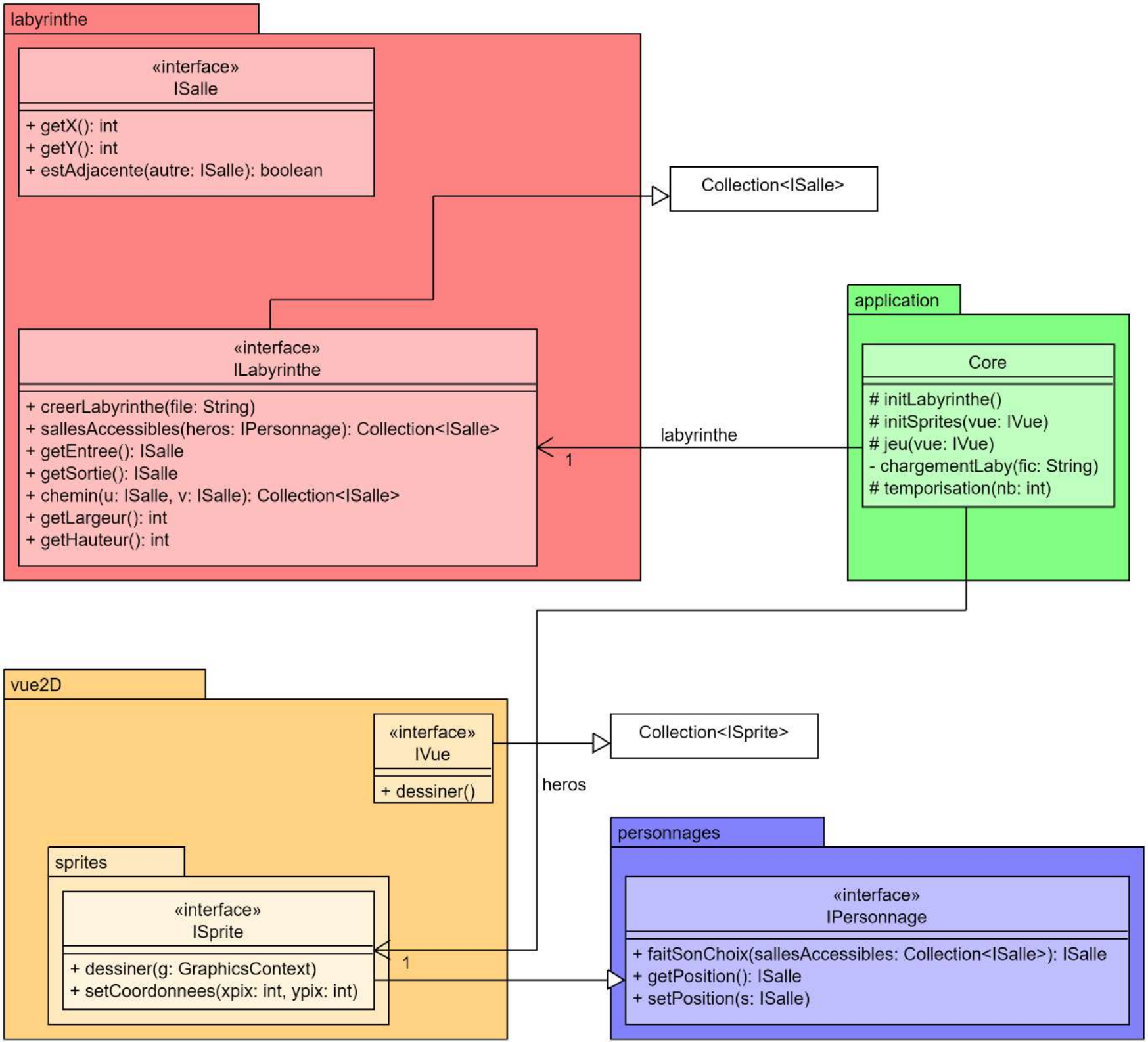
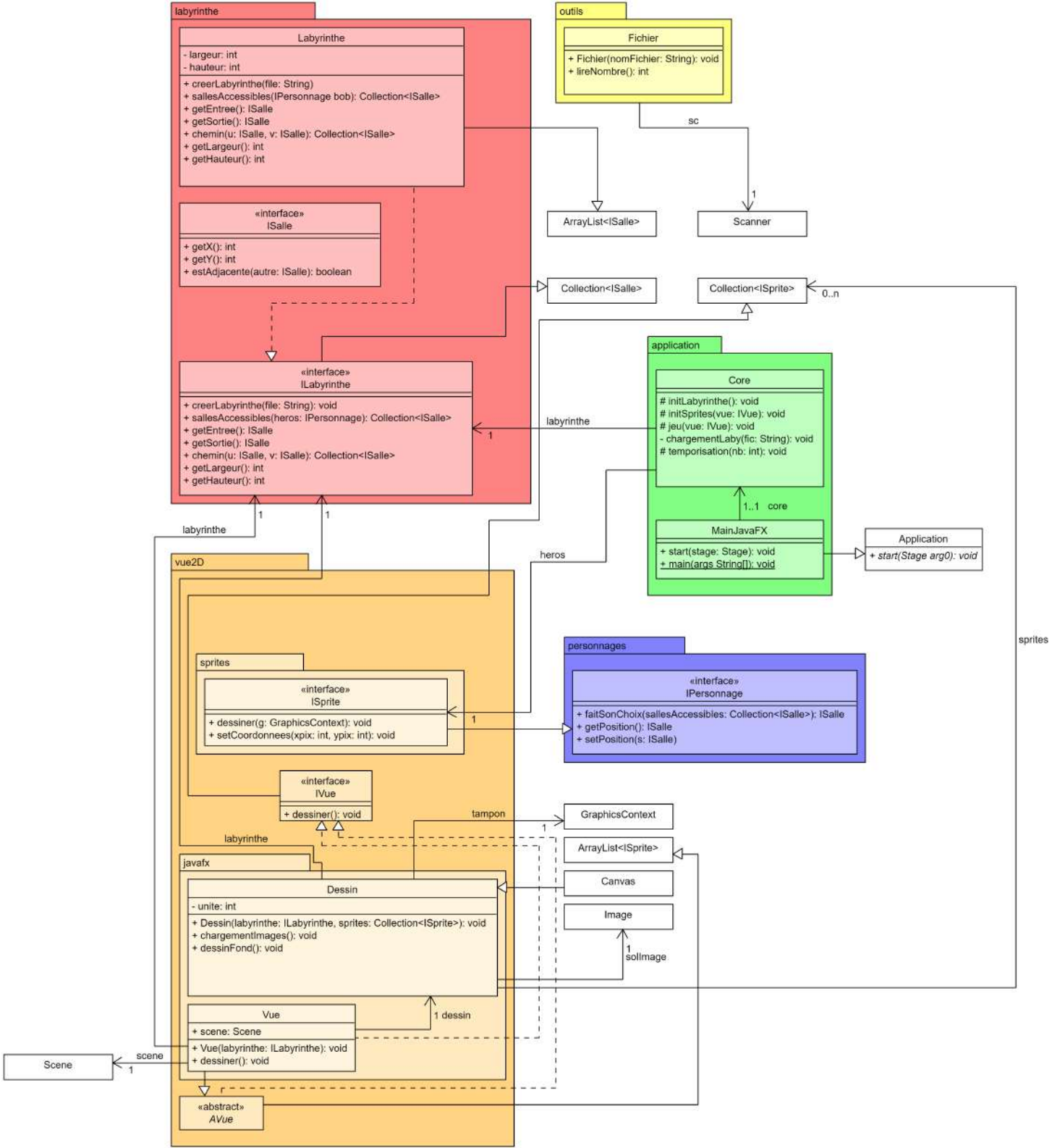
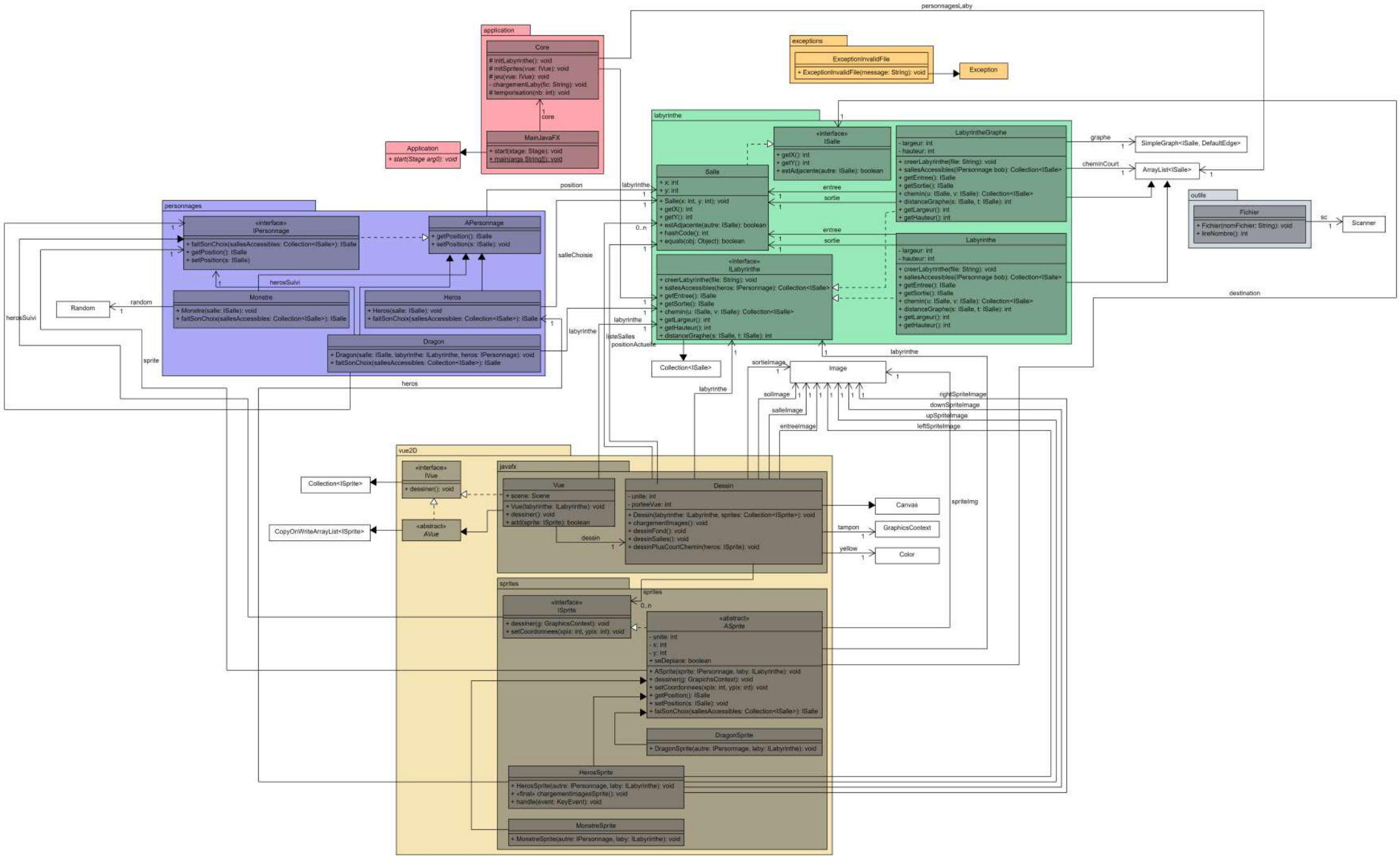

labyrinthe
«interface»
ISalle
+ getX(): int
+ getY(): int
+ estAdjacente(autre: ISalle): boolean
Collection<ISalle>
«interface»
ILabyrinthe
+ creerLabyrinthe(file: String)
+ sallesAccessibles(heros: IPersonnage): Collection<ISalle>
+ getEntree(): ISalle
+ getSortie(): ISalle
+ chemin(u: ISalle, v: ISalle): Collection<ISalle>
+ getLargeur(): int
+ getHauteur(): int
1
labyrinthe
application
Core
initLabyrinthe()
initSprites(vue: IVue)
jeu(vue: IVue)
- chargementLaby(fic: String)
temporisation(nb: int)
vue2D
«interface»
IVue
+ dessiner()
Collection<ISprite>
heros
sprites
«interface»
ISprite
+ dessiner(g: GraphicsContext)
+ setCoordonnees(xpix: int, ypix: int)
1
personnages
«interface»
IPersonnage
+ faitSonChoix(sallesAccessibles: Collection<ISalle>): ISalle
+ getPosition(): ISalle
+ setPosition(s: ISalle)


labyrinthe
Labyrinthe
- largeur: int
- hauteur: int
+ creerLabyrinthe(file: String)
+ sallesAccessibles(IPersonnage bob): Collection<ISalle>
+ getEntree(): ISalle
+ getSortie(): ISalle
+ chemin(u: ISalle, v: ISalle): Collection<ISalle>
+ getLargeur(): int
+ getHauteur(): int
«interface»
ISalle
+ getX(): int
+ getY(): int
+ estAdjacente(autre: ISalle): boolean
