**FINALE 2000**

# Laisse faire

## Table des matières

# Sujet

## 1 Introduction

Cette année, vous allez devoir travailler dans un monde assez particulier, pour y faire des choses plutôt étranges que nous allons détailler plus loin. Remontons à l'époque de sa création: Dieu est alors un petit garçon de 8 ans, qui mène une vie a peu près

normale: il vient d'entrer en grande section de maternelle.

Et oui, Dieu a un peu de mal a l'école, il est assez dissipe et au lieu d'écouter, il passe son temps à faire toutes sortes d'expériences avec tout ce qui lui passe sous la main. Jusque-la, ça n'a rien donne d'intéressant, mais à force de persévérance, on arrive à tout!

# 2 Lundi: les cubes

C'est la rentrée! Dieu est content, il a une nouvelle maîtresse et elle a l'air très gentille. Bon, elle l'a un peu gronde le matin quand il a voulu tester l'aérodynamisme de son nouveau cahier, mais l'après-midi, elle l'a laisse jouer au cubes autant qu'il voulait dans son coin. Dieu se découvre alors une passion. Des le soir, à la maison, il décide de se faire son propre jeu de cubes!

Il déterre plein de gros cailloux bien cubiques dans la rue en rentrant de l'école, et réussit aussi a fabriquer ses propres cubes comme on lui a montre pendant ses vacances d'été a la plage. (Il a trouve un pot bien cubique dans le salon et l'a vide sur la table. Il est ensuite allé chercher du sable dans le chantier d'à cote qu'il a ramené dans le landau de sa petite sœur, la pauvre).

Dieu a donc déjà trouve deux types de cubes! Il les entasse tous dans la baignoire et fait des jolis reliefs.

## 2.1 Présentation

Le monde est un parallélépipède rectangle de 126*126*21 cubes maximum.

Chaque position dans ce terrain est repérée par ses coordonnées absolues (x, y, z). Ce sont ces coordonnées que vous utiliserez pour toutes vos opérations.

Si les dimensions x et y du terrain peuvent varier selon la carte, la hauteur du terrain est toujours de 21. La première couche de cubes se situe à l'altitude 0 et il ne peut donc pas y avoir de cube à une position de coordonnée z supérieure à 20.

Bien sur, cela ne veut pas dire que le terrain contient des cubes jusqu'a cette hauteur. Chaque case contient un cube ou rien. Il peut y avoir des cubes qui atteignent le plafond, mais en moyenne, le paysage a plutôt une altitude autour de 10 cases. De plus, le sol sera souvent constitue de 2 couches de cubes de roche.

L'évolution du monde s'effectue tour par tour. Nous verrons tout au long du sujet ce qu'il peut se passer pendant un tour donne. Et vous verrez qu'il peut se passer beaucoup de choses!

## 2.2 La pesanteur

Les cubes ne sont pas fixes entre eux, loin de la! Ils sont simplement empiles les uns sur les autres.

Ils sont soumis à une pesanteur: si pour une quelconque raison un cube se retrouve au dessus d'une case vide, il descendra d'une case au tour suivant.

Si plusieurs cubes se retrouvent les uns sur les autres, au dessus d'une case vide, tous les cubes descendront d'une case au tour suivant.

Quoi qu'il arrive, un cube ne peut se déplacer que d'une case par tour. La vitesse d'un cube est donc soit nulle, soit d'une case par tour.

# 3 Mardi: l'eau

Il y a eu une inondation à l'école. Dieu n'a pas trop compris ce qui s'était passe, mais la directrice l'a gronde très fort, et lui a interdit d'aller jouer dans le lavabo avec sa pâte à modeler.

Tout ceci lui donne une idée: il va installer la pomme de douche juste au dessus de ses cubes, et ouvrir le robinet, pour faire la pluie! Il obtient un joli paysage avec des lacs, et même des rivières!

## 3.1 Présentation

Vous le comprendrez plus tard, l'eau occupe une place très importante dans ce monde. Il y en a beaucoup, et heureusement, car elle est très utile. Le paysage est en fait un archipel d'îles, éparpilles dans l'eau. Et pour compléter le tout, il pleut régulièrement.

L'une des difficultés du jeu sera d'être capable de contrôler l'eau, pour en faire la meilleure utilisation possible.

## 3.2 Comportement de l'eau

Chaque case contient une certaine quantité d'eau de 0 à 84 unités. 0 correspond à un case complètement sèche, 84 correspond a une case complètement gorgée d'eau. Les cubes de roche (les paves) font exception car ils ne contiennent jamais d'eau.

L'eau réagit dans le terrain aussi naturellement que possible. Elle est en particulier soumise a la gravite, c'est a dire qu'elle va chercher à se déplacer pour arriver le plus bas possible, qu'elle va s'infiltrer dans les trous et que les nappes d'eau vont s'aplanir (normal, non?).

L'eau peut couler d'une case vers les cases qui lui sont adjacentes pour se déplacer. Elle le fera généralement à la vitesse d'une case par tour.

