# Pendahuluan

Aplikasi untuk game “1..2..3.. Dor!” dibuat dengan menggunakan framework libGDX oleh badlogicgames. Mengembangkan game dengan framework tersebut memudahkan dalam segi pembagian logika pengerjaan game. Prinsip pengerjaannya dibagi dengan beberapa konsep utama yakni, Screen, Stage, dan Actor. Screen digunakan untuk melakukan pemindahan dari satu tampilan ke tampilan lainnya. Stage digunakan untuk menampung Actor dan menangani aksi pada stage. Pada pembuatan game ini, class dengan tujuan yang sama ditempatkan pada folder yang sama. File – file utama yang digunakan yakni:

1. AndroidLauncher.java

File ini memuat class androidlauncher yang meng-extend AndroidApplication. Tujuan dari class ini cukup sederhana, yakni menampilkan launcher(bila ada) dan menjalankan program utama. Di dalam onCreate() class ini menginisialisasi konfigurasi aplikasi dan memanggil program utama(CowboyDor)

1. CowboyDor.java

Memuat class CowboyDor yang meng-extend Game. Di dalam class tersebut dilakukan fungsi create untuk inisialisasi ScreenManager dan menampilkan Screen utama yang diinginkan, yakni Main Menu.

1. Cowboy.java

Memuat class Cowboy. Class cowboy merupakan object yang dapat digunakan sebagai actor dalam Stage.

1. GameMode.java

Memuat class GameMode untuk menentukan mode game yang diinginkan. Masing – masing mode game disertakan dengan settingan environment tersendiri. Seperti jumlah ronde yang diinginkan, tipe game, limit score, dan mencatat ronde.

1. World.java

Memuat class world. Class ini digunakan ketika melakukan inisialisasi terhadap tampilan permainan dimulai. Class ini akan memasukkan masing – masing cowboy ke posisinya.

# Utility

Game ini terdiri dari 3 buah utility sederhana yang digunakan untuk manajemen Screen dan menambahkan komponen dengan mudah. 3 utility tersebut diantaranya :

1. Screen Enum

Screen enum digunakan untuk menginisialisasi dan mempermudah pemanggilan Screen. Tugas dari screen enum cukup sederhana yakni menampung dan menginisialisasi setiap Screen yang ada pada game untuk mempermudah pemanggilan.

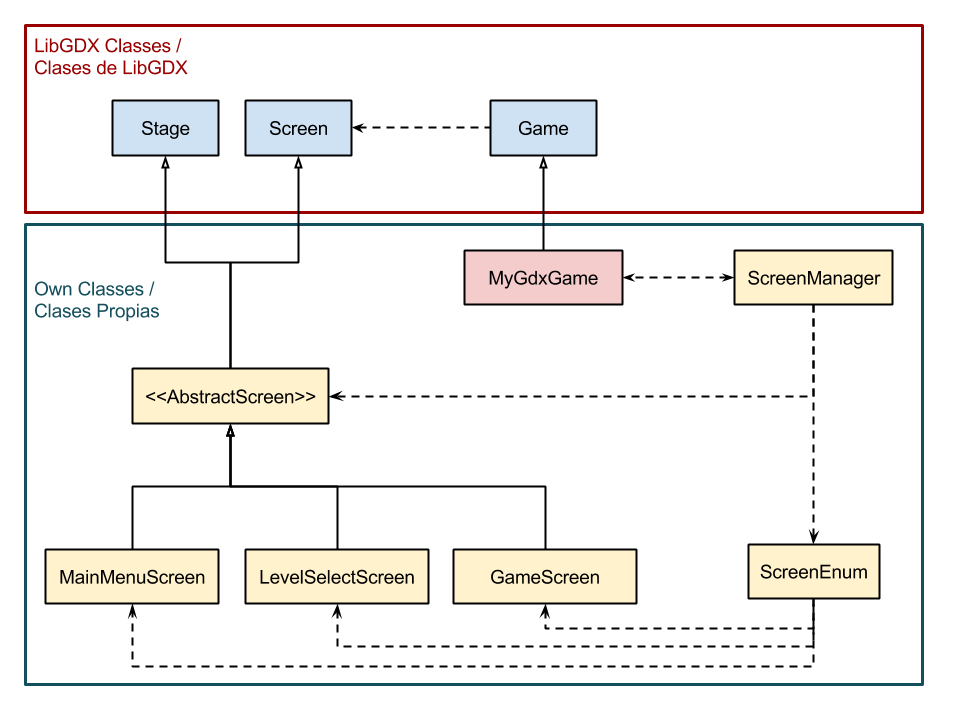
1. Screen Manager

Screen Manager digunakan untuk manajemen Screen seperti menampilkan dan menghapus Screen beserta komponen – komponen yang ada di dalamnya. Fungsi Show Screen di dalam class tersebut akan menampilkan Screen yang dipilih dengan menerima parameter Screen enum dan parameter data lain.

1. UIFactory

Fungsi UIFactory cukup sederhana yakni, dapat digunakan untuk membuat button dan juga dapat digunakan untuk menginisialisasi Listener pada button yang sudah dibuat.

# Screen



Game ini terdiri dari 4 jenis class screen, yakni:

1. AbstractScreen

AbstractScreen digunakan sebagai class screen dasar untuk menampilkan screen lainnya. Screen lainnya akan menampilkan isi dengan ciri – ciri yang disamakan melalui AbstractScreen tanpa harus di definiskan ulang pada masing – masing screen. Screen pada dasarnya dapat juga digunakan untuk menentukan lebar dan panjang dari viewport aplikasi. Penentuan ini berguna untuk menyamakan titik perhitungan viewport pada ukuran layar yang berbeda.

public abstract void buildStage();

fungsi buildStage() harus dipanggil untuk setiap Screen yang meng-extends AbstractScreen. Fungsi buildStage ini digunakan untuk membuat stage dan memasukkan actor ke dalam stage.

