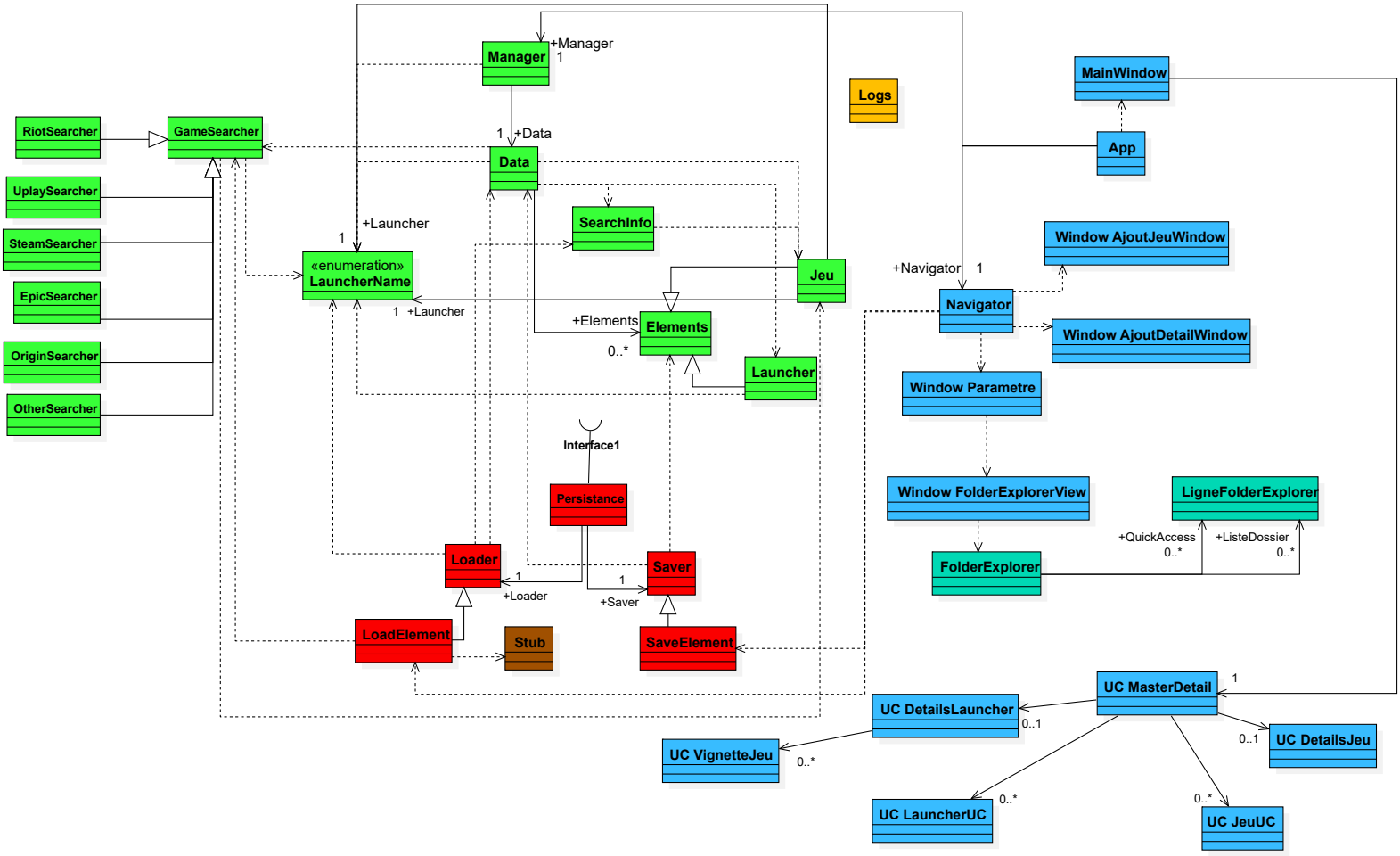
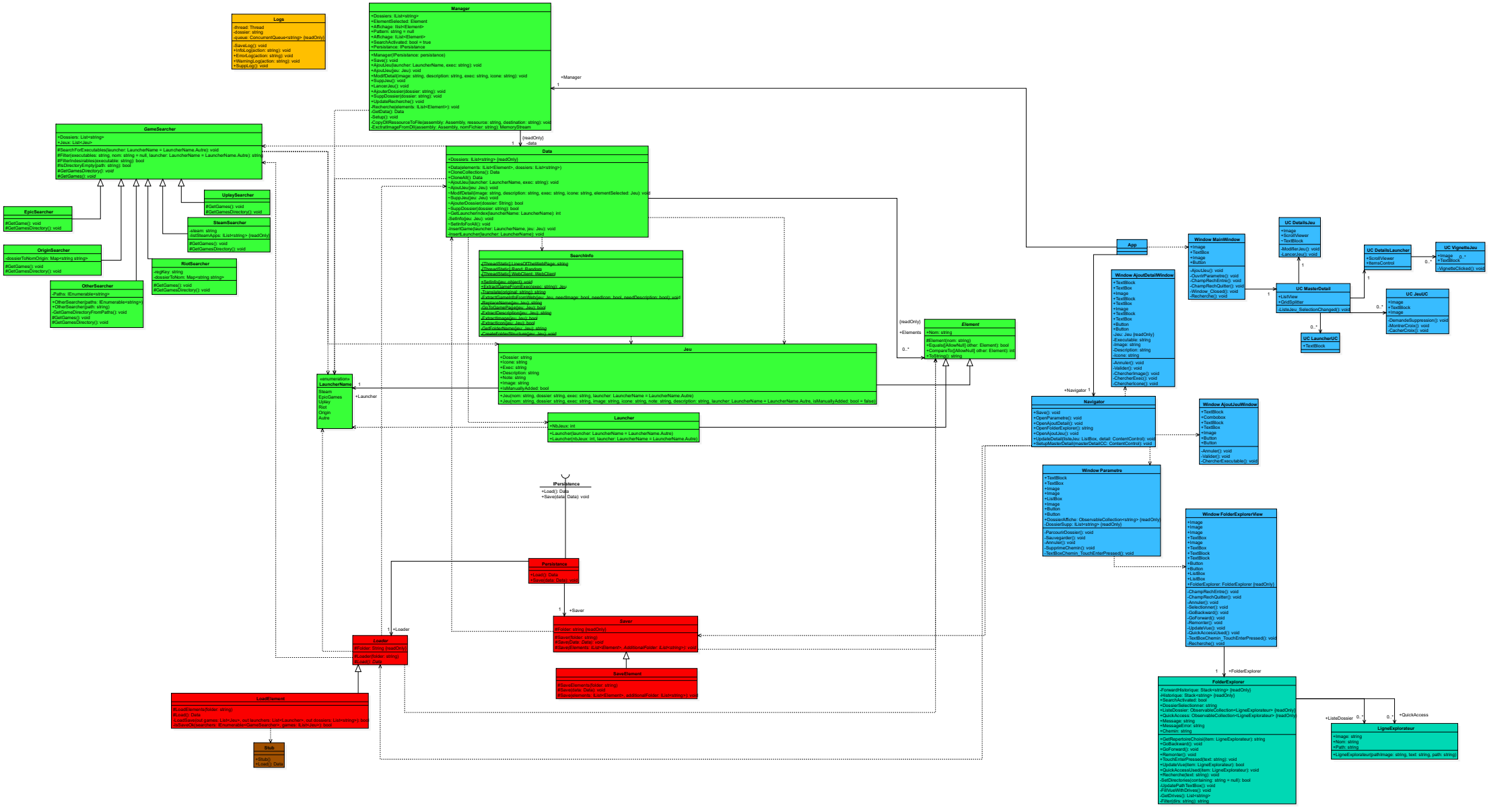
Documentation C#

