



Documentation CrocoLib

0) Sommaire

[1\) Introduction](#)

[2\) Prototypes](#)

[3\) Détails](#)

[3.1\) CrocoCreate](#)

[3.2\) CrocoSetData](#)

[3.3\) CrocoUpdate](#)

[3.4\) CrocoClear](#)

[3.5\) CrocoPutPixel](#)

[3.6\) CrocoDrawLine](#)

[3.7\) CrocoWriteBMP](#)

[3.8\) CrocoRelease](#)

[4\) Exemple](#)

[5\) Informations](#)

[6\) Download](#)

1) Introduction

La CrocoLib est une petite lib graphique, qui permet de faire des rendus 2D (voir meme 3D) de maniere tres rapide. Comme meilleure exemple, enfin c'est un mauvais exemple, mais bon quand meme, enfin on s'en fout, c'est le systeme de X11, qui permet de creer un buffer en memoire, de "tracer" dedans et de flusher le buffer une fois le rendu en "background" terminer. J'ai ecrit cette lib car je n'en trouvais pas d'equivalente sous Win32, apart TinyPTC, mais apres quelques grief avec cette lib, et ne me plaisant pas vraiment, j'ai decider de recoder la mienne.

2) Prototypes

Rien de tel que des bons vieux prototypes pour expliquer la synthèse d'une lib, j'ai pas raison ? =) Ils sont tellement simple et explicite, que je ne les commenterai pas, cependant, je détaillerais quelques fonctions par la suite.

```
int CrocoCreate(char *title, int w, int h);
int CrocoSetData(int *buffer);
int CrocoUpdate();
int CrocoClear();
int CrocoPutPixel(int x, int y, int color);
void CrocoDrawLine(int x1, int y1, int x2, int y2, int color);
int CrocoWriteBMP(char *filename);
int CrocoRelease();
```

Jusque la rien de bien compliqué.

3) Détails

3.1) CrocoCreate

Elle crée une fenêtre, portant le nom `title` et de taille `w` (+x) et `h` (+y). `x` et `y` étant des valeurs d'ajustement afin que se soit votre zone d'affichage qui soit de taille `w` et `h`, et non pas la taille de la fenêtre.

Il n'est pas prévu de pouvoir en créer plusieurs, mais c'est volontaire, le but était de faire le truc le plus simple possible, il n'était pas question de faire apparaître le moindre accès à l'API.

NB : Cette fonction désactive l'écran de veille...

3.2) CrocoSetData

Cette fonction vous permet de définir le buffer de travail que vous aller utiliser. Cette fonction vous permet d'utiliser plusieurs buffer, vous pouvez les swaper quand vous le souhaitez.

La fonction ne prend qu'un `int*` car elle travail en 32 bits, encore un choix arbitraire, mais c'est pour que celui qui ne souhaite pas utiliser le canal alpha puisse travailler tout en le zappant, et aussi permettre a celui qui souhaite s'en servir, de ne pas se prendre la tete a regler le depth et autre (un changement de format de pixel implique plusieurs petit changement qui peuvent s'averer chiant dans l'utilisation d'une lib censer etre simple).

NB : Vous pouvez set a `NULL`, mais sachez que rien ne se produira, l'affichage sera conservé, mais si vous deplacer la fenetre ou que vous la redimensionné, ca fera des choses étrange à l'affichage.

3.3) CrocoUpdate

Cette fonction permet d'updater le contenu du buffer afin de l'afficher a l'écran, en gros, c'est un `PutImage`, je n'ai rien a ajouté sur cette fonction si ce n'est qu'il est logique que cette fonction update le buffer définit par `CrocoSetData()`.

3.4) CrocoClear

Je ne vous cacherais pas que cette fonction fait un `memset` sur le buffer que vous avez définit avec `CrocoSetData()`, rien de plus... pour les curieux :

```
int    CrocoClear()
{
    if (!_croco_buffer)
        return (0);

    memset(_croco_buffer, 0, _croco_w * _croco_h * sizeof(int));
    return (1);
}
```

Cette fonction clear le buffer definit par `CrocoSetData()`.