«interface»
ILabyrinthe
+ creerLabyrinthe(file: String): void
+ sallesAccessibles(heros: IPersonnage): Collection<ISalle>
+ getEntree(): ISalle
+ getSortie(): ISalle
+ chemin(u: ISalle, v: ISalle): Collection<ISalle>
+ getLargeur(): int
+ getHauteur(): int
outils
Fichier
+ Fichier(nomFichier: String): void
+ lireNombre(): int
sc
1
ArrayList<ISalle>
Scanner
Collection<ISalle>
Collection<ISprite>
0..n
application
Core
initLabyrinthe(): void
initSprites(vue: IVue): void
jeu(vue: IVue): void
- chargementLaby(fic: String): void
temporisation(nb: int): void
labyrinthe
1
1..1 core
MainJavaFX
+ start(stage: Stage): void
+ main(args String[]): void
Application
+ start(Stage arg0): void
labyrinthe
1
1
heros
sprites
vue2D
sprites
«interface»
ISprite
+ dessiner(g: GraphicsContext): void
+ setCoordonnees(xpix: int, ypix: int): void
1
personnages
«interface»
IPersonnage
+ faitSonChoix(sallesAccessibles: Collection<ISalle>): ISalle
+ getPosition(): ISalle
+ setPosition(s: ISalle)
«interface»
IVue
+ dessiner(): void
tampon
1
GraphicsContext
ArrayList<ISprite>
Canvas
Image
1
solImage
labyrinthe
javafx
Dessin
- unite: int
+ Dessin(labyrinthe: ILabyrinthe, sprites: Collection<ISprite>): void
+ chargementImages(): void
+ dessinFond(): void
1 dessin
Vue
+ scene: Scene
