Gravity

Group 10

Daniel Walz

Jesper Ekhall

Lukas Kalinski

Fredrik Nordh

Alexander Nyberg

5.5 Detailed Design (& 5.6 Package Diagram)

For more clarity and better overview of the design we have decided to merge the package diagrams in the detailed design, see figures 1 to 23.

5.5.1 Audio

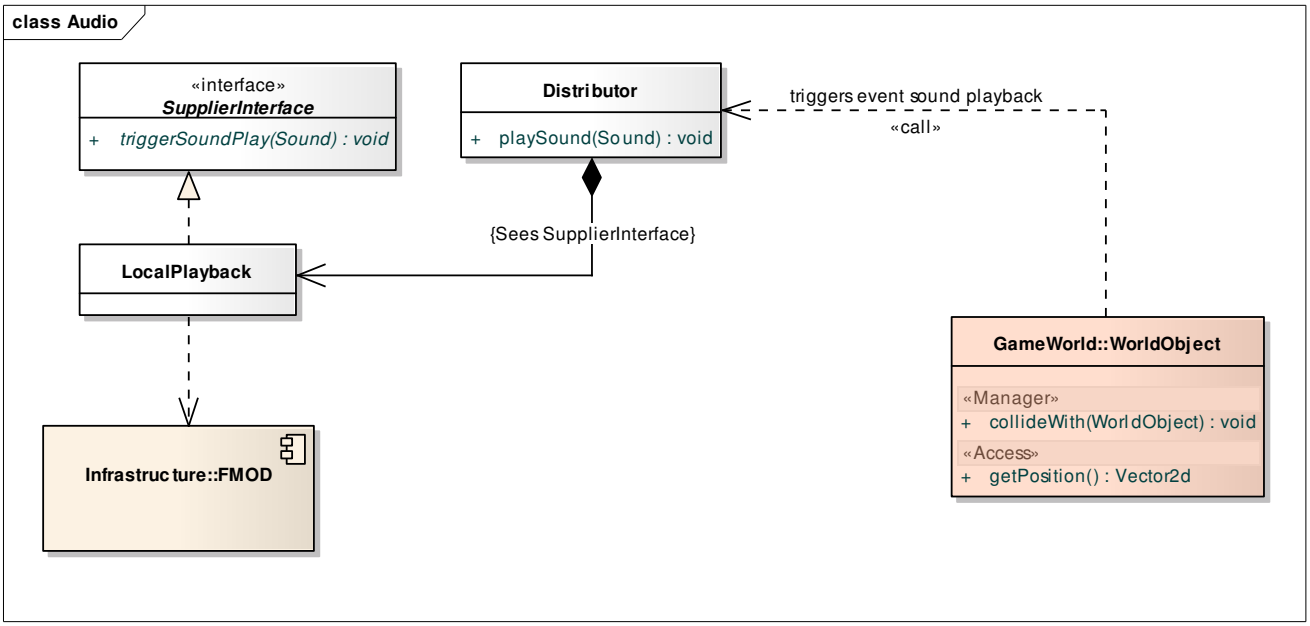

Figure 1 : Audio

5.5.1.1 Audio::Distributor

public Class: Responsible for distributing the input sound play orders to all suppliers implementing the SupplierInterface.

Audio::Distributor Connections

Connector	Source	Target	Notes
Aggregation source < target	LocalPlayback unordered, none	Distributor unordered, none	*Constraints* Invariant: Sees SupplierInterface
Dependency «use» source > target	GamePlayState Client Element	Distributor Server Element	
Dependency «use» source > target	State Client Element	Distributor Server Element	
Dependency triggers event sound playback «call» source > target	WorldObject Client Element	Distributor Server Element	

Audio::Distributor Methods

Method	Type	Notes
playSound (*Sound*)	public: *void*	param: sound [Sound - in] Distributes the sound play order to all registered receivers.

5.5.1.2 Audio::LocalPlayback

public Class

Implements: SupplierInterface. : Plays sounds locally on the computer that is running the game engine.

Audio::LocalPlayback Connections

Connector	Source	Target	Notes
Aggregation source < target	LocalPlayback unordered, none	Distributor unordered, none	*Constraints* Invariant: Sees SupplierInterface
Dependency source > target	LocalPlayback Client Element	FMOD Server Element	
Realisation source > target	LocalPlayback Child	SupplierInterface Parent	

5.5.1.3 Audio::SupplierInterface

public abstract «interface» Interface: Defines an interface required for supplying sound playback orders.

Audio::SupplierInterface Connections

Connector	Source	Target	Notes
Realisation source > target	LocalPlayback Child	SupplierInterface Parent	

Audio::SupplierInterface Interfaces

Method	Type	Notes
triggerSoundPlay (*Sound*)	public abstract: *void*	param: sound [Sound - in] Triggers a sound play somewhere. The sound will be played until it ends. Post-condition: Functional A sound is played somewhere

5.5.2 Controller

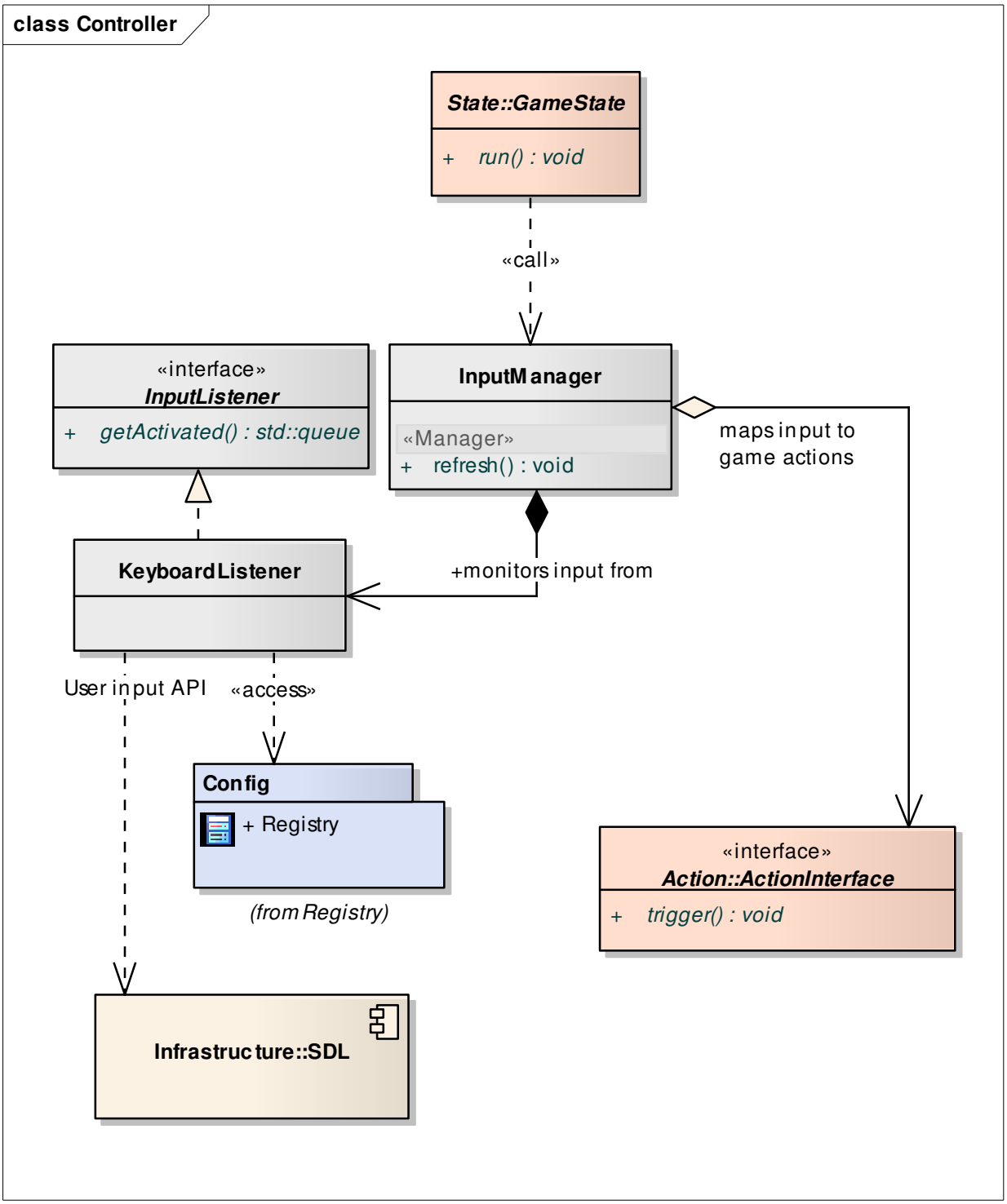

Figure 2 : Controller

5.5.2.1 Controller::InputManager

public Class: Monitors the statuses of all available input listeners and triggers corresponding game actions, according to the translation map(s) found in the Registry::Config package.

Controller::InputManager Connections

Connector	Source	Target	Notes
Aggregation source < target	KeyboardListener unordered, none	InputManager +monitors input from unordered, none	
Aggregation maps input to game actions source < target	ActionInterface unordered, none	InputManager unordered, none	
Dependency «call» source > target	GameState Client Element	InputManager Server Element	

Controller::InputManager Methods

Method	Type	Notes
refresh ()	«Manager» public: *void*	Polls available input listeners and takes appropriate action according to the result.

5.5.2.2 Controller::KeyboardListener

public Class

Implements: InputListener. : Takes keyboard input from SDL.

Controller::KeyboardListener Connections

Connector	Source	Target	Notes
Aggregation source < target	KeyboardListener unordered, none	InputManager +monitors input from unordered, none	
Dependency «access» source > target	KeyboardListener Client Element	Config Server Element	
Dependency User input API source > target	KeyboardListener Client Element	SDL Server Element	
Realisation source > target	KeyboardListener Child	InputListener Parent	

5.5.2.3 Controller::InputListener

public abstract «interface» Interface: Provides an interface telling whether a control/input is active or not.

Controller::InputListener Connections

Connector	Source	Target	Notes
Realisation source > target	KeyboardListener Child	InputListener Parent	

Controller::InputListener Interfaces

Method	Type	Notes
getActivated ()	public: *std::queue*	Returns a queue of codes of the currently pressed keys. Post-condition: Functional The codes of the keys that were pressed before the method call are returned.

5.5.3 Game

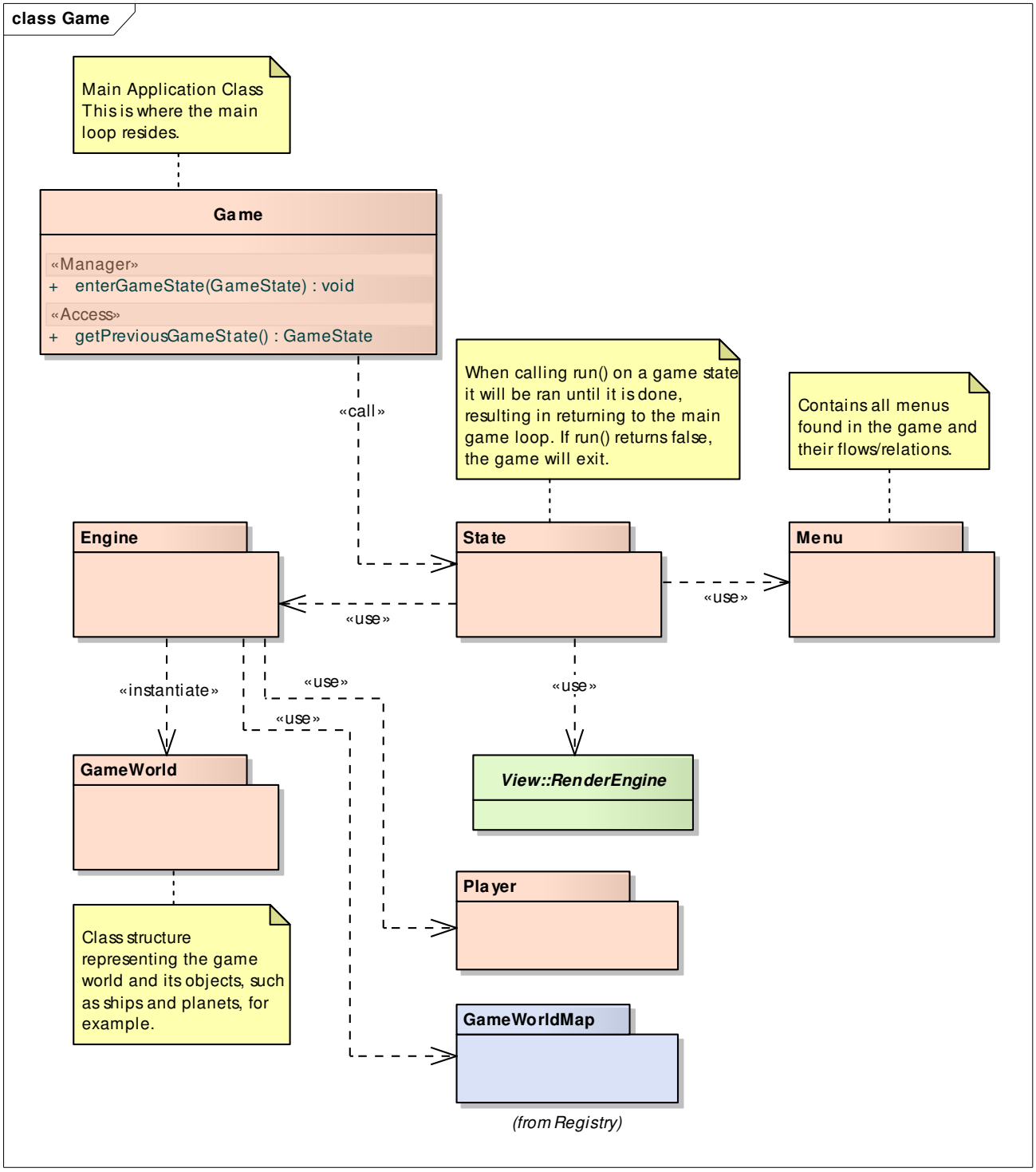

Figure 3 : Game

User

User Connections

Connector	Source	Target	Notes
Sequence startGameSession source > target	User Client Element	MenuState Server Element	
Sequence setGameControls source > target	User Client Element	MenuState Server Element	
Sequence start source > target	User Client Element	Game Server Element	
Sequence initControlSetLoop source > target	User Client Element	Menu Server Element	
Sequence start source > target	User Client Element	Game Server Element	
Sequence source > target	Menu Client Element	User Server Element	
Sequence startGameSession source > target	User Client Element	MenuState Server Element	

5.5.3.1 Game::Game

public Class: Runs the main loop and forwards control to other game states.

