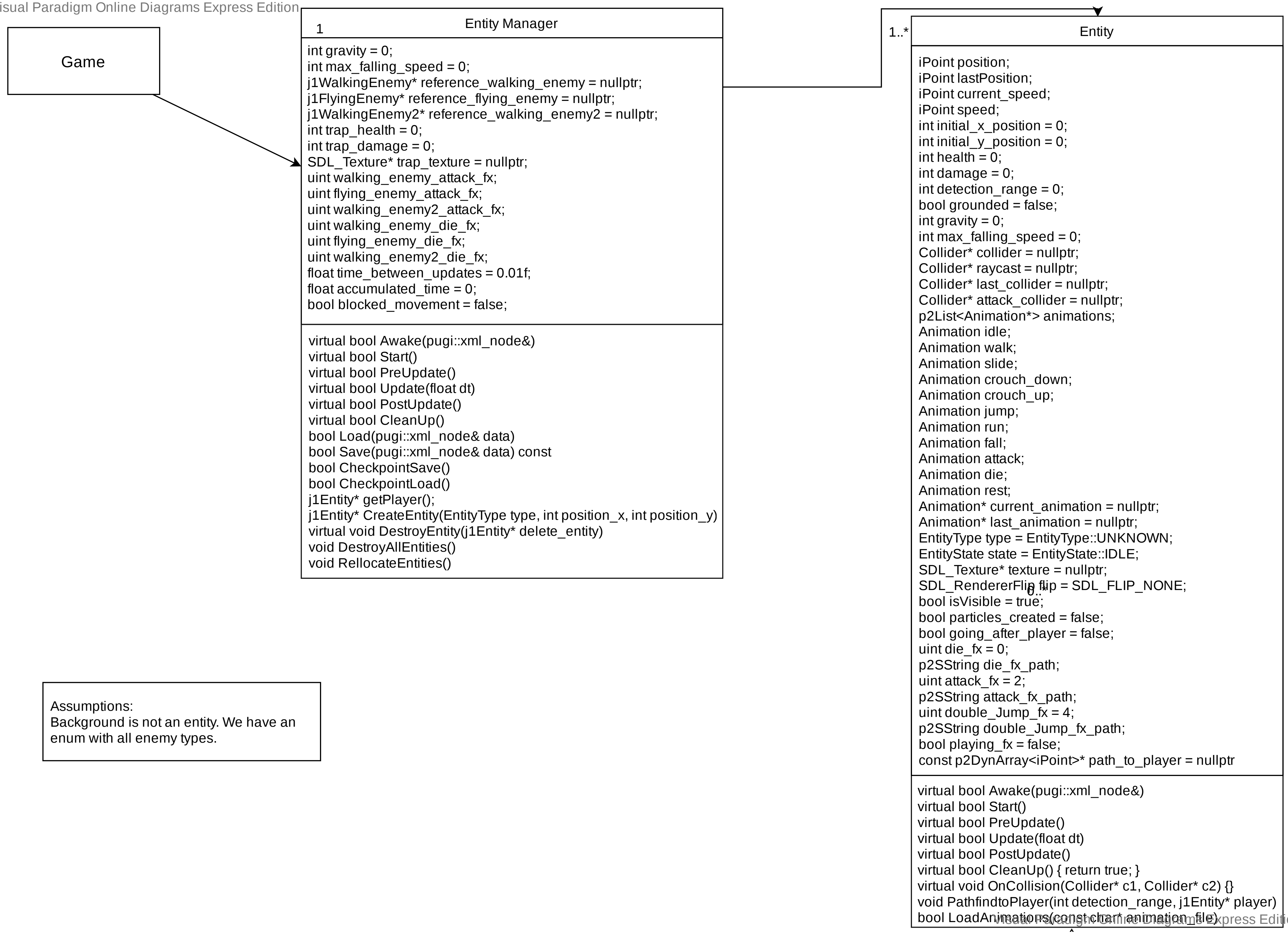


Entity Manager

```
int gravity = 0;
int max_falling_speed = 0;
j1WalkingEnemy* reference_walking_enemy = nullptr;
j1FlyingEnemy* reference_flying_enemy = nullptr;
j1WalkingEnemy2* reference_walking_enemy2 = nullptr;
int trap_health = 0;
int trap_damage = 0;
SDL_Texture* trap_texture = nullptr;
uint walking_enemy_attack_fx;
uint flying_enemy_attack_fx;
uint walking_enemy2_attack_fx;
uint walking_enemy_die_fx;
uint flying_enemy_die_fx;
uint walking_enemy2_die_fx;
float time_between_updates = 0.01f;
float accumulated_time = 0;
bool blocked_movement = false;
```

