Key Concepts

Intro to LibGdx

Installing/Setup

Android Studio

Essential Classes

Device Manager

Hello World

What is **LibGDX?**

LibGDX is a **free** and **open source** 2D game framework built within **Android Studio**

Where is **LibGDX?**

libgdx.badlogicgames.com/download.html

Download the **Setup App** to get started

Installing LibGdx

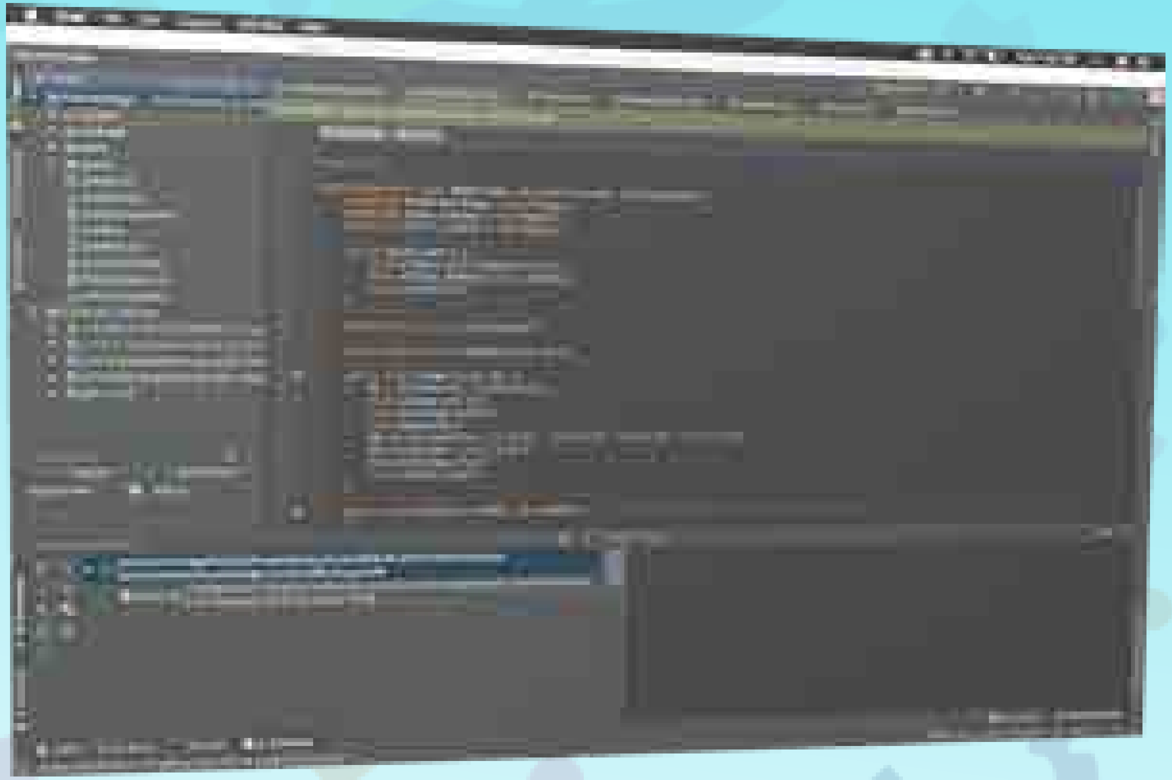
Open the **gdx-setup.jar** executable

.JAR setup

You wanna make sure you have Java installed before opening the .jar file