Game::Game Connections

Connector	Source	Target	Notes
Dependency «call» source > target	Game Client Element	State Server Element	
Sequence switchGameState source > target	MenuState Client Element	Game Server Element	
Sequence start source > target	User Client Element	Game Server Element	
Sequence source > target	Game Client Element	MenuState Server Element	
Sequence init source > target	Game Client Element	MenuState Server Element	
Sequence start source > target	User Client Element	Game Server Element	

Game::Game Methods

Method	Type	Notes
enterGameState (*GameState*)	«Manager» public: *void*	param: state [GameState - in] Enters a new game state. Post-condition: Functional A new game state has been entered.
getPreviousGameState ()	«Access» public: *GameState*	Returns the underlying state of the current state. Pre-condition: Functional There is a previous game state.

5.5.4 Engine

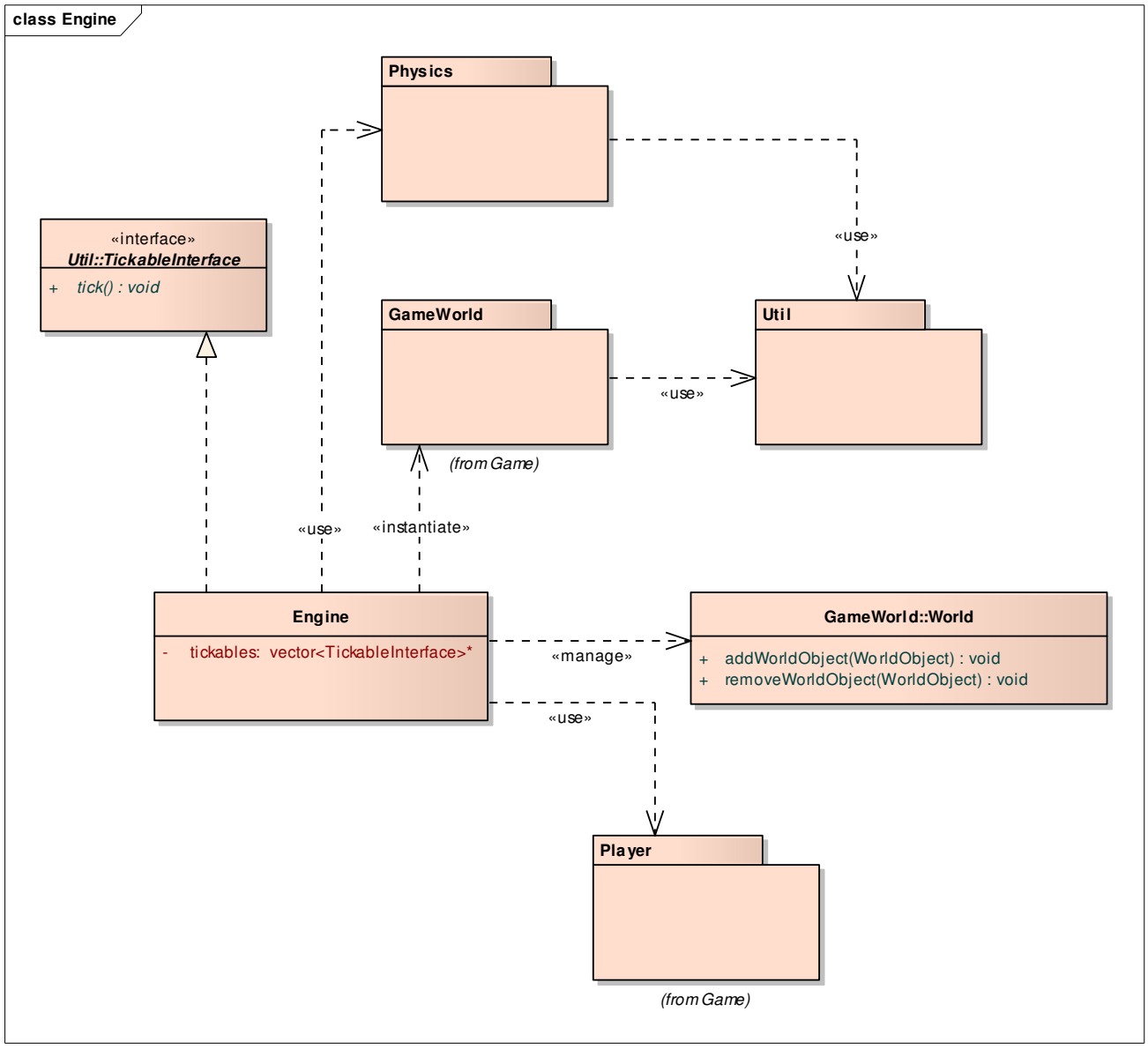

Figure 4 : Engine

5.5.4.1 Engine::Engine

public Class

Implements: TickableInterface. : Responsible for maintaining the game world, applying physical constraints to it and to trigger its visualization and event sounds.

Engine::Engine Connections

Connector	Source	Target	Notes
Dependency «use» source > target	Engine Client Element	Physics Server Element	
Dependency «manage» source > target	Engine Client Element	World Server Element	
Dependency «use» source > target	GamePlayState Client Element	Engine Server Element	
Dependency «use» source > target	Engine Client Element	Player Server Element	
Dependency «instantiate» source > target	Engine Client Element	GameWorld Server Element	
Realisation source > target	Engine Child	TickableInterface Parent	

5.5.5 Physics

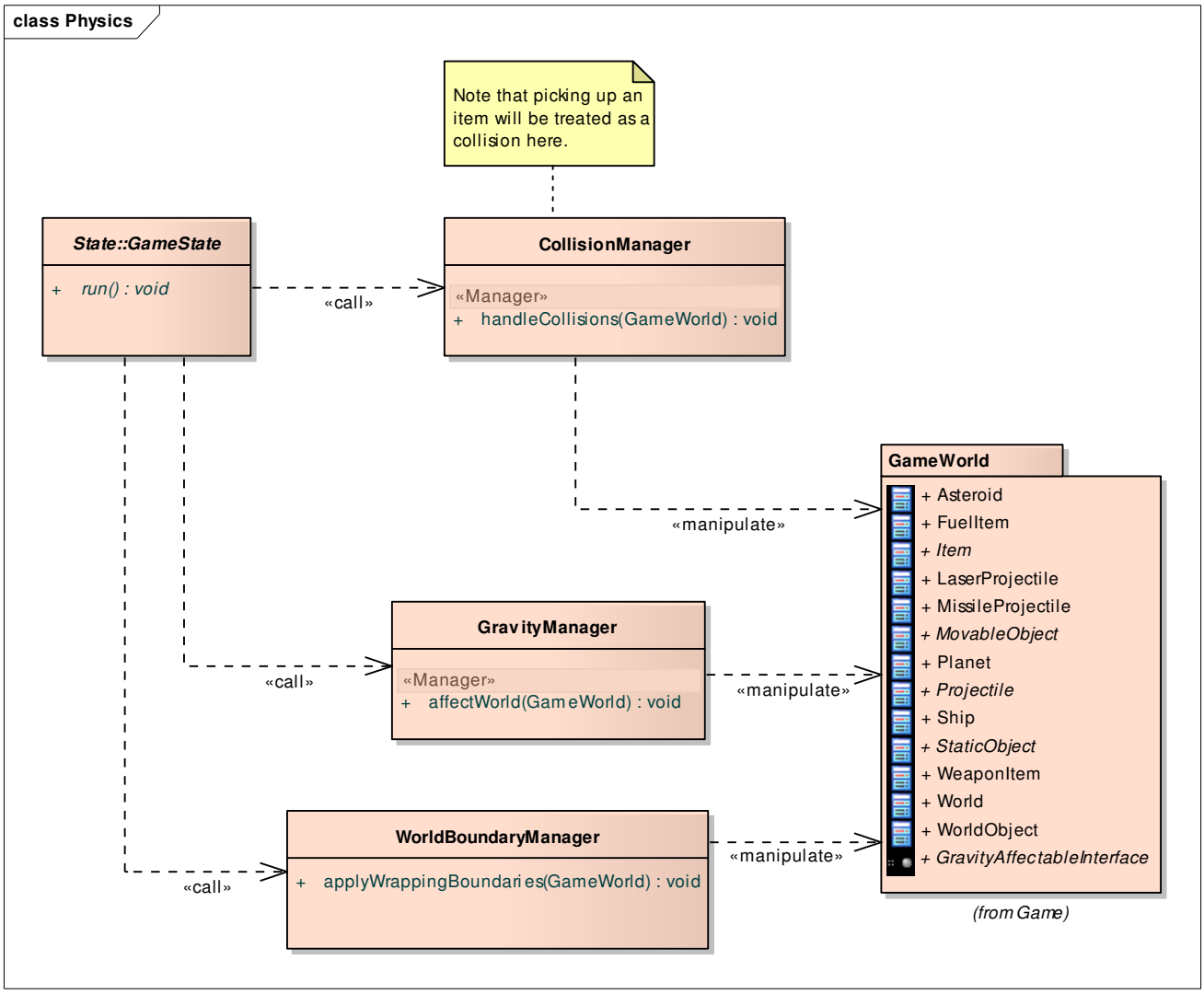

Figure 5 : Physics

5.5.5.1 Physics::CollisionManager

public Class: Responsible for detecting and managing collisions between objects in the game world, provided a game world instance.

Physics::CollisionManager Connections

Connector	Source	Target	Notes
Dependency «call» source > target	GameState Client Element	CollisionManager Server Element	
Dependency «manipulate» source > target	CollisionManager Client Element	GameWorld Server Element	

Physics::CollisionManager Methods

Method	Type	Notes
handleCollisions (*GameWorld*)	«Manager» public: *void*	param: world [GameWorld - in] Takes a world and handles possible collisions within it.

		Post-condition: Functional Collisions handled - Each object that participated in a collision has had a chance to take appropriate action.

5.5.5.2 Physics::GravityManager

public Class: Responsible for calculating and applying gravity affections for each gravity-affectable game world object.

Physics::GravityManager Connections

Connector	Source	Target	Notes
Dependency «call» source > target	GameState Client Element	GravityManager Server Element	
Dependency «manipulate» source > target	GravityManager Client Element	GameWorld Server Element	

Physics::GravityManager Methods

Method	Type	Notes
affectWorld (*GameWorld*)	«Manager» public: *void*	param: wo [GameWorld - in] Affects all movable world objects with the gravities of the present planets. Post-condition: Functional Movable object movements updated - The objects found in the range of one or more planetary gravity fields are affected with the corresponding gravity vectors.

5.5.5.3 Physics::WorldBoundaryManager

Physics::WorldBoundaryManager Connections

Connector	Source	Target	Notes
Dependency «call» source > target	GameState Client Element	WorldBoundaryManager Server Element	
Dependency «manipulate» source > target	WorldBoundaryManager Client Element	GameWorld Server Element	

Physics::WorldBoundaryManager Methods

Method	Type	Notes
applyWrappingBoundaries (*GameWorld*)	public: *void*	param: world [GameWorld - in] Causes all world objects that cross the world boundary to appear on the opposite side of the world map. Post-condition: Functional World objects are within world boundaries - No world object is placed outside the world boundaries.

5.5.6 Util

Contains common utilities, such as coordinate representations etc.

