Base du Web - TP n°8

Initiation à Javascript
Création d'un Chat avec Firebase

1 - Présentation de Firebase

En programmation web, la tendance actuelle est de déporter les traitements vers le client.

Même l'interaction avec les bases de données peut être effectuée par des programmes javascript.

Firebase est une plate-forme permettant le stockage et la synchronisation de données en ligne.

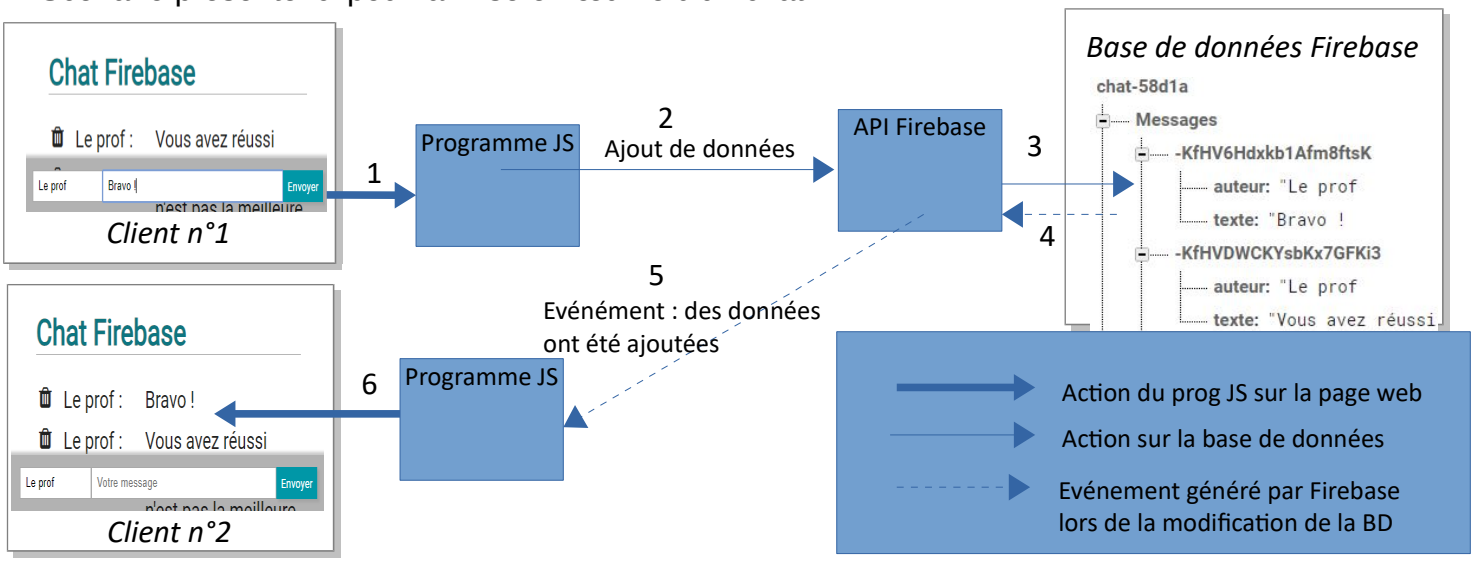
Firebase est "platform agnostic", cela signifie qu'il est utilisable par les clients Android, iOS et web.

Cette plate-forme a été choisie par l'équipe de *Google cloud platform* pour être l'une des solutions de stockage de données privilégiée pour les applications mobiles.

Les données stockées sont manipulables par des programmes javascript via une API (un ensemble de fonctions que nos programmes javascript pourront utiliser pour accéder aux bases de données).

L'ajout et la modification des données ainsi stockées génèrent des événements qui permettent d'actualiser en temps réel l'affichage des pages web chez les internautes :

Scénario présenté ici pour la mise en œuvre d'un chat :



1. Le programme javascript détecte la soumission du formulaire, il récupère le nom de l'utilisateur ("Le prof") et le texte de son message ("Bravo !")
2. Le programme javascript envoie ces données à l'API
3. L'API *Firebase* ajoute le couple de données *{auteur : "Le prof", texte : " Bravo !"}* à la base de données de cette application
4. L'ajout de données génère l'événement *child_added*
5. Cet événement est envoyé à tous les clients, avec les nouvelles données
6. L'événement est capté par le programme javascript, qui injecte les nouvelles données dans la page web.

2 - Manipulation : Création d'un chat

A. Ressources techniques

Sur *Moodle*, récupérez le site à compléter.
Tous les étudiants utiliserons la même base de données *Firebase* fournie par l'enseignant.

