# Manuel développeur

## Environnement logiciel

### Systèmes d’exploitation :

* Windows 10
* Ubuntu 14.04 (Ocaml)

### Logiciels utilisés :

* IDE : intelliJ IDEA 2016.1.2
* Java 8
* JavaFX Scene Builder (par Gluon)
* Emacs avec mode tuareg (ocaml)

## Création assistée d’automate

## Structure du document XML contenant un automate

Le document XML généré s’organise comme ci-dessous :

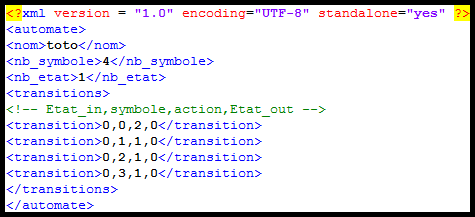


Figure : structure du document XML

Les valeurs modifiables manuellement (sans précautions particulières, hors incohérence) sont les valeurs des balises :

* <nom>  </nom> : nom de l’automate tel qu’affiché dans le jeu
* <nb\_etat> </nb\_etat> : nombre d’états de l’automate
* <transition></transition> : valeur de l’état d’entrée, symbole lu, action à effectuer, état de sortie

La balise <nb\_symbole> ne peut être modifiée qu’en fonction des symboles implémentés dans le moteur de jeu.

## Partie graphique

### UI :

### Fenêtre de jeu :

Elle s’organise en 8 couches graphiques superposées, chaque couche graphique correspond à une grille de type Gridpane, dont chaque nœud contient, à chaque tour, une seule image 32x32pixels.

On affiche une image dans une couche en accédant au tableau « tab3d [colonne][ligne][couche] » qui référence la totalité des images à afficher.

Les 8 couches graphiques sont, du bas vers le haut :

* 0 : décor
* 1 : régénération case, back (consommation case (piégée, non piégé), piégeage case)
* 2 : tombes
* 3 : extinction population, split population (dans les 4 directions)
* 4 : piège en place
* 5 : migration population (dans les 4 directions)
* 6 : population (les 5 niveaux)
* 7 : conflits, action invalide, front (conso case (piégée, non piégée), piégeage case)

Il est donc aisé de remplacer, ajouter ou, supprimer une image (ce dernier cas consiste en une image transparente).