

# Programmation – DM3 : Denki Blocks!<sup>TM\*</sup>

{ Jeremie.Detrey, Emmanuel.Jeandel } @ens-lyon.fr

à rendre pour le 4 janvier 2005 à 23h59'59" (joyeux Noël et bonne année)

Ce DM est à faire en *binôme*.

Pour ce DM, vous allez devoir réaliser un jeu complet, à partir de rien. Votre programme devra être rendu sous la forme d'une archive contenant tous les fichiers sources nécessaires, ainsi qu'un Makefile compilant le tout correctement.

Il vous faudra aussi écrire un rapport (dans tout style de votre choix dont la prononciation est litigieuse) décrivant le fonctionnement de votre programme, les choix de conception que vous aurez rencontrés...

## 1 Denki Blocks!<sup>TM</sup>

### 1.1 Brève description

Denki Blocks!<sup>TM</sup> est un puzzle dans lequel le joueur déplace des *pièces* sur un *plateau* pour les regrouper selon leur couleur.

Le plateau est un carré de 16 cases de côté. Certaines de ces cases peuvent être occupées par des *obstacles* immobiles (représentés en gris clair), et d'autres par des pièces. Les obstacles sont en fait des pièces immobiles d'un certain type.

Chaque pièce (aussi appelée *gumblock*) occupe une case. Il peut y avoir différents types de pièces, représentés par différentes couleurs (rouge, vert, ...). Les pièces ne peuvent sortir du plateau, ni se recouvrir l'une l'autre. Plusieurs pièces de la même couleur posées les unes à côté des autres forment un groupe (*clump*) et les pièces deviennent alors solidaires. Dit autrement, si deux pièces d'une même couleur sont placées côte-à-côte, elles *fusionnent* en s'associant l'une à l'autre, pour ne former qu'un seul bloc.

L'objectif du jeu est de regrouper les pièces selon leur type (même s'il n'est parfois pas nécessaire de regrouper tous les types de pièces).

Dans certains niveaux, on peut gagner des points supplémentaires en réalisant une forme prédéfinie avec les pièces du niveau.

### 1.2 Règles du jeu

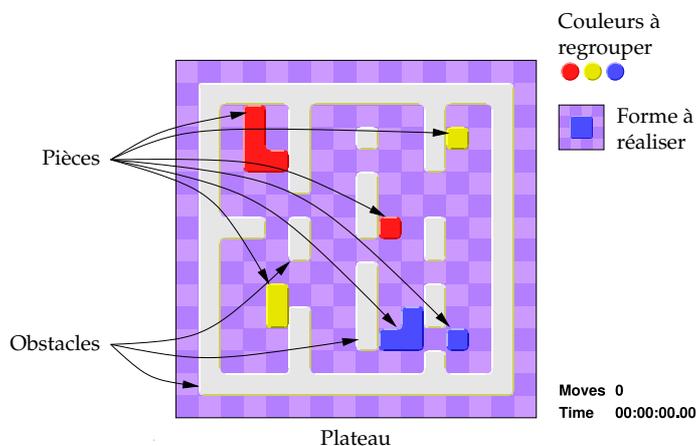
*Remarque* : vos TDmen n'ont pas les moyens de vous offrir un sujet en couleur, donc pour voir correctement les jolis dessins, allez lire le sujet à partir de la page habituelle.

Le jeu commence donc avec la donnée d'une configuration initiale du plateau et des pièces, des types de pièces à regrouper, ainsi que de l'éventuel motif à réaliser.

---

\*Denki Blocks!<sup>TM</sup> is a registered trademark by Denki Ltd. <sup>TM</sup>®© bla bla bla... Donc n'essayez pas de commercialiser votre programme, ou vous allez finir en prison<sup>TM</sup>.

