LE SCROLLING

I rôle de cette partie dans le projet

Le scrolling est l'élément principal du déroulement du jeu. Appelé aussi défilement, ce système permet de faire défiler à l'écran les différentes couches que nous avons définies plus haut.
Tous les événements du jeu doivent intervenir pendant le jeu et donc pendant le scrolling.
Grâce à notre système en couche, il devient aisé de faire défiler les couches à l'écran sans pour autant modifier l'emplacement des tuiles en mémoire.
Notre scrolling consiste donc en un défilement graphique puisque les tuiles contenues en mémoire gardent leur position et doivent être remises à jour seulement lors des collisions.

II Etude de l'éxistant

Une recherche sur différents codes sources à été réalisée. Une analyse a été faite ce qui a permis les remarques suivantes:

En général, dans les jeux de type 'shoot them up' on rencontre deux type de scrolling.
Le premier système le plus simple se contente de générer un décor de fond aléatoirement.
Le deuxième type concerne l'élaboration de cartes ou 'map' qui determinent le niveau à l'avance.

III Réalisation

Nous avons choisi d'integrer les deux systèmes vus plus haut.

- D'une part nous générons aléatoirement des étoiles pour le fond sur notre couche fond
- D'autre part nous répartissons des tuiles sur notre couche objet.
- Enfin le scrolling fait défiler toutes nos couches.

a) Le but recherché

Le but de notre jeu est de mettre l'accent sur la fluidité, par conséquent d'assurer un déroulement efficace de notre scrolling. J'ai donc concu trois types de scrolling complètement indépendants mais qui ont en commun le défilement des couches.

b) Les types de scrolling implémentés

Prenons comme exemple un niveau de 10 écrans. Nous voulons faire défiler ces dix écrans les uns à la suite des autres. Chaque écran est constitué de tuiles de 16 pixels.
Par exemple sur une résolution de 640X480,
le nombre de tuiles est de 640/16 X 480/16 ==> 40 X 30 tuiles de 16 X 16 pixels.
Il nous faut donc faire défiler 10 écrans de 1200 tuiles (40 colonnes de 30 tuiles).
Plusieurs solutions s'offrent à nous, les voici:

1) Scrolling de toute la map.

Le plus simple à priori c'est de construire une seule grande bitmap de la taille de nos 10 écrans et de n'en faire afficher qu'une partie, en la décalant au fur et à mesure.

2) Scrolling en chargant les 10 écrans au pendant le scrolling.

Il est possible de « découper » le niveau en plusieurs bitmaps. Par exemple une bitmap de 640X480 pour chaque écran et il suffira de « charger » ces bitmaps au fur et à mesure.

3) Scrolling des écrans déjà chargés en mémoire

Cette dernière méthode reprend la précédente sauf que la mémoire pour les bitmaps est déjà allouée, ce qui est plus rapide car il n'y a aucun chargement pendant le scrolling.

c) Les problemes rencontrés

Cas 1) Le problème se situe au niveau de la fluidité car faire afficher à chaque fois une partie de la map de 6400X480 pixels entraine un ralentissement dû à la taille de la bitmap.

Cas 2) La fluidité est bonne mais le fait de charger les bitmaps pendant le scroling engendre des sacades. Ces sacades sont légèrement perceptibles mais cette méthode ne peut pas être retenue dans l'optique d'un scrolling le plus fluide possible.

Cas 3) Apparement aucun problème, le seul inconvénient est qu'il faut allouer de la mémoire a l'avance pour les bitmaps pour que l'éxécution se fasse avec une bonne optimisation.

d) Le scrolling choisi

Nous conservons donc la méthode 3, dont je vais détailler le fonctionnement dans la prochaine partie.

IV Fonctionnement

Notre scrolling est constité des différents éléments suivants:

- Des couches (fond, objet, sprite et propriété).
- Des buffers pour le décalage (scrolling).

a) Les couches

L'image de fond est une bitmap sur laquelle nous avons choisi de dessiner aléatoirement des petits cercles pleins pour représenter des étoiles. Cette image est ensuite chargée dans la couche fond.

Les tuiles de décor sont générées a des emplacements aléatoires dans la couche objet.

Les tuiles des sprites(vaisseaux) sont générées elles aussi de façon aléatoire dans la couche sprite.

b) Les buffers

Les couches sont superposées dans les buffers, ce qui permet alors de les décaler pour créer l'effet de scrolling. En fait, nous nous servons de trois buffers: le premier pour la partie droite, un second pour la partie gauche et un buffer final.

Les deux buffers sont affichés selon la position courante dans le buffer final. Celui-ci est effacé à chaque pas du scrolling car son état est différent à chaque décalage.

c) Le scrolling

Nous avons choisi de faire défiler les couches objet et fond mais la couche sprite ne défile pas. En effet, car cette couche est dynamique et on est obligé de la rafraîchir à chaque pas du scrolling. Donc, lors du scrolling, on décale l'affichage des couches du décor et du fond et on décale en mémoire les sprites (objets dynamiques).

Le résultat du décalage des couches et du rafraîchissement des sprites est mémorisé dans le buffer final puis affiché à l'écran.

Tous ces choix ont pour conséquence de produire un scrolling fluide. A chaque pas de ce scrolling, toutes les modifications se font en mémoire mais un seul affichage est éxécuté: celui du buffer final sur l'écran.

== > Ce qu'on voit à l'écran et qui vous donne l'impression d'un niveau qui défile n'est que l'assemblage de plusieurs couches sur lesquelles les tuiles ne se décalent pas à la même vitesse.