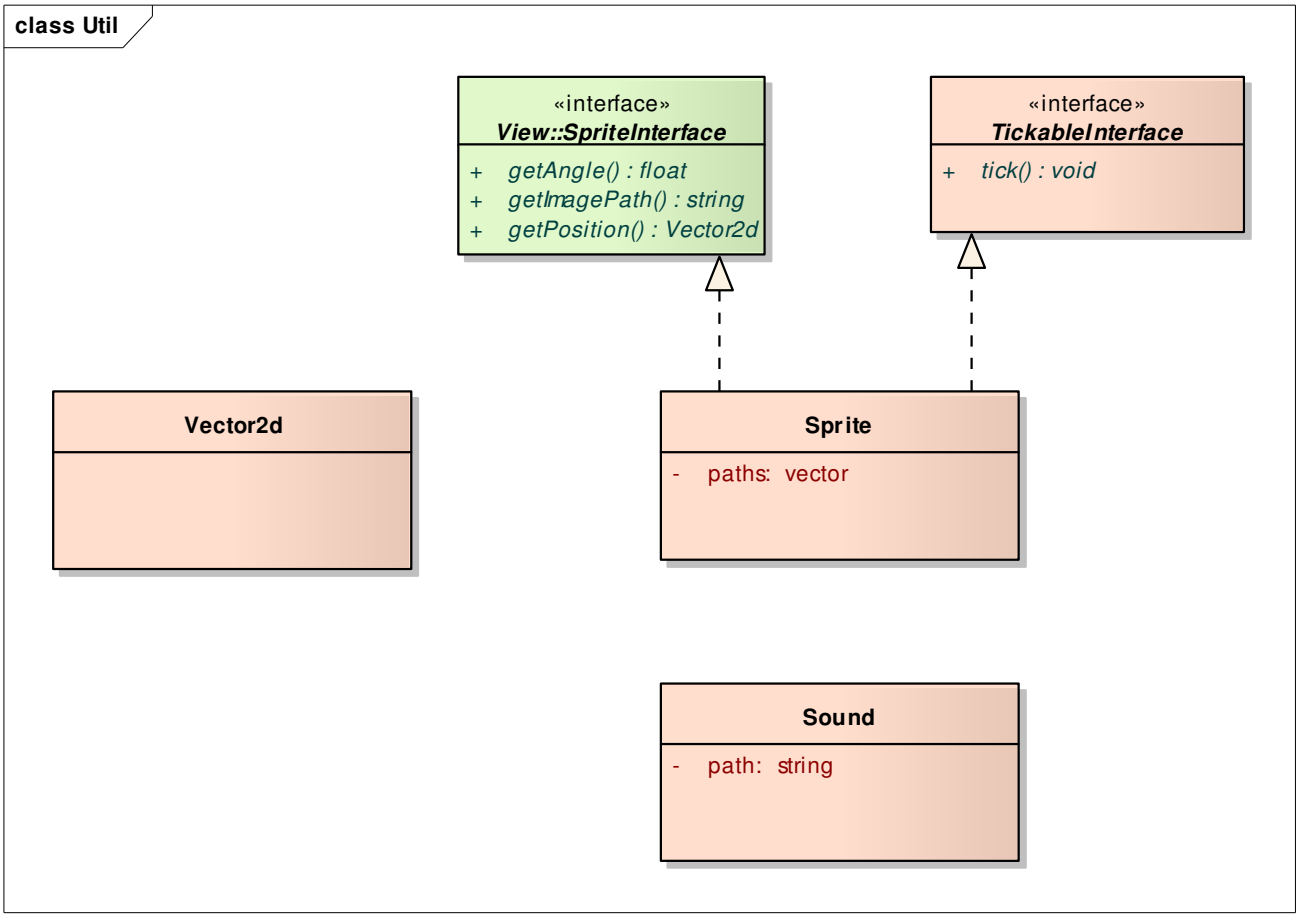

Figure 6 : Util

5.5.6.1 Util::Sound

public Class: Abstract representation of a sound. Will not contain the actual sound data but rather data about where the sound may be found.

Util::Sound Connections

Connector	Source	Target	Notes
Aggregation destruction sound source < target	Sound 1, unordered, none	Ship 1, unordered, none	
Aggregation fire sound source < target	Sound 1, unordered, none	Projectile 1, unordered, none	
Aggregation collision sound source < target	Sound 0..1, unordered, none	WorldObject unordered, none	

5.5.6.2 Util::Sprite

public Class

Implements: SpriteInterface, TickableInterface. : A 2D image or animation that is used in the graphical presentation of the game. Will not contain the actual image data but rather data about where the image(s) may be found. If the sprite contains more than one image, then it will be considered an animation, which will be synchronized using tick().

Util::Sprite Connections

Connector	Source	Target	Notes
Aggregation graphical representation source < target	Sprite 1, unordered, none	WorldObject unordered, none	
Aggregation source < target	Sprite unordered, none	Item unordered, none	
Generalization source > target	StatusSprite Child	Sprite Parent	
Realisation source > target	Sprite Child	TickableInterface Parent	
Realisation source > target	Sprite Child	SpriteInterface Parent	

5.5.6.3 Util::Vector2d

public Class: Defines a vector in 2D space by combining a coordinate and an angle.

Util::Vector2d Connections

Connector	Source	Target	Notes
Aggregation movement and pointing vectors source < target	Vector2d 2, unordered, none	MovableObject 1, unordered, none	One vector defines the movement and the other the direction that the object is pointing in.

5.5.6.4 Util::TickableInterface

public abstract «interface» Interface: When a class object need to receive a tick, it has to implement this interface.

Util::TickableInterface Connections

Connector	Source	Target	Notes
Realisation source > target	MovableObject Child	TickableInterface Parent	
Realisation source > target	Sprite Child	TickableInterface Parent	
Realisation source > target	Engine Child	TickableInterface Parent	
Realisation source > target	Menu Child	TickableInterface Parent	

