Javsmash

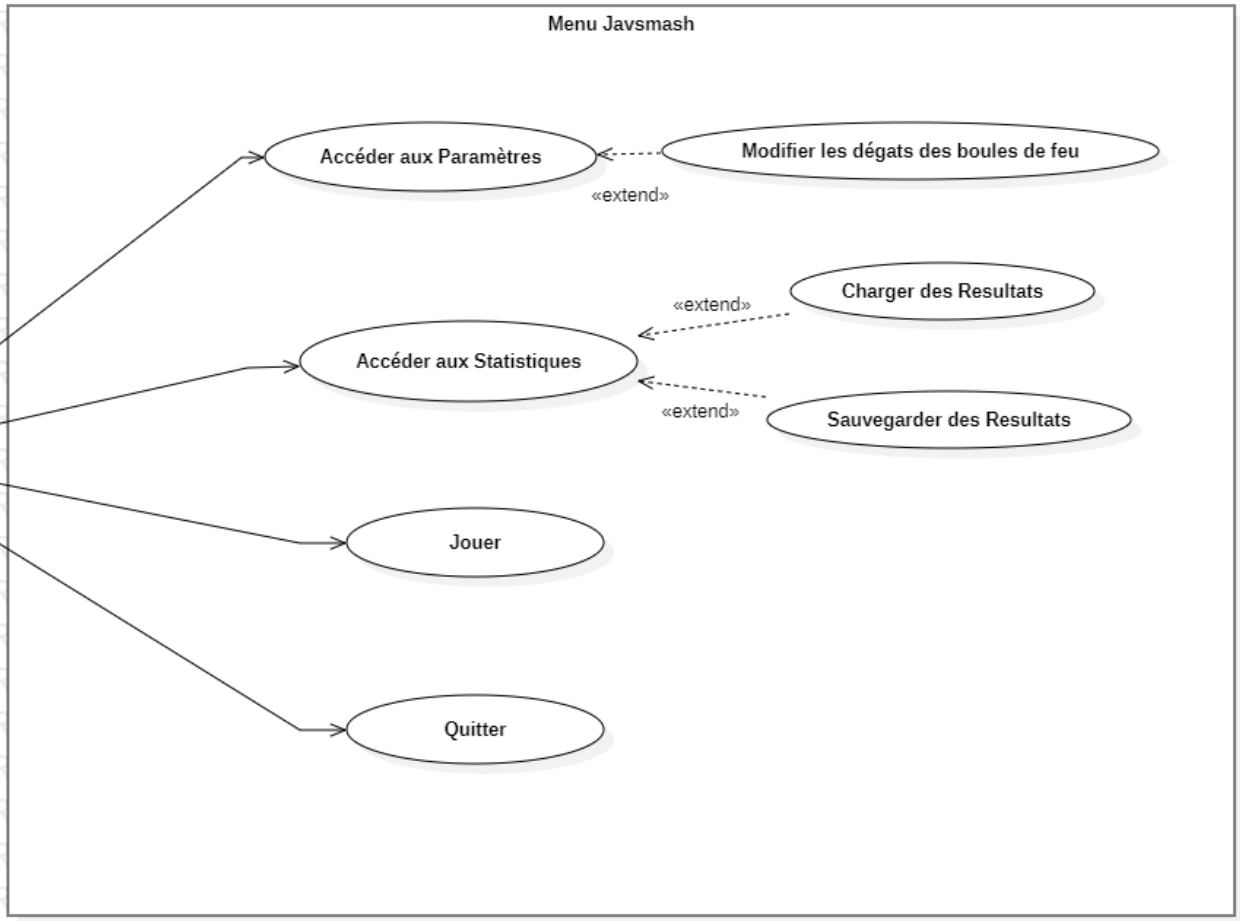
Par Clément GUYON & Maxime DACISAC

Table des matières

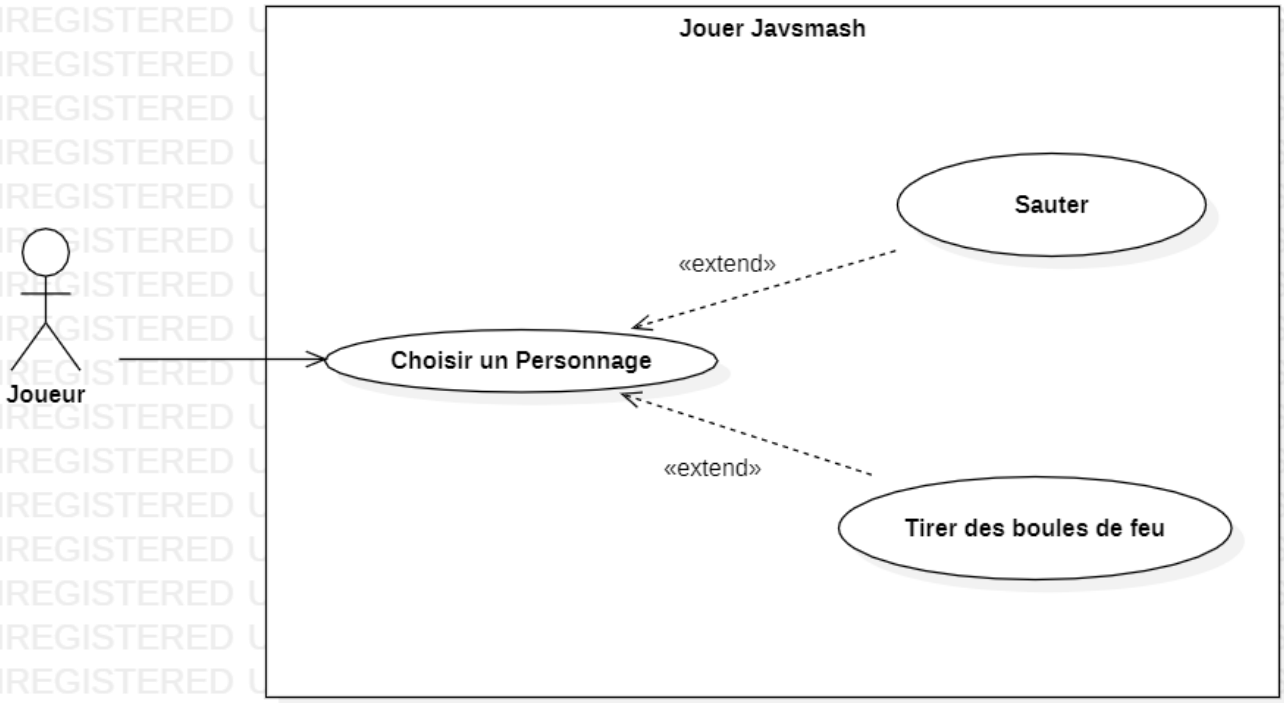
1) Présentation :

Javsmash est un jeu de type Versus Fighting dans lequel deux personnages se battent. Vous pouvez choisir un personnage chacun parmi une liste de personnages préinstallés. Vous pouvez également accéder à vos statistiques de jeu et à des paramètres de configuration via le menu d'accueil.

Voici le diagramme de cas d'utilisation pour le menu principal représentant les fonctionnalités accessibles pour l'instant :



Ce diagramme présente les différentes fonctionnalités disponibles une fois que le joueur à cliquer le bouton « Jouer ».

