Cave Race

Parcourir une caverne à l'aide d'un vaisseau sans heurter les parois.

Dans plusieurs méthodes, *w* désigne la largeur, *h* la hauteur et *g* le Graphics du DrawPanel. *pWidth* ou *width* désignent la largeur de la grotte.

Gate

Il s'agit d'une petite partie de la grotte, de couleur grise.

x1, x2	position horizontale du début et de la fin de l'ouverture
constructeur (pWidth, pX1, pX2)	*pX1*, *pX2* : *x1* et *x2* du *gate* précédent Quand un gate est créé, c'est pour le mettre à la fin de la liste, mais il apparaît tout en haut sur le dessin. Son ouverture, définie par x1 et x2, est calculée à l'aide de la méthode *init(...)* à qui elle passe ses paramètres.
draw(g, height, ypos)	Pour dessiner l'objet il faut dessiner un rectangle gris de hauteur height et à la position verticale ypos. Le rectangle commence à la position horizontale *x1* et s'arrête à la et la position horizontale *x2*.
init(pWidth, pX1, pX2)	Recalcul de x1 et x2 à partir des paramètres donnés : Pour calculer x1, on prend pX1+2 et on y additionne un nombre aléatoire entre -13 et +13. Pour calculer x2, on prend pX2-2 et on y additionne un nombre aléatoire entre -13 et +13. Il faut que *0<=x1* et *x1+64<=x2<pWidth*. Sinon il faut recalculer les attributs x1 et x2 jusqu'à ce que ces conditions soient remplies.

Cave

La grotte avec la liste des gate et la position de la navette.

gateList	Liste d'objetc du type *gate.* Le *gate* dessiné tout en haut est le dernier de la liste. Le *gate* numéro 3 est celui qui est à la hauteur du vaisseau.
width	largeur de la grotte
shipXPos	position x du milieu du vaisseau
score	le score, initialisé avec 0
Cave(w, pGateCount)	*width* (largeur de la grotte) est défini par *w* (largeur du DrawPanel). *shipXPos* est défini à la moitié *w* Même si la largeur du drawPanel peut varier, la largeur de la grotte reste fixe. Le constructeur ajoute pGateCount objets du type *gate* dans la liste. Le premier gate est créé par l'appel *Gate(width,0,width). Chaque autre gate* se base sur le *gate* qui le précède.
draw(g, caveHeight)	Dessine la grotte. h est la hauteur de la grotte d'où on peut calculer la hauteur et la position verticale de chaque *gate*. Ensuite on dessine le vaisseau qui est composé de 3 ovals qui se chevauchent. La largeur maximale est de 30 pixels. Le vaisseau est dessiné de manière à ce que cette largeur maximale se situe au niveau du 4^{e} gate à partir du bas. A la fin on dessine le score en haut à gauche.
step()	A chaque step, le premier gate (dessiné tout en bas sur le dessin) est enlevé de la liste et ré-ajouté à la fin de la liste (dessiné tout en haut sur le dessin). Il est est en plus redéfini su la base des données de l'avant-dernier gate, à l'aide de la méthode *init(...)* selon les descriptions données plus haut. Le score est incrémenté. Finalement il faut tester si le vaisseau touche le mur. (On ne contrôle que si le vaisseau touche le 4^{e} *gate* de la liste. La méthode retourne faux si le vaisseau touche, sinon vrai.

reset()	Cette méthode permet de recommencer après un crash. Elle redéfini donc le *score*, *shipXPos*, et la *gateList*.

DrawPanel

cave	Le même objet *Cave* que celui du *MainFrame*
paintComponent(g)	efface le DrawPanel et dessine *cave* s'il n'est pas null

MainFrame

cave	objet de la classe Cave initialisé avec 30 objets du type *gate*
timer	Chrono démarré au début avec une fréquence de 25 fois par seconde, sert aux déplacements du vaisseua dans la grotte. Lié au bouton *stepButton*.
drawPanelMousePressed	faire un reset et relancer le timer, uniquememt après un crash
drawPanelMouseMoved	positionner la navette selon la position x de la souris
stepButtonActionPerformed	faire un "step". Arrêter le timer dans le cas d'un crash