Util::TickableInterface Interfaces

Method	Type	Notes
tick ()	public abstract: *void*	

5.5.7 GameWorld

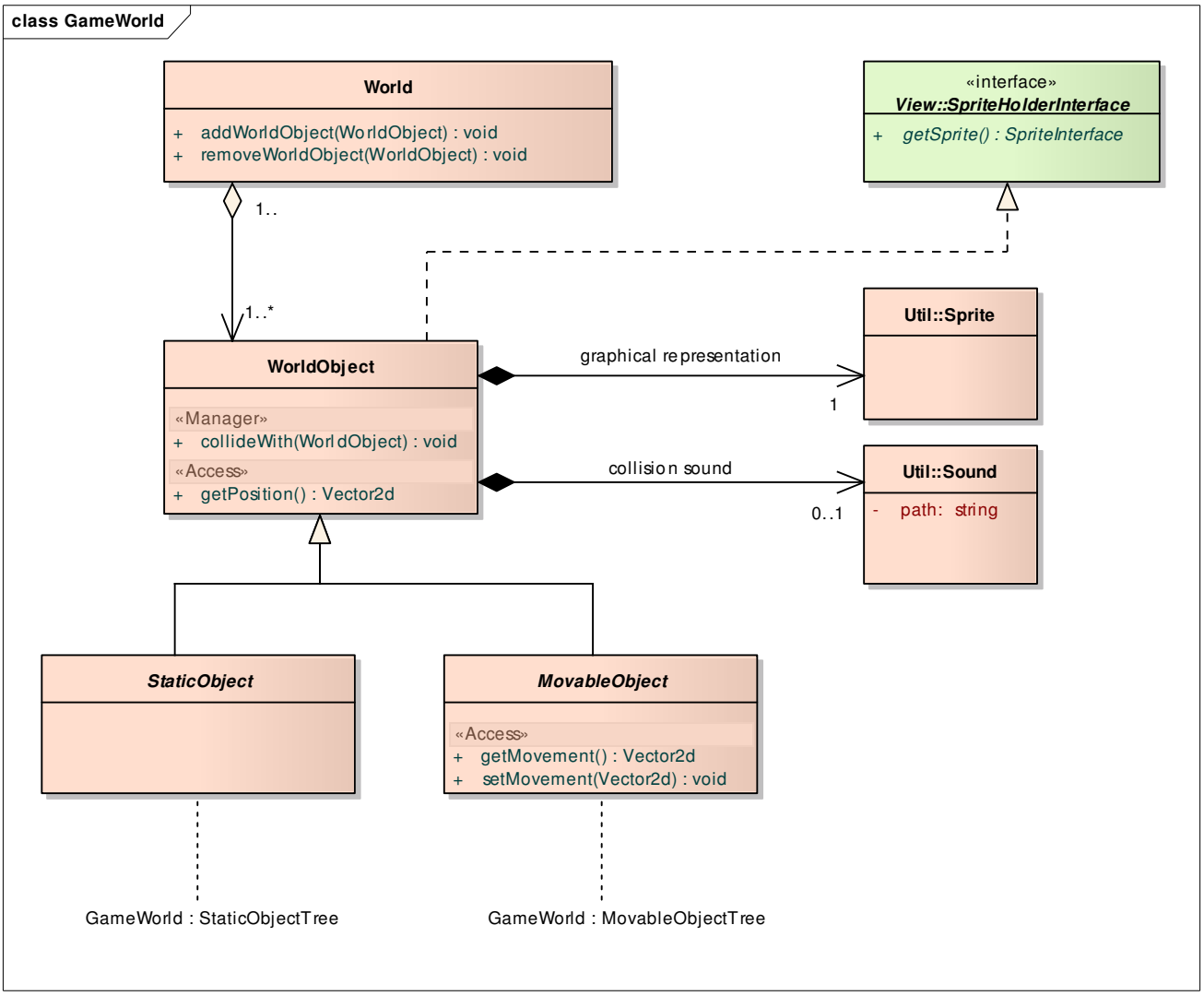

Figure 7 : GameWorld

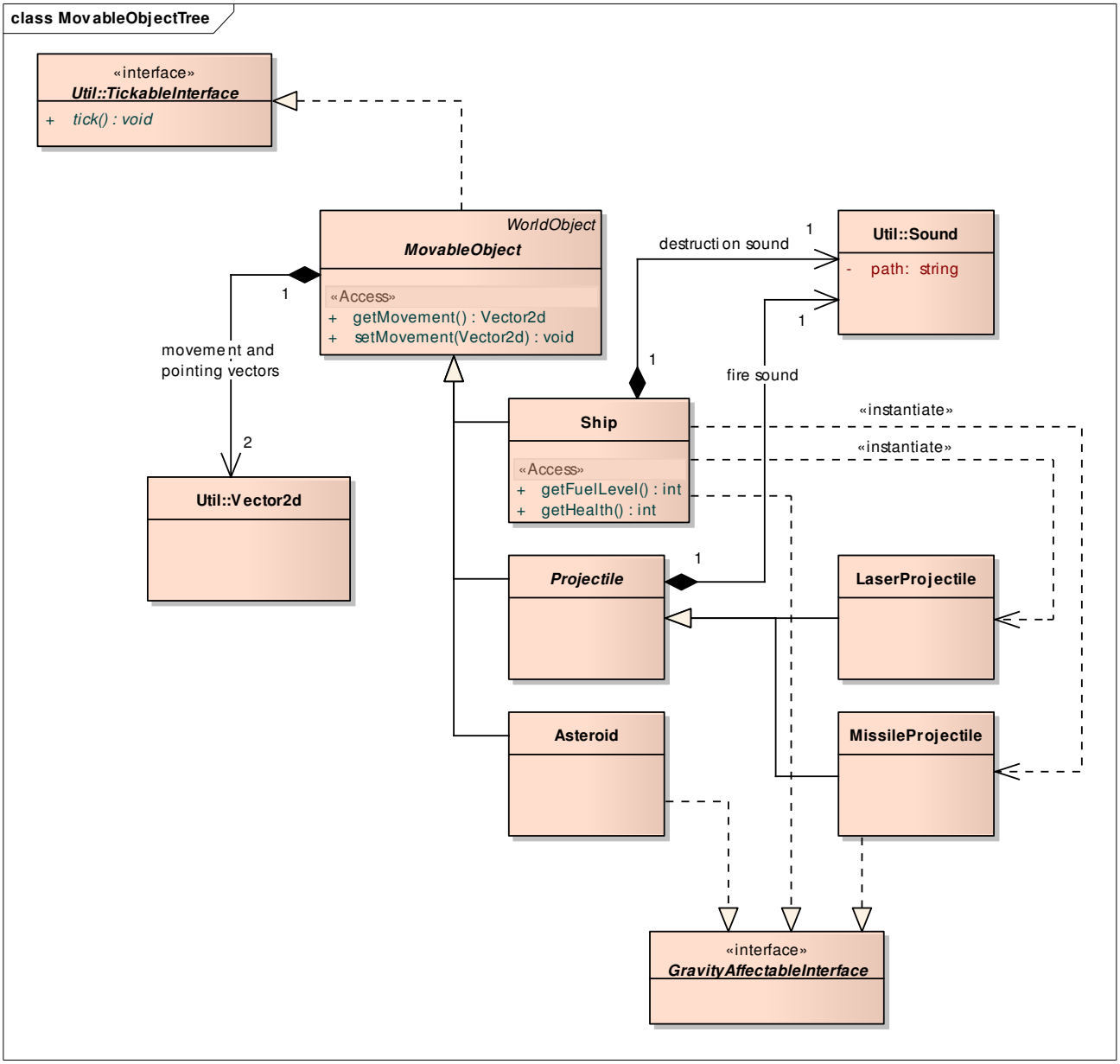

Figure 8 : MovableObjectTree

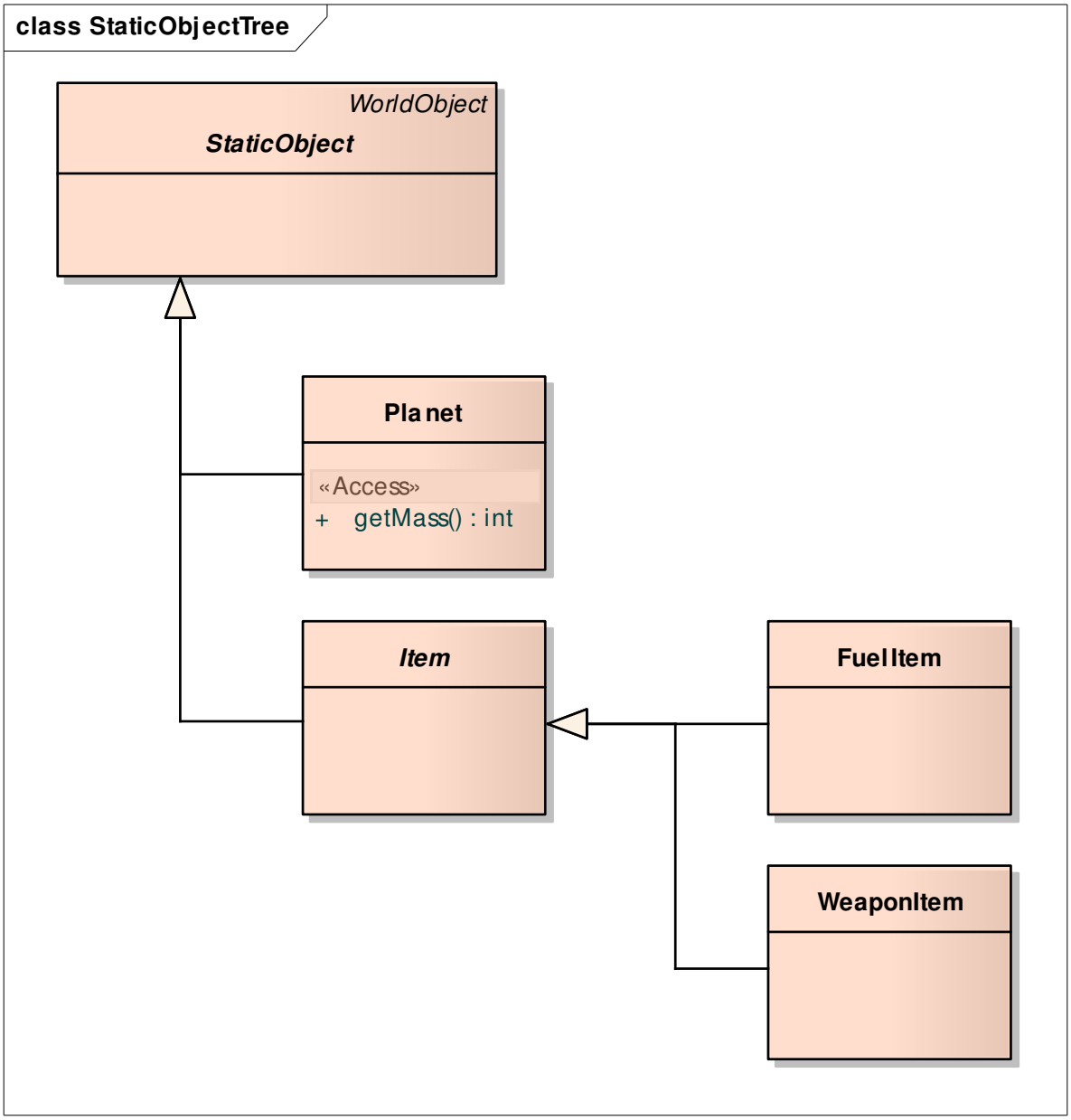

Figure 9 : StaticObjectTree

5.5.7.1 GameWorld::Asteroid

public Class

Extends: MovableObject. Implements: GravityAffectableInterface. : Representation of an asteroid flying around randomly in the game world.

GameWorld::Asteroid Connections

Connector	Source	Target	Notes
Generalization source > target	Asteroid Child	MovableObject Parent	
Realisation source > target	Asteroid Child	GravityAffectableInterface Parent	

5.5.7.2 GameWorld::FuelItem

public Class

Extends: Item. : Representation of an item that contains a fuel powerup.

GameWorld::FuelItem Connections

Connector	Source	Target	Notes
Generalization source > target	FuelItem Child	Item Parent	

5.5.7.3 GameWorld::Item

public abstract Class

Extends: StaticObject. : Contains common behavior and properties of items occuring in the game world.

GameWorld::Item Connections

Connector	Source	Target	Notes
Generalization source > target	FuelItem Child	Item Parent	
Generalization source > target	WeaponItem Child	Item Parent	
Generalization source > target	Item Child	StaticObject Parent	

5.5.7.4 GameWorld::LaserProjectile

public Class

Extends: Projectile. : Representation of a laser projectile, which is not affected by gravities.

GameWorld::LaserProjectile Connections

Connector	Source	Target	Notes
Dependency «instantiate» source > target	Ship Client Element	LaserProjectile Server Element	
Generalization source > target	LaserProjectile Child	Projectile Parent	

5.5.7.5 GameWorld::MissileProjectile

public Class

Extends: Projectile. Implements: GravityAffectableInterface. : Representation of a gravity-affectable missile projectile.

GameWorld::MissileProjectile Connections

Connector	Source	Target	Notes
Dependency «instantiate» source > target	Ship Client Element	MissileProjectile Server Element	
Generalization source > target	MissileProjectile Child	Projectile Parent	
Realisation source > target	MissileProjectile Child	GravityAffectableInterface Parent	

5.5.7.6 GameWorld::MovableObject

public abstract Class

Extends: WorldObject. Implements: TickableInterface. : Contains common behavior and properties of movable game world objects.

GameWorld::MovableObject Connections

Connector	Source	Target	Notes
Aggregation movement and pointing vectors source < target	Vector2d 2, unordered, none	MovableObject 1, unordered, none	One vector defines the movement and the other the direction that the object is pointing in.
Generalization source > target	Ship Child	MovableObject Parent	
Generalization source > target	MovableObject Child	WorldObject Parent	
Generalization source > target	Asteroid Child	MovableObject Parent	
Generalization source > target	Projectile Child	MovableObject Parent	
Realisation source > target	MovableObject Child	TickableInterface Parent	

GameWorld::MovableObject Methods

Method	Type	Notes
getMovement ()	«Access» public: *Vector2d*	Returns the movement vector of the movable object.
setMovement (*Vector2d*)	«Access» public: *void*	param: vector [Vector2d - in] Sets the movable object's movement vector.

5.5.7.7 GameWorld::Planet

public Class

Extends: StaticObject. : Representation of a planet.

GameWorld::Planet Connections

Connector	Source	Target	Notes
Generalization source > target	Planet Child	StaticObject Parent	

GameWorld::Planet Methods

Method	Type	Notes
getMass ()	«Access» public: *int*	Returns the mass of the planet.

5.5.7.8 GameWorld::Projectile

public abstract Class

Extends: MovableObject. : Contains common behavior and properties of weapon projectiles fired by a ship.

GameWorld::Projectile Connections

Connector	Source	Target	Notes
Aggregation fire sound source < target	Sound 1, unordered, none	Projectile 1, unordered, none	
Generalization source > target	MissileProjectile Child	Projectile Parent	
Generalization source > target	Projectile Child	MovableObject Parent	

Generalization source > target	LaserProjectile Child	Projectile Parent	

5.5.7.9 GameWorld::Ship

public Class

Extends: MovableObject. Implements: GravityAffectableInterface. : Representation of the ship that a game player will control.

GameWorld::Ship Connections

Connector	Source	Target	Notes
Aggregation destruction sound source < target	Sound 1, unordered, none	Ship 1, unordered, none	
Aggregation source < target	Ship 0..*, unordered, none	Player 1, unordered, none	
Dependency «instantiate» source > target	Ship Client Element	LaserProjectile Server Element	
Dependency «instantiate» source > target	Ship Client Element	MissileProjectile Server Element	
Generalization source > target	Ship Child	MovableObject Parent	
Realisation source > target	Ship Child	GravityAffectableInterface Parent	

GameWorld::Ship Methods

Method	Type	Notes
getFuelLevel ()	«Access» public: *int*	Returns the ship's fuel level.
getHealth ()	«Access» public: *int*	Returns the ship's health.

5.5.7.10 GameWorld::StaticObject

public abstract Class

Extends: WorldObject. : Contains common behavior and properties of static game world objects.

GameWorld::StaticObject Connections

Connector	Source	Target	Notes
Generalization source > target	Planet Child	StaticObject Parent	
Generalization source > target	Item Child	StaticObject Parent	
Generalization source > target	StaticObject Child	WorldObject Parent	

5.5.7.11 GameWorld::WeaponItem

public Class

Extends: Item. : Representation of an item that contains a ship weapon upgrade.

GameWorld::WeaponItem Connections

Connector	Source	Target	Notes
Generalization source > target	WeaponItem Child	Item Parent	

5.5.7.12 GameWorld::World

public Class: Representation of the whole game world, containing all world objects that are supposed to exist at a certain moment during a game session.

GameWorld::World Connections

Connector	Source	Target	Notes
Aggregation source < target	WorldObject 1..*, unordered, none	World 1.., unordered, none	
Dependency «manage» source > target	Engine Client Element	World Server Element	

GameWorld::World Methods

Method	Type	Notes
addWorldObject (*WorldObject*)	public: *void*	param: wo [WorldObject - in]
removeWorldObject (*WorldObject*)	public: *void*	param: wo [WorldObject - in]

5.5.7.13 GameWorld::WorldObject

public Class

Implements: SpriteHolderInterface. : Contains common behavior and properties of game world objects.

GameWorld::WorldObject Connections

Connector	Source	Target	Notes
Aggregation source < target	WorldObject 1..*, unordered, none	World 1.., unordered, none	
Aggregation graphical representation source < target	Sprite 1, unordered, none	WorldObject unordered, none	
Aggregation collision sound source < target	Sound 0..1, unordered, none	WorldObject unordered, none	
Dependency triggers event sound playback «call» source > target	WorldObject Client Element	Distributor Server Element	
Generalization source > target	MovableObject Child	WorldObject Parent	
Generalization source > target	StaticObject Child	WorldObject Parent	
Realisation source > target	WorldObject Child	SpriteHolderInterface Parent	

GameWorld::WorldObject Methods

Method	Type	Notes
collideWith (*WorldObject*)	«Manager» public: *void*	param: wo [WorldObject - in] Will be called when a collision is detected between the world object and another world object. Pre-condition: Functional The object participates in a collision. Post-condition: Functional The object may somehow be affected by the collision.
getPosition ()	«Access» public: *Vector2d*	Returns the position of the object in the world, according to the object's centre.

5.5.7.14 GameWorld::GravityAffectableInterface

public abstract «interface» Interface: If a world object implements this interface it means it is supposed to be affected by planetary gravities.

GameWorld::GravityAffectableInterface Connections

Connector	Source	Target	Notes
Realisation source > target	Ship Child	GravityAffectableInterface Parent	
Realisation source > target	Asteroid Child	GravityAffectableInterface Parent	
Realisation source > target	MissileProjectile Child	GravityAffectableInterface Parent	

5.5.8 Menu

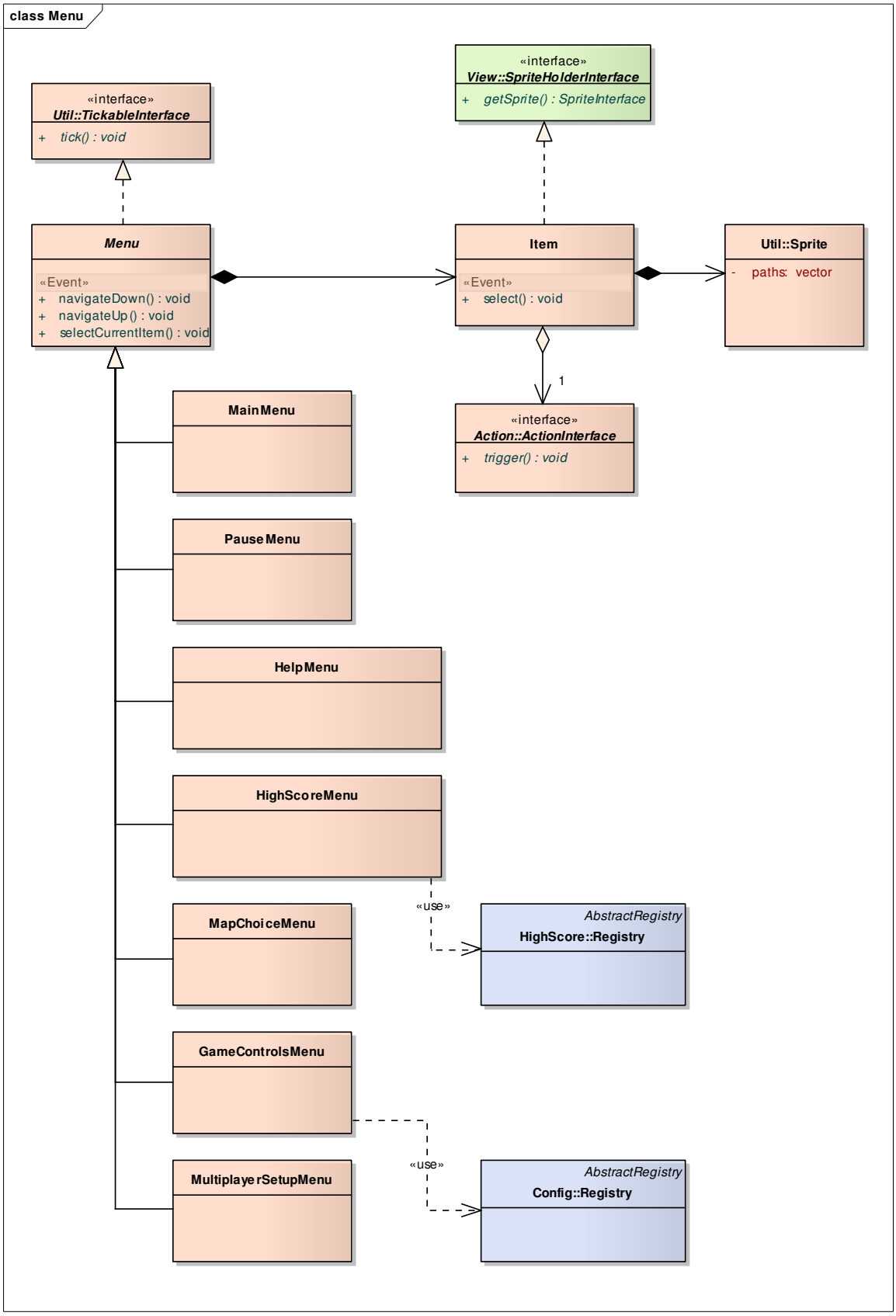

Figure 10 : Menu

5.5.8.1 Menu::GameControlsMenu

public Class

Extends: Menu. : The game controls configuration menu structure.

