Équipe 3

Turb0_P41nt_F1v3_Th0u54nd.exe
Protocole de communication

Version 1.2

Historique des révisions

Date	Version	Description	Auteur
2019-09-12	1.0	Première écriture du document.	Gabriel Bottari-Defrance
2019-09-12	1.1	Révision	Justine Lambert
2019-09-26	1.2	Addition de plus d'information à propos des paquets	Gabriel Bottari-Defrance

Table des matières

Protocole de communication

1. Introduction

Le protocole de communication sert à définir les règles de communication de plusieurs entités. Dans ce cas-ci, nous définissons la communication et ses mécanismes entre nos clients et notre serveur de jeu. Premièrement, nous expliquons quels protocoles de communication seront utilisés et dans quel contexte. Ensuite, nous fournissons davantage de détails sur ces protocoles et sur leur manière de fonctionner.

2. Communication client-serveur

La communication entre le client et le serveur utilise deux protocoles bien connus: premièrement HTTP, et ensuite, les WebSockets. Les communications entre le client et le serveur sont nécessaires pour plusieurs opérations telles que la messagerie et le jeu en temps réel.

HTTP, HyperText Transfer Protocol, est un protocole de communication largement utilisé dans le monde de l'Internet. Il s'agit d'un protocole suivant le format requête-réponse, dans le modèle client-serveur. Le client envoie une requête au serveur dans le but d'obtenir ou de sauvegarder des données, et le serveur répond en conséquence.

Dans le cas de l'application développée, plusieurs fonctionnalités utiliseront le protocole HTTP pour sa simplicité d'implémentation et d'utilisation. Premièrement, pour la fonctionnalité de clavardage, il est nécessaire d'aller chercher les informations concernant les différents canaux, les utilisateurs connectés et l'historique des messages d'une conversation. Par historique de messages, on entend ceux qui étaient préexistants avant la connexion de l'utilisateur (et non en temps réel). En ce qui concerne le jeu lui-même, celui-ci doit aller chercher les parties, créer de nouvelles parties, et stocker les scores et les classements.

HTTP a des limitations, principalement le fait que le serveur nécessite une requête pour fournir une réponse; ainsi, la communication du serveur vers le client est impossible. Pour obtenir une expérience en temps réel, nous utiliserons un autre protocole, soit celui des WebSockets. WebSocket est aussi un protocole de communication du *World Wide Web* qui vise à créer des canaux de communications bidirectionnels par-dessus TCP pour les navigateurs Web. Une fois un client connecté au serveur, le canal peut être utilisé par les deux parties prenantes pour échanger de l'information à tout moment.

Pour notre application, certaines fonctionnalités requièrent des actions en temps réel, soit de nouvelles informations provenant du serveur qui effectuent une mise à jour sur le client. Ces fonctionnalités sont entre autres le clavardage, qui implique la réception de nouveaux messages provenant des différents clients en passant par le serveur, et le jeu de dessin lui-même, qui transmet les lignes tracées d'un client à l'autre.

Tous les canaux de communication du serveur sont protégés par une authentification par jetons de type *Bearer*. Les deux protocoles permettent l'envoi d'un tel jeton sur chaque requête/ouverture de connexion.

3. Description des paquets

Le protocole de communication WebSockets est basé sur TCP et donc utilise des trames semblables à celle de ce protocole: un champ de terminaison, des champs réservés, le code d'opération, la longueur du contenu, le masque et le contenu lui-même.

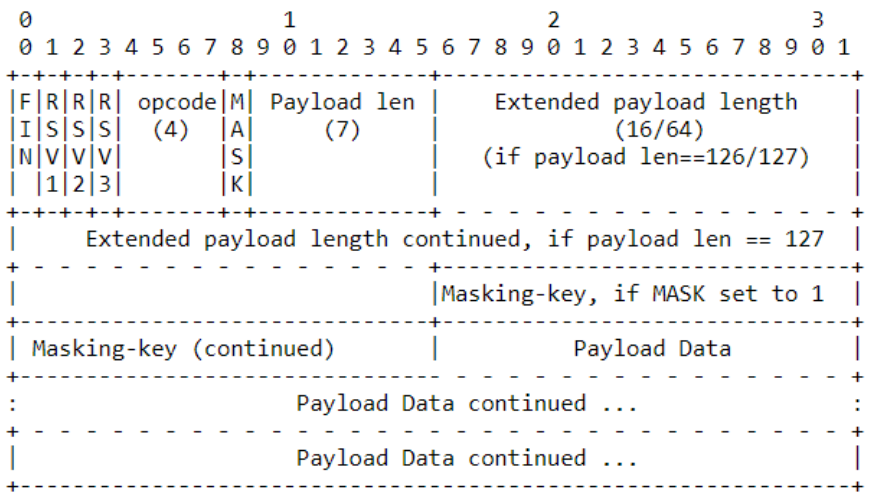

Figure 1. Trame WebSocket

Une fois la connexion établie, à l'aide d'un *handshake*, des trames de données sont envoyées, contenant les différentes informations que l'application souhaite transmettre. La librairie utilisée transforme cette information en binaire/texte et la brise en morceaux, dans le but de l'envoyer sur plusieurs trames.