L'eau ne se déplace pas de la même manière dans tous les cubes. Elle coulera au dessous des matériaux imperméables comme la roche, s'infiltrera dans le sable (de la même manière que dans les cases vides). Il reste une matière spéciale qui sera définie mercredi dans laquelle l'eau aura un tout autre comportement: le coton.

Le coton est un matériau absorbant. Il peut contenir jusqu'a 84 unités d'eau comme les autres, mais les 63 premières sont absorbes. Cette quantité d'eau ne va pas se déplacer par écoulement mais par capillarite, indépendamment de la direction. Le coton réagit un peu comme une éponge.

A chaque tour, l'eau absorbe par un cube de coton a tendance a se repartir dans toutes les cases adjacentes contenant du coton, pour aller vers un équilibre de la quantité d'eau entre ces cases.

Les cases de mer, autour des îles, ont un rôle de régulation. Ce sont à la fois une réserve et un gouffre infinis d'eau. On peut y ajouter autant d'eau que l'on souhaite, mais on peut également y puiser à volonté.

Le niveau de la mer reste donc stable quoi qu'il arrive, et il aura généralement une altitude de 6.

## 3.3 La pluie

Il pleut régulièrement sur le terrain, sous la forme de gouttes d'eau qui arrivent directement sur le sol.

Chaque goutte d'eau contient 21 unités d'eau. Lorsqu'une d'entre elle tombe a des coordonnées (x, y) données, elle arrive directement

à l'intérieur du cube si c'est d'un coton ou dans la case juste au dessus s'il s'agit d'une autre matière ou si la case contient déjà trop d'eau pour tout absorber.

La pluie tombe selon un cycle de 40 tours (une année composée de 2 saisons). Les 20 premiers tours du cycle, c'est la saison humide: à chaque tour, il pleut une goutte sur 1 position (x, y) de la surface de la carte sur 5. Au bout des 20 tours de la saison humide, toutes les cases de la surface auront reçu 4 gouttes exactement (une tous les 5 tours).

Les 20 derniers tours de l'année, c'est la saison sèche: c'est exactement la même chose que pour la saison humide, à part qu'il pleut sur une case de la surface sur 20 a chaque tour. Chaque case de la surface reçoit donc une seule goutte pendant cette saison.

# 4 Mercredi: les haricots

Dieu a été très sage à l'école, aujourd'hui. Et pour cause: sa maîtresse aussi s'est mise à faire des expériences! Elle a donne du coton à tous les élèves de la classe pour qu'ils les mettent dans les pots de yaourts qu'ils ont amenés. Elle leur a ensuite distribué des graines de haricots afin que les élèves les cachent dans le coton. Enfin, la maîtresse a dit que si on les arrose, un haricot devrait pousser rapidement.

Comme Dieu ne savait pas qu'il fallait ramener un pot de yaourt, il n'a pas pu faire l'expérience. Mais il est très malin, il a ramené après l'école le sac de graines et tout le coton qui restait à la maison. Ensuite, il a forme des cubes avec le coton et a vide le tout dans la baignoire.

## 4.1 Présentation

Les haricots sont des plantes qui poussent dans le coton, pour peu qu'on ait mis une graine, que le coton contienne de l'eau et que l'on attende un peu.

Des le départ, une partie des cases de coton du jeu contient des haricots qui ont déjà fini de pousser.

Un cube de coton ne peut contenir qu'une graine à la fois. Une fois plantée, cette graine va se transformer petit à petit en haricot.

## 4.2 Culture des haricots

Il faut 42 étapes pour qu'une graine se transforme complètement en haricot. A chaque tour, on ne passe à l'étape suivante que si la case dans laquelle se trouve le haricot contient assez d'eau (au moins 5 unités), mais n'est pas saturée (84 unités) pour ne pas noyer le haricot.

S'il peut pousser, le haricot absorbe donc 5 unités d'eau sur sa case. S'il ne peut pas, il n'absorbe pas d'eau du tout.

Une fois qu'il est arrive à maturité, le haricot continue a absorber 5 unités d'eau par tour, tant que la case en contient. Il reste cependant dans le même état.

Un haricot peut pousser dans un cube de coton, même si celui-ci est sous la surface.

# 5 Jeudi: les sphères

Pour son goûter, dieu a envie de manger les nouveaux corn-flakes (a base de mais transgénique) qu'il a trouve dans la réserve. Mais comme il y a une plante et plein de terre sur la table, il va manger devant sa baignoire. Et la, alors que Dieu essayait d'ouvrir le paquet, ce qui devait arriver arriva... Les nouveaux corn-flakes commencèrent une nouvelle vie dans la baignoire.

Dieu était à peine parti chercher quelque chose d'autre à manger que les petits tas de corn-flakes se transformèrent, prirent la forme de cubes, et commencèrent à bouger...

