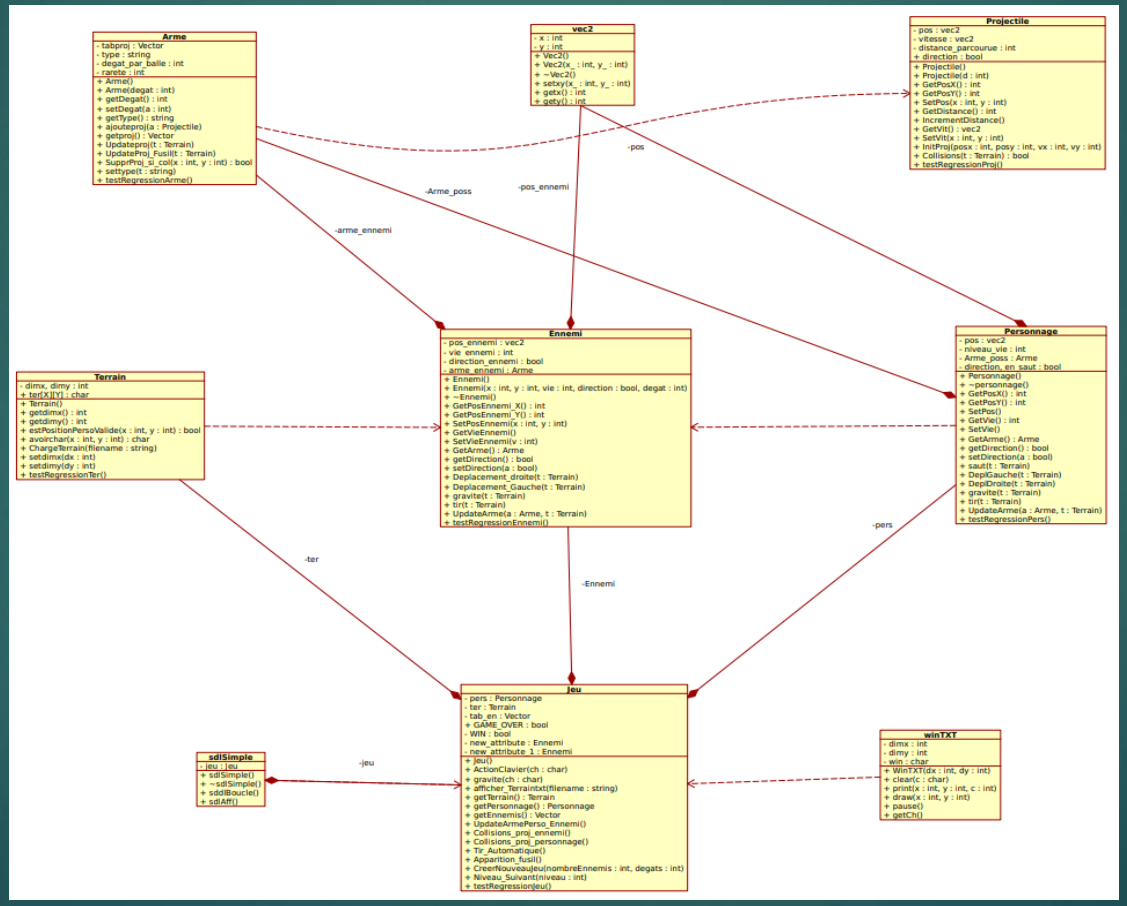
Gun Runner

PROJET LIFAPCD

AUTEURS: NECHADI MEHDI, EMMANUEL GOKANA

Le but du jeu est d'éliminer tout les
ennemis présent sur le terrain sans se faire tuer par les nombreux
obstacles (et les ennemis), c'est un jeu de plateforme qui se joue en solo.

Vue d'ensemble du diagramme des Classes



Classe Personnage

```
class Personnage
{
    public:
        ///@brief constructeur de la classe personnage
        Personnage() ;
        ///@brief destructeur de la classe personnage
        ~Personnage();
        ///@brief recupere la donnee x de la position du personnage
        int GetPosX() const;  // v◆rifi◆
        ///@brief recupere la donnee y de la position du personnage
        int GetPosY() const;  // v◆rifi◆
        ///@brief change les donnes x et y de la position du personnage
        void SetPos(int x, int y);  //v◆rifi◆
        ///@brief recupere les points de vie du personnage
        int GetVie();
        ///@brief change les points de vie du personnage
        void SetVie(int v);
        ///@brief recupere l'arme du personage
        Arme& GetArme();
        ///@brief recupere la direction du personnage
        bool getDirection() const;
        ///@brief change la direction du personnage 0 pour gauche et 1 pour droite
        void setDirection(bool a);
        ///@brief gere le saut du personnage
        void saut(const Terrain& t);
        ///@brief permet le deplacement vers la  gauche du personnage
        void DeplGauche(const Terrain& t) ;
        ///@brief permet le deplacement vers la droite du personnage
        void DeplDroite(const Terrain& t) ;
        ///@brief permet de faire tomber le personnage au sol s'il n'est pas sur une plateforme
        void gravite(const Terrain& t);
        ///@brief gere la capacité de tir du personnage
        void tir(const Terrain& t);
        ///@brief met a jour l'arme du personnage en mettant a jour ses projectiles en fonction du type de l'arme
        void UpdateArme(Arme& a, const Terrain& t);
        ///@brief fais le test de regression
        void testRegressionPers();
```