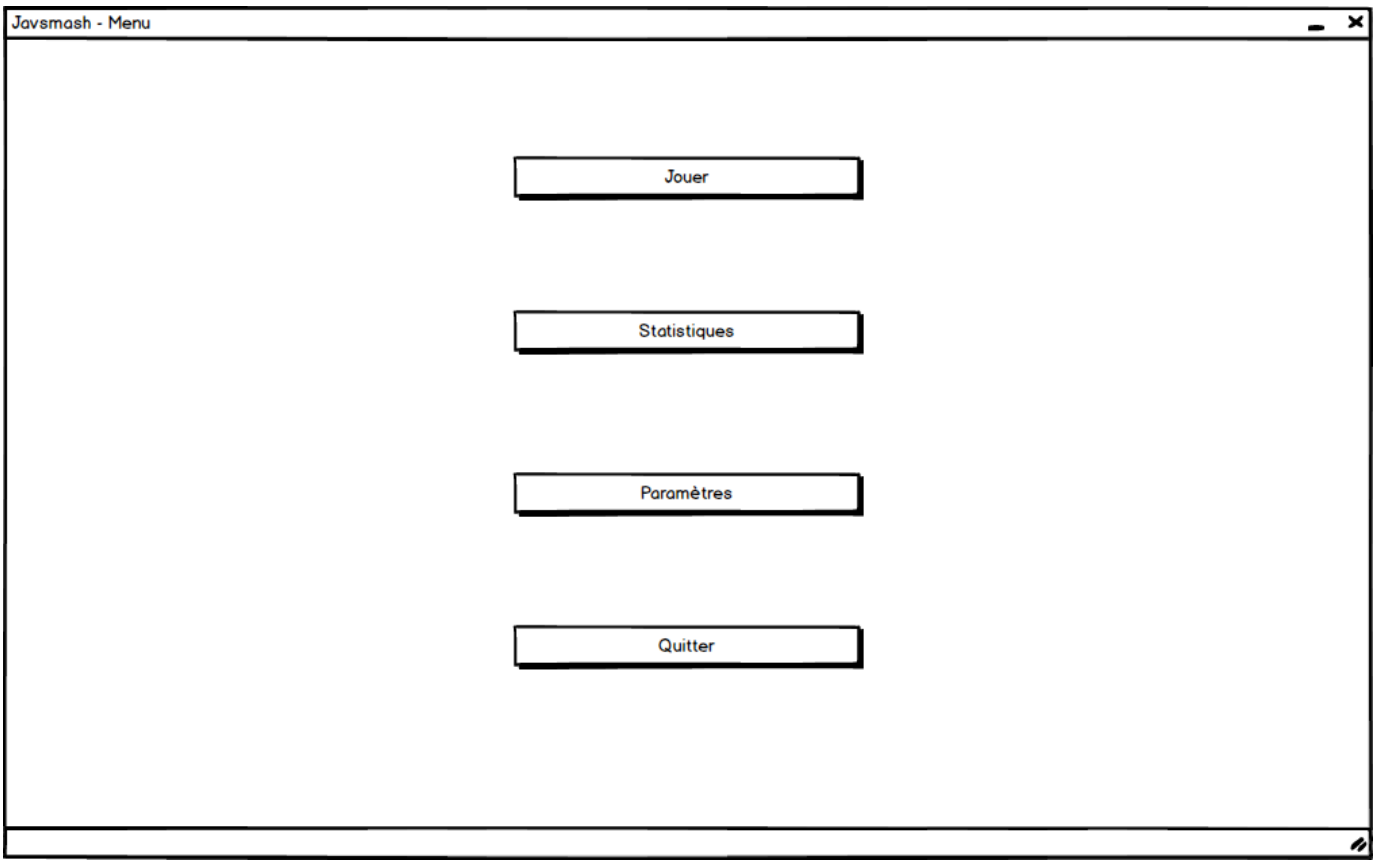


Les différentes fonctionnalités sont réparties sur les fenêtres qui se succèdent lors de l'utilisation de l'application.

2) Aperçu de l'interface

2.1) Page d'accueil

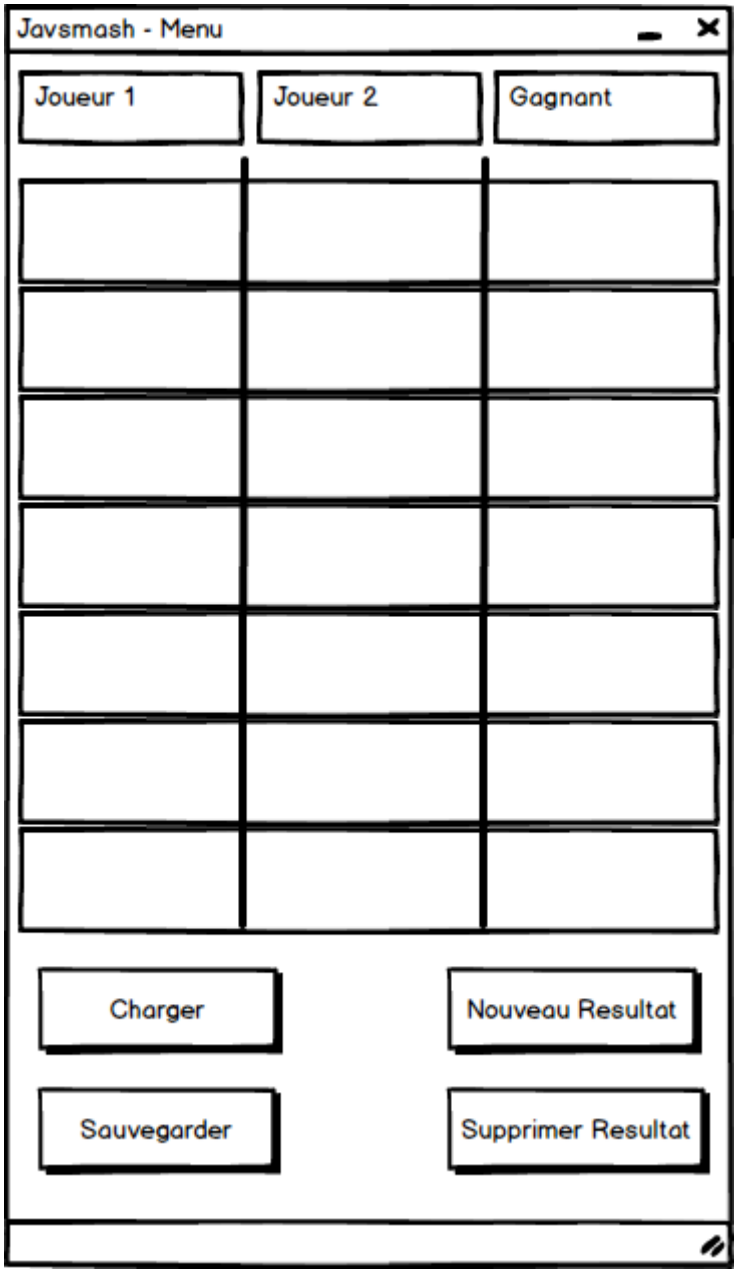
Cette fenêtre est la page d'accueil de l'application lorsque cette dernière est démarrée. Nous pouvons y voir différents boutons rassemblés dans un *StackPane* comme présenté ci-bas :



