This is miscellaneous PSP tutorial m9

Introduction

Hi everybody,

This is a very short tutorial on creating a frames per second display. This is useful for seeing what impact new code has on your game. We will use the code from tutorial 1: The game loop as the basis for this tutorial.

Frames Per Second (FPS)

Whenever you have implemented some code you like to know what impact it has. With a FPS you can see how much frames per second are being drawn. With this we can see how much frames we lost since we added the new code. If you loose a lot of frames per second the code is heavy and maybe you can find a way to lessen the strain on your game with revising the code.

The code itself is not very difficult. It is really the same as with every other language. FPS calculation is done using ticks. A tick is the smallest unit of time recognized by the PSP, which runs at 222 or 333 MHz. This means that there are 222 or 333 million clock ticks (or cycles) per second. With this very small timestep we can calculate the frames per second very precise.

Lets take a look at the source:

GameApp.h

```
int frames; // for calculating the frames per second
u32 res; // resolution
u64 now; // the ticks at current point
u64 previous; // the ticks at the previous point
char fps[100]; // the text buffer
```

These are the variables that are to be added in the gameapp header. The frames are use to see how many frames passed in a second. The res value will be explained with the .cpp file. The now and previous variables are to save the ticks at the current moment and the previously checked moment. The fps holds the frames per second value.

GameApp.cpp

```
sceRtcGetCurrentTick(&previous;);
res = sceRtcGetTickResolution();
```

These two lines are to be added in the load function. The first line fill up the previous with the current tick because we are going to use that in the render function. The second line saves the number of ticks in a second into the variable res. Lets take a look at the render function:

```
void GameApp::Render() {
   // render information.
   frames++;
   sceRtcGetCurrentTick( &now; );

   if( ((now - previous)/((float)res)) >= 1.0f ) {
```

```
        previous = now;
        sprintf(fps, "FPS: %d", frames);
        frames = 0;
    }

    pspDebugScreenSetXY(0, 0);
    pspDebugScreenPrintf("Frames per second: %s\n", fps);
    sceDisplayWaitVblankStart();
    sceGuSwapBuffers();
}
```

In the render function we count frames. Then we check if a second has passed by subtracting the previous frametickcount from the current tickcount and divide it by the resolution. If it is, we reset the framecount and save the number of frames into a char array.

Then we setup the position to render our debug text with the frames per second. This is all the code we need. The frames per second are displayed in the left top corner.

MakeFile

The makefile for tutorial m9

```
TARGET = out
OBJS = $(wildcard *.cpp) $(wildcard *.c)

INCDIR =
CFLAGS = -O2 -G0 -Wall -g
CXXFLAGS = $(CFLAGS) -fno-exceptions -fno-rtti -g
ASFLAGS = $(CFLAGS)

LIBDIR =
LDFLAGS =
LIBS = -lc -g -lpspgum -lpspgu -lstdc++ -lm -lpsppower -lpsprtc

PSPSDK=$(shell psp-config --pspsdk-path)
include $(PSPSDK)/lib/build.mak
```

Compile and see what happens :D

Source files

Download the source files.