Лістинг А.1 Layout-код файлу ButtonSoundManager

public class ButtonSoundManager {

private static MediaPlayer mediaPlayer;

public static void playButtonClickSound(Context context) {

if (mediaPlayer == null) {

mediaPlayer = MediaPlayer.create(context, R.raw.click\_sound);

}

mediaPlayer.start();

}

public static void release() {

if (mediaPlayer != null) {

mediaPlayer.release();

mediaPlayer = null;

}

}

}

Лістинг А.2 Layout-код файлу LevelSelectionActivity

public class LevelSelectionActivity extends AppCompatActivity {

private ImageButton soundToggleButton;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_level\_selection);

SystemUIHelper.hideSystemUI(this);

ImageButton level1Button = findViewById(R.id.level1Button);

startService(new Intent(this, MusicService.class));

level1Button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(LevelSelectionActivity.this, MazeActivity1.class);

intent.putExtra("level", 1);

startActivity(intent);

}

});

ImageButton level2Button = findViewById(R.id.level2Button);

level2Button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(LevelSelectionActivity.this, MazeActivity2.class);

intent.putExtra("level", 2);

startActivity(intent);

}

});

ImageButton level3Button = findViewById(R.id.level3Button);

level3Button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(LevelSelectionActivity.this, MazeActivity3.class);

intent.putExtra("level", 3);

startActivity(intent);

}

});

ImageButton level4Button = findViewById(R.id.level4Button);

level4Button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(LevelSelectionActivity.this, MazeActivity4.class);

intent.putExtra("level", 3);

startActivity(intent);

}

});

ImageButton level5Button = findViewById(R.id.level5Button);

level5Button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(LevelSelectionActivity.this, MazeActivity5.class);

intent.putExtra("level", 3);

startActivity(intent);

}

});

ImageButton level6Button = findViewById(R.id.level6Button);

level6Button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(LevelSelectionActivity.this, MazeActivity6.class);

intent.putExtra("level", 3);

startActivity(intent);

}

});

soundToggleButton = findViewById(R.id.soundToggleButton);

// Оновлюємо іконку при старті

updateSoundButtonIcon();

soundToggleButton.setOnClickListener(v -> {

boolean isMusicEnabled = MusicService.isMusicEnabled();

MusicService.setMusicEnabled(this, !isMusicEnabled);

updateSoundButtonIcon();

});

}

private void updateSoundButtonIcon() {

if (MusicService.isMusicEnabled()) {

soundToggleButton.setImageResource(R.drawable.sound\_on); // твоя іконка увімкненого звуку

} else {

soundToggleButton.setImageResource(R.drawable.sound\_of); // твоя іконка вимкненого звуку

}

}

@Override

protected void onResume() {

super.onResume();

MusicService.setVolume(0.5f);

MusicService.resumeMusic();

}

@Override

protected void onPause() {

super.onPause();

MusicService.pauseMusic();

}

}

Лістинг А.3 Layout-код файлу MainActivity

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

SystemUIHelper.hideSystemUI(this);

ImageButton btnStart = findViewById(R.id.btnStart);

startService(new Intent(this, MusicService.class));

btnStart.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

ButtonSoundManager.playButtonClickSound(MainActivity.this);

Intent intent = new Intent(MainActivity.this, LevelSelectionActivity.class);

startActivity(intent);

}

});

}

@Override

protected void onResume() {

super.onResume();

MusicService.setVolume(0.5f);

MusicService.resumeMusic();

}

@Override

protected void onPause() {

super.onPause();

MusicService.pauseMusic();

}

}

Лістинг А.4 Layout-код файлу MazeActivity1

public class MazeActivity1 extends AppCompatActivity {

private AlertDialog settingsDialog;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_maze\_level\_1);

startService(new Intent(this, MusicService.class));

SystemUIHelper.hideSystemUI(this);

ImageButton btnSettings = findViewById(R.id.btnSettings);

btnSettings.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

showSettingsDialog();

});

}

@Override

protected void onResume() {

super.onResume();

MusicService.setVolume(0.2f);

MusicService.resumeMusic();

}

@Override

protected void onPause() {

super.onPause();

MusicService.pauseMusic();

}

private void showSettingsDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(this, R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(this, R.font.pixel);

TextView titleView = new TextView(this);

titleView.setText("НАЛАШТУВАННЯ");

titleView.setTextSize(30);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.parseColor("#e6f7ff"));

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

LinearLayout mainContainer = new LinearLayout(this);

mainContainer.setOrientation(LinearLayout.VERTICAL);

mainContainer.setGravity(Gravity.CENTER\_HORIZONTAL);

mainContainer.setPadding(40, 40, 40, 40);

mainContainer.addView(titleView);

float scale = getResources().getDisplayMetrics().density;

int buttonHeightDp = 55;

int marginPx = (int) (1 \* scale);

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.MATCH\_PARENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(0, marginPx, 0, marginPx);

ImageButton menuButton = new ImageButton(this);

menuButton.setImageResource(R.drawable.menu\_menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

Intent intent = new Intent(this, LevelSelectionActivity.class);

intent.addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);

startActivity(intent);

});

ImageButton restartButton = new ImageButton(this);

restartButton.setImageResource(R.drawable.menu\_restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

recreate();

});

ImageButton musicToggleButton = new ImageButton(this);

updateMusicButtonImage(musicToggleButton);

musicToggleButton.setBackgroundColor(Color.TRANSPARENT);

musicToggleButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

musicToggleButton.setLayoutParams(buttonParams);

musicToggleButton.setOnClickListener(v -> {

boolean current = MusicService.isMusicEnabled();

MusicService.setMusicEnabled(this, !current);

updateMusicButtonImage(musicToggleButton);

});

mainContainer.addView(menuButton);

mainContainer.addView(restartButton);

mainContainer.addView(musicToggleButton);

builder.setView(mainContainer);

builder.setCancelable(true);

settingsDialog = builder.create();

settingsDialog.getWindow().setBackgroundDrawable(new ColorDrawable(Color.BLACK));

settingsDialog.show();

if (settingsDialog.getWindow() != null) {

settingsDialog.getWindow().setLayout(

(int) (getResources().getDisplayMetrics().widthPixels \* 0.9),

LinearLayout.LayoutParams.WRAP\_CONTENT

);

settingsDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private void updateMusicButtonImage(ImageButton button) {

if (MusicService.isMusicEnabled()) {

button.setImageResource(R.drawable.off\_sound);

} else {

button.setImageResource(R.drawable.on\_sound);