-Diagramme de classe

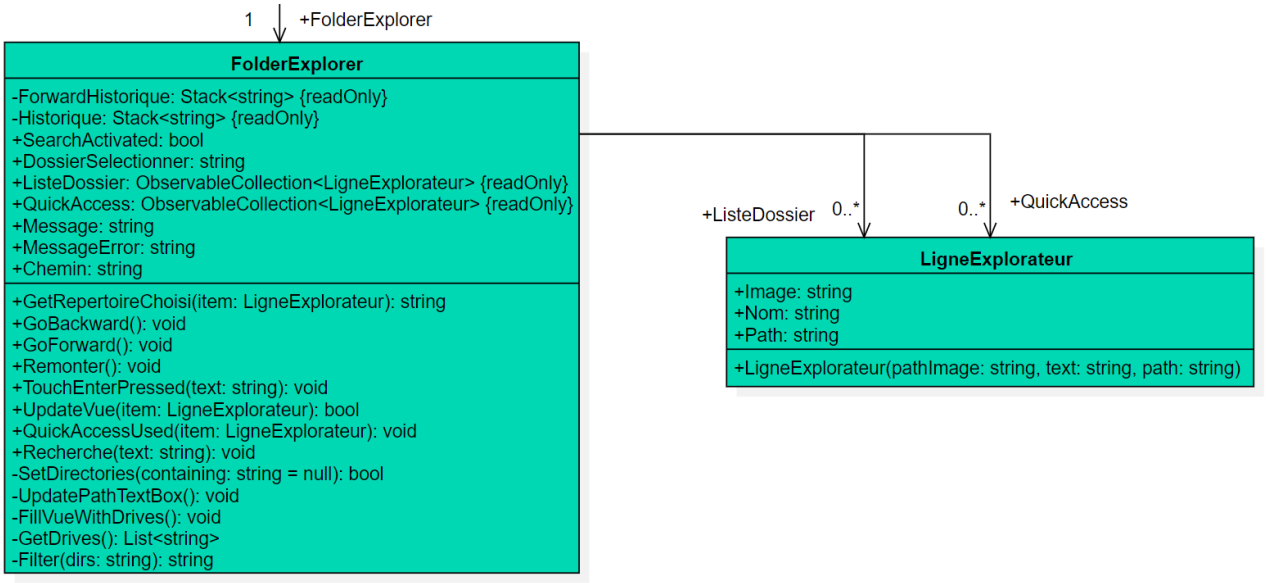
Pour notre application nous avons décidé d'utiliser le patron de conception structurel de la « façade ». L'utilisation de ce patron nous permet d'avoir une seule et unique interface avec la vue. Donc pour toutes les actions de la vue autre que la navigation, une fonction de notre manager est lancée pour répondre à l'action demander par l'utilisateur. De cette manière aucune modification des données ou autre changement ne peux se faire sans « l'accord » du manager.
Ce manager est situé dans le paquet Modele dans la classe Manager.
(Pour une raison inconnue les flèches en pointillés dans les SVG sont affiché sous forme de flèche pleine, veuillez regardez directement le fichier).