Visualisez la base de données présentée à l'écran par l'enseignant.

B. Analyse

Dans cette application, quels sont les événements déclenchés par l'internaute ?
Quelles seront les actions à associer à ces événements ?
Quel autre événement devra être capté par le programme gérant le contenu de la page web ?
Quelle devra être l'action à mettre en œuvre à la réception de cet événement ?

C. Site fourni

Avec votre éditeur préféré, ouvrez le site fourni.
Notez qu'*index.html* fait appel à plusieurs programmes javascript :

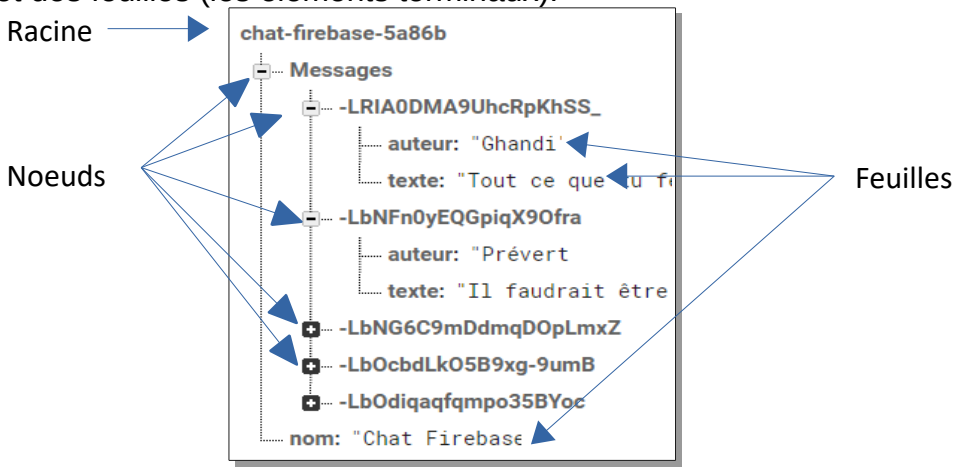
- *firebase.js* : la librairie définissant les fonctions *Firebase*
- *chat.js* : le programme à compléter, définissant le fonctionnement de notre application

Dans *chat.js* le programme de notre application initialise la communication avec la base de données Firebase.

D. Récupération des nœuds de la BD

Vous avez remarqué que la base de données Firebase est une BD NoSQL, dont les enregistrements sont stockés au format JSON dans un cloud.

Selon le format JSON, une donnée est un arbre au sens "structure de données" du terme. Un arbre possède une racine (le sommet de la base de données), des nœuds (les éléments intermédiaires) et des feuilles (les éléments terminaux).



Il est possible d'accéder à la base de données en entrant par la racine ou par l'un des nœuds.

En javascript et grâce à la librairie *firebase.js*, la récupération de la référence de la racine se fait de la manière suivante :

var racineBD = firebase.database();

A partir de la racine, on peut atteindre un nœud grâce à son nom, par exemple, pour connaître la référence du nœud "Message" : *var BDMessages = racineBD.ref('Messages') ;*

Après l'initialisation de la communication avec la BD, mettez en place ces deux instructions.

E. Ajout d'un message

Vous allez commencer par l'ajout d'un message dans la base de données.

Un message contient deux informations : l'auteur du message, et le texte du message.

En JSON, un message peut être décrit de la manière suivante :

{ "auteur" : "Simon", "texte" : "Chuis super content !"}

Utilisez *BDMessages.push(nouveauMessage) ;* pour ajouter dans la base de données, un message tel que celui donné en exemple. Personnalisez le message afin de le reconnaître dans la base de donnée affichée au tableau par l'enseignant.

Complétez votre programme pour que cet ajout se produise lorsque l'on valide le formulaire.

Modifiez le programme pour que le message contiennent le nom de l'auteur et le texte saisis grâce au formulaire.

F. Désactivation du rafraîchissement de la page

Notez que lorsque l'on valide le formulaire, le navigateur recharge toute la page web.

C'est le comportement par défaut lié à l'événement "submit".

Trouvez comment empêcher ce comportement.

G. Détection d'un message ajouté

Lorsqu'un objet est ajouté à la base de données, *Firebase* envoie un événement *child_added'* à votre programme.

Notez que le programme qui vous est fourni détecte cet événement avec :

BDMessages.on('child_added', insererDansLaPage) ;

Vérifiez que la fonction *insererDansLaPage* est bien déclenchée lorsqu'un message est ajouté dans la base de données.

