Appli Web :
DecubX

Kévin CARENOU Sacha VANLEENE Thibault MEUNIER
Matthieu Perrier

15 Janvier 2017

Présentation de notre application

Notre projet s'intitule Décubx, à l'image d'un Agar.io c'est un jeu qui se joue sur un navigateur web.Le concept est assez simple, c'est un endless game basé sur la mémoire du joueur. Effectivement 4 cases sont affichées à l'écran, une séquence est jouée et l'utilisateur dispose d'un temps limité pour la reproduire. Si il réussi le serveur génére la suite de la séquence, la renvoie au joueur et ainsi de suite jusqu'à échec du joueur. Une fois la partie finis le score de la partie est enregistré afin que l'utilisateur puisse accéder à son historique de partie. Tout cela implique évidemment la gestion de comptes, utilisateurs , de base de données, nous préciserons cela plus tard... Image Jeu

Sommaire

1 Architecture **2**
1.1 Architecture client . . . 3
1.1.1 Communication avec le serveur . . . 3
1.1.2 Fonctionnement . . . 3
1.2 Architecture serveur . . . 4
1.2.1 Authentification . . . 4
1.2.2 WebSocket . . . 4
1.2.3 Liens Ldab/Base de données . . . 4

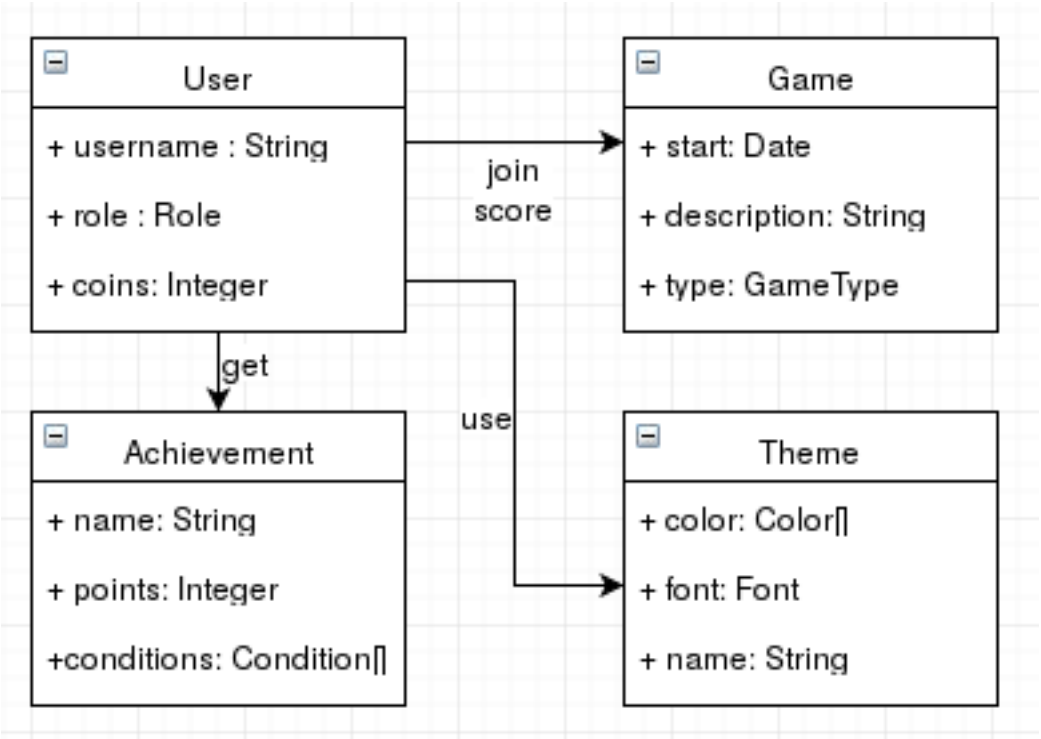
2 Liens avec des services externes **5**

3 Répartition du travail **5**
3.0.4 Utilisation de GitLab . . . 5

4 Annexes **6**
4.0.5 Exemple de GitLab . . . 6

1 Architecture

Dans cette partie nous développerons les différents points techniques sur lesquels nous avons travaillé.



1.1 Architecture client

Nous avons fait le choix de ne disposer que d'une seul et unique page dynamique, composé de scripts,fichiers CSS etc...L'accés aux différentes fonctionnalités est géré par des overlay sans changer de page donc.

1.1.1 Communication avec le serveur

Premièrement nous avions réaliser une communication via servlet, or nous nous sommes rendus compte que ce n'étais absolument pas optimisé pour notre application. Effectivement, en utlisant la servlet nous étions obligé de renvoyer une page qui ne contennais, au final, qu'une très infime quantité d'information(Majoritairement une liste de chiffre). C'est pourquoi nous nous sommes orienté sur l'utilisation de Web Socket
Effectivement les Web Socket ont l'avantage :

- Avoir une connexion billatéral persistente au contraire de l'utilisation de servlet
- De simplifier l'implémentation de la communication
- De clarifier la lecture
- D'optimiser les requêtes en temps réel car les données transférées sont plus légères.

