Fais-moi un dessin

Protocole de communication

Version 2.0

Historique des révisions

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Auteur** |
| 2020-02-03 | 1.0 | Rédaction de la première ébauche | Maxime Bernier |
| 2020-02-04 | 1.1 | Rédaction des paquets REST API | Maxime Bernier |
| 2020-02-04 | 1.2 | Rédaction des collections de la base de données | Maxime Bernier |
| 2020-02-07 | 1.3 | Mise en page | Maxime Bernier |
| 2020-02-07 | 1.4 | Révision | Geneviève Laroche |
| 2020-03-02 | 1.5 | Mise à jour de l’authentification | Maxime Bernier |
| 2020-04-13 | 2.0 | Modifications apportées après la correction de l’appel d’offres | Maxime Bernier |

Table des matières

[**1. Introduction**](#_heading=h.30j0zll) **5**

[**2. Communication client-serveur**](#_heading=h.1fob9te) **5**

[**3. Description des paquets pour les sockets**](#_heading=h.3znysh7) **7**

[3.1 Clavardage](#_heading=h.x6nfj2jw6hqi) 7

[3.1.1 Vers le serveur](#_heading=h.c1ty5ngjfa7j) 7

[3.1.1.1 Channel](#_heading=h.hm5gcqy2y3xg) 7

[3.1.1.2 Message](#_heading=h.ft8e5mxrbkq) 7

[3.1.2 Vers le client](#_heading=h.9m35rh9529t3) 8

[3.1.2.1 Channel](#_heading=h.lsjqjthhntwg) 8

[3.1.2.2 Message](#_heading=h.8oog4wijkyhf) 8

[3.2 Déroulement d’une partie](#_heading=h.453e2p5athn5) 8

[3.2.1 Bidirectionnel](#_heading=h.bu242w8rkkyx) 8

[3.2.1.1 Stroke](#_heading=h.azn0kvk24vko) 8

[3.2.1.2 Transmission des informations concernant un tour de jeu](#_heading=h.bm3db38fr2rt) 8

[3.2.2 Vers le serveur](#_heading=h.xn5lnlvxq97p) 8

[3.2.2.1 Réponse](#_heading=h.cgi3v2gies6p) 8

[3.2.3 Vers le client](#_heading=h.wbryysvk8lsc) 9

[3.2.3.1 Réponse](#_heading=h.6fv6ujm6vva3) 9

[3.3 Notifications](#_heading=h.bvhuszaa0vsq) 9

[3.3.1 Vers le client](#_heading=h.8rwe4ibelf5u) 9

[3.3.1.1 Notification](#_heading=h.bhnsq93j4mh) 9

[**4. Description des paquets pour les requêtes respectant le REST API**](#_heading=h.z0267zx7vr41) **9**

[4.1 Authentification](#_heading=h.wu7dulzeokke) 9

[4.1.1 LogIn](#_heading=h.l79ouy63v0c3) 9

[4.1.2 SignIn](#_heading=h.2ikmur25np0) 10

[4.2 Jeu](#_heading=h.w7aqdagriehg) 10

[4.2.1 Créer le jeu](#_heading=h.y3say29jzq0w) 10

[4.3 Partie](#_heading=h.ltd52gz43dh6) 11

[4.3.1 Créer une partie](#_heading=h.jmjzokw532qm) 11

[4.3.2 Supprimer une partie](#_heading=h.51ugz8iq0myt) 11

[4.4 Notifications](#_heading=h.5blzdfimpj9w) 12

[4.4.1 Obtenir toutes les notifications associées à un utilisateur](#_heading=h.q20ycxt8eza) 12

[4.4.2 Supprimer une notification après la réponse](#_heading=h.7mtcm25d5m5y) 12

[4.4.3 Supprimer un challenge refusé](#_heading=h.4wf58exljtxd) 12

[4.4.4 Supprimer une demande d’amitié après la réponse](#_heading=h.8954lqbbrfat) 13