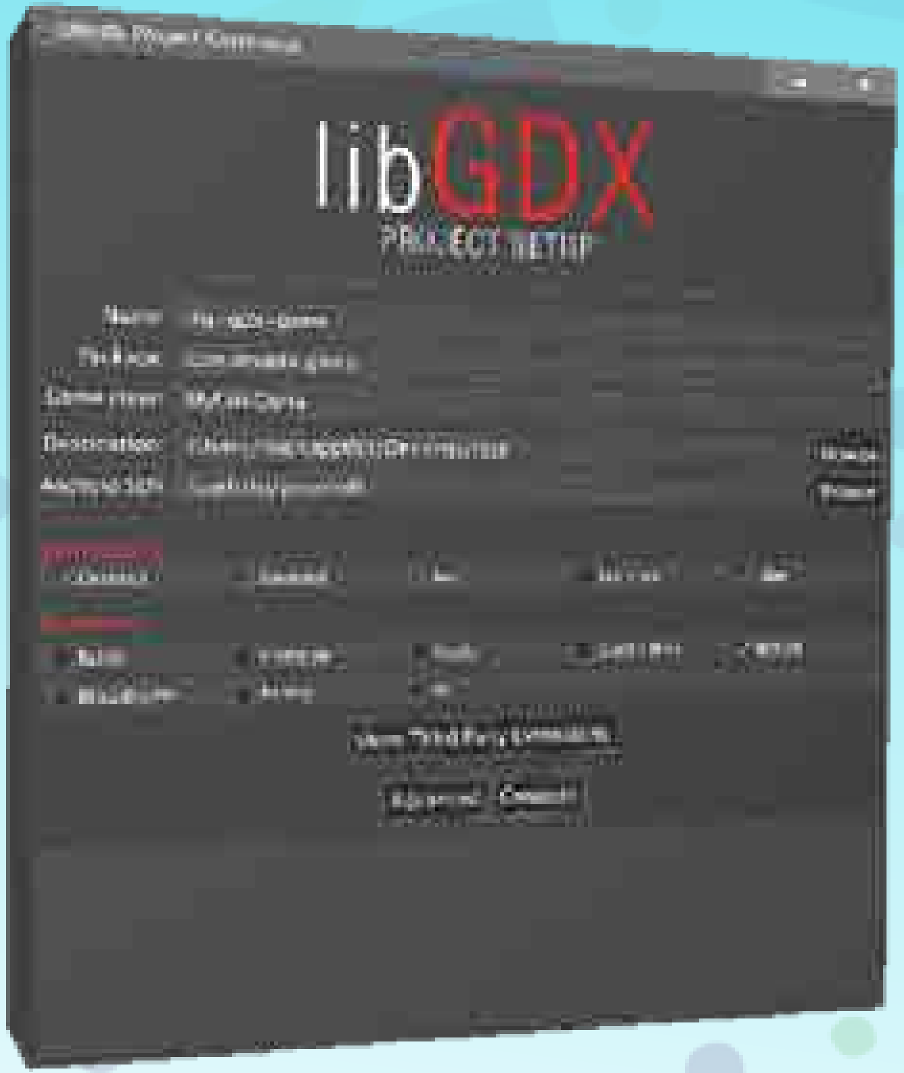
Android Studio

Begin to explore Android Studio and see what you are familiar with



Project Setup

Insert the path to the Android SDK and check off only ‘Android’ to generate



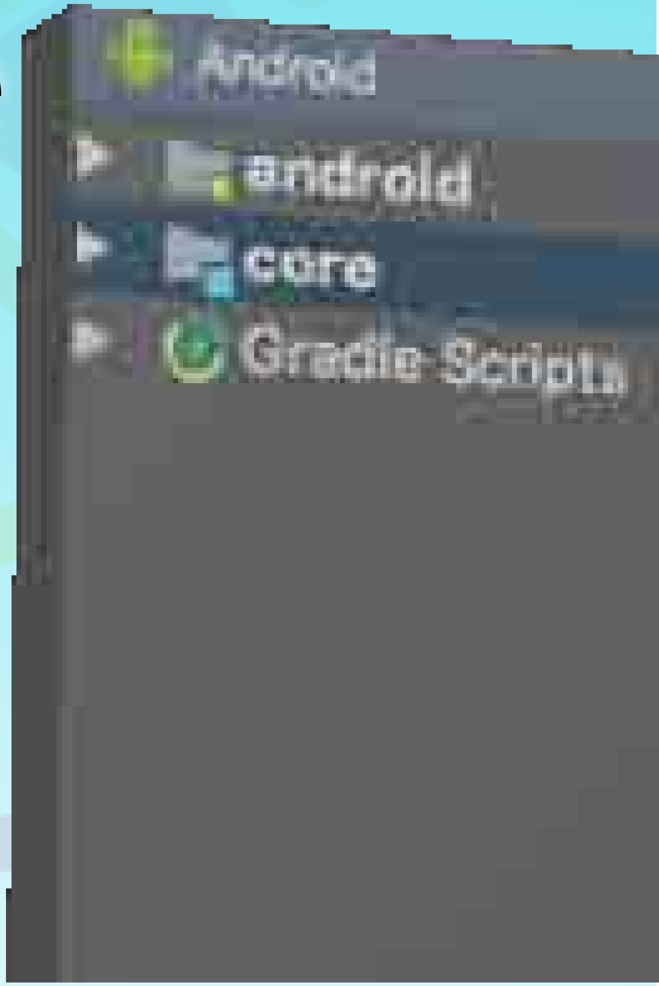
Android SDK

Find the location path in Android Studio

Search **‘Andriod Studio’** within preferences

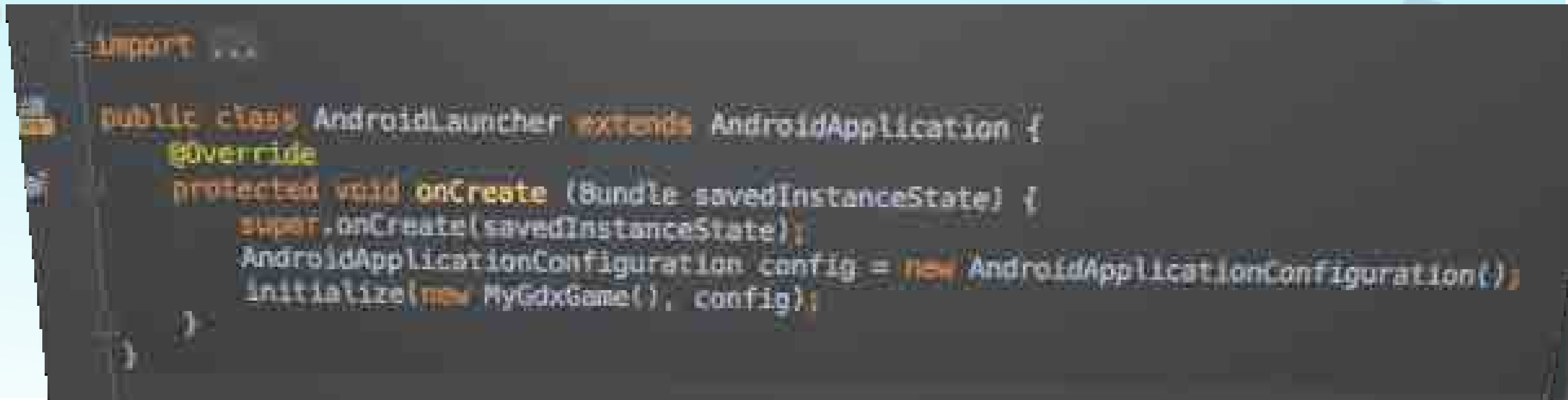
Open Project

Core folder will manage the game itself, while **Android** will handle native Android configuration