2.2) Page statistiques

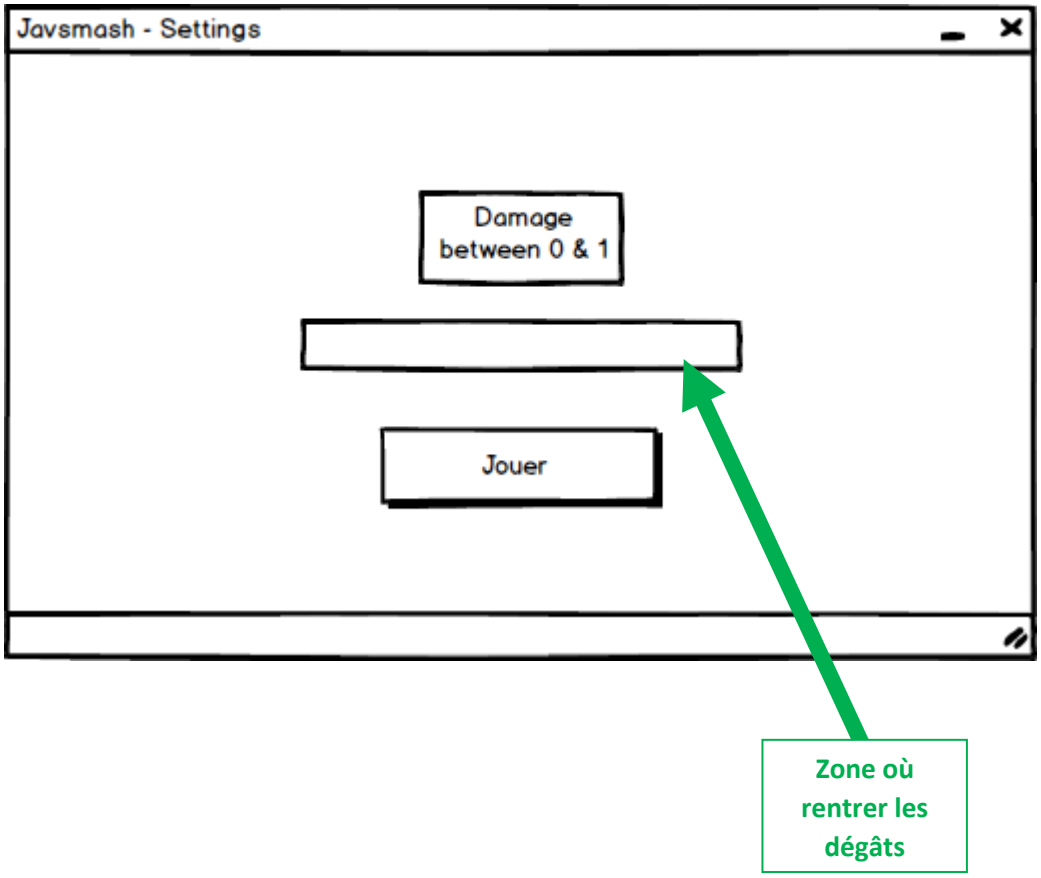
Les statistiques permettent aux différents utilisateurs de voir les résultats des matchs qu'ils ont effectués, des joueurs présents et le joueur qui a gagné.

Les boutons Charger et Sauvegarder permettent – comme leurs noms l'indique – de charger des résultats à partir d'un fichier mais également de sauvegarder ces derniers.



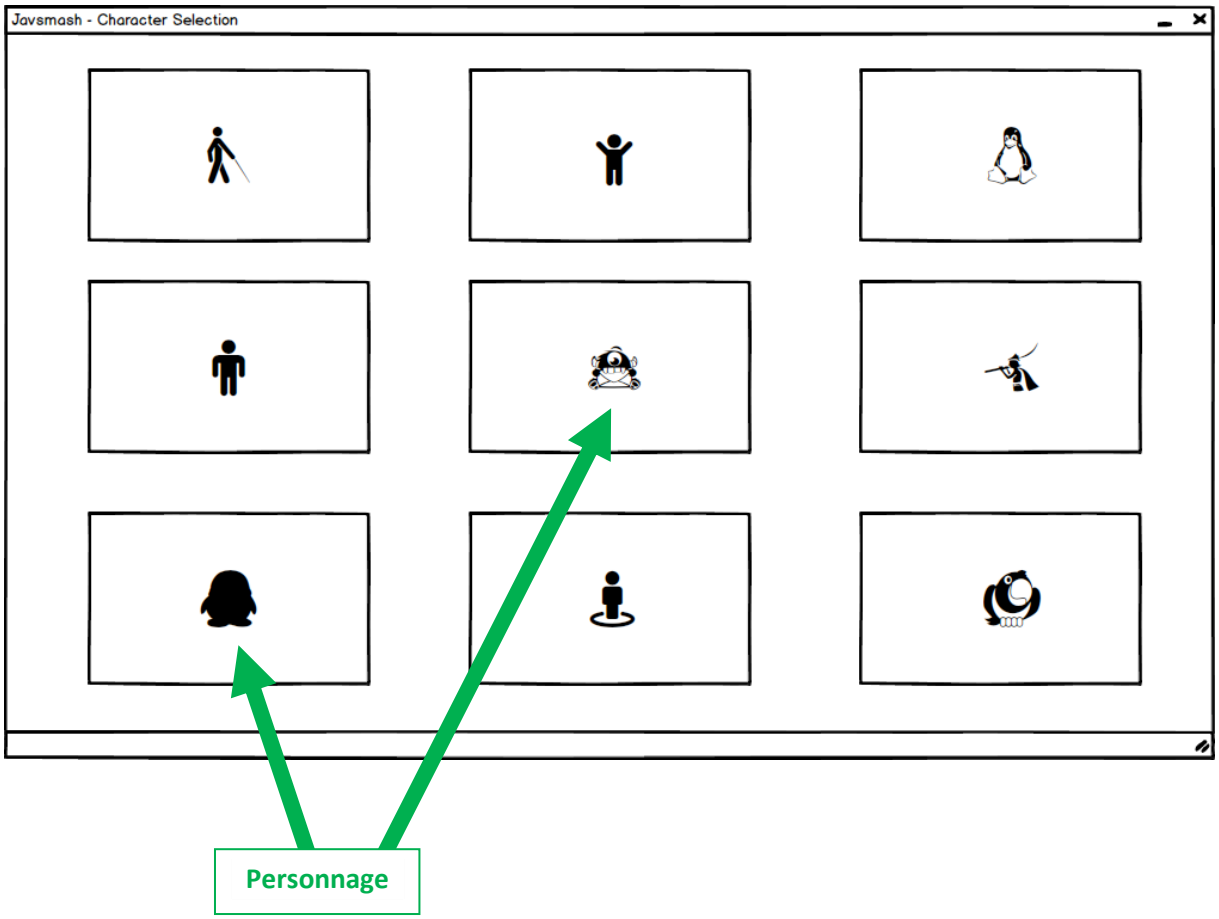
2.3) Page paramètres

Les paramètres – à ce stade de l'application – ne permettent pas une grande possibilité de modification sur le jeu : L'utilisateur peut modifier les dégâts des boules de feu pour augmenter la difficulté (dégâts infimes jusqu'aux boules de feu qui tuent dés le touché). L'interface est simple : une box où rentrer les dégâts désirés et un bouton pour jouer.



