- Je maîtrise la notion d'immuabilité de la classe String. [sur 0.5 point]
  PREUVE :

- Je maîtrise les règles de nommage Java. [sur 1 point]
  PREUVE :
  Les packages sont écrit en minuscule, les noms de classes et interfaces commence par une majuscule, et les variables tout en minuscule en séparant chaque mot constituant la variable par une majuscule, exemple de nom de variable : uneVariable, index..

- Je sais binder bidirectionnellement deux propriétés JavaFX. [sur 1 point]
  PREUVE :

  Bind Bidirectionnel entre le textField et le nomAvion (FenetreStart, ligne 138)

- Je sais binder unidirectionnellement deux propriétés JavaFX. [sur 1 point]
  PREUVE :

  Bind unidirectionnel entre le textField et le nomAvion , si le textField est vide (FenetreStart ligne 145)

- Je sais coder une classe Java en respectant des contraintes de qualité de lecture de code. [sur 1 point]
  PREUVE :
  Je respecte l'indentation dans mes classes pour permettre une lecture aisé du code.

- Je sais contraindre les éléments de ma vue, avec du binding FXML. [sur 1 point]
  PREUVE :

  Binding du score du joueur sur la FenetreJeu (ligne 85)

- Je sais définir et utiliser une classe abstraite. [sur 0.5 point]
  PREUVE :
  J'ai défini plusieurs classe abstraite dans mon projet et utilisé. Par exemple j'ai fait une classe abstraite Entite, qui est ensuite utilisé au sein des classes Projectile et Personnage, cette classe permettant la représentation des entités dans le jeu

- Je sais définir et utiliser une interface. [sur 0.5 point]
  PREUVE :

  Utilisation interface sauvegarde + chargement ISauvegarde et IChargement

- Je sais définir un attribut de classe. [sur 0.5 point]
  PREUVE :
  mot clé static devant un attribut . Exemple : private static joueur;

- Je sais définir une CellFactory fabriquant des cellules qui se mettent à jour au changement du modèle. [sur 2 points]
  PREUVE :
  Utilisation d'une CellFactory dans le fichier FenetreStart.java

- Je sais définir une méthode de classe. [sur 0.5 point]
  PREUVE :
  N'ayant pas besoin de méthode de classe dans notre projet, nous n'en avons pas implémenté, mais pour définir une méthode de classe en java, il faut faire précéder le nom de la méthode de la classe par le mot clé static.

  exemple: public static void exempleMéthode() {}

- Je sais définir une variable ou un attribut constant. [sur 0.5 point]
  PREUVE : Utilisation du mot clé final , exemple : private final nombre;

- Je sais développer une application graphique en JavaFX en utilisant FXML. [sur 1 point]
  PREUVE :

Utilisation du FXML dans les fenêtres de l'application, exemple : FenetreMenuJeu.fxml, FenetreJeu.fxml, FenetreShop.fxml...

- Je sais éviter la duplication de code. [sur 1 point]
  PREUVE :
  J'évite la duplication de code par exemple dans les controller en intégrant dans mon projet un controller abstrait contenant les méthodes qui se répète entre le controller. Fenetre.java

- Je sais hiérarchiser mes classes pour spécialiser leur comportement. [sur 2 points]
  PREUVE :

  Classe Personnage qui a pour classe fille : Joueur et Ennemi , qui ont 2 comportements différents mais des méthodes et attributs en commun.

- Je sais intercepter des événements en provenance de la fenêtre JavaFX. [sur 2 points]
  PREUVE :

  Gestion des clics sur les buttons , Textfield (fenetreArgent ligne 76)

  Gestion d'une sélection d'un élement de la liste view (FenetreStart.java ligne 87)

- Je sais maintenir une encapsulation adéquate dans mes classes. [sur 2 points]
  PREUVE :
  J'utilise les visibilités private, protected et public selon les cas pour avoir une encapsulation adéquate dans mes classes.

- Je sais maintenir, dans un projet, une responsabilité unique pour chacune de mes classes. [sur 2 points]
  PREUVE :

  Cela est visible dans les classes de notre modèle, par exemple Joueur.java contient une classe qui ne représente qu'un joueur, Entite.java une classe qui ne représente qu'une Entite et rien de plus en terme de fonctionnalité.