+ Vue(labyrinthe: ILabyrinthe): void
+ dessiner(): void
scene
1
Scene
r
«abstract»
AVue


application
Core
initLabyrinthe(): void
initSprites(vue: IVue): void
jeu(vue: IVue): void
- chargementLaby(fic: String): void
temporisation(nb: int): void
core
MainJavaFX
+ start(stage: Stage): void
+ main(args String[]): void
Application
+ start(Stage arg0): void
personnagesLaby
exceptions
ExceptionInvalidFile
+ ExceptionInvalidFile(message: String): void
Exception
labyrinthe
«interface»
ISalle
+ getX(): int
+ getY(): int
+ estAdjacente(autre: ISalle): boolean
Salle
+ x: int
+ y: int
+ Salle(x: int, y: int): void
+ getX(): int
+ getY(): int
+ estAdjacente(autre: ISalle): boolean
+ hashCode(): int
+ equals(obj: Object): boolean
LabyrintheGraphe
- largeur: int
- hauteur: int
+ creerLabyrinthe(file: String): void
+ sallesAccessibles(IPersonnage bob): Collection<ISalle>
+ getEntree(): ISalle
+ getSortie(): ISalle
+ chemin(u: ISalle, v: ISalle): Collection<ISalle>
+ distanceGraphe(s: ISalle, t: ISalle): int
+ getLargeur(): int
+ getHauteur(): int
Labyrinthe
- largeur: int
- hauteur: int
+ creerLabyrinthe(file: String): void
+ sallesAccessibles(IPersonnage bob): Collection<ISalle>
+ getEntree(): ISalle
+ getSortie(): ISalle
+ chemin(u: ISalle, v: ISalle): Collection<ISalle>
