Analyse- und Designmodell

Gruppe 1: Wiesn-Run

Übersicht

- Spielidee, Ziel
  - Spielerstatus, Powerups, Gegnertypen, Angriffe
- Levels
- Menüs
- Klassen

Spielidee: Wiesn-Run

- Oktoberfest-Grafik
- Figur fix, Welt bewegt sich

- Steuerung:
  - rechts laufen
  - springen: entweder senkrecht oder schräg vorne
  - werfen horizontal

Ziel des Spiels

Punkte sammeln durch

- Gegner töten
- Strecke zurücklegen
- Items sammeln
- Level mit möglichst hohem “Pegel” beenden

Spielerstatus

- Pegel: +/- über Powerups und Zeit

  Ansporn, Level schnell zu beenden

- Punkte

- Gesundheit

Powerups

- Bier: + Pegel, +Gesundheit, +Munition
- Hendl: - Pegel, +Gesundheit

Gegner

- Türsteher: Schaden durch Kontakt
- Betrunkener Tourist: wirft Masskrüge
- Endgegner im letzten Level

- Treffer reduzieren Gesundheit
  - Auswertung durch Kollisionskontrolle und Eventhandling
  - nach Treffern kurze Unverwundbarkeit

Angriffsarten

- Player:
  - auf Gegner springen, Masskrug werfen

- Gegner:
  - gegen Spieler laufen, Masskrug werfen

Menü

- graphisch
- Steuerung über Tastatur
- Startmenü
- Statistik
  - getötete Gegner, gesammelte Items, Zeit
  - Namenseingabe
- Highscore
  - wird in Textdatei gespeichert


WILLKOMMEN ZUM
Wiesn Run
Pack ma's!
Highscores
Servus!

Levels

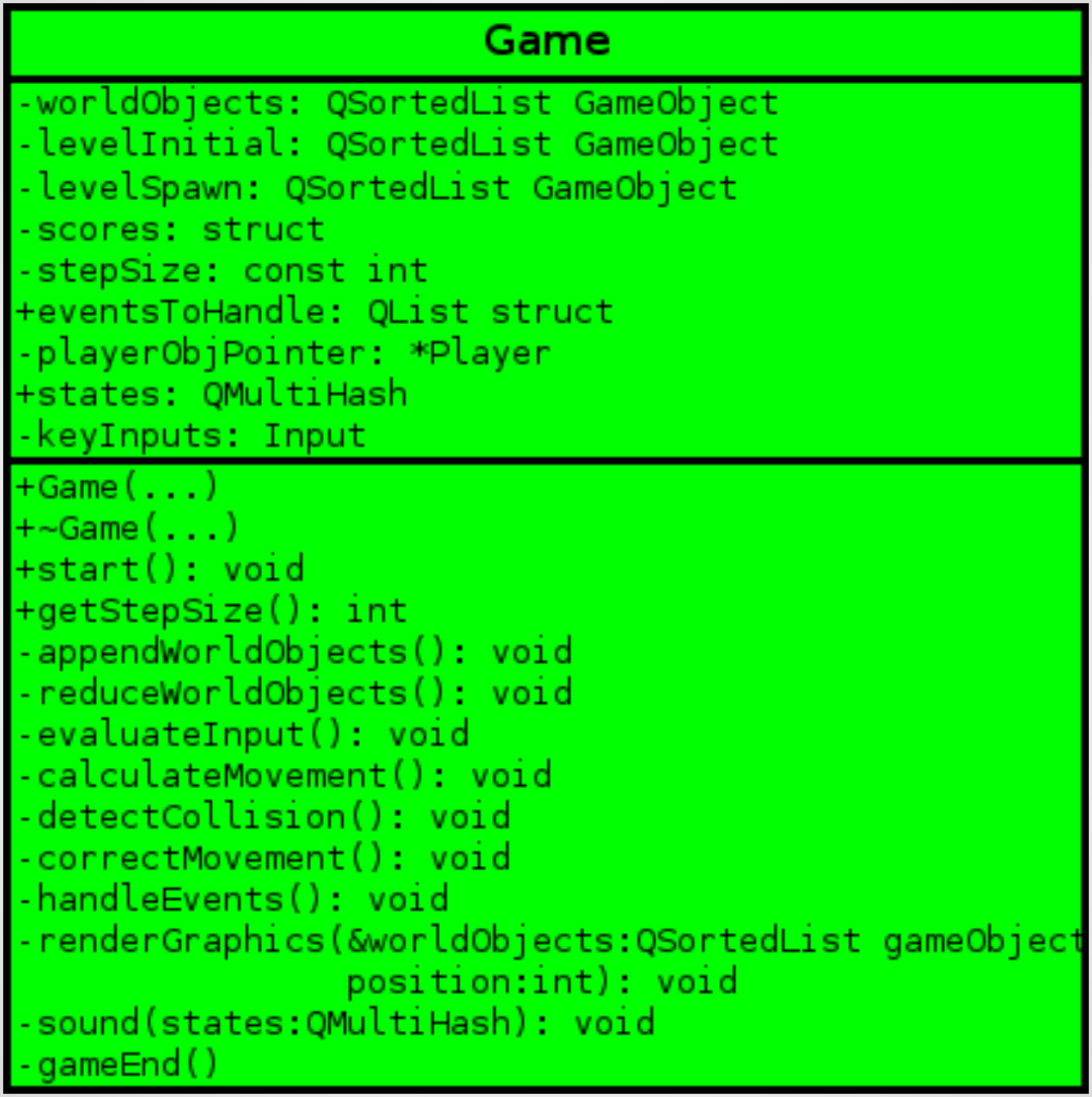
- Weg zum Zelt (Hauptstraße)
  - nur Hindernisse, keine Gegner
- Hacker-Zelt
  - Türsteher als Gegner
- Käfer-Zelt
  - masskrugwerfende Touristen, mehr Türsteher
  - Endgegner am Ende