Diagramme de classe simplifié :




Logs
-thread: Thread
-dossier: string
-queue: ConcurrentQueue<string> {readOnly}
-SaveLog(): void
+InfoLog(action: string): void
+ErrorLog(action: string): void
+WarningLog(action: string): void
+SuppLog(): void
Manager
+Dossiers: IList<string>
+ElementSelected: Element
+Affichage: IList<Element>
+Pattern: string = null
+Affichage: IList<Element>
+SearchActivated: bool = true
+Persistance: IPersistance
+Manager(Persistance: persistance)
+Save(): void
+AjoutJeu(launcher: LauncherName, exec: string): void
+AjoutJeu(jeu: Jeu): void
-ModifDetail(image: string, description: string, exec: string, icone: string): void
+SuppJeu(): void
+LancerJeu(): void
+AjouteDossier(dossier: string): void
+SuppDossier(dossier: string): void
+UpdateRecherche(): void
-Recherche(elements: IList<Element>): void
-GetData(): Data
-Setup(): void
-CopyDllRessourceToFile(assembly: Assembly, ressource: string, destination: string): void
-ExtraitImageFromDll(assembly: Assembly, nomFichier: string): MemoryStream
+Manager
1
{readOnly}
-data
GameSearcher
+Dossiers: List<string>
+Jeux: List<Jeu>
#SearchForExecutables(launcher: LauncherName = LauncherName.Autre): void
#Filter(executables: string, nom: string = null, launcher: LauncherName = LauncherName.Autre): string
#FilterInvisibles(executable: string): bool
#IsDirectoryEmpty(path: string): bool
#GetGamesDirectory(): void
#GetGames(): void
EpicSearcher
#GetGame(): void
#GetGamesDirectory(): void
OriginSearcher
-dossierToNomOrigin: Map<string string>
#GetGames(): void
#GetGamesDirectory(): void
OtherSearcher
-Paths: IEnumerable<string>
+OtherSearcher(paths: IEnumerable<string>)
+OtherSearcher(path: string)
-GetGameDirectoryFromPaths(): void
#GetGames(): void
#GetGamesDirectory(): void
UplaySearcher
#GetGames(): void
#GetGamesDirectory(): void
SteamSearcher
-steam: string
-listSteamApps: IList<string> {readOnly}
#GetGames(): void
#GetGamesDirectory(): void
RiotSearcher
-regKey: string
-dossierToNom: Map<string string>
#GetGames(): void
#GetGamesDirectory(): void
Data
+Dossiers: IList<string> {readOnly}
+Data(elements: IList<Element>, dossiers: List<string>)
+CloneCollections(): Data
+CloneAll(): Data
~AjoutJeu(launcher: LauncherName, exec: string): void
~AjoutJeu(jeu: Jeu): void
~ModifDetail(image: string, description: string, exec: string, icone: string, elementSelected: Jeu): void
~SuppJeu(jeu: Jeu): void
~AjouteDossier(dossier: String): bool
~SuppDossier(dossier: string): bool
~GetLauncherIndex(launcherName: LauncherName): int
-SetInfo(jeu: Jeu): void
~SetInfoForAll(): void
-InsertGame(launcher: LauncherName, jeu: Jeu): void
-InsertLauncher(launcher: LauncherName): void
SearchInfo
-[ThreadStatic] LinesOfTheWebPage: string
-[ThreadStatic] Rand: Random
-[ThreadStatic] WebClient: WebClient
+SetInfo(jeu: object): void
+ExtractGameFromExe(exec: string): Jeu
-TranslateIcon(original_string): string
-ExtractGameInfoFromWeb(jeu: Jeu, needImage: bool, needIcon: bool, needDescription: bool): void
-ReplaceName(jeu: Jeu): string
-GoToGamePage(jeu: Jeu): bool
-ExtractDescription(jeu: Jeu): string
-ExtractImage(jeu: Jeu): bool
-ExtractIcon(jeu: Jeu): bool
-GetFolderName(jeu: Jeu): string
-CreateFolderStructure(jeu: Jeu): void
Jeu
+Dossier: string
+Icone: string
+Exec: string
+Description: string
+Note: string
+Image: string
+IsManuallyAdded: bool
+Jeu(nom: string, dossier: string, exec: string, launcher: LauncherName = LauncherName.Autre)
+Jeu(nom: string, dossier: string, exec: string, image: string, icone: string, note: string, description: string, launcher: LauncherName = LauncherName.Autre, isManuallyAdded: bool = false)
{readOnly}
+Elements
0..*
Element
+Nom: string
#Element(nom: string)
+Equals([AllowNull] other: Element): bool
+CompareTo([AllowNull] other: Element): int
+ToString(): string
<enumeration>
LauncherName
Steam
EpicGames
Uplay
Riot
Origin
Autre
1
+Launcher
Launcher
+NbJeux: int
+Launcher(launcher: LauncherName = LauncherName.Autre)
+Launcher(nbJeux: int, launcher: LauncherName = LauncherName.Autre)
IPersistence
+Load(): Data
+Save(data: Data): void
Persistance
+Load(): Data
+Save(data: Data): void
1
+Saver
Saver
#Folder: string {readOnly}
#Saver(folder: string)
#Save(data: Data): void
#Save(elements: IList<Element>, additionalFolder: IList<string>): void
SaveElement
#SaveElements(folder: string)
#Save(data: Data): void
#Save(elements: IList<Element>, additionalFolder: IList<string>): void
1
+Loader
Loader
#Folder: String {readOnly}
#Loader(folder: string)
#Load(): Data
LoadElement
#LoadElements(folder: string)
#Load(): Data
LoadSave(out games: List<Jeu>, out launchers: List<Launcher>, out dossiers: List<string>): bool
SaveObjects(launchers: IEnumerable<GameSearcher>, games: IList<Jeu>): bool
Stub
+Stub()
+Load(): Data
App
Window MainWindow
+Image
+TextBox
+Image
+Button
-AjoutJeu(): void
-OuvrirParametre(): void
-ChampRechEntre(): void
-ChampRechQuitter(): void
-Window_Closed(): void
-Recherche(): void
1
Window AjoutDetailWindow
+TextBlock
+TextBox
+Image
+TextBlock
+TextBox
+Image
+TextBlock
+TextBox
+Button
+Button
-Jeu: Jeu {readOnly}
-Executable: string
-Image: string
-Description: string
-Icone: string
-Annuler(): void
-Valider(): void
-ChercherImage(): void
-ChercherExec(): void
-ChercherIcone(): void
+Navigator 1
Navigator
+Save(): void
+OpenParametre(): void
+OpenAjoutDetail(): void
+OpenFolderExplorer(): string
+OpenAjoutJeu(): void
+UpdateDetail(listeJeu: ListBox, detail: ContentControl): void
+SetupMasterDetail(masterDetailCC: ContentControl): void
Window AjoutJeuWindow
+TextBlock
+Combobox
+TextBlock
+TextBox
+Image
+Button
+Button
-Annuler(): void
-Valider(): void
-ChercherExecutable(): void
Window Parametre
+TextBlock
+TextBox
+Image
+Image
+ListBox
+Image
+Button
+Button
+DossierAffiche: ObservableCollection<string> {readOnly}
-DossierSupp: IList<string> {readOnly}
-ParcourirDossier(): void
-Sauvegarder(): void
-Annuler(): void
-SuppImmeChemin(): void
-TextBoxChemin_TouchEnterPressed(): void
Window FolderExplorerView
+Image
+Image
+Image
+TextBox
+Image
+TextBox
+TextBlock
+TextBlock
+Button
+Button
+ListBox
+ListBox
+FolderExplorer: FolderExplorer {readOnly}
-ChampRechEntre(): void
-ChampRechQuitter(): void
-Annuler(): void
-Selectionner(): void
-GoBackward(): void
-GoForward(): void
-Remonter(): void
-UpdateVue(): void
-QuickAccessUsed(): void
-TextBoxChemin_TouchEnterPressed(): void
-Recherche(): void
1
+FolderExplorer
FolderExplorer
-ForwardHistorique: Stack<string> {readOnly}
-Historique: Stack<string> {readOnly}
+SearchActivated: bool
+DossierSelectionne: string
+ListeDossier: ObservableCollection<LigneExplorateur> {readOnly}
+QuickAccess: ObservableCollection<LigneExplorateur> {readOnly}
+Message: string
+MessageError: string
+Chemin: string
+GetRepertoireChoisi(item: LigneExplorateur): string
+GoBackward(): void
+GoForward(): void
+Remonter(): void
+TouchEnterPressed(text: string): void
+UpdateVue(item: LigneExplorateur): bool
+QuickAccessUsed(item: LigneExplorateur): void
+Recherche(text: string): void
-SetDirectories(containing: string = null): bool
-UpdatePathTextBox(): void
-FillVueWithDrives(): void
-GetDrives(): List<string>
-FilterJeu(jeu: string): string
+ListeDossier
0..*
+QuickAccess
LigneExplorateur
+Image: string
+Nom: string
+Path: string
+LigneExplorateur(pathImage: string, text: string, path: string)
UC DetailsJeu
+Image
+ScrollViewer
-TextBlock
-ModifierJeu(): void
-LancerJeu(): void
1
UC DetailsLauncher
+ScrollViewer
+ItemsControl
1
UC VignetteJeu
+Image
+TextBlock
0..*
-VignetteClicked(): void
UC MasterDetail
+ListView
+GridSplitter
-ListeJeu_SelectionChanged(): void
1
0..*
0..*
UC JeuUC
+Image
+TextBlock
+Image
-DemandeSuppression(): void
-MontrerCroix(): void
-CacherCroix(): void
UC LauncherUC
+TextBlock

