Le jeu du pendu



Objectifs

- Compléter le développement en JAVA d'une application avec IHM
- Commencer à apprendre à développer une application avec méthode

L'objectif est de terminer le développement d'une application pour jouer au jeu du pendu, un jeu dans lequel il faut retrouver un mot dont les lettres sont cachées. À chaque tour de jeu l'utilisateur propose une lettre. Si elle se trouve dans le mot, toutes les occurrences de la lettre dans le mot sont dévoilées sinon un élément supplémentaire du pendu est dessiné. Si le pendu est entièrement dessiné le joueur a perdu la partie.

L'archive pendu_pour_etu.zip disponible sur Celene contient les fichiers utiles à ce TP.

- Le répertoire src contient les fichiers JAVA du projet. Dans ce projet tu devras compléter certaines classes d'autres te sont données. Les méthodes à implémenter sont indiquées dans les commentaires.
- Le répertoire doc contient la documentation JAVA (Javadoc) du projet, tu peux la consulter via un navigateur web.
- Le répertoire imp contient les images utilisées par l'application.

En Annexe, tu trouveras le diagramme de classes, les maquettes et la liste des tâches à réaliser.

Exercice 1 Comprendre le projet

- Identifier toutes les classes qui constituent le Modèle, celles qui constituent la Vue et celles qui constituent les Contrôleurs
- 1.2 Complète le diagramme de classe avec les associations unidirectionnelles et les héritages.

Exercice 2 Coder le projet avec méthode

Les tâches à réaliser sont données en annexe. Certaines sont suffisamment détaillées pour être utilisées dans un sprint, d'autres non.

- 2.1Ajoute éventuellement d'autres tâches. Priorise les tâches à effectuer.
- Parmi les tâches prioritaires, découpe celles qui ne sont pas suffisamment détaillées. 2.2
- 2.3Parmi les tâches prioritaires, choisis-en cinq puis effectue les.
- 2.4 Reprends à la question 1 jusqu'à ce qu'il ne reste plus aucune tâche à effectuer

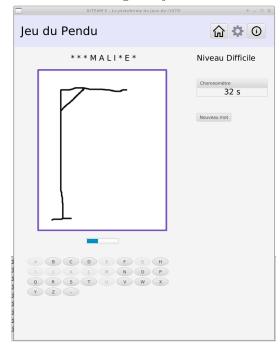
Remarque : il sera nécessaire d'ajouter des méthodes pour rendre le code plus lisible et donc plus maintenable. Certaines méthodes sont en commentaires dans le code source de la classe Pendu : il s'agit se suggestions de méthodes auxiliaires que tu peux créer.

${\bf Annexe: les\ maquettes}$

Page d'accueil

Page de jeu

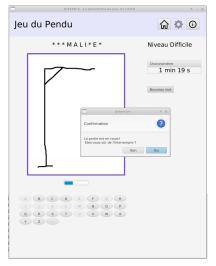


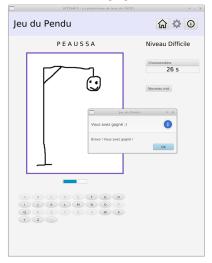


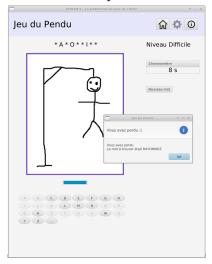
Nouveau mot

Partie gagnée

Partie perdue







Annexe : diagramme de classes Modele Modele MotMystere Dictionnaire + FACILE :int; + MOYEN :int; + DIFFICILE :int ; + Dictionnaire(nomFichier:String, + EXPERT :int; longueurMinDesMots:int, longueurMaxDesMots:int) + MotMystere(motATrouver :String, niveau :int, nbErreursMax :int) + choisirMot() : String + MotMystere(nomFichier :String, longMin :int, longMax :int, niveau:int, nbErreursMax:int) + initMotMystere(motATrouver:String, niveau:int, nbErreursMax:int):void + getMotATrouve() :String + getMotCrypte() :String getLettresEssayees() :Set<String> getNbLettresRestantes():int+ getNbEssais() :int ControleurLettres getNbErreursMax():int getNbErreursRestants():int Controleur getNiveau():int setMotATrouver(motATrouver:String):void setMotATrouver():void setNiveau(niveau:int):void ControleurParametres essaiLettre(lettre : char) :int perdu():boolean Controleur gagne():boolean Vue ControleurLancerPartie TilePane Controleur ControleurNiveau Clavier - clavier :List<Button>; Controleur + Clavier(touches:String, actionTouches: EventHandler < ActionEvent >) + desactiveTouches(touchesDesactivees :Set<String>) 1 Pendu Text Vue + init() : void+ start(stage :Satge) :void - laScene() :Scene - titre():Pane Chronometre Vue - chargerImages(repertoire :String) :void + modeAccueil() :void + modeJeu() :void + Chronometre() + modeParametre():void + setTime(tempsMillisec :long) + lancePartie() :void + start() :void + majAffichage() :void + stop() :void + getChrono() :Chronometre + resetTime() :void + popUpReglesDuJeu() :Alert + popUpMessageGagne() :Alert + popUpMessagePerdu() :Alert ${\bf ControleurChronometre}$ Controleur ControleurInfos Application Controleur

Nom:	Groupe :
------	----------

Liste des tâches à effectuer

Mettre en place le projet (projet VSCode + dépot git)
Coder la classe ControleurLettres
Coder la classe ControleurParametres
Coder la classe ControleurLancerPartie
Coder la classe ControleurNiveau
Coder la classe ControleurChronometre
Coder la classe ControleurInfos
Coder la classe Clavier
□ Coder le constructeur de Clavier
□ Coder la méthode desactiveTouches() de Clavier
Coder la classe Chronometre
Coder la classe Pendu
□ Afficher la page d'accueil
□ Afficher la page de jeu
$\hfill\Box$ La barre de progression doit indiquer l'avancement du jeu
□ Coder la méthode init() de Pendu
□ Coder la méthode titre() de Pendu
□ Coder la méthode modeAccueil() de Pendu
□ Coder la méthode modeJeu() de Pendu
□ Coder la méthode titre() de Pendu
□ Coder la méthode leChrono() de Pendu
□ Coder la méthode majAffichage() de Pendu
□ Coder la méthode getChrono() de Pendu
□ Coder la méthode popUpReglesDuJeu() de Pendu
□ Coder la méthode popUpMessageGagne() de Pendu
□ Coder la méthode popUpMessagePerdu() de Pendu
□ Coder la méthode modeParametres() de Pendu
Rendre tous les boutons fonctionnels
Rendre le chronomètre
Ajouter une infobulle sur les bouton "Home" , "Parametres" et "Info"
Griser le bouton "Home" sur la page d'acceuil
Griser le bouton "Parametres" sur la page de jeu
Refactoriser le code
Faire en sorte que les lettres déjà proposées par l'utilisateur soient désactivées
Gérer la fin de partie (gagné ou perdue)
Ajouter la possibilité de jouer à un autre jeu : le démineur

Remarque : je rappelle que "coder une méthode" inclut également "écrire sa documentation"