Essential Classes

AndroidLauncher class is used as the **“main”** to initalize and run a new instance of our game

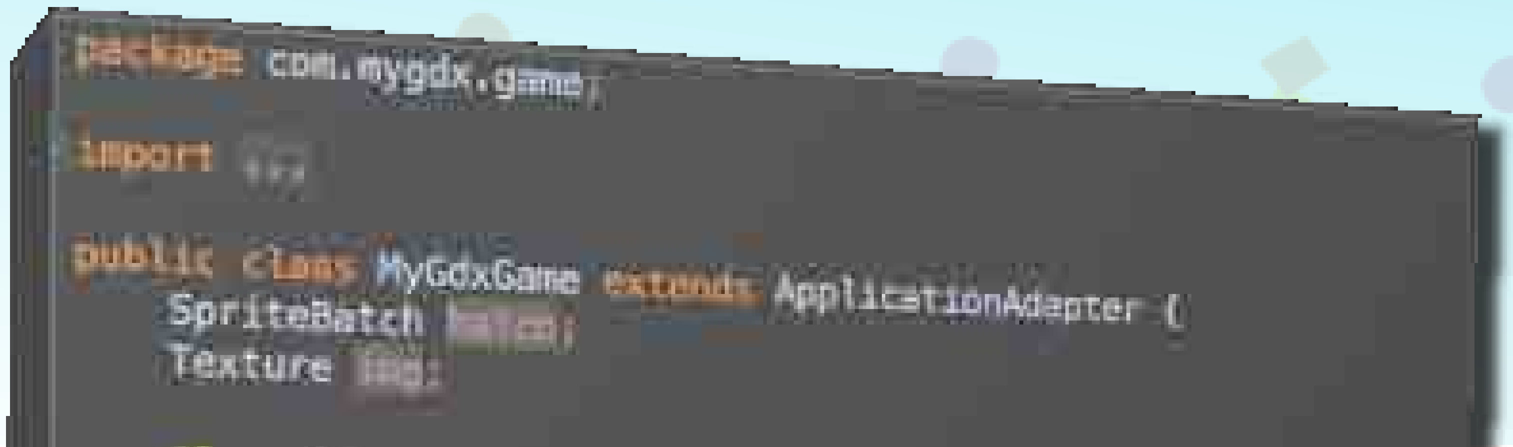


```java
import ...

public class AndroidLauncher extends AndroidApplication {
    @Override
    protected void onCreate (Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        AndroidApplicationConfiguration config = new AndroidApplicationConfiguration();
        initialize(new MyGdxGame(), config);
    }
}
```

