1 Liste des messages et leur encodage

L'encodage des messages est relativement simple et extrêmement sous-optimal. Le but est de rendre les messages facile à analyser.

1.1 Messages à destination du serveur

1.1.1 Enregistrement d'un nouvel auteur

```
{ "register" : public_key }
```

Un auteur enregistré sera attendu avant de passer au tour suivant. Le serveur répond à ce message par un sac de lettres. La public key aura été préalablement générée à l'aide du système de signature `ed25519`. Vous pourrez, pour cela utiliser une fonction de génération de clefs publiques et privées d'une bibliothèque adaptée à votre language, comme par exemple la bibliothèqe `Hacl` pour OCaml.

1.1.2 Écoute continue des messages

A l'aide des deux messages suivants, un client pourra se mettre dans un mode d'écoute continue pour éviter d'avoir demander régulièrement au serveur son état.

```
{ "listen" : null }
```

Après qu'un client aura envoyé ce message au serveur, ce dernier lui retransmettra directement toutes les nouvelles lettres et les nouveaux mots injectés ainsi que les opérations brutes (raw ops). Le client peut arrêter le flux de messages en envoyant :

```
{ "stop_listen" : null }
```

1.1.3 Bassins de lettre

Pour obtenir l'ensemble des lettres injectées depuis le début de la partie, on enverra le message :

```
{ "get_full_letterpool": null}
```

Pour obtenir seulement les lettres injectées depuis une époque précédente, on pourra utiliser

```
{ "get_letterpool_since" : period }
```

1.1.4 Bassins de mots

De même que pour les lettres, on pourra obtenir les mots injectés par le message :

```
{ "get_full_wordpool": null}
```

et les obtenir de manière incrémentale grâce à :

```
{ "get_wordpool_since" : period}
```

1.1.5 Injection de lettres et de mots

Une lettre pourra être injectée par un auteur via un message

```
{ "inject_letter": letter }
```

et un mot sera injecté par un politicien par le message

```
{ "inject_word" : word }
```

1.1.6 Opération brute

Enfin si vos politiciens ou vos auteurs souhaitent s'échanger des messages non-prévus dans le serveur sans implanter une couche paire à paires, ils peuvent envoyer une opération brute, à vous de voir ce que vous faites du buffer.

```
{ "Inject_raw_op" : buf }
```

1.2 Messages envoyés par le serveur

1.2.1 Sac de lettres

Après qu'un client s'est enregistré, le serveur lui envoie un sac de lettre qu'il a le droit d'utiliser.

```
{ "letters_bag" : [ letter... ] }
```

1.2.2 Tour suivant

À tout moment, le serveur peut annoncer qu'il passe au tour suivant en envoyant un message :

```
{ "next_turn" : i }
```

Ce message pourra être reçu en cas d'écoute continue mais aussi en réponse à un message. Dans ce cas, le serveur enverra deux messages succéssivement, le premier annonçant le passage au tour suivant, suivi de la réponse "normale" au message envoyé par le client.

1.2.3 Bassin de lettres et de mots

En réponse aux demandes de bassin lettres et respectivement de mots depuis le début de la partie, le serveur enverra

```
{ "full_letterpool" : letterpool }
```

respectivement

```
{ "full_wordpool" : wordpool }
```

De même pour une demande de bassins partiels, il enverra :

```
{ "diff_letterpool" : {since : period , letterpool } }
```