3.5) CrocoPutPixel

Cette fonction n'est pas non plus obligatoire, libre a vous de faire `mon_buffer[(y * width) + x] = color_32_bits`. Cependant, cette fonction est sécurisée et fait tout les test pour vous, a savoir, depacement de la fenetre (pratique pour pas tracer a gauche quand on dessine a droite (dans un buffer lineaire)) ou encore le test du buffer, si vous en avez pas definit.

Cette fonction trace dans le buffer definit par `CrocoSetData()`.

3.6) CrocoDrawLine

Bon bah pas besoin de détails, ca trace dans le buffer definit par `CrocoSetData()` et puis voila, c'est tout.

NB : Le CrocoDrawLine repose sur l'algo de tracé de ligne le plus rapide qui existe, vous pouvez lui faire confiance.

3.7) CrocoWriteBMP

Bon bah pareil, ca vous sauvegarde une image bitmap 32 bits a l'endroit definit par `filename`. Cette bmp sera l'image du buffer en cours, a savoir celui qui aura été definit par `CrocoSetData()`.

3.8) CrocoRelease

Cette fonction cloture votre session de CrocoLibing ;) mais elle est cependant importante, elle detruit ce qu'elle a allouée, elle detruit la fenetre de maniere conventionnel, et elle réactive aussi l'ecran de veille, bref, c'est pas trop négligeable.

Cette fonction est appelée automatiquement a la fermeture de la fenetre quand vous cliquez sur la petite croix en haut a gauche.

4) Exemple

Un exemple minimale au niveau des Croco fonctions consisterais juste a créer une fenetre, définir un buffer, updater, et fermer. On peut donc écrire un truc avec 4 appels de fonctions. Biensure, on pourais vider le buffer au moins, mais ce n'est pas forcément necessaire, tout depend de ce que l'on fait.

Aussi, faire un tracer quelque part serais un plus, mais cela ne rentre pas dans le carde de la CrocoLib.

```
#include "crocolib.h"

#define WIDTH          640
#define HEIGHT        480
#define PECHERESSE    5

int    gl_buffer[WIDTH * HEIGHT];

int    main(int ac, char **av)
{
    CrocoCreate("Exemple de programme fait en CrocoLib",
                WIDTH, HEIGHT);
    CrocoSetData(gl_buffer);

    // ici, un tracer serais le bienvenue;

    while (33 || 51 || 1664 || PECHERESSE)
    {
        // ou encore ici pour le tracé;
        CrocoUpdate();
    }

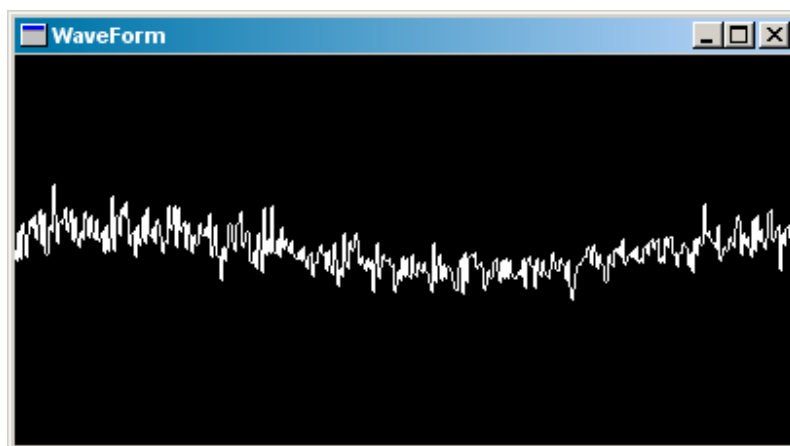
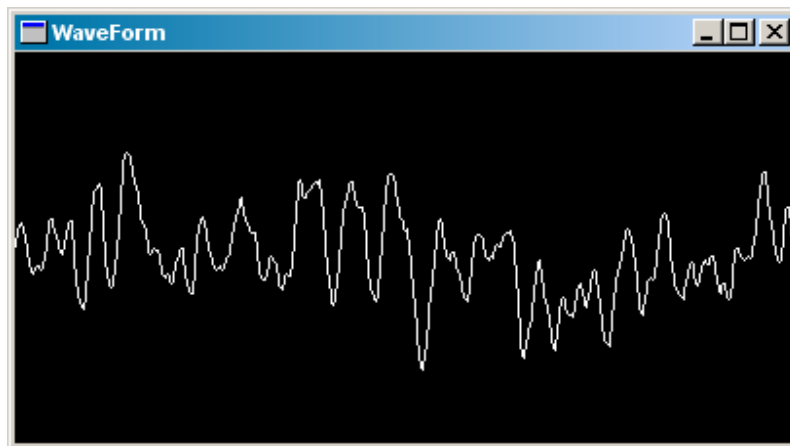
    CrocoRelease();
    return (0);
}
```