## 5.1 Présentation

Les sphères sont des cubes qui ont la propriété d'être autonomes. Elles sont libres de leurs mouvements, et peuvent même soulever d'autres cubes (mais pas pousser ou porter par dessous).

Mise a part la possibilité de se déplacer, les sphères peuvent accomplir diverses actions qui seront détaillées par la suite.

## 5.2 Les déplacements

### 5.2.1 Mouvements

Chaque sphère peut se déplacer dans les 6 directions correspondant aux 6 cases adjacentes à celle ou elle se trouve. Elle est donc

capable de voler et peut aussi faire du sur-place. Elle ne peut se déplacer que d'une case par tour.

### 5.2.2 Transport

A chaque fois qu'une sphère effectue un mouvement ; ou qu'elle reste sur place, elle va être en mesure de tenir un certain nombre de cubes en dessous d'elle. Ils peuvent être de n'importe quel type sauf les puits (voir plus bas) qui sont fixes. Il peut donc même s'agir d'autres sphères qui passeraient par la.

Par contre, il ne doit y avoir aucun vide entre la sphère et chacun des cubes tenus pour que la sphère puisse les attraper. La sphère ne peut prendre que 4 cubes maximum et si elle essaye de porter plus de cubes que possible (pour une raison quelconque), le nombre maximum est attrapé.

Le groupe de cubes ainsi forme est solidaire lors des déplacements.

Enfin, si une sphère fait une autre action qu'un déplacement (ou du sur-place) à un tour, elle lâche automatiquement sa cargaison.

### 5.2.3 Fatigue

Chaque sphère dispose d'une quantité d'énergie limitée à 64 unités. Chaque déplacement a un coût énergétique qui varie selon le type et le nombre de cubes transportés.

Un déplacement vers le bas ne coûte pas d'énergie.

Les déplacements horizontaux ont des coûts dépendants du nombre de cubes transportes:

Aucun cube porté: 7 unités
1 cube porté: 9 unités
2 cubes portés: 12 unités
3 cubes portés: 21 unités
4 cubes portés: 64 unités

Un déplacement vers le haut coûte exactement le double du déplacement horizontal, pour le même nombre de cubes portes.

**Attention**: ne pas bouger est considère comme un mouvement horizontal.

Lorsqu'à la fin d'un tour, la sphère et ce qu'elle vient de porter sont poses sur un cube, la sphère récupère toute son énergie (64 unités).

### 5.2.4 Echec des déplacements

La réussite du déplacement des sphères n'est pas garantie, il y a en effet de nombreux cas ou le déplacement est empêche ou impossible.

La sphère doit tout d'abord avoir suffisamment d'énergie pour réaliser ce qu'elle désire:

- Si une sphère tente de se déplacer vers le haut, mais n'a plus assez d'énergie, on considère qu'elle essaie de ne pas bouger.
- Si une sphère tente de se déplacer horizontalement, ou de faire du sur-place, mais n'a pas assez d'énergie, elle descend.

Si une sphère a assez d'énergie pour effectuer un déplacement, mais que celui-ci est impossible pour une autre raison, elle utilise quand même l'énergie correspondant à ce déplacement.

Voici la liste des cas dans lesquels une sphère ne peut pas se déplacer bien qu'elle ait assez d'énergie:

- L'endroit ou elle veut aller (en comptant sa cargaison) est occupe par un cube qui ne va pas se déplacer. (Ok, c'est une définition récursive qui ne suffit pas a définir tout ce qui se passe. Si ce n'est pas déterminé, bah... de toutes façons ça ne devrait pas arriver souvent! Quoi qu'il arrive, deux cubes ne peuvent pas se croiser.)
- Une autre sphère (ou son chargement) ou un quelconque cube, essaie aussi de se déplacer vers l'une des cases que la sphère (ou son chargement) essaie d'atteindre. Dans ce cas, seul celui qui descend, s'il y en a un, effectue son déplacement.
- La sphère est bloquée ou prisonnière (voir ci-dessous).

Pour finir, les sphères peuvent se déplacer sans se soucier de la quantité d'eau qui se trouve dans les cases par lesquelles leur cargaison et elles passent. Elles n'ont pas besoin de respirer (n'oubliez pas leur origine) et elles n'influent pas sur le comportement de l'eau.

## 5.3 Blocage et capture

Lorsqu'une sphère q juste au-dessus d'elle un ou plusieurs cubes posés (qui ne sont donc tenus par aucune autre sphère) ou une autre sphère qui veut descendre elle est dite bloquée.

Lorsqu'une sphère fait partie de la cargaison d'une autre sphère, elle est dite prisonnière.

Les sphères bloquées ou prisonnières ne peuvent faire aucune action (en particulier les déplacements) et elles lâchent ce qu'elles portent.

Une sphère en vol qui se fait bloquer va descendre automatiquement jusqu'au sol ou elle restera coincée jusqu'à ce qu'on la libère.

Les sphères prisonnières sont considérées comme des cubes de matière quelconque dans les déplacements. L'échec du déplacement d'une sphère n'influe pas sur sa cargaison, c'est à dire que les éventuelles sphères de la cargaison restent quand même prisonnières.

