Fais-moi un dessin

Protocole de communication

Version 1.5

Historique des révisions

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Auteur** |
| 2020-01-21 | 1.0 | Première ébauche | Ismael Gbian |
| 2020-01-24 | 1.1 | Ajout des principaux protocoles | Ismael Gbian |
| 2020-01-28 | 1.2 | Ajout des flots de communication | Ismael Gbian |
| 2020-01-28 | 1.3 | Revue du document et vérification du respect des exigences | Ismael Gbian, Ibrahim Choukier, Talet Kayhan |
| 2020-02-07 | 1.4 | Nouvelle revue du document et correction du français. | Roger Kazma |
| 2020-04-13 | 1.5 | Mise à jour de tout le document. | Ismael Gbian |

Table des matières

[1. Introduction 4](#_Toc37703974)

[2. Communication client-serveur 4](#_Toc37703975)

[3. Description des paquets 4](#_Toc37703976)

[3.1. Communication des clients vers le serveur avec les sockets 4](#_Toc37703977)

[**3.1.1.** Système de clavardage 4](#_Toc37703978)

[**3.1.2.** Système d'authentification et de profil 5](#_Toc37703979)

[**3.1.3.** Système de partie 6](#_Toc37703980)

[**3.1.4.** Système de jeu 6](#_Toc37703981)

[**3.1.5.** Système de création de jeu 6](#_Toc37703982)

[3.2. Communication du serveur vers les clients avec les sockets 7](#_Toc37703983)

[**3.2.1.** Système de clavardage 7](#_Toc37703984)

[**3.2.2.** Système d'authentification et de profil 7](#_Toc37703985)

[**3.2.3.** Système de partie 8](#_Toc37703986)

[**3.2.4.** Système de jeu 9](#_Toc37703987)

[**3.2.5.** Système de création de jeu 9](#_Toc37703988)

[4. Détails des objets dans les paquets 9](#_Toc37703989)

[4.1. Avatar 9](#_Toc37703990)

[4.2. Message 9](#_Toc37703991)

[4.3. STATE 10](#_Toc37703992)

[4.4. ProfileDetails 10](#_Toc37703993)

[4.5. PartyData 10](#_Toc37703994)

[4.6. ImageData 11](#_Toc37703995)

[4.7. StylusPoint 11](#_Toc37703996)

[4.8. GameData 11](#_Toc37703997)

[4.9. SelectedDrawer 11](#_Toc37703998)

[5. Flot de communication 11](#_Toc37703999)

Protocole de communication

# 1. Introduction

Ce document présente le protocole de communication entre les clients (lourd/léger) et le serveur. Il s'agit de définir comment chacun d'entre eux doit interagir avec l'autre. La section 2 présentera les technologies qui seront utilisées. S'en suivra une description complète des paquets pour chaque message.

# 2. Communication client-serveur

Les clients et le serveur communiquent principalement via des sockets pour ce qui est du système de clavardage et durant les jeux. Pour ce faire, la librairie *Socket.io* sera utilisée du côté serveur (développé en TypeScript avec NodeJs) et sur le client léger (développé en Java sur la plateforme Android) alors que la librairie *SocketIoClientDotNet* sera utilisée sur le client lourd (développé en C# avec .Net). *Socket.io* et *SocketIoClientDotNet* sont des librairies libres qui facilitent l'utilisation des sockets. Les autres interactions comme le téléchargement des images se feront via une API REST avec des requêtes HTTP. Il est important de noter que le serveur n'effectuera aucune distinction entre les deux clients lors des communications.

Le serveur sera déployé sur la plateforme Heroku et accessible via l'URL ws://pi3-2020.herokuapp.com/ pour les sockets et <https://pi3-2020.herokuapp.com/> pour les requêtes HTTP. Les clients seront quant à eux installés sur un ordinateur et une tablette, pour le client lourd et le client léger respectivement. Des ports libres ainsi qu'une adresse IP étant nécessaires, la connexion utilisera le protocole TCP/IP. De plus, toutes les données transférées entre le client et le serveur seront en format JSON.

# 3. Description des paquets

Les paquets devront suivre le format standard décrit par le protocole TCP/IP. En voici un exemple générique.

|  |  |  |
| --- | --- | --- |
| Description | Type de message | Données |

# 

Note: Tous les messages seront sérialisés. Une désérialisation est donc effectuée sur les clients.

## 3.1. Communication des clients vers le serveur avec les sockets

Il est important de noter que pour la plupart des requêtes présentées ici, le serveur n'aura point besoin de l'identifiant de l'utilisateur puisque cet identifiant sera associé à l'objet "Socket" représentant l'utilisateur sur le serveur.