Essential Classes

Game classes can extend **ApplicationAdapter**, which is the main lifecycle for LibGdx



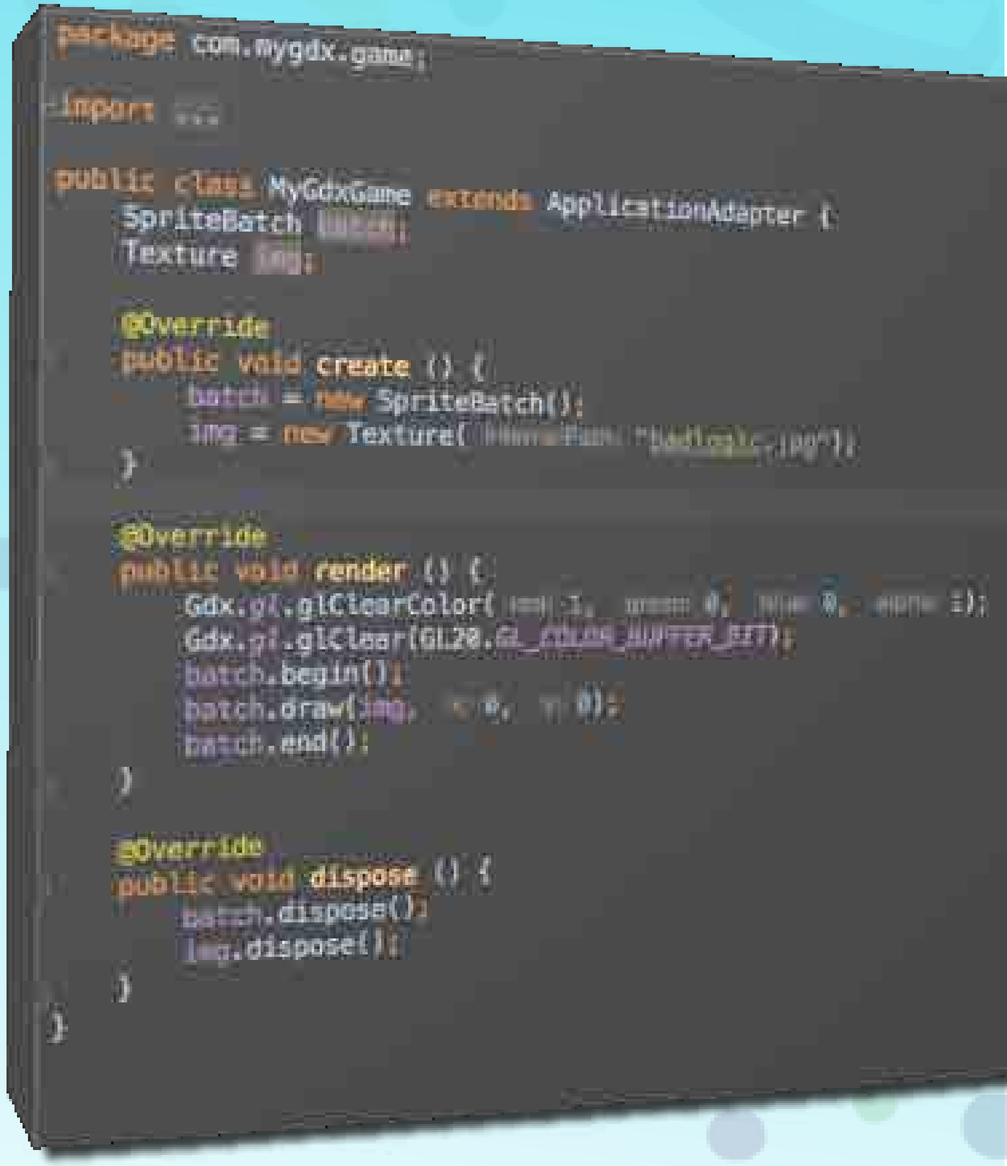
```
package com.mygdx.game;

import ...;

public class MyGdxGame extends ApplicationAdapter {
    SpriteBatch batch;
    Texture img;
```

