MY HUNTER

THE DUCK HUNT LEGACY

MY HUNTER

binary name: my_hunter
language: C
compilation: via Makefile, including re, clean and fclean rules

- ✓ The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- ✓ All the bonus files (including a potential specific Makefile) should be in a directory named bonus.
- ✓ Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

In this project, you are asked to make a small video game based on the rules of Duck Hunt.

The basic rules for the my_hunter are as follows:

- ✓ the player is a hunter who shoots ducks.
- ✓ ducks must appear on the screen and move from one side to another.
- ✓ the player can click on them to shoot them.

Being able to manage inputs from the user and to display animated sprites are key skills when you want to develop basic games.

You will reuse your work for other video game projects, so think about it to make it the most scalable way.

Your project should of course fit the requirements below but we are expecting it to be more than just a technical demonstration: we want you to think about it as a real game.

This project is probably your first video game ever. Try to have fun making it! You don't have to stick to the Duck Hunt theme as long as you respect the set of rules given here.

Requirements

MUST

✓ The window **must** be closed using events.
✓ The program **must** manage the input from the mouse click.
✓ The program **must** contain animated sprites rendered thanks to sprite sheets.
✓ The program **must** contain moving (rotating, translating, or scaling) elements.
✓ The program **must** accept the “-h” option, then display a short description of the program, and the available user inputs.

SHOULD

✓ Animations and movements in your program **should** not depend on the speed of your computer.
✓ Animations and movements in your program **should** be timed by *sfClock* elements.
✓ Your window **should** stick between 800x600 pixels and 1920x1080 pixels.
✓ Your window **should** have a limited frame rate such that it can be compute without lagging.

COULD

✓ The program **could** have several different levels.
✓ The program **could** display the score of the player.
✓ The program **could** store the highest score made.
✓ The program **could** display a small target under the mouse cursor.

The size of your repository (including the assets) must be as small as possible. Think about the format and the encoding of your resource files (sounds, musics, images, etc.).

An average maximal size might be 15MB, all included.

Conception

Like for any other project, we are obviously asking you to respect the coding style.

For you it will probably be the first time that you make a video game, it will be tempting to code everything inside of your main function because you are not yet familiar with graphical programming and with the CSFML. But you should take seriously the principles given in the coding style, they will only help you have a better project from which you will be able to reuse parts for future game and CSFML projects in this unit.

Authorized functions

Here is the full list of authorized functions.

from the C library

- ✓ malloc
- ✓ free
- ✓ memset
- ✓ rand
- ✓ srand
- ✓ time (only with srand)
- ✓ (f)open
- ✓ (f)read
- ✓ (f)write
- ✓ (f)close
- ✓ getline

from the CSFML library
All functions

from the math library
All functions

Any unspecified functions are de facto banned v 2.2

{EPITECH}