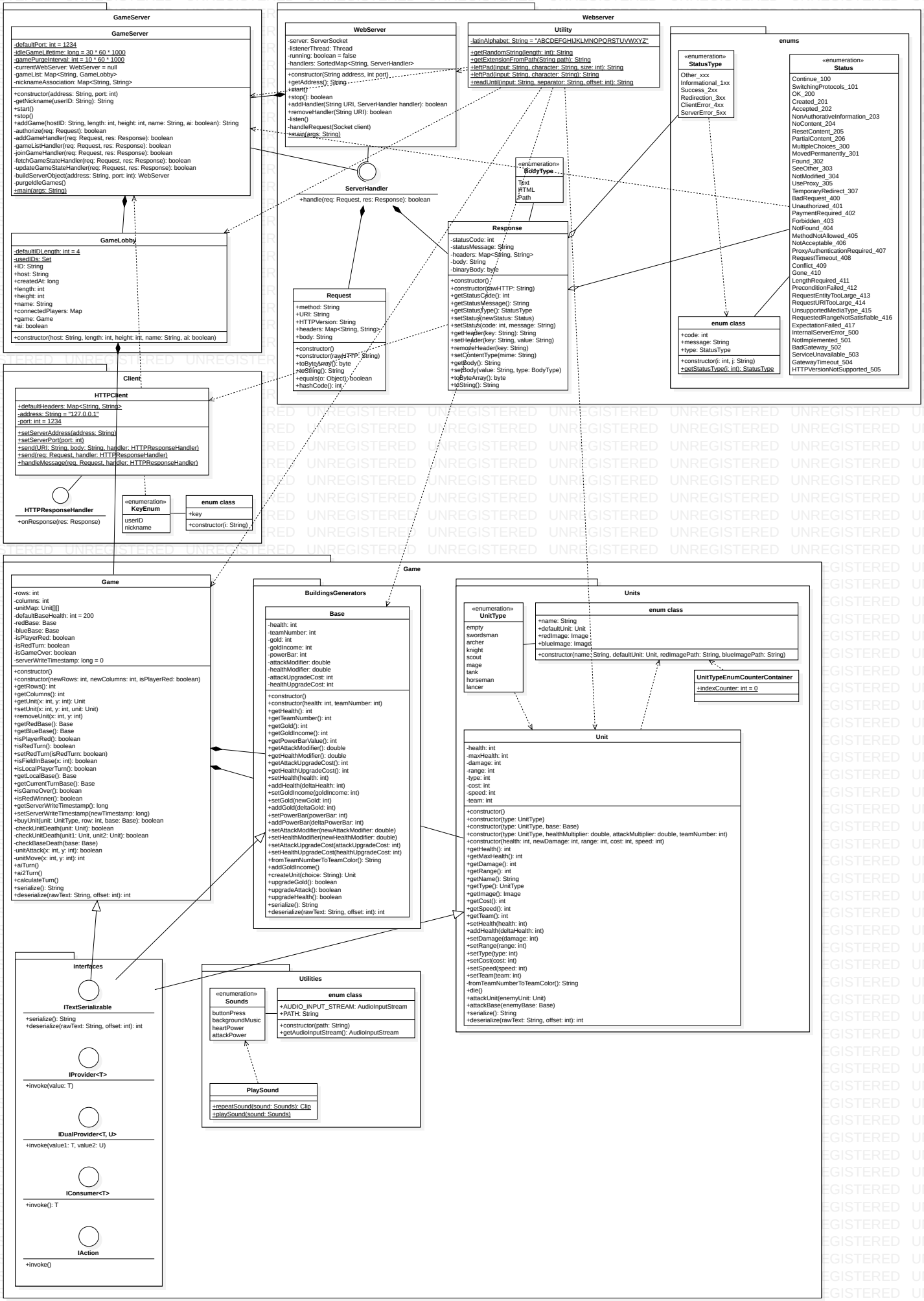
GameServer
GameServer
-defaultPort: int = 1234
-idleGameLifetime: long = 30 * 60 * 1000
-gamePurgeInterval: int = 10 * 60 * 1000
-currentWebServer: WebServer = null
-gameList: Map<String, GameLobby>
-nicknameAssociation: Map<String, String>
+constructor(address: String, port: int)
-getNickname(userID: String): String
+start()
+stop()
+addGame(hostID: String, length: int, height: int, name: String, ai: boolean): String
-authorize(req: Request): boolean
-addGameHandler(req: Request, res: Response): boolean
-gameListHandler(req: Request, res: Response): boolean
-joinGameHandler(req: Request, res: Response): boolean
-fetchGameStateHandler(req: Request, res: Response): boolean
-updateGameStateHandler(req: Request, res: Response): boolean
-buildServerObject(address: String, port: int): WebServer
-purgeIdleGames()
+main(args: String)
GameLobby
-defaultIDLength: int = 4
-usedIDs: Set
+ID: String
+host: String
+createdAt: long
+length: int
+height: int
+name: String
+connectedPlayers: Map
+game: Game
+ai: boolean
+constructor(host: String, length: int, height: int, name: String, ai: boolean)
Webserver
WebServer
-server: ServerSocket
-listenerThread: Thread
-running: boolean = false
-handlers: SortedMap<String, ServerHandler>
+constructor(String address, int port)
+getAddress(): String
+start()
+stop(): boolean
+addHandler(String URI, ServerHandler handler): boolean
+removeHandler(String URI): boolean
-listen()
-handleRequest(Socket client)
+main(args: String)
Utility
-latinAlphabet: String = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
+getRandomString(length: int): String
+getExtensionFromPath(String path): String
+leftPad(input: String, character: String, size: int): String
+leftPad(input: String, character: String): String
+readUntil(input: String, separator: String, offset: int): String
enums
«enumeration»
StatusType
Other_xxx
Informational_1xx
Success_2xx
Redirection_3xx
ClientError_4xx
ServerError_5xx
«enumeration»
Status
Continue_100
SwitchingProtocols_101
OK_200
Created_201
Accepted_202
NonAuthoritativeInformation_203
NoContent_204
ResetContent_205
PartialContent_206
MultipleChoices_300
MovedPermanently_301
Found_302
SeeOther_303
NotModified_304
UseProxy_305
TemporaryRedirect_307
BadRequest_400
Unauthorized_401
PaymentRequired_402
Forbidden_403
NotFound_404
MethodNotAllowed_405
NotAcceptable_406
ProxyAuthenticationRequired_407
RequestTimeout_408
Conflict_409
Gone_410
LengthRequired_411
PreconditionFailed_412
RequestEntityTooLarge_413
RequestURITooLarge_414
UnsupportedMediaType_415
RequestedRangeNotSatisfiable_416
ExpectationFailed_417
InternalServerError_500
NotImplemented_501
BadGateway_502
ServiceUnavailable_503