Par exemple :



Le joueur déplace alors les pièces d’une case dans une des quatre directions classiques : haut, bas, gauche ou droite. Toutes les pièces sont déplacées en même temps, mais certaines peuvent être bloquées lors de ce déplacement par le bord du plateau, par un obstacle ou encore par une autre pièce :



Au fur et à mesure des déplacements, des pièces de même type peuvent devenir solidaires lorsqu’elles sont placées l’une à côté de l’autre :



Le puzzle est réussi lorsque toutes les pièces indiquées ont été regroupées par couleur. Le score réalisé dépend du nombre de mouvements effectués et du temps passé pour résoudre le puzzle.

Allez consulter le site officiel du jeu : <http://www.denki.co.uk/pages/denkiBlocks.html> pour plus d’informations. Allez aussi voir la démo pour faire joujou un peu, et vous familiariser avec tout ça.

### 1.3 Des niveaux

Pour pouvoir tester votre programme avec plusieurs niveaux, une archive de niveaux est disponible à l’adresse habituelle. Chaque niveau dans cette archive est représenté par un fichier `.lv1`.

Un fichier `.lv1` contient donc tout d’abord la description de la configuration initiale, sous la forme de 16 lignes de 16 caractères chacune. Chaque caractère code donc le contenu d’une case du plateau, avec la correspondance suivante :

- `.` : case vide ;
- `#` : obstacle ;
- `i` (avec  $i \in \{1, 2, \dots\}$ ) : pièce de type  $i$ .

Suit alors une ligne contenant une liste d’entiers séparés par un espace, représentant les types des pièces qu’il faut regrouper. Enfin, optionnellement, un motif requis avec la sémantique précédente.

Ainsi, le puzzle donné en exemple précédemment sera codé par le fichier suivant :

```

.....
.#####.
.#.1.#....#.#.
.#.1.#..#.#2.#.
.#.11#....#.#.
.#...#.#....#.
.#.....#....#.
.###.#..#1.#..#.
.#...#....#.#.
.#.....#....#.
.#..2...#..#.#.
.#..2#..#.#3#..#.
.#...#..#33.3.#.
.#...#....#.#.
.#####.
.....
1 2 3
33
33
    
```

} Configuration initiale du plateau

} Couleurs des pièces à regrouper

} Motif à réaliser

## 2 Pièces spéciales

Pour l’instant, vous n’avez vu que deux types de pièces : les pièces de couleur et les obstacles.

Rappelons rapidement leurs propriétés :

- Un obstacle (*blocker*) est une pièce immobile.
- Une pièce de couleur (*gumblock*) peut bouger. Lorsqu’elle rencontre une ou plusieurs pièces de la même couleur, elles fusionnent.

Il existe dans Denki Blocks!<sup>TM</sup> des tas d’autres pièces sympas. Ainsi :

- Le boulet (*plod*) : il s’agit d’un type de pièce qui ne peut avancer que si elle est poussée par une autre. Les boulets ne fusionnent pas entre eux.

- Le fourbe (*rogue*) : il s’agit d’un type de pièce qui peut bouger mais qui ne fusionne pas.
- Les blocs peintres (*painters*) : un bloc peintre de couleur bleue peut se déplacer, mais ne fusionne pas. Lorsqu’il rencontre un gumblock d’une autre couleur, les pièces qui le touchent (mais pas tout le groupe!) deviennent bleues.
- Le joker (*joker*) : un joker peut se déplacer et ne fusionne pas avec les autres jokers. Lorsqu’il rentre en contact avec un gumblock, il prend la couleur de ce gumblock (ce qui peut entraîner des effets cascades).
- La clé (*key*) couplée à une serrure (*lock*) : une clé est une pièce qui ne peut avancer que poussée par une autre. La serrure est un type de plancher particulier. Le niveau n’est donné gagnant que si les clé se trouvent dans les serrures. Lorsqu’une clé est dans une serrure, elle devient immobile.
- Le bouton (*switch*) : il s’agit d’un type de plancher particulier. Le niveau n’est donné gagnant que si tous les boutons sont recouverts par des pièces.
- La porte (*gate*) : il s’agit d’un type de plancher particulier coloré qui ne permet le passage que d’une pièce de sa couleur.
- La colle (*sticker*) : il s’agit d’un type de plancher particulier tel que toute pièce qui touche ce plancher devient immobile.
- Et plein (en fait un) d’autres encores ...