### **3.1.1.** Système de clavardage

|  |  |  |
| --- | --- | --- |
| Description | Nom de la requête | Données |
| Entrer un canal de discussion | enter-chat-room | {roomId: String} |
| Quitter un canal de discussion | leave-chat-room | {roomId: String} |
| Diffuser un message dans un canal de discussion | send-message | {roomId: String, message: String} |
| Créer un canal de discussion | create-chat-room | {roomId: String} |
| Supprimer un canal de discussion | delete-chat-room | {roomId: String} |
| Obtenir l'historique des messages dans un canal de discussion | chat-room-history | {roomId: String} |
| Obtenir tous les canaux de discussion à l'exception des canaux créés pour des parties | get-all-rooms |  |

## 

### **3.1.2.** Système d'authentification et de profil

|  |  |  |
| --- | --- | --- |
| Description | Nom de la requête | Données |
| Se connecter | login | {nickname: String, password: String} |
| Se déconnecter | logout |  |
| Créer un nouveau compte | create-account | {name: String, surname: String, password: String, nickname: String, avatar: Avatar} |
| Obtenir les informations d'un compte | view-account | {userId?: String} |
| Modifier les informations d'un compte | modify-account | {name: String, surname: String, password: String, nickname: String, avatar: Avatar} |
|  | modify-avatar | {avatar: Avatar} |

## 

### **3.1.3.** Système de partie

Note: La *roomId* est à la fois un identifiant pour une partie et pour l'unique canal de discussion créé au début de cette même partie.

|  |  |  |
| --- | --- | --- |
| Description | Nom de la requête | Données |
| Créer une partie | create-party | {mode: Number} |
| Démarrer une partie | start-game |  |
| Ajouter un joueur virtuel à une partie en attente | add-virtual-player | {partyId: String} |
| Retirer un joueur virtuel d'une partie en attente | remove-virtual-player | {partyId: String, id: String} |
| Rejoindre une partie en attente | join-party | {partyId: String} |
| Quitter une partie en attente | leave-party | {partyId: String} |
| Obtenir la liste de toutes les parties | get-all-parties | {mode: Number} |

### 

### **3.1.4.** Système de jeu

|  |  |  |
| --- | --- | --- |
| Description | Nom de la requête | Données |
| Proposer un choix de réponse | answer | {answer: String} |
| Obtenir un indice | get-clue |  |

## 

### **3.1.5.** Système de création de jeu

|  |  |  |
| --- | --- | --- |
| Description | Nom de la requête | Données |
| Créer un jeu | create-game | {game: GameData} |

## 

## 3.2. Communication du serveur vers les clients avec les sockets

### **3.2.1.** Système de clavardage

|  |  |  |
| --- | --- | --- |
| Description | Nom de la requête | Données |
| Entrer un canal de discussion | enter-chat-room | {state: STATE} |
| Quitter un canal de discussion | leave-chat-room | {state: STATE} |
| Diffusion d'un message dans un canal de discussion | send-message | {timestamp: Number, author: String, message: String} |
| Créer un canal de discussion | create-chat-room | {state: STATE} |
| Supprimer un canal de discussion | delete-chat-room | {state: STATE} |
| Obtenir l'historique des messages dans un canal de discussion | chat-room-history | {messages: Message[]} |
| Obtenir tous les canaux de discussion à l'exception des canaux créés pour des parties | get-all-rooms | [Room] |
| Indique la création d'un nouveau canal de discussion à tous les clients à l'exception de celui qui l'a créé | new-chat-room | {roomId: String} |

## 

## 

### **3.2.2.** Système d'authentification et de profil

Le serveur inclura les informations privées liées à un profil uniquement si la requête est issue d'un socket associé au profil requérant.

|  |  |  |
| --- | --- | --- |
| Description | Nom de la requête | Données |
| Se connecter | login | {state: STATE} |
| Obtenir les informations d'un compte | view-account | {name?: String, surname?: String, nickname: String, stats?: ProfileDetails} |
| Modifier les informations d'un compte | modify-account | {state: STATE} |
| Modifier l'avatar associé à un compte | modify-avatar | {state: STATE} |
| Indique à tous les clients qu'un nouvel utilisateur vient de se connecter | new-user-connected | {userId: String} |
| Indique à tous les clients qu'un nouvel utilisateur vient de se déconnecter | user-disconnected | {userId: String} |

### **3.2.3.** Système de partie

|  |  |  |
| --- | --- | --- |
| Description | Nom de la requête | Données |
| Créer une partie | create-party | {state: STATE, partyId: String} |
| Rejoindre une partie en attente | join-party | {state: STATE, partyId: String} |
| Quitter une partie en attente | leave-party | {state: STATE, partyId: String} |
| Obtenir la liste de toutes les parties | get-all-parties | [PartyData] |
| Indique à tous les clients qu'une partie a été supprimée | party-removed | {partyId: String} |
| Indique à tous les clients qu'une partie a été créée | new-party | {partyId: String} |
| Indique à tous les clients qu'un joueur a rejoint une partie | player-joined | {partyId: String, id: String} |
| Indique à tous les clients qu'un joueur a quitté une partie | player-left | {partyId: String, id: String} |
| Indique à tous les clients qu'une partie a commencé | party-started | {partyId: String} |

### 

### 

### **3.2.4.** Système de jeu

|  |  |  |
| --- | --- | --- |
| Description | Nom de la requête | Données |
| Proposer un choix de réponse | answer | {state: STATE} |
| Démarrer une partie | start-game | GameData |

### 

### **3.2.5.** Système de création de jeu

|  |  |  |
| --- | --- | --- |
| Description | Nom de la requête | Données |
| Créer un jeu | create-game | {state: STATE} |

### 

## 4. Détails des objets dans les paquets

### 4.1. Avatar

Il s'agit d'un string contenant l'image de l'avatar au format Base64.