}

}

}

Лістинг А.5 Layout-код файлу MazeActivity2

public class MazeActivity2 extends AppCompatActivity {

private AlertDialog settingsDialog;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_maze\_level\_2);

startService(new Intent(this, MusicService.class));

SystemUIHelper.hideSystemUI(this);

ImageButton btnSettings = findViewById(R.id.btnSettings);

btnSettings.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

showSettingsDialog();

});

}

@Override

protected void onResume() {

super.onResume();

MusicService.setVolume(0.2f);

MusicService.resumeMusic();

}

@Override

protected void onPause() {

super.onPause();

MusicService.pauseMusic();

}

private void showSettingsDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(this, R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(this, R.font.pixel);

TextView titleView = new TextView(this);

titleView.setText("НАЛАШТУВАННЯ");

titleView.setTextSize(30);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.parseColor("#e6f7ff"));

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

LinearLayout mainContainer = new LinearLayout(this);

mainContainer.setOrientation(LinearLayout.VERTICAL);

mainContainer.setGravity(Gravity.CENTER\_HORIZONTAL);

mainContainer.setPadding(40, 40, 40, 40);

mainContainer.addView(titleView);

float scale = getResources().getDisplayMetrics().density;

int buttonHeightDp = 55;

int marginPx = (int) (1 \* scale);

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.MATCH\_PARENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(0, marginPx, 0, marginPx);

ImageButton menuButton = new ImageButton(this);

menuButton.setImageResource(R.drawable.menu\_menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

Intent intent = new Intent(this, LevelSelectionActivity.class);

intent.addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);

startActivity(intent);

});

ImageButton restartButton = new ImageButton(this);

restartButton.setImageResource(R.drawable.menu\_restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

recreate();

});

ImageButton musicToggleButton = new ImageButton(this);

updateMusicButtonImage(musicToggleButton);

musicToggleButton.setBackgroundColor(Color.TRANSPARENT);

musicToggleButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

musicToggleButton.setLayoutParams(buttonParams);

musicToggleButton.setOnClickListener(v -> {

boolean current = MusicService.isMusicEnabled();

MusicService.setMusicEnabled(this, !current);

updateMusicButtonImage(musicToggleButton);

});

mainContainer.addView(menuButton);

mainContainer.addView(restartButton);

mainContainer.addView(musicToggleButton);

builder.setView(mainContainer);

builder.setCancelable(true);

settingsDialog = builder.create();

settingsDialog.getWindow().setBackgroundDrawable(new ColorDrawable(Color.BLACK));

settingsDialog.show();

if (settingsDialog.getWindow() != null) {

settingsDialog.getWindow().setLayout(

(int) (getResources().getDisplayMetrics().widthPixels \* 0.9),

LinearLayout.LayoutParams.WRAP\_CONTENT

);

settingsDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private ImageButton createImageButton(int drawableResId, LinearLayout.LayoutParams params) {

ImageButton button = new ImageButton(this);

button.setImageResource(drawableResId);

button.setBackgroundColor(Color.TRANSPARENT);

button.setScaleType(ImageView.ScaleType.FIT\_CENTER);

button.setAdjustViewBounds(true);

button.setLayoutParams(params);

return button;

}

private void updateMusicButtonImage(ImageButton button) {

if (MusicService.isMusicEnabled()) {

button.setImageResource(R.drawable.on\_sound);

} else {

button.setImageResource(R.drawable.off\_sound);

}

Лістинг А.6 Layout-код файлу MazeActivity3

public class MazeActivity3 extends AppCompatActivity {

private AlertDialog settingsDialog;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_maze\_level\_3);

startService(new Intent(this, MusicService.class));

SystemUIHelper.hideSystemUI(this);

ImageButton btnSettings = findViewById(R.id.btnSettings);

btnSettings.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

showSettingsDialog();

});

}

@Override

protected void onResume() {

super.onResume();

MusicService.setVolume(0.2f);

MusicService.resumeMusic();

}

@Override

protected void onPause() {

super.onPause();

MusicService.pauseMusic();

}

private void showSettingsDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(this, R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(this, R.font.pixel);

TextView titleView = new TextView(this);

titleView.setText("НАЛАШТУВАННЯ");

titleView.setTextSize(30);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.parseColor("#e6f7ff"));

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

LinearLayout mainContainer = new LinearLayout(this);

mainContainer.setOrientation(LinearLayout.VERTICAL);

mainContainer.setGravity(Gravity.CENTER\_HORIZONTAL); // тільки по горизонталі

mainContainer.setPadding(40, 40, 40, 40);

mainContainer.addView(titleView);

float scale = getResources().getDisplayMetrics().density;

int buttonHeightDp = 55; // Висота кнопки (більше)

int marginPx = (int) (1 \* scale); // Відступи 5 пікселів

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.MATCH\_PARENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(0, marginPx, 0, marginPx);

ImageButton menuButton = new ImageButton(this);

menuButton.setImageResource(R.drawable.menu\_menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

Intent intent = new Intent(this, LevelSelectionActivity.class);

intent.addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);

startActivity(intent);

});

ImageButton restartButton = new ImageButton(this);

restartButton.setImageResource(R.drawable.menu\_restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

recreate();

});

ImageButton musicToggleButton = new ImageButton(this);

updateMusicButtonImage(musicToggleButton);

musicToggleButton.setBackgroundColor(Color.TRANSPARENT);

musicToggleButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

musicToggleButton.setLayoutParams(buttonParams);

musicToggleButton.setOnClickListener(v -> {

boolean current = MusicService.isMusicEnabled();

MusicService.setMusicEnabled(this, !current);

updateMusicButtonImage(musicToggleButton);

});

mainContainer.addView(menuButton);

mainContainer.addView(restartButton);

mainContainer.addView(musicToggleButton);

builder.setView(mainContainer);

builder.setCancelable(true);

settingsDialog = builder.create();

settingsDialog.getWindow().setBackgroundDrawable(new ColorDrawable(Color.BLACK));

settingsDialog.show();

if (settingsDialog.getWindow() != null) {

settingsDialog.getWindow().setLayout(

(int) (getResources().getDisplayMetrics().widthPixels \* 0.9),

LinearLayout.LayoutParams.WRAP\_CONTENT

);

settingsDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private ImageButton createImageButton(int drawableResId, LinearLayout.LayoutParams params) {

ImageButton button = new ImageButton(this);

button.setImageResource(drawableResId);

button.setBackgroundColor(Color.TRANSPARENT);

button.setScaleType(ImageView.ScaleType.FIT\_CENTER);

button.setAdjustViewBounds(true);

button.setLayoutParams(params);

return button;

