目錄

# 全域變數定義Global Variable

## 通用定義

定義Pixel per meter作用於使每Pixel平均分配，通常不做更動。

public static final int PPM = 100;

介面定義與繪畫範圍，分別有邊界與繪畫比例(放大係數)。

public static String GAME\_TITLE = "New Game";

public static int SCALE = 3;

public static int WIDTH = 320;

public static int HEIGHT = 200;

## 碰撞體定義

編輯各種物件的碰撞定義或是事件碰撞等。

public static final short NOTHING\_BIT = 0;

public static final short GROUND\_BIT = 1;

public static final short PLAYER\_BIT = 2;

## 控制器設定

[預設] 經典鍵盤: 右手方向鍵控制

街機鍵盤: 左手方向鍵控制

控制器 (Xbox360)

[19/1/11] TODO:

檢查預設是否存在，後續需要編輯使用者輸入，現行使用固定定義可能需要配合使用Java的元件去取輸入。

按鈕定義

public static int buttonUp = Keys.UP;

public static int buttonLeft = Keys.LEFT;

public static int buttonRight = Keys.RIGHT;

public static int buttonDown = Keys.DOWN;

public static int button1; // A

public static int button2; // B

public static int button3; // X

public static int button4; // Y

public static int button5; // L Button

public static int button6; // R Button

public static int button7; // Select Button

public static int button8; // Start Button

# 場景介定

場景的整體架構大略為(待補圖)：

Main ->

StageManager()

setScreen(Screen) : void

VarableManager()

每個場景設定控制器(Stage Manager)

[19/01/11]可能可以換成共用類別，待考量後決定更換與否。

PlayableStage : Screen ->

InputHandler()

StageVarableManager ()

顯示動畫後並轉場至介面選單(MenuScreen)

TitleScreen

MenuScreen

## 場景類別

PlayableStage

定義基礎元件，有關攝影機視角、玩家與物件生成、碰撞體等定義

地圖繪製上使用TileMap協助繪製，需要嵌入LibGDX內建的類別使用。

### 變數定義

基礎物件相關

public Main game;

public World world;

public Box2DDebugRenderer b2dr;

public OrthographicCamera camera;

public Viewport cameraViewPort;

地圖相關

private TmxMapLoader mapLoader;

private TiledMap map;

private OrthogonalTiledMapRenderer mapRenderer;

物件相關

public Player player;

public BodyDef bdef;

public Body b2body;

## 控制器

[19/1/15]

待確定是否要放置在場景內部設定，或是使用其他類別交由場景來執行？

移動速度數據交由 玩家類別中設定，在行取值

方向控制與不按按鍵時取消柔滑移動

if (Gdx.input.isKeyPressed(Keys.RIGHT) && player.b2Body.getLinearVelocity().x <= 1.25) {

player.b2Body.applyLinearImpulse(new Vector2(1.25f, 0), player.b2Body.getWorldCenter(), true);

}

else if (Gdx.input.isKeyPressed(Keys.LEFT) && player.b2Body.getLinearVelocity().x >= -1.25) {

player.b2Body.applyLinearImpulse(new Vector2(-1.25f, 0), player.b2Body.getWorldCenter(), true);

}

else if (!Gdx.input.isKeyPressed(Keys.RIGHT) && !Gdx.input.isKeyPressed(Keys.LEFT) ){

player.b2Body.setLinearVelocity(0, player.b2Body.getLinearVelocity().y);

}

# 地圖

地圖塊(TileSet)要再重新排列

地圖繪製順序依序：

圖層：

BackgroundColor

景色底色

CloudBack (offset Movable)

黑雲層, other?

BackgroundBack

山後景圖塊, 顏色圖塊, 草色圖塊, 天空景色塊

CloudFront (offset Movable)

白雲層, other?

BackgroundFront

山前景圖塊, 地板塊, 後雲層色圖塊, 草邊角背景

Frontground

空中地塊, 裝飾物件, 前雲層色圖塊

物件區域：

Wall

GameObject

Platform

讀取問題地圖若無法取到時，有資源檔案上的問題，然而使用setToOrtho可以解決，參考以下：

camera.setToOrtho(false, cameraViewPort.getWorldWidth() / 2, cameraViewPort.getWorldHeight() / 2);

舊有的方法：

camera.position.set(cameraViewPort.getScreenWidth() / 2, cameraViewPort.getWorldHeight() / 2, 0);

另行注意，若是地圖的攝影機會出現破圖黑線參考以下計算方法：

(float) Math.round(player.b2body.getPosition().x \* 100f) / 100f;

# 動畫

圖片匯入需要使用到工具

匯入圖片後須注意使用上須從

Atlas > TextureRegion > Texture 等順序來分，圖片讀取時都是以Texture做使用

動畫部分需要依賴到 Animation<TextureRegion>

範例

playerAtlas = new TextureAtlas("img/player/Player.atlas");

idel = new TextureRegion(playerAtlas.findRegion("idel").getTexture(), 2, 2, 40, 50);

設定圖片大小

setBounds(x, y, width, height) 需計算PPM

設定圖片(張)

setRegion(idel);

需要將圖片與Sprite跟著物件走，需另行定義每次更新的事件

public void update(float dt) {

setPosition((b2Body.getPosition().x - getWidth() / 2)+0.05f, (b2Body.getPosition().y - getHeight() / 2)+0.15f);

}

且在主畫面時需要將物件加入Render事件中

game.batch.setProjectionMatrix(camera.combined);

game.batch.begin();

player.draw(game.batch);

game.batch.end();