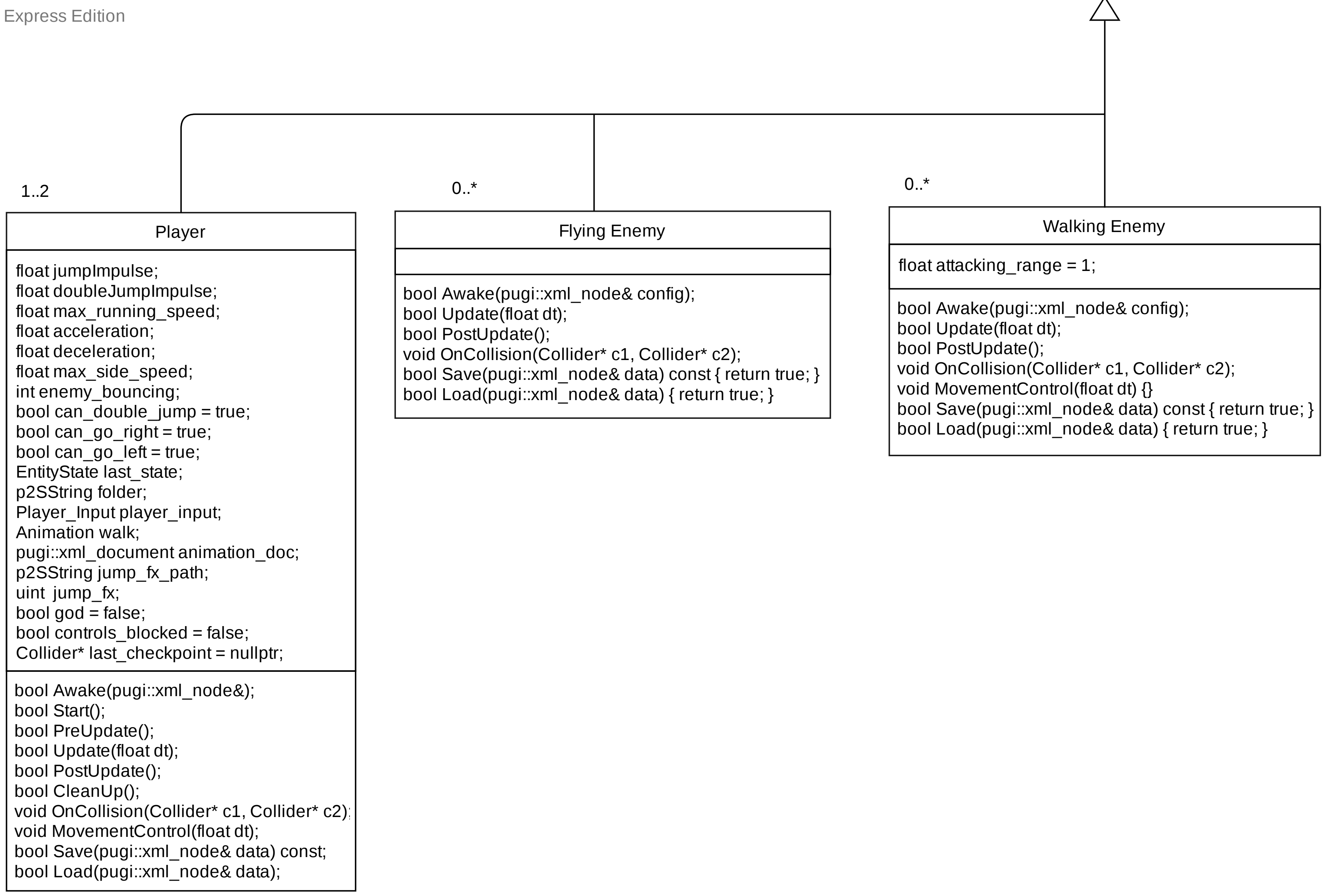
```
virtual bool Awake(pugi::xml_node&)
virtual bool Start()
virtual bool PreUpdate()
virtual bool Update(float dt)
virtual bool PostUpdate()
virtual bool CleanUp()
bool Load(pugi::xml_node& data)
bool Save(pugi::xml_node& data) const
bool CheckpointSave()
bool CheckpointLoad()
j1Entity* getPlayer();
j1Entity* CreateEntity(EntityType type, int position_x, int position_y)
virtual void DestroyEntity(j1Entity* delete_entity)
void DestroyAllEntities()
void RellocateEntities()
```

