

### Présentation du contexte

#### La société JMS

La société JMS est une unité d'entretien agréée par le Groupement de Sécurité de l'Aviation Civile (GSAC), administration chargée en France de la sécurité dans le domaine aéronautique. JMS assure la maintenance des avions pour le compte de plusieurs aéro-clubs.

La société JMS emploie du personnel administratif et des mécaniciens. Elle dispose d'un atelier où sont réceptionnés les avions acheminés là pour y subir leurs visites d'entretien, programmées régulièrement. Un magasin jouxte l'atelier et permet aux mécaniciens de disposer immédiatement des pièces détachées utiles à leurs interventions.

L'activité d'entretien des avions implique pour la société JMS la gestion d'un stock important de pièces détachées.

#### Les pièces détachées

Toutes les pièces possèdent un numéro de série. Elles sont rangées chacune dans un casier et accompagnées d'une fiche d'inventaire. Sur chaque fiche d'inventaire, sont mentionnés le N° de série de la pièce, son libellé (exemples : "anémomètre", "horizon artificiel") ainsi que le nombre d'heures de fonctionnement.

On distingue deux types de pièces :

- **Les pièces agréées**

Ces pièces ont reçu un agrément "Aéronautique" valable 2 ans attesté par un document appelé "JAA FORM ONE" délivré par le constructeur. Ce document mentionne la date de l'agrément en vigueur. Après contrôle de la pièce, le constructeur peut décider de ne pas renouveler l'agrément, la pièce est alors supprimée du magasin.

- **Les pièces non agréées**

Certaines pièces stockées dans le magasin – non essentielles pour la sécurité – n'ont pas de "JAA FORM ONE". Dans les casiers du magasin, le document d'agrément est alors remplacé par un ticket qui peut avoir l'une des trois couleurs suivantes :

- VERT : matériel neuf ou réparé
- ORANGE : matériel à réparer ou à réviser après une certaine période d'utilisation
- ROUGE : matériel « à rebuter »

### Ressources


Une application orientée objet permettant de gérer le magasin de pièces détachées, agréées et non agréées est en cours de développement.

Vous disposez des éléments suivants :

- **Diagramme des classes métier UML (Doc 1)**
- **Solution Visual Studio** `SolutionJMS.sln` comportant :
  - o **l'implémentation partielle des classes métier (projet `ClassJMS`)**
  - o **l'implémentation partielle des tests unitaires (projet `ClassJMSTests`)**

## Travail à faire

0.	<p><b>Récupération du projet et mise en place du versionning</b></p> <p><b>0.a) Créer un dossier</b> TP02_JMS, y enregistrer la solution SolutionJMS.sln</p> <p><b>0.b) Initialiser un dépôt Git</b></p> <pre>git init // créer un dépôt Git : initialiser un nouveau dépôt ou convertir un projet existant en un dépôt Git</pre> <p><b>Versionner le code</b></p> <p><b>0.c) Sur Github, créer un nouveau repository JMS</b></p> <p><b>0.d) Synchronisez votre dépôt local avec votre dépôt distant et poussez votre code</b></p>
1.	<p><b>Mise au point de la classe</b> Piece</p> <p><i>Spécifications</i></p> <ul style="list-style-type: none"> <li>- l'attribut <b>numSerie</b> correspond au numéro de série de la pièce</li> <li>- l'attribut <b>libelle</b> correspond au libellé de la pièce</li> <li>- l'attribut <u>protégé</u> <b>nbHeures</b> correspond au nombre d'heures de fonctionnement de la pièce</li> <li>- la méthode <b>GetNumSerie()</b> est un accesseur sur l'attribut privé <i>numSerie</i></li> <li>- la méthode <b>ObtenirInfos()</b> retourne une chaîne de la forme : <b>274 - Courroie</b></li> <li>- la méthode <b>AControler()</b> retourne un booléen indiquant si la pièce doit faire l'objet d'un contrôle ou non. <i>Remarque : Cette méthode sera redéfinie dans les classes dérivées</i></li> </ul> <p><b>1.a) Vérifier la déclaration des attributs de la classe</b> Piece <b>et corriger le cas échéant</b></p> <p><b>1.b) Coder le constructeur de la classe</b> Piece</p> <p><b>1.c) Implémenter les méthodes</b> GetNumSerie() <b>et</b> ObtenirInfos()</p> <p><b>1.d) Vérifier que les tests unitaires de la classe</b> PieceTests <b>s'exécutent avec succès</b></p>
2.	<p><b>Implémentation de la classe</b> PieceAgrée</p> <p><i>Spécifications</i></p> <ul style="list-style-type: none"> <li>- l'attribut <b>dateAgrément</b> correspond à la date du dernier agrément en vigueur</li> <li>- l'attribut <b>nomConstructeur</b> correspond nom du constructeur de la pièce</li> <li>- la méthode <b>RenouvelerAgrément()</b> permet de modifier la date d'agrément de la pièce</li> <li>- la méthode <b>CalculerDureeAgrément()</b> retourne la durée de l'agrément en nombre d'années</li> <li>- la méthode <b>AControler()</b> retourne la valeur <i>true</i> si l'agrément n'est plus valable ou la valeur <i>false</i> dans le cas contraire. Une pièce agréée est à contrôler si l'agrément en vigueur est vieux de plus de deux ans</li> <li>- la méthode <b>ObtenirInfos()</b> retourne une chaîne de la forme : <b>125 - Anémomètre</b> <b>Constructeur : ZZZ</b> <b>Date Agrément : 12/03/2012</b></li> </ul> <p><b>2.a) Créer la classe</b> PieceAgrée</p> <p><b>2.b) Déclarer les attributs</b></p> <p><b>2.c) Implémenter le constructeur</b></p> <p><b>2.d) Implémenter les méthodes</b></p> <p><b>2.e) Vérifier que les tests unitaires de la classe</b> PieceAgréeTests <b>s'exécutent avec succès</b></p>