}

private void updateMusicButtonImage(ImageButton button) {

if (MusicService.isMusicEnabled()) {

button.setImageResource(R.drawable.on\_sound);

} else {

button.setImageResource(R.drawable.off\_sound);

}

}

}

Лістинг А.7 Layout-код файлу Level\_Selection

public class MazeActivity4 extends AppCompatActivity {

private AlertDialog settingsDialog;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_maze\_level\_4);

startService(new Intent(this, MusicService.class));

SystemUIHelper.hideSystemUI(this);

ImageButton btnSettings = findViewById(R.id.btnSettings);

btnSettings.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

showSettingsDialog();

});

}

@Override

protected void onResume() {

super.onResume();

MusicService.setVolume(0.2f);

MusicService.resumeMusic();

}

@Override

protected void onPause() {

super.onPause();

MusicService.pauseMusic();

}

private void showSettingsDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(this, R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(this, R.font.pixel);

TextView titleView = new TextView(this);

titleView.setText("НАЛАШТУВАННЯ");

titleView.setTextSize(30);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.parseColor("#e6f7ff"));

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

LinearLayout mainContainer = new LinearLayout(this);

mainContainer.setOrientation(LinearLayout.VERTICAL);

mainContainer.setGravity(Gravity.CENTER\_HORIZONTAL);

mainContainer.setPadding(40, 40, 40, 40);

mainContainer.addView(titleView);

float scale = getResources().getDisplayMetrics().density;

int buttonHeightDp = 55;

int marginPx = (int) (1 \* scale);

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.MATCH\_PARENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(0, marginPx, 0, marginPx);

ImageButton menuButton = new ImageButton(this);

menuButton.setImageResource(R.drawable.menu\_menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

Intent intent = new Intent(this, LevelSelectionActivity.class);

intent.addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);

startActivity(intent);

});

ImageButton restartButton = new ImageButton(this);

restartButton.setImageResource(R.drawable.menu\_restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

recreate();

});

ImageButton musicToggleButton = new ImageButton(this);

updateMusicButtonImage(musicToggleButton);

musicToggleButton.setBackgroundColor(Color.TRANSPARENT);

musicToggleButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

musicToggleButton.setLayoutParams(buttonParams);

musicToggleButton.setOnClickListener(v -> {

boolean current = MusicService.isMusicEnabled();

MusicService.setMusicEnabled(this, !current);

updateMusicButtonImage(musicToggleButton);

});

// Додаємо кнопки одна під одною

mainContainer.addView(menuButton);

mainContainer.addView(restartButton);

mainContainer.addView(musicToggleButton);

builder.setView(mainContainer);

builder.setCancelable(true);

settingsDialog = builder.create();

settingsDialog.getWindow().setBackgroundDrawable(new ColorDrawable(Color.BLACK));

settingsDialog.show();

if (settingsDialog.getWindow() != null) {

settingsDialog.getWindow().setLayout(

(int) (getResources().getDisplayMetrics().widthPixels \* 0.9),

LinearLayout.LayoutParams.WRAP\_CONTENT

);

settingsDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private ImageButton createImageButton(int drawableResId, LinearLayout.LayoutParams params) {

ImageButton button = new ImageButton(this);

button.setImageResource(drawableResId);

button.setBackgroundColor(Color.TRANSPARENT);

button.setScaleType(ImageView.ScaleType.FIT\_CENTER); // змінено

button.setAdjustViewBounds(true);

button.setLayoutParams(params);

return button;

}

private void updateMusicButtonImage(ImageButton button) {

if (MusicService.isMusicEnabled()) {

button.setImageResource(R.drawable.on\_sound);

} else {

button.setImageResource(R.drawable.off\_sound);

}

}

Лістинг А.8 Layout-код файлу MazeActivity5

public class MazeActivity5 extends AppCompatActivity {

private AlertDialog settingsDialog;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_maze\_level\_5);

startService(new Intent(this, MusicService.class));

SystemUIHelper.hideSystemUI(this);

ImageButton btnSettings = findViewById(R.id.btnSettings);

btnSettings.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

showSettingsDialog();

});

}

@Override

protected void onResume() {

super.onResume();

MusicService.setVolume(0.2f);

MusicService.resumeMusic();

}

@Override

protected void onPause() {

super.onPause();

MusicService.pauseMusic();

}

private void showSettingsDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(this, R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(this, R.font.pixel);

TextView titleView = new TextView(this);

titleView.setText("НАЛАШТУВАННЯ");

titleView.setTextSize(30);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.parseColor("#e6f7ff"));

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

LinearLayout mainContainer = new LinearLayout(this);

mainContainer.setOrientation(LinearLayout.VERTICAL);

mainContainer.setGravity(Gravity.CENTER\_HORIZONTAL);

mainContainer.setPadding(40, 40, 40, 40);

mainContainer.addView(titleView);

float scale = getResources().getDisplayMetrics().density;

int buttonHeightDp = 55;

int marginPx = (int) (1 \* scale);

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.MATCH\_PARENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(0, marginPx, 0, marginPx);

ImageButton menuButton = new ImageButton(this);

menuButton.setImageResource(R.drawable.menu\_menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

Intent intent = new Intent(this, LevelSelectionActivity.class);

intent.addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);

startActivity(intent);

});

ImageButton restartButton = new ImageButton(this);

restartButton.setImageResource(R.drawable.menu\_restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

recreate();

});

ImageButton musicToggleButton = new ImageButton(this);

updateMusicButtonImage(musicToggleButton);

musicToggleButton.setBackgroundColor(Color.TRANSPARENT);

musicToggleButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

musicToggleButton.setLayoutParams(buttonParams);

musicToggleButton.setOnClickListener(v -> {

boolean current = MusicService.isMusicEnabled();

MusicService.setMusicEnabled(this, !current);

updateMusicButtonImage(musicToggleButton);

});

// Додаємо кнопки одна під одною

mainContainer.addView(menuButton);

mainContainer.addView(restartButton);

mainContainer.addView(musicToggleButton);

builder.setView(mainContainer);

builder.setCancelable(true);

settingsDialog = builder.create();

settingsDialog.getWindow().setBackgroundDrawable(new ColorDrawable(Color.BLACK));

settingsDialog.show();

if (settingsDialog.getWindow() != null) {

settingsDialog.getWindow().setLayout(

(int) (getResources().getDisplayMetrics().widthPixels \* 0.9),

LinearLayout.LayoutParams.WRAP\_CONTENT

);

settingsDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private ImageButton createImageButton(int drawableResId, LinearLayout.LayoutParams params) {

