**Multiplayer Assembly Package  
Tutorials  
Kill-Feeds**

This tutorial will teach you how to create a kill feed. A kill feed as it is called in a first person shooter game is a display usually on the bottom left side of the screen showing recent kills, containing who killed who and by what mean.

Start by opening the file: game/art/gui/playGui.gui. At the bottom of the file add the following code:

new GuiContainer(RecentKillList) {

margin = "0 0 0 0";

padding = "0 0 0 0";

anchorTop = "1";

anchorBottom = "0";

anchorLeft = "1";

anchorRight = "0";

position = "12 456";

extent = "339 303";

minExtent = "8 2";

horizSizing = "relative";

vertSizing = "relative";

profile = "GuiDefaultProfile";

visible = "1";

active = "1";

tooltipProfile = "GuiToolTipProfile";

hovertime = "1000";

isContainer = "1";

canSave = "1";

hidden = "1";

canSaveDynamicFields = "0";

firstElement = "0";

};

Create a new TorqueScript file on the client side, name it killFeed.cs and execute it in the init.cs file located in the client folder. Add the following to the file and save it.

$RKL::maxRecentDisplays = 8;

$RKL::ImagePath = "game/data/image/killGui/";

function clientCmdAddGameKill(%killer, %killerWeapon, %victim, %kt, %vt) {

addToKillList(%killer, %killerWeapon, %victim, %kt, %vt);

}

function addToKillList(%killer, %killerWeapon, %victim, %kt, %vt) {

%this = RecentKillList;

%element = %this.firstElement;

cancel(%this.hideSched);

%this.setVisible(true);

%this.hideSched = schedule(10000, 0, removeKillList);

//code for deploying elements - We start by pushing down the existing elements

while(%this.killLine[%element] !$= "") {

if(%element > $RKL::maxRecentDisplays) {

%this.killLine[%element].delete();

%this.firstElement++;

}

else {

//move down

%this.killLine[%element].setPosition(0, getWord(%this.killLine[%element].getPosition(), 1)+50);

%element++;

}

}

//code to add the new line to the top

if(%killer $= "") {

//this is the case if special deaths occured, such as falling, suicide, ect.

AddKillElement(%element, %victim, %killerWeapon, "", %vt, "");

}

else {

AddKillElement(%element, %killer, %killerWeapon, %victim, %kt, %vt);

}

}

function removeKillList() {

%this = RecentKillList;

%this.setVisible(false);

%element = 0;

while(%this.killLine[%element] !$= "") {

%this.killLine[%element].delete();

%element++;

}

%this.firstElement = 0;

}

function AddKillElement(%kID, %killer, %killerWeapon, %victim, %killerTeam, %victimTeam) {

%this = RecentKillList;

%image = $RKL::ImagePath@%killerWeapon@".png";

%kCol = %killerTeam == 0 ? "<color:ffffff>" : (%killerTeam == 1 ? "<color:660198>" : (%killerTeam == 2 ? "<color:3D5229>" : (%killerTeam == 3 ? "<color:00688B>" : "<color:990000>")));

%vCol = %victimTeam == 0 ? "<color:ffffff>" : (%victimTeam == 1 ? "<color:660198>" : (%victimTeam == 2 ? "<color:3D5229>" : (%victimTeam == 3 ? "<color:00688B>" : "<color:990000>")));

%this.killLine[%kID] = new GuiContainer() {

margin = "0 0 0 0";

padding = "0 0 0 0";

anchorTop = "1";

anchorBottom = "0";

anchorLeft = "1";

anchorRight = "0";

position = "0 0";

extent = "334 50";

minExtent = "8 2";

horizSizing = "right";

vertSizing = "bottom";

profile = "GuiDefaultProfile";

visible = "1";

active = "1";

tooltipProfile = "GuiToolTipProfile";

hovertime = "1000";

isContainer = "1";

canSave = "1";

canSaveDynamicFields = "0";

new GuiMLTextCtrl() {

lineSpacing = "2";

allowColorChars = "0";

maxChars = "-1";

text = "<Just:Center><Font:Arial:18>"@%kCol@""@%killer;

useURLMouseCursor = "0";

position = "5 16";

extent = "122 18";

minExtent = "8 2";

horizSizing = "right";

vertSizing = "bottom";

profile = "GuiMLTextProfile";

visible = "1";

active = "1";

tooltipProfile = "GuiToolTipProfile";

hovertime = "1000";

isContainer = "0";

canSave = "1";

canSaveDynamicFields = "0";