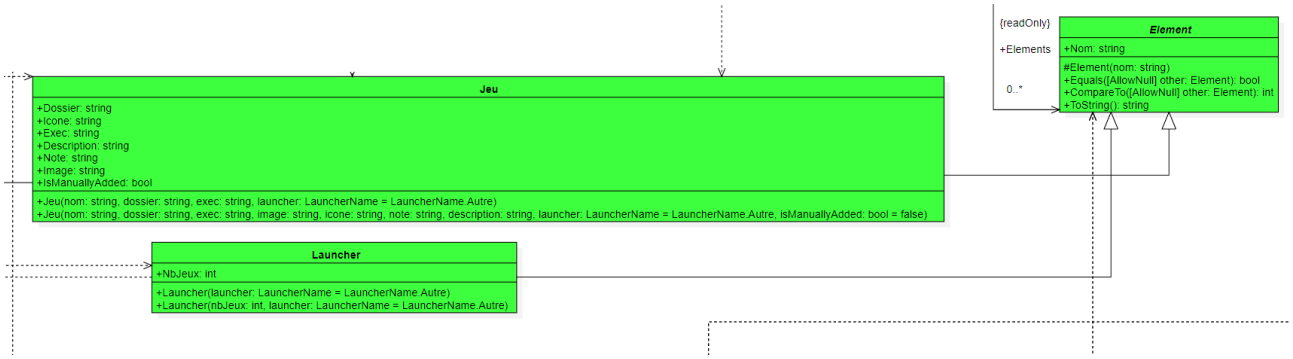
Paquet FolderExplorer :



Voir description dans « Documentation Projet tut »