GatewayTimeout_504
HTTPVersionNotSupported_505
enum class
+code: int
+message: String
+type: StatusType
+constructor(i: int, j: String)
+getStatusType(i: int): StatusType
«enumeration»
BodyType
Text
HTML
Path
ServerHandler
+handle(req: Request, res: Response): boolean
Response
-statusCode: int
-statusMessage: String
-headers: Map<String, String>
-body: String
-binaryBody: byte
+constructor()
+constructor(rawHTTP: String)
+getStatusCode(): int
+getStatusMessage(): String
+getStatusType(): StatusType
+setStatus(newStatus: Status)
+setStatus(code: int, message: String)
+getHeader(key: String): String
+setHeader(key: String, value: String)
+removeHeader(key: String)
+setContentType(mime: String)
+getBody(): String
+setBody(value: String, type: BodyType)
+toByteArray(): byte
+toString(): String
Request
+method: String
+URI: String
+HTTPVersion: String
+headers: Map<String, String>
+body: String
+constructor()
+constructor(rawHTTP: String)
+toByteArray(): byte
+toString(): String
+equals(o: Object): boolean
+hashCode(): int
Client
HTTPClient
+defaultHeaders: Map<String, String>
-address: String = "127.0.0.1"
-port: int = 1234
+setServerAddress(address: String)
+setServerPort(port: int)
+send(URI: String, body: String, handler: HTTPResponseHandler)
+send(req: Request, handler: HTTPResponseHandler)
+handleMessage(req: Request, handler: HTTPResponseHandler)
HTTPResponseHandler
+onResponse(res: Response)
«enumeration»
KeyEnum
userID
nickname
enum class
+key
+constructor(i: String)
Game
Game
-rows: int
-columns: int
-unitMap: Unit[][]
-defaultBaseHealth: int = 200
-redBase: Base
-blueBase: Base
-isPlayerRed: boolean
-isRedTurn: boolean
-isGameOver: boolean
-serverWriteTimestamp: long = 0
+constructor()
+constructor(newRows: int, newColumns: int, isPlayerRed: boolean)
+getRows(): int
+getColumns(): int
+getUnit(x: int, y: int): Unit
+setUnit(x: int, y: int, unit: Unit)
+removeUnit(x: int, y: int)
+getRedBase(): Base
+getBlueBase(): Base
+isPlayerRed(): boolean
+isRedTurn(): boolean
+setRedTurn(isRedTurn: boolean)
+isFieldInBase(x: int): boolean
+isLocalPlayerTurn(): boolean
+getLocalBase(): Base
+getCurrentTurnBase(): Base
+isGameOver(): boolean
+isRedWinner(): boolean
+getServerWriteTimestamp(): long
+setServerWriteTimestamp(newTimestamp: long)
+buyUnit(unit: UnitType, row: int, base: Base): boolean
-checkUnitDeath(unit: Unit): boolean
-checkUnitDeath(unit1: Unit, unit2: Unit): boolean
-checkBaseDeath(base: Base)
-unitAttack(x: int, y: int): boolean
-unitMove(x: int, y: int): int
+aiTurn()
+ai2Turn()
+calculateTurn()
+serialize(): String
+deserialize(rawText: String, offset: int): int
BuildingsGenerators
Base
-health: int
-teamNumber: int
-gold: int
-goldIncome: int
