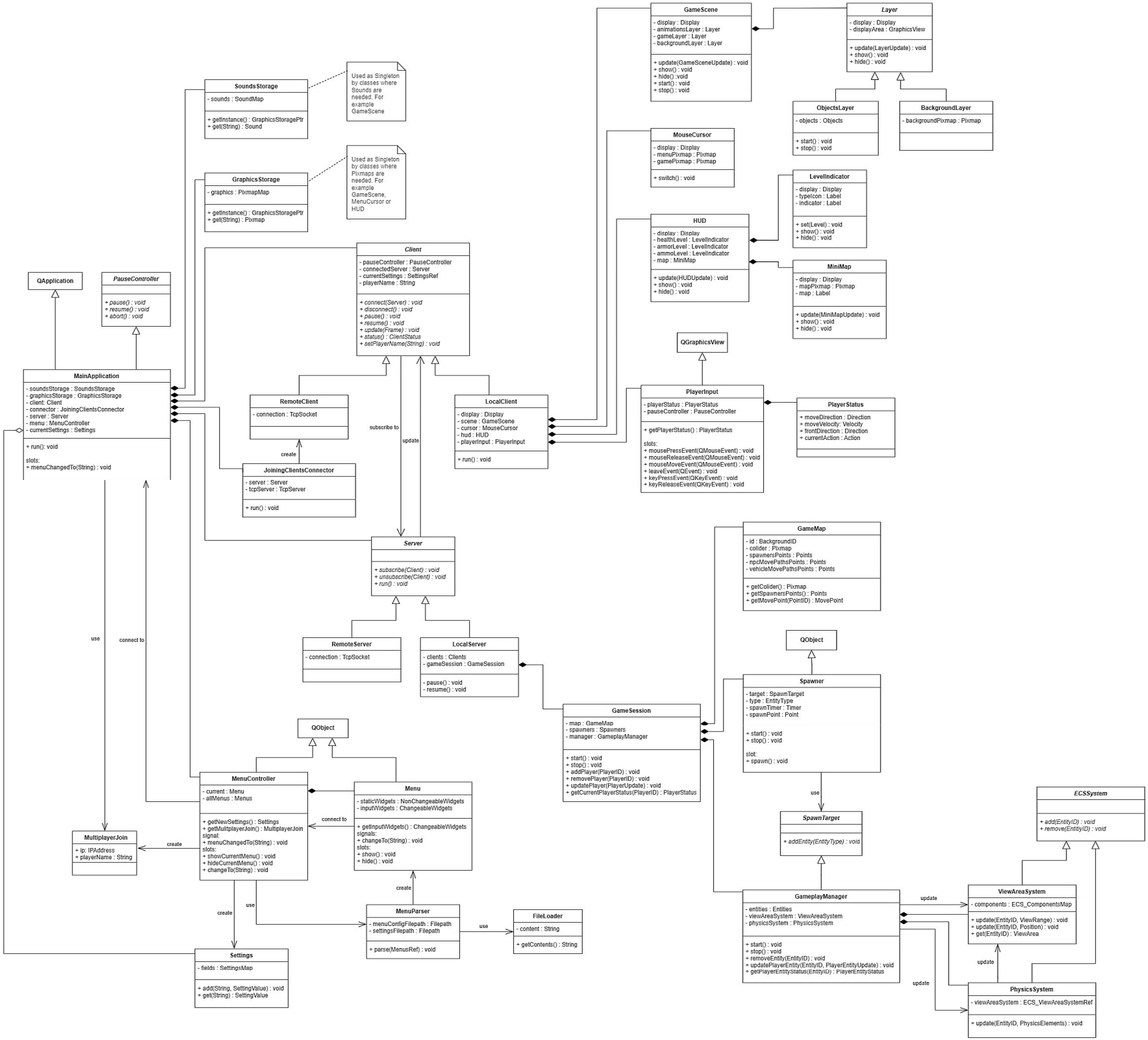
GameScene
- display : Display
- animationsLayer : Layer
- gameLayer : Layer
- backgroundLayer : Layer
+ update(GameSceneUpdate) : void
+ show() : void
+ hide() : void
+ start() : void
+ stop() : void
Layer
- display : Display
- displayArea : GraphicsView
+ update(LayerUpdate) : void
+ show() : void
+ hide() : void
ObjectsLayer
- objects : Objects
+ start() : void
+ stop() : void
BackgroundLayer
- backgroundPixmap : Pixmap
SoundsStorage
- sounds : SoundMap
+ getInstance() : GraphicsStoragePtr
+ get(String) : Sound
Used as Singleton by classes where Sounds are needed. For example GameScene
MouseCursor
- display : Display
- menuPixmap : Pixmap
- gamePixmap : Pixmap
+ switch() : void
GraphicsStorage
- graphics : PixmapMap
+ getInstance() : GraphicsStoragePtr
+ get(String) : Pixmap
Used as Singleton by classes where Pixmaps are needed. For example GameScene, MenuCursor or HUD
LevelIndicator
- display : Display
- typeIcon : Label
- indicator : Label
+ set(Level) : void
+ show() : void
+ hide() : void
HUD
- display : Display
- healthLevel : LevelIndicator
- armorLevel : LevelIndicator
- ammoLevel : LevelIndicator
- map : MiniMap
+ update(HUDUpdate) : void
+ show() : void
+ hide() : void
MiniMap
- display : Display
- mapPixmap : Pixmap
- map : Label
+ update(MiniMapUpdate) : void
+ show() : void
+ hide() : void
Client
- pauseController : PauseController
- connectedServer : Server
- currentSettings : SettingsRef
- playerName : String
+ connect(Server) : void
+ disconnect() : void
+ pause() : void
+ resume() : void
+ update(Frame) : void
+ status() : ClientStatus
+ setPlayerName(String) : void
QApplication
PauseController
+ pause() : void
+ resume() : void
+ abort() : void
QGraphicsView
MainApplication
- soundsStorage : SoundsStorage
- graphicsStorage : GraphicsStorage
- client: Client
- connector : JoiningClientsConnector
- server : Server
- menu : MenuController
- currentSettings : Settings
+ run(): void
slots:
+ menuChangedTo(String) : void
RemoteClient
- connection : TcpSocket
subscribe to
update
LocalClient
- display : Display
- scene : GameScene
- cursor : MouseCursor
- hud : HUD
- playerInput : PlayerInput
+ run() : void
PlayerInput
- playerStatus : PlayerStatus
- pauseController : PauseController
+ getPlayerStatus() : PlayerStatus
slots:
+ mousePressEvent(QMouseEvent) : void
+ mouseReleaseEvent(QMouseEvent) : void
+ mouseMoveEvent(QMouseEvent) : void
+ leaveEvent(QEvent) : void
+ keyPressEvent(QKeyEvent) : void
+ keyReleaseEvent(QKeyEvent) : void
