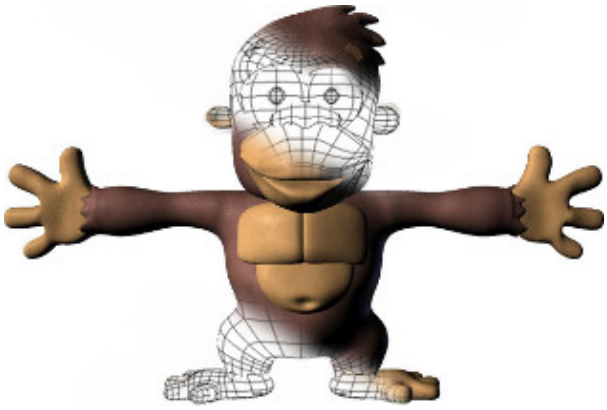
B2 - C Graphical Programming

B-MUL-200

my_rpg

It's your turn to create your final fantasy



1.4

my_rpg

binary name: my_rpg
repository name: MUL_my_rpg_$ACADEMIC_YEAR
repository rights: ramassage-tek
language: C

- Your repository must contain the totality of your source files, but no useless files (binary, temp files, obj files,...).
- 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.

Your game must follow the following rules:

- The player needs to have characteristics which you can find in the status menu.
- The player can fight enemies, statistics will impact the fights results.
- There must be NPC in your game.
- You need to implement at least one quest.
- The player must have an inventory which can contain a limited set of items.
- The player can earn experience by winning fights and accomplishing specific actions.
- With enough experience, the player can level up, upgrading its statistics.

To give the users the feeling that you're delivering a complete product you need to polish as much as possible your game.

- 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.

Requierements

Mandatory

The following features are **mandatory** (if your project is missing one of them it will not be evaluated further):

- the window can be closed using events,
- the game manages the input 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.

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.
- A simple particle system that can display at least 2 types of particles.
- Particle effects (changing colors, scaling, bouncing, fading) to simulate realistic environment (wind, fire, rain, snow…).
- Camera movements (zoom, translation, rotation).
- 3D effects (depth scaling, isometric projection…).

Must

The game **must** have:

- A starting menu with at least two buttons, one to launch a game, and one to quit the game.
- An `escape` key to pause the game when launched.
- A menu when the game is paused with at least two buttons, one to go to the starting menu and the other to leave the game.
- A basic fighting system.
- An inventory and status menu.

The starting menu and the game **must** be two different scenes.

Should

- Your window **should** stick between 800x600 pixels and 1920x1080 pixels.
- The game **should** have an "How To play" menu, explaining how to play your game.
- The game **should** have NPC with whom the player can interact (fight, quest, discuss).
- As much information as possible about the game **should** be stored in a configuration file.
- The buttons in your game **should** have at least three visual states: idle, hover, and clicked.
- If your game has cut scenes or an animated intro (and it **should**) the player **should** be able to skip it.
- The game **should** have a beginning and an end.
- The game **should** have an advanced collision system to manage complex fighting.

Could

The game **could**:

- let the player save and load its own save.
- let the user customize its character.
- have different types of enemies.
- have a skill tree, unlocking different abilities (active and passive).
- have a "settings" menu that **could** contain sound options and/or screen size options.
- have a particle engine.
- use scripting to describe entities.
- have a map editor.

Would

Your program **would** be a real video game.

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.

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	**(f)open**	**opendir**
memset	**(f)read**	**readdir**
(s)rand	**(f)close**	**closedir**

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