Classe Ennemi

```
class Ennemi {
    public:
        ///@brief constructeur de la classe ennemi
        Ennemi();
        /**
         *@brief constructeur de la classe ennemi avec les positions la direction de l'ennemis et les dégats qu'il inflige
         *@param x coordonnée x de la position de l'ennemi
         *@param y coordonnée y de la position de l'ennemi
         *@param vie points de vie de l'ennemi
         *@param direction direction dans laquelle l'ennemi est
         *@param degat degats qu'inlfige l'ennemi*/
        Ennemi(int x, int y, int vie, bool direction, int degat);
        ///@brief destructeur de la classe ennemi
       ~Ennemi();
       ///@brief recupere la position x de l'ennemi
        int GetPosEnnemi_X() const;
       ///@brief recupere la position y de l'ennemi
        int GetPosEnnemi_Y() const;
        /**
         * @brief change les composantes x et y de la position de l'ennemi
         * @param x coordonnée x de la position de l'ennemi
         * @param y coordonnée y de la position de l'ennemi
         * */
        void SetPosEnnemi(int x, int y);
        ///@brief recupere les points de vie de l'ennemi
        int GetVieEnnemi()const ;
        /**
         * @brief change les points de vie de l'ennemi
         * @param v points de vie de l'ennemi
        */
        void SetVieEnnemi(int v);
```

```
        Arme& GetArme();
        ///@brief recupere la direction de l'ennemi
        bool getDirection() const;
        /**
         * @brief change la direction de l'ennemi
         * @param a direction dans laquelle on veut que l'ennemi se tourne
         * */
        void setDirection(bool a);
        /**
         * @brief deplace l'ennemi vers la droite
         * @param t terrain dans lequel l'ennemi se déplace
         * */
        void Deplacement_droite(const Terrain &t);
        /**
         * @brief deplace l'ennemi vers la gauche
         * @param t terrain dans lequel l'ennemi se déplace
         * */
        void Deplacement_gauche(const Terrain &t);
        /**
         * @brief ramene l'ennemi vers le sol s'il n'est pas sur une plateforme
         * @param t terrain dans lequel l'ennemi se déplace
         * */
        void gravite(const Terrain& t);
        /**
         * @brief fais tirer l'ennemi
         * @param t terrain dans lequel les projectiles de l'arme se déplace
         * */
        void tir(const Terrain& t);
        /**
         * @brief met tout les projectiles de l'arme a jour
         * @param a arme de l'ennemi
         * @param t terrain ou se trouve l'ennemi
         * */
        void UpdateArme(Arme& a, const Terrain& t);
        ///@brief fais le test de regression
        void testRegressionEnnemi();
```

Classe Jeu

```
class Jeu
{
public:
    ///@brief constructeur de la classe jeu
    Jeu();
    /**
     * @brief gere les actions du personnage en fonctions des différentes touches utilisés
     * @param ch caractere designant l'action que l'on veut réaliser
     * */
    void ActionClavier(const char ch);
    /**
     * @brief fais redescendre le personage si il saute ou si il n'est plus sur une plateforme
     * @param ch caracteres soit g (pour gauche), soit d (pour droite) pour changer de direction lors de la déscente
     * */
    void gravite(const char ch);
    /**
     * @brief permet l'affichage du terrain en mode texte
     * @param filename nom du ficher terrain a charger pour l'affichage
     * */
    void afficher_Terraintxt(const string& filename);
    ///@brief permet de recuperer un terrain
    Terrain& getTerrain();
    ///@brief permet de recupérer un personnage
    Personnage& getPersonnage();
    ///@brief permet de recuperer le tableau d'ennemis
    vector<Ennemi> getEnnemis() const; // Retourne une référence constante sur le tableau d'ennemis
    ///@brief permet de mettre a jour l'arme du personnage et ceux des ennemis
    void UpdateArmePerso_Ennemi();
    ///@brief permet de gerer les collisions des projectiles qui touchent les ennemis
    void Collisions_proj_ennemi();
    ///@brief permet de gerer les collisions des qui touchent le personnage
    void Collisions_proj_personnage();
    ///@brief gere les tirs des ennemis en fonction de la position du personnage
    void Tir_Automatique();
    ///@brief permet de faire apparaitre un fusil que l'on peut eventuelement récuperer sur la map
    void Apparition_fusil();
    ///@brief permet d'ouvrir une nouvelle partie
    void CreerNouveauJeu(int nombreEnnemis,int degats);
    ///@brief permet de changer de niveau
    void Niveau_Suivant(int niveau);
    ///@brief fais le test de regression
    void testRegressionJeu();
    ///@brief booleen indiquant si la partie est finie
    bool GAME_OVER;
    ///@brief booleen indiquant si le joueur a gagné la partie
    bool WIN;
```

Conclusion