Entity

```
iPoint position;
iPoint lastPosition;
iPoint current_speed;
iPoint speed;
int initial_x_position = 0;
int initial_y_position = 0;
int health = 0;
int damage = 0;
int detection_range = 0;
bool grounded = false;
int gravity = 0;
int max_falling_speed = 0;
Collider* collider = nullptr;
Collider* raycast = nullptr;
Collider* last_collider = nullptr;
Collider* attack_collider = nullptr;
p2List<Animation*> animations;
Animation idle;
Animation walk;
Animation slide;
Animation crouch_down;
Animation crouch_up;
Animation jump;
Animation run;
Animation fall;
Animation attack;
Animation die;
Animation rest;
Animation* current_animation = nullptr;
Animation* last_animation = nullptr;
EntityType type = EntityType::UNKNOWN;
EntityState state = EntityState::IDLE;
SDL_Texture* texture = nullptr;
SDL_RendererFlip flip = SDL_FLIP_NONE;
bool isVisible = true;
bool particles_created = false;
bool going_after_player = false;
uint die_fx = 0;
p2SString die_fx_path;
uint attack_fx = 2;
p2SString attack_fx_path;
uint double_Jump_fx = 4;
p2SString double_Jump_fx_path;
bool playing_fx = false;
const p2DynArray<iPoint>* path_to_player = nullptr
```

```
virtual bool Awake(pugi::xml_node&)
virtual bool Start()
virtual bool PreUpdate()
virtual bool Update(float dt)
virtual bool PostUpdate()
virtual bool CleanUp() { return true; }
virtual void OnCollision(Collider* c1, Collider* c2) {}
void PathfindtoPlayer(int detection_range, j1Entity* player)
bool LoadAnimations(const char* animation_file)
```

Assumptions:
Background is not an entity. We have an enum with all enemy types.



Player

```
float jumpImpulse;
float doubleJumpImpulse;
float max_running_speed;
float acceleration;
float deceleration;
float max_side_speed;
int enemy_bouncing;
bool can_double_jump = true;
bool can_go_right = true;
bool can_go_left = true;
EntityState last_state;
p2SString folder;
Player_Input player_input;
Animation walk;
pugi::xml_document animation_doc;
p2SString jump_fx_path;
uint  jump_fx;
bool god = false;
bool controls_blocked = false;
Collider* last_checkpoint = nullptr;
```

```
bool Awake(pugi::xml_node&);
bool Start();
bool PreUpdate();
bool Update(float dt);
bool PostUpdate();
bool CleanUp();
void OnCollision(Collider* c1, Collider* c2);
void MovementControl(float dt);
bool Save(pugi::xml_node& data) const;
bool Load(pugi::xml_node& data);
```

Flying Enemy

```
bool Awake(pugi::xml_node& config);
bool Update(float dt);
bool PostUpdate();
void OnCollision(Collider* c1, Collider* c2);
bool Save(pugi::xml_node& data) const { return true; }
bool Load(pugi::xml_node& data) { return true; }
```

Walking Enemy

```
float attacking_range = 1;
```

```
bool Awake(pugi::xml_node& config);
bool Update(float dt);
bool PostUpdate();
void OnCollision(Collider* c1, Collider* c2);
void MovementControl(float dt) {}
bool Save(pugi::xml_node& data) const { return true; }
bool Load(pugi::xml_node& data) { return true; }
```