Nous nous intérésserons désormais au fonctionnement de l'application côté client.

1.1.2 Fonctionnement

Comme expliqué précedemment, la règle de notre jeu est simple :
Réussir à répeter la séquence envoyer par le serveur le plus de fois possible.
C'est pourquoi nous nous sommes orienté sur une programmation orienté événementielle, effectivement nous utilisons :

- L'utilisation des fonctions setTimeout/ClearTimeout afin de gérer :
  - Le temps d'affichage de chaque case
  - Le temps laissé au joueur pour cliquer sur une case
- Des listener, pour détécter les click de l'utilisateur

Nous nous attacherons par la suite à la conception de la partie serveur de notre application.

1.2 Architecture serveur

Dans cette partie , nous passerons en revue l'ensemble des points techniques proprent au développement du Backend.

1.2.1 Authentification

Notre authentification est directement gérée par Jbossa l'(aide de ses security realms ce qui permet de gérer différentes methodes d'authentification (Bqse de donnees, ldap, google et facebook, ...) De plus nous avons fait le choix d'utiliser un serveur Ldap (OpenDJ) contrairement à une base de données car mieux optimisé pour notre application. l'utilisqtion d'un ldap presente de nombreux avantages :

— Plus sécurisé, les mots de passe ne sont pas manipules directement et l'authentification par certificat est plus aisee.
— Information mieux architecturée ce qui implique des accés bien plus rapides et efficaces.

De plus nous disposons de plusieurs groupes d'utilisateurs egalement controlee par le module security integre a Jboss :

— Users : Compte basique accessible à tous.
— Premiums : Users ayant acheter (via Paypal) un accés Premium pour 1 mois leur permettant de ne plus être soumis aux publicites tout en garantissant un soutient financier.
— Admins : Utilisateur ayant le droit de modérer les gens dans le chat, supprimer des comptes etc... Toutefois pour le moment cela n'est pas implémenté mais le Backend est déjà opérationnel pour la mise en service.

1.2.2 WebSocket

Nous avons déjà éxpliqué pourquoi nous nous sommes tourné vers les WebSocket nous allons ici préciser leur utilisation dans notre implémentation. Nous utilisons les WebSocket pour les fonctionnalités suivantes :

— Envoie et récéption de séquences
— Utilisation du chat
— Informations sur les joueurs (Meilleur score etc...)

1.2.3 Liens Ldab/Base de données

La notion d'utilisateur est définie pour le Ldap en effet c'est ici que sont stockés les informations personnelles tel que :

— Adresse email
— Nom, Prénom

— Mot de passe

Ceci est différent de la notion de joueur qui est une représentation de l'utilisateur dans une partie qui quand à elle sera stockée dans une base de données. On peut par exemple retenir la monnaie virtuelle accumulée au cours des parties. Le lien entre base de donnees et ldap se fait par l'intermediaire du pseudo du joueur, correspondant a l'uid du user ldap.

2 Liens avec des services externes

Nous avons décider d'ajouter à notre application plusieurs spécifictés modernes tel que :

— La possibilité d'évoluer son compte au niveau de premium en utilisant Paypal.
— La connexion via un compte Google a été mise en place, evidemment l'inscription via un formulaire est également possible.
— L'ajout de son, lors de l'activation des différentes cases.
— L'utilisation de Adsense dans le but de générer de l'argent via les pubs. (Désactivé pour les utilisateurs premium).

3 Répartition du travail

Nous avons fait le choix de séparer le développement de l'application en deux groupes :

— La partie Client(FrontEnd) réalisée par Sacha et Kevin
— La partie Serveur(Backend) réalisée par Matthieu et Thibault

Evidemment même si le travail étais divisé en deux, nous avons beaucoup échangé tout au long du développement de notre application. En effet il est inenvisageable de développer une partie cliente et une partie serveur sans se mettre d'accord sur la manière de communiquer par exemple. C'est pourquoi, malgrés la division du travail, la cocneption à quand à elle été réalisée par l'ensemble du quadrinôme.

3.0.4 Utilisation de GitLab

Afin de mieux se répartir les tâches nous avons définis l'ensemble des tâches à réaliser, puis nous les avons trier par priorité. Chaque fois qu'un membre du groupe commencait ou finissait une tâche, il l'a mettait à jour sur le GitLab ce qui permettait aux autres membres du groupe de voir l'avancé du développement.

4 Annexes

4.0.5 Exemple de GitLab