



## Ant Hill

---

Le but de ce projet est de réaliser un simulateur de fourmilière. Ce simulateur régit l'évolution d'une fourmilière en faisant respecter des règles du jeu. Outre la description de la fourmilière en elle-même, le simulateur fait intervenir des fourmis ainsi que des insectes nuisibles aux fourmis. Le comportement des fourmis est dicté par un Dieu des fourmis, qui joue avec la fourmilière au fil de la simulation. Après un réglage initial de paramètres, la fourmilière devra être autonome dans son évolution jusqu'à la mort de la reine.

---

## Bibliothèque graphique

Le projet est à réaliser avec la bibliothèque graphique *lawrence* (disponible à <http://gforgeigm.univ-mlv.fr/projects/lawrence>). Elle doit prendre en charge toute la partie graphique de la manière suivante:

- elle permet l'ouverture d'une fenêtre représentant une grille dont chaque case est remplie par une ou plusieurs icônes (image bitmap ou rendu vectoriel svg) ;
- chaque icône est représentée par un objet d'un certain type `E` et l'application modifie l'affichage d'une case en indiquant la liste de ses icônes sous forme de `Collection<E>`.

Un exemple de jeu de chasse à la grenouille est disponible sur le site web de *lawrence*.

Il n'est pas autorisé de gérer l'affichage graphique par un autre moyen, ni de modifier ou recopier le code du paquetage, ni de placer ses propres classes dans le paquetage `fr.umlv.lawrence`.

## Éléments du jeu

### La fourmilière

La fourmilière contient les éléments suivants, répartis dans des cases, c'est-à-dire des zones géographiques repérées par leurs coordonnées :

- une zone de ponte, possiblement constituée de plusieurs cases contiguës, où sera initialement placée la reine de la fourmilière ;
- des zones prédéfinies où de la nourriture apparaîtra en quantité fixée et se remplissant périodiquement ;
- des obstacles (e.g. paroi, etc.) ; rien d'autre ne peut se trouver sur une case contenant un obstacle ;
- des zones où l'on peut passer (e.g. galerie, chemin, etc.) ;
- des fourmis ;
- des insectes.

La fourmilière est paramétrée à sa création par un ensemble de valeurs (emplacements, caractéristiques, périodicité et rendement des ressources, etc.) spécifiées sous la forme d'un fichier de configuration lu par le simulateur.

### Les fourmis

Les fourmis ont deux caractéristiques essentielles qui sont leur nombre de points de vie et la capacité de leur jabot social <sup>(1)</sup>. Les fourmis ont les capacités suivantes :

- se déplacer <sup>(2)</sup> ;
- ingurgiter une quantité de nourriture à condition que ladite quantité soit présente dans la même case que la fourmi et que son jabot social puisse accueillir cette quantité ;
- régurgiter tout ou partie de la nourriture contenue dans son jabot social.

La reine des fourmis a la particularité de pouvoir créer des fourmis si elle est située dans sa zone de ponte. Le principe général est qu'une reine peut créer une fourmi à partir du contenu de son jabot social.

Les fourmis comme la reine agissent conformément à la volonté de GotA, dieu des fourmis (voir plus loin).

Les fourmis comme leur reine perdent des points de vie au contact des insectes, et meurent lorsqu'elles n'ont plus de point de vie.

<sup>(1)</sup> La trophallaxie est un mode de transfert de nourriture utilisé, entre autres, par les fourmis. Les fourmis possèdent deux estomacs. Lorsque l'une d'elles ingurgite de la nourriture, la majeure partie de celle-ci est stockée dans le second estomac, le jabot social, appelé également estomac social. La trophallaxie consiste en une régurgitation de la nourriture pré-digérée contenue dans ce dernier afin de nourrir d'autres fourmis. On considère dans ce projet que la fourmi n'a pas besoin de s'alimenter pour survivre. Elle ne possède donc qu'un seul estomac -- le jabot social -- qui lui sert à apporter à la reine de la nourriture.

<sup>(2)</sup> Tout déplacement d'une fourmi ou d'un insecte se fait en demandant à la fourmi ou à l'insecte de se déplacer d'une case vers une des quatre cases immédiatement adjacentes (au dessus, en dessous, à gauche ou à droite -- pas en diagonale). Si un obstacle empêche le déplacement, aucune erreur ne sera signalée mais le déplacement n'aura pas lieu. Dans le cas particulier d'un insecte, le déplacement n'est possible que si la case de départ ne contient pas de fourmi. En la présence d'au moins une fourmi, l'insecte ne peut pas se mouvoir.

## Les insectes

Les insectes nuisent à la fourmilière. Les insectes n'ont pas besoin de se nourrir et sont immortels. Un insecte inflige des dégâts (en nombre de points de vie) aux fourmis qui se trouvent dans la même case que lui ainsi que dans les huit cases adjacentes. Un insecte peut se déplacer <sup>(2)</sup> à condition qu'aucune fourmi en vie ne se trouve dans la même case que lui (tant que des fourmis en vie sont dans sa case, il est contraint d'y rester). C'est le simulateur et non GotA qui dicte leur comportement aux insectes.

