Drawing Text with SDL

Note: This tutorial assumes that you already know how to display a window and draw a sprite with SDL.

Setting Up the SDL TrueType Font Library

To display text with SDL, you need the `SDL_ttf` library. You can download the library here (For windows, grab **SDL_ttf-devel-2.0.7-VC6.zip**). Once you've downloaded the file, extract it to a folder of your choice.

You now need to tell your IDE where to find the SDL_ttf files. This is the same process as telling SDL where to find the SDL files (remember the **lib** and **include** folders?). Since I'm using Visual Studio .NET 2003, I'll go over how to set up SDL_ttf with that IDE.

Select **Tools->Options**. Go to **Projects** in the left pane and select **VC++ Directories**. In the **Show directories for:** menu, select **Include files**. Click the **New Line** button (looks like a folder with a star behind it) and then click the **...** button that appears. Navigate to where you extracted the SDL_ttf files, highlight the **include** folder and click **Open**.

Again, select **Tools->Options**. Go to **Projects** in the left pane and select **VC++ Directories**. In the **Show directories for:** menu, select **Library files**. Click the **New Line** button (looks like a folder with a star behind it) and then click the **...** button that appears. Navigate to where you extracted the SDL_ttf files, highlight the **lib** folder and click **Open**.

Once you've created a project and added a C++ file to it, you need to go to **Project->*Your Project* Properties**, click the **Linker** folder, and select **Input**. In **Additional Dependencies**, you need to have "sdl.lib sdlmain.lib sdl_ttf.lib", without the quotes.

In the **lib** folder, under the folder where you extracted the SDL_ttf files, you'll find the file "SDL_ttf.dll". Copy and paste this file into your project directory. When you distribute your program, be sure to include this file in the same directory as the .exe.

The TrueType font library requires a font. Fonts come in font files. Click here to get the font file for Arial. Put this file in your project directory.

Displaying Text with SDL_ttf

To display text, we first initialize SDL_ttf. We then create a surface that holds the text and blit that surface to our screen buffer. Finally, we shut down SDL_ttf and free the text surface.

Initializing SDL_ttf.

We initialize SDL_ttf with a call to `TTF_Init()`, which takes no parameters.

Creating a surface with some text on it.

To create a surface with the text we want to display, we call `TTF_RenderText_Shaded()`, which takes four parameters and returns a pointer to an `SDL_Surface`.

The first parameter to `TTF_RenderText_Shaded()` is a pointer to a TTF_Font structure. We get this structure with a call to `TTF_OpenFont()`, which takes two paramters and returns a pointer to a `TTF_Font` structure. The first parameter to `TTF_OpenFont()` is the name of the file that contains the font information. The second parameter is the size we want our text to be.

The second parameter to `TTF_RenderText_Shaded()` is the text we want displayed.

The third parameter is an `SDL_Color` structure that stores the foreground color of our text. The fourth parameter is an `SDL_Color` structure that stores the background color of our text. The foreground color of the text you're reading right now is white; the background color is dark blue (at least I think it is...apparently I'm partially color blind, whatever that means).

The `SDL_Color` structure has three variables: a red value, a blue value, and a green value.

Blitting the text surface.

Blitting the text surface is the same as blitting a sprite surface. The only difference is that you'll usually want to blit the entire text surface. This means that you only need to fill in an `SDL_Rect` structure for the location of the text on the screen. You can just pass `NULL` to `SDL_BlitSurface()` for the source location.

Shutting down SDL_ttf.

SDL_ttf is shutdown with a call to `TTF_Quit()`, which takes no parameters.

You also have to free the memory that the `TTF_Font` structure is using. This is done with a call to `TTF_CloseFont()`.

Since we created a surface to store our text, we shouldn't forget to free it with a call to `SDL_FreeSurface()`.

Here's some code that draws a string of text to the screen. I've bolded the parts that relate to drawing text.

```
#include "SDL.h"
#include "SDL_TTF.h"

const int WINDOW_WIDTH = 640;
const int WINDOW_HEIGHT = 480;
const char* WINDOW_TITLE = "SDL Start";

int main(int argc, char **argv)
{
   SDL_Init( SDL_INIT_VIDEO );

   TTF_Init();

   SDL_Surface* screen = SDL_SetVideoMode( WINDOW_WIDTH,
WINDOW_HEIGHT, 0,
      SDL_HWSURFACE | SDL_DOUBLEBUF );
   SDL_WM_SetCaption( WINDOW_TITLE, 0 );

   TTF_Font* font = TTF_OpenFont("ARIAL.TTF", 12);

   SDL_Color foregroundColor = { 255, 255, 255 };
   SDL_Color backgroundColor = { 0, 0, 255 };

   SDL_Surface* textSurface = TTF_RenderText_Shaded(font, "This is my
text.",
      foregroundColor, backgroundColor);

   // Pass zero for width and height to draw the whole surface
   SDL_Rect textLocation = { 100, 100, 0, 0 };

   SDL_Event event;
   bool gameRunning = true;

   while (gameRunning)
   {
      if (SDL_PollEvent(&event))
      {
         if (event.type == SDL_QUIT)
         {
            gameRunning = false;
         }
```

```
      }
      SDL_FillRect(screen, NULL, SDL_MapRGB(screen->format, 0, 0,
0));
      SDL_BlitSurface(textSurface, NULL, screen, &textLocation);
      SDL_Flip(screen);
   }
   SDL_FreeSurface(textSurface);
   TTF_CloseFont(font);
   TTF_Quit();
   SDL_Quit();
   return 0;
}
```

Exercises

1. If you have a lot of text to draw, you'll probably want to put all of the code to do so in a function. Try implementing such a function on your own.

Here's an example of a function that draws text.

```
// TTF_Init() must be called before using this function.
// Remember to call TTF_Quit() when done.
void drawText(SDL_Surface* screen,
              char* string,
              int size,
              int x, int y,
              int fR, int fG, int fB,
              int bR, int bG, int bB)
{
   TTF_Font* font = TTF_OpenFont("ARIAL.TTF", size);
   SDL_Color foregroundColor = { fR, fG, fB };
   SDL_Color backgroundColor = { bR, bG, bB };
   SDL_Surface* textSurface = TTF_RenderText_Shaded(font, string,
      foregroundColor, backgroundColor);
   SDL_Rect textLocation = { x, y, 0, 0 };
   SDL_BlitSurface(textSurface, NULL, screen, &textLocation);
   SDL_FreeSurface(textSurface);
   TTF_CloseFont(font);
}
```

One thing to notice here is that the font is created and destroyed each time the function is called. I've never found this to be a problem, but if you find yourself having performance issues with your text drawing, this would be one place to look at.