Dans le cas d'HTTP, aussi basé sur TCP, le message est converti en une suite de trames, commençant par des en-têtes suivies du contenu (information transmise de l'application) en binaire. Plusieurs librairies permettent l'envoi facile d'information en utilisant ce protocole. Habituellement, le format JSON est utilisé, pour être ensuite déconstruit en petites parties pour respecter la taille permise des trames.

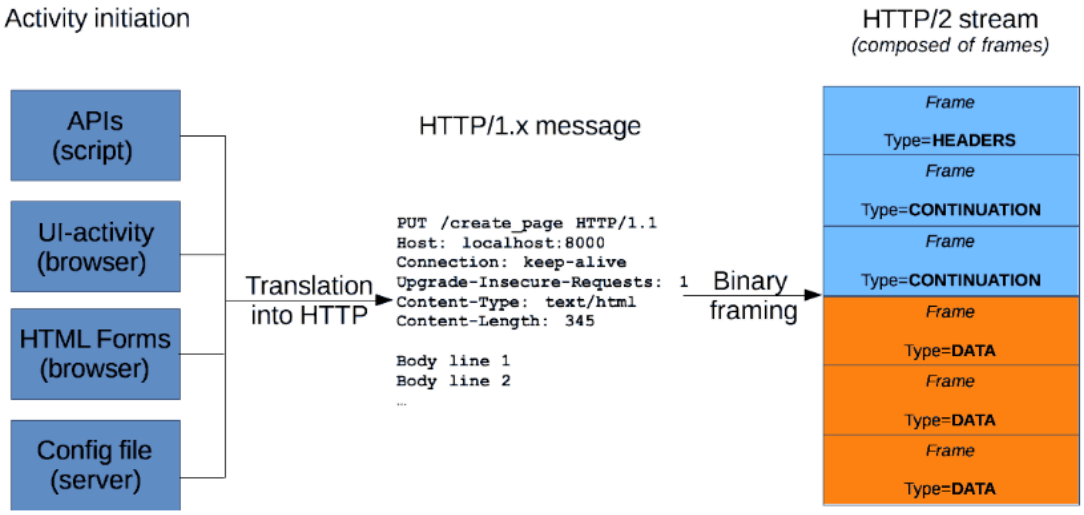

Figure 2. Protocole HTTP

En ce qui concerne les paquets envoyés par l'application vers le serveur et vice versa sur les canaux de communication, une distinction se fait en termes de fonctionnalités.

En ce qui a trait aux WebSockets, ils interviennent lors de toute communication qui nécessite des mises à jour en temps réel.

Premièrement, ceux-ci sont utilisés pour la messagerie. Lors d'une connexion à salle de clavardage, tout nouveau message envoyé se fait transmettre à travers SignalR. Il contient l'identificateur de l'auteur, son pseudonyme, l'identificateur de la salle et le contenu du message. Lors de l'arrivée au serveur, le message est retransmis à tous les clients connectés, en ajoutant une estampille de temps.

Ensuite, pour ce qui est du jeu en soi, lors d'une partie, un joueur trace les lignes directrices de son dessin. À chaque ligne tracée, cet ensemble de données est transmis dans un paquet, partant du client utilisé par l'auteur (lourd ou léger) vers le serveur, et retransmis par la suite à tous les clients concernés. Cet ensemble contient tous les pixels dessinés et leur couleur, dans l'ordre. Pour les autres fonctionnalités du jeu, telles que la tentative de réponse, le client répondant envoie un paquet contenant la réponse donnée sous forme de chaîne de caractères. Le serveur transmet par la suite l'état du jeu à tout client concerné, et ce, à chaque essai de tout joueur. L'état du jeu contient le pointage, au tour de quel joueur nous sommes rendus, et l'estampille de temps du début de partie.

Par la suite, les requêtes HTTP s'occupent d'envoyer le reste des données au serveur.

Pour la messagerie, des paquets provenant du serveur contiennent l'ensemble des salles de clavardage disponibles, l'historique des messages pour chaque salle, ainsi que les usagers connectés à une salle, lorsque demandé par la route appropriée.

Pour le jeu, le serveur retourne des paquets contenant la liste des parties existantes, qui englobent le nom de la partie, le type de partie, les joueurs en attente, etc. Lors d'une demande de création de partie, le client envoie dans un paquet toute information définie par l'utilisateur, soit le nom, le type de jeu, les dessins, les indices, la réponse, etc. Ce paquet est par la suite envoyé au serveur et celui-ci retourne un paquet qui contient l'état de la demande. Si la création se voit réussie, la partie se retrouve alors dans le paquet contenant l'ensemble des parties pour affichage publique.

Références

Fette, I et Melnikov, A. (2011). *The WebSocket Protocol*. Tiré de https://tools.ietf.org/html/rfc6455#section-5.4

Mozilla (2019). *HTTP Messages*. Tiré de https://developer.mozilla.org/en-US/docs/Web/HTTP/Messages