Construisez progressivement la fonction de traitement, elle doit :

1. Afficher "un nouveau message a été publié !" dans la console.
2. Afficher l'auteur et le texte du message dans la console
3. Insérer le message dans la page

Les messages devront avoir le format suivant :

```
<li>    <i class="fa fa-trash"></i> <!-- Icône pour la suppression -->
        <span class="auteur">Bob</span>
        <span class="contenuMessage">Coucou!</span>
</li>
```

H. Suppression de messages

Si vous êtes en avance, essayez de mettre en place un système permettant de supprimer des messages. L'événement à détecter est 'child_removed'.

Le gestionnaire d'événement reçoit le message à supprimer. Vous pouvez afficher ce message pour voir que l'attribut *key* permet d'identifier le message ...

I. Récupération du nom du chat

La base de données contient le nom du chat.

Complétez le programme afin qu'il affiche ce nom dans le *h2#nomChat* de la page.

3 - En savoir plus sur Firebase

A. Création de règles d'accès au données

Le cahier des charges de notre web app peut préciser que le nom de l'auteur et le texte du message ne doivent pas être vides.

Ceci peut être assuré par le programme javascript, mais il est aussi possible de définir des règles de validation des données dans l'interface web de *Firebase*.

Vous ne pouvez définir ce type de règle, que si vous travaillez sur un jeu de données vous appartenant, et pas sur une base de données de démonstration.

Sur *firebase.com*, il faudra créer une nouvelle application et faire en sorte que le site web fonctionne avec la base de données de cette application.

Le menu *Manage App --> Security & Rules* permet de définir les règles d'accès aux données.

Ces règles peuvent être décrites sous la forme d'un texte JSON qui peut utiliser des opérateurs du langage javascript. Il faudra alors compléter la description suivante :

```
{    "rules": {                    // Racine des règles, le mot "rules" est obligatoire
      "Messages": {                // Description des règles concernant le nœud "Messages"
        "$unMessage" :{       // "$unMessage" désigne chaque élément du niveau supérieur ("Messages")
          ".read" : "true" ,          // Droit de lecture pour tous
          ".write" : true ,           // Droit d'écriture pour tous
          "auteur" :{".validate" :  // Pour préciser que la nouvelle valeur de auteur doit être non vide}
      }  // Fin de "Messages"
    } // Fin "rules"
} // Fin
```

B. Gestion de l'authentification des utilisateurs via un compte Facebook

On peut faire en sorte que seuls les internautes connectés puissent ajouter des messages.

Côté base de données, il faut mettre en place des règles de sécurité concernant l'accès aux données. L'appli doit donc utiliser votre propre base de données *FireBase*, au lieu de la base de données commune.

Au lieu de gérer une liste d'utilisateurs, il est courant de demander aux internautes de se connecter en utilisant leur compte *Facebook*.
Firebase fournit des outils facilitant cette méthode de connexion.

Le scénario de connexion est le suivant :

1. L'internaute clique sur un lien de connexion, ce qui déclenche un événement capté par un programme javascript.
2. Le programme javascript fait appel à une des fonctions de l'API Firebase (par exemple *authWithOAuthPopup*) qui va interroger la plate-forme *Firebase.com*.
3. Celle-ci vérifie que la connexion via Facebook a été configurée, puis elle tente d'accéder à une application Facebook (que vous devrez créer).
4. L'application *Facebook* demande le login et le mot de passe pour la connexion, et redirige l'internaute vers une page dont l'URL est stockée dans les paramètres de l'application *Facebook*.

Sur *firebase.com*, dans la rubrique *Authentification* de l'application, il faudra :

- Activer "Enable Facebook Authentification",
- Whitelister l'URL de votre application : *nom-de-mon-appli.firebaseio.com*. et *lpmiaw.univ-lr.fr* (ou *127.0.0.1* ou *localhost* si vous utilisez un serveur web local).

Il est possible de vérifier la communication entre *Firebase* et votre application *Facebook* en allant à l'adresse *https://auth.firebase.com/v2/nom-de-votre-appli/auth/facebook/callback.*

Il faut ensuite compéter le programme javascript pour qu'il permette la connexion lorsqu'on clique sur *Connexion Facebook*.

C. Prise en compte de l'authentification dans les règles d'accès aux données

Pour finir il faut compléter les règles d'accès aux données afin que seuls les internautes connectés puissent ajouter et modifier les données filles de "Messages".