[4.4.5 Accepter un défi](#_heading=h.lmiwawak76g7) 13

[4.4.6 Accepter une demande d’amitié (2 requêtes)](#_heading=h.8rnwjas5z0ja) 13

[4.5 Statistiques](#_heading=h.gsvthtbaykwi) 14

[4.5.1 Après une partie](#_heading=h.97is51upsqaa) 14

[4.5.2 Étape additionnelle après un challenge](#_heading=h.v68qvp78pko9) 14

[4.6 Liste d’amis](#_heading=h.dj4w2wf0y5i9) 15

[4.6.1 Récupérer tous les amis](#_heading=h.w2y890ru4l9y) 15

[**5. Documents de la base de données MongoDB**](#_heading=h.7e7gxxrrkji) **15**

[5.1 ConnectionHistory](#_heading=h.6a09jtc20ylo) 15

[5.2 MatchResult](#_heading=h.etm4lo9fcsow) 16

[5.3 GeneralStatistics](#_heading=h.qerd5311iq9q) 16

[5.4 ChallengeMatch](#_heading=h.wmuwqpxz4xpb) 16

[5.5 FriendshipRequest](#_heading=h.1asjtqrgs15q) 16

[5.6 Friends](#_heading=h.tlr8ca7qx3x5) 17

[5.7 Game](#_heading=h.cqi9zgjt28g) 17

[5.8 Drawing](#_heading=h.7y59ia9nt5l2) 17

[5.9 Strokes](#_heading=h.su4v19g1csg7) 17

[5.10 Message](#_heading=h.yq8f8xqr3jpq) 18

[5.11 Player](#_heading=h.tlpjcp417f9b) 18

Protocole de communication

# 1. Introduction

Ce document sert à décrire les communications entre le serveur et les clients. Dans la seconde section, les protocoles de communications choisis seront présentés. Dans la troisième section, les paquets utilisés lors des communications seront détaillés.

# 2. Communication client-serveur

Deux types de protocoles seront utilisés: socketIo et REST API pour notre application.

Le premier protocole de communication utilisé sera SocketIo. Au niveau du serveur, la libraire *Socket.Io* sera utilisé. Au niveau du client lourd, la librairie *SocketIoClientDotNet* sera utilisé. Bien que cette librairie soit obsolète, son utilisation ne sera pas problématique puisque seules les fonctionnalités de base seront utilisées. Au niveau du client léger, la librairie *socket.io-java* sera utilisé.

Socket.Io permet la communication bi-directionnelle entre le serveur et le client. Il établit une connexion de type WebSockets avec une alternative de type *long-polling connection using xhr-polling* si les WebSockets ne sont pas disponibles. La communication des sockets se fait en fonction du protocole TCP/IP.

Les fonctionnalités utilisant ce protocole sont:

* Clavardage
* Répondre au cours d’un jeu
* Transmission des données nécessaires pour dessiner
* Transmission des informations concernant la partie

La communication bidirectionnelle est essentielle au déroulement sans accroc d’un jeu. Cela permet une meilleure synchronisation entre les différents joueurs. Pour le clavardage, la capacité de filtrer les clients qui recevront le message est l’élément déterminant. De plus, le serveur doit simuler la conversation des joueurs virtuels. Il lui revient donc de transmettre des messages sans que les clients ne les demandent.