## 5.4 La perception

### 5.4.1 La vision

Les sphères possèdent un oeil sur chaque face! Elles peuvent donc voir partout autour d'elles. Elles peuvent observer tout ce qui est visible dans un cube de 15 cases d'arête dont elles sont le centre.

Elle ne sont cependant pas capables de voir a travers les cubes de matière (autre que le vide). Une sphère ne peut voir un cube donne que s'il se trouve dans son cube de vision, et que la droite entre le cube regarde et elle ne passe que par des cubes vides. Par contre, ces derniers peuvent contenir n'importe quelle quantité d'eau, celle-ci ne gêne absolument pas.

### 5.4.2 La sonde

En plus de leurs yeux, les sphères possèdent sur leur face inférieure une sonde permettant d'analyser les cubes qui se trouvent en-dessous d'elles. La sonde est liée à leur capacité de transporter des cubes. Elles ne sont donc capables que de sonder les cubes qu'elles pourraient porter (mêmes règles: 4 cubes maximum et il ne doit pas y avoir de cases vides entre le cube à sonder et la sphère).

## 5.5 Les messages

Les sphères sont télépathes. Elles peuvent entre autres percevoir les sentiments de toutes les autres sphères, quelle que soit la distance qui les sépare.

Des qu'il arrive quelque chose d'intéressant à une sphère, toutes les autres reçoivent un message décrivant ce qui se passe.

Les différents événements qui provoquent l'émission d'un message sont:

- Je suis bloquée.
- Je suis prisonnière.
- Je récolte un haricot.
- J'ai déposé des haricots dans un puits.

Ces messages sont émis au moment ou l'événement a lieu et sont répètes s'il dure plusieurs tours.

## 6 Vendredi: la récolte

Dieu est tout surpris: en rentrant a la maison vendredi soir, sa baignoire a pris vie. Il remarque que les sphères ont une activité intense. Il essaie de leur montrer qu'on peut ramasser les haricots, qui ont maintenant bien pousse.

Et surprise, les sphères comprennent, et commencent à récolter tous les haricots. Comme au bout d'un moment, elles portent trop de haricots pour continuer à en ramasser, il se dit qu'il leur faudrait quelque chose pour pouvoir les déposer.

Par chance, dans le placard de la salle de bains, il trouve plein de rouleaux de sopalin. Il récupère les tubes de carton du milieu, et les installe dans la baignoire pour faire des sortes de puits ou les sphères pourront déposer tous leurs haricots!

### 6.1 Plantage et récolte

Les sphères sont capables de planter des graines et récolter des haricots. Le plantage et la récolte se font toujours dans la case juste en-dessous de la sphère. Si il n'y a pas de cube de coton dans cette case, il ne se passe rien.

Une sphère ne peut récolter un haricot qu'une fois qu'il a entièrement poussé. Si elle essaie d'en récolter un qui n'est pas

arrive à maturité, il ne se passe rien.

Une sphère ne peut transporter que 10 haricots en même temps. Si elle essaie d'en ramasser un autre alors qu'elle en possède déjà 10, il ne se passe rien.

Une sphère peut planter des graines de haricots, si elle en a en réserve. Si elle essaie de planter une graine alors qu'elle n'en possède pas, elle va automatiquement transformer un de ses haricots en 3 nouvelles graines dont une sera plantée. Si elle ne possédait aucun haricot, rien ne se passe. Une sphère ne peut donc pas posséder plus de 2 graines, car il n'existe aucun autre moyen d'en acquérir.

Au départ, les sphères n'ont ni graine ni haricot. Elles doivent aller récolter des haricots sauvages (présents sur la carte au début) avant de pouvoir planter les leurs.

## 6.2 Les transactions

Les sphères ont la possibilité de s'échanger les haricots qu'elles possèdent. Pour le faire, il suffit que deux sphères viennent au contact l'une de l'autre (dans 2 cases adjacentes quelconques) ; l'une peut alors donner une partie de ses haricots à l'autre.

Les sphères ne peuvent bien sur pas donner plus de haricots que ce qu'elles possèdent, mais la limite de 10 haricots doit aussi être respectée. Si une sphère tente de donner plus de haricots que possible, le nombre de haricots échangés est limité.

## 6.3 Les puits

Une fois des haricots récoltés, les sphères peuvent (doivent) aller les déposer dans les puits.

Un puits est constitue d'une colonne de cubes de type puits pose sur le fond de la carte et qui monte jusqu'à une certaine hauteur (généralement jusqu'à la surface du terrain).

**Attention**: les cubes des puits sont fixes (les sphères ne peuvent pas les attraper) et ils laissent filtrer l'eau comme le sable.

Les haricots mis dans les puits tombent au fond et y restent définitivement. Il n'y a pas de limite au nombre de haricots qu'on peut y déposer.