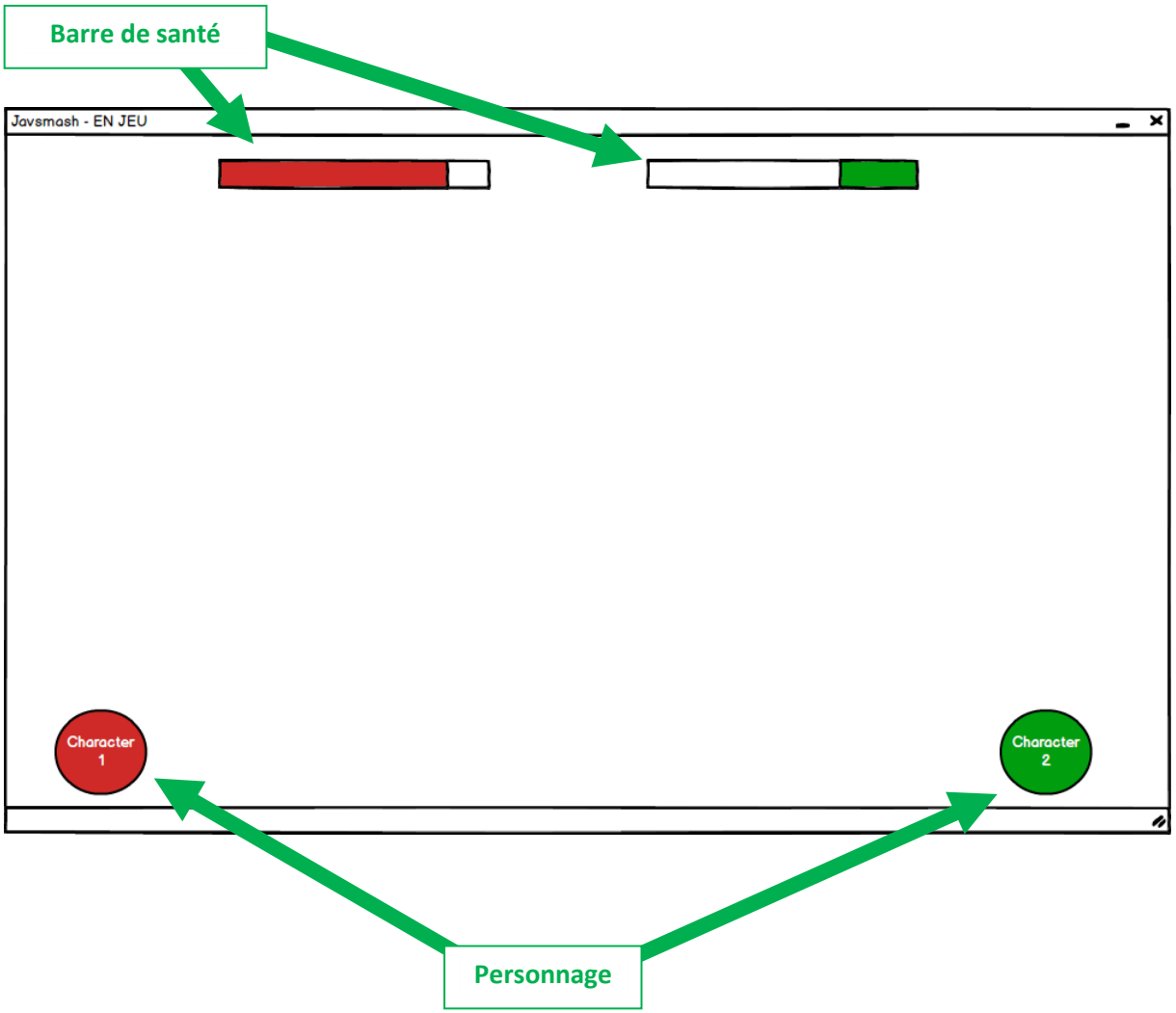
2.4) Page de sélection des personnages

Lorsque l'utilisateur clique sur « Jouer » dans le menu principal, il arrive sur une page où il doit sélectionner des personnages en cliquant dessus. Il n'a pas la possibilité de choisir deux fois le même personnage.



2.5) Page « en Jeu »

Cette page est relativement simple… comme n'importe quel jeu : Une scène où avec des personnages se déplacent, lancent des attaques, saute…



3) Structure de l'application

La structure globale du projet est découpée en deux *packages principaux :* « res » où se trouve toutes les ressources liées au jeu (Images, Sprite, Background, fichiers *FXML…*), et un fichier « src » où se trouvent différentes classes *métiers, contrôleurs*… que je vais détailler par la suite.

3.1) Les relations entre les classes

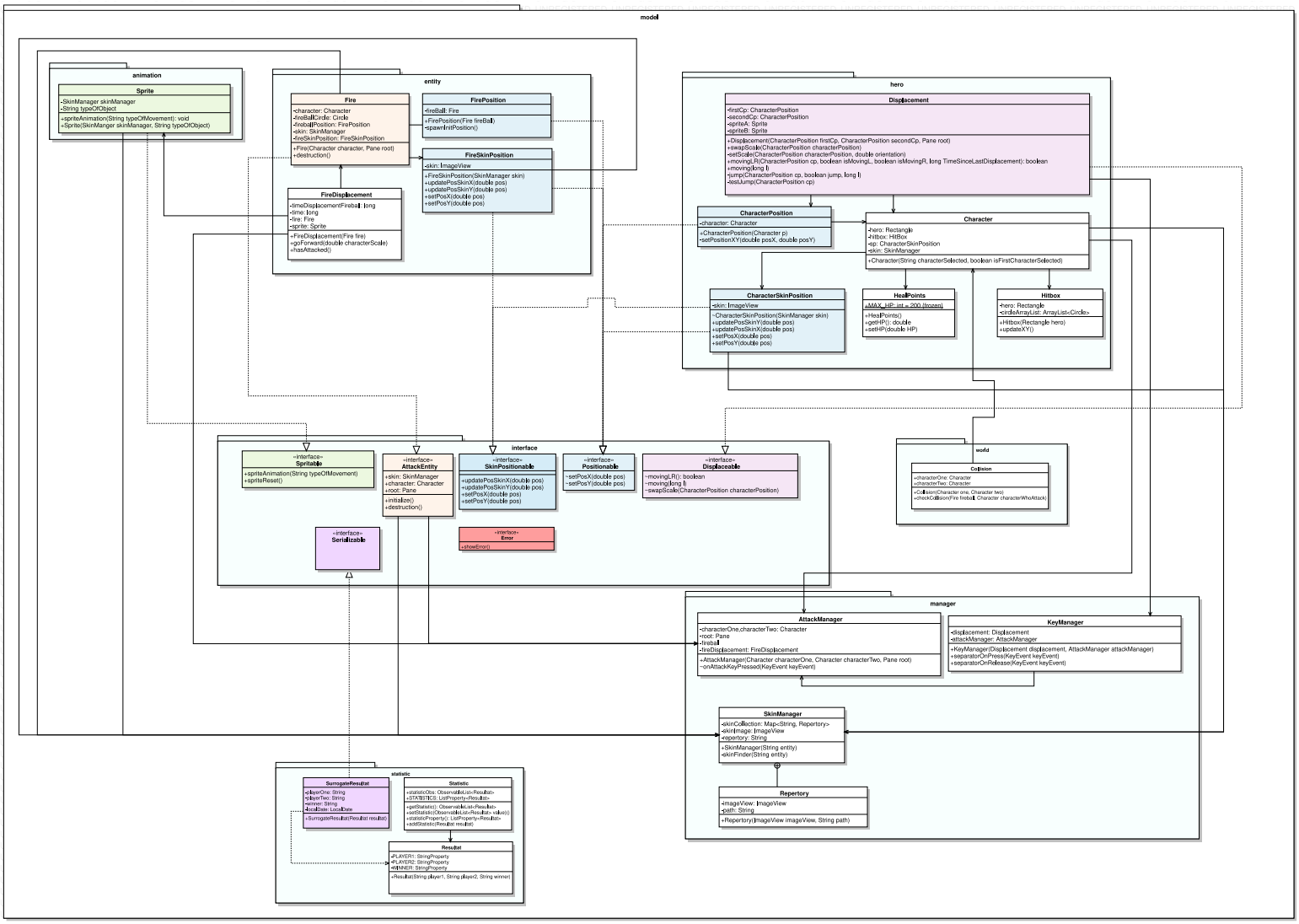
3.1.1) Le package « src »

