# Rappel important et points à surveiller avant la remise des documents du Sprint 3 et la préparation du Sprint 4

Journaux de bord :

Les dates et les heures sont très importantes. Elles me démontrent votre participation au projet. Le nombre d'heures fait dans le lab et en dehors du lab.

Ça manque bien souvent de détails.

**Il est très important que chaque membre de l'équipe en réalise un.**

Backlog de Sprint :

Concordance avec celui des produits, synchronisation des numéros d'histoires et coloration (vert, jaune, rouge et/ou rien).

Niveau de détails dans la section « description des items à faire ». Parfois le contenu semble important mais est bien souvent trop vague. Plusieurs des items d'histoire que j'ai lus, me seraient difficiles à développer, car les détails sont trop de haut niveau ou sont des histoires en soi !!!

La période de « sprint planning » sert à détailler les items d'histoire en terme de **ce qui doit être fait et comment cela doit être fait**. Il y a une différence entre :

1.1 Faire une classe pour les personnages.
  1.1.1 Faire une classe qui permet éventuellement de représenter les personnages du jeu.

**Et**

1.1 Faire une classe abstraite pour représenter les personnages du jeu

Un diagramme de classe…

***OU***

1.1.1 Nom du package…
1.1.2 Nom exact de la classe…
1.1.3 Liste des interfaces à implémenter…
  1.1.3.1 …
1.1.4 Constantes et valeurs…
  1.1.4.1 …
1.1.5 Liste des attributs, type et domaine de données pour chaque…
  1.1.5.1 …
1.1.6 Signature des méthodes abstraites…
  1.1.6.1
1.1.7 Signature des méthodes concrètes…
  1.1.7.1
1.1.8 Méthodes statiques…

1.1.9 …

1.2 Faire le JUnit de la classe « nom de la classe ».

Faire aussi attention à la différence entre les tests de conformité pour une histoire **fonctionnelle** (tests visuels bien souvent) et une histoire **technique** (tester avec du code bien souvent).

Le code :

Une méthode une action (factorisation).

Division en packages.

**Noms significatifs pour l'auto-documentation** et commentaires dans les parties de code plus complexe.

JavaDoc, toujours.

Supprimer tout le code en commentaire ou inutile, gardez votre code propre.

Formater son code selon les normes.

Tests unitaires, les faire au fur et à mesure et prévoir leur temps de développement dans les détails des items d'histoire.

Bien faire la différence entre **l'héritage** et la **composition**.

Le futur :

Il reste 2 sprints (4 semaines), le 4 et le 5. Il faut donc bien choisir les histoires que l'on veut développer pour que le produit soit pleinement fonctionnel dans 2 sprints et qu'il forme un tout cohérent. Il ne faut pas avoir l'air d'avoir manqué de temps !!! Vos décisions et l'utilisation de votre temps sont très importantes.