ImageButton button = new ImageButton(this);

button.setImageResource(drawableResId);

button.setBackgroundColor(Color.TRANSPARENT);

button.setScaleType(ImageView.ScaleType.FIT\_CENTER);

button.setAdjustViewBounds(true);

button.setLayoutParams(params);

return button;

}

private void updateMusicButtonImage(ImageButton button) {

if (MusicService.isMusicEnabled()) {

button.setImageResource(R.drawable.on\_sound);

} else {

button.setImageResource(R.drawable.off\_sound);

}

}

}

Лістинг А.9 Layout-код файлу MazeActivity6

public class MazeActivity6 extends AppCompatActivity {

private AlertDialog settingsDialog;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_maze\_level\_6);

startService(new Intent(this, MusicService.class));

SystemUIHelper.hideSystemUI(this);

ImageButton btnSettings = findViewById(R.id.btnSettings);

btnSettings.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

showSettingsDialog();

});

}

@Override

protected void onResume() {

super.onResume();

MusicService.setVolume(0.2f);

MusicService.resumeMusic();

}

@Override

protected void onPause() {

super.onPause();

MusicService.pauseMusic();

}

private void showSettingsDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(this, R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(this, R.font.pixel);

TextView titleView = new TextView(this);

titleView.setText("НАЛАШТУВАННЯ");

titleView.setTextSize(30);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.parseColor("#e6f7ff"));

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

LinearLayout mainContainer = new LinearLayout(this);

mainContainer.setOrientation(LinearLayout.VERTICAL);

mainContainer.setGravity(Gravity.CENTER\_HORIZONTAL);

mainContainer.setPadding(40, 40, 40, 40);

mainContainer.addView(titleView);

float scale = getResources().getDisplayMetrics().density;

int buttonHeightDp = 55;

int marginPx = (int) (1 \* scale);

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.MATCH\_PARENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(0, marginPx, 0, marginPx);

ImageButton menuButton = new ImageButton(this);

menuButton.setImageResource(R.drawable.menu\_menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

Intent intent = new Intent(this, LevelSelectionActivity.class);

intent.addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);

startActivity(intent);

});

ImageButton restartButton = new ImageButton(this);

restartButton.setImageResource(R.drawable.menu\_restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(this);

settingsDialog.dismiss();

recreate();

});

ImageButton musicToggleButton = new ImageButton(this);

updateMusicButtonImage(musicToggleButton);

musicToggleButton.setBackgroundColor(Color.TRANSPARENT);

musicToggleButton.setScaleType(ImageView.ScaleType.CENTER\_INSIDE);

musicToggleButton.setLayoutParams(buttonParams);

musicToggleButton.setOnClickListener(v -> {

boolean current = MusicService.isMusicEnabled();

MusicService.setMusicEnabled(this, !current);

updateMusicButtonImage(musicToggleButton);

});

mainContainer.addView(menuButton);

mainContainer.addView(restartButton);

mainContainer.addView(musicToggleButton);

builder.setView(mainContainer);

builder.setCancelable(true);

settingsDialog = builder.create();

settingsDialog.getWindow().setBackgroundDrawable(new ColorDrawable(Color.BLACK));

settingsDialog.show();

if (settingsDialog.getWindow() != null) {

settingsDialog.getWindow().setLayout(

(int) (getResources().getDisplayMetrics().widthPixels \* 0.9),

LinearLayout.LayoutParams.WRAP\_CONTENT

);

settingsDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private ImageButton createImageButton(int drawableResId, LinearLayout.LayoutParams params) {

ImageButton button = new ImageButton(this);

button.setImageResource(drawableResId);

button.setBackgroundColor(Color.TRANSPARENT);

button.setScaleType(ImageView.ScaleType.FIT\_CENTER);

button.setAdjustViewBounds(true);

button.setLayoutParams(params);

return button;

}

private void updateMusicButtonImage(ImageButton button) {

if (MusicService.isMusicEnabled()) {

button.setImageResource(R.drawable.on\_sound);

} else {

button.setImageResource(R.drawable.off\_sound);

}

}

}

Лістинг А.10 Layout-код файлу MazeView1

public class MazeView1 extends View {

private final Paint exitPaint;

private final MediaPlayer coinSound;

private AlertDialog levelCompleteDialog;

private final int[][] maze = {

{1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},

{1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1},

{1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1},

{1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1},

{1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1},

{1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1},

{1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1},

{1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1},

{1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1}

};

private int cellSize;

private int playerRow = 21, playerCol = 5;

private final int exitRow = 0, exitCol = 5;

private float startX, startY;

private final Handler handler = new Handler();

private final List<int[]> coins;

private int collectedCoins = 0;

private Bitmap playerBitmap;

private Bitmap playerBlinkBitmap;

private boolean isBlinking = false;

private final int BLINK\_TIME = 500;

private Handler blinkHandler = new Handler();

private Bitmap[] coinBitmaps;

private Bitmap wallBitmap;

private int coinFrame = 0;

private final int ANIMATION\_SPEED = 200;

public MazeView1(Context context, AttributeSet attrs) {

super(context, attrs);

exitPaint = new Paint();

exitPaint.setColor(0xFF00FF00);

playerBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.player);

playerBlinkBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.palyer1);

coinBitmaps = new Bitmap[6];

coinBitmaps[0] = BitmapFactory.decodeResource(getResources(), R.drawable.coin0);

coinBitmaps[1] = BitmapFactory.decodeResource(getResources(), R.drawable.coin1);

coinBitmaps[2] = BitmapFactory.decodeResource(getResources(), R.drawable.coin2);

coinBitmaps[3] = BitmapFactory.decodeResource(getResources(), R.drawable.coin4);

coinBitmaps[4] = BitmapFactory.decodeResource(getResources(), R.drawable.coin5);

coinBitmaps[5] = BitmapFactory.decodeResource(getResources(), R.drawable.coin0);

coins = new ArrayList<>();

coins.add(new int[]{12, 7});

coins.add(new int[]{9, 3});

coins.add(new int[]{20, 1});

animateCoin();

coinSound = MediaPlayer.create(getContext(), R.raw.click\_sound);

wallBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.wall\_texture\_1);

showGameRules();

}

private void animateCoin() {

handler.postDelayed(new Runnable() {

@Override

public void run() {

coinFrame++;

if (coinFrame >= coinBitmaps.length) {

coinFrame = 0;

}

invalidate();

animateCoin();

}

}, ANIMATION\_SPEED);