LibGDX Game **LifeCycle**

create
resize(int w, int h)
render
pause
resume
dispose



```
package com.mygdx.game;

import ...

public class MyGdxGame extends ApplicationAdapter {
    SpriteBatch batch;
    Texture img;

    @Override
    public void create () {
        batch = new SpriteBatch();
        img = new Texture( internalPath: "badlogic.jpg");
    }

    @Override
    public void render () {
        Gdx.gl.glClearColor( red: 1, green: 0, blue: 0, alpha: 1);
        Gdx.gl.glClear(GL20.GL_COLOR_BUFFER_BIT);
        batch.begin();
        batch.draw(img, x: 0, y: 0);
        batch.end();
    }

    @Override
    public void dispose () {
        batch.dispose();
        img.dispose();
    }
}
```

create

Create is called only **once** when the application **is first initialized**

resize(**int w, int h**)

Resize is called whenever the game screen is **re-sized** and the game is **not** in the paused state

resize(**int w, int h**)

Parameters (int w, int h) are the new **width** and **height** the screen has been **resized** to

render

Method called **every time** rendering is performed

Game logic updates usually are performed here

pause

Pause is called whenever Home button is pressed or incoming call is received

Sprite Batch

The SpriteBatch can be called in the render method to draw Texture objects

Texture

A texture object can used to load an image within the assets folder

pause

This is a good place to save the game state

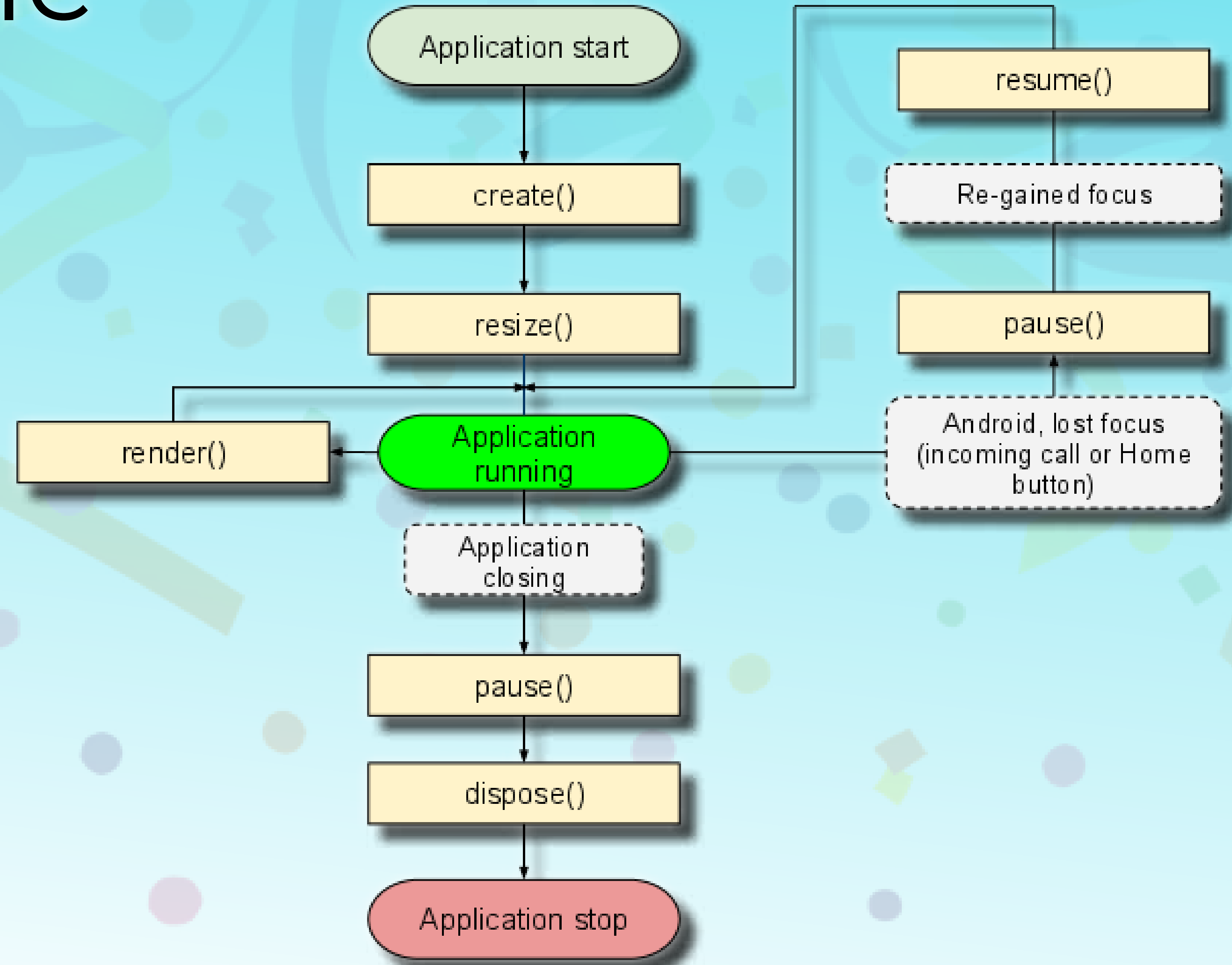
resume

This method is called whenever the application resumes from pause state

dispose

Called whenever application is **destroyed**

Life Cycle


Application start
create()
resize()
Application running
render()
Application closing
pause()
dispose()
Application stop
Android, lost focus (incoming call or Home button)
pause()
Re-gained focus
resume()

Essential Classes

These methods are used for creating the game scene, handling drawing, resizing, disposing, pausing and resuming play

Using **AVD**

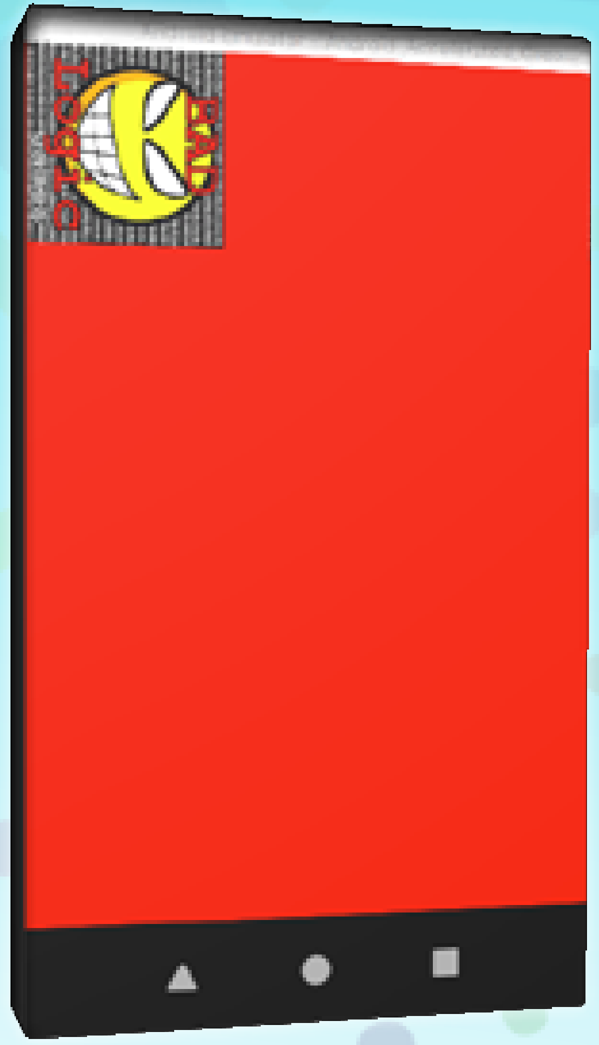
AVD, is also known as **Android Virtual Device,** allows you to create and manage virtual devices to run your application on

Using **AVD**

Add a virtual device to AVD Manager, and click the run button to begin testing

Using AVD

The emulator will open your game once hitting the run button and selecting the virtual device