-powerBar: int
-attackModifier: double
-healthModifier: double
-attackUpgradeCost: int
-healthUpgradeCost: int
+constructor()
+constructor(health: int, teamNumber: int)
+getHealth(): int
+getTeamNumber(): int
+getGold(): int
+getGoldIncome(): int
+getPowerBarValue(): int
+getAttackModifier(): double
+getHealthModifier(): double
+getAttackUpgradeCost(): int
+getHealthUpgradeCost(): int
+setHealth(health: int)
+addHealth(deltaHealth: int)
+setGoldIncome(goldIncome: int)
+setGold(newGold: int)
+addGold(deltaGold: int)
+setPowerBar(powerBar: int)
+addPowerBar(deltaPowerBar: int)
+setAttackModifier(newAttackModifier: double)
+setHealthModifier(newHealthModifier: double)
+setAttackUpgradeCost(attackUpgradeCost: int)
+setHealthUpgradeCost(healthUpgradeCost: int)
+fromTeamNumberToTeamColor(): String
+addGoldIncome()
+createUnit(choice: String): Unit
+upgradeGold(): boolean
+upgradeAttack(): boolean
+upgradeHealth(): boolean
+serialize(): String
+deserialize(rawText: String, offset: int): int
Units
«enumeration»
UnitType
empty
swordsman
archer
knight
scout
mage
tank
horseman
lancer
enum class
+name: String
+defaultUnit: Unit
+redImage: Image
+blueImage: Image
+constructor(name: String, defaultUnit: Unit, redImagePath: String, blueImagePath: String)
UnitTypeEnumCounterContainer
+indexCounter: int = 0
Unit
-health: int
-maxHealth: int
-damage: int
-range: int
-type: int
-cost: int
-speed: int
-team: int
+constructor()
+constructor(type: UnitType)
+constructor(type: UnitType, base: Base)
+constructor(type: UnitType, healthMultiplier: double, attackMultiplier: double, teamNumber: int)
+constructor(health: int, newDamage: int, range: int, cost: int, speed: int)
+getHealth(): int
+getMaxHealth(): int
+getDamage(): int
+getRange(): int
+getName(): String
+getType(): UnitType
+getImage(): Image
+getCost(): int
+getSpeed(): int
+getTeam(): int
+setHealth(health: int)
+addHealth(deltaHealth: int)
+setDamage(damage: int)
+setRange(range: int)
+setType(type: int)
+setCost(cost: int)
+setSpeed(speed: int)
+setTeam(team: int)
-fromTeamNumberToTeamColor(): String
+die()
+attackUnit(enemyUnit: Unit)
+attackBase(enemyBase: Base)
+serialize(): String
+deserialize(rawText: String, offset: int): int
interfaces
ITextSerializable
+serialize(): String
+deserialize(rawText: String, offset: int): int
IProvider<T>
+invoke(value: T)
IDualProvider<T, U>
+invoke(value1: T, value2: U)
IConsumer<T>
+invoke(): T
IAction
+invoke()
Utilities
«enumeration»
Sounds
buttonPress
backgroundMusic
heartPower
attackPower
enum class
+AUDIO_INPUT_STREAM: AudioInputStream
+PATH: String
+constructor(path: String)
+getAudioInputStream(): AudioInputStream
PlaySound
+repeatSound(sound: Sounds): Clip
+playSound(sound: Sounds)