Le dépôt de haricots dans un puits fonctionne comme une transaction. Il suffit que la sphère vienne au contact d'un des cubes qui forment le puits (dessus ou sur un cote) et elle lui "donne" des haricots.

Les puits sont repartis de manière à peu près homogène sur l'ensemble du monde. En moyenne, il y aura un puits pour une surface de 25*25 cases.

## 7 Samedi: la peinture

Le samedi matin, à l'école, on fait de la peinture! Dieu, lui, n'a rien dessiné. Il a préfère ne rien gaspiller...

Il passe toute son après-midi à essayer d'attraper une par une, toutes les sphères de la baignoire. Il y avait juste assez de peinture pour tout le monde!

Maintenant que les sphères sont peintes de toutes les couleurs, il va pouvoir faire des équipes!

## 7.1 Les races

Puisque les sphères n'ont plus toutes la même couleur, elles finissent par se regrouper selon leur type. Elles commencent a développer leurs capacités de télépathie au sein de ces groupes et a trouver des techniques spécifiques de plantation pour affirmer leur sentiment d'appartenance à une communauté!

Par contre, elles deviennent sensiblement agressives envers les sphères des autres couleurs. Rien de bien méchant, rassurez-vous!

## 7.2 La communication

Avec le développement de leurs capacités, les sphères apprennent a mieux contrôler leurs sentiments et leur expression. Elles deviennent alors capables de s'échanger des messages précis au sein de leur communauté.

## 7.3 La spécialisation des récoltes

Bien que les graines et les haricots produits soient les mêmes, chaque peuple de sphère cultive les haricots d'une manière particulière.

Les sphères vont pouvoir ramasser tous les haricots de la carte, mais elles vont se spécialiser dans la récolte des cultures de leur peuple. Elles seront alors plus rapides a récolter leurs haricots que ceux des autres. Voici les temps de récolte des différents haricots:

haricots de leur communauté: 1 tour
haricots d'une autre communauté: 5 tours
haricots sauvages: 1 tour

## 7.4 Les interruptions de récolte

La séparation de la population en plusieurs communautés est aussi source de tensions entre les différentes sphères. Elles ont rapidement tendance à se jouer des mauvais tours en particulier en voyant les sphères des autres peuples faire des récoltes.

A chaque fois qu'une sphère passe au contact d'une sphère d'un autre peuple qui est en train de récolter un haricot, celle-ci la dérange automatiquement. Cela a pour effet d'interrompre la récolte. Le haricot reste à sa place et la sphère qui récoltait doit recommencer depuis le début.

# 8 Dimanche : fight!

Tout est prêt! Dieu a ramène tous ses copains dans sa salle de bains. Ca tombe bien, il a autant de copains que de tubes de peinture vides. Il va pouvoir organiser un super concours! Le but du jeu sera de récupérer le plus grand nombre de haricots!

## 8.1 Présentation des parties

Il va y avoir un peuple par joueur. Chaque peuple aura le même nombre de sphères. Les sphères vont évoluer dans le terrain et vont devoir ramener les haricots dans les puits.

Au départ, toutes les sphères d'un même peuple sont regroupe à un endroit quelconque de la carte.

Plusieurs méthodes d'évolution sont possibles, mais la principale tache des sphères sera de modifier le terrain de manière à être le plus productif possible.

## 8.2 Possession des puits

Au début de la partie, les puits n'appartiennent a personne. Des qu'une sphère déposé un haricot, le puits passe en possession du peuple de la sphère et le reste jusqu'à la fin du jeu.

Les sphères peuvent déposer leurs haricots dans n'importe quel puits, quel que soit le peuple auquel celui-ci appartient. Le joueur vainqueur est celui qui a le plus de haricots dans son ou ses puits à la fin de la partie.

## 9 Conclusion

Croyez-vous que ce sujet a pour origine diverses réflexions philosophiques ou métaphysiques profondes à propos de l'apparition de l'homme sur terre, les mystères de la création de l'univers, et sur l'existence et le rôle d'un créateur universel? Certainement pas une réflexion philosophique ni métaphysique! Toute ressemblance avec des événements ou des personnages ayant réellement existe ne serait qu'une pure coïncidence. De plus, pour les associations de défense et de protection des céréales battues, sachez que pendant la réalisation du serveur, aucun animal ou autre forme vivante (hormis notre cher Président, quelques Hamsters et un Patatos) n'a été maltraite de quelque manière que ce soit.

## 10 Ce que vous devez faire

Votre but va être d'écrire un programme permettant de contrôler une sphère. Ce programme sera lance plusieurs fois parallèlement pour gérer chacun une sphère.

Vous ne devez pas écrire le corps du programme, car il est déjà contenu dans la librairie qui vous est fournie. Vous devez simplement écrire trois fonctions, qui sont appelées par la librairie:

- La fonction appelée au premier tour de la partie : InitGame.
- La fonction appelée à chaque tour : PlayTurn.
- Une fonction qui vous permet de donner le nom de votre champion, appelée au lancement du programme : GetClientName.

