ქვეფანჯრები

განვიხილოთ როგორ უნდა შევქმნათ ქვეფანჯრები და მენიუები GLUT-ის გამოყენებით. რამოდეიმე ფანჯარა საშუალებას გვაძლევს ერთდოულად დავხატოთ სხვადასხა სცენა ან იგივე სცენა სხვადასხვა პარამეტრებით.

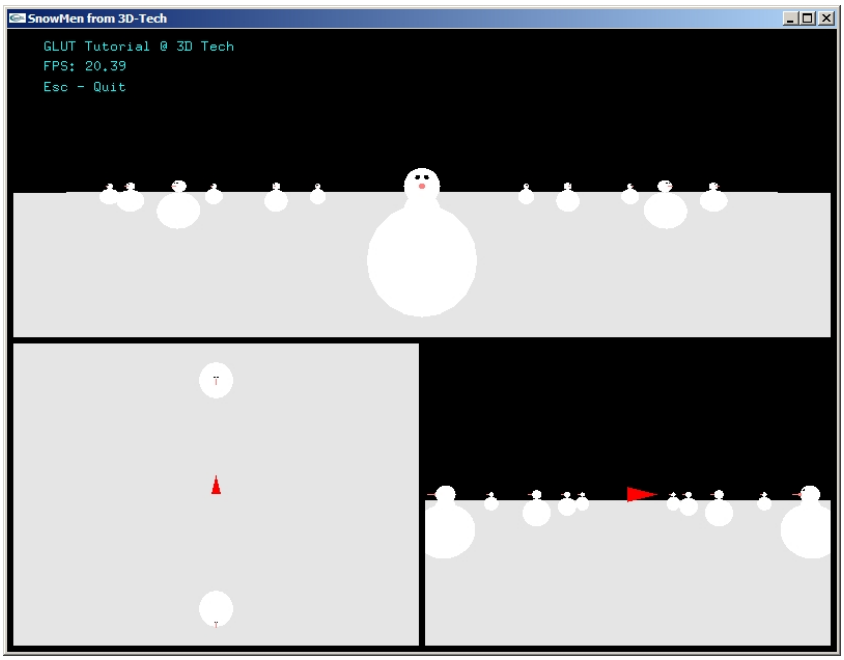
მაგალითად გაანვიხილოთ პროგრამა სადაც გვექნება 3 ქვეფანჯარა საკუთარი რენდერ ფუნქციებით.

- idle ფუნქცია - renderSceneAll
- მთავარი ფანჯრის სახატავი - renderScene
- 1 ფანჯრის სახატავი - renderScenesw1
- 2 ფანჯრის სახატავი - renderScenesw2
- 3 ფანჯრის სახატავი - renderScenesw3

CODE

```
mainWindow = glutCreateWindow("GLUT Multiwindow");
....

glutDisplayFunc(renderScene);
glutIdleFunc(renderSceneAll);

subWindow1 = glutCreateSubWindow(mainWindow, …
glutDisplayFunc(renderScene);

subWindow2 = glutCreateSubWindow(mainWindow, …
glutDisplayFunc(renderScenesw2);

subWindow3 = glutCreateSubWindow(mainWindow, …
glutDisplayFunc(renderScenesw3);
```

ქვემოთ მოყვანილ სურათზე ხდება ერთიდაიგივე სცენის, სხვადასხვა კუთხიდან დახატვა, 3 სხვადასხვა ფანჯარაში.



ქვემოთ მოყვანილია კოდის ფრაგმენტები 3-ე ფანჯრის გადახატვის ფუნქციებიდან.