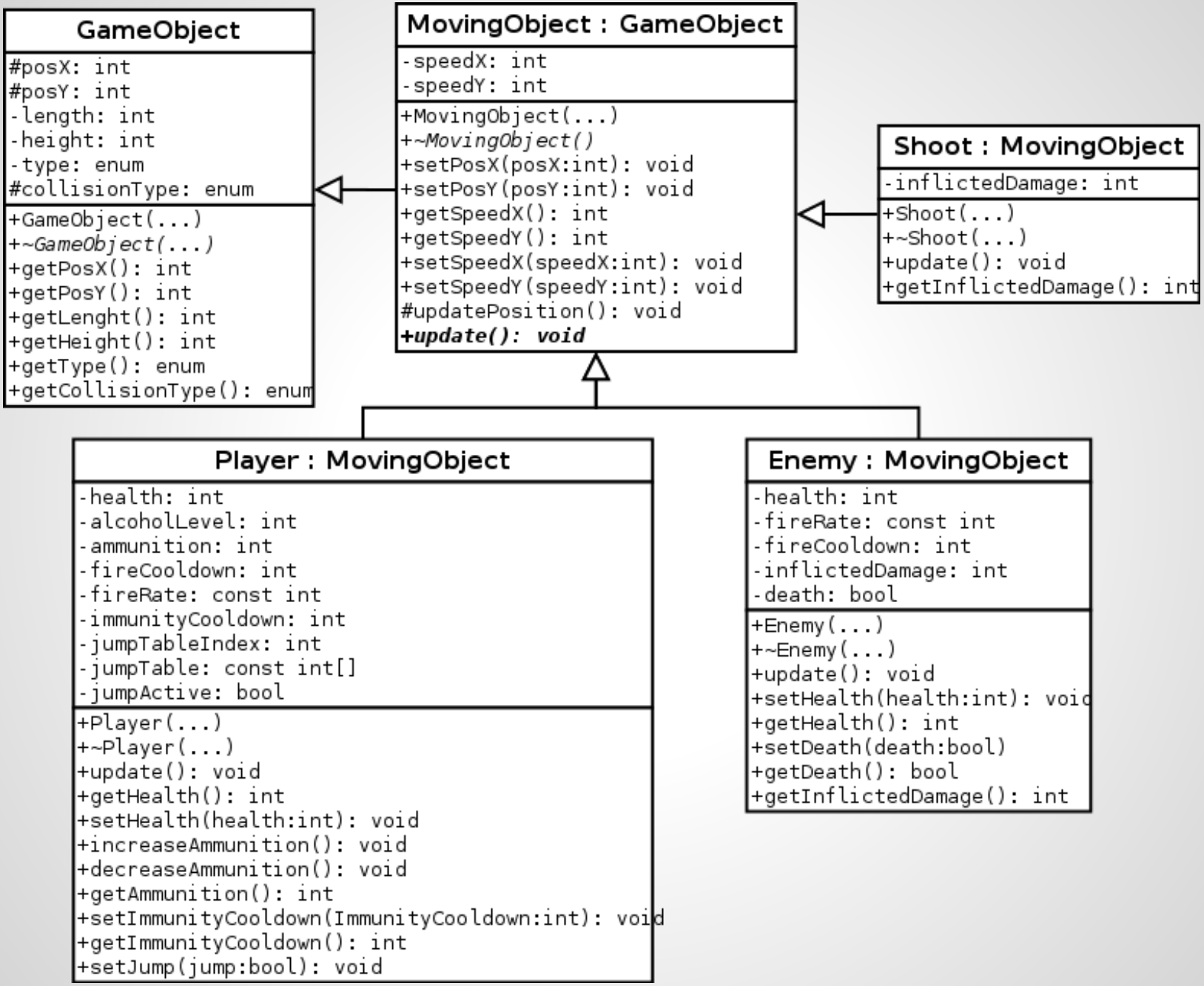

0001670

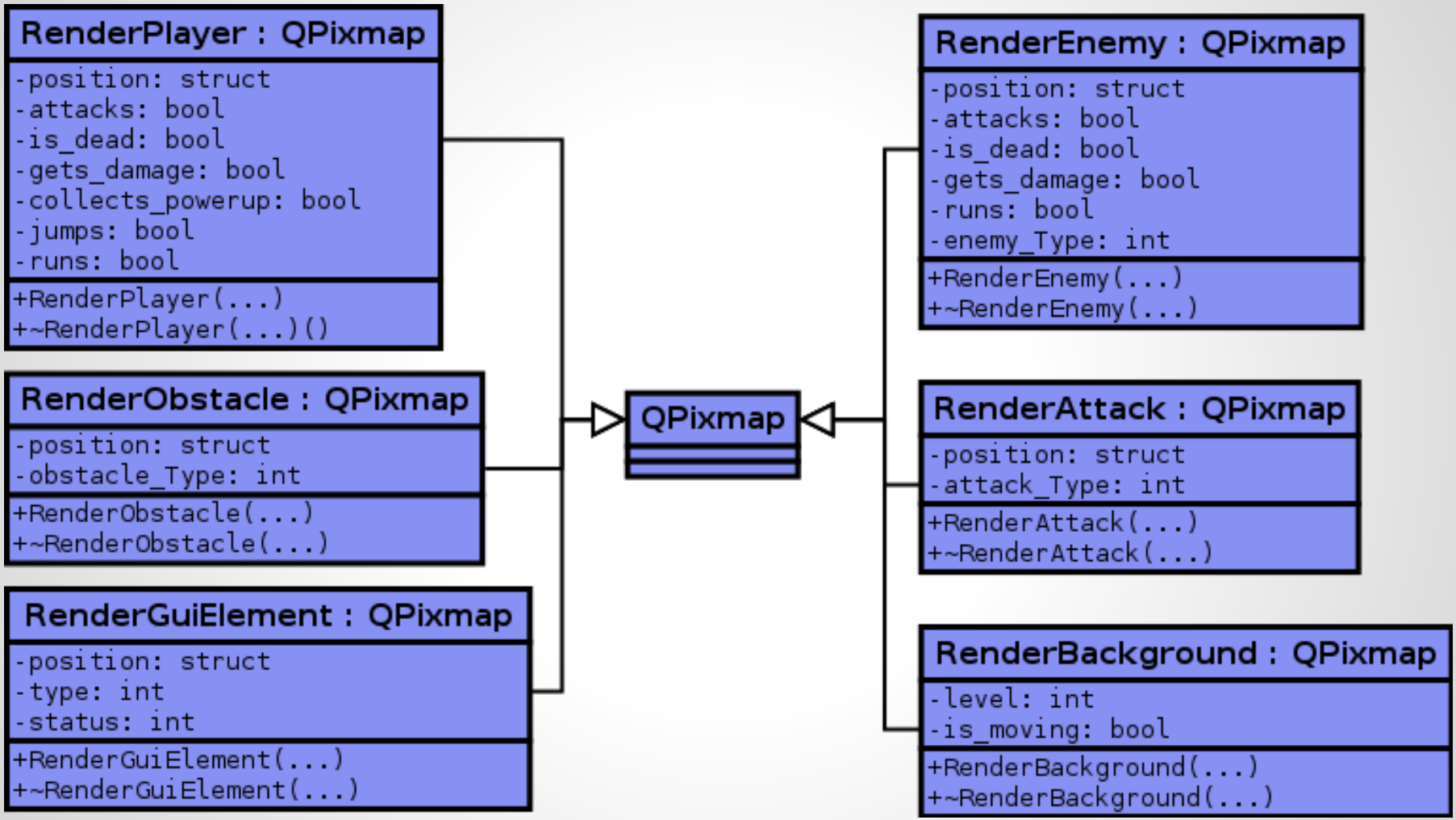