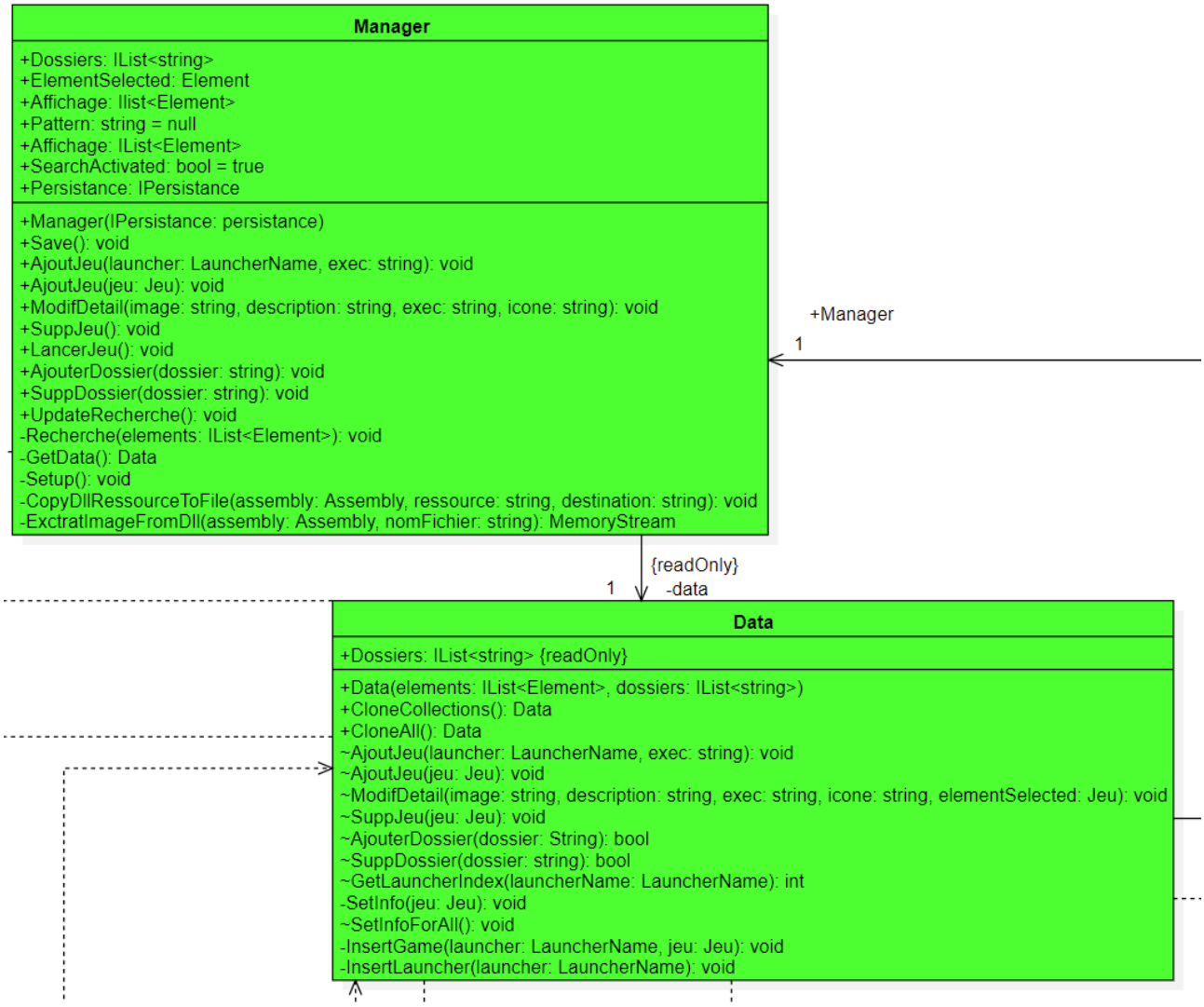
Paquet Modele :

-Partie Element :



Jeu et Launcher héritent d'Element. Element possède un attribut Nom et permet de manipuler les Jeu et Launcher au sein d'une seule liste via polymorphisme.
Jeu regroupe les informations principales d'un jeu et Launcher possède le nombre de jeux qu'il contient.
L'attribut IsManuallyAdded permet de savoir si le jeu a été ajouter par l'utilisateur ou s'il est trouvé par l'application.

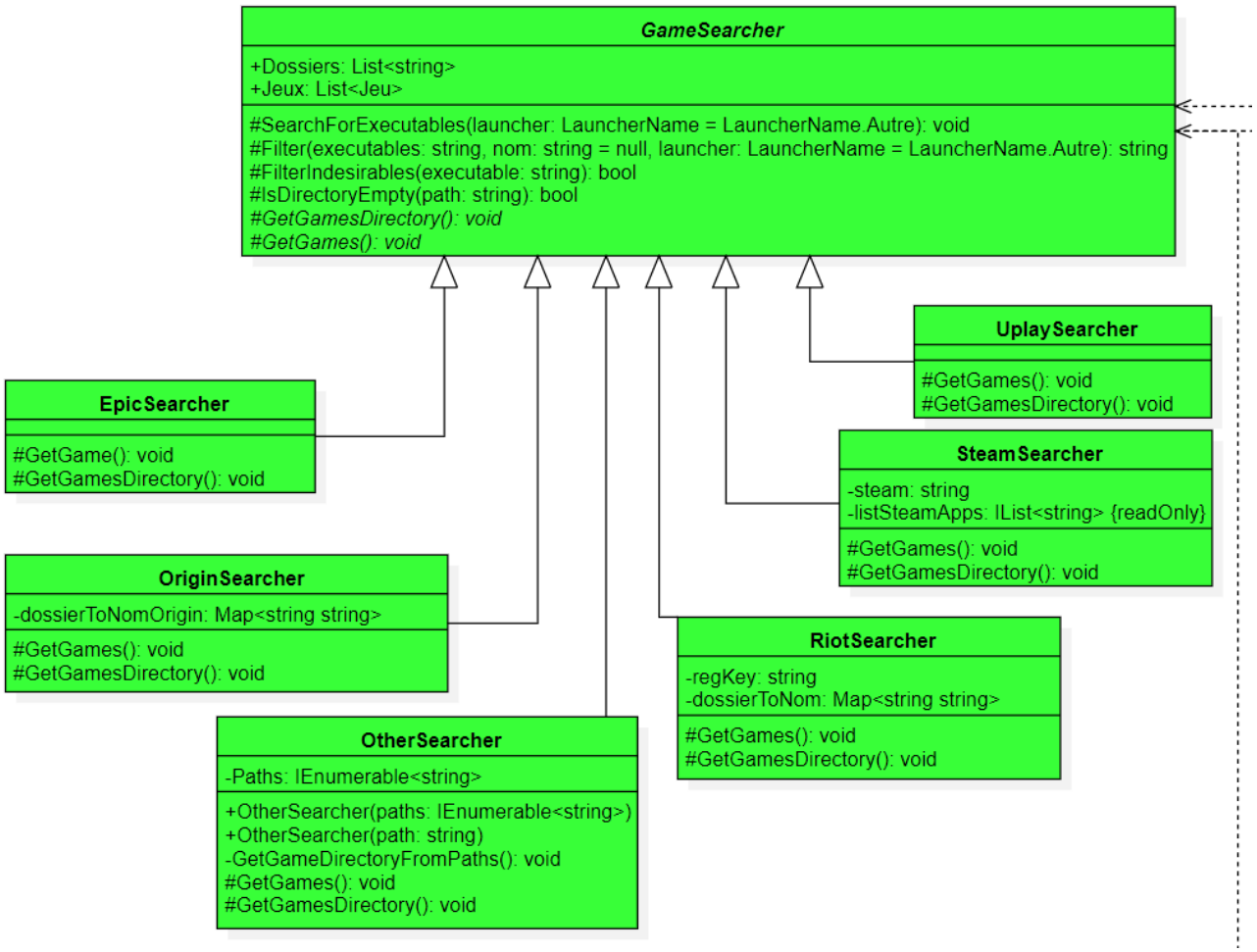
-Partie Manager/Data :