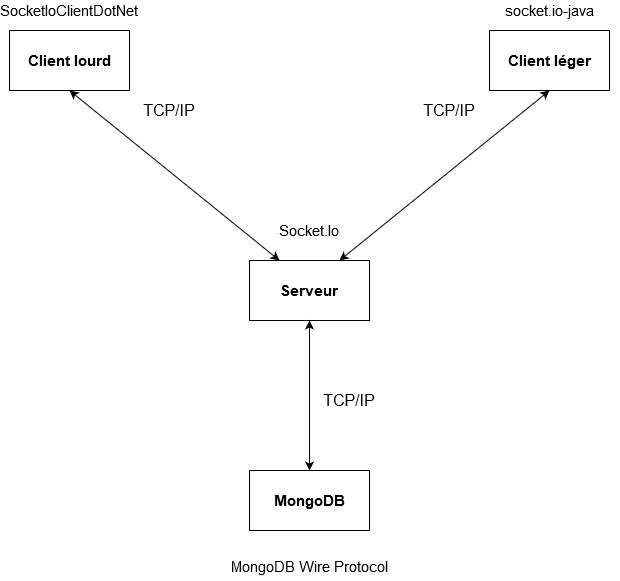


Figure 1: Communication bidirectionnelle des sockets

Le deuxième protocole est le REST API. Celui-ci consiste à associer des adresses url à une méthode spécifique et à des données spécifiques. Cela repose sur la communication TCP/IP entre le client et le serveur. Ce protocole sera utilisé pour les fonctionnalités suivantes:

* Création d’un jeu
* Authentification de l’utilisateur
* Créer/Joindre une partie
* Naviguer à travers les menus (leaderboard, type de parties, options, tutoriel, statistiques)

Ce protocole est utilisé pour les fonctionnalités qui ne nécessitent pas la communication entre plusieurs joueurs. Cela facilite l’ordonnancement des séquences des actions et de leurs réponses. De plus, il est possible de contrôler l’accès à l’application avec ce protocole. En effet, la technologie Json Web Token (JWT) permet d'authentifier l’utilisateur ainsi que de lui associer différentes caractéristiques. Cela permet donc d’éviter que des utilisateurs non inscrits accèdent l’application en entrant une adresse pour contourner l’application.

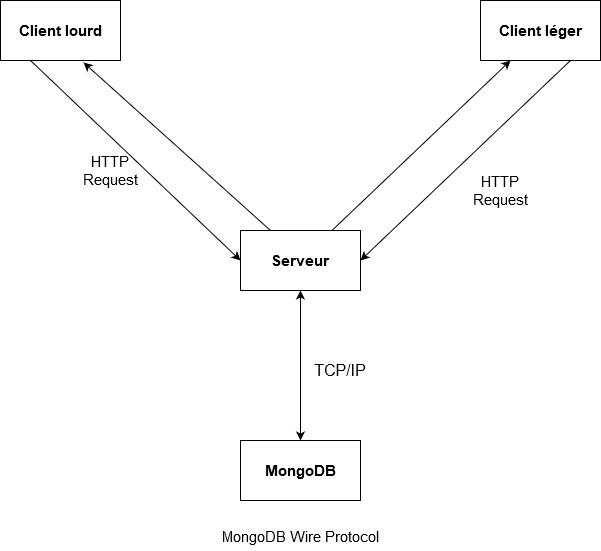


Figure 2: Communication client-serveur pour la communication REST API

Dans les deux protocoles, la communication des messages se fera à travers des JSON. Ce format est un standard qui permet d’associer des valeurs à une clé. Voici un exemple de JSON : {“clé 1”: “texte”, “clé 2”: [entier, entier], “clé 3”: { “clé 3.1”: “string”}}. L’utilisation des protocoles SocketIo et REST API permet d’avoir une application répondant aux besoins des utilisateurs.

.

# 3. Description des paquets pour les sockets

## 3.1 Clavardage

### 3.1.1 Vers le serveur

#### 3.1.1.1 Channel

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Author | String | Auteur du message. |
| Channel | String | Canal de discussion concerné par l’événement |
| Event | String   * CreateChannel * JoinChannel * QuitChannel | Nature de l’opération concernant les canaux de discussion |

#### 3.1.1.2 Message

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Author | String | Auteur du message. |
| Channel | String | Canal de discussion d’où provient le message. |
| Message | String | Message envoyé. |

### 3.1.2 Vers le client

#### 3.1.2.1 Channel

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Confirmation | Bool | Confirmation que l’action sur le canal de discussion a fonctionné |
| Reason | String | En cas d’échec, la cause de cet échec. |

#### 3.1.2.2 Message

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Message | Message | Message envoyé |

