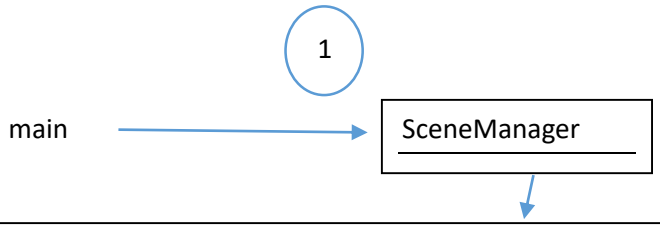
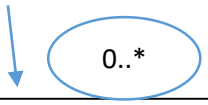


EntityManager

```
p2SString     folder;
p2SString     texture_path;
SDL_Texture*  texture = nullptr;
p2List<Entity*> entities;
Player*       player = nullptr;
uint          gargoyle_count;
uint          skeleton_count;
```

```
Entity* CreateEntity(entityType type);
bool DestroyEntity(Entity* entity);
void OnCollision(Collider* collider1, Collider* collider2);
bool Awake(pugi::xml_node & config);
bool Start();
bool Restart();
bool PreUpdate();
bool Update(float dt);
bool PostUpdate();
bool CleanUp();
bool Save(pugi::xml_node&) const;
bool Load(pugi::xml_node&);
```



Assumptions: Not all variables in player are included, only the most importants. Fx are not necessary no include it for example.

Entity

```
entityType type;
fPoint speed, position, max_speed, acceleration;
Collider* collider;
Animation* current_animation;
SDL_Texture* texture;

enum class entityType{PLAYER, FLYING_ENEMY, LAND_ENEMY, COIN, NO_TYPE,};
enum MOVEMENT {IDLE, RIGHT, LEFT, UP, DOWN, DEAD, LEFT_HIT, RIGHT_HIT};
enum STATE {FLOOR, AIR, DEATH, WIN };
STATE current_state;
MOVEMENT last_movement;
MOVEMENT current_movement;
```

```
virtual bool Awake(pugi::xml_node & config) { return true; };
virtual bool Start(uint i) { return true; };
virtual bool Restart() { return true; };
virtual bool PreUpdate() { return true; };
virtual bool Update(float dt) { return true; };
virtual bool PostUpdate() { return true; };

virtual bool CleanUp() { return true; };
virtual bool Save(pugi::xml_node& file) const { return true; };
virtual bool Load(pugi::xml_node& file) { return true; };
void          AddFX(const int channel, const int repeat) const;
Bool          LoadAnimation(pugi::xml_node &node, Animation &anim);
virtual void OnCollision(Collider*, Collider*) {};
void          AddCollider();

virtual fPoint GetPosition();
virtual void SetPosition(const float &x, const float &y);
```

Player

entityType::PLAYER

```
Animation     idlefire, runfire, jumpfire, deadfire,
hitfire;
Animation     idleice, runice, jumpice, deadice, hitice;
Animation     godmode_anim;
int    jump_speed, hit_speed;
fPoint lastPosition;
GODMOVE current_godmove;
STATE  current_state;
ELEMENT current_element;
bool   godmode;
bool   visibility;
int score, lifes;
```

```
Player(entityType type);
~Player();
bool Awake(pugi::xml_node& config);
bool Start(uint i);
bool PreUpdate();
bool Update(float dt);
bool PostUpdate();
bool CleanUp();
bool Load(pugi::xml_node&);
bool Save(pugi::xml_node&) const;
fPoint GetPosition() const;
void   SetPosition(const float &x, const float &y);
void OnCollision(Collider* collider1);
void   AddColliderPlayer();
void   Restart(ELEMENT element);
bool   isDead();
void   AddFX(const int channel, const int repeat) const;
bool LoadAnimation(pugi::xml_node &node, Animation &anim);
```

OfficerSkeleton

entityType::LAND_ENEMY

```
SDL_Texture * skeleton_tex;
p2SString   skeleton_texture;
Animation*    current_animation;
Animation idle;
Animation walk;
Animation dead;
```

```
OfficerSkeleton();
OfficerSkeleton(entityType type);
~OfficerSkeleton();
Bool Awake(pugi::xml_node & config);
bool Start(uint i);
bool Restart(uint i);
bool PreUpdate();
bool Update(float dt);
bool PostUpdate();
void OnCollision(Collider* collider1);
void Walk(const p2DynArray<iPoint>
*path);
bool Load(pugi::xml_node&);
bool Save(pugi::xml_node&) const;
bool LoadAnimation(pugi::xml_node
&node, Animation &anim);
bool CleanUp();
```

JrGargoyle

entityType::FLYING_ENEMY

```
SDL_Texture* gargoyle_tex;
p2SString   gargoyle_texture;
Animation*    current_animation;
Animation idle;
Animation fly;
Animation dead;
```

```
JrGargoyle();
JrGargoyle(entityType type);
~JrGargoyle();
bool Awake(pugi::xml_node & config);
bool Start(uint i);
bool Restart(uint i);
bool PreUpdate();
bool Update(float dt);
bool PostUpdate();
void OnCollision(Collider* collider1);
void Fly(const p2DynArray<iPoint> *path);
bool Load(pugi::xml_node&);
bool Save(pugi::xml_node&) const;
bool LoadAnimation(pugi::xml_node &node, Animation &anim);
bool CleanUp();
```

Coin

entityType::COIN

```
SDL_Texture* coin_tex = nullptr;
p2SString   coin_texture, coin_fx_name;
Animation     flip;
```

```
Coin(entityType type);
~Coin();
bool Awake(pugi::xml_node & config);
bool Start(uint i);
bool Restart(uint i);
bool PreUpdate();
bool Update(float dt);
bool PostUpdate();
void OnCollision(Collider* collider1);
bool Load(pugi::xml_node&);
bool Save(pugi::xml_node&) const;
bool LoadAnimation(pugi::xml_node
&node, Animation &anim);
bool CleanUp();
```