}

@Override

protected void onDraw(Canvas canvas) {

super.onDraw(canvas);

int width = getWidth();

int height = getHeight();

cellSize = Math.min(width / maze[0].length, height / maze.length);

float offsetX = (width - cellSize \* maze[0].length) / 2;

float offsetY = (height - cellSize \* maze.length) / 2;

for (int row = 0; row < maze.length; row++) {

for (int col = 0; col < maze[row].length; col++) {

float left = offsetX + col \* cellSize;

float top = offsetY + row \* cellSize;

float right = left + cellSize;

float bottom = top + cellSize;

if (maze[row][col] == 1) {

canvas.drawBitmap(wallBitmap, null, new RectF(left, top, right, bottom), null);

} else if (maze[row][col] == 0) {

}

}

}

for (int[] coin : coins) {

float cx = offsetX + coin[1] \* cellSize + cellSize / 2;

float cy = offsetY + coin[0] \* cellSize + cellSize / 2;

Bitmap currentCoinBitmap = coinBitmaps[coinFrame];

Bitmap scaledCoinBitmap = Bitmap.createScaledBitmap(currentCoinBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledCoinBitmap, cx - scaledCoinBitmap.getWidth() / 2, cy - scaledCoinBitmap.getHeight() / 2, null);

}

float playerX = offsetX + playerCol \* cellSize + cellSize / 2;

float playerY = offsetY + playerRow \* cellSize + cellSize / 2;

Bitmap currentPlayerBitmap = isBlinking ? playerBlinkBitmap : playerBitmap;

Bitmap scaledPlayerBitmap = Bitmap.createScaledBitmap(currentPlayerBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledPlayerBitmap, playerX - scaledPlayerBitmap.getWidth() / 2, playerY - scaledPlayerBitmap.getHeight() / 2, null);

}

private static final int SWIPE\_THRESHOLD = 50;

@Override

public boolean onTouchEvent(MotionEvent event) {

switch (event.getAction()) {

case MotionEvent.ACTION\_DOWN:

startX = event.getX();

startY = event.getY();

return true;

case MotionEvent.ACTION\_UP:

float endX = event.getX();

float endY = event.getY();

float dx = endX - startX;

float dy = endY - startY;

if (Math.abs(dx) > SWIPE\_THRESHOLD || Math.abs(dy) > SWIPE\_THRESHOLD) {

if (Math.abs(dx) > Math.abs(dy)) {

if (dx > 0) movePlayerSmoothly(1, 0);

else movePlayerSmoothly(-1, 0);

} else {

if (dy > 0) movePlayerSmoothly(0, 1);

else movePlayerSmoothly(0, -1);

}

}

return true;

}

return super.onTouchEvent(event);

}

private void showToastMessage(String message) {

Toast.makeText(getContext(), message, Toast.LENGTH\_SHORT).show();

}

private void movePlayerSmoothly(final int dx, final int dy) {

handler.postDelayed(new Runnable() {

@Override

public void run() {

int nextRow = playerRow + dy;

int nextCol = playerCol + dx;

if (nextRow >= 0 && nextRow < maze.length && nextCol >= 0 && nextCol < maze[0].length && maze[nextRow][nextCol] != 1) {

playerRow = nextRow;

playerCol = nextCol;

collectCoin();

if (playerRow == exitRow && playerCol == exitCol) {

showLevelCompleteDialog();

}

// Перевіряємо, чи гравець потрапив на пастку

if (maze[playerRow][playerCol] == 2) {

resetGame();

return;

}

invalidate();

movePlayerSmoothly(dx, dy);

}

}

}, 5);

}

private void collectCoin() {

for (int i = 0; i < coins.size(); i++) {

if (coins.get(i)[0] == playerRow && coins.get(i)[1] == playerCol) {

coins.remove(i);

collectedCoins++;

if (coinSound != null) {

coinSound.start();

}

isBlinking = true;

invalidate();

blinkHandler.postDelayed(new Runnable() {

@Override

public void run() {

isBlinking = false;

invalidate();

}

}, BLINK\_TIME);

break;

}

}

}

@SuppressLint("SetTextI18n")

private void showLevelCompleteDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(getContext(), R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(getContext(), R.font.pixel);

TextView titleView = new TextView(getContext());

titleView.setText("Рівень пройдено!");

titleView.setTextSize(24);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.WHITE);

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

TextView messageView = new TextView(getContext());

messageView.setText("Кількість зібраних монет: " + collectedCoins + "/3");

messageView.setTextSize(18);

messageView.setTypeface(pixelFont);

messageView.setTextColor(Color.WHITE);

messageView.setGravity(Gravity.CENTER);

messageView.setPadding(20, 10, 20, 30);

LinearLayout containerLayout = new LinearLayout(getContext());

containerLayout.setOrientation(LinearLayout.VERTICAL);

containerLayout.setGravity(Gravity.CENTER);

containerLayout.setPadding(20, 20, 20, 20);

containerLayout.addView(titleView);

containerLayout.addView(messageView);

LinearLayout buttonLayout = new LinearLayout(getContext());

buttonLayout.setOrientation(LinearLayout.HORIZONTAL);

buttonLayout.setGravity(Gravity.CENTER);

buttonLayout.setPadding(20, 20, 20, 20);

int marginInDp = 2;

float scale = getContext().getResources().getDisplayMetrics().density;

int marginInPx = (int) (marginInDp \* scale + 0.5f);

int buttonHeightDp = 50;

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.WRAP\_CONTENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(marginInPx, 0, marginInPx, 0);

ImageButton nextButton = new ImageButton(getContext());

nextButton.setImageResource(R.drawable.next);

nextButton.setBackgroundColor(Color.TRANSPARENT);

nextButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

nextButton.setAdjustViewBounds(true);

nextButton.setLayoutParams(buttonParams);

nextButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

goToNextLevel();

});

ImageButton menuButton = new ImageButton(getContext());

menuButton.setImageResource(R.drawable.menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setAdjustViewBounds(true);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

returnToMenu();

});

ImageButton restartButton = new ImageButton(getContext());

restartButton.setImageResource(R.drawable.restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

restartButton.setAdjustViewBounds(true);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

resetGame();

});

buttonLayout.addView(nextButton);

buttonLayout.addView(menuButton);

buttonLayout.addView(restartButton);

containerLayout.addView(buttonLayout);

builder.setView(containerLayout);

builder.setCancelable(false);

levelCompleteDialog = builder.create();

levelCompleteDialog.show();

if (levelCompleteDialog.getWindow() != null) {

levelCompleteDialog.getWindow().setLayout(

dpToPx(400),

WindowManager.LayoutParams.WRAP\_CONTENT

);

levelCompleteDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private void goToNextLevel() {

Context context = getContext();