Le Manager relie la vue au Modele. Il dépend de Data qui permet de lancer les fonctions affectant les données. Data modifie les données et Manager ne sert que d'interface entre la vue et Data.

SetInfo() appel la fonction SearchInfo.SetInfo() dans un nouveau thread.
InsertGame() insert les jeux par ordre alphabétique dans leurs launchers respectifs.
InsertLauncher() insert les launchers par ordre alphabétique en laissant le launcher « Autre » toujours en dernier.

-Partie GameSearcher :



Dossiers et Jeux sont des propriétés calculées à partir de GetGamesDirectory (Dossiers) et GetGames (Jeux) une fois calculé le résultat est stockées dans un variable privée du même nom.

Chaque classe fille de GameSearcher spécialise GetGames et GetGamesDirectory pour un launcher spécifique.

OtherSearcher se comporte comme les autres mais se base sur un chemin (ou une liste de chemin) pour trouver des jeux.

LauncherName est une enum qui contient les noms des principaux launcher.

-Partie SearchInfo

SearchInfo permet d'aller chercher les informations d'un jeu sur internet tel que l'icône, la description et l'image.

La classe possède des attribut ThreadStatic pour permettre de multithreadé la recherche d'info (qui peux prendre jusqu'à 2 sec par jeu voir plus).

SetInfo reçoit un objet car il doit être compatible avec le deleguate ParameterizedThreadStart pour pouvoir lui donner un jeu en paramètre lors du démarrage du thread. SetInfo va regarder quelle sont les infos manquantes.

ExtractGameFromExec sert à récupérer des infos (dossier, nom) à partir de l'exécutable.

Tanslate s'appuie sur un serveur de traduction pour traduire de l'anglais vers français.

ExtractGameInfoFromWeb lance les extractions nécessaires à partir des infos recu par SetInfo.

ReplaceName sert à obtenir le nom du jeu modifié convenant au site qu'on utilise pour aller chercher les infos (IGDB).

ExctractDescription/Image/Icon servent à extraire les infos.

GetFolderName renvoie un nom de dossier correspondant au spécification de Windows à partir du nom du jeu.

CreateFolderStructure s'assure de créer le chemin dans lequel les infos seront placé.

Paquet Vue :

-Partie MainWindow

App possède un Navigator de cette manière le Navigator est accessible à l'entièreté de la Vue.
App possède un Manager de cette manière le Manager est accessible à l'entièreté de la Vue.