3.	<p><b>Implémentation de la classe</b> <code>PieceNonAagreee</code></p> <p><i>Spécifications</i></p> <ul style="list-style-type: none"> <li>- l'attribut <b>etat</b> correspond à la couleur (« VERT », « ORANGE » ou « ROUGE ») associée à l'état de la pièce ; état par défaut = « VERT »</li> <li>- l'attribut <b>seuil</b> correspond au nombre d'heures d'utilisation au-delà duquel la pièce doit faire l'objet d'un contrôle</li> <li>- la méthode <b>GetEtat()</b> est un accesseur sur l'attribut privé <code>etat</code></li> <li>- la méthode <b>ChangerEtat()</b> permet de modifier l'état de la pièce</li> <li>- la méthode <b>AControler()</b> retourne la valeur <i>true</i> si la pièce est à contrôler (dans ce cas, son état passe à « ORANGE ») ou <i>false</i> dans le cas contraire. Une pièce non agréée qui est à l'état « VERT » et dont le nombre d'heures d'utilisation égale ou dépasse le seuil</li> <li>- la méthode <b>ObtenirInfos()</b> retourne une chaîne de la forme : </li> </ul> <p><b>3.a) Créer la classe</b> <code>PieceNonAagreee</code></p> <p><b>3.b) Déclarer les attributs</b></p> <p><b>3.c) Implémenter le constructeur</b></p> <p><b>3.d) Implémenter les méthodes</b></p> <p><b>3.e) Créer les tests unitaires pour les méthodes</b> <code>ChangerEtat()</code>, <code>AControler()</code> <b>et</b> <code>ObtenirInfos()</code> <b>en respectant le scénario de test lorsqu'il est indiqué</b></p>
4.	<p><b>Mise au point de la classe</b> <code>Magasin</code></p> <p><i>Spécifications</i></p> <ul style="list-style-type: none"> <li>- l'attribut <b>lesPieces</b> recense toutes les pièces du magasin</li> <li>- la méthode <b>AjouterPiece()</b> permet d'ajouter une pièce à la collection <i>lesPieces</i>. Retourne <i>true</i> si l'ajout est possible ou <i>false</i> dans le cas contraire (cas où le numéro de série de la pièce existe déjà)</li> <li>- la méthode <b>AfficherMagasin()</b> permet d'afficher les informations de chaque pièce du magasin</li> <li>- la méthode <b>ObtenirTauxPNA()</b> retourne le pourcentage de pièces non agréées présentes dans le magasin</li> <li>- la méthode <b>ControlerPieces()</b> retourne la liste de toutes les pièces agréées et non agréées qui doivent faire l'objet d'un contrôle</li> </ul> <p><b>4.a) Implémenter le constructeur de la classe</b> <code>Magasin</code></p> <p><b>4.b) Implémenter les méthodes</b></p> <p><b>4.c) Vérifier que le test unitaire associé à la méthode</b> <code>AjouterPiece()</code> <b>passe</b></p> <p><b>4.d) Créer les tests unitaires pour les méthodes</b> <code>ObtenirTauxPNA()</code> <b>et</b> <code>ControlerPieces()</code> <b>en respectant le scénario de test indiqué</b></p>

## Doc1. Diagramme des classes métiers

