Sujet Rush Tek’av

En couleur et en 3D !

Dans ce rush, vous allez apprendre à utiliser la technologie webGL, qui permet d’utiliser la 3D dans un navigateur.

Vous aurez besoin d’un navigateur récent (Chrome ou Firefox) et d’un éditeur de texte (notepad++)

# La base du projet.

Durant ce rush, vous partirez d’une base contenant déjà des fonctions utiles. Vous pourrez ainsi vous concentrer sur votre sujet sans devoir maitriser les parties annexes (shaders, …)

Vous pouvez ouvrir le fichier index.html avec votre navigateur et voir en temps réel les modifications que vous ferez. Il suffira d’actualiser la page.

Ouvrer le fichier avec notepad++. Dans le code du fichier, chercher les fonctions *webGLStart* et *drawScene.* C’est ici que vous placerez votre code.

# Démarrer webGL

Vous devez en premier créer l’environnement webGL et créer les shaders. Il vous suffit de copier le code suivant au début de *webGLStart* :

//On charge le canvas

canvas = document.getElementById("canvas");

//On chharge webGL, et les shaders.

gl = initGL(canvas);

shaderProgram = initShaders(gl);

Après ce code, vous créerez vos objets 3D ici, laissez donc un peu de place pour cela.

À la fin de la fonction *webGLStart*, lorsque tout est créé, il faut mettre ces lignes :

//On configure webGL

gl.clearColor(0.0, 0.0, 0.2, 1.0);

gl.enable(gl.DEPTH\_TEST);

Les paramètres de gl.clearColor définissent la couleur de fond. Les 3 premiers nombre sont les valeurs des couleurs rouge, verts, bleu et alpha (entre 0 et 1, 1 étant coloré au maximum.)

Toutes les couleurs seront sur ce format. Les autres couleurs sont faites à partir de mélange de ces valeurs (par exemple tout mettre à 1.0 donne du blanc). La dernière valeur est l’alpha, et représente la transparence.

# Créer un objet

La création d’objet se fait dans *webGLStart*. Pour créer un objet, il suffit d’appeler la fonction *makeWebGLObject.* Elle prend en paramètre l’environnement webGL, un tableau de points 3D et un tableau de couleurs.

var triangle = makeWebGLObject(gl,

[

0.0, 1.0, 0.0,

-1.0, -1.0, 0.0,

1.0, -1.0, 0.0

], [

1.0, 0.0, 0.0, 1.0,

0.0, 1.0, 0.0, 1.0,

0.0, 0.0, 1.0, 1.0

]);

Pour chaque point, on donne ses coordonnées x, y et z. ET à chaque point on donne une couleur.

Essayez donc de faire d’autres formes, ou d’autres couleurs.

# Afficher l’objet

L’objet est créé, mais il ne s’affiche pas. Et c’est normal : il faut lui dire de s’afficher.

Première étape, nous avons besoin d’une caméra. Elle est créé avec *initCamera* qui a besoin du contexte openGL et retourne une caméra.

Une fois ceci fait, il vous fait préparer l’affichage de l’objet. La fonction *prepareDrawObject* va envoyer votre objet dans la mémoire vidéo. Comment l’appeler ? Ce sera à vous de regarder ça dans le code déjà existant.

Votre objet est prêt ? affichez le avec gl.drawArrays(gl.TRIANGLE\_STRIP, 0, votre\_objet.numItems);

Essayez donc d’afficher plusieurs objets, des carrés, …

# Bouger les objets

Tous les objets sont au même endroit, cela pose problème.

Comment les bouger ?

Chaque objet possède une matrice, qui définit sa position et sa direction. Elle est accessible dans *mon\_objet.matrix*.

Il existe plein de fonctions pour modifier ces matrices, notamment *mat4.translate(matrice, [0, 0, 0]);* et *mat4.rotate(matrice, 45, [0, 1, 0])* ;

À vous de tester ca et de modifier les valeurs !

Astuce : la caméra aussi est une matrice. Vous pouvez la bouger.

Vous avez fini ? Venez nous voir, on vous montrera comment animer tout cela.