

# Devoir de programmation PC2R : iSketch

Clara MULLER & Théo LEBOURG

## 1 Côté serveur

Pour développer la partie serveur de iSketch, nous avons opté pour le langage OCaml. Nous allons dans un premier temps expliquer quels ont été nos choix pour organiser notre serveur avant de s'attarder un peu plus en détail sur le code.

### 1.1 Création du serveur

Lorsque l'on lance l'exécutable, la première étape du programme est la création du serveur (pour qu'il soit prêt à recevoir les connexions des futurs clients) ainsi que l'initialisation de variables qui seront utilisées une fois le jeu lancé.

La création de notre objet serveur reprend les étapes classiques vues en cours : on crée une **une prise** (en utilisant le domaine Internet, le flot d'octets et le protocole de communication par défaut) et on récupère **une adresse dans le domaine internet** (à l'aide de l'adresse Internet de la machine locale et d'un numéro de port défini par l'utilisateur) afin de pouvoir associer les deux puis ensuite, on rend la prise capable d'accepter les connexions et enfin, on met en place une boucle infinie qui accepte les demandes de connexions et qui va créer un thread joueur à chaque nouvelle demande (nous verrons plus tard comment nous gérons le nombre limite de joueurs par partie).

En parallèle des demandes de connexions des joueurs, on lance un thread qui va gérer une partie.

- Un tableau de taille le nombre de joueurs maximal pour une partie et dont chaque case contiendra une string avec le *pseudo* du joueur ainsi que son nombre de *points* au cours du jeu sous la forme *pseudo/points/*
- Un tableau contenant l'ensemble des mots présent dans le fichier *dictionnaire* donné par l'utilisateur et dans lequel on tirera au hasard un mot à chaque tour

### 1.2 Création des joueurs

#### 1.2.1 Quand un mot est trouvé par un joueur

Voici le déroulement général des étapes effectuées par le serveur lorsqu'un joueur a trouvé le mot que le dessinateur était en train de dessiner. Notons que puisque les joueurs évoluent chacun sur un thread, nous avons protégé cette série d'opérations par un **mutex**.

1. Le serveur se charge d'envoyer la commande `WORD_FOUND/joueur/` à tous les joueurs de la partie.
2. Le serveur attribue le nombre de points qui va bien au joueur
- 3.

## **1.3 Extensions**

### **1.3.1 Comptes utilisateurs**

Côté serveur, nous avons opté pour le stockage des noms et mots de passe des joueurs dans un simple fichier texte. Cependant, avant d'être stocké sur ce fichier, le mot de passe est salé (dynamiquement pour éviter les attaques par tables arc-en-ciel) puis hashé avec la fonction de hashage MD5 (cette fonction de hachage n'est certes plus fiable depuis longtemps mais nous n'avons pas trouvé dans la librairie standard d'OCaml d'autres fonctions plus performantes telle SHA-256). Ainsi, à chaque inscription, on ajoute une ligne contenant le nom du joueur, le mot de passé hashé et salé et le sel.

## **2 Côté client**