



Merge branch 'master' of framagit.org:mdefalco/projet_mp2i
Marc de Falco authored 1 week ago

b6f290a8

README.md 8.6 KB

Instructions et exemples pour le projet des MP2I du CIV

Description du projet

Le projet « vacances d'hiver » consiste à réaliser un jeu par groupe de deux ou trois élèves.

L'idée est de pratiquer l'informatique sur un programme avec plus de *programmation élémentaire*, pas de révolutionner le monde du jeu vidéo.

Je vous suggère donc de partir sur un projet pas forcément très ambitieux et de vous concentrer sur le gameplay plus que sur les graphismes. L'écueil à éviter c'est de passer tout votre temps à faire un magnifique affichage et de n'avoir aucun temps restant pour le jeu en lui-même. L'autre manière d'éviter de perdre trop de temps est de ne pas faire un jeu *temps réel* mais en tour par tour. Cela permet d'avoir une boucle interactive assez simple : on affiche, on lit l'entrée, on calcule le prochain état du jeu, et on répète. L'avantage étant que tout le monde joue par intervalle de temps discret, et c'est beaucoup plus simple.

Quand on parle de jeu, cela peut-être un jeu de réflexion ou un casse-tête. La dimension *ludique* n'est pas forcément la plus importante.

On essaiera de faire en sorte de dégager des parties algorithmiques intéressantes :

- dans un jeu à deux joueurs cela peut être de faire une IA pour l'adversaire
- s'il y a un système de grille, réaliser des recherches de chemins, de visibilité, ...
- Développer une notion de données procédurales : niveaux, ennemis aléatoires

Des idées de jeux simples :

- Tetris ou autres
- Aventure en mode texte : Zork, ...
- Roguelike en console : Nethack, Dwarf Fortress, ...
- Jeux multi simples comme Bomberman

Installation d'OCaml

On va avoir besoin d'utiliser OCaml avec un environnement permettant de développer, d'installer des programmes et de compiler des exécutables.

- Sous Windows :
 - Installez <https://fdopen.github.io/opam-repository-mingw/installation/> en prenant l'installateur graphique 64 bit (ou 32 bit si vous avez une vieille machine). **Attention** c'est long.
 - Vous avez maintenant un Terminal Cygwin qui vous permet d'accéder à un pseudo système Unix déjà configuré pour faire du OCaml .
 - Tapez dans le terminal `opam install dune graphics`
 - **Attention** vos fichiers sont accessibles dans `C:\OCaml64\home\utilisateur\`
- Sous Linux : il faut installer opam avec votre distribution
- Sous OSX :
 - Installez <https://brew.sh/>
 - une fois que c'est fait vous pouvez installer opam avec `brew install opam`

Récupérer ces fichiers

- Vous pouvez taper `git clone https://framagit.org/mdefalco/projet_mp2i.git`

Compiler des programmes avec dune

On va utiliser dune pour compiler des programmes. C'est un peu une variante évoluée de make adaptée à OCaml . Dans le répertoire principal de votre projet, on va créer un fichier dune qui a en général la structure suivante

```
(executable
  (name monprogramme))
```

```
(libraries liste des bibliotheques utilisées))
```

Ici le point d'entrée du programme, cela peut être le seul fichier, est `monprogramme.ml` dans le même répertoire que le fichier `dune`.

Pour compiler, on se place à la racine du projet et on tape `dune build`.

Ça va créer un fichier `monprogramme.exe` dans le répertoire `_build/default` créé lui aussi par `dune`. Pour le lancer on écrira donc `./_build/default/monprogramme.exe`.

Pistes pour faire des jeux avec OCaml

Mode texte

On peut très bien faire un jeu en mode texte avec des entrées et des sorties purement textuelles. C'est le cas des jeux d'aventures en mode texte. Par exemple, <https://fr.wikipedia.org/wiki/Zork> et les jeux similaires peuvent être intéressants. Surtout avec une dimension lecture des entrées de manière intelligente. C'est en partant de variante de ces jeux que le concept de MOOC a été créé et qu'on a dérivé celui de MMORPG.

Utiliser graphics

Cette bibliothèque est assez primitive mais suffisamment riche pour faire un petit jeu. On trouvera un exemple dans le répertoire `graphics`.

Il faut commencer par l'installer avec `opam install graphics`.

Deux exemples plus fournis dans `civhack` et `pacman`, ce dernier utilisant des graphismes depuis un format d'image `bmp`.