Physical Device

In order to run on a physical device, you should turn your developer options settings on

Log Cat

You can use **Gdx.app.log()** to print things to console

```
ystem.err(28238): log4j:WARN No appenders could be found for logger (eu.medsea.
ystem.err(28238): log4j:WARN Please initialize the log4j system properly.
ystem.err(28238): log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#no
alvikvm(28238): GC_CONCURRENT freed 258K, 6% free 6428K/6791K, paused 3ms-
urfaceFlinger( 223): id=248 Removed com.android.launcher/com.android.launcher2.L
urfaceFlinger( 223): id=248 Removed com.android.launcher/com.android.launcher2.L
alvikvm(28238): GC_CONCURRENT freed 223K, 6% free 6676K/7047K, paused 3ms-
alvikvm[illegible] paused 2ms-
alvikvm[illegible] paused 26ms
alvikvm(28238): GC_FOR_ALLOC freed 104K, 10% free 6654K/7367K, paused 23ms
alvikvm(28238): GC_FOR_ALLOC freed 64K, 10% free 6677K/7367K, paused 24ms
alvikvm(28238): GC_FOR_ALLOC freed 24K, 10% free 6677K/7367K, paused 23ms
alvikvm(28238): GC_FOR_ALLOC freed 18K, 10% free 6685K/7367K, paused 23ms
alvikvm(28238): GC_FOR_ALLOC freed <1K, 10% free 6701K/7367K, paused 22ms
lvikvm-heap(28238): Grow heap (frag case) to 6.652MB for 7412-byte allocation
alvikvm(28238): GC_FOR_ALLOC freed 1K, 10% free 6708K/7431K, paused 22ms
LINE (28238): new: android.text.TextLine@4080a638
indowManager( 289): addWindowToListInOrder(): current mSkipWidgetFocus=true
alvikvm(28238): GC_FOR_ALLOC freed 163K, 10% free 6745K/743[illegible] paused 27ms
lvikvm-heap(28238): Grow heap (frag case) to 6.750MB for [illegible]
alvikvm(28238): GC_FOR_ALLOC freed 10K, 11% free 679[illegible]
urfaceFlinger( 223): id=253(1) createSurface 0x13be34 fo[illegible]
```

Log Cat

Open the **Log Cat** window to see your output message

```
ystem.err(28238): log4j:WARN No appenders could be found for logger (eu.medsea
ystem.err(28238): log4j:WARN Please initialize the log4j system properly.
ystem.err(28238): log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#no
alvikvm(28238): GC_CONCURRENT freed 256K, 6% free 6428K/6791K, paused 3ms-
urfaceFlinger(  223): id=248 Removed com.android.launcher/com.android.launcher2.
urfaceFlinger(  223): id=248 Removed com.android.launcher/com.android.launcher2.
alvikvm(28238): GC_CONCURRENT freed 223K, 6% free 6676K/7047K, paused 3ms-
alvikvm[illegible] paused 2ms-
alvikvm[illegible] paused 26ms
alvikvm[illegible] GC_FOR_ALLOC freed 104K, 10% free [illegible] paused 23ms
alvikvm(28238): GC_FOR_ALLOC freed 64K, 10% free 6677K/7367K, paused 24ms
alvikvm(28238): GC_FOR_ALLOC freed 24K, 10% free 6677K/7367K, paused 23ms
alvikvm(28238): GC_FOR_ALLOC freed 16K, 10% free 6685K/7367K, paused 23ms
alvikvm(28238): GC_FOR_ALLOC freed <1K, 10% free 6701K/7367K, paused 22ms
alvikvm-heap(28238): Grow heap (frag case) to 6.652MB for 7412-byte allocation
alvikvm(28238): GC_FOR_ALLOC freed 0K, 10% free 6708K/7431K, paused 22ms
LINE  (28238): new: android.text.TextLine@4080a638
indowManager(  289): addWindowToListInOrder() : current mSkipWidgetFocus=true
alvikvm(28238): GC_FOR_ALLOC freed 163K, 10% free 6745K/[illegible] paused 27ms
alvikvm-heap(28238): Grow heap (frag case) to 6.750MB fo[illegible]
alvikvm(28238): GC_FOR_ALLOC freed 10K, 11% free 679[illegible]
urfaceFlinger(  223): id=253(1) createSurface 0x13be34 fo[illegible]
```