

BLACK HOLE GLOW

Arquitectura del Proyecto

Version 4.0.1

Diciembre 2024

Paquete: com.secret.blackholeglow

Plataforma: Android (Java)

OpenGL: ES 2.0 / 3.0

176 clases | 16 paquetes | 3 escenas activas

Tabla de Contenidos

1. Resumen Ejecutivo
2. Estructura de Carpetas
3. Arquitectura General
4. Diagramas UML
5. Catalogo de Clases
 - 5.1 Activities
 - 5.2 Core - Pipeline de Renderizado
 - 5.3 Scenes - Escenas
 - 5.4 OpenGL Base
 - 5.5 Objetos 3D - Espacio
 - 5.6 Objetos 3D - Navidad
 - 5.7 Sistema de Combate
 - 5.8 Sistema de Musica
 - 5.9 UI y HUD
 - 5.10 Systems - Gestores Globales
6. Clases Obsoletas
7. Flujo de Datos
8. Guia para IAs

1. Resumen Ejecutivo

Que es Black Hole Glow?

Black Hole Glow es un Live Wallpaper para Android que renderiza escenas 3D animadas usando OpenGL ES. El usuario puede seleccionar diferentes escenas tematicas que se muestran como fondo de pantalla animado.

Caracteristicas Principales

Feature	Descripcion
Escenas 3D	Renderizado OpenGL ES 2.0/3.0
Musica Reactiva	Visualizacion de audio en tiempo real
Sistema de Combate	Meteoritos, naves, escudos
Nieve Interactiva	Particulas GPU (escena Navidena)
IA Gemini	Saludos personalizados
Firebase	Auth, stats, remote config

Escenas Disponibles (Diciembre 2024)

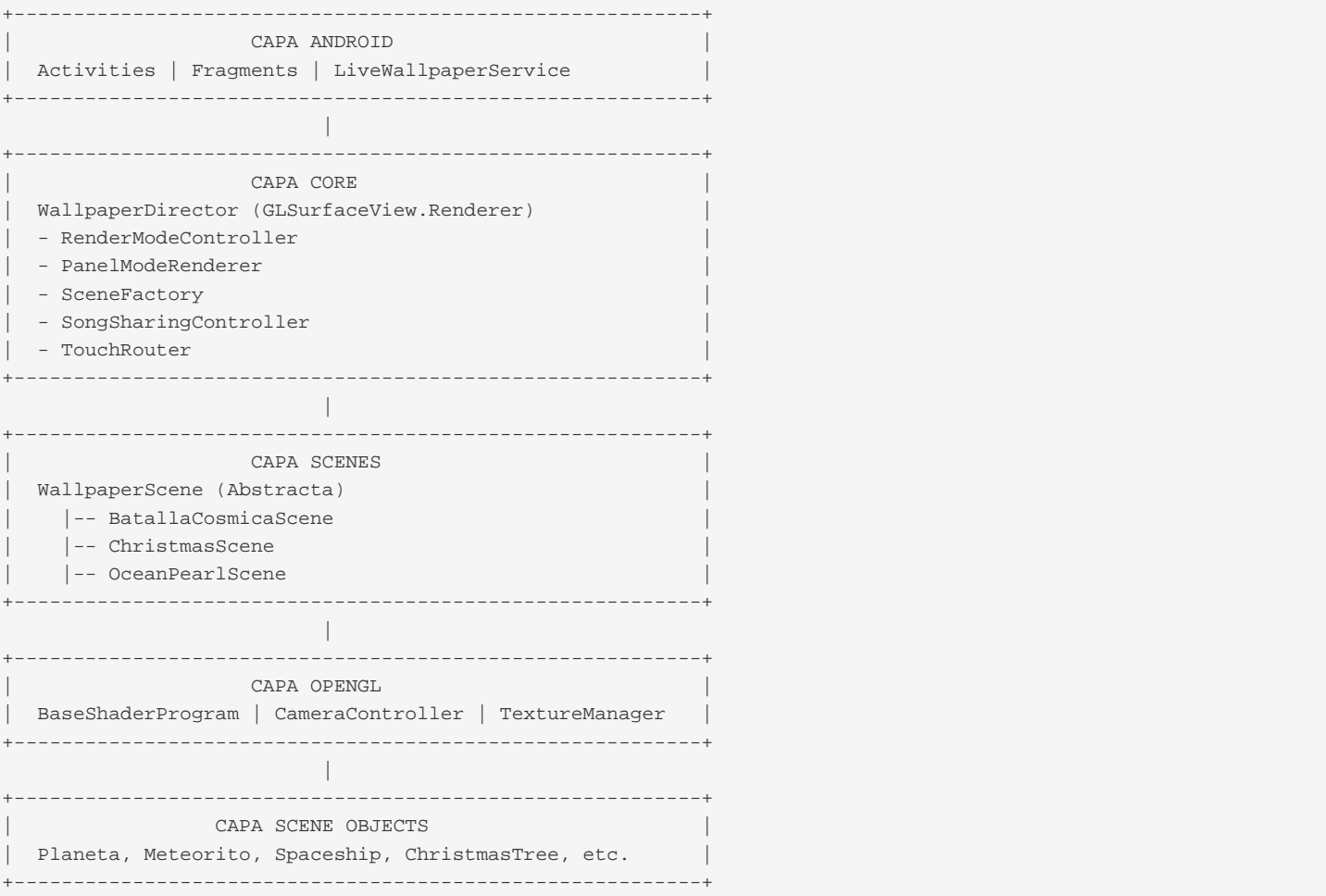
Escena	Estado	Clase
Batalla Cosmica	Activa	BatallaCosmicaScene
Bosque Navideno	Activa	ChristmasScene
Ocean Pearl	Coming Soon	OceanPearlScene
La Mansion	Coming Soon	No implementada

