// CORE ENGINE VARIABLES

// An entitymap is defined as a link from an ID to a pointer to Entity

typedef std::multimap<int, shared\_ptr<Entity> > EntityMap;

// List of entities to be removed at the end of the frame

std::vector<int> idRemovalList;

// Keep track of currently available IDs

long long currentId;

long long currentViewId;

// Is vertical synchronisation on?

bool vSync;

// Pointer to the current music track

sf::Sound \*currentMusic;

// We sure need an event handler

EventHandler eventHandler;

// Is the application closing?

bool isClosing;

// The game's window for rendering to

sf::RenderWindow renderWindow;

// Camera transform when following player

sf::View gameView;

// A definition of how large the game field is, changes on directory size

PlayingField field;

// Contains all of the entities in the application

EntityMap entities;

// The text to draw above the player

std::string playerName;

// A list of messages displayed at the bottom left corner of the screen

std::vector<std::string> messageList;

// 5 at the moment, how many messages can show up on the screen, scrolling

int maxMessages;

// For camera following

bool following; // Am I following anything?

Entity \*toFollow; // What am I following?

sf::Vector2f cdelta; // The current direction of the camera

// Keeps track of loaded sounds

std::map< std::string, sf::SoundBuffer > soundMap;

// Keeps track of sounds that are actually playing

std::vector< sf::Sound\* > playingSounds;

// Keep track of game options

std::map< std::string, std::string > gameOptions;

// Keep track of loaded images

std::map< std::string, sf::Image > imageMap;

// Keep track of loaded fonts

std::map< std::string, sf::Font > fontMap;

// Previous, current directory

std::string oldpath;

std::string currentPath;

// For game dialogue

std::string currentDialogue; // Filename

std::string currentSubtitles; // Subtitles shown on the screen

sf::Sound \*currentDialoguePtr; // So that we can see if it's finished

// Is the game currently running (else in menu)

bool inGame;

// Have all the enemies been destroyed in this level?

bool currentLevelUnlocked;

// A timer to respawn the player if they've been dead for a while

sf::Clock playerDeathTimer;