AJAX - Enoncé 2

D.Moreaux

8 septembre 2024

Résumé

Ce second projet sert d'introduction à l'objet XMHttpRequest et aux accès à la base de données Postgresql. Il consistera à écrire des informations dans certaines tables.

1 Enoncé

Pour ce projet, on agira exclusivement au niveau de l'interface d'édition. On ajoutera les fonctionnalités suivantes :
— ajout d'une pièce
— ajout d'un message
— ajout d'un objet

Dans un projet ultérieur, l'interface correspondante devra être un peu revue (ajout du listing des éléments présents et possibilité de modification).

1.1 HTML

Au niveau de l'éditeur, on ajoutera trois divisions (si elles ne sont pas déjà présentes) : Description des pièces, messages et objets.

Le menu permettra de passer à ces affichages en utilisant une fonction séparée pour chaque situation (plus tard, il faudra ajouter dans ces fonctions le code qui chargera la liste des entrées présentes). On peut utiliser une fonction supplémentaire commune pour effectuer le changement de division (page) active (mais ce n'est pas une obligation)

Ces divisions proposeront un bouton : "nouveau". ce bouton affichera une division au dessus de la page (positionnement absolu et z-index) contenant une zone d'entrée (textarea) qui permettra d'entrer le texte, un bouton pour l'envoyer et un bouton pour annuler. Ces divisions *boîte de dialogue* seront distinctes (elles seront amenées à évoluer à l'avenir).

1.2 JS

Il faudra d'abord prévoir les fonctions qui permettent de passer d'une page à l'autre et d'afficher ou masquer les divisions boîte de dialogue (ne pas oublier de les masquer quand on arrive sur la page).

Au niveau des formulaires d'entrée, on effectuera une requête POST pour envoyer le texte au serveur. Ces fonctions seront modifiées ultérieurement pour déclencher une mise à jour de l'affichage.

1.3 PHP

Il faudra prévoir trois routines en PHP qui pousseront chacune un type de données dans la DB. Attention, pour les objets, pousser la position de départ -1 (non créé) et ignorer le champ wid (pour le moment). Les mécanismes permettant de gérer ces champs seront implémentés plus tard.

2 La DB

Les DB ont été alimentées automatiquement avec les tables vides (les messages systèmes et les paramètres sont déjà alimentés en données).

Pour vérifier le contenu des tables, il faudra utiliser l'interface de base de données de PHPStorm.

Pour se connecter à la DB, on utilisera la chaîne de connexion suivante :

```
pgsql:host=localhost;port=5433;dbname=js24xx;user=js24xx;
    password=motde12passe
```

où les valeurs de dbname, user et password seront adaptées à votre profil.

Pour insérer des données, on prendra soin de ne pas préciser la valeur du champ d'index auto-incrémenté (le type `SERIAL`) afin de permettre la génération d'une nouvelle valeur et de soit omettre soit préciser à NULL le champ `wid` de la table des objets `obj`.

Le fichier quill.sql permet de créer les tables et le fichier drop.sql permet de les effacer. Pour resetter les DB, il suffira donc d'exécuter ces deux fichiers au niveau de la liste des schémas, d'abord le drop.sql puis le quill.sql.

3 Structure de la DB

3.1 usr

Cette table contiendra les utilisateurs (joueurs)

uid	numéro d'utilisateur
login	nom d'utilisateur
passwd	mot de passe hashé en sha256
autosave	partie en cours sérialisée

3.2 savegame

Les parties sauvées seront stockées ici. Le numéro de slot permettra d'identifier les différentes parties d'un joueur

sid	numéro de sauvegarde
slot	emplacement de sauvegarde
uid	numéro de l'utilisateur
savedata	partie sérialisée

3.3 params

La table params contiendra les paramètres globaux de la partie : taille des mots et nombre d'objets pouvant être transportés. D'autres variables pourront apparaître plus tard

param	nom du paramètre
value	valeur (numérique) du paramètre

3.4 location

Cette table contiendra les description des différentes pièces. Cette table peut potentiellement contenir de l'HTML.

La pièce 0 (pièce de départ) est précréée.

rid	Numéro de la pièce
roomdesc	description de la pièce

3.5 obj

Cette table contiendra les informations sur les objets : description, position de départ et mot associé.

La position sera soit le numéro de la pièce associée, soit -1 (non créé), -2 (porté) ou -3 (transporté, dans l'inventaire) [1]

Le mot associé vaudra soit NULL si aucun mot n'est associé soit le numéro du mot. Si l'objet a un mot dont le numéro est supérieur ou

1. Dans le programme d'origine, les valeurs 252, 253 et 254 étaient utilisées

égal à 1000, il s'agira d'un objet pouvant être *porté* (vêtement, arme, ...) [2].

L'objet 0 (source de lumière) est précréé.

objid	numéro de l'objet
objdesc	description de l'objet, peut contenir de l'HTML
startloc	position de départ
wid	mot du dictionnaire associé à l'objet

3.6 msg

Cette table reprendra les messages. Ces messages pourront être en HTML.

mid	Numéro du message
message	texte du message, peut contenir de l'HTML

3.7 smsg

Cette table contiendra les messages système. Aucun message ne doit y être ajouté (leurs numéros sont réservés) et il sera possible de les éditer pour changer la langue de l'aventure.

Les messages sont précréés dans cette table et ils peuvent contenir de l'HTML.

smid	Numéro du message
message	texte du message, peut contenir de l'HTML

3.8 vocab

l|l Cette table contient le vocabulaire. Les mots sont uniques par contre les numéros ne le sont pas.

Les mots dont le numéro est inférieur ou égal à 12 correspondent aux directions. Les mots dont les numéros sont supérieurs ou égaux à 1000 correspondent à des objets qui peuvent être portés (vêtements,...)

La taille des mots sera limitée à *wordsize* lettres. Les mots devront donc être tronqués à cette taille avant d'être sauvés.

Les mots possédant le même numéro seront considérés comme synonymes. On pourrait par exemple définir GET, TAKE et GRAB comme étant synonymes.

wid	numéro du mot
word	mot

3.9 move

Cette table décrira les mouvements *automatiques*. Si la combinaison pièce/mot est trouvée, elle indiquera la nouvelle pièce dans laquelle se

2. dans le programme d'origine, c'était les mots dont le numéro était supérieur à 200

rendre.

rid	numéro de la pièce concernée
wid	numéro du mot concerné
newroom	numéro de la pièce destination

3.10 action

Cette table regroupera les informations de deux tables du jeu : la table de status et la table d'event (ou d'action).

Le champ tbl vaudra 0 pour un status (exécution avant l'entrée d'une commande) et 1 pour un evénement (effectue la commande entrée).

Les champs wid1 et wid2 seront testés avec les mots décodés de la ligne de commande dans le cas d'un événement. Une valeur NULL indiquera que n'importe quel mot est acceptable.

Le champ pgm contiendra les données sérialisées concernant les tests et les actions à effectuer.

aid	id de l'entrée
tbl	0 ou 1 selon que l'entrée concerne la table de status ou d'event
wid1	premier mot associé (verbe)
wid2	deuxième mot associé (objet)
pgm	conditions et actions sérialisées