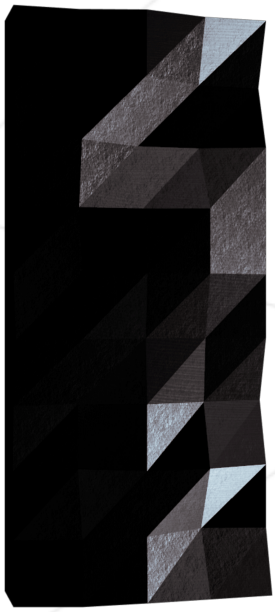

Retro MFA

Retro-engineering of MFA files

Summary: Searching images hidden in a closed file format.

Version: 1.00

Contents

Chapter I

Forewords

Originally born in 1993 as “futureSplash”, Flash and its Action Script programming language did spread fast over the early internet, used on many website. Providing enhanced graphics and animations, may web-based games were created using this technology. The arrival of the HTML5 features did start the slow decrease of Flash, down to its extinction in 2020. Well known for its many security holes and fixes, the world is safer now (or not, make your choice). Hopefully, this rush is absolutely not about Flash.

Chapter II

Introduction

The goal of this project Retro MFA is to find pictures hidden in a closed file format. We have at our disposal several `.mfa` files. They come from a very old tool used to easily create small 2D games. All these MFA files contain sprites: small pictures used in the game.

But we do have a problem: we do not know the format of these MFA files. Your job will be to partially retro-engineer these files, and create a program that can extract and display multiple images from thoses files.

This rush may seem complex but with determination you can do it!

To make it easier to reverse the files, there are a lot of different possibilities. Strong debate between all the participants is probably the key to get interesting tools and ideas.

Chapter III

General guidelines

- Your project must be written in **C**.
- Your program should not quit unexpectedly (segmentation fault, bus error, double free, etc) apart from undefined behaviors. If this happens, your project will be considered non functional and will receive a `0` during the evaluation.
- All heap allocated memory space must be properly freed when necessary. No leaks will be tolerated.
- You must submit a `Makefile` which will compile your source files to the required output with the flags `-Wall`, `-Wextra` and `-Werror`, use cc, and your Makefile must not relink.
- Your `Makefile` must at least contain the rules `$(NAME)`, `all`, `clean`, `fclean` and `re`.

`There is no norm for this project.`

Chapter IV

Mandatory part

You have to create a program. The program shoud be written in **C**.

Your program must respect these rules:

- The program name is **retromfa**.
- Your program must expect a "mfa" file as argument.
- If an error occurs during the execution of the program an error message should be displayed on the **standard error**.

There are no forbidden functions.

A few extra indications:

- This rush and the defense will only use the provided .mfa files
- Your program must display **at least 5 different images** from each file when you run it.
- You don't need to understand the entire file structure, just extract some images.
- You can add any hard-coded data for the specific files if you need it

Your program must be able to handle all provided mfa files without re-compiling.

Since you can use all the functions you want you can also use the graphic libraries of your choice. It is however recommended to use the **mlx** for this rush.

Here is an example of the expected output:

Disclaimer: this rush is for learning purpose only. The retro-engineering done during this project cannot be used for any action against or infrigement to the copyrights and intellectual property of the owner of the file format and associated software.

Disclaimer 2: this rush will not be offered in the future if the MFA file format description can be found online.

Chapter V

Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double check the names of your folders and files to ensure they are correct.