Intent intent = new Intent(context, MazeActivity4.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void returnToMenu() {

Context context = getContext();

Intent intent = new Intent(context, LevelSelectionActivity.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void resetGame() {

playerRow = 21;

playerCol = 5;

collectedCoins = 0;

coins.clear();

coins.add(new int[]{12, 7});

coins.add(new int[]{9, 3});

coins.add(new int[]{20, 1});

if (levelCompleteDialog != null && levelCompleteDialog.isShowing()) {

levelCompleteDialog.dismiss();

}

invalidate();

}

private int dpToPx(int dp) {

float density = getContext().getResources().getDisplayMetrics().density;

return Math.round(dp \* density);

}

private void showGameRules() {

AlertDialog.Builder builder = new AlertDialog.Builder(getContext(), R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(getContext(), R.font.pixel);

TextView titleView = new TextView(getContext());

titleView.setText("Правила гри:");

titleView.setTextSize(24);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.WHITE);

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

TextView messageView = new TextView(getContext());

messageView.setText("🕹️ Твоя місія — пройти лабіринт!\n\n\n" +

"Свайпай щоб рухатись та збирай монети на шляху.\n\n\n" +

"Дійди до виходу, щоб перемогти!");

messageView.setTextSize(18);

messageView.setTypeface(pixelFont);

messageView.setTextColor(Color.WHITE);

messageView.setGravity(Gravity.CENTER);

messageView.setPadding(20, 10, 20, 30);

ImageButton closeButton = new ImageButton(getContext());

closeButton.setImageResource(R.drawable.start\_game);

closeButton.setBackgroundColor(Color.TRANSPARENT);

closeButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(

dpToPx(250),

dpToPx(80)

);

params.gravity = Gravity.CENTER;

closeButton.setLayoutParams(params);

closeButton.setPadding(10, 10, 10, 10);

closeButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

resetGame();

});

LinearLayout containerLayout = new LinearLayout(getContext());

containerLayout.setOrientation(LinearLayout.VERTICAL);

containerLayout.setGravity(Gravity.CENTER);

containerLayout.setPadding(20, 20, 20, 20);

containerLayout.addView(titleView);

containerLayout.addView(messageView);

containerLayout.addView(closeButton);

builder.setView(containerLayout);

builder.setCancelable(false);

levelCompleteDialog = builder.create();

levelCompleteDialog.show();

if (levelCompleteDialog.getWindow() != null) {

levelCompleteDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

Лістинг А.11 Layout-код файлу MazeView2

public class MazeView2 extends View {

private final MediaPlayer coinSound;

private AlertDialog levelCompleteDialog;

private final int[][] maze = {

{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1},

{1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},

{1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1},

{1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1},

{1, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1},

{1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1},

{1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1},

{1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1},

{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1},

{1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1},

{1, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1},

{1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 2, 1, 0, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},

{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1}

};

private int playerRow = 37, playerCol = 10;

private final int exitRow = 0, exitCol = 10;

private float startX, startY;

private final Handler handler = new Handler();

private final List<int[]> coins;

private int collectedCoins = 0;

private final Bitmap[] trapBitmaps = new Bitmap[8];

private final Bitmap[] coinBitmaps = new Bitmap[6];

private final Bitmap playerBitmap;

private final Bitmap playerBlinkBitmap;

private boolean isBlinking = false;

private final Handler blinkHandler = new Handler();

private final Bitmap wallBitmap;

//змінна для зберігання індексів кадрів

private int coinFrame = 0;

private int trapFrame = 0;

public MazeView2(Context context, AttributeSet attrs) {

super(context, attrs);

Paint exitPaint = new Paint();

exitPaint.setColor(0xFF00FF00);

playerBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.player);

playerBlinkBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.palyer1);

for (int i = 0; i < coinBitmaps.length; i++) {

int resId = getResources().getIdentifier("coin" + i, "drawable", getContext().getPackageName());

coinBitmaps[i] = BitmapFactory.decodeResource(getResources(), resId);

}

for (int i = 0; i < trapBitmaps.length; i++) {

int resId = getResources().getIdentifier("trap" + i, "drawable", getContext().getPackageName());

trapBitmaps[i] = BitmapFactory.decodeResource(getResources(), resId);

}

coins = new ArrayList<>();

coins.add(new int[]{25, 16});

coins.add(new int[]{11, 16});

coins.add(new int[]{23, 6});

coins.add(new int[]{1, 1});

coins.add(new int[]{33, 1});

animateCoin();

animateTrap();

coinSound = MediaPlayer.create(getContext(), R.raw.click\_sound);

wallBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.wall\_texture\_2);

showGameRules();

}

private void animateCoin() {

handler.postDelayed(() -> {

coinFrame = (coinFrame + 1) % coinBitmaps.length;

invalidate();

animateCoin();

}, 150);

}

private void animateTrap() {

handler.postDelayed(() -> {

trapFrame = (trapFrame + 1) % trapBitmaps.length;

invalidate();

animateTrap();

}, 150);

}

@SuppressLint("DrawAllocation")

@Override

protected void onDraw(@NonNull Canvas canvas) {

super.onDraw(canvas);

int width = getWidth();

int height = getHeight();

int cellSize = Math.min(width / maze[0].length, height / maze.length);

float offsetX = (float) (width - cellSize \* maze[0].length) / 2;

float offsetY = (float) (height - cellSize \* maze.length) / 2;

for (int row = 0; row < maze.length; row++) {

for (int col = 0; col < maze[row].length; col++) {

float left = offsetX + col \* cellSize;

float top = offsetY + row \* cellSize;

float right = left + cellSize;

float bottom = top + cellSize;

if (maze[row][col] == 1) {

canvas.drawBitmap(wallBitmap, null, new RectF(left, top, right, bottom), null);

} else if (maze[row][col] == 0) {

} else if (maze[row][col] == 2) {

canvas.drawBitmap(trapBitmaps[trapFrame], null, new RectF(left, top, right, bottom), null);

}

}

}

for (int[] coin : coins) {

float cx = offsetX + coin[1] \* cellSize + (float) cellSize / 2;

float cy = offsetY + coin[0] \* cellSize + (float) cellSize / 2;

Bitmap currentCoinBitmap = coinBitmaps[coinFrame];

Bitmap scaledCoinBitmap = Bitmap.createScaledBitmap(currentCoinBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledCoinBitmap, cx - (float) scaledCoinBitmap.getWidth() / 2, cy - (float) scaledCoinBitmap.getHeight() / 2, null);

}

float playerX = offsetX + playerCol \* cellSize + (float) cellSize / 2;

float playerY = offsetY + playerRow \* cellSize + (float) cellSize / 2;