გამოყენებული ცვლადების მნიშვნელობები:

- x,y,z : ჩვენი მიმდინარე პოზიცია

- lx,ly,lz : ეს ვექტორი განსაზღვრავს ხედვის არეს

- deltaMove, deltaAngle, angle : კამერის მოძრაობის პარამეტრები

გეომეტრიის რენდერი ხდება renderScene2 ფუნქციაში. სხვადასხვა სცენების რენდერისას განსხვავება არის მხოლოდ ის რომ ვასუფთავებთ MODEL_VIEW მატრიცას (loadIdentity), და ინიციალიზაციას ვუკეთებთ კამერას გარკვეულ პოზიციაზე(gluLookAt) ფუნქციის დახმარებით.

CODE

```
//main window
void renderScene() {

glutSetWindow(mainWindow);
glClear(GL_COLOR_BUFFER_BIT);
glutSwapBuffers();

}

//subwindow 1 - camera = current position
void renderScenesw1() {

glutSetWindow(subWindow1);
glLoadIdentity();
gluLookAt(x, y, z,
   x + lx,y + ly,z + lz,
   0.0f,1.0f,0.0f);

renderScene2(subWindow1);

}

// subwindow 2 - top view
void renderScenesw2() {

glutSetWindow(subWindow2);
glLoadIdentity();
gluLookAt(x, y+15, z,
   x ,y - 1,z,
   lx,0,lz);
renderScene2(subWindow2);

}

// subwindow 3 - right view
void renderScenesw3() {

glutSetWindow(subWindow3);
glLoadIdentity();
gluLookAt(x-lz*10 , y, z+lx*10,
   x ,y ,z ,
   0.0f,1.0f,0.0f);
renderScene2(subWindow3);

}
```

მთავარი ფანჯრის რენდერი იძახებს ყველა დანარჩენის გადახატვის ფუნქციებს.

CODE

```
void renderSceneAll() {

if (deltaMove)
 moveMeFlat(deltaMove);
if (deltaAngle) {
 angle += deltaAngle;
 orientMe(angle);
 }

renderScenesw1();
renderScenesw2();
renderScenesw3();

}
```

ქვეფანჯრების გადახატვისას აუცილებელია ჯერ ავირჩიოთ მიმდინარე ფანჯარა (glutSetWindow) ფუნქციის დახმარებით.

კონტექსტური მენიუს და ქვემენიუს შექმნა GLUT-ში

მენიუს შექმნა ხდება შემდეგი ფუნქციის საშუალებით:
CODE

```
int glutCreateMenu(void (*func)(int value));
```

პარამეტრად გადაეცემა ინ ფუნქციის სახელი რომელიც დაამუშავებს ამ მენიუს გამოძახებებს. ეს გვიბრუნებს მენიუს უნიკალურ იდენტიფიკატორს.

შემდეგი რაც უნდა გავაკეთოთ ესაა მენიუში პუნქტების ჩამატება:
CODE

```
void glutAddMenuEntry(char *name, int value);
```

name - ეს სახელი გამოჩნდება ეკრანზე მენიუს გააქტიურებისას.
value - ეს რიცხვი დაბრუნდება ამ პუნქტის არჩევისას.

და ბოლოს რაც უნდა გავაკეთოთ ესააა მენიუს უნდა მივაბათ თაგუნიას რომელიმე ღილაკზე:

CODE

```
void glutAttachMenu(int button);
```

button - შეიძლება იყოს ერთერთი ამათგან:

- GLUT_LEFT_BUTTON

- GLUT_MIDDLE_BUTTON

- GLUT_RIGHT_BUTTON

მენიუს მიბმის მოხსნა
CODE

```
void glutDetachMenu(int button);
```

და განადგურება
CODE

```
void glutDestroyMenu(int menuIdentifier);
```

ქვემენიუების შექმნა ხდება

CODE

```
void glutAddSubMenu(char *entryName, int menuIndex);
```

საშუალებით.

entryName - ეკრანზე გამოსაჩენი სტრიქონი.

menuIndex - მენიუს პუნქტის ინდექსი სადაც უნდა მიებას ქვემენიუ.

მენიუების შექმნის სადემონსტრაციო პროგრამა შეგიძლიათ იხილოთ და გაარჩიოთ საიტზე.