Pour générer votre exécutable, nous vous fournissons des scripts qui automatisent la compilation (avec une surprise a la clé). Une documentation vous sera distribuée des que nous l'aurons écrite.

De son cote, le serveur vous met une série de fonctions à

disposition. Celles-ci vous permettront d'obtenir des informations et donner des ordres a vos sphères. Des librairies existent pour différents langages, et les prototypes et la description des fonctions sont fournis en annexe.

## 11 Temps de réponse

Vous disposez à chaque tour d'environ 500ms (cette valeur pourra changer en fonction du déroulement de la finale) pour que TOUS vos clients effectuent leurs opérations et sortent de PlayTurn.

En cas de dépassement, nous ajouterons peut-être une réserve de temps. Vous pourrez alors y puiser en cas de dépassement de la limite. Quand cette réserve atteint 0, le joueur n'est pas élimine mais les sphères de son peuple deviennent complètement inertes.

## 12 La librairie

Voici la liste des différentes fonctions accessibles depuis la librairie (que vous ne verrez jamais d'ailleurs).

Toutes les fonctions travaillent dans le système de coordonnées absolues de la carte.

## 12.1 GetCubeAtPos

Java: `public int GetCubeAtPos(int x, int y, int z);`
Caml: `external get_cube_at_pos : int -> int -> int -> int`
C: `extern signed GetCubeAtPos(unsigned char x, unsigned char y, unsigned char z);`
Pascal: `function GetCubeAtPos(x, y, x: ShortInt): LongInt;`

Cette fonction permet d'obtenir de type de cube contenu dans la case de coordonnées (x, y, z). Voici la liste des valeurs retournées:

- -2 si la case n'est pas dans la carte.
- -1 si la case n'est pas visible ou sondable.
- 0: Case vide, la case ne contient aucun cube. L'eau y coule sans résistance.
- 1: Sphère, il s'agit d'une case normalement vide ou se trouve une sphère.
- 2: Cube de coton, c'est le matériau dans lequel on fait pousser les haricots. L'eau s'y propage par capillarité.
- 3: Cube de sable, c'est un matériau basique dans lequel l'eau

s'infiltre très facilement.

4: Cube de roche, c'est un matériau basique imperméable, il ne contient jamais d'eau.

5: Cube de puits, c'est le matériau qui sert a composer les puits. L'eau s'y propage comme dans le sable.

## 12.2 GetWaterAtPos

Java: `public int GetWaterAtPos(int x, int y, int z);`
Caml: `external get_water_at_pos : int -> int -> int -> int`
C: `extern signed GetWaterAtPos(unsigned char x, unsigned char y, unsigned char z);`
Pascal: `function GetWaterAtPos(x, y, x: ShortInt): LongInt;`

Cette fonction permet d'obtenir la quantité d'eau contenue dans la case de coordonnées (x, y, z). Voici la liste des valeurs retournées:

-2 si la case n'est pas dans la carte.

-1 si la case n'est pas visible ou sondable.

n correspondant à la quantité d'eau comprise entre 0 et 84 inclus.

## 12.3 GetBreedAtPos

Java: `public int GetBreedAtPos(int x, int y, int z);`
Caml: `external get_breed_at_pos : int -> int -> int -> int`
C: `extern signed GetBreedAtPos(unsigned char x, unsigned char y, unsigned char z);`
Pascal: `function GetBreedAtPos(x, y, x: ShortInt): LongInt;`

Cette fonction permet d'obtenir l'attribut de race du cube dans la case de coordonnées (x, y, z). Cette fonction a uniquement un sens si la case contient une sphère, un cube de coton avec une plantation ou un cube de puits. Elle représente respectivement le peuple auquel la sphère appartient, le peuple qui a plante la graine de la culture ou le peuple qui possède le puits. La valeur retournée peut être:

-3 si la case ne contient pas quelque chose qui possède un attribut de race (un cube de sable ou un cube de coton sans culture...).

-2 si la case n'est pas dans la carte.

-1 si la case n'est pas visible ou sondable.

0 si la race n'est pas déterminée. Cette valeur arrive uniquement

dans le cas d'un haricot sauvage ou d'un puits encore inutilisé.

n dans les autres cas ou la valeur correspond au numéro du peuple concerne.

## 12.4 GetIDAtPos

Java: `public int GetIDAtPos(int x, int y, int z);`
Caml: `external get_id_at_pos : int -> int -> int -> int`
C: `extern signed GetIDAtPos(unsigned char x, unsigned char y, unsigned char z);`
Pascal: `function GetIDAtPos(x, y, x: ShortInt): LongInt;`

Cette fonction permet d'obtenir le numéro d'identification de la sphère située dans la case de coordonnées (x, y, z). Chaque sphère possède un numéro d'identification de 0 à n - 1 ou n est le nombre de sphères de chaque peuple. Voici la liste des valeurs retournées:

-3 s'il n'y a pas de sphère dans la case.