Physik

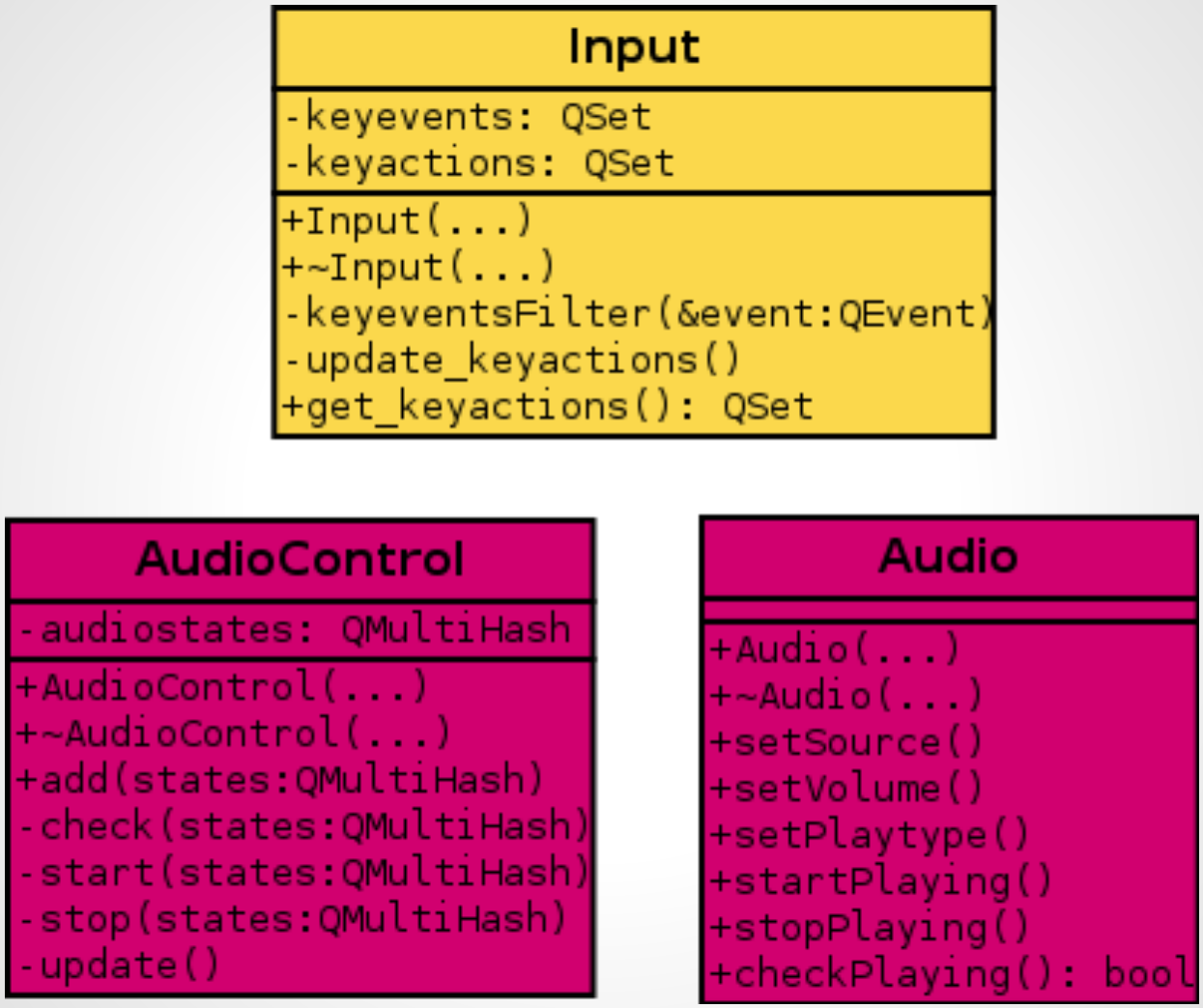
- Bewegungsberechnung
- Kollisionskontrolle
    - keine Floats, nur Integer
- Sprünge über Lookup-Table