+ distanceGraphe(s: ISalle, t: ISalle): int
+ getLargeur(): int
+ getHauteur(): int
«interface»
ILabyrinthe
+ creerLabyrinthe(file: String): void
+ sallesAccessibles(heros: IPersonnage): Collection<ISalle>
+ getEntree(): ISalle
+ getSortie(): ISalle
+ chemin(u: ISalle, v: ISalle): Collection<ISalle>
+ getLargeur(): int
+ getHauteur(): int
+ distanceGraphe(s: ISalle, t: ISalle): int
graphe
SimpleGraph<ISalle, DefaultEdge>
cheminCourt
ArrayList<ISalle>
entree
sortie
position
labyrinthe
salleChoisie
listeSalles
positionActuelle
Collection<ISalle>
destination
outils
Fichier
+ Fichier(nomFichier: String): void
+ lireNombre(): int
sc
Scanner
personnages
«interface»
IPersonnage
+ faitSonChoix(sallesAccessibles: Collection<ISalle>): ISalle
+ getPosition(): ISalle
+ setPosition(s: ISalle)
APersonnage
+ getPosition(): ISalle
+ setPosition(s: ISalle): void
herosSuivi
Monstre
+ Monstre(salle: ISalle): void
+ faitSonChoix(sallesAccessibles: Collection<ISalle>): ISalle
Heros
+ Heros(salle: ISalle): void
+ faitSonChoix(sallesAccessibles: Collection<ISalle>): ISalle
Dragon
+ Dragon(salle: ISalle, labyrinthe: ILabyrinthe, heros: IPersonnage): void
+ faitSonChoix(sallesAccessibles: Collection<ISalle>): ISalle
Random
random
sprite
heros
Image
sortieImage
sollImage
salleImage
entreeImage
rightSpriteImage