## Le Dieu des fourmis, GotA

GotA, dieu des fourmis (*God of the Ants*), est un Dieu omniscient qui connaît la fourmilière, sait où sont les insectes, et commande leurs actions aux fourmis. Il doit néanmoins respecter les règles du jeu imposées par le simulateur.

En pratique, GotA interagit de deux manières avec le reste du jeu. D'une part, il dispose d'un ensemble de méthodes qui sont appelées par le simulateur pour lui faire savoir ce qui se passe dans la fourmilière :

- GotA est prévenu lorsque de la nourriture apparaît, sans savoir où ni s'il s'agit de nourriture régurgitée ;
- GotA est prévenu lorsqu'une fourmi ou la reine est blessée ou morte.

D'autre part, GotA peut consulter l'état de la fourmilière à chaque instant, et accéder à chacune des fourmis auxquelles il dicte leurs actions. Plus particulièrement :

- GotA peut obtenir le contenu de toute case de la fourmilière ;
- GotA sait localiser chaque fourmi et connaît leurs points de vie ;
- GotA peut connaître certaines caractéristiques de la fourmilière comme le nombre d'insectes, le coût en nourriture de la création d'une fourmi par la reine, etc.

## Le simulateur

Le simulateur est le maître du jeu, qui régit les interactions entre une fourmilière, des fourmis guidées par leur dieu GotA, et des insectes. Le principe est que le simulateur "laisse" jouer GotA avec ses fourmis dans la fourmilière, en le tenant au courant de certaines évolutions, tout en vérifiant qu'il ne triche pas. C'est le simulateur qui fait bouger les insectes.

Après avoir chargé une fourmilière, des insectes, des fourmis et le dieu GotA, le simulateur établit les paramètres régissant le fonctionnement de la fourmilière comme le nombre d'insectes, les quantité et périodicité d'apparition de la nourriture, etc. Le simulateur démarre ensuite le GotA par l'appel à sa méthode `play()` avec en argument un objet permettant d'interagir avec la fourmilière. Au fil de la simulation, le simulateur appelle des méthodes de GotA pour lui indiquer les évolutions de la fourmilière.

Chaque ordre donné par GotA à une fourmi prend un certain temps qui est paramétrable par le simulateur via un fichier de configuration : pendant cet intervalle de temps, la fourmi est "indisponible" (occupée à la réalisation de son action) et tout nouvel ordre donné par GotA "bloque" ce dernier jusqu'à la fin de la durée nécessaire à la réalisation de l'ordre précédent. Néanmoins, pendant qu'une fourmi exécute un ordre, GotA peut donner des ordres aux autres fourmis ou traiter les informations transmises par le simulateur. À titre indicatif, voici des exemples de "durée" d'une action :

- une fourmi prend 100 millisecondes pour se déplacer ;
- une fourmi prend 200 millisecondes pour manger et 100 millisecondes pour régurgiter ;
- une reine prend 1 seconde pour pondre.

## Principe du jeu et travail demandé

Vous devez principalement réaliser 3 choses pour ce projet :

- le jeu en lui-même, c'est-à-dire le simulateur gérant l'évolution de la fourmilière et ses acteurs avec son interface graphique utilisant la bibliothèque *lawrence* ;
- un ensemble de classes et d'interfaces visibles et utilisables par GotA pour lui permettre de jouer avec ses fourmis et d'influer sur l'évolution de la fourmilière ;
- un GotA particulier qui joue avec votre jeu.

## Rendu

Le projet (simulateur, fourmilière, insectes, fourmis et votre GotA) est à rendre par mail à l'enseignant de cours et aux chargés de TD au plus tard le **7 janvier 2008**. Le format de rendu est une archive au format zip (tout tar.gz, rar, 7z et autre ne sera pas ouvert) contenant :

- un répertoire **src** contenant les sources du projets et les éventuelles ressources (images, sons, etc.) à recopier à côté des classes ;
- un répertoire **classes** **vide**, qui contiendra les classes compilées (ne les incluez pas lors de votre livraison)
- un répertoire **docs** contenant un manuel de l'utilisateur (**user.pdf**) et un manuel qui explique votre architecture (**dev.pdf**) au format PDF ;
- un répertoire **lib** contenant les éventuelles bibliothèques dont a besoin votre projet pour fonctionner (dont celles de *lawrence*)
- un jar exécutable **antHill.jar** qui fonctionne avec **java -jar antHill.jar** et qui possède donc une directive **Class-Path** correcte dans son manifest ; **il est interdit** de mettre les classes de *lawrence* et/ou de *batik* dans **antHill.jar** ;
- un **build.xml** qui permet de
  - compiler les sources (target **compile**)
  - créer le jar exécutable (target **jar**)
  - générer la javadoc dans **docs/api** (target **javadoc**)
  - nettoyer le projet pour qu'il ne reste plus que les éléments demandés (target **clean**)

## Polythéisme et soutenance

Un peu avant la soutenance, nous vous confierons le projet d'un autre binôme (tel qu'il aura été rendu avec documentations, code sources, librairies, etc.) et il vous sera demandé de créer un dieu pour leur simulateur. Le but sera alors de créer un dieu qui triche et qui arrive à faire le maximum de choses interdites (planter le programme fait partie des choses interdites). Du point de vue du développeur du simulateur, veillez donc à ce que votre logiciel ne puisse pas planter, ait une architecture de sécurité, etc.