# Documentation relative au Développement et à la notation des Versions

## Table des matières



## Introduction

Afin de garder une bonne cohésion lors de modifications à droite à gauche, un système précis de versions de jeux* et de modules doit être mis en place, ce dans le but de rechercher plus facilement des versions antérieures ou tout simplement voir, par la simple version (chiffres), si des modifications de tel ou tel fichier ont eu lieu dans la version jeu* analysée.

*Par jeu(x) comprendre Projet de l'Optimisateur de Wargame, et ce pour toute la suite de ce document

## Développement du jeu

Un dossier "developpement" est présent, par conséquent, ce dossier doit remplir une tâche, et ne pas être inutile. Contrairement aux fichiers de jeux, le fichier développement peut être modifié à foison. Ce dossier peut, et même doit, être structuré de la même manière que la version actuellement en cours de codage.

Un dossier "jeux" est aussi présent, celui-ci contient les différentes versions du jeux (déjà codé). Pour plus d'informations concernant les versions du jeu, voir ci-dessous. **Aucun dossier ou fichier du dossier "jeux" ne doit être supprimé et/ou modifié**. Seul des ajouts peuvent/doivent y figurer. En toute simplicité, lors de modifications du dossier "developpement", vu ci-dessus, ce dernier peut être copié puis collé, dans la bonne version majeure (Voir ci-dessous), puis renommé par son nom de version entière. (Exemple: On travaille actuellement sur des fichiers de la version 1.0.0, on bidouille donc tout ça dans "developpement". On en a finit pour la journée alors on va fixer cette nouvelle version. On copie alors "developpement" et on le colle dans "jeux/V1.X.X/", puis on renomme "developpement" en "V1.1.2" par exemple).

Lors du développement des modules, leur version doit être mis à jour. Détail sur les versions ci-dessous. Ceci permet de voir rapidement les modifications réalisées sur les nouvelles versions. Un système sera mis en place afin de référencer automatiquement les versions de modules d'une version jeu fixée, ainsi nous pourrons comparer facilement deux versions (Ce qui pourra, entre autres, permettre une meilleure détection de l'origine de certains bugs et une meilleure visibilité des améliorations apportées d'une version à l'autre).

## Versions des jeux :

**Modèle :** *Nombre1.Nombre2.Nombre3*

*Nombre1* représente le point fort de la version, c'est à dire que passer d'une version **1.X.X** à une version **2.X.X** correspondant à de (très) grosses modifications de code. Pour exemple, une version **0.X.X** correspondrait à une version non jouable à l'heure actuelle (Ou du moins, de façon non automatisée), une autre version **1.X.X** correspondrait à une version jouable, en console uniquement (De façon automatisée), quant à une autre version **2.X.X**, le jeu serait jouable via une interface graphique. Les versions suivantes pourraient correspondre à des changements majeurs du jeu (Refonte de l'interface graphique, passage d'un plateau simple à un plateau d'hexagones, etc...).

*Nombre2* indiquera bien évidemment un changement moins important que *Nombre1*, néanmoins la ou les modification(s) resteront importantes pour la version signalée. De manière globale, *Nombre2* varie en fonction de modifications plus ou moins importantes de certains modules et/ou objets. Pour exemple, dans une version **X.0.X**, dans le cas où l'ensemble, ou une partie importante, de l'objet Plateau( ) serait modifié, une version **X.1.X** serait appliquée. De même lors de l'ajout de modules tels que les IA, *Nombre2* serait itéré d'une unité.

*Nombre3* est un poids faible de la version, en réalité, il est/sera généralement là pour créer une sauvegarde au cas où les infimes modifications du (nouveau) code poserait problème. *Nombre3* sera itéré notamment lors d'une légère optimisation de code (Du type suppression d'une boucle inutile), ou encore lors du passage d'un self.attribut en argument d'une méthode au lieu de l'avoir de façon brute dans le code, et inversement. En théorie, nous devrions avoir une même expérience de jeu avec une version **X.X.Y** qu'avec une version **X.X.Z**, les modifications étant infimes et relevant plutôt du nettoyage de code. (A moins de ne coder que quelques minutes à peine, Nombre3 ne devrait pas être très utile).

## Où notifier ces versions ?

Chacunes des versions du jeu sera contenu dans des dossiers. Le dossier racine portera le nom de "**V***Nombre1.X.X*", ce dans le but d'épurer une première fois l'accès aux dossiers, ensuite, dans ce dossier sera créé les sous-dossiers+ portant le nom "**V***nombre1.Nombre2.Nombre3*", il y aura donc autant de dossiers que de versions de jeu.

## Avantages et Inconvénients ?

Cette méthode permettra de pouvoir récupérer n'importe quelle version antérieure du jeu, ce afin d'éliminer des possibles bugs amassés au fur et à mesure de certaines modifications, nous aurons aussi la possibilité de retracer l'évolution du projet en toute simplicité. Bien évidemment, cela amplifiera le nombre de dossiers existant, ceci sera essentiellement problématique lors d'un clone complet ou lorsqu'il y aura un grand intervalle entre la version locale actuelle et les nouvelles versions disponibles. Néanmoins, lors de mise à jour régulière (git pull régulièrement), aucun soucis ne sera à signaler, seul les fichiers nouveaux ou modifiés seront téléchargés.

## Versions des Modules :

### Modèle : *Nombre1*.*Nombre2*.*Nombre3*

*Nombre1* est le poids fort de la version. Pour les Modules, itérer *Nombre1* signiferait une refonte totale (ou presque) du module en question. En d'autres mots, *Nombre1* devrait n'être que peu de fois différent de 1.

*Nombre2* représente les modifications importantes du module, notamment l'ajout, la modification (importante) ou la suppression de fonctions, de méthodes d'objets, etc...

*Nombre3* est le poids faible de la version, il représente les modifications mineures réalisées au code. Chaque fonctions, méthodes et autres déjà présent doivent recevoir et fournir les mêmes éléments qu'avant la/les modification(s), les changements se font donc intrinsèquement aux fonctions (et autres), ce pour une histoire de nettoyage de code ou légère optimisation.

## Où notifier ces versions ?

Ces versions de modules devront être notifiées en entête de chacun de leur fichier. On trouvera donc, en début de chacun d'eux, une ligne du type "#Version : 1.0.0".

## Avantages et Inconvénients ?

La notification des versions de modules permettra de connaitre précisémment la version du module utlisé dans tel ou tel version du jeu. Cela sera aussi important lorsque chacun travaille sur une version "personnelle"/chacun de leur côté en même temps. Regarder simplement la version du module pourra permettre d'éviter de devoir rechercher dans le code entier si la personne ayant effectué des modifications sur le jeu a oui ou non effectué des modifications majeures sur un module donné (Ce qui peut donc modifier la façon de coder, par exemple la méthode positionnerUnite(args) de Plateau( ) évite de devoir faire "manuellement" quelque chose comme Plateau( ).troupes[y][x] = Unite(args) en vérifiant préalablement si les coordonnées ne sont pas déjà prise ou non accessible à cause du terrain).