PlayerStatus
+ moveDirection : Direction
+ moveVelocity: Velocity
+ frontDirection : Direction
+ currentAction : Action
create
JoiningClientsConnector
- server : Server
- tcpServer : TcpServer
+ run() : void
Server
+ subscribe(Client) : void
+ unsubscribe(Client) : void
+ run() : void
GameMap
- id : BackgroundID
- colider : Pixmap
- spawnersPoints : Points
- npcMovePathsPoints : Points
- vehicleMovePathsPoints : Points
+ getColider() : Pixmap
+ getSpawnersPoints() : Points
+ getMovePoint(PointID) : MovePoint
use
connect to
RemoteServer
- connection : TcpSocket
LocalServer
- clients : Clients
- gameSession : GameSession
- pause() : void
- resume() : void
QObject
Spawner
- target : SpawnTarget
- type : EntityType
- spawnTimer : Timer
- spawnPoint : Point
+ start() : void
+ stop() : void
slot:
+ spawn() : void
GameSession
- map : GameMap
- spawners : Spawners
- manager : GameplayManager
+ start() : void
+ stop() : void
+ addPlayer(PlayerID) : void
+ removePlayer(PlayerID) : void
+ updatePlayer(PlayerUpdate) : void
+ getCurrentPlayerStatus(PlayerID) : PlayerStatus
QObject
MenuController
- current : Menu
- allMenus : Menus
+ getNewSettings() : Settings
+ getMultiplayerJoin() : MultiplayerJoin
signal:
+ menuChangedTo(String) : void
slots:
+ showCurrentMenu() : void
+ hideCurrentMenu() : void
+ changeTo(String) : void
Menu
- staticWidgets : NonChangeableWidgets
- inputWidgets : ChangeableWidgets
+ getInputWidgets() : ChangeableWidgets
signals:
+ changeTo(String) : void
slots:
+ show() : void
+ hide() : void
connect to
MultiplayerJoin
+ ip: IPAddress
+ playerName : String
create
use
create
create
MenuParser
- menuConfigFilepath : Filepath
- settingsFilepath : Filepath
+ parse(MenusRef) : void
use
FileLoader
- content : String
+ getContents() : String
Settings
- fields : SettingsMap
+ add(String, SettingValue) : void
+ get(String) : SettingValue
use
SpawnTarget
+ addEntity(EntityType) : void
ECSSystem
+ add(EntityID) : void
+ remove(EntityID) : void
GameplayManager
- entities : Entities
- viewAreaSystem : ViewAreaSystem
- physicsSystem : PhysicsSystem
+ start() : void
+ stop() : void
+ removeEntity(EntityID) : void
+ updatePlayerEntity(EntityID, PlayerEntityUpdate) : void
+ getPlayerEntityStatus(EntityID) : PlayerEntityStatus
update
ViewAreaSystem
- components : ECS_ComponentsMap
+ update(EntityID, ViewRange) : void
+ update(EntityID, Position) : void
+ get(EntityID) : ViewArea
update
update
PhysicsSystem
- viewAreaSystem : ECS_ViewAreaSystemRef
+ update(EntityID, PhysicsElements) : void