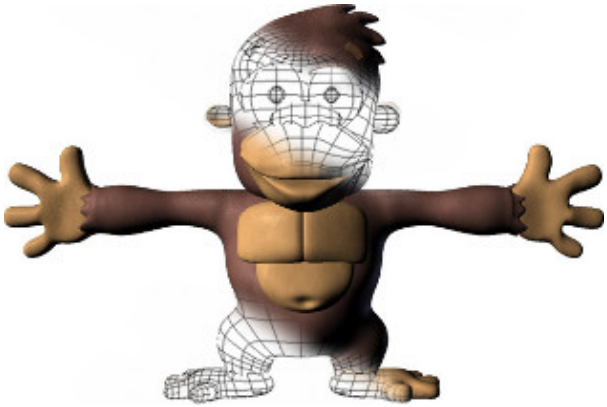
B2 - C Graphical Programming

B-MUL-200

my_rpg

It's your turn to create your final fantasy



my_rpg

binary name: my_rpg
language: C

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- 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).

This project is one of the freest project of your first year. Create your own RPG.
Your main challenge for this game will be to create a complete product using everything that you and your team know.

To give the users the feeling that you're delivering a complete product you need to polish as much as possible your game.
Your game must follow the following rules:

- Having a pleasant user interface.
- Create a coherent universe (visual assets, audio assets, scenario, …)
- Create a funny game where the player has at least one goal.
- Create a game with a beginning and an end.

You should think and look for information about how to create a simple yet entertaining RPG.

The size of your repository (including the assets) must be as small as possible.
Think of the format and the encoding of your resource files (sounds, music, images,..).
An average maximal size might be 30MB, all included.
Any repository exceeding this limit might not be evaluated at all.

Requirements

Mandatory

- The player needs to have characteristics
- The player can fight enemies AND the characteristics impact the fights results
- The player have an inventory which can contain a limited set of items
- The player can earn experience by winning fights and/or accomplishing specific actions
- The player can level up thanks to experience AND upgrading its characteristics
- The game contains NPC
- The game contains at least one quest
- The game contains a how to play system at the very beginning

Technical Requierements

This project, being the last project of the module, the following requierement are the mathematical and technical parts which has to be present in your final project:

- A collision system including moving and static elements with different shapes.
- Camera movements (zoom, translation, rotation).
- 3D effects (depth scaling, isometric projection…).
- Buttons must have at least 3 differents state
- Saving system

You can try to use `sfView` elements

Must

General

- The window may be closed using events,
- The windows may have differents mode :

- Window mode
- Full-screen mode

- The windows may have differents resolutions, at least two (1920x1080, and an other)
- The game manages inputs from the mouse click and keyboard,
- The game contains animated sprites rendered thanks to sprite sheets,
- Animations in your program are frame rate independent,
- Animations and movements in your program are timed by clocks.

Main menu

- Must contain at least these options
  - Start the game
  - Resume game (grayed out if not possible)
  - Settings
  - Quit the game
- The settings option must contain
  - Sound and music volume options
  - A window size and resolution options

In game menu

- Must be accessible by pressing the `Escape` key
- Must contain at least these options
  - Character options
    - equipment and inventory management
    - characteristics management
  - Saving
  - Loading
  - Settings
    - the same that main menu
    - commands list

The starting and the game menu MUST be two different scenes.

SHOULD

- Splash screen at the beginning
- Command edit into the in game menu
- As much information as possible about the game should be stored in a configuration file.
- The game should have an advanced collision system to manage complex fighting.
- The game should contains simple particle system :

- with at least 2 types of particle,
- particle effects (changing colors, scaling, bouncing, fading) to simulate realistic environment (wind, fire, rain, snow…).

Could

- Have a skill tree, unlocking different abilities (active and passive).
- Have a complete particle engine.
- Use scripting to describe entities.
- Have a map editor.

AUTHORIZED FUNCTIONS

Here is the full list of authorized functions.

from the CSFML library
All functions

from the math library
All functions

from the C library

malloc	**getline**	**(f)write**
free	**stat**	**opendir**
memset	**(f)open**	**readdir**
(s)rand	**(f)read**	**closedir**
	(f)close	

Any unspecified functions are de facto banned, except for bonus features.