- Je sais gérer la persistance de mon modèle. [sur 2 points]
  PREUVE :

  La persistance est géré grâce à l'interface ISauvegarde et IChargement, ils sont implémenté dans les classes XMLSauvegarde et XMLChargement pour réaliser la persistance avec des fichier de type .xml

- Je sais surveiller l'élément sélectionné dans un composant affichant un ensemble de données. [sur 2 points]
  PREUVE : ComboBox de la fenetreShop (ligne 106) ou ListView dans fenetreStart (ligne 87)

- Je sais utiliser à mon avantage le polymorphisme. [sur 2 points]
  PREUVE :
  Nous avons utilisé le polymorphisme à notre avantage, par exemple dans la classe joueur et avion nous avons dû leur ajouter en plus de leur constructeur respectif, un autre constructeur ayant aucun paramètre, qui est nécessaire pour la mise en place de la persistance.

- Je sais utiliser certains composants simples que me propose JavaFX. [sur 0.5 point]
  PREUVE : Utilisation des bouttons et grids sur les pages , Canvas sur la fenetreJeu

- Je sais utiliser certains layout que me propose JavaFX. [sur 0.5 point]
  PREUVE :

  Utilisation de GridPane, VBox (FenetreMenuJeu)

- Je sais utiliser GIT pour travailler avec mon binôme sur le projet. [sur 2 points]
  PREUVE :
  Grâce à git nous avons effectué la création de plusieurs branche, dans lequel chaque membre travaillé, et nous mettions en commun notre projet sur la branche master. Nous avons utilisés des fonctions tel que git push (envoyer le projet sur le repo), git pull (récupérer les modification), git add (ajout de fichier

modifié), git commit -m (ajout d'un message au commit) et git checkout (changement de branche, désactiver les changements effectuer …) pour utiliser git à travers l'interface de commande bash.

- Je sais utiliser le type statique adéquat pour mes attributs ou variables. [sur 0.5 point]
  PREUVE :
  J'ai utilise le type statique adéquat pour mes attributs ou variables, par exemple le type String pour le nom du joueur, le type int pour le meilleurScore du joueur etc ...

- Je sais utiliser les collections. [sur 1 point]
  PREUVE : Utilisation de listEnnemi dans le gameManager ou listAvionPosseder comme attribut de joueur.

- Je sais utiliser les différents composants complexes (listes, combo…) que me propose JavaFX. [sur 1 point]
  PREUVE : Combobox de la fenetreShop, ListView de la fenetreStart.

- Je sais utiliser les lambda-expression. [sur 1 point]
  PREUVE : FenetreStart , utilisation d'une lambda expression (ligne 72)

- Je sais utiliser les listes observables de JavaFX. [sur 1 point]
  PREUVE :
  Utilisé dans Joueur.java ligne 46 et dans la FenetreStart pour la liste view.

- Je sais utiliser les packages à bon escient dans un projet. [sur 1 point]
  PREUVE :
  J'ai utilisé les packages à bon escient dans mon projet, par exemple j'ai un package model pour les classes métier, un package pers pour la persistance, un package vue pour les controller etc...

- Je sais utiliser l'API fluent sur les collections Java8 et le binding. [sur 1 point]
  PREUVE : Pas d'utilisation de l'API fluent.

- Je sais utiliser un convertisseur lors d'un bind entre deux propriétés JavaFX. [sur 1 point]
  PREUVE : Ajout d'un StringConverter sur la combobox du menuShop dans la classe FenetreShop (Ligne 85)

- Je sais utiliser un fichier CSS pour styler mon application JavaFX. [sur 0.5 point]
  PREUVE :
  /src/resource/stylesheets, contient un fichier .css utilisé au sein de l'application certains éléments de l'application telle que les boutons.

- Je sais utiliser un formateur lors d'un bind entre deux propriétés JavaFX. [sur 1 point]
  PREUVE :

  Ajout d'un formateur sur le textField de la fenetreUsername qui n'accepete pas les numéros. (Ligne 48)