Klassen


Game
-worldObjects: QSortedList GameObject
-levelInitial: QSortedList GameObject
-levelSpawn: QSortedList GameObject
-scores: struct
-stepSize: const int
+eventsToHandle: QList struct
-playerObjPointer: *Player
+states: QMultiHash
-keyInputs: Input
+Game(...)
+~Game(...)
+start(): void
+getStepSize(): int
-appendWorldObjects(): void
-reduceWorldObjects(): void
-evaluateInput(): void
-calculateMovement(): void
-detectCollision(): void
-correctMovement(): void
-handleEvents(): void
-renderGraphics(&worldObjects:QSortedList gameObject
position:int): void
-sound(states:QMultiHash): void
-gameEnd()


GameObject
#posX: int
#posY: int
-length: int
-height: int
-type: enum
#collisionType: enum
+GameObject(...)
+~GameObject(...)
+getPosX(): int
+getPosY(): int
+getLenght(): int
+getHeight(): int
+getType(): enum
+getCollisionType(): enum
MovingObject : GameObject
-speedX: int
-speedY: int
+MovingObject(...)
+~MovingObject()
+setPosX(posX:int): void
+setPosY(posY:int): void
+getSpeedX(): int
+getSpeedY(): int
+setSpeedX(speedX:int): void
+setSpeedY(speedY:int): void
#updatePosition(): void
+update(): void
Shoot : MovingObject
-inflictedDamage: int
+Shoot(...)
+~Shoot(...)
+update(): void
+getInflictedDamage(): int
Player : MovingObject
-health: int
-alcoholLevel: int
-ammunition: int
-fireCooldown: int
-fireRate: const int
-immunityCooldown: int
-jumpTableIndex: int
-jumpTable: const int[]
-jumpActive: bool
+Player(...)
+~Player(...)
+update(): void
+getHealth(): int
+setHealth(health:int): void
+increaseAmmunition(): void
+decreaseAmmunition(): void
+getAmmunition(): int
+setImmunityCooldown(ImmunityCooldown:int): void
+getImmunityCooldown(): int
+setJump(jump:bool): void
Enemy : MovingObject
-health: int
-fireRate: const int
-fireCooldown: int
-inflictedDamage: int
-death: bool
+Enemy(...)
+~Enemy(...)
+update(): void
+setHealth(health:int): voic
+getHealth(): int
+setDeath(death:bool)
+getDeath(): bool
+getInflictedDamage(): int


RenderPlayer : QPixmap
-position: struct
-attacks: bool
-is_dead: bool
-gets_damage: bool
-collects_powerup: bool
-jumps: bool
-runs: bool
+RenderPlayer(...)
+~RenderPlayer(...)()
RenderObstacle : QPixmap
-position: struct
-obstacle_Type: int
+RenderObstacle(...)
+~RenderObstacle(...)
RenderGuiElement : QPixmap
-position: struct
-type: int
-status: int
+RenderGuiElement(...)
+~RenderGuiElement(...)
QPixmap
RenderEnemy : QPixmap
-position: struct
-attacks: bool
-is_dead: bool
-gets_damage: bool
-runs: bool
-enemy_Type: int
+RenderEnemy(...)
+~RenderEnemy(...)
RenderAttack : QPixmap
-position: struct
-attack_Type: int
+RenderAttack(...)
+~RenderAttack(...)
RenderBackground : QPixmap
-level: int
-is_moving: bool
+RenderBackground(...)
+~RenderBackground(...)


Input
-keyevents: QSet
-keyactions: QSet
+Input(...)
+~Input(...)
-keyeventsFilter(&event:QEvent)
-update_keyactions()
+get_keyactions(): QSet
AudioControl
-audiostates: QMultiHash
+AudioControl(...)
+~AudioControl(...)
+add(states:QMultiHash)
-check(states:QMultiHash)
-start(states:QMultiHash)
-stop(states:QMultiHash)
-update()
Audio
+Audio(...)
+~Audio(...)
+setSource()
+setVolume()
+setPlaytype()
+startPlaying()
+stopPlaying()
+checkPlaying(): bool

Fragen?