2. Estructura de Carpetas

```
blackholeglow/  
  app/src/main/  
    java/com/secret/blackholeglow/  
      activities/      - Activities de Android  
      adapters/        - Adaptadores RecyclerView  
      christmas/       - Componentes escena navidena  
      core/            - Pipeline de renderizado  
      effects/         - Efectos post-proceso  
      fragments/       - Fragments UI  
      gl3/             - Utilidades OpenGL 3.0  
      models/          - Modelos de datos  
      opengl/          - Componentes OpenGL  
      scenes/          - Escenas de wallpaper  
      sharing/         - Sistema compartir musica  
      systems/         - Sistemas globales  
      ui/              - Componentes UI custom  
      util/            - Utilidades (OBJ loader)  
      wallpaper/       - Gestion de wallpapers  
      *.java           - Clases paquete raiz  
  assets/  
    shaders/          - Shaders GLSL  
    *.obj             - Modelos 3D
```

3. Arquitectura General

Diagrama de Capas



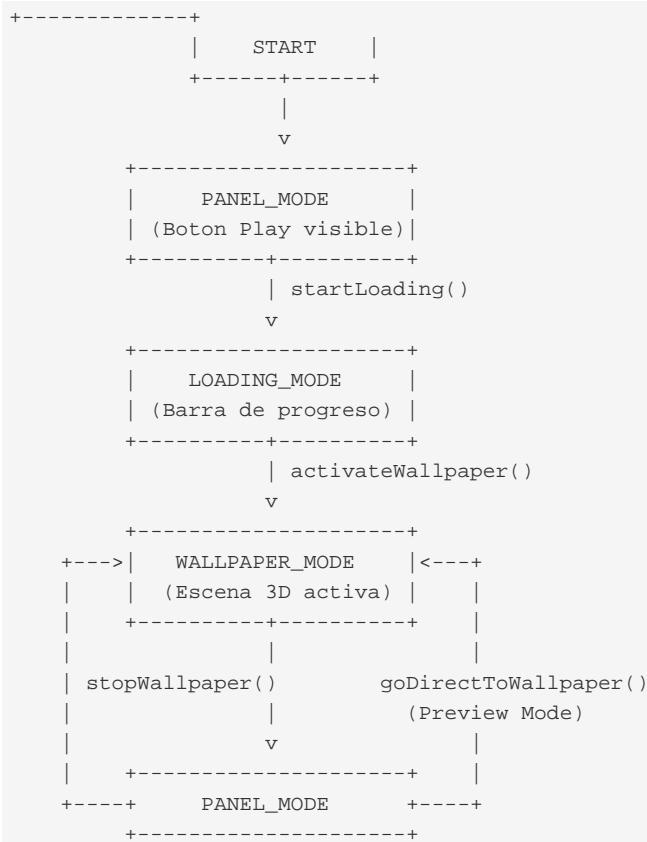
Patron Actor Model

El proyecto usa el patron Actor donde cada componente tiene una responsabilidad unica:

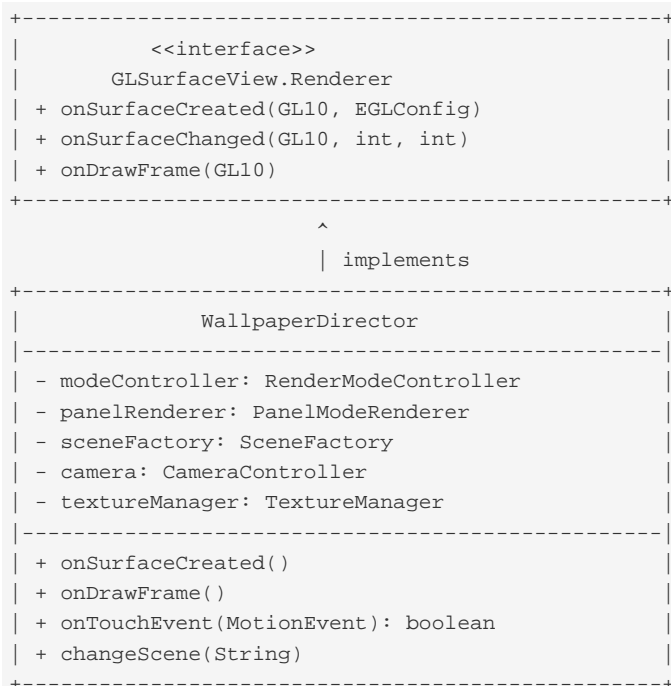
Actor	Responsabilidad
WallpaperDirector	Orquesta todo, implementa Renderer
RenderModeController	Maquina de estados
PanelModeRenderer	UI del panel de control
SceneFactory	Crea y destruye escenas
SongSharingController	Musica y Gemini AI
TouchRouter	Distribuye eventos tactiles

4. Diagramas UML

4.1 Diagrama de Estados - RenderModeController



4.2 Diagrama de Clases - Core



4.3 Diagrama de Clases - Scenes



5. Catalogo de Clases

5.1 Activities (5 clases)

Clase	Proposito
SplashActivity	Pantalla inicio con logo animado
MainActivity	Pantalla principal con NavigationDrawer
WallpaperPreviewActivity	Preview antes de aplicar wallpaper
WallpaperLoadingActivity	Pantalla de carga
GeminiChatActivity	Chat con Gemini AI

5.2 Core - Pipeline de Renderizado (7 clases)

Clase	Proposito
WallpaperDirector	Director principal - GLSurfaceView.Rende
RenderModeController	Maquina de estados (PANEL/LOADING/WALLPA
PanelModeRenderer	Renderiza UI del panel
SceneFactory	Factory para crear/destruir escenas
SongSharingController	Gestion musica y Gemini AI
TouchRouter	Distribuye eventos tactiles
ResourcePreloader	Precarga recursos en background

5.3 Scenes - Escenas (8 clases)

Clase	Proposito
WallpaperScene	Clase base abstracta para escenas
BatallaCosmicaScene	Batalla espacial con meteoritos
ChristmasScene	Bosque navideno con nieve
OceanPearlScene	Escena submarina (Coming Soon)
SceneConstants	Constantes configurables por escena
SceneManager	Gestiona cambios entre escenas
SceneCallbacks	Interface de callbacks
Disposable	Interface para liberar recursos

5.4 OpenGL Base

Clase	Proposito
BaseShaderProgram	Clase base para shaders GLSL
CameraController	Sistema camara multi-modo
TextureManager	Cache lazy-loading de texturas
ShaderUtils	Utilidades compilar shaders
SceneObject	Interface - update() y draw()
CameraAware	Interface - objetos que necesitan camara

5.5 Objetos 3D - Espacio (~30 clases)

Planetas y Cuerpos Celestes

Clase	Proposito
Planeta	Planeta generico con orbita y textura
TierraLiveHD	Tierra con texturas HD
TierraMeshy	Tierra modelo Meshy AI
SolMeshy	Sol con modelo 3D
SolRealista	Sol procedural con corona
SaturnoMeshy	Saturno con anillos

