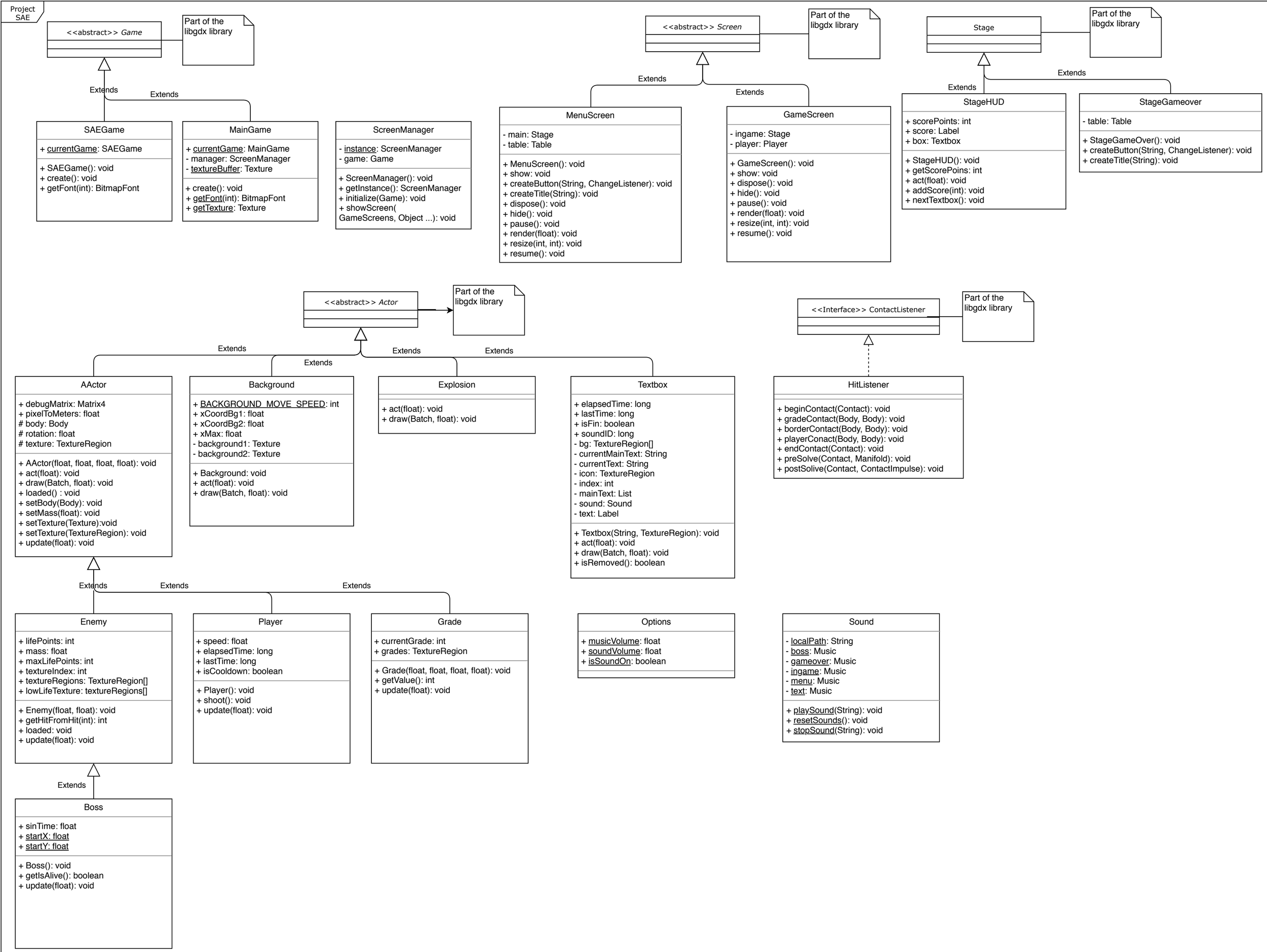
Project SAE


<<abstract>> Game
Part of the libgdx library
Extends
Extends
SAEGame
+ currentGame: SAEGame
+ SAEGame(): void
+ create(): void
+ getFont(int): BitmapFont
MainGame
+ currentGame: MainGame
- manager: ScreenManager
- textureBuffer: Texture
+ create(): void
+ getFont(int): BitmapFont
+ getTexture: Texture
ScreenManager
- instance: ScreenManager
- game: Game
+ ScreenManager(): void
+ getInstance(): ScreenManager
+ initialize(Game): void
+ showScreen(
GameScreens, Object ...): void
<<abstract>> Screen
Part of the libgdx library
Extends
Extends
MenuScreen
- main: Stage
- table: Table
+ MenuScreen(): void
+ show: void
+ createButton(String, ChangeListener): void
+ createTitle(String): void
+ dispose(): void
+ hide(): void
+ pause(): void
+ render(float): void
+ resize(int, int): void
+ resume(): void
GameScreen
- ingame: Stage
- player: Player
+ GameScreen(): void
+ show: void
+ dispose(): void
+ hide(): void
+ pause(): void
+ render(float): void
+ resize(int, int): void
+ resume(): void
Stage
Part of the libgdx library
Extends
Extends
StageHUD
+ scorePoints: int
+ score: Label
+ box: Textbox
+ StageHUD(): void
+ getScorePoins: int
+ act(float): void
+ addScore(int): void
+ nextTextbox(): void
StageGameover
- table: Table
+ StageGameOver(): void
+ createButton(String, ChangeListener): void
+ createTitle(String): void
<<abstract>> Actor
Part of the libgdx library
Extends
Extends
Extends
Extends
AActor
+ debugMatrix: Matrix4
+ pixelToMeters: float
# body: Body
# rotation: float
# texture: TextureRegion
+ AActor(float, float, float, float): void
+ act(float): void
+ draw(Batch, float): void
+ loaded() : void
+ setBody(Body): void
+ setMass(float): void
+ setTexture(Texture):void
+ setTexture(TextureRegion): void
+ update(float): void