Menu::GameControlsMenu Connections

Connector	Source	Target	Notes
Dependency «use» source > target	GameControlsMenu Client Element	Registry Server Element	
Generalization source > target	GameControlsMenu Child	Menu Parent	

5.5.8.2 Menu::HelpMenu

public Class

Extends: Menu. : The help menu structure.

Menu::HelpMenu Connections

Connector	Source	Target	Notes
Generalization source > target	HelpMenu Child	Menu Parent	

5.5.8.3 Menu::HighScoreMenu

public Class

Extends: Menu. : The high score menu structure.

Menu::HighScoreMenu Connections

Connector	Source	Target	Notes
Dependency «use» source > target	HighScoreMenu Client Element	Registry Server Element	
Generalization source > target	HighScoreMenu Child	Menu Parent	

5.5.8.4 Menu::Item

public Class

Implements: SpriteHolderInterface. : A menu item representation, bringing together the graphical representation (sprite) and the action to take when activated.

Menu::Item Connections

Connector	Source	Target	Notes
Aggregation source < target	ActionInterface 1, unordered, none	Item unordered, none	
Aggregation source < target	Item unordered, none	Menu unordered, none	
Aggregation source < target	Sprite unordered, none	Item unordered, none	
Realisation source > target	Item Child	SpriteHolderInterface Parent	
Sequence source > target	MenuState Client Element	Item Server Element	

Menu::Item Methods

Method	Type	Notes
select ()	«Event» public: *void*	Causes the item to activate its associated action. Post-condition: Functional The item triggers its action

5.5.8.5 Menu::MainMenu

public Class

Extends: Menu. : The main menu structure.

Menu::MainMenu Connections

Connector	Source	Target	Notes
Generalization source > target	MainMenu Child	Menu Parent	

5.5.8.6 Menu::MapChoiceMenu

public Class

Extends: Menu. : The map choice menu structure.

Menu::MapChoiceMenu Connections

Connector	Source	Target	Notes
Generalization source > target	MapChoiceMenu Child	Menu Parent	

5.5.8.7 Menu::Menu

public abstract Class

Implements: TickableInterface. : Contains common properties and operations for a menu.

Menu::Menu Connections

Connector	Source	Target	Notes
Aggregation source < target	Item unordered, none	Menu unordered, none	
Generalization source > target	MultiplayerSetupMenu Child	Menu Parent	
Generalization source > target	GameControlsMenu Child	Menu Parent	
Generalization source > target	MainMenu Child	Menu Parent	
Generalization source > target	HelpMenu Child	Menu Parent	
Generalization source > target	HighScoreMenu Child	Menu Parent	
Generalization source > target	PauseMenu Child	Menu Parent	
Generalization source > target	MapChoiceMenu Child	Menu Parent	
Realisation source > target	Menu Child	TickableInterface Parent	
Sequence source > target	MenuState Client Element	Menu Server Element	
Sequence initControlSetLoop source > target	User Client Element	Menu Server Element	

Sequence setControlForAction source > target	Menu Client Element	Menu Server Element	
Sequence source > target	Menu Client Element	User Server Element	
Sequence source > target	MenuState Client Element	Menu Server Element	

Menu::Menu Methods

Method	Type	Notes
navigateDown ()	«Event» public: *void*	Moves the menu cursor to the item below.
navigateUp ()	«Event» public: *void*	Moves the menu cursor to the above item.
selectCurrentItem ()	«Event» public: *void*	Causes the current item to be selected, i.e., the item's action is triggered. Post-condition: Functional The item's action is triggered.

5.5.8.8 Menu::MultiplayerSetupMenu

public Class

Extends: Menu. : The menu to show before starting a multiplayer game session.

Menu::MultiplayerSetupMenu Connections

Connector	Source	Target	Notes
Generalization source > target	MultiplayerSetupMenu Child	Menu Parent	

5.5.8.9 Menu::PauseMenu

public Class

Extends: Menu. : The pause menu structure.

Menu::PauseMenu Connections

Connector	Source	Target	Notes
Generalization source > target	PauseMenu Child	Menu Parent	

5.5.9 Player

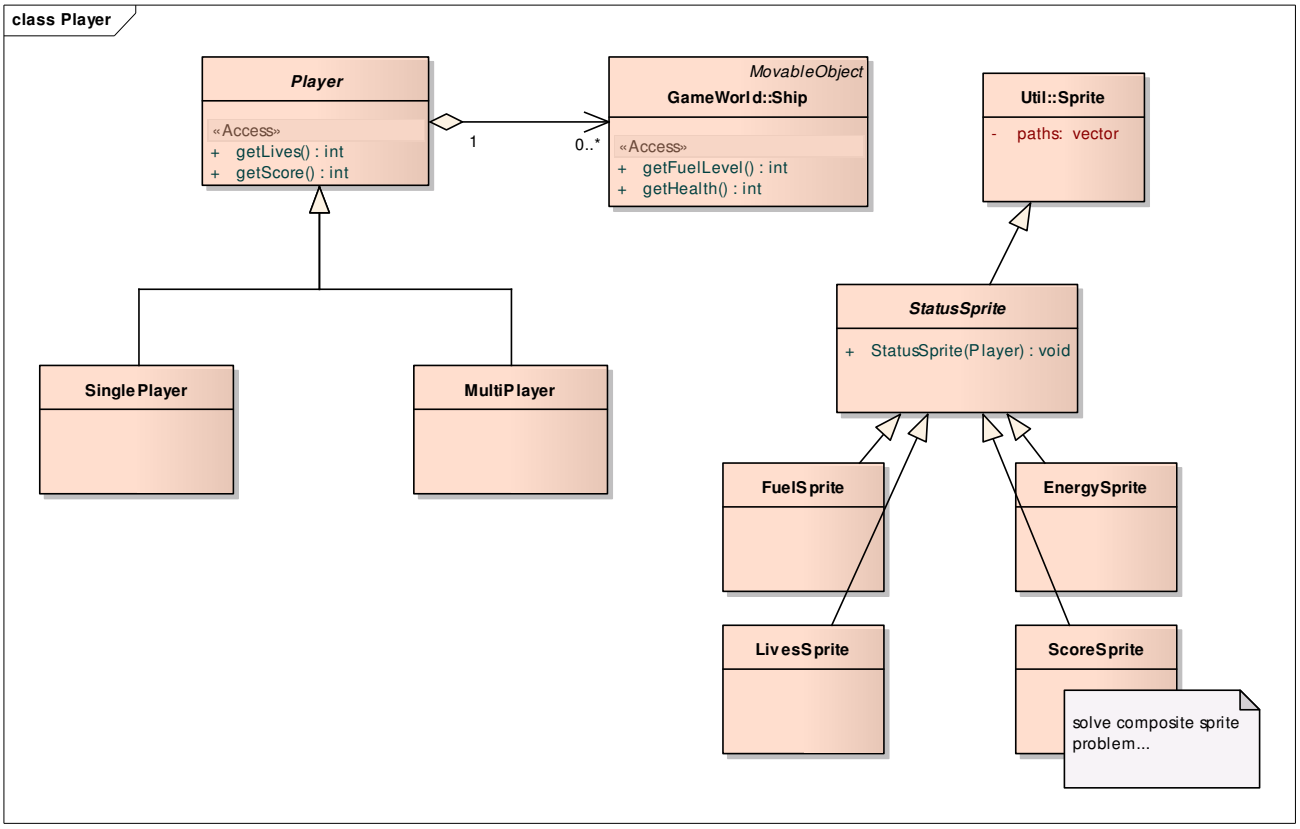

Figure 11 : Player

5.5.9.1 EnergySprite

EnergySprite Connections

Connector	Source	Target	Notes
Generalization source > target	EnergySprite Child	StatusSprite Parent	

5.5.9.2 FuelSprite

FuelSprite Connections

Connector	Source	Target	Notes
Generalization source > target	FuelSprite Child	StatusSprite Parent	

5.5.9.3 LivesSprite

LivesSprite Connections

Connector	Source	Target	Notes
Generalization source > target	LivesSprite Child	StatusSprite Parent	

5.5.9.4 MultiPlayer

public Class

Extends: Player. : Holds statistics about a player when playing in multiplayer mode.

MultiPlayer Connections

Connector	Source	Target	Notes
Generalization source > target	MultiPlayer Child	Player Parent	

5.5.9.5 Player

public abstract Class: Holds common properties for an abstract player.

Player Connections

Connector	Source	Target	Notes
Aggregation source < target	Ship 0..*, unordered, none	Player 1, unordered, none	
Generalization source > target	SinglePlayer Child	Player Parent	
Generalization source > target	MultiPlayer Child	Player Parent	

Player Methods

Method	Type	Notes
getLives ()	«Access» public: *int*	Gets the number of lives left.
getScore ()	«Access» public: *int*	Returns the player's score.

5.5.9.6 ScoreSprite

ScoreSprite Connections

Connector	Source	Target	Notes
Generalization source > target	ScoreSprite Child	StatusSprite Parent	

5.5.9.7 SinglePlayer

public Class

Extends: Player. : Holds statistics about a player when playing in single player mode.

SinglePlayer Connections

Connector	Source	Target	Notes
Generalization source > target	SinglePlayer Child	Player Parent	

5.5.9.8 StatusSprite

StatusSprite Connections

Connector	Source	Target	Notes
Generalization source > target	StatusSprite Child	Sprite Parent	
Generalization source > target	LivesSprite Child	StatusSprite Parent	
Generalization source > target	ScoreSprite Child	StatusSprite Parent	
Generalization source > target	FuelSprite Child	StatusSprite Parent	
Generalization source > target	EnergySprite Child	StatusSprite Parent	

StatusSprite Methods

Method	Type	Notes
StatusSprite (*Player*)	public: *void*	param: player [Player - in]

5.5.10 State

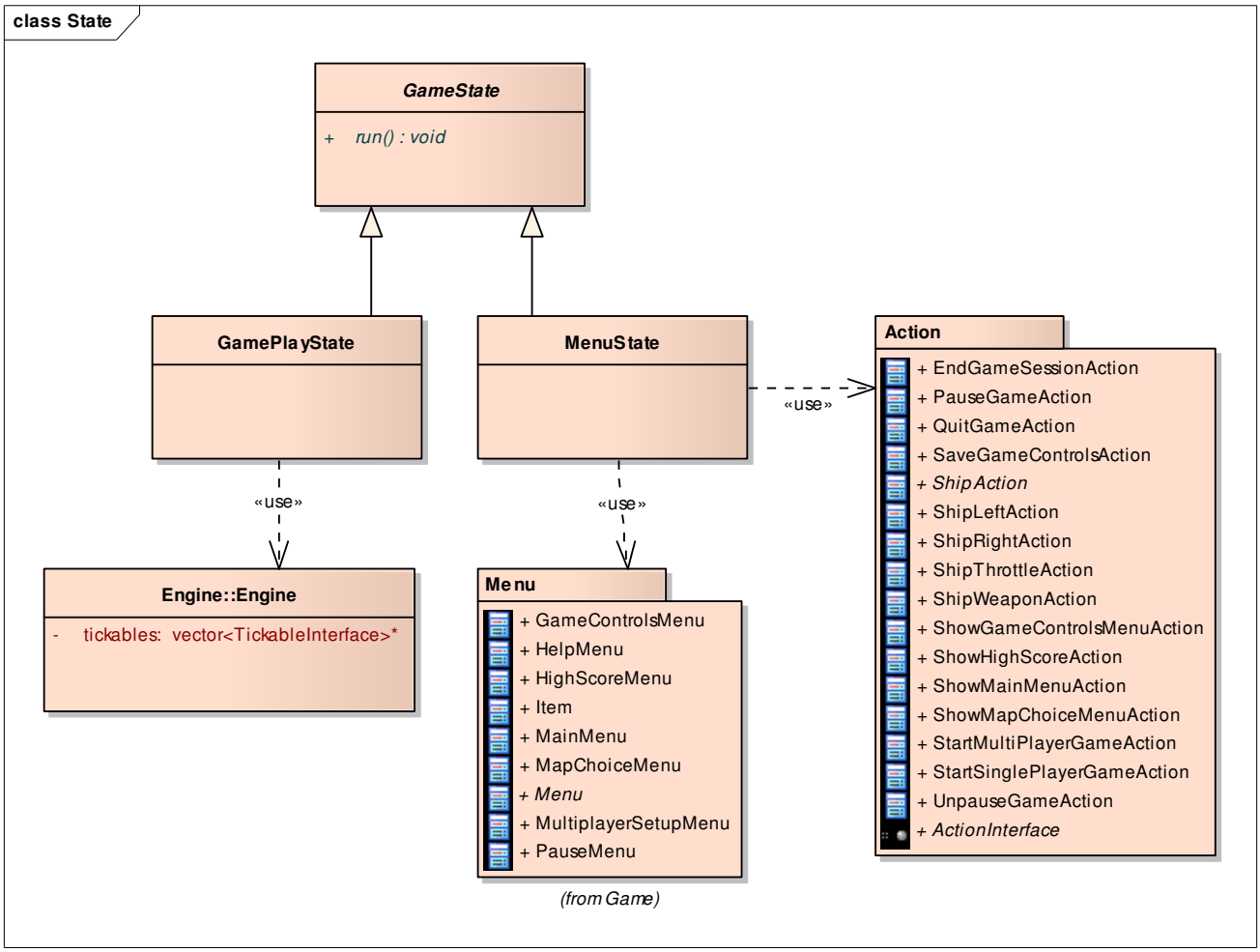

Figure 12 : State

5.5.10.1 State::GamePlayState

public Class

Extends: GameState. : The game state to be in when a game session is active.