Notez que le buffer est déclarer en global, c'est fait exprès car les variables déclarées dans les fonctions sont allouées sur la pile, et un tel buffer, s'il avait été déclarer dans le main, aurais provoqué un stack overflow directe à l'exécution.

Vous trouverez avec l'archive crocotest.zip, le code source d'un exemple qui fait un dégradé de couleur, je m'étais servie de la CrocoLib pour en faire un pour le HeatDifuser, ca donne ca, c'est tout con, mais ca m'as permis de tester mon dégradé en 2 secondes :



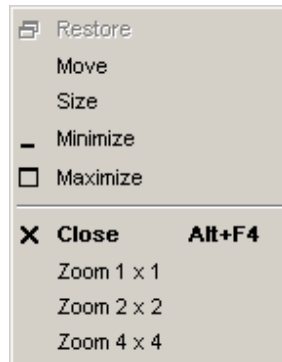
Aussi pour un logiciel de communication par Internet, la CrocoLib m'as permis de tester extremement rapidement si j'acquierais bien les frequences vocal comme il fallais, force etais de constater que oui =)



Mais on pourrais aussi par exemple créer une fenetre en 240x160, certains comprendront ce que cela signifie ;)

5) Informations

Dans le menu contextuel qui apparaît lorsque l'on clique sur l'icône de programme, sur la Croco fenêtre, se trouvent 3 options de "zoom", 1x1, 2x2 et 4x4, ceci est pour redimensionner la fenêtre dans des proportions égales tout en agrandissant le display, c'est aussi pratique lorsque vous avez redimensionné dans tous les sens et que vous souhaitez revenir en 1x1.



Sachez aussi que redimensionner la fenêtre implique un "stretch" de la zone affichable, et que ceci diminue, non pas considérablement, mais quand même bien méchamment, les performances de l'application. Faites y attention.

Pour finir, je reste entièrement disponible pour d'éventuelles questions et/ou modifications sur la lib, n'hésitez pas à me solliciter.

6) Download

CrocoLib :

Tout ce dont vous avez besoin pour programmer rapidement une petite application graphique. Cette archive inclut le header de prototypes, et la lib qui va avec (.h/.lib)

CrocoLib [<http://crocodile.free.fr/crocolib/crocolib.zip>]

CrocoTest :

Ceci est le code source du programme de test qui vous permettra de comprendre tout les mécanismes "*Ô combien compliqués*" :P de la CrocoLib. Nan mais étant donné qu'il fonctionne bien, il peut vous servir de base, ou encore de modèle pour déboguer votre propre programme s'il ne fonctionnait pas correctement.

CrocoTest [<http://crocodile.free.fr/crocolib/crocotest.zip>]

CrocoDoc :

Cette archive contient la documentation que vous êtes actuellement en train de lire, ceci peut être pratique si vous souhaitez en avoir une copie lisible offline, ou encore si jamais la doc est temporairement indisponible, bah c'est cool de pouvoir quand même la consulter, même si elle n'est franchement pas utile une fois lue.

CrocoDoc [<http://crocodile.free.fr/crocolib/crocodoc.zip>]

Sébastien ROBERT
sebseb@securesphere.net