université PARIS-SACLAY | FACULTÉ DES SCIENCES D'ORSAY

semaine du 9 janvier 2023

Projet PFA
L3 Informatique

TP n° 1

1 Introduction

Le but de ce TP est :
— de se familiariser avec un projet OCaml complexe
— de réviser des notions simples sur OCaml (types de bases, modules, références)
— d'apprendre à tester un programme OCaml compilé en JavaScript
— d'appréhender la bibliothèque graphique et son mode de fonctionnement

Ce TP est évidemment là pour servir de base à votre Projet.

Vous pouvez faire ce premier TP seul, mais il faudra vous organiser, avant la séance 2, pour être en binômes.

2 Code

L'archive `zip` disponible sur la page du cours contient un squelette de code. L'architecture du code est la suivante :

`dune` : fichier de configuration de l'outil `dune` permettant de compiler le code

`dune-project` : autre fichier de configuration

`index.html` : fichier HTML simple permettant de lancer le jeu dans un navigateur.

`README.md` : fichier contenant quelques informations sur la façon de compiler le projet, ainsi que des pointeurs bibliographiques.

`src/` : répertoire contenant le code source.

`prog/` : le répertoire contenant le fichier `.ml` servant de driver, dont le seul but est d'appeler la fonction `main` écrite en dans `src`

`lib/` : un répertoire (à ne pas modifier) contenant les bibliothèquges `Gfx` (TP1) et `Ecs` (TP2)

La version JavaScript du projet (plus portable) peut être construit simplement avec la commande `dune build`. Le code natif peut être construit avec `dune build @sdl`.

3 Organisation

Le but est de mettre en place l'environnement de travail et de commencer à prendre en main la bibliothèque de rendu graphique. On recommande de travailler de cette façon :
— éditer le code avec VSCode
— ouvrir un terminal à côté avec deux onglets (ou deux terminaux)
— dans un terminal, lancer un serveur Web, par exemple avec `python3 -m http.server`, depuis les sources du projet
— dans un second terminal, taper `dune build` au fur et à mesure que le fichier est modifié. Ce second terminal peut être démarré dans la fenêtre VSCode pour pouvoir pointer facilement les erreurs.
— pointer un navigateur Web vers `http://localhost:80000`. Bien recharger avec CTRL-SHIFT-R entre deux compilations.

4 But du TP

Le but du TP est de faire un petit programme graphique, qui évoluera jusqu'à devenir (au bout de la séance 3) un clone de `Pong`. Il pourra alors être modifié pour en faire un vrai jeu, plus complexe. Le but est d'explorer l'API de la bibliothèque `Gfx` developpée pour ce projet, qui encapsule deux bibliothèques :

— SDL, pour un rendu natif sous Linux (OpenGL), MacOS (OpenGL) et Windows (DirectX)
— JavaScript (l'API Canvas), pour un rendu dans le navigateur

On se concentre dans un premier temps sur le rendu dans le navigateur, plus simple à tester.

1. éditer le fichier `src/game.ml` ajouter du code pour créer une fenêtre de `800x600`, puis y tracer un rectangle noir de `200x200` pixels, aux coordonnées `100, 100`.
2. modifier le code pour créer une fonction

   ```
   draw_rect: Gfx.color -> int -> int -> int -> int -> unit
   ```

   qui dessine un rectangle arbitraire dans la fenêtre, de la couleur donnée
3. modifier le code pour dessiner un rectangle de `200x200` aux coordonnées `100,100` 60 fois par secondes.
4. modifier le code pour que le rectangle alterne de couleur chaque seconde (Rouge, Vert, Bleu, Rouge, Vert, Bleu, ...)
5. modifier le code pour que le rectangle se déplace en pressant sur les flêches du clavier. À quoi faut-il faire attention ?
6. compiler maintenant le code natif : `dune build prog/game_sdl.exe`. Lancer le programme. Que se passe-t'il lorsque l'on presse une touche ?
7. définir un type de donnée `type config = { ... }` comme un enregistrement. Faire en sorte de passer une valeur de ce type à la fonction `Game.run` dans les fichiers `prog/game_js.ml` et `prog/game_sdl.ml`. On pourra se servir de cette valeur pour passer des paramètres de configuration différents.

Basé sur ce squelette, tenter de faire un `Pong`. On constate que l'API de `Gfx` (fonction `main_loop`) nous contraint à utiliser des références. Pour un jeu plus complexe, avoir un ensemble de références globales est peu maintenable. On verra comment solutionner ce problème au TP2.