Meteoritos y Particulas

Clase	Proposito
Meteorito	Meteorito individual
MeteorShower	Sistema lluvia de meteoros
MeteorTrail	Estela de fuego
MeteorExplosion	Explosion al impactar
SpaceComets	Cometas espaciales
SpaceDust	Polvo cosmico

Naves Espaciales

Clase	Proposito
Spaceship	Nave espacial base
Spaceship3D	Nave 3D con modelo OBJ
UfoAttacker	OVNI atacante con IA
UfoScout	OVNI explorador
DefenderShip	Nave defensora humana
SpaceStation	Estacion espacial

5.6 Objetos 3D - Navidad (4 clases)

Clase	Proposito
ChristmasTree	Arbol 3D con modelo OBJ, animacion vient
ChristmasBackground	Fondo bosque nevado
SnowGround	Suelo con textura nieve
SnowParticles	Sistema particulas nieve GPU

5.7 Sistema de Combate (~15 clases)

Clase	Proposito
CollisionSystem	Deteccion de colisiones
EarthShield	Escudo protector Tierra
Laser	Proyectil laser
PlasmaBeamWeapon	Arma rayo plasma
Projectile	Proyectil generico
ProjectilePool	Object pool (optimizacion)
TargetingSystem	Sistema apuntado automatico
EnemyAI	IA de enemigos
BattleHUD	HUD de batalla

5.8 Sistema de Musica (~10 clases)

Clase	Proposito
MusicVisualizer	Analisis audio FFT tiempo real
MusicReactive	Interface objetos reactivos
MusicStars	Estrellas pulsas con beat
MusicIndicator	Ecualizador visual barras
EqualizerBarsDJ	Barras ecualizador estilo DJ
MusicSystem	Sistema central musica

5.9 UI y HUD (~15 clases)

Clase	Proposito
HealthBar / HPBar	Barra de vida
ComboBar	Barra de combo
LoadingBar	Barra carga animada
HolographicTitle	Titulo efecto holografico
PlayPauseButton	Boton play/pause animado
MiniStopButton	Boton detener wallpaper
SimpleTextRenderer	Renderizador texto OpenGL

5.10 Systems - Gestores Globales (16 clases)

Clase	Proposito
WallpaperCatalog	Catalogo wallpapers disponibles
WallpaperPreferences	Preferencias SharedPreferences
EventBus	Bus eventos comunicacion desacoplada
GLStateManager	Estado global OpenGL
ScreenManager	Dimensiones y orientacion
ResourceManager	Gestion recursos
AdsManager	Gestion anuncios
SubscriptionManager	Suscripciones premium
RemoteConfigManager	Firebase Remote Config
FirebaseQueueManager	Batching operaciones Firebase

6. Clases Obsoletas

6.1 Completamente Huerfanos (Sin referencias)

Estas clases NO estan siendo usadas en ningun lugar del codigo:

Clase	Razon
ArcadePreview.java	Comentada como REMOVIDO
EstrellaBailarina.java	Feature removida
CloudLayer.java	Sin imports ni instanciaciones
SunHeatEffect.java	Comentada como REMOVIDO
CircularLoadingRing.java	Sin referencias
DiscoBallShaderProgram.java	Sin referencias

6.2 Features Removidas (Codigo comentado)

Clase	Razon
EarthShield.java	Comentada en BatallaCosmicaScene
ForceField.java	Nunca instanciada
BirthdayManager.java	Feature removida
BirthdayMarquee.java	Feature removida
LeaderboardManager.java	Feature removida
MagicLeaderboard.java	Feature removida
HoroscopeManager.java	Inicializacion comentada
HoroscopeDisplay.java	Inicializacion comentada

Total: ~14 clases que podrian eliminarse para limpiar el proyecto

7. Flujo de Datos

7.1 Flujo de Inicio de Wallpaper

```
Usuario selecciona wallpaper en MainActivity
|
v
WallpaperAdapter.onClick()
|
| Guarda preferencia
v
WallpaperPreferences.setSelectedWallpaper("Batalla Cosmica")
|
v
Intent -> WallpaperPreviewActivity
|
| Usuario presiona "Definir fondo"
v
ACTION_CHANGE_LIVE_WALLPAPER
|
v
Android crea LiveWallpaperService
|
| onCreateEngine()
v
new GLWallpaperEngine()
|
v
WallpaperDirector.onSurfaceCreated()
|
| Lee preferencia
| SceneFactory.createScene("Batalla Cosmica")
v
BatallaCosmicaScene.onCreate()
|
v
Loop de renderizado activo (60 FPS)
```