Background
+ BACKGROUND_MOVE_SPEED: int
+ xCoordBg1: float
+ xCoordBg2: float
+ xMax: float
- background1: Texture
- background2: Texture
+ Background: void
+ act(float): void
+ draw(Batch, float): void
Explosion
+ act(float): void
+ draw(Batch, float): void
Textbox
+ elapsedTime: long
+ lastTime: long
+ isFin: boolean
+ soundID: long
- bg: TextureRegion[]
- currentMainText: String
- currentText: String
- icon: TextureRegion
- index: int
- mainText: List
- sound: Sound
- text: Label
+ Textbox(String, TextureRegion): void
+ act(float): void
+ draw(Batch, float): void
+ isRemoved(): boolean
<<Interface>> ContactListener
Part of the libgdx library
HitListener
+ beginContact(Contact): void
+ gradeContact(Body, Body): void
+ borderContact(Body, Body): void
+ playerConact(Body, Body): void
+ endContact(Contact): void
+ preSolve(Contact, Manifold): void
+ postSolive(Contact, ContactImpulse): void
Extends
Extends
Extends
Enemy
+ lifePoints: int
+ mass: float
+ maxLifePoints: int
+ textureIndex: int
+ textureRegions: TextureRegion[]
+ lowLifeTexture: textureRegions[]
+ Enemy(float, float): void
+ getHitFromHit(int): int
+ loaded: void
+ update(float): void
Player
+ speed: float
+ elapsedTime: long
+ lastTime: long
+ isCooldown: boolean
+ Player(): void
+ shoot(): void
+ update(float): void
Grade
+ currentGrade: int
+ grades: TextureRegion
+ Grade(float, float, float, float): void
+ getValue(): int
+ update(float): void
Options
+ musicVolume: float
+ soundVolume: float
+ isSoundOn: boolean
Sound
- localPath: String
- boss: Music
- gameover: Music
- ingame: Music
- menu: Music
- text: Music
+ playSound(String): void
+ resetSounds(): void
+ stopSound(String): void
Extends
Boss
+ sinTime: float
+ startX: float
+ startY: float
+ Boss(): void
+ getIsAlive(): boolean
+ update(float): void