downSpriteImage
upSpriteImage
leftSpriteImage
vue2D
«interface»
IVue
+ dessiner(): void
«abstract»
AVue
Collection<ISprite>
CopyOnWriteArrayList<ISprite>
javafx
Vue
+ scene: Scene
+ Vue(labyrinthe: ILabyrinthe): void
+ dessiner(): void
+ add(sprite: ISprite): boolean
dessin
Dessin
- unite: int
- porteeVue: int
+ Dessin(labyrinthe: ILabyrinthe, sprites: Collection<ISprite>): void
+ chargementImages(): void
+ dessinFond(): void
+ dessinSalles(): void
+ dessinPluaCourtChemin(heros: ISprite): void
Canvas
tampon
GraphicsContext
yellow
Color
spriteImg
sprites
«interface»
ISprite
+ dessiner(g: GraphicsContext): void
+ setCoordonnees(xpix: int, ypix: int): void
«abstract»
ASprite
- unite: int
- x: int
- y: int
+ seDeplace: boolean
+ ASprite(sprite: IPersonnage, laby: ILabyrinthe): void
+ dessiner(g: GrapichsContext): void
+ setCoordonnees(xpix: int, ypix: int): void
+ getPosition(): ISalle
+ setPosition(s: ISalle): void
+ faitSonChoix(sallesAccessibles: Collection<ISalle>): ISalle
DragonSprite
+ DragonSprite(autre: IPersonnage, laby: ILabyrinthe): void
HerosSprite
+ HerosSprite(autre: IPersonnage, laby: ILabyrinthe): void
+ «final» chargementImagesSprite(): void
+ handle(event: KeyEvent): void
MonstreSprite
+ MonstreSprite(autre: IPersonnage, laby: ILabyrinthe): void