};

new GuiBitmapCtrl() {

bitmap = %image;

wrap = "0";

position = "129 2";

extent = "67 46";

minExtent = "8 2";

horizSizing = "right";

vertSizing = "bottom";

profile = "GuiDefaultProfile";

visible = "1";

active = "1";

tooltipProfile = "GuiToolTipProfile";

hovertime = "1000";

isContainer = "0";

canSave = "1";

canSaveDynamicFields = "0";

};

//if(%victim !$= "") {

new GuiMLTextCtrl() {

lineSpacing = "2";

allowColorChars = "0";

maxChars = "-1";

text = "<Just:Center><Font:Arial:18>"@%vCol@""@%victim;

useURLMouseCursor = "0";

position = "204 16";

extent = "122 18";

minExtent = "8 2";

horizSizing = "right";

vertSizing = "bottom";

profile = "GuiMLTextProfile";

visible = "1";

active = "1";

tooltipProfile = "GuiToolTipProfile";

hovertime = "1000";

isContainer = "0";

canSave = "1";

canSaveDynamicFields = "0";

};

//}

};

%this.add(%this.killLine[%kID]);

}

(Note: This file is located in the Tutorial Files Folder)

You will need to change the first line to match a path of where you will store the killed by image files, I have included samples for the Lurker, Ryder, and Disposition Shotgun (if you have the GG Soldier Art Pack), also located in the tutorial files folder Now we have the client side methods complete, lets add functioning to the server so it can send messages down.

On the server side files, open the file gameCore.cs and scroll down to the method named: function GameCore::onDeath, and modify it to read as follows:

function GameCore::onDeath(%game, %client, %sourceObject, %sourceClient, %damageType, %damLoc)

{

//echo (%game @"\c4 -> "@ %game.class @" -> GameCore::onDeath");

// clear the weaponHUD

%client.RefreshWeaponHud(0, "", "");

// Clear out the name on the corpse

%client.player.setShapeName("");

// Update the numerical Health HUD

%client.player.updateHealth();

// Switch the client over to the death cam and unhook the player object.

if (isObject(%client.camera) && isObject(%client.player))

{

%client.camera.setMode("Corpse", %client.player);

%client.setControlObject(%client.camera);

}

%client.player = 0;

// Display damage appropriate kill message

%sendMsgFunction = "sendMsgClientKilled\_" @ %damageType;

if ( !isFunction( %sendMsgFunction ) )

%sendMsgFunction = "sendMsgClientKilled\_Default";

call( %sendMsgFunction, 'MsgClientKilled', %client, %sourceClient, %damLoc );

// Dole out points and check for win

if ( %damageType $= "Suicide" || %sourceClient == %client )

{

%game.incDeaths( %client, 1, true );

%game.incScore( %client, -1, false );

for(%i = 0; %i < ClientGroup.getCount(); %i++) {

commandToClient(ClientGroup.getObject(%i), 'AddGameKill', "", %damageType, %client.namebase, "", %client.player.team);

}

}

else

{

%game.incDeaths( %client, 1, false );

%game.incScore( %sourceClient, 1, true );

%game.incKills( %sourceClient, 1, false );

if(%damageType $= " TurretDamage") {

if(%sourceObject.client $= "" || !%sourceObject.client) {

commandToClient(ClientGroup.getObject(%i), 'AddGameKill', "", %damageType, %client.namebase, "", %client.player.team);

}

else {

commandToClient(ClientGroup.getObject(%i), 'AddGameKill', %sourceClient.namebase, %damageType, %client.namebase, %sourceObject.team, %client.player.team);

}

}

else {

for(%i = 0; %i < ClientGroup.getCount(); %i++) {

commandToClient(ClientGroup.getObject(%i), 'AddGameKill', %sourceClient.namebase, %damageType, %client.namebase, 1, 2);

}

}

// If the game may be ended by a client getting a particular score, check that now.

if ( $Game::EndGameScore > 0 && %sourceClient.kills >= $Game::EndGameScore )

%game.cycleGame();

}

}

Now the server will send messages to every client when they die regarding the messages pushing the proper image. But wait… Will it? In the stock engine itself, you’re not quite done just yet. The field sent down to the game in this regard is damageType, which is defined in the projectiles of your weapons. So ensure the defined damageType field matches the name of your images. And you’re all set!