### 3 À vous maintenant !

1. Programmez Denki Blocks!™ (sans les pièces et planchers spéciaux).
2. Programmez quelques uns (**mais pas tous !**) des types spéciaux. Vous devrez dans le rapport expliquer à vos TDmen comment ils peuvent eux-mêmes étendre votre programme pour ajouter les autres types de pièces. Vous serez donc notés sur la facilité qu’auront les TDmen à ajouter ces types.

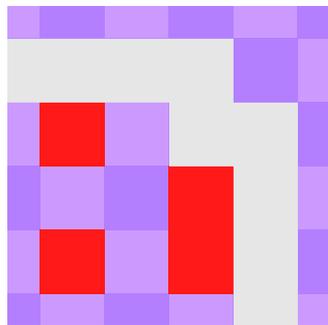
Et puisqu’on a aussi envie de vous mettre une note artistique :

3. Créez trois niveaux pour Denki Blocks!™.

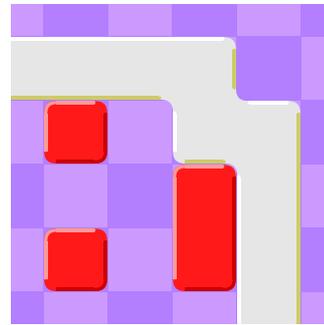
Quelques consignes / directions pour votre programme :

- Langage : OCaml ou C, plus un joli Makefile.
- Portabilité : votre programme doit compiler et tourner sur les machines des salles informatiques.
- Graphismes : ce que vous voulez. Vu que vous êtes maintenant familiers avec Cairo, nous vous conseillons fortement d’utiliser cette bibliothèque. Mais vous pouvez aussi lui préférer X, GTK, SDL, ...

*Remarque* : essayez de faire des graphismes “jolis”. Plutôt que de représenter les pièces comme un tas de gros carrés moches, inspirez-vous des screenshots de ce sujet :



Moche



Joli

#### 4 *Bonus* : Quelques extensions

Voici des idées pour étendre votre jeu :

- annulation du (ou des) dernier(s) mouvement(s) effectué(s) (touche `BackSpace` dans la démo);
- une version multijoueur ;
- un générateur de niveaux aléatoires **résolubles** ;
- bruitages, musique (attention, tout mauvais choix de musique fera perdre des points. .).

#### 5 *Bonus++* : La même chose en PostScript<sup>1</sup>

Après tout, PostScript est un langage de programmation comme un autre, non ? Rappelez-vous du Sokoban de la dernière fois. En plus, les graphismes en PostScript, c’est plutôt facile. Ainsi, si l’envie vous en prend :

4. Amusez-vous à coder Denki Blocks!<sup>TM</sup> en PostScript.
5. Testez-le avec votre interpréteur PostScript du DM2.

#### 6 *Bonus Culture*

6. Écrivez un taquin et expliquez pourquoi vos TDmen vous ont demandé de le faire.

#### 7 *Bonus Master*

7. Montrez que le problème de savoir si un niveau de Denki Blocks!<sup>TM</sup> est résoluble est  $M$ -complet pour une classe de complexité  $M$  bien choisie ( $M = \mathbf{NP}, \mathbf{PSPACE}, \dots$ ).
8. Proposez un algorithme ou une heuristique pour résoudre le jeu.

---

<sup>1</sup>Si si, c’est possible ! Regardez de près les jolis screenshots de ce sujet : au lieu de passer 10h à faire des dessins en XFig, vos dévoués TDmen ont préféré passer 30h à programmer Denki Blocks!<sup>TM</sup> en PostScript. ‘gnifique, non ?