State::GamePlayState Connections

Connector	Source	Target	Notes
Dependency «use» source > target	GamePlayState Client Element	Distributor Server Element	
Dependency «use» source > target	GamePlayState Client Element	Engine Server Element	
Dependency «use» source > target	GamePlayState Client Element	Engine Server Element	
Generalization source > target	GamePlayState Child	GameState Parent	
Sequence source > target	MenuState Client Element	GamePlayState Server Element	

5.5.10.2 State::GameState

public abstract Class: Contains common functionality for a game state.

State::GameState Connections

Connector	Source	Target	Notes
Dependency «call» source > target	GameState Client Element	WorldBoundaryManager Server Element	
Dependency «call» source > target	GameState Client Element	InputManager Server Element	
Dependency «call» source > target	GameState Client Element	GravityManager Server Element	
Dependency «call» source > target	GameState Client Element	CollisionManager Server Element	
Dependency «call» source > target	GameState Client Element	RenderEngine Server Element	Instantiates and calls.
Generalization source > target	GamePlayState Child	GameState Parent	
Generalization source > target	MenuState Child	GameState Parent	

State::GameState Methods

Method	Type	Notes
run ()	public abstract: *void*	Runs the main loop of the game, refreshing input listener, causing the view to refresh etc. Post-condition: Functional The main game loop resides in the game state.

5.5.10.3 State::MenuState

public Class

Extends: GameState. : The state to be in when navigating through menus.

State::MenuState Connections

Connector	Source	Target	Notes
Dependency «use» source > target	MenuState Client Element	Menu Server Element	
Dependency «use» source > target	MenuState Client Element	Action Server Element	
Generalization source > target	MenuState Child	GameState Parent	
Sequence startGameSession source > target	User Client Element	MenuState Server Element	
Sequence switchGameState source > target	MenuState Client Element	Game Server Element	
Sequence source > target	MenuState Client Element	Menu Server Element	
Sequence setGameControls source > target	User Client Element	MenuState Server Element	

Sequence source > target	Game Client Element	MenuState Server Element	
Sequence init source > target	Game Client Element	MenuState Server Element	
Sequence source > target	MenuState Client Element	Item Server Element	
Sequence source > target	MenuState Client Element	GamePlayState Server Element	
Sequence source > target	MenuState Client Element	Menu Server Element	
Sequence startGameSession source > target	User Client Element	MenuState Server Element	

5.5.11 Action

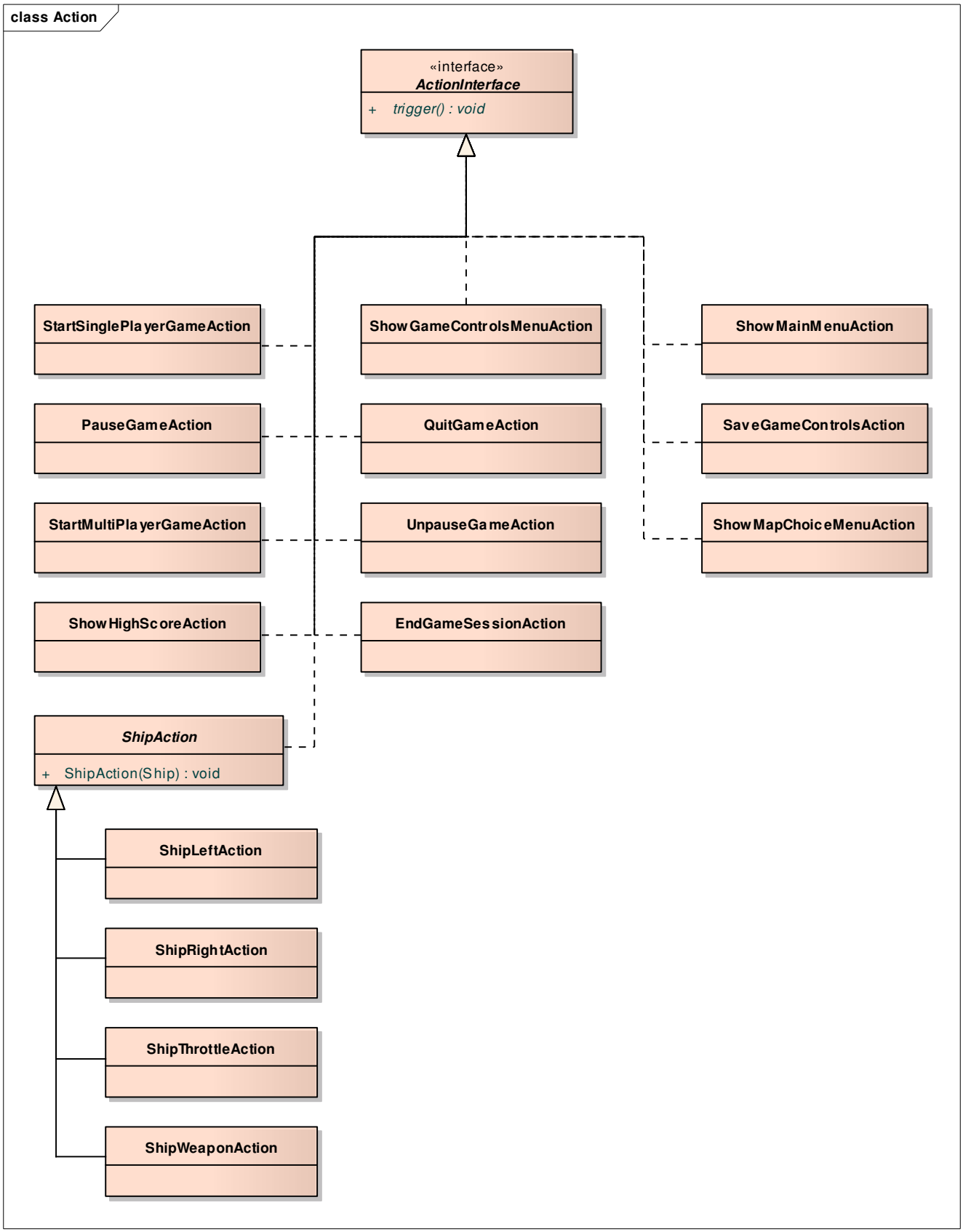

Figure 13 : Action

5.5.11.1 Action::EndGameSessionAction

public Class

Implements: ActionInterface. : End a game session.

Action::EndGameSessionAction Connections

Connector	Source	Target	Notes
Realisation source > target	EndGameSessionAction Child	ActionInterface Parent	

5.5.11.2 Action::PauseGameAction

public Class

Implements: ActionInterface. : Pause a game session.

Action::PauseGameAction Connections

Connector	Source	Target	Notes
Realisation source > target	PauseGameAction Child	ActionInterface Parent	

5.5.11.3 Action::QuitGameAction

public Class

Implements: ActionInterface. : Quit the whole game application.

Action::QuitGameAction Connections

Connector	Source	Target	Notes
Realisation source > target	QuitGameAction Child	ActionInterface Parent	

5.5.11.4 Action::SaveGameControlsAction

public Class

Implements: ActionInterface. : Save game controls when set.

Action::SaveGameControlsAction Connections

Connector	Source	Target	Notes
Realisation source > target	SaveGameControlsAction Child	ActionInterface Parent	

5.5.11.5 Action::ShipAction

public abstract Class

Implements: ActionInterface. : Common properties and operations for ship actions.

Action::ShipAction Connections

Connector	Source	Target	Notes
Generalization source > target	ShipWeaponAction Child	ShipAction Parent	
Generalization source > target	ShipLeftAction Child	ShipAction Parent	
Generalization source > target	ShipRightAction Child	ShipAction Parent	
Generalization source > target	ShipThrottleAction Child	ShipAction Parent	
Realisation source > target	ShipAction Child	ActionInterface Parent	

Action::ShipAction Methods

Method	Type	Notes
ShipAction (*Ship*)	public: *void*	param: ship [Ship - in] The ship to affect.

5.5.11.6 Action::ShipLeftAction

public Class

Extends: ShipAction. : Turn a ship left.

Action::ShipLeftAction Connections

Connector	Source	Target	Notes
Generalization source > target	ShipLeftAction Child	ShipAction Parent	

5.5.11.7 Action::ShipRightAction

public Class

Extends: ShipAction. : Turn a ship right.

Action::ShipRightAction Connections

Connector	Source	Target	Notes
Generalization source > target	ShipRightAction Child	ShipAction Parent	

5.5.11.8 Action::ShipThrottleAction

public Class

Extends: ShipAction. : Throttle a ship.

Action::ShipThrottleAction Connections

Connector	Source	Target	Notes
Generalization source > target	ShipThrottleAction Child	ShipAction Parent	

5.5.11.9 Action::ShipWeaponAction

public Class

Extends: ShipAction. : Fire a ship's weapon.

Action::ShipWeaponAction Connections

Connector	Source	Target	Notes
Generalization source > target	ShipWeaponAction Child	ShipAction Parent	

5.5.11.10 Action::ShowGameControlsMenuAction

public Class

Implements: ActionInterface. : Load the game controls configuration menu.

Action::ShowGameControlsMenuAction Connections

Connector	Source	Target	Notes
Realisation source > target	ShowGameControlsMenuAction Child	ActionInterface Parent	

5.5.11.11 Action::ShowHighScoreAction

public Class

Implements: ActionInterface. : Show high score screen/menu.

Action::ShowHighScoreAction Connections

Connector	Source	Target	Notes
Realisation source > target	ShowHighScoreAction Child	ActionInterface Parent	

5.5.11.12 Action::ShowMainMenuAction

public Class

Implements: ActionInterface. : Show the main menu.

Action::ShowMainMenuAction Connections

Connector	Source	Target	Notes
Realisation source > target	ShowMainMenuAction Child	ActionInterface Parent	

5.5.11.13 Action::ShowMapChoiceMenuAction

public Class

Implements: ActionInterface. : Show map choice menu.

Action::ShowMapChoiceMenuAction Connections

Connector	Source	Target	Notes
Realisation source > target	ShowMapChoiceMenuAction Child	ActionInterface Parent	

5.5.11.14 Action::StartMultiPlayerGameAction

public Class

Implements: ActionInterface. : Start a multiplayer game on a selected map.

Action::StartMultiPlayerGameAction Connections

Connector	Source	Target	Notes
Realisation source > target	StartMultiPlayerGameAction Child	ActionInterface Parent	

5.5.11.15 Action::StartSinglePlayerGameAction

public Class

Implements: ActionInterface. : Start a single player game on a selected map.

Action::StartSinglePlayerGameAction Connections

Connector	Source	Target	Notes
Realisation source > target	StartSinglePlayerGame Action Child	ActionInterface Parent	

5.5.11.16 Action::UnpauseGameAction

public Class

Implements: ActionInterface. : Unpause a game session.

Action::UnpauseGameAction Connections

Connector	Source	Target	Notes
Realisation source > target	UnpauseGameAction Child	ActionInterface Parent	

5.5.11.17 Action::ActionInterface

public abstract «interface» Interface: Interface that shall be implemented by each class that represents an action to take within a specific game state. Examples of possible actions: "start game", "load game controls menu", "pause game".

Action::ActionInterface Connections

Connector	Source	Target	Notes
Aggregation source < target	ActionInterface 1, unordered, none	Item unordered, none	
Aggregation maps input to game actions source < target	ActionInterface unordered, none	InputManager unordered, none	
Realisation source > target	StartSinglePlayerGame Action Child	ActionInterface Parent	
Realisation source > target	UnpauseGameAction Child	ActionInterface Parent	
Realisation source > target	ShipAction Child	ActionInterface Parent	
Realisation source > target	ShowHighScoreAction Child	ActionInterface Parent	
Realisation source > target	StartMultiPlayerGame Action Child	ActionInterface Parent	
Realisation source > target	ShowGameControlsMe nuAction Child	ActionInterface Parent	
Realisation source > target	EndGameSessionActio n Child	ActionInterface Parent	
Realisation source > target	ShowMainMenuAction Child	ActionInterface Parent	
Realisation source > target	ShowMapChoiceMenu Action Child	ActionInterface Parent	
Realisation source > target	SaveGameControlsAct ion	ActionInterface Parent	

	Child		
Realisation source > target	PauseGameAction Child	ActionInterface Parent	
Realisation source > target	QuitGameAction Child	ActionInterface Parent	

Action::ActionInterface Interfaces

Method	Type	Notes
trigger ()	public abstract: *void*	This method is called every time an action is requested to be triggered. Pre-condition: Functional Some event requires an action to be triggered. Post-condition: Functional The action does what it is supposed to do with the system.