Utiliser la console comme un canevas

Avec la bibliothèque `curses` on peut utiliser la console comme un tableau de pixels. Cette bibliothèque est très riche mais n'est pas forcément très simple d'accès. Vous trouverez dans le répertoire `curses` un exemple avec des indications sur la manière d'obtenir de la documentation.

L'énorme avantage d'utiliser la console, c'est qu'on peut reposer sur l'imagination des joueurs sans avoir à faire de graphismes. Par exemple, `@` pourra représenter l'avatar dans un jeu de rôle ou un jeu d'action. C'est ce qui se passe dans les rogues comme `Nethack` ou `Dwarf Fortress` qui ont inspirés beaucoup de jeux récents alors qu'ils sont très vieux.

Installer curses :

- sur les plateformes autre que windows, `opam install curses` suffit
- sur Windows directement ça a l'air plus compliqué et j'ai du mal à comprendre comment le faire marcher simplement avec cygwin. Mais pour compiler :
 - `opam install depext depext-cygwinports`
 - `opam depext -i curses`
 - en fait la première commande n'est plus nécessaire pour installer d'autres bibliothèques similaires.
 - ensuite, il faut rajouter à la variable d'environnement `PATH` (tapez `env` et windows vous propose le dialogue) le chemin `/usr/x86_64-w64-mingw32/sys-root/mingw/bin` ou similaire que vous pouvez trouver avec `parcourir`. C'est celui qui doit contenir `libncursesw6.dll`
 - on fonctionne alors avec console par défaut de Windows qui a de gros soucis de nombre de couleurs :-/
- avec WSL, c'est-à-dire Linux dans Windows, ça doit bien marcher.

Utiliser la lib sfml

On l'installe avec `opam install sfml` et on va sur internet pour se documenter. Attention, ici c'est une bibliothèque suffisante pour faire un jeu commercialisable, donc il y a sûrement beaucoup de boulot pour l'utiliser, mais le rendu final sera meilleur.

Charger des images avec camlimages

- Installer avec `opam install camlimages`
- Dans le répertoire `images` un exemple où on charge un jpeg et on l'affiche
- A noter** que la boucle `graphics` est assez optimisée et peut servir de base.
- Dans le fichier `dune` il faut rajouter la bibliothèque correspondant aux formats qu'on souhaite ouvrir comme `camlimages.jpeg` et si on veut afficher directement des images avec `graphics` on rajoute `camlimage.graphics`

Rendu des projets

Idéalement, vous travaillez avec `git` et vous rendez les projets sur `github` ou `framagit`. Sinon, un `zip` par mail avant la fin des vacances.

Les projets sont à faire par groupe de 2 ou 3.

Vos fichiers doivent être accompagnés d'un descriptif de ce que vous avez voulu faire, de comment vous l'avez réalisé, etc...

Coder avec git et github

Mise en place

- Allez sur `github.com` et créez un compte
 - si vous êtes le premier de votre groupe créez un repository initialisé avec un `README.md`
 - sinon, passez votre nom de compte à celui qui le gère pour qu'il vous donne les droits d'écriture
- Dans le terminal, on va générer une clé SSH :
 - tapez `ssh-keygen`
 - laissez tout par défaut et pas de pass phrase
- Sur le site de `github`
 - vous cliquez sur le triangle descendant en haut à droite puis `Settings`
 - Ensuite dans `SSH and GPG keys` et `new ssh key`
 - Vous copiez le contenu du fichier `C:\0Caml64\home\utilisateur\.ssh\id_rsa.pub` et vous validez
- Maintenant vous clonez le `repository` en choisissant l'adresse avec `SSH` dans le menu déroulant de `Code` .
 - par exemple vous allez taper `git clone git@github.com:MarcdeFalco/MarcdeFalco.github.io.git` l'adresse doit bien commencer par un `git@github...`

Utilisation minimale

Git fonctionne par lot de modification de fichiers, on appelle ça un `commit` . Ces modifications sont locales. Vous allez donc faire des modifications de vos programmes, quand vous êtes content faire un `commit` avec un message explicatif, envoyer ces modifs sur github ou recevoir les modifs des autres.

- Pour commiter toutes les modifications :
 - `git commit -a -m "un message descriptif"`
- Pour récupérer les dernières modifications :
 - `git pull`
 - **Attention** il ne faut pas que vous ayez de fichiers modifiés mais non commités
- Pour envoyer vos modification :
 - `git push`