### 

### 4.2. Message

Il s'agit de l'objet représentant chaque message envoyé par les utilisateurs. Il se forme comme suit (au format JSON) :

{**timestamp**: Timestamp, **author**: String, **message**: String, **roomId**: String, **avatar**: String}

### 

### 4.3. STATE

Il s'agit d'un chiffre indiquant l'état de succès ou d'échec lors d'une requête. Les valeurs sont relatives au contexte de la requête et se présentent comme suit :

|  |  |
| --- | --- |
| Valeurs (Number) | Signification |
| -2 | Impossible de satisfaire la requête (problème interne) |
| -1 | Mauvais format pour la requête (valeurs manquantes ou erronées) |
| 0 | Succès de la requête |
| 1 | Le pseudonyme choisi est déjà utilisé |
| 2 | Un utilisateur est déjà connecté avec ce pseudonyme |
| 3 | Le mot de passe ne correspond pas au pseudonyme |

### 

### 4.4. ProfileDetails

Il s'agit de l'objet contenant les détails du profil de l'utilisateur.

Il se forme comme suit :

{

**activity**: [{**connectionDate**: Timestamp, **disconnectionDate**: Timestamp}],

**previousParties**:

[{

**date**: Timestamp,

**type**: Number,

**players**: [String],

**won**: Boolean,

**duration**: Number,

**score**: Number

}]

}

### 4.5. PartyData

Il s'agit du nom (en String) de la plateforme externe sur laquelle l'utilisateur souhaite effectuer un partage. Les valeurs possibles sont comme suit :

{

**id**: String,

**name**: String,

**players**:[String],

**mode**: Number,

**playersCount**: Number,

**playerCapacity**: Number,

**started**: Boolean;

}

### 4.6. ImageData

Il s'agit de l'objet contenant les détails concernant une image de jeu.

{

**paths**:

[{

**points**: Point[{**x**: Number, **y**: Number}],

**hexColor**: String,

**stylusPoint**: StylusPoint,

**strokeWidth**: Number,

}]

}

Note : L'attribut **hexColor** contient la couleur au format hexadécimal.

### 4.7. StylusPoint

Il s'agit simplement d'un string pouvant prendre les valeurs suivantes:

|  |  |
| --- | --- |
| Valeur (String) | Signification |
| Ellipse | Pointe ronde pour le crayon |
| Rectangle | Pointe carrée pour le crayon |

### 

### 4.8. GameData

Il s'agit de l'objet contenant les détails concernant un jeu.

{

**image**: **ImageData**,

**clues**: [String],

**secretWord**: String,

**difficulty**: Number,

**drawingMode**: Number,

**drawingDirection**: Number,

**selectedDrawer**: SelectedDrawer

}

### 4.9. SelectedDrawer

Cette valeur indique quel client doit dessiner l'image.

|  |  |
| --- | --- |
| Valeur (String) | Signification |
| 0 | Le client actuel. Le joueur doit donc être autorisé à dessiner. |
| 1 | Un autre client. |
| 2 | Un joueur virtuel. Les animations doivent donc être utilisées pour afficher l'image. |

### 

## 5. Flot de communication

Cette partie présente le flot des interactions. Il s'agit des routes sur lesquelles le serveur répondra pour chaque requête.

Note : Pour les routes non précisées ici, le serveur répondra sur la route utilisée pour la requête.

|  |  |  |
| --- | --- | --- |
| Route de la requête | Route de la réponse | Commentaire |
| create-account | login | L'utilisateur est immédiatement connecté si le compte est créé avec succès |
| add-virtual-player | player-joined | Tous les clients (incluant celui qui a effectué la requête) sont informés de l'arrivée du joueur virtuel. Ceci reste également valide pour les joueurs non virtuels. |
| remove-virtual-player | player-left | Tous les clients (incluant celui qui a effectué la requête) sont informés du départ du joueur virtuel. Ceci reste également valide pour les joueurs non virtuels. |
| get-clue | send-message | L'indice est envoyé dans le canal de communication de la partie. |
| - | party-removed | Tous les clients sont automatiquement informés de la suppression d'une partie. |
| - | new-party | Tous les clients sont automatiquement informés de la création d'une partie |
| - | party-started | Tous les clients sont automatiquement informés du démarrage d'une partie |