Comme expliquer plus haut, « src » est le « répertoire père » de notre application : il abrite le package « model » où sont classés les différentes données métiers, le package « view » contenant les *Contrôleurs* et le package « launcher ».

3.1.2) Le package « model »

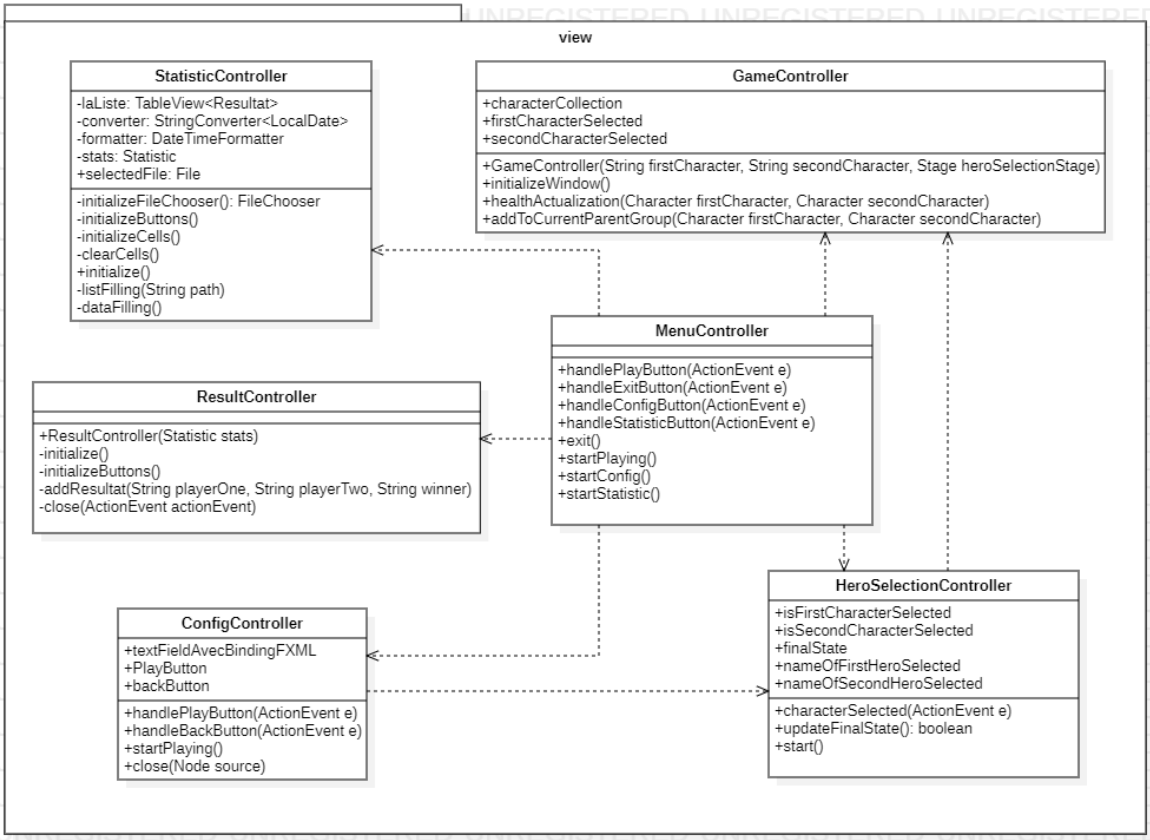
Il regroupe les classes métier de l'application, voici son diagramme de classe.

L'image présente ci-bas est une image en format EMF (vectorielle) donc zoomable à l'infini pour faire apparaître le texte de manière plus clair. Si ce n'est pas assez clair, vous retrouverez l'image sous format PNG dans le répertoire Documentation/Class Diagram sous le nom de PackageModel.png.



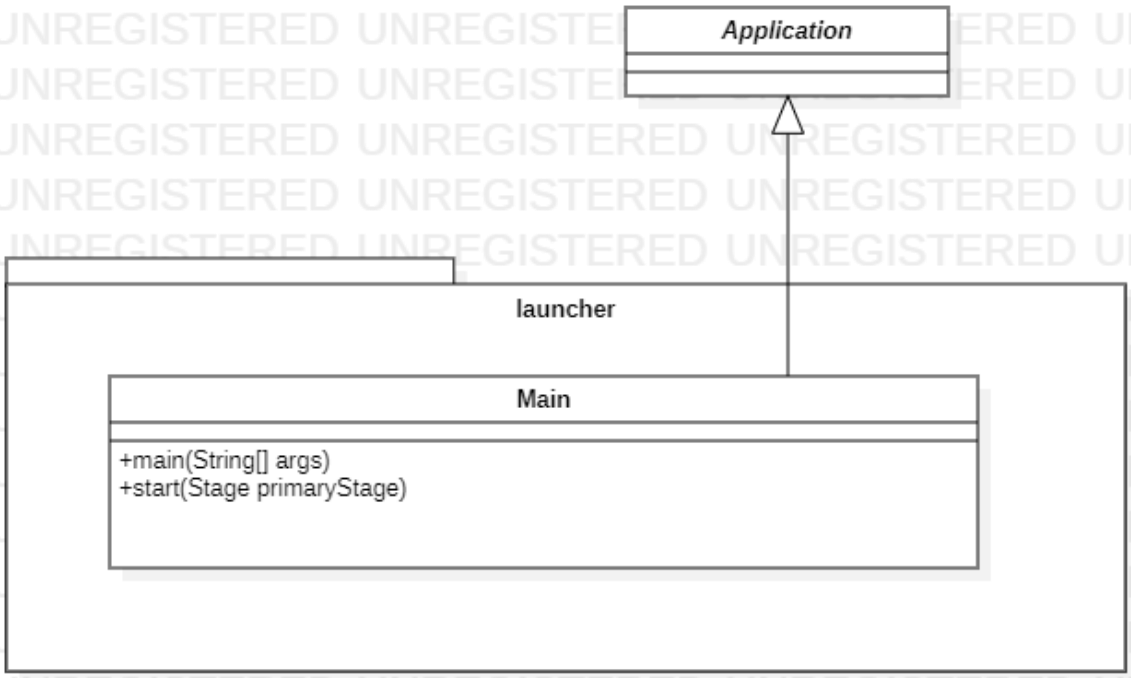
3.1.3) Le package « view »

Ce package contient les différents contrôleurs des vues.