## 3.2 Déroulement d’une partie

### 3.2.1 Bidirectionnel

#### 3.2.1.1 Stroke

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Stroke | Stroke | Trait dessiné par le dessinateur |

#### 3.2.1.2 Transmission des informations concernant un tour de jeu

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Drawer | String | Joueur qui doit dessiner |
| Game | Game | Mot/Expression à dessiner avec le temps alloué, la difficulté et les indices |

### 3.2.2 Vers le serveur

#### 3.2.2.1 Réponse

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Author | String | Auteur de la réponse. |
| Answer | String | Réponse du joueur |

### 3.2.3 Vers le client

#### 3.2.3.1 Réponse

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Author | String | Auteur de la réponse tentée |
| Validation | Bool | Indique si la réponse est bonne |
| ValidAnswer | Int | Nombre de joueurs ayant trouvé la solution |
| Score | Int | Score associé à la bonne réponse |
| ElapseTime | int | Temps écoulé en seconde |

## 3.3 Notifications

### 3.3.1 Vers le client

#### 3.3.1.1 Notification

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Destinataire | String | Le nom du joueur |
| Type | Enum   * Friend * Challenge | La nature de la notification |

# 4. Description des paquets pour les requêtes respectant le REST API

Pour les url suivantes, elles sont toutes relatives.

## 4.1 Players

### 4.1.1 Authentification

#### 4.1.1.1 LogIn

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/players/login |  |
| Requête | POST |  |
| Request body | | |
| username | String | Nom unique de l’utilisateur |
| password | String | Mot de passe encodé |
| Valid response | | |
| Status | 200 | Authentification acceptée |
| Avatar | String | Image en Base64 |
| Invalid response | | |
| Status | 401 | Connection non autorisé |

#### 4.1.1.2 SignIn

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/players |  |
| Requête | POST |  |
| Request body | | |
| Player | | |
| Username | String | Nom unique de l’utilisateur |
| Password | String | Mot de passe encodé |
| FirstName | String | Prénom du joueur |
| LastName | String | Nom du joueur |
| Avatar | String | Image en Base64 optionnelle du joueur |
| Valid response | | |
| Status | 201 | Utilisateur créé |
| Invalid response | | |
| Status | 409 | Conflit: Le nom d’utilisateur est déjà utilisé |

#### 4.1.1.3 Logout

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/players/:username/logout |  |
| Requête | PATCH |  |
| Request header | | |
| Authorization | String | Bearer token de l’utilisateur |
| Username | String | Nom de l’utilisateur qui veut se déconnecter |
| Valid response | | |
| Status | 200 | Utilisateur a été déconnecté. |
| Invalid response | | |
| Status | 401 | Le token a été refusé. |

### 4.1.2 Statistiques

#### 4.1.2.1 Après une partie

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/players/stats |  |
| Requête | PATCH |  |
| Request body | | |
| matchResult | MatchResult | Les résultats de la partie |
| Valid response | | |
| Status | 200 | Ok |

### 4.1.2.2 Étape additionnelle après un challenge

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/challenge/:id |  |
| Requête | PATCH |  |
| Request body | | |
| Score | Score | Le résultat obtenu par le joueur relevant le défi |
| Win | Bool | Indique si le score a été battu |
| Valid response | | |
| Status | 200 | Ok |
| Invalid response | | |
| Status | 404 | Aucun défi trouvé |

#### 4.1.2.3 Obtenir les statistiques générales d’un joueur

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/players/:username/general-statistics |  |
| Requête | GET |  |
| Request header | | |
| Authorization | String | Bearer token pour donner accès à la page |
| Username | String | Nom d’utilisateur dont les statistiques nous intéressent |
| Valid response | | |
| Status | 200 | Ok |
| GeneralStats | GeneralStatistics | Le nom d’utilisateurs avec le nombre de parties jouées et gagnées. |
| Invalid response | | |
| Status | 400 | Aucun utilisateur correspondant n’a été trouvé. |