-2 si la case n'est pas dans la carte.

-1 si la case n'est pas visible ou sondable.

n correspondant au numéro de la sphère.

## 12.5 GetCultureAtPos

Java: `public int GetCultureAtPos(int x, int y, int z);`
Caml: `external get_culture_at_pos : int -> int -> int -> int`
C: `extern signed GetCultureAtPos(unsigned char x, unsigned char y, unsigned char z);`
Pascal: `function GetCultureAtPos(x, y, x: ShortInt): LongInt;`

Cette fonction permet d'obtenir le nombre d'étapes estimées avant l'arrivée à maturité du haricot qui pousse dans le cube coton de la case de coordonnées (x, y, z). On ne peut récolter que lorsque le nombre d'étapes est à zéro. Voici la liste des valeurs retournées:

-3 si la case ne contient pas un cube de coton ou s'il n'y a pas de haricot en train de pousser dans la case.

-2 si la case n'est pas dans la carte.

-1 si la case n'est pas visible ou sondable.

n correspondant au nombre d'étapes restantes avant maturité.

## 12.6 GetPos

Java: `public void GetPos(int[] xyz);`
Caml: `external get_pos : unit -> int * int * int`
C: `extern void GetPos(unsigned char * x, unsigned char * y, unsigned char * z);`
Pascal: `Procedure GetPos(var x, y, z: ShortInt);`

Cette fonction fourni la position courante de la sphère dans la carte.

## 12.7 GetNbBean

Java: `public int GetNbBean();`
Caml: `external get_nb_bean : unit -> int`
C: `extern unsigned GetNbBean(void);`
Pascal: `function GetNbBean: LongInt;`

Cette fonction indique à la sphère combien elle possède de haricots.

## 12.8 GetNbSeed

Java: `public int GetNbSeed();`
Caml: `external get_nbr_seed : unit -> int`
C: `extern unsigned GetNbSeed(void);`
Pascal: `function GetNbSeed: LongInt;`

Cette fonction indique à la sphère combien elle possède de graines.

## 12.9 GetStamina

Java: `public int GetStamina();`
Caml: `external get_stamina : unit -> int`
C: `extern signed GetStamina(void);`
Pascal: `function GetStamina: LongInt;`

Cette fonction indique à la sphère combien il lui reste de points d'énergie.

## 12.10 Move

Java: `public void Move(int x, int y, int z, int n);`
Caml: `external move : int -> int -> int -> int -> unit`
C: `extern void Move(unsigned char x, unsigned char y, unsigned char z, unsigned char n);`
Pascal: `procedure Move(x, y, z, n: LongInt);`

Cette fonction indique au serveur que la sphère va tenter de se déplacer dans la case de coordonnées (x, y, z). Durant ce déplacement, la sphère tentera de porter n case(s). Si la destination n'est pas une case adjacente à celle de la sphère ou la case ou se trouve actuellement la sphère, la sphère tentera alors de rester sur place. Cependant, même si le déplacement échoue de cette manière, la sphère porte le nombre de cubes voulus (si possible). Voir la note sur les actions ci-dessous.

## 12.11 Plant

Java: `public void Plant();`
Caml: `external plant : unit -> unit`
C: `extern void Plant(void);`
Pascal: `procedure Plant;`

Cette fonction indique au serveur que la sphère va tenter de planter une graine dans la case juste en-dessous de sa position. Voir la note sur les actions ce-dessous.

## 12.12 Harvest

Java: `public void Harvest();`
Caml: `external harvest : unit -> unit`
C: `extern void Havest(void);`
Pascal: `procedure Harvest;`

Cette fonction indique au serveur que la sphère va tenter de récolter l'éventuel haricot qui se trouve dans la case juste en-dessous de sa position. Voir la note sur les actions ci-dessous.

## 12.13 ListenPublic

Java: `public byte[] ListenPublic();`
Caml: `external listen_public : unit -> char array option`
C: `extern signed ListenPublic(char * data);`
Pascal: `function ListenPublic(data: CString): LongInt;`

Cette fonction permet à la sphère de récupérer le message suivant de la liste publique. Les messages publics correspondent aux messages télépathiques instinctifs des sphères. La fonction prend en paramètre un pointeur sur un tableau de 43 (j'ai bien dis 43) caractères qui sera rempli avec le message s'il en reste. La fonction renvoie 1 s'il restait un message disponible dans la liste, 0 sinon. En

Java et Caml, la fonction renvoie le tableau ou un objet nul.

## 12.14 ListenPrivate

Java: `public signed ListenPrivate();`
Caml: `external listen_private : unit -> char array option`
C: `extern signed ListenPublic(char * data);`
Pascal: `function ListenPublic(data: CString): LongInt;`

Cette fonction agit comme ListenPublic mais avec les messages prives, c'est à dire les messages envoies avec Send (voir ci-dessous).