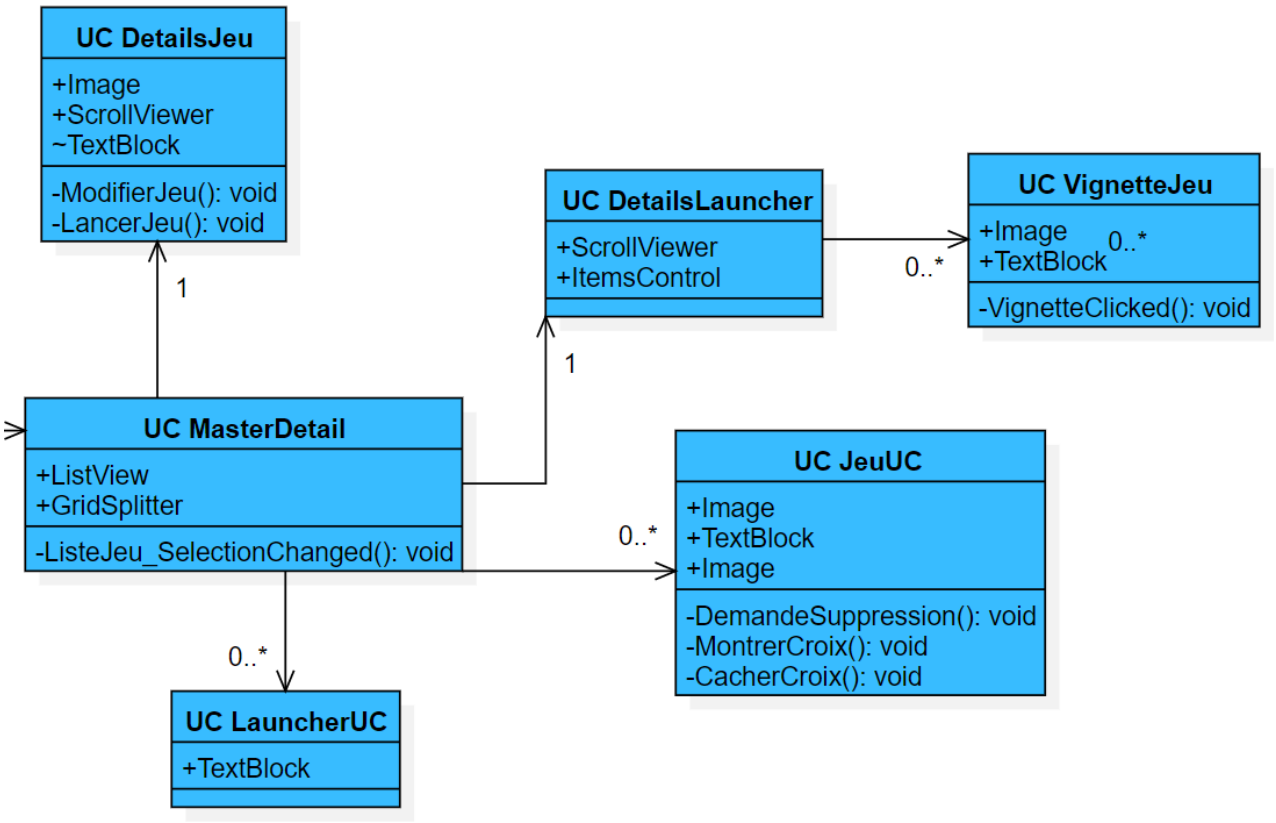
Navigator est la classe qui gère la navigation de la vue, c'est elle qui modifie les ContentControl et ouvre les fenêtres.
Navigator.Setup() permet de chargé l'application à partir de la sauvegarde (Stub ou réel) et Save() lance la sauvegarde.
UpdateDetail permet de changer le contenu du ContentControl passer en paramètre afin d'avoir l'affichage souhaité par rapport au type de l'élément sélectionné dans ListBox.
SetupMasterDetail met un MasterDetail dans le ContentControl passer en paramètre.

La MainWindow est la fenêtre d'accueil de l'application, c'est elle qui possède le Master Detail.
Window_Closed lance Navigator.Save().

AjoutDetailWindow permet de modifier les détail d'un jeu.
AjoutJeuWIndow permet d'ajouter un jeu.

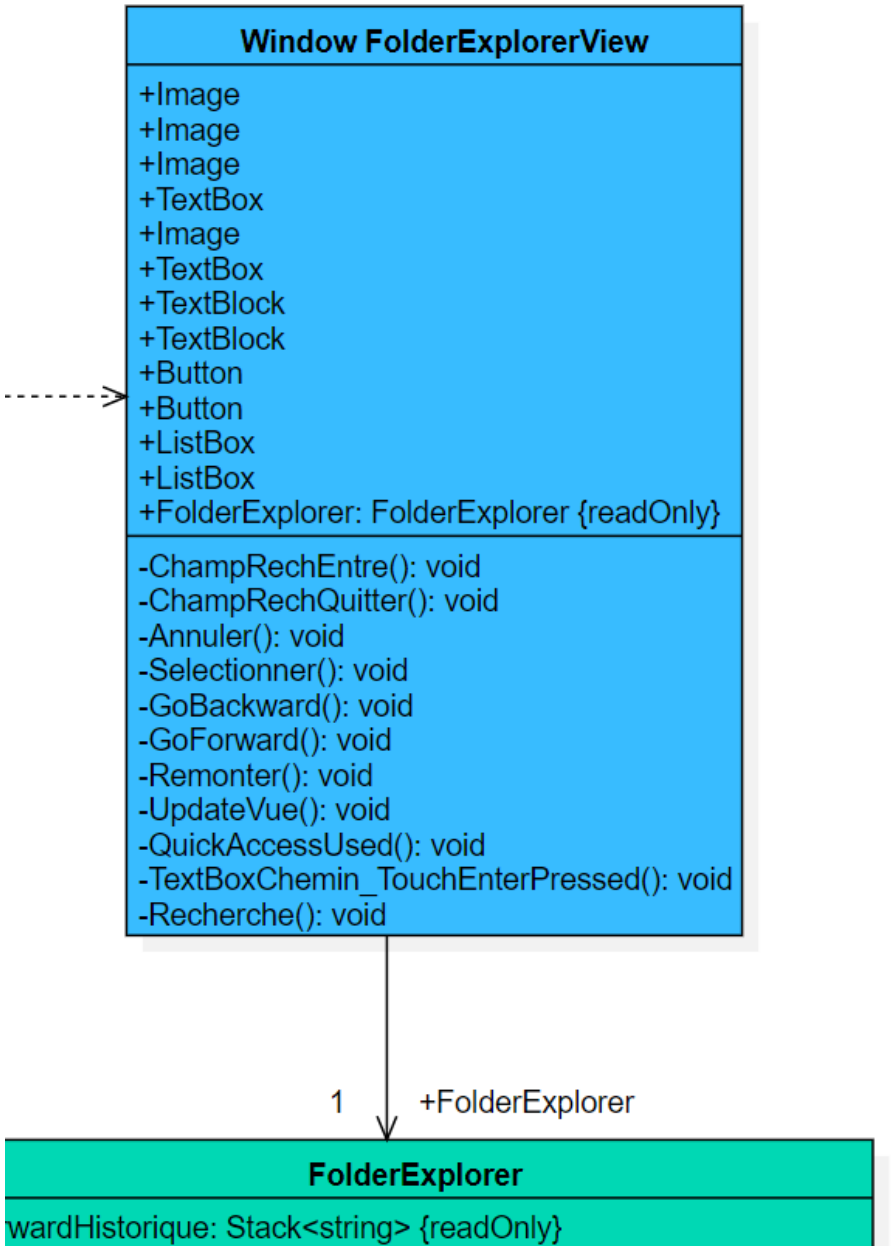
Parametre permet d'ajouter un dossier contenant des dossiers de jeux.