Bitmap currentPlayerBitmap = isBlinking ? playerBlinkBitmap : playerBitmap;

Bitmap scaledPlayerBitmap = Bitmap.createScaledBitmap(currentPlayerBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledPlayerBitmap, playerX - (float) scaledPlayerBitmap.getWidth() / 2, playerY - (float) scaledPlayerBitmap.getHeight() / 2, null);

}

private static final int SWIPE\_THRESHOLD = 50;

@SuppressLint("ClickableViewAccessibility")

@Override

public boolean onTouchEvent(MotionEvent event) {

switch (event.getAction()) {

case MotionEvent.ACTION\_DOWN:

startX = event.getX();

startY = event.getY();

return true;

case MotionEvent.ACTION\_UP:

float endX = event.getX();

float endY = event.getY();

float dx = endX - startX;

float dy = endY - startY;

if (Math.abs(dx) > SWIPE\_THRESHOLD || Math.abs(dy) > SWIPE\_THRESHOLD) {

if (Math.abs(dx) > Math.abs(dy)) {

if (dx > 0) movePlayerSmoothly(1, 0);

else movePlayerSmoothly(-1, 0);

} else {

if (dy > 0) movePlayerSmoothly(0, 1);

else movePlayerSmoothly(0, -1);

}

}

return true;

}

return super.onTouchEvent(event);

}

private void showToastMessage(String message) {

Toast.makeText(getContext(), message, Toast.LENGTH\_SHORT).show();

}

private void movePlayerSmoothly(final int dx, final int dy) {

handler.postDelayed(new Runnable() {

@Override

public void run() {

int nextRow = playerRow + dy;

int nextCol = playerCol + dx;

if (nextRow >= 0 && nextRow < maze.length && nextCol >= 0 && nextCol < maze[0].length && maze[nextRow][nextCol] != 1) {

playerRow = nextRow;

playerCol = nextCol;

collectCoin();

if (playerRow == exitRow && playerCol == exitCol) {

showLevelCompleteDialog();

}

if (maze[playerRow][playerCol] == 2) {

LayoutInflater inflater = LayoutInflater.from(getContext());

View layout = inflater.inflate(R.layout.custom\_toast, null);

TextView text = layout.findViewById(R.id.toast\_text);

text.setText("Ти потрапив у пастку! Обережніше!");

Toast toast = new Toast(getContext());

toast.setDuration(Toast.LENGTH\_SHORT);

toast.setView(layout);

toast.show();

resetGame();

return;

}

invalidate();

movePlayerSmoothly(dx, dy);

}

}

}, 5);

}

private void collectCoin() {

for (int i = 0; i < coins.size(); i++) {

if (coins.get(i)[0] == playerRow && coins.get(i)[1] == playerCol) {

coins.remove(i);

collectedCoins++;

if (coinSound != null) {

coinSound.start();

}

isBlinking = true;

invalidate();

int BLINK\_TIME = 500;

blinkHandler.postDelayed(new Runnable() {

@Override

public void run() {

isBlinking = false;

invalidate();

}

}, BLINK\_TIME);

break;

}

}

}

@SuppressLint("SetTextI18n")

private void showLevelCompleteDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(getContext(), R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(getContext(), R.font.pixel);

TextView titleView = new TextView(getContext());

titleView.setText("Рівень пройдено!");

titleView.setTextSize(24);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.WHITE);

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

TextView messageView = new TextView(getContext());

messageView.setText("Кількість зібраних монет: " + collectedCoins + "/5");

messageView.setTextSize(18);

messageView.setTypeface(pixelFont);

messageView.setTextColor(Color.WHITE);

messageView.setGravity(Gravity.CENTER);

messageView.setPadding(20, 10, 20, 30);

LinearLayout containerLayout = new LinearLayout(getContext());

containerLayout.setOrientation(LinearLayout.VERTICAL);

containerLayout.setGravity(Gravity.CENTER);

containerLayout.setPadding(20, 20, 20, 20);

containerLayout.addView(titleView);

containerLayout.addView(messageView);

LinearLayout buttonLayout = new LinearLayout(getContext());

buttonLayout.setOrientation(LinearLayout.HORIZONTAL);

buttonLayout.setGravity(Gravity.CENTER);

buttonLayout.setPadding(20, 20, 20, 20);

int marginInDp = 2;

float scale = getContext().getResources().getDisplayMetrics().density;

int marginInPx = (int) (marginInDp \* scale + 0.5f);

int buttonHeightDp = 50;

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.WRAP\_CONTENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(marginInPx, 0, marginInPx, 0);

ImageButton nextButton = new ImageButton(getContext());

nextButton.setImageResource(R.drawable.next);

nextButton.setBackgroundColor(Color.TRANSPARENT);

nextButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

nextButton.setAdjustViewBounds(true);

nextButton.setLayoutParams(buttonParams);

nextButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

goToNextLevel();

});

ImageButton menuButton = new ImageButton(getContext());

menuButton.setImageResource(R.drawable.menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setAdjustViewBounds(true);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

returnToMenu();

});

ImageButton restartButton = new ImageButton(getContext());

restartButton.setImageResource(R.drawable.restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

restartButton.setAdjustViewBounds(true);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

resetGame();

});

buttonLayout.addView(nextButton);

buttonLayout.addView(menuButton);

buttonLayout.addView(restartButton);

containerLayout.addView(buttonLayout);

builder.setView(containerLayout);

builder.setCancelable(false);

levelCompleteDialog = builder.create();

levelCompleteDialog.show();

if (levelCompleteDialog.getWindow() != null) {

levelCompleteDialog.getWindow().setLayout(

dpToPx(400),

WindowManager.LayoutParams.WRAP\_CONTENT

);

levelCompleteDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private void goToNextLevel() {

Context context = getContext();

Intent intent = new Intent(context, MazeActivity5.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void returnToMenu() {

Context context = getContext();

Intent intent = new Intent(context, LevelSelectionActivity.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void resetGame() {

playerRow = 37;

playerCol = 10;

collectedCoins = 0;

coins.clear();

coins.add(new int[]{25, 16});

coins.add(new int[]{11, 16});

coins.add(new int[]{23, 6});

coins.add(new int[]{1, 1});

coins.add(new int[]{33, 1});

if (levelCompleteDialog != null && levelCompleteDialog.isShowing()) {

levelCompleteDialog.dismiss();

}

invalidate();

}

private int dpToPx(int dp) {

float density = getContext().getResources().getDisplayMetrics().density;

return Math.round(dp \* density);