7.2 Flujo de Render Loop

```
onDrawFrame() [60 FPS]
|
| 1. GLStateManager.beginFrame()
|    - Calcula deltaTime, limpia buffers
|
| 2. switch(currentMode):
|
|    PANEL_MODE:
|      - panelRenderer.updatePanelMode(dt)
|      - panelRenderer.drawPanelMode()
|
|    LOADING_MODE:
|      - panelRenderer.updateLoadingMode(dt)
|      - panelRenderer.drawLoadingMode()
|      - checkLoadingComplete()
|
|    WALLPAPER_MODE:
|      - musicVisualizer.getFrequencyBands()
|      - scene.updateMusicBands(bands)
|      - sceneFactory.updateCurrentScene(dt)
|        -> [cada objeto] obj.update(dt)
|      - bloomEffect.beginCapture()
|      - sceneFactory.drawCurrentScene()
|        -> [cada objeto] obj.draw()
|      - screenEffects.draw()
|      - bloomEffect.endCaptureAndApply()
|      - panelRenderer.drawWallpaperOverlay()
|      - songSharing.draw()
```


8. Guia para IAs

8.1 Contexto Rapido

```

PROYECTO: Black Hole Glow
TIPO: Android Live Wallpaper
LENGUAJE: Java (NO Kotlin)
RENDERING: OpenGL ES 2.0/3.0
ARQUITECTURA: Actor Model + Scene Graph

ENTRY POINT: LiveWallpaperService.java
DIRECTOR: WallpaperDirector.java
ESCENAS: Extienden WallpaperScene.java
OBJETOS 3D: Implementan SceneObject interface

```

8.2 Patrones Usados

Patron	Implementacion
Singleton	WallpaperCatalog, EventBus, GLStateManag
Factory	SceneFactory crea escenas por nombre
Observer	EventBus comunicacion desacoplada
State Machine	RenderModeController
Scene Graph	WallpaperScene contiene SceneObjects
Object Pool	ProjectilePool reutiliza proyectiles

8.3 Archivos Importantes

Archivo	Importancia
WallpaperDirector.java	***** Core del renderizado
WallpaperScene.java	***** Base de todas las escenas
SceneFactory.java	**** Creacion de escenas
BatallaCosmicaScene.java	**** Escena principal
WallpaperCatalog.java	*** Catalogo de wallpapers
LiveWallpaperService.java	*** Entry point Android

8.4 Como Agregar una Nueva Escena

```
// 1. Crear clase que extienda WallpaperScene
public class MiNuevaScene extends WallpaperScene {

    @Override
    public String getName() { return "Mi Nueva Escena"; }

    @Override
    public String getDescription() { return "Descripcion..."; }

    @Override
    public int getPreviewResourceId() {
        return R.drawable.mi_preview;
    }

    @Override
    protected void setupScene() {
        // Crear objetos 3D aqui
        MiObjeto obj = new MiObjeto(context, textureManager);
        addSceneObject(obj);
    }

    @Override
    protected void releaseSceneResources() {
        // Liberar recursos especificos
    }
}

// 2. Registrar en SceneFactory.registerDefaultScenes()
registerScene("Mi Nueva Escena", MiNuevaScene.class);

// 3. Agregar al catalogo en WallpaperCatalog
catalog.add(new WallpaperItem.Builder("Mi Nueva Escena")
    .descripcion("...")
    .preview(R.drawable.mi_preview)
    .sceneName("Mi Nueva Escena")
    .tier(WallpaperTier.FREE)
    .build());
```

8.5 Preguntas Frecuentes

P: Donde esta el main loop de renderizado?

R: WallpaperDirector.onDrawFrame()

P: Como se cambia de escena?

R: WallpaperDirector.changeScene("nombre") -> SceneFactory.createScene()

P: Donde se guardan las preferencias?

R: WallpaperPreferences usa SharedPreferences

P: Como funciona la musica reactiva?

R: MusicVisualizer analiza audio -> pasa bandas de frecuencia a la escena -> objetos reaccionan

P: Donde estan los shaders?

R: app/src/main/assets/shaders/*.glsl

Fin del Documento

Generado: Diciembre 2024

Proyecto: Black Hole Glow v4.0.1

Clases totales: 176

Clases obsoletas identificadas: ~14