images des lots sera effectué en fonction du nombre. Il peux aussi en rajouter via le menu « Ajout » puis « Lot » ce qui ouvrira une instance AjoutLot pour choisir une description, le niveau, la catégorie et en fonction de cette dernière attribuer une image ou un prix.

Pendant le jeu

Une fois que le jeu est parametré, les joueur ajoutés et les cartes achetées, le joueur peut alors cliquer sur « Démarrer Jeu » dans le menu « Action », ce qui lui permettra de Tirer des nombres via le bouton. A chaque clique sur ce bouton un évènement est déclenché, un nombre est généré de manière aléatoire compris entre zéro et la valeur de la variable MAX tant que ce dernier n'a pas été tiré. Lorsque il n'est pas déjà tiré, la valeur est affiché dans l'historique et la valeur 1 est attribué à case i correspondant au nombre dans le tableau boules ce qui permet de savoir que la valeur a déjà été sortie.

Ensuite en parcourant toute la liste des joueurs, on vérifie si le numéro est présent sur les cartes des joueurs, et si c'est le cas on passe un booléen à true et l'on simule le placement d'un pion via la méthode placePion() et l'on rafraichit l'affichage des cartes.

Si le booléen a été activé, c'est pour permettre de vérifier si les conditions pour gagner la partie on été remplie en parcourant les cartes de tous les joueurs. Si la condition est réalisé, on affiche dans l'historique un message qui l'indique et on ajoute le joueur dans la liste des gagnants.

Enfin, si plusieurs joueurs ont remplis les condition, on génère un nombre entre zéro et le nombre de gagnant présent dans la collection pour choisir un indice et donc tiré le joueur au sort. Cependant si un seul joueur a remplis la condition, il gagne la partie et un message est aussi affiché dans l'historique et impossible de tirer un autre nombre.

Poursuivre le jeu

Une fois la partie gagné, il est possible de continuer le jeu et de passer à l'option suivante sauf si l'option choisis est le carton plein ou alors tous les lots ont été gagné. Cela aura pour effet de modifier l'option du jeu à la suivante, de choisir aléatoirement le nouveau lot en fonction de la nouvelle option et de pouvoir tirer des nombres à nouveau pour finir la partie.

Fin de partie

Thomas Vendredi 30 Avril 2021
Nicolle

Afin de remporter la partie, il est nécessaire de vérifier les conditions à chaque tirage. Pour cela, le programme doit vérifier pour chaque carte de chaque joueur si le carton est gagnant en fonction de l'option du jeu via la méthode cartonGagnant(option). Lorsque un joueur remplis la condition, un indicateur est initialisé à true, le joueur est ajouté à la collection des gagnants, il ne peux plus effectuer un nouveau tirage et un message indiquant sa réussite est affiché dans l'historique. Enfin comme au moins un gagnant est présent dans la liste indiqué par l'indicateur booléen, il faut vérifier si plusieurs joueurs ont remplis la condition de carton gagnant pour effectuer un tirage au sort, afin de désigner le gagnant final et lui attribuer le lot sinon aucun tirage n'est effectué et le seul gagnant remporte la partie ainsi que son lot.