3.1.4) Le package « launcher »

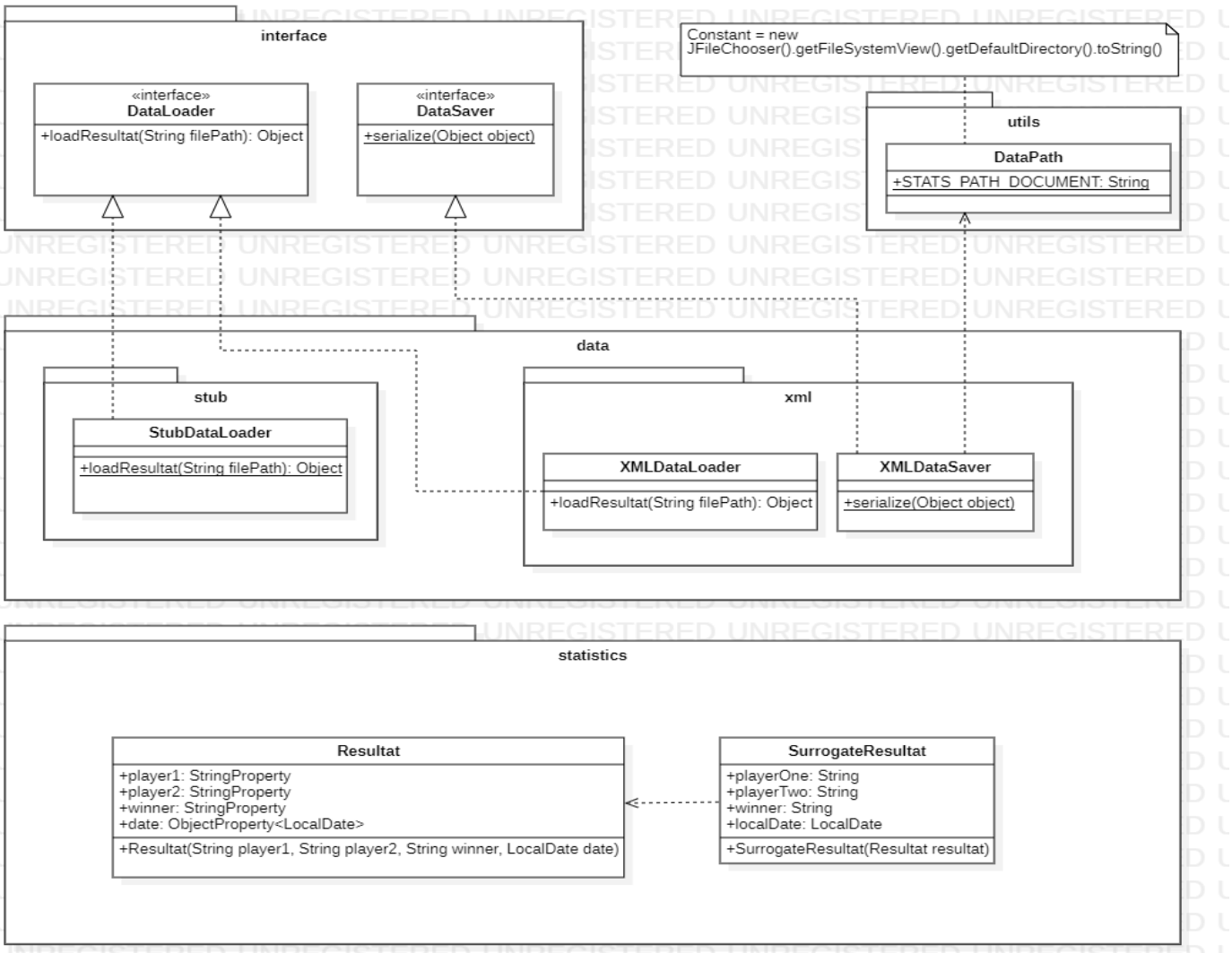
Ce package contient la classe « Main » nécessaire au démarrage de l'application, elle s'étend de la classe abstraite *Application* soit le point d'entrée de l'application.



3.1.5) Le package « data »

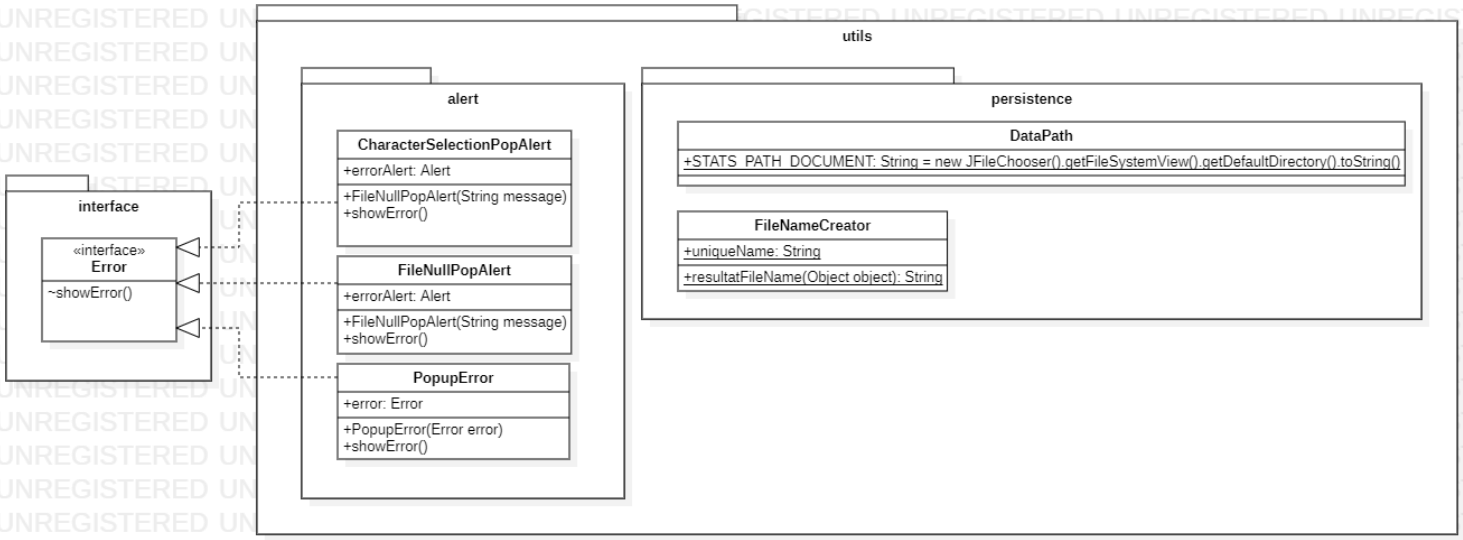
Ce package est dédié à la manipulation des données liées à l'application (statistiques…) et inclus donc le chargement de données ainsi que leurs sauvegardes.

