Subject: Real Time on the HUD
Posted by Raptor RSF on Fri, 05 Feb 2010 20:58:55 GMT
View Forum Message <> Reply to Message

Hey guys, i made some c++ real time clock for renegade HUD's.

Anybody that can make a Working ini reader for this, will be my hero I cannot get that to work because my lack of experience. I wanted it so that people can give up their timezone in the ini file. Its all because i want valid daylight saving times for the clock.
More info:
 http://wwp.greenwichmeantime.com/time-zone/europe/european-union/central-europea n-time/
http://www.timeanddate.com/library/abbreviations/timezones/na/est.html

HUD.ini

```
[General]
; System Time (Created by: Raptor RSF)
SystemTimeEnabled=true

[SystemTime]
Text.Font.File = font18x24radiobm-big.tga
Text.Font.AverageCharacterHeight = 0
Text.Position.Centered = false
Text.Position.X = -100.0
Text.Position.Y = -155.0
;PLZ anyone, get this to valid cpp for me :)= Standard.Time.Zone = EST ;By default its EST
;PLZ anyone, get this to valid cpp for me :)= Daylight.Time.Zone = EDT  ;By default its EDT

SystemTimeColor = 1
```

systemtime.h

```
/* SystemTimeItemClass
 Copyright 2009 Mark Sararu
 This part of the shaders code was created by Raptor*[RSF]

 This file is part of the Renegade scripts.dll
 The Renegade scripts.dll is free software; you can redistribute it and/or modify it under
 the terms of the GNU General Public License as published by the Free
 Software Foundation; either version 2, or (at your option) any later
 version. See the file COPYING for more details.
 In addition, an exemption is given to allow Run Time Dynamic Linking of this code with any
closed source module
 that does not contain code covered by this licence.
 Only the source code to the module(s) containing the licenced code has to be released.
*/

#ifndef SHADERS_SYSTEMTIME_H_
#define SHADERS_SYSTEMTIME_H_

class SystemTimeItemClass
{
protected:
 bool Enabled;
 bool Enabled2;

 Render2DClass*  Render2D;
 Render2DTextClass* Render2DText;

 Vector2    TextPosition;
 char *    TextFontFile;

public:
 SystemTimeItemClass();
 ~SystemTimeItemClass();
 void Load(INIClass* ini);
 void Render();
};

extern SystemTimeItemClass SystemTime;

#endif
```