5.5.12 Persistence

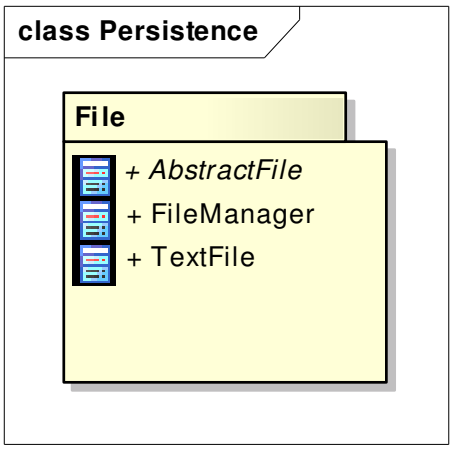

Figure 14 : Persistence

5.5.13 File

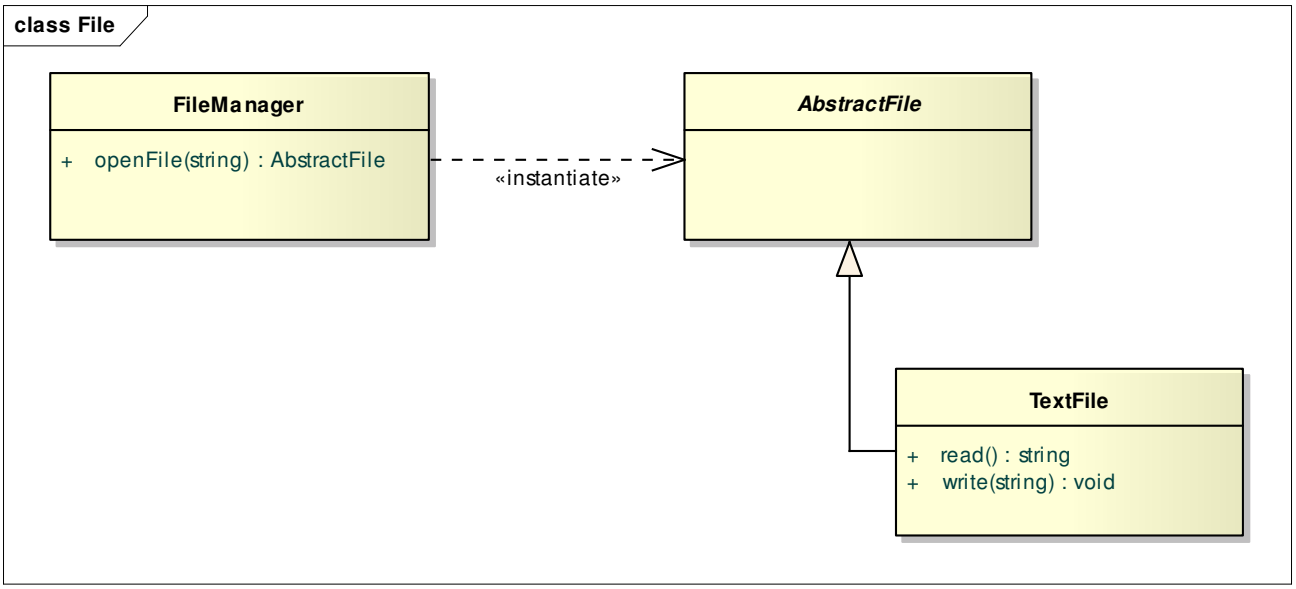

Figure 15 : File

5.5.13.1 File::AbstractFile

public abstract Class: Contains common operations and functionality of a file system file.

File::AbstractFile Connections

Connector	Source	Target	Notes
Dependency «instantiate» source > target	FileManager Client Element	AbstractFile Server Element	
Generalization source > target	TextFile Child	AbstractFile Parent	

5.5.13.2 File::FileManager

public Class: Provides functionality to create, read, modify and write files in the file system.

File::FileManager Connections

Connector	Source	Target	Notes
Dependency retrieve map data from file «use» source > target	Registry Client Element	FileManager Server Element	
Dependency save and retrieve config from file «use» source > target	Registry Client Element	FileManager Server Element	
Dependency read and write highscore file «use» source > target	Registry Client Element	FileManager Server Element	
Dependency «instantiate» source > target	FileManager Client Element	AbstractFile Server Element	

File::FileManager Methods

Method	Type	Notes
openFile (*string*)	public: *AbstractFile*	param: path [string - in] Opens a file stream and encapsulates it in an object.

5.5.13.3 File::TextFile

public Class

Extends: AbstractFile. : Represents a plain text file and provides methods to read and write such files.

File::TextFile Connections

Connector	Source	Target	Notes
Generalization source > target	TextFile Child	AbstractFile Parent	

File::TextFile Methods

Method	Type	Notes
read ()	public: *string*	Reads content from a text file and returns it in a string. Pre-condition: Functional The file exists. Post-condition: Functional The file content is read and returned in a string
write (*string*)	public: *void*	param: str [string - in]

5.5.14 Registry

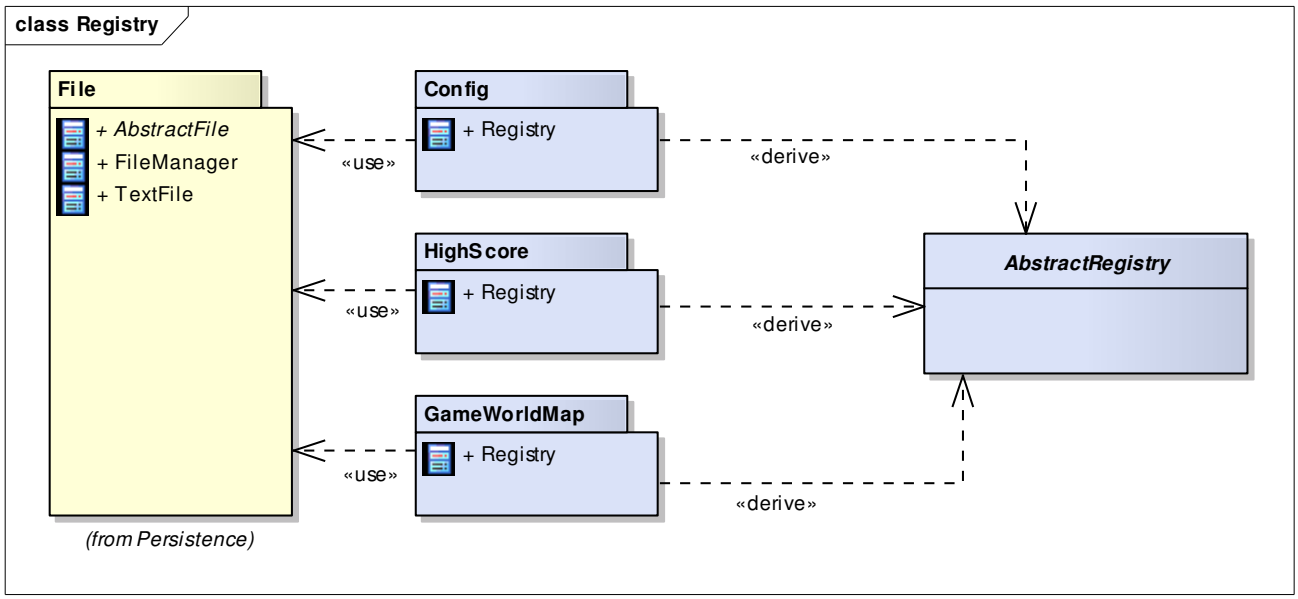

Figure 16 : Registry

5.5.14.1 Registry::AbstractRegistry

public abstract Class: Common functionality for the registry pattern.

Registry::AbstractRegistry Connections

Connector	Source	Target	Notes
Dependency «derive» source > target	HighScore Client Element	AbstractRegistry Server Element	
Dependency «derive» source > target	GameWorldMap Client Element	AbstractRegistry Server Element	
Dependency «derive» source > target	Config Client Element	AbstractRegistry Server Element	
Generalization source > target	Registry Child	AbstractRegistry Parent	
Generalization source > target	Registry Child	AbstractRegistry Parent	
Generalization source > target	Registry Child	AbstractRegistry Parent	

5.5.15 Config

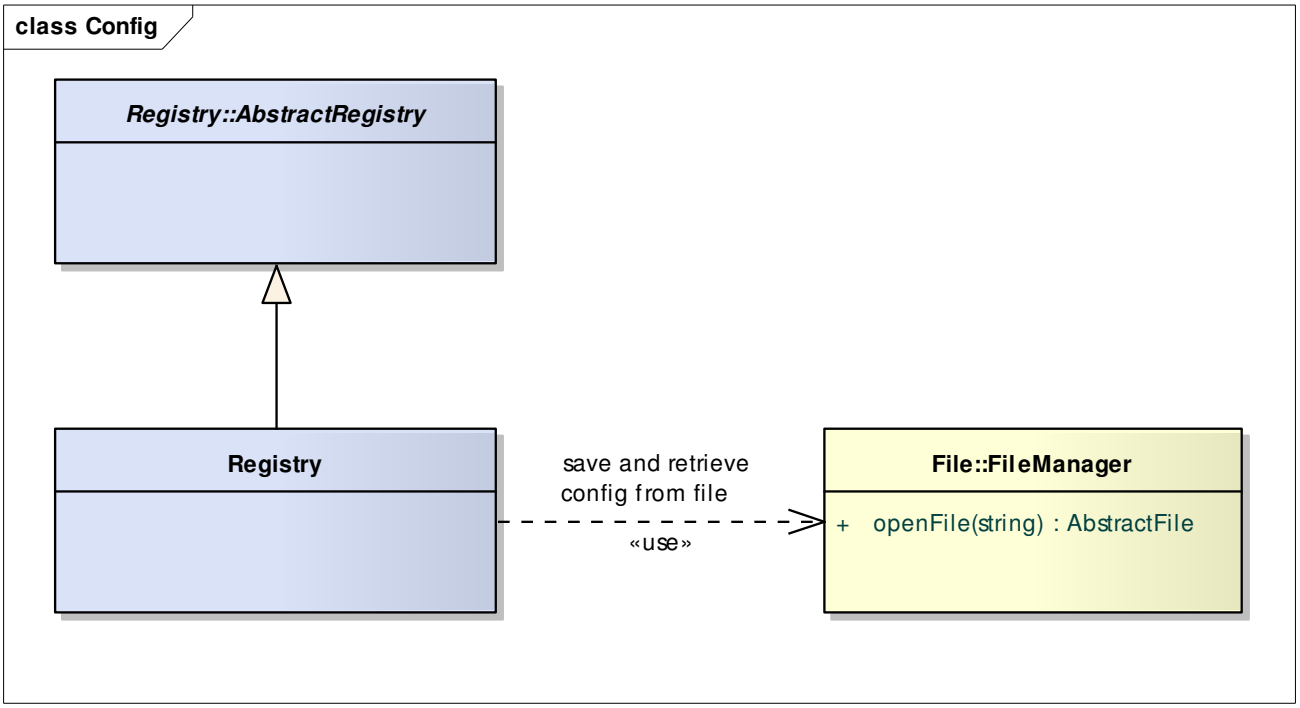

Figure 17 : Config

5.5.15.1 Config::Registry

public Class

Extends: AbstractRegistry. : Registry of the game configuration settings.

Config::Registry Connections

Connector	Source	Target	Notes
Dependency save and retrieve config from file «use» source > target	Registry Client Element	FileManager Server Element	
Dependency «use» source > target	GameControlsMenu Client Element	Registry Server Element	
Generalization source > target	Registry Child	AbstractRegistry Parent	

5.5.16 GameWorldMap

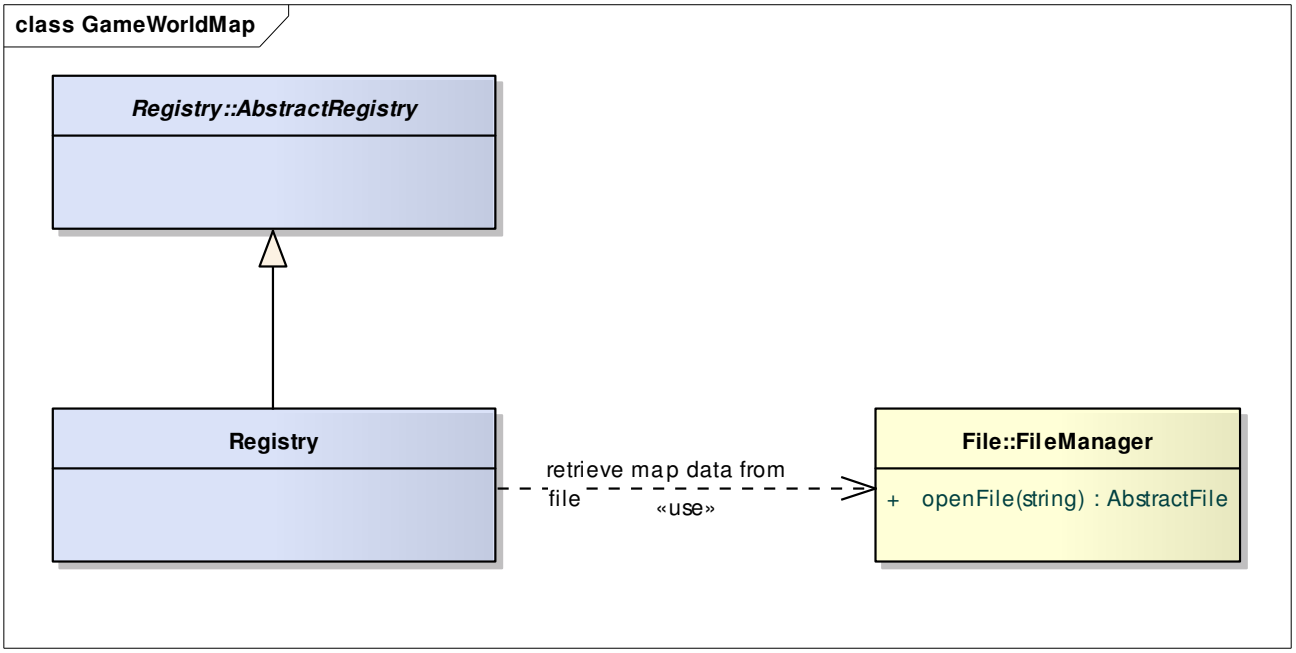

Figure 18 : GameWorldMap

5.5.16.1 GameWorldMap::Registry

public Class

Extends: AbstractRegistry. : Provides an interface for accessing game world map data.