Pour la sauvegarde en fichier nous avons utiliser le patron de conception **Proxy**, car notre objet à Serializer (Resultat) n'était pas Serializable, nous avons donc créer une classe *SurrogateResultat* qui implémente Serializable et qui se construit grâce à un objet Resultat donné en paramètre du constructeur.

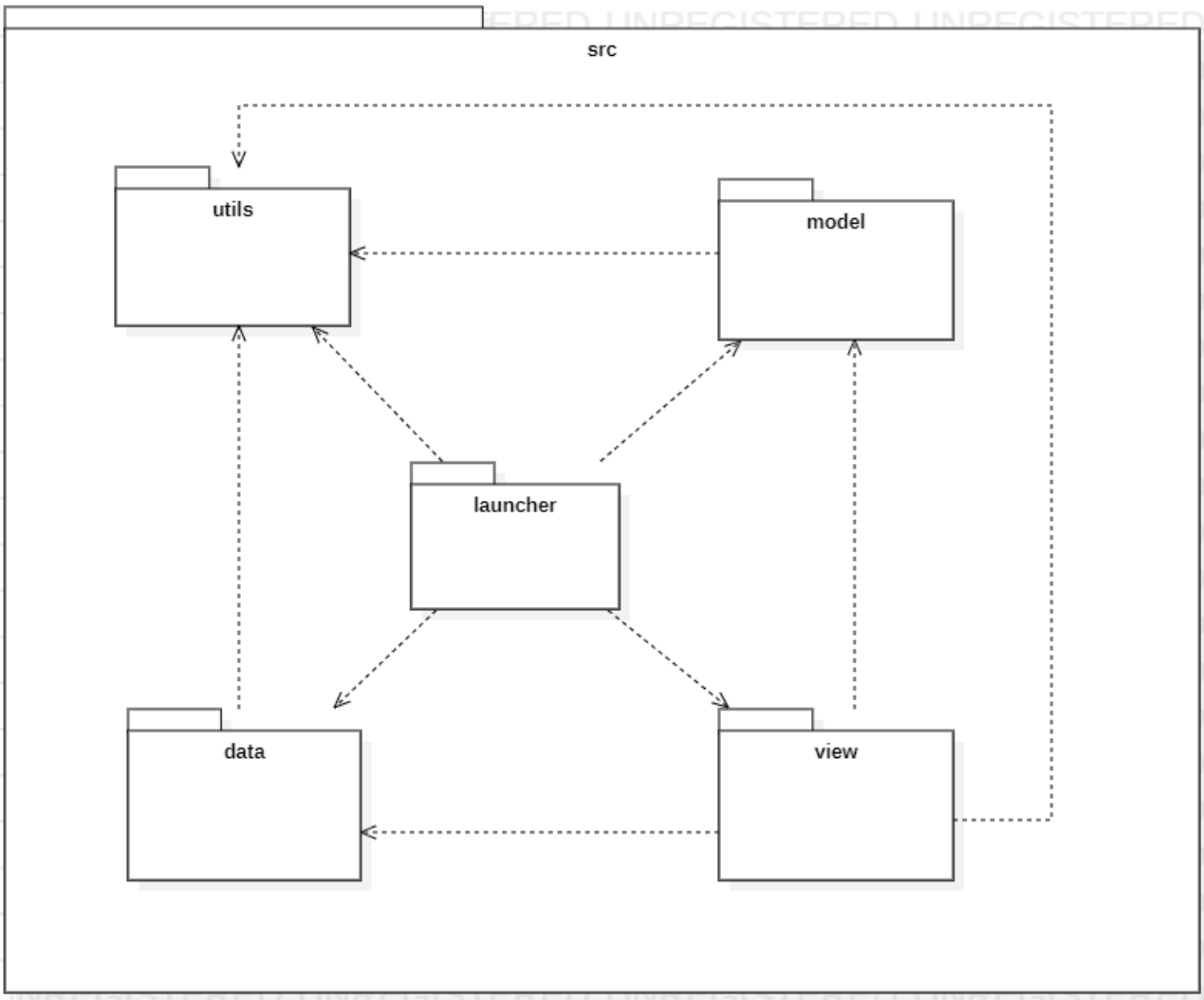


3.1.6) Le package « utils »

Ce package regroupe différentes classes et constantes utiles pour les autres classes de l'application, comme les classes liées à la manipulation des données ou la gestion des Alertes de type *Pop-up*.