## 12.15 Send

Java: `public void Send(byte[] data);`
Caml: `external send : char array -> unit`
C: `extern void Send(char * data);`
Pascal: `procedure Send(data: CString);`

Cette fonction indique au serveur que la sphère va tenter d'envoyer un message prive. data pointera vers un tableau de 42 caractères qui contiennent le message. En Java et en Caml, on passera le tableau directement. Voir la note sur les actions ci-dessous.

## 12.16 Success

Java: `public int Success();`
Caml: `external success : unit -> bool`
C: `extern signed Success(void);`
Pascal: `function Success: LongInt;`

Cette fonction permet à la sphère de savoir si l'action qu'elle a effectuée au tour précédant s'est bien déroulée. La fonction renverra 1 en cas de succès et 0 sinon. Une action a été faite avec succès uniquement si elle s'est EXACTEMENT déroulée conne prévu. Par exemple, si une sphère tente de se déplacer en portant 3 cubes, que le déplacement se passe bien, mais qu'elle n'a pu prendre que 2 cubes, Success indiquera que l'action a échoue.

Note à propos des actions:

Une sphère ne peut faire qu'une seule action à un tour donne. Or les appels des fonctions de requête d'action ne sont pas terminaux. Si

une sphère envoie plusieurs ordres au cours du même tours, seul le dernier est pris en compte. D'autre part, si la sphère n'a pas envoyé d'ordre durant le tour, l'ordre de rester sur place sans rien porter est pris par défaut.

## 12.17 GetNbSphere

Java: `public int GetNbSphere();`
Caml: `external get_nb_sphere : unit -> int`
C: `extern signed GetNbSphere(void);`
Pascal: `function GetNbSphere(): LongInt;`

Cette fonction permet à la sphère de savoir de combien de sphère est composé son peuple.

# 13 Les fonctions à écrire

## 13.1 Obtention du nom du client

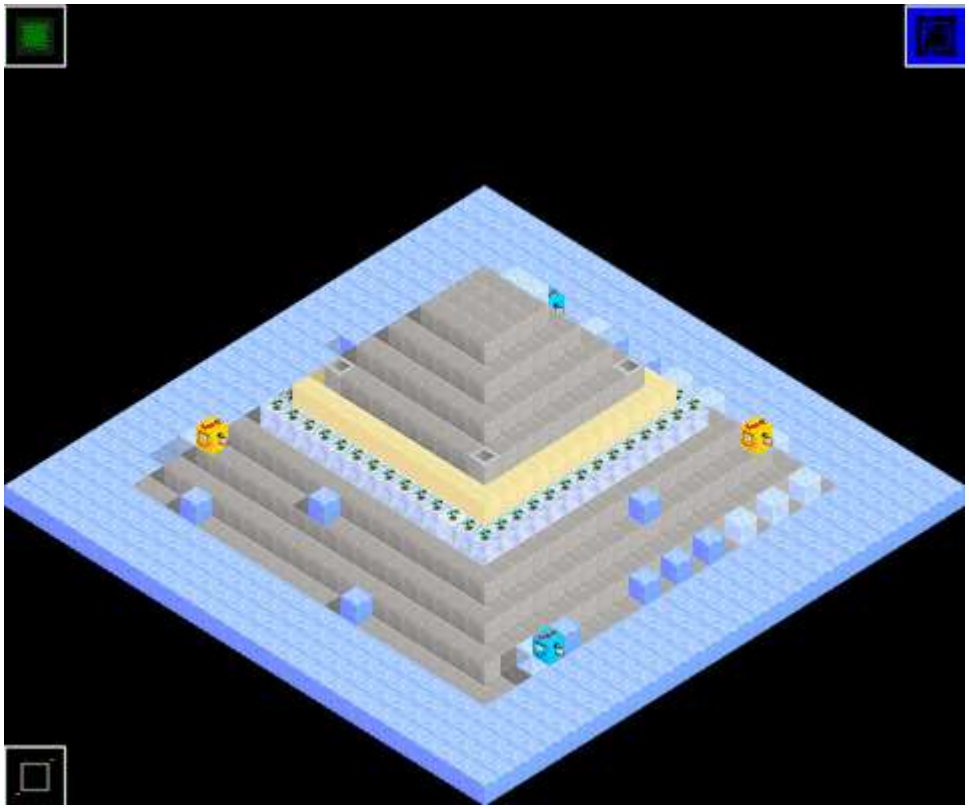
Java: `public String GetClientName();`
Caml: `val get_client_name : unit -> string`
C: `extern const char * GetClientName(void);`
Pascal: `function GetClientName: CString;`

## 13.2 Premier tour de la partie

Java: `public void InitGame();`
Caml: `val init_game : unit -> unit`
C: `extern void InitGame(void);`
Pascal: `procedure InitGame;`

## 13.3 Jouer un tour

Java: `public void PlayTurn(void);`
Caml: `val play_turn : unit -> unit`
C: `extern void PlayTurn(void);`
Pascal: `procedure PlayTurn;`



Bonne chance, tu as 36h42 !

Dernière modification: 1/08/2000