AJAX - Enoncé 7

D.Moreaux

8 décembre 2024

Résumé

Réalisation du moteur de jeu sans les fonctionnalités de sauvegarde et de message demandant si on veut terminer/recommencer

1 Interface de jeu

Au niveau de l'interface de jeu, on ajoutera les fonctionnalités PAUSE et LOGOUT.

Pour PAUSE, si le bouton envoyer est enfoncé ou si une touche est enfoncée [1], annule le timer et revient au PHP. Si la durée envoyée est non nulle, ajoute un timer sur cette durée au terme duquel on prévoit aussi un retour au PHP

Pour LOGOUT, le programme reviendra à la page de connexion.

2 Machine d'état

Certains états devront être modifiés comme suit :

— INIT : Le tableau des drapeaux de 0 à 32 sera initialisé à 0. On chargera ensuite dans un tableau les positions initiales des objets. Le nombre d'objets ayant la valeur -3 (transporté) sera compté dans le drapeau 1.

— DESC : Si le drapeau 0 est non nul, on considérera pour la suite qu'il fait noir. On diminuera les drapeaux de 2 à 4 si ils sont nons nuls, 2 à chaque fois, 3 s'il fait noir et 4 s'il fait noir et l'objet 0 (source de lumière) est absent (ni inventaire ni position courante).

Si il fait noir et l'objet 0 est absent, on affichera le message système 0 (il fait sombre) sinon on affichera un texte comprenant la description et, si des objets sont présents, le message 1 et la liste des objets. On préparera l'analyse de la table de status, règle 0

1. optionnel

— TBL : Si on était pas en train de traiter les instructions d'une règle, on essaie de lire la règle courante. Si elle n'existe pas et qu'on était dans la table de status, on passe à l'état CMD en affichant un message au hasard parmi les messages systèmes 2 à 5.

  Si on était dans la table d'actions et qu'aucune action n'a été validée, on affiche le message système 7 ou 8 (1er mot < 13 ou pas) en ayant préparé le parcours de la table de status.

  Si on a pu lire la règle courante, on vérifie les conditions et (si table d'actions) les mots. Si ils ne sont pas validés, on passe à la règle suivante et on envoie un NOP. Sinon, on devra traiter les instructions une à la fois. On peut éventuellement utiliser un NOP pour revenir dans la machine d'état ou commencer directement.

  Si on arrive à la dernière instruction, on passe à la règle suivante et on envoie un NOP

  On peut utiliser deux états séparés pour les tables de status et d'action ou utiliser le même état et faire la différence autrement. Par exemple, la table devant être traitée peut être chargée dans la session.
— CMD Si un mouvement n'a pas pu être fait, on passera à l'état TBL à l'aide d'un NOP en sélectionnant la table d'action (sélectionner la règle 0).

  Les drapeaux 5 à 8 seront décrémentés si ils ne sont pas nuls. Le drapeau 9 le sera si il fait noir, le drapeau 10 si il fait noir et que l'objet 0 est absent. Le nombre de tours (flag 31) sera augmenté [2]
— Les états QUIT, END, LOAD et SAVE seront implémentés plus tard.

3 Les conditions

Les descriptions des conditions sont explicites. Les conditions seront testées dans un switch, une variable indiquant si une condition retourne une valeur fausse (pour arrêter le traitement).

La seule ajoute : si un drapeau inexistant est testé, le tableau sera étendu jusqu'à cette valeur et le drapeau sera mis à 0 avant d'être testé.

3.1 Les instructions

Les fonctionnements d'une partie des instructions sont décrits ici. Les autres instructions sont assez évidentes et ne demandent pas de description détaillée.

2. Le flag 32 ne sera pas lié au nombre de tours

N	Instr	description
1	INVENTORY	Envoie un message composé du message système 9 et de la liste des objets portés et transportés (ceux là seront suivi du message système 10). Si l'inventaire est vide, le message système 11 sera affiché à la place.
2	DESCRIBE	Repasse à l'état DESC (avec une action NOP)
3	QUIT	temporairement, passe à l'instruction suivante
4	END	temporairement, envoie LOGOUT
5	DONE	Arrête le traitement de la table en cours (les instructions restant et les règles restant ne seront pas examinés)
6	OK	Affiche le message système 15 puis fait DONE
10	TURNS	Affiche SM17, le nombre de tours, SM18, si le nombre est supérieur à 1 affiche SM19 et termine avec SM20 (tout sur une ligne)
11	SCORE	Affiche SM21, le score et SM22
13	DROPALL	Flag1 est mis à 0 et tous les objets transportés sont placés dans la pièce courante
19	REMOVE	si on ne porte pas l'objet, affiche SM23 et effectue DONE. Si flag1=maxobj, affiche SM24 et fait DONE. sinon, l'objet indiqué est mis comme transporté et flag1 incrémenté
20	GET	si l'objet est porté ou transporté, SM25 et DONE. Si il n'est pas dans la pièce courante, SM26 et done, si flag1=maxobj, SM27 et DONE, sinon, objet placé dans l'inventaire et flag1 incrémenté.
21	WEAR	Si objet déjà porté, SM29 et DONE. Si pas dans l'inventaire, SM28 et DONE. Sinon, il est mis comme porté et flag1 décrémenté
22	DROP	Si porté et flag1=maxobj, SM24 et DONE. Si ni porté ni transporté, SM28 et DONE. Sinon, placé dans la pièce courante et flag1 décrémenté.
23	CREATE	Objet mis à la pièce courante. Si il était transporté, flag1 est décrémenté.
24	DESTROY	Objet mis en not created. Si il était transporté, flag1 est décrémenté.
25	PLACE	L'objet est mis dans la pièce indiquée et flag1 décrémenté si il était transporté

Pour les drapeaux, si le résultat se retrouve en négatif, le met à 0.

Pour AUTOGET/DROP/REM/WEAR, le 2eme mot est recherché dans la table des objets. Si un objet est trouvé avec le mot en question, effectue l'opération correspondante avec ce numéro comme paramètre. sinon, affiche SM8 et fait DONE.

4 Conseils

Pour les actions, l'idéal est de traiter les AUTOGET/DROP/REM/WEAR en même temps que les GET/DROP/REM/WEAR. On peut mettre les deux `case` ensemble et utiliser un if() pour faire la différence (qui ne concerne que le choix de l'objet).

On se trouve devant 3 listes à traiter : les listes de règles, les listes de conditions et les listes d'instructions.

Pour les listes de règles, on peut au choix lire toutes les règles à chaque fois et prendre celle qui convient, utiliser LIMIT et OFFSET pour ne prendre qu'une règle après en avoir passé un certain nombre ou charger toutes les règles dans la session le temps du traitement.

Pour les listes de conditions, on fera une boucle dont on pourra sortir une fois toutes les conditions testées (les conditions changent une variable si elles ne sont pas vérifiées ou à chaque itération, tester la variable pour arrêter les conditions prématurément. A l'intérieur de la boucle, un `switch` permettra d'aiguiller vers les différents codes

Pour les listes d'instructions, il faudra faire plusieurs aller-retours entre le PHP et le navigateur. Le mieux est de prévoir une variable indiquant de sortir d'une boucle (certaines instructions comme LET, PLUS,...ne demandent pas d'actions côté navigateur). Mais, dans le pire des cas, on peut faire un passage par le navigateur à chaque fois (en utilisant l'action NOP).

De nouveau, un `switch` permettra d'aiguiller sur les différentes instructions.

Une fois les switchs d'instructions et de conditions mis en place, les instructions et les conditions pourront être implantées une après l'autre, séparément les unes des autres. Cette partie du code est plus longue que difficile.