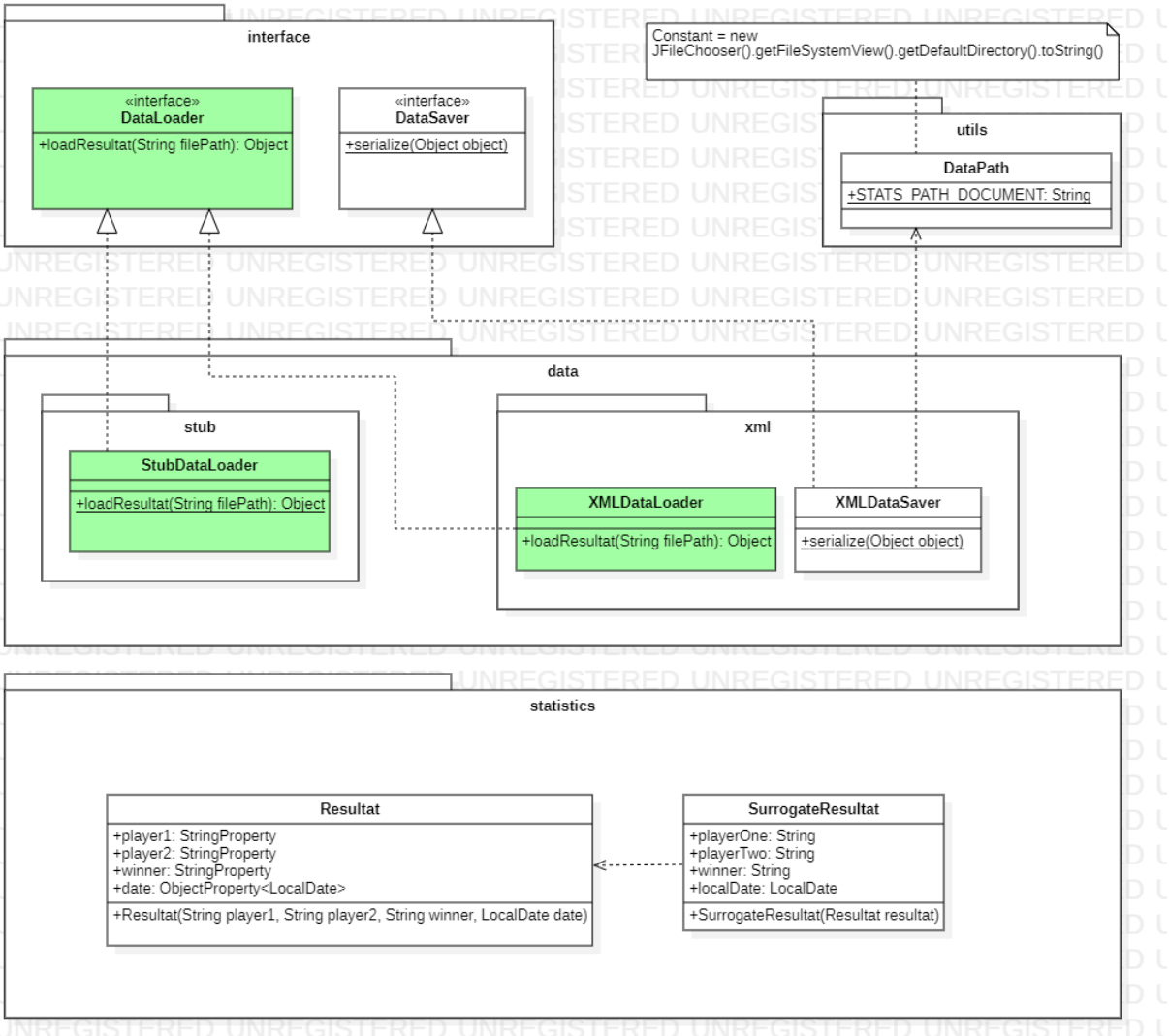
3.2) Les relations entre les packages



4) Patrons de conception utilisés

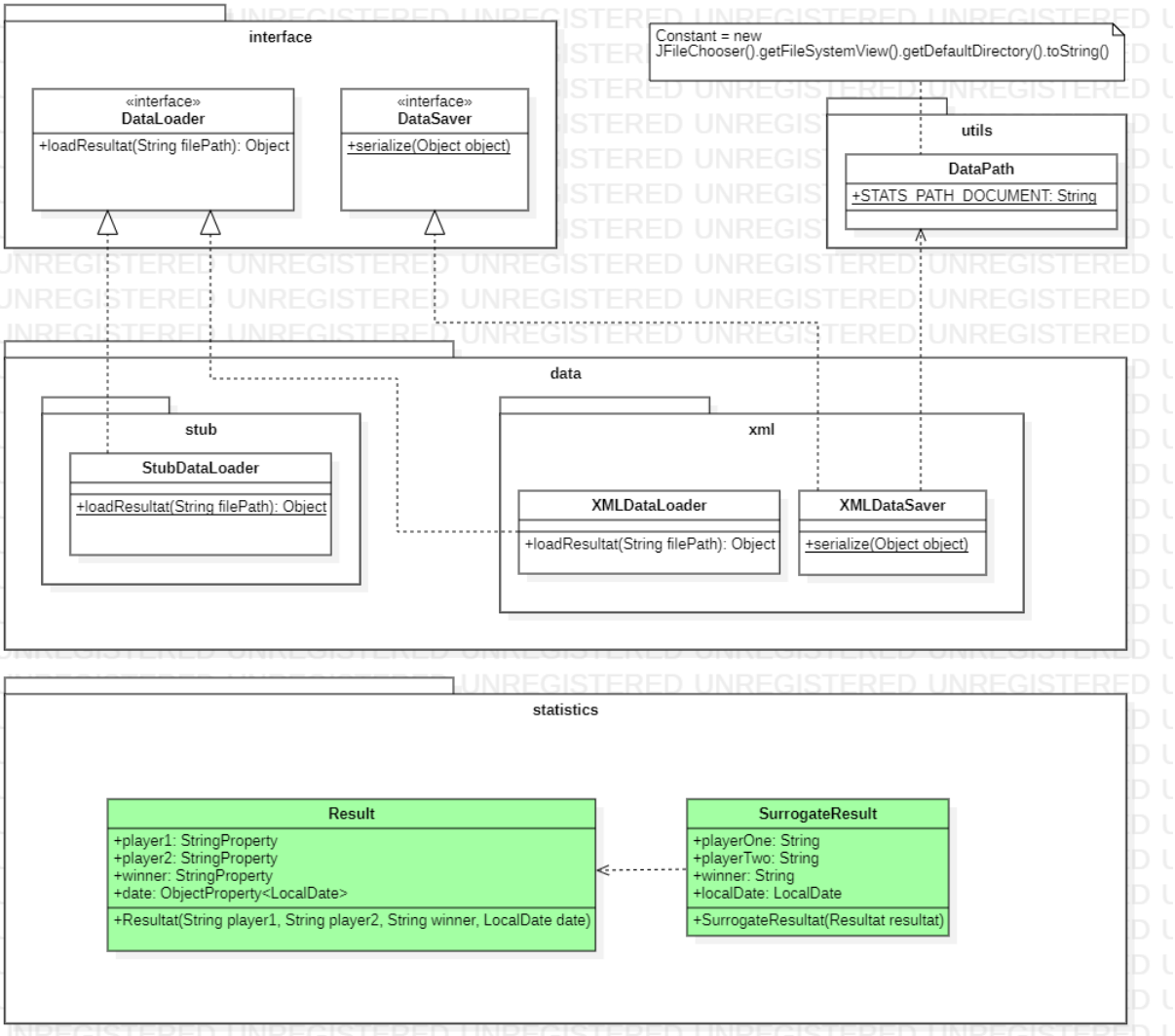
La stratégie :

La stratégie a été utilisé pour les différents processus de chargement de données. Cette dernière est choisie lors du lancement de la page de Statistiques : via le Stub ou via les fichiers, cela dépend de l'existence des fichiers ou non.



Le proxy:

A défaut de pouvoir serializer la classe Result, nous avons créé un *substitut (proxy)* à cette dernière – nommée SurrogateResult qui implémente *Serializable*. C'est le substitut qui est utilisé pour effectuer la sauvegarde et il est construit grâce à un Result donné en paramètre de son constructeur.



5) Etat du projet

5.1) Actuellement et possibilité d'évolutions

Dans l'état actuel des choses, le jeu ne possède pas une grande variété de fonctionnalités, mais les principaux éléments sont présents : Un joueur peut se déplacer de la manière qu'il le désire, sauter, effectuer un double saut mais également lancer des boules de feu.

Les interfaces utilisateurs sont développées de manière à ce que l'utilisation soit intuitive et la plus simple possible avec un menu d'accueil permettant de jouer, accéder aux paramètres du jeu, consulter des statistiques ou bien quitter.

Entre le menu et le jeu, le joueur a la possibilité de sélectionner deux personnages et ainsi donc jouer avec un partenaire en local.

Notre choix a été d'implémenter le minimum nécessaire afin que le jeu soit jouable et présentable mais indéniablement il manque des fonctionnalités et des possibilités pour enrichir l'expérience de l'utilisateur.

L'écran de mort - lorsqu'un joueur n'a plus de vie - n'a pas été implémenté mais il peut l'être plutôt aisément. Le menu de sélection de personnages n'est pas designé et n'est pas optimal en matière de conception.

6) Remerciements

Je tenais à remercier mon collègue Maxime DACISAC pour sa contribution et sa participation au projet, ainsi que notre professeur encadrant Laurent PROVOT. Mais également Damien NGUYEN pour ses conseils précieux pour la conception et l'élaboration de notre projet.