### 4.1.3 Profil

#### 4.1.3.1 Changer d’avatar

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/players/:username/avatar |  |
| Requête | PATCH |  |
| Request header | | |
| Authorization | String | Bearer token pour donner accès à la page |
| Username | String | Nom d’utilisateur dont les statistiques nous intéressent |
| Request body | | |
| Avatar | String | Nouvel avatar en Base64 |
| Valid response | | |
| Status | 200 | Ok |
| Invalid response | | |
| Status | 401 | Accès refusé à la ressource. |

## 4.2 Jeu

### 4.2.1 Créer le jeu

|  |  |  |
| --- | --- | --- |
| Attribut | Valeur | Description |
| URL | ~/game/:id |  |
| Requête | POST |  |
| Request body | | |
| Game | Game | Le jeu à enregistrer |
| Valid response | | |
| Status | 201 | Jeu créé |
| Invalid response | | |
| Status | 409 | Erreur lorsque le serveur empêche d’écraser un jeu existant |

## 4.3 Partie

### 4.3.1 Créer une partie

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/match/:mode |  |
| Requête | POST |  |
| Request body | | |
| VirtualPlayer1 | Personnalité | Personnalité du joueur virtuel 1 |
| VirtualPlayer2 | Personnalité | Personnalité du joueur virtuel 2 |
| AllowedTime | Int | Temps alloué à chaque jeu pour le mode free-for-all |
| Difficulty | Enum   * Facile * Intermédiaire * Difficile * Aléatoire | Indicateur du niveau de difficultés des jeux choisis pour la partie pour le mode solo |
| Valid response | | |
| Status | 201 | Partie créée |
| Invalid response | | |
| Status | 409 | Erreur lorsque le serveur empêche d’écraser une partie existante |

### 4.3.2 Supprimer une partie

|  |  |  |
| --- | --- | --- |
| Attribut | Valeur | Description |
| URL | ~/match/:mode/:id |  |
| Requête | DELETE |  |
| Request body | | |
| Valid response | | |
| Status | 200 | Partie supprimée |
| Invalid response | | |
| Status | 404 | La partie n’a pas été trouvée |

## 4.4 Notifications

### 4.4.1 Obtenir toutes les notifications associées à un utilisateur

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/users/:username/notifications |  |
| Requête | GET |  |
| Request body | | |
| Valid response | | |
| Status | 200 | Ok |
| Body | Collection de notifications | Toutes les notifications de l’usager. |
| Invalid response | | |
| Status | 404 | Aucune notification trouvée |

### 4.4.2 Supprimer une notification après la réponse

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/users/:username/notifications/:id |  |
| Requête | DELETE |  |
| Request body | | |
| Valid response | | |
| Status | 200 | Notification supprimée |
| Invalid response | | |
| Status | 404 | Notification non trouvée sur la base de données |

### 

### 4.4.3 Supprimer un challenge refusé

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/challenge/:id |  |
| Requête | DELETE |  |
| Request body | | |
| Valid response | | |
| Status | 200 | Défi supprimé |
| Invalid response | | |
| Status | 404 | Défi non trouvé sur la base de données |

### 

### 4.4.4 Supprimer une demande d’amitié après la réponse

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/notifications/friend-request/:id |  |
| Requête | DELETE |  |
| Request body | | |
| Valid response | | |
| Status | 200 | Demande d’amitié supprimée |
| Invalid response | | |
| Status | 404 | Demande d’amitié non trouvée sur la base de données |

### 

### 4.4.5 Accepter un défi

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/challenge/:id |  |
| Requête | GET |  |
| Request body | | |
| Valid response | | |
| Status | 200 | Défi obtenu avec succès |
| Body | ChallengeMatch | Toute l’information concernant le défi |
| Invalid response | | |
| Status | 403 | Refuse l’accès si le joueur ne fait pas partie du défi |