1. LevelSelectScreen

Tujuan dari Screen ini cukup sederhana. LevelSelectScreen meng-extends AbstractScreen dan memiliki ciri – ciri yang sama. LevelSelectScreen diwajibkan untuk meng-override buildStage dan kemudian dapat digunakan untuk meletakkan actor di dalamnya. Dalam screen ini, kami meletakkan enam jenis actor diantaranya background, tombol back, dan sisanya adalah tombol untuk mode game. Untuk berpindah halaman, setiap button diberikan sebuah listener dengan memberikan parameter – parameter yang sesuai. Contohnya seperti tombol back yang diberikan sebuah listener untuk dapat kembali ke MainMenuScreen.

btnBack.addListener(UIFactory.*createListener*(ScreenEnum.*MAIN\_MENU*));

1. MainMenuScreen

Hampir mirip dengan LevelSelectScreen, hanya saja MainMenuScreen hanya memuat 2 actor, yakni background dan tombol play. Tombol play diberikan listener yang dapat mengganti screen ke LevelSelectScreen.

1. GameScreen

Game Screen juga memiliki ciri – ciri yang serupa. GameScreen merupakan Screen yang digunakan untuk melakukan permainan berdasarkan mode – mode yang dipilih pada LevelSelectScreen. Terdapat beberapa fungsi yang digunakan di dalam Class Game Screen diantaranya:

1. Constructor GameScreen(GameType GT, boolean debug)

Constructor GameScreen menerima dua jenis parameter yakni yang pertama adalah GameType yang berdasarkan Enum GameType di dalam class GameMode, yang kedua adalah debug yang menampilkan garis dan kolom dimensi object dan actor.

1. Render(float delta)

Fungsi render ini merupakan fungsi turunan dari class Stage. Fungsi render digunakan untuk me-render objek – objek yang ada dalam class, seperti actor, text, gambar dan lainnya. Fungsi ini sebenarnya tidak diperlukan bila tidak menggunakan animasi untuk objek tertentu. Bila tidak menggunakan render(), actor dan settingnya dapat dipanggil melalui fungsi BuildStage().   
Dalam game ini, kami menggunakan fungsi render untuk melakukan Count Down tembakan cowboy dan perhitungan speed, meletakkan background dan sprite, animasi cowboy berdasarkan state, dan refresh mode Game.

@Override  
public void render(float delta) {  
 Gdx.*gl*.glClearColor(0, 1, 0, 0.8f);  
 Gdx.*gl*.glClear(GL20.*GL\_COLOR\_BUFFER\_BIT*);  
  
 spriteBatch.begin();  
 drawBackground(spriteBatch);  
 drawCounter(spriteBatch);  
 drawCowboy();  
 drawGame(spriteBatch);  
 spriteBatch.end();  
  
 this.draw();  
  
 if(debug)  
 drawDebug();  
}

Pemanggilan this.draw pada coding di atas berfungsi untuk me-render actor yang dipanggil pada fungsi buildStage() *(akan dijelaskan pada bagian fungsi buildStage())*.

1. Resize(int w, int h)

Merupakan fungsi turunan dari class Stage. Dalam fungsi ini, dilakukan resize untuk resolusi layar game dan pengambilan titik ukur.

@Override  
public void resize(int w, int h) {  
 this.width = w;  
 this.height = h;  
 ppuX = (float)width / *CAMERA\_WIDTH*;  
 ppuY = (float)height / *CAMERA\_HEIGHT*;  
 Log.*d*("HEIGHT", Float.*toString*(ppuX));  
 Log.*d*("WIDTH", Float.*toString*(ppuY));  
}

ppuX(pixels per unit X) dan ppuY(pixels per unit Y) merupakan titik ukur yang digunakan untuk mempermudah penempatan posisi objek pada layar.

1. loadTextures()

Sesuai namanya, loadTextures digunakan untuk menginisialisasi gambar dan sprite yang ada pada GameScreen. Texture yang di load diantaranya texture untuk state cowboy(IDLE, SHOOT, DIE, WIN) dan background. Texture untuk cowboy dibagi menjadi dua yakni texture untuk cowboy di posisi kiri dan di posisi kanan.

1. loadSound()

Sesuai namanya, fungsi loadSound digunakan untuk menginisialisasi sound untuk GameScreen. Sound yang digunakan diantaranya hitSound(ketika cowboy mengenai sasaran), missSound(ketika cowboy tidak kena sasaran), reloadSound(suara reload), winSound(ketika cowboy menang).

1. loadGameMode()

Fungsi loadGameMode digunakan untuk refresh mode game dan menginisialisasi settingan – settingan yang diberikan pada cowboy dan inisialisasi flag sesuai dengan tipe game.

1. getAnimation(CowboyState state, String side)

Fungsi ini digunakan untuk mengembalikan animasi cowboy yang sesuai dengan state dan posisi dari si cowboy(kiri atau kanan). Animasi yang dikembalikan kemudian dapat di jalankan pada fungsi drawCowboy().

1. drawCowboy()

Fungsi ini digunakan untuk menjalankan animasi. Di dalam fungsi ini diambil timer yang disediakan oleh libGDX untuk mengambil frame animasi yang sesuai. Fungsi ini juga digunakan untuk mengatur pengambilan animasi yang sesuai dengan state dari masing – masing cowboy.

1. drawGame()