GameWorldMap::Registry Connections

Connector	Source	Target	Notes
Dependency retrieve map data from file «use» source > target	Registry Client Element	FileManager Server Element	
Generalization source > target	Registry Child	AbstractRegistry Parent	

5.5.17 HighScore

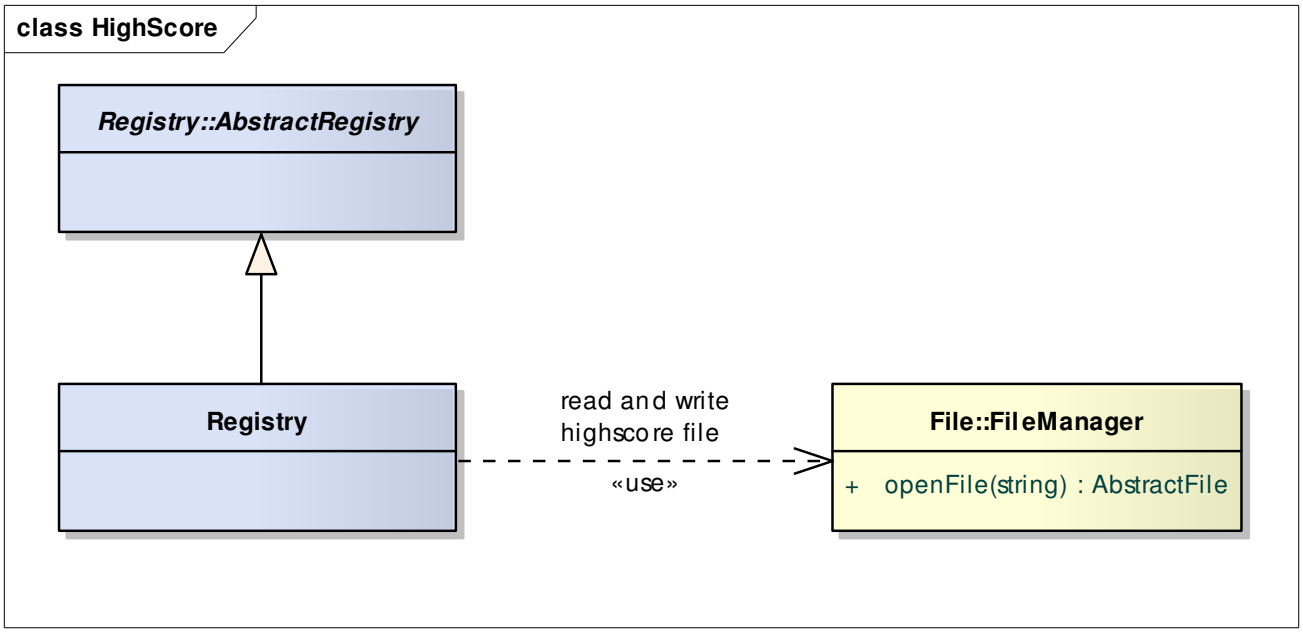

Figure 19 : HighScore

5.5.17.1 HighScore::Registry

public Class

Extends: AbstractRegistry. : Manages player high scores.

HighScore::Registry Connections

Connector	Source	Target	Notes
Dependency read and write highscore file «use» source > target	Registry Client Element	FileManager Server Element	
Dependency «use» source > target	HighScoreMenu Client Element	Registry Server Element	
Generalization source > target	Registry Child	AbstractRegistry Parent	

5.5.18 View

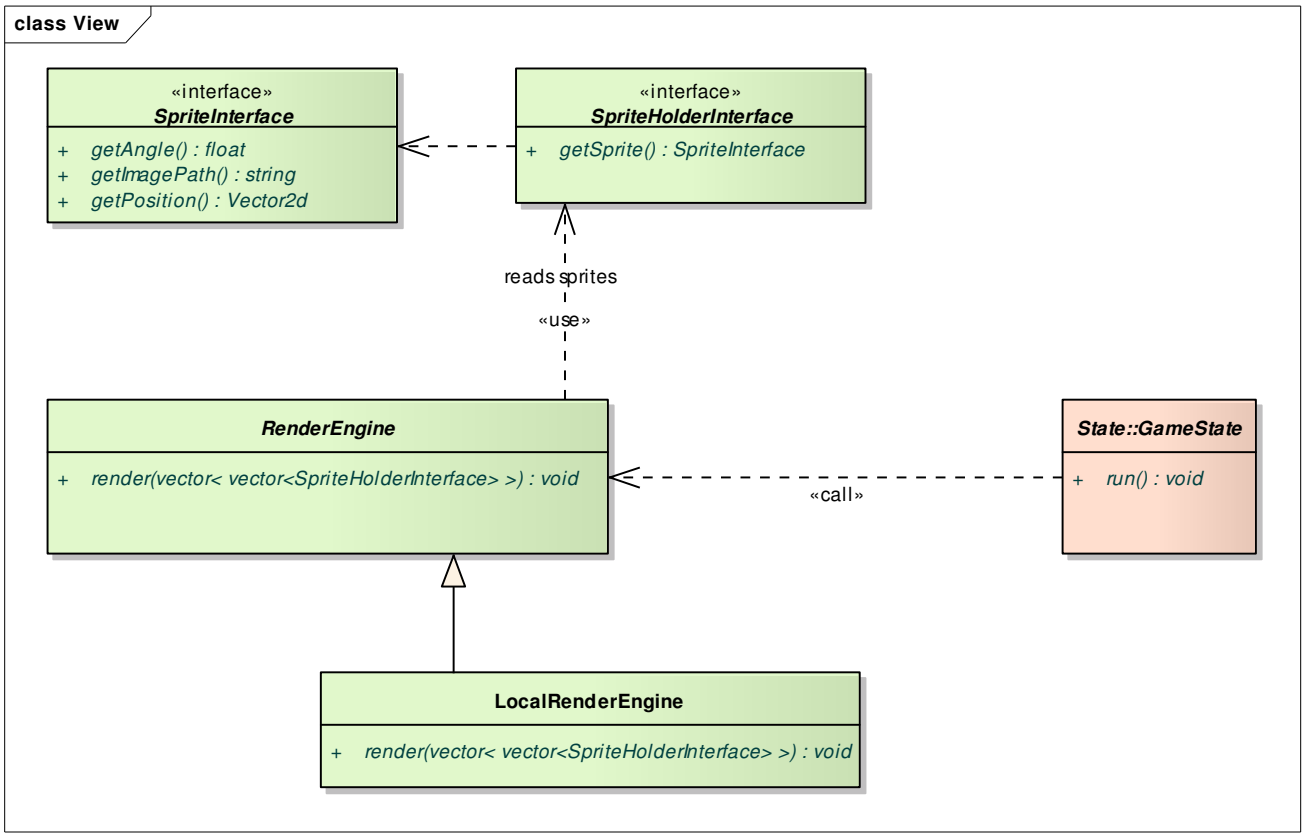

Figure 20 : View

5.5.18.1 View::LocalRenderEngine

public Class

Extends: RenderEngine. : Renders at most 2 screens locally, i.e., on the same computer.

View::LocalRenderEngine Connections

Connector	Source	Target	Notes
Generalization source > target	LocalRenderEngine Child	RenderEngine Parent	

View::LocalRenderEngine Methods

Method	Type	Notes
render (*vector< vector<SpriteHolderInterface > >*)	public abstract: *void*	param: screens [vector< vector<SpriteHolderInterface> > - in] Render a vector of sprite vectors, each one for a distinct screen. Pre-condition: Functional The screens vector has a length of 1-2 screen vectors. Pre-condition: Functional There is a game world that needs rendering. Post-condition: Display All player screens are updated with the new view.

5.5.18.2 View::RenderEngine

public abstract Class: Contains common render engine operations and attributes.

View::RenderEngine Connections

Connector	Source	Target	Notes
Dependency reads sprites «use» source > target	RenderEngine Client Element	SpriteHolderInterface Server Element	
Dependency «call» source > target	GameState Client Element	RenderEngine Server Element	Instantiates and calls.
Dependency «use» source > target	State Client Element	RenderEngine Server Element	
Generalization source > target	LocalRenderEngine Child	RenderEngine Parent	

View::RenderEngine Methods

Method	Type	Notes
render (*vector< vector<SpriteHolderInterface > >*)	public abstract: *void*	param: screens [vector< vector<SpriteHolderInterface> > - in] Render a vector of sprite vectors, each one for a distinct screen. Pre-condition: Functional There is a game world that needs rendering. Post-condition: Display All player screens are updated with the new view.

5.5.18.3 View::SpriteHolderInterface

View::SpriteHolderInterface Connections

Connector	Source	Target	Notes
Dependency source > target	SpriteHolderInterface Client Element	SpriteInterface Server Element	
Dependency reads sprites «use» source > target	RenderEngine Client Element	SpriteHolderInterface Server Element	
Realisation source > target	Item Child	SpriteHolderInterface Parent	
Realisation source > target	WorldObject Child	SpriteHolderInterface Parent	

View::SpriteHolderInterface Interfaces

Method	Type	Notes
getSprite ()	public abstract: *SpriteInterface*	Returns the sprite that the sprite holder is holding.

5.5.18.4 View::SpriteInterface

public abstract «interface» Interface: Interface defining what methods must be present in a sprite for the view rendering to function properly.

View::SpriteInterface Connections

Connector	Source	Target	Notes
Dependency source > target	SpriteHolderInterface Client Element	SpriteInterface Server Element	
Realisation source > target	Sprite Child	SpriteInterface Parent	

View::SpriteInterface Interfaces

Method	Type	Notes
getAngle ()	public abstract: *float*	Returns the angle that the sprite is supposed to have. The angle is calculated according to the unit circle centered on the sprite. Used primarily by the view module.
getImagePath ()	public abstract: *string*	Returns the relative path to the sprite image.
getPosition ()	public abstract: *Vector2d*	Returns the position of the sprite's upper left corner.

5.5.19 Functional requirements cross reference

5.5.19.1 Game configuration

6.2.1. Quick Start Help

Gravity::Game::Menu::HelpMenu

6.2.2. Map Choice

Gravity::Game::Menu::MapChoiceMenu

6.2.3. Controls Configuration

Gravity::Game::Menu::GameControlsMenu

Gravity::Game::State::MenuState

6.2.4. Single Player or Two Players Choice

Gravity::Game::Menu::MainMenu

Gravity::Player::SinglePlayer

6.2.5. Two Players Game Rule Choice

Gravity::Game::Menu::MultiplayerSetupMenu

6.2.6. Single Player High Score List

Gravity::Game::Menu::HighScoreMenu

6.2.7. Exiting The Game

Gravity::Game::Menu::MainMenu

5.5.19.2 Game play

6.2.8. World Boundary Wrapping

Game::Engine::Physics::WorldBoundaryManager

6.2.9. Player's World View

Gravity::View::RenderEngine

6.2.10. Game Play Information

Gravity::Player::StatusSprite

6.2.11. Scoring in Single Player Mode

Gravity::Player::SinglePlayer

6.2.12. Scoring in Two Players Mode

Gravity::Player::MultiPlayer

6.2.13. Single Player Ship Disposal (Lives)
Gravity::Game::SinglePlayer

6.2.14. Ship Speed Restriction
Gravity::Game::Engine::Physics

6.2.15. Ship Fuel Restriction
Gravity::Game::GameWorld::Ship

6.2.16. Ship Damage
Gravity::Game::GameWorld::Ship

6.2.17. Operating a ship
Gravity::Controller
Gravity::GameWorld::Ship

6.2.18. Ship Laser Gun
Gravity::GameWorld::LaserProjectile

6.2.19. Ship Missile Launcher
Gravity::GameWorld::MissileProjectile

6.2.20. Operating a Ship's Weapons
Gravity::Controller
Gravity::GameWorld::MissileProjectile
Gravity::GameWorld::LaserProjectile

5.5.19.3 Passive Objects

6.2.21. Planets
Gravity::GameWorld::Planet

6.2.22. Asteroids
Gravity::GameWorld::Asteroid

6.2.23. Items
Gravity::GameWorld::Item

5.5.19.4 Misc

6.2.24. Sound Effects
Gravity::Audio