-Partie MasterDetail



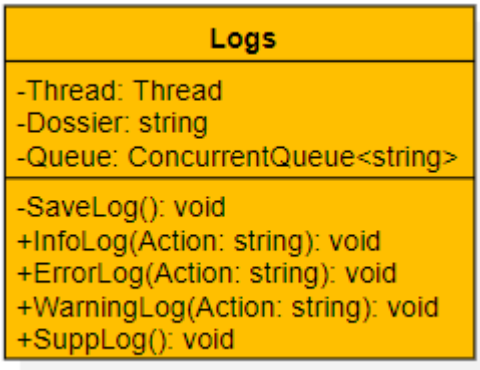
MasterDetail permet de voir les différents launchers et jeux ainsi que leurs informations.
ListeJeu_SelectionChanged() appelle Navigator.
UpdateDetail() DetailJeu et DetailLuncher corresponde à la partie détails de leurs types respectifs.
JeuUC et LuncherUC corresponde à la partie Master de leurs types respectifs.
VignetteJeu correspond à l'affichage d'un jeu dans le détail de launcher. VignetteClicked permet d'aller sur le jeu cliquer.

-Partie FolderExplorerView



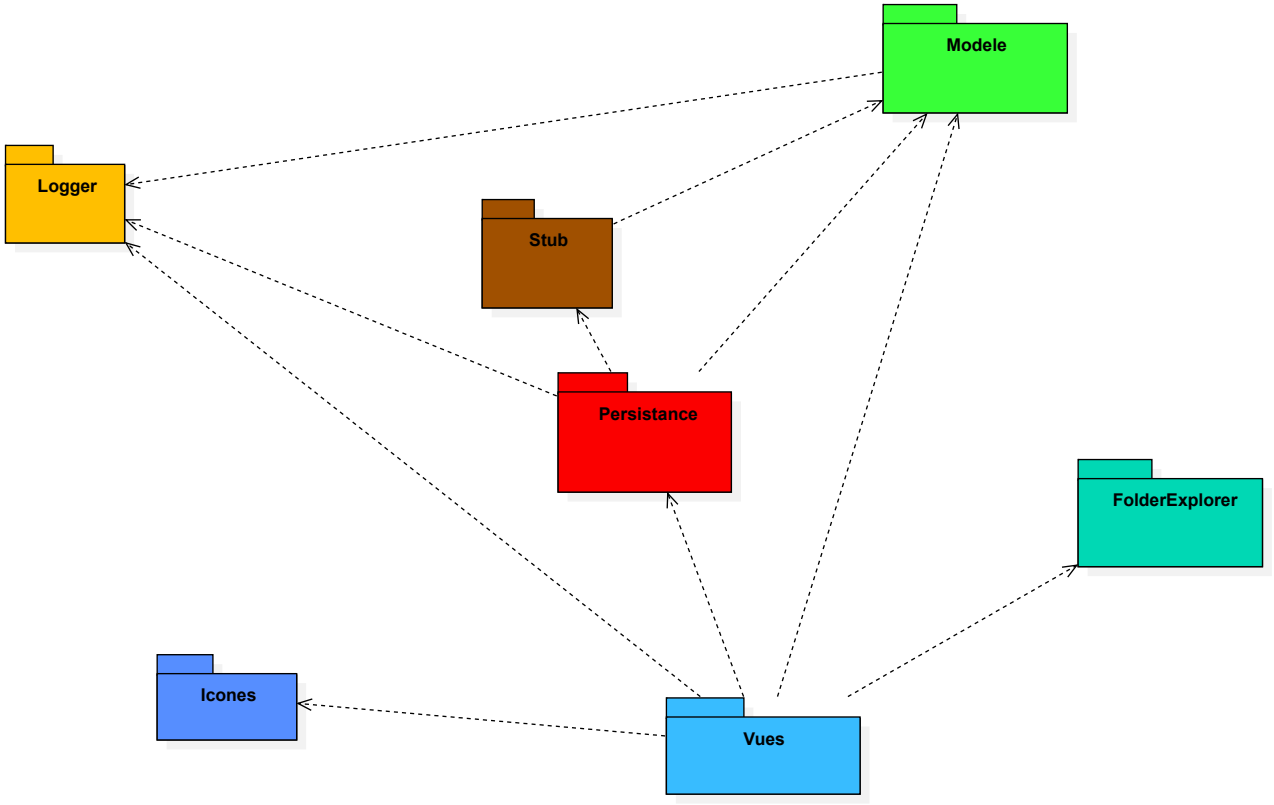
Voir description dans « Documentation Projet tut »

Paquet Logger :



Voir description dans « Documentation Projet tut »

-Diagramme de paquet



Les paquets FolderExplorer et Icones n'ont pas de dépendances, Vue est dépendant de tous les paquets sauf Stub. Persistance dépend de Modele, Stub et Logger et Modele dépend uniquement de Logger.
Persistance s'occupe de sauvegarder et charger les données de l'application.
Modele contient une classe Manager servant d'interface entre la logique de l'application et la Vue.

Le Logger aura pour but de permettre un débogage plus simple une fois l'application déployer chez le client. Il est utilisé pas plusieurs paquets.

Le paquet Icone permettra de mettre à jour les icones sans avoir à mettre à jour le paquet qui les contiens.
FolderExplorer implémente le comportement d'un explorateur de dossier.

-Diagramme de séquence du loader