systemtime.cpp

```
/* SystemTimeItemClass
 Copyright 2009 Mark Sararu
 This part of the shaders code was created by Raptor*[RSF]

 This file is part of the Renegade scripts.dll
 The Renegade scripts.dll is free software; you can redistribute it and/or modify it under
 the terms of the GNU General Public License as published by the Free
 Software Foundation; either version 2, or (at your option) any later
 version. See the file COPYING for more details.
 In addition, an exemption is given to allow Run Time Dynamic Linking of this code with any
closed source module
 that does not contain code covered by this licence.
 Only the source code to the module(s) containing the licenced code has to be released.
*/

#include "scripts.h"
#include "shadereng.h"
#include "systemtime.h"
#include <time.h>

// structure of color data
struct Color
{
 unsigned int ColorValue;
 float Value;
};


SimpleDynVecClass<unsigned int> *Colors8;
unsigned long SystemTimeColor = 0;

SystemTimeItemClass::SystemTimeItemClass():
 Enabled(false),
 Enabled2(false),
 Render2D(NULL),
 Render2DText(NULL),
 TextPosition(0, 0),
 TextFontFile(NULL)
{
};

SystemTimeItemClass::~SystemTimeItemClass()
{
 SAFE_DELETE(Render2D);
 SAFE_DELETE(Render2DText);
 SAFE_DELETE(TextFontFile);
};

void SystemTimeItemClass::Load(INIClass *ini)
{
 if (!ini) return; // if you don't have an ini, something is horribly wrong!

 const char* section_name = "SystemTime";

 Enabled = ini->Get_Bool(section_name, "SystemTimeEnabled", false);
 Enabled2 = ini->Get_Bool("General", "SystemTimeEnabled", false);
 if ((!Enabled) && (!Enabled2)) return;
```

```
// Gathers the colors from hud.ini
Colors8 = new SimpleDynVecClass<unsigned int>;

unsigned int color = RGB(255,255,255)+0xFF000000;
Colors8->Add(color);
unsigned int colors8 = ini->Get_Int("General","ColorCount",0);
for (unsigned int i = 0;i < colors8;i++)
{
 char section[10];
 sprintf(section,"Color%d",i+1);
 unsigned int Red = ini->Get_Int(section,"Red",255);
 unsigned int Green = ini->Get_Int(section,"Green",255);
 unsigned int Blue = ini->Get_Int(section,"Blue",255);
 unsigned int Alpha = (ini->Get_Int(section,"Alpha",255) << 24);
 color = RGB(Blue,Green,Red)+Alpha;
 Colors8->Add(color);
}
unsigned int SystemTimeCol = ini->Get_Int(section_name,"SystemTimeColor",0);
SystemTimeColor = (*Colors8)[SystemTimeCol];

Render2D = CreateRender2DClass();

Vector2 screen_center;
screen_center.X = (ScreenResolution->Right - ScreenResolution->Left) / 2.0f;
screen_center.Y = (ScreenResolution->Bottom - ScreenResolution->Top) / 2.0f;

char temp[512];
ini->Get_String(section_name, "Text.Font.File", "DEFAULT_FONT", temp, 512);
Render2DText = CreateRender2DTextClass(temp);
TextFontFile = newstr(temp);

float average_height = ini->Get_Float(section_name, "Text.Font.AverageCharacterHeight", 16);

bool text_centered = ini->Get_Bool(section_name, "Text.Position.Centered", true);
TextPosition.X = ini->Get_Float(section_name, "Text.Position.X", 0.0f);
TextPosition.Y = ini->Get_Float(section_name, "Text.Position.Y", 0.0f);
if (TextPosition.X < 0)
{
 TextPosition.X += ScreenResolution->Right;
}
if (TextPosition.Y < 0)
{
 TextPosition.Y += ScreenResolution->Bottom;
}
if (text_centered)
{
 TextPosition = TextPosition + screen_center;
```

```
  TextPosition.Y -= average_height / 2.0f;
 }
};



void SystemTimeItemClass::Render()
{
 if ((!Enabled) && (!Enabled2)) return;


    char Time_Zone[4];
 const char Standard_Time_Zone[32] = "CET";
 const char Daylight_Time_Zone[32] = "CEST";

 tm *ptm;
 time_t *cur_time;

    // Set up the memory for the time and time time struct.
    cur_time = new time_t;
    ptm = new tm;

    // Get the time, then create the struct with time values.
    time(cur_time);
    ptm = localtime(cur_time);

    // Determine whether it is daylight savings time or not.
    if (ptm->tm_isdst)
        strcat(Time_Zone,Daylight_Time_Zone);
    else
        strcat(Time_Zone,Standard_Time_Zone);

 unsigned int color = 0;
 color = SystemTimeColor;

 Render2DText->Reset();
 RectClass *r = (RectClass *)((char *)Render2DText+0x5B8);
 r->Top = TextPosition.Y;
 r->Left = TextPosition.X;
 r->Bottom = TextPosition.Y;
 r->Right = TextPosition.X;
 char text[64];
 sprintf(text,"%02d:%02d:%02d" ,ptm->tm_hour,ptm->tm_min,ptm->tm_sec);
 Render2DText->Draw_Text(text, color);
 Render2DText->Render();
};

  //---------------------------------------------------------------------------- --
```

```
// globals
  //--------------------------------------------------------------------------- --
SystemTimeItemClass SystemTime;
```

shaderhud.cpp

```
#include "systemtime.h" // SystemTime

SystemTime.Load(hudini); // SystemTime

SystemTime.Render(); // SystemTime
```