**Tutorial 5**

This tutorial is continuation of setting up game object and controlling them. This time we will render pong paddles for two players, set user control from the keyboard for puddles to move.

Declare global variable for puddle to operate.

irr::core::vector2df paddle1Pos;

irr::core::vector2df paddle2Pos;

bool paddle1Up = false;

bool paddle1Down = false;

bool paddle2Up = false;

bool paddle2Down = false;

The following code goes into event handler function. The purpose is to set up handling process for users pressing a particular button on keyboard. When button is pressed it will produce a response True/False whether button is pressed. Then this information can be used in the code to for purpose, updating puddles position.

if (event.EventType == irr::EET\_KEY\_INPUT\_EVENT)

{

// Record keys that are down into the key array.

KeyIsDown[event.KeyInput.Key] = event.KeyInput.PressedDown;

// Paddle1 control values.

if (KeyIsDown[irr::KEY\_KEY\_W]) { paddle1Up = true; }

else { paddle1Up = false; }

if (KeyIsDown[irr::KEY\_KEY\_D]) { paddle1Down = true; }

else { paddle1Down = false; }

// Paddle2 control values.

if (KeyIsDown[irr::KEY\_KEY\_O]) { paddle2Up = true; }

else { paddle2Up = false; }

if (KeyIsDown[irr::KEY\_KEY\_K]) { paddle2Down = true; }

else { paddle2Down = false; }

}

This code goes inside your updatePossition() function. Same story as for ball, when game receive an event from Event Handler function that particular button is pressed then we should update position vector(X,Y) of the buddle.

if (paddle1Up == true) { paddle1Pos.Y += dt\*paddle1Speed; }

if (paddle1Down == true) { paddle1Pos.Y -= dt\*paddle1Speed; }

if (paddle2Up == true) { paddle2Pos.Y += dt\*paddle2Speed; }

if (paddle2Down == true) { paddle2Pos.Y -= dt\*paddle2Speed; }

This code belongs to your main function but not inside game loop, that initially set up your puddles position on a screen.

paddle1Pos.X = 10;

paddle1Pos.Y = screenHeight\*.5 - paddle1->getSize().Height\*.5;

paddle2Pos.X = screenWidth - 10 - paddle2->getSize().Width;

paddle2Pos.Y = screenHeight\*.5 - paddle2->getSize().Height\*.5;

Now let’s draw puddles on the screen for each user. Code belongs to the main loop just under the code for ball rendering.

driver->draw2DImage(paddle1,

core::position2d<s32>(paddle1Pos.X, paddle1Pos.Y),

core::rect<s32>(0, 0, paddle1->getSize().Width, paddle1->getSize().Height),

0,

video::SColor(255, 255, 255, 255),

1);

driver->draw2DImage(paddle2,

core::position2d<s32>(paddle2Pos.X, paddle2Pos.Y),

core::rect<s32>(0, 0, paddle2->getSize().Width, paddle2->getSize().Height),

0,

video::SColor(255, 255, 255, 255),

1);

Note: Rendering order in the code matters which object will be on the top when two object overlap each other.

For now your code should be able to compile and run. You will see two puddles on the each border of the screen and a boll moving all over the place. Puddles can be controlled by W-D keys for player1 and O-K for player 2. Boll for now can be controlled by mouse clicking event.

That is it for this lecture!