Diagrama de Classes - Projeto P.O.O. Frase Animado game : GameManage letraDigitada: char - varF: float - tamanhoChar: int ogLetrasPosX : float escalaLogo : float minuscula: char frase: StringBuffer loop: boolear + pontos : int escalaLogolndoVindo : boolean - frasePos: Vector2 play: boolean boxSize : Vector2 vida : int offsetX: float sequencia: boolean[] originX : float acertosConsecutivos : int offsetY: float[][] + Animado(varF: float, loop: boolean, sequencias: int) boxY : float + multiplicador : int animadoX: Animado boxX : float[] digitou : boolean + intervalo(start: float, end: float, step: float, indiceSequencia: int): float animadoY: ArrayList<ArrayList<Animado> margemBox : float limiteLinha: int play(): void textY : float + Player() indiceLinha: int textX : float[1 + confereAcerto(game : GameManager, Frase : Frase) : void button : Rectangle[] linhaSemEspaco: StringBuffer buttonHover : boolean[] multiplicaPontos(): void + linhaAtual: int animandoPassword : boolean bam : Music texArray: TextureRegion[errou: boolean started : boolean animacao: Animation + linha: ArrayList<StringBuffer> MainMenu(game : GameManager) + Frase(frase: String) stateTime: float recalcularBoxPos(): void criandoLinhas(): void width: float renderBoxes() : void criarLinha(indice: int): void + height: float + treme(velocidade: float): void render(delta : float) : void animaLinha(): void + Spritesheet(textura: Texture, frames: int, width: int, height: int, linhas: int, checarTecla(libgdxChar:int, asciiCharDiferenca:int):void dispose(): void apagarLinhas(): void + play(batch: SpriteBatch, x: float, y: float, loop: boolean): void + play(batch: SpriteBatch, x: float, y: float, loop: boolean, flag: boolean): void + getFrame(batch: SpriteBatch, n: int, x: float, y: float, w: float, h: float): void reset(): void password : String GameMan + frase : Frase textAnimAcerto : Texture bgm : Music cursor : Texture nausado : hoolean NineSlice levelJD : LevelJSONData titulo : String mousePos: Vector2 cantoOffsetYL : int y: float[] batch : SpriteBatch texArray: TextureRegion[] fontP2white : BitmapFont velocidadeAnimacaoBG : int animacao: Animation fontP2black : BitmapFont pausaTimer : int + NineSlice(textura: Texture, cantoWidth: float, cantoHeight: float) fontP2grey : BitmapFont btnCooldown : int introTimer : int + draw(batch: SpriteBatch, x: float, y float, width: float, height: float): void fontArmaFiveBlack : BitmapFont timer : int[] fontArmaEiveWhite : BitmanEont bgOffset : int[] sfxOn : boolean piscaVida : boolean started : boolean player : Player LevelJSONData gameOverScreen : boolean titulo : ArrayList<String> + hud : Texture sucessoScreen : boolean rect : Texture frase : ArrayList<String> + password : ArrayList<String> hudHeight: float div : Texture timer : ArrayList<String> boxX : float[] + heart : Texture maxVida : ArrayList<String boxY : float[] bgLetras : Texture bg : ArrayList<String> boxW : float[] boxPq : Texture + bgm : ArrayList<String> boxH : float[] boxGd : Texture arteCantos : ArrayList<String> milisegundos : long + boxPausa : Texture bgTex : Map<String, Texture> segundos : long logo : Texture arteCantosTex : Map<String, Texture> segundoAnterior : long timerBG : Texture bgmFile : Map<String, Music> placar : StringBuilder cursorState : Spritesheet btnOpcoesRect : Rectangle animAcerto : Spritesheet read(json : Json, jsonData : JsonValue) : void bg : Texture dispose(): void ► height : int popularMapTex(name: ArrayList<String>, map : Map<String, Texture>) : void boxBtn : Rectangle[] acertoSnd : Sound[] popularMapMusic(name: ArrayList<String>, map : Map<String, Music>) : void + Level(game : GameManager, LJD : LevelJSONData, i : int) erroSnd : Sound[] disposeMapTex(name: ArrayList<String>, map : Map<String, Texture>) : void disposeMapMusic(name: ArrayList<String>, map : Map<String, Music>) : void render (delta : float) : void levels : ArrayList<Level> animarBG(): void hideCursor : boolean random : Random setBoxes(): void gameOver(): void btn : NineSlice fimDaTela(): void create(): void porCentoH(h : float) : float fontSize(f : BitmapFont) : void trocaCursor(): void render() : void + dispose() : void