### 4.4.6 Accepter une demande d’amitié (2 requêtes)

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/user/:username/friends/:friend-username |  |
| Requête | PATCH |  |
| Request body | | |
| Score | Score | Le résultat obtenu par le joueur relevant le défi |
| Win | Bool | Indique si le score a été battu |
| Valid response | | |
| Status | 200 | Demande d’amitié enregistrée |
| Invalid response | | |
| Status | 404 | Aucune demande d’amitié n’a été trouvée |

## 4.5 Statistiques

## 4.6 Liste d’amis

### 4.6.1 Récupérer tous les amis

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| URL | ~/users/:username/friends |  |
| Requête | GET |  |
| Request body | | |
| Valid response | | |
| Status | 200 | Défi obtenu avec succès |
| Body | Liste de string | Liste des noms des joueurs |
| Invalid response | | |
| Status | 401 | Accès non autorisé à une liste d’ami d’un autre joueur |

# 5. Documents de la base de données MongoDB

## 5.1 ConnectionHistory

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Timestamp | Timestamp | Moment que le joueur se connecte à l’application |
| Statut | ConnectionStatus   * CONNECTED * DISCONNECTED | Statut de connection du joueur |
| Username | String | Nom du joueur qui se connecte |

## 5.2 MatchResult

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Timestamp | Timestamp | Moment de début de la partie |
| MatchMode | MatchMode   * Solo * Free-for-all * Challenge | Mode de la partie |
| Players | Collection des joueurs dans la partie | Tous les joueurs dans la partie |
| Winner | String | Gagnant de la partie |
| Duration | Int | Durée de la partie en seconde |

## 5.3 GeneralStatistics

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Username | String | Nom du joueur |
| MatchPlayed | Int | Nombre de parties jouées |
| MatchWon | Int | Nombre de parties gagnées |

## 5.4 ChallengeMatch

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| AllowedTime | Int | Temps total alloué en seconde |
| ChallengerScore | Int | Score à battre |
| OpponentScore | Int | Score de la personne qui relève le défi |
| Games | Collection de string | Liste de tous les identifiants des jeux constituant le challenge |
| Challenger | String | Nom du joueur qui lance le défi |
| Opponent | String | Nom du joueur qui relève le défi |
| Winner | String | Nom du joueur qui gagne le défi |

## 5.5 FriendshipRequest

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Id | String | Identifiant de la requête |
| Source | String | Nom du joueur qui a fait la demande |
| Destination | String | Nom du joueur qui reçoit la demande |
| Message | String | Message accompagnant la demande d’amitié |

## 5.6 Friends

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Username | String | Nom du joueur |
| Friends | Liste de string | Liste des amis du joueur défini avec l’attribut username |

## 5.7 Game

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Id | String | Identifiant du jeu |
| Expression | String | Nom du joueur |
| Clues | Collection de string | Les indices associés au dessin |
| AllowedTime | Int | Temps alloué en seconde |
| Drawing | Drawing | L’image à dessiner |

## 5.8 Drawing

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Id | String | Identifiant du jeu auquel appartient l’image |
| Strokes | Vecteur de Strokes | Vecteur de tous les traits constituant l’image |

## 5.9 Strokes

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| GameId | String | Identifiant du jeu auquel appartient l’image |
| Sequence | Int | Position dans la séquence des traits |
| x0 | Int (pixels) | Position initiale en x |
| y0 | Int (pixels) | Position initiale en y |
| x1 | Int (pixels) | Position finale en x |
| y1 | Int (pixels) | Position finale en y |
| stroke | rgb(int,int,int) | Couleur du trait |

## 5.10 Message

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Author | String | Auteur du message. |
| Channel | String | Canal de discussion d’où provient le message. |
| Timestamp | Timestamp | Moment que le serveur envoie le message. |
| Body | String | Message envoyé |

## 5.11 Player

|  |  |  |
| --- | --- | --- |
| **Attribut** | **Valeur** | **Description** |
| Username | String | Nom unique de l’utilisateur |
| Password | String | Mot de passe encodé |
| FirstName | String | Prénom du joueur |
| LastName | String | Nom du joueur |
| Avatar | Image | Image du joueur visible à tous |