}

private void showGameRules() {

AlertDialog.Builder builder = new AlertDialog.Builder(getContext(), R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(getContext(), R.font.pixel);

TextView titleView = new TextView(getContext());

titleView.setText("Складність зростає:");

titleView.setTextSize(24);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.WHITE);

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

TextView messageView = new TextView(getContext());

messageView.setText("Тепер у лабіринтах з’явились пастки. Якщо наступиш на одну з них — рівень почнеться спочатку.\n\n" +

"Удачі!");

messageView.setTextSize(18);

messageView.setTypeface(pixelFont);

messageView.setTextColor(Color.WHITE);

messageView.setGravity(Gravity.CENTER);

messageView.setPadding(20, 10, 20, 30);

ImageButton closeButton = new ImageButton(getContext());

closeButton.setImageResource(R.drawable.start\_game);

closeButton.setBackgroundColor(Color.TRANSPARENT);

closeButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(

dpToPx(250), // ширина в dp

dpToPx(80) // висота в dp

);

params.gravity = Gravity.CENTER;

closeButton.setLayoutParams(params);

closeButton.setPadding(10, 10, 10, 10);

closeButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

resetGame();

});

LinearLayout containerLayout = new LinearLayout(getContext());

containerLayout.setOrientation(LinearLayout.VERTICAL);

containerLayout.setGravity(Gravity.CENTER);

containerLayout.setPadding(20, 20, 20, 20);

containerLayout.addView(titleView);

containerLayout.addView(messageView);

containerLayout.addView(closeButton);

builder.setView(containerLayout);

builder.setCancelable(false);

levelCompleteDialog = builder.create();

levelCompleteDialog.show();

if (levelCompleteDialog.getWindow() != null) {

levelCompleteDialog.getWindow().setLayout(

dpToPx(350),

WindowManager.LayoutParams.WRAP\_CONTENT

);

levelCompleteDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

}

Лістинг А.12 Layout-код файлу MazeView3

public class MazeView3 extends View {

private final MediaPlayer coinSound;

private AlertDialog levelCompleteDialog;

private final int[][] maze = {

{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},

{0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1 ,0 ,0 ,0 ,0, 0, 1},

{1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0 ,0 ,1 ,0 ,1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1 ,1 ,1 ,0 ,1, 0, 1},

{1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0 ,0 ,0 ,0 ,1, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1 ,1 ,1 ,1 ,1, 0, 1},

{1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0 ,0 ,0 ,0 ,1, 0, 1},

{1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0 ,0 ,0 ,0 ,1, 1, 1},

{1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1 ,1 ,1 ,0 ,0, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0 ,0 ,0 ,2 ,1, 0, 1},

{1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1 ,0 ,0 ,0 ,1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1 ,0 ,0 ,0 ,1, 0, 1},

{1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1 ,0 ,1 ,0 ,0, 0, 1},

{1, 0, 0, 1, 0, 0, 2, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1 ,0 ,1 ,1 ,1, 0, 1},

{1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1 ,0 ,0 ,0 ,1, 0, 1},

{1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0 ,0 ,0 ,0 ,1, 0, 1},

{1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1 ,1 ,1 ,0 ,1, 0, 1},

{1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0 ,0 ,1 ,0 ,1, 0, 1},

{1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1 ,0 ,1 ,0 ,1, 0, 1},

{1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 0, 0, 0, 1, 0, 1 ,0 ,0 ,0 ,1, 0, 1},

{1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1 ,1 ,1 ,1 ,1, 0, 1},

{1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0 ,0 ,0 ,0 ,1, 0, 1},

{1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1 ,1 ,1 ,0 ,1, 0, 1},

{1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 2, 1, 0, 1, 0, 1, 0, 0, 0, 0 ,0 ,0 ,0 ,1, 0, 1},

{1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1 ,1 ,1 ,0 ,1, 0, 1},

{1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0 ,0 ,0 ,0 ,1, 0, 1},

{1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1 ,0 ,1 ,0 ,1, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1 ,0 ,1 ,0 ,1, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1 ,0 ,1 ,1 ,1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 ,0 ,0 ,0 ,0, 0, 1},

{1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1},

{1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 1, 2, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1},

{1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},

{1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},

{1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},

{1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1},

{1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}

};

private int playerRow = 42, playerCol = 29;

private final int exitRow = 1, exitCol = 0;

private float startX, startY;

private final Handler handler = new Handler();

private final List<int[]> coins;

private int collectedCoins = 0;

private final Bitmap[] trapBitmaps = new Bitmap[8];

private final Bitmap[] coinBitmaps = new Bitmap[6];

private final Bitmap playerBitmap;

private final Bitmap playerBlinkBitmap;

private boolean isBlinking = false;

private final Handler blinkHandler = new Handler();

private final Bitmap wallBitmap;

private int coinFrame = 0;

private int trapFrame = 0;

public MazeView3(Context context, AttributeSet attrs) {

super(context, attrs);

Paint exitPaint = new Paint();

exitPaint.setColor(0xFF00FF00);

playerBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.player);

playerBlinkBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.palyer1);

for (int i = 0; i < coinBitmaps.length; i++) {

int resId = getResources().getIdentifier("coin" + i, "drawable", getContext().getPackageName());

coinBitmaps[i] = BitmapFactory.decodeResource(getResources(), resId);

}

for (int i = 0; i < trapBitmaps.length; i++) {

int resId = getResources().getIdentifier("trap" + i, "drawable", getContext().getPackageName());

trapBitmaps[i] = BitmapFactory.decodeResource(getResources(), resId);

}

coins = new ArrayList<>();

coins.add(new int[]{40, 26});

coins.add(new int[]{9, 3});

coins.add(new int[]{20, 1});

coins.add(new int[]{6, 28});

coins.add(new int[]{27, 26});

animateCoin();

animateTrap();

coinSound = MediaPlayer.create(getContext(), R.raw.click\_sound);

wallBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.wall\_texture\_3);

}

private void animateCoin() {

handler.postDelayed(() -> {

coinFrame = (coinFrame + 1) % coinBitmaps.length;

invalidate();

animateCoin();

}, 150);

}

private void animateTrap() {

handler.postDelayed(() -> {

trapFrame = (trapFrame + 1) % trapBitmaps.length;

invalidate();

animateTrap();

}, 150);

}

@SuppressLint("DrawAllocation")

@Override

protected void onDraw(@NonNull Canvas canvas) {

super.onDraw(canvas);

int width = getWidth();

int height = getHeight();

int cellSize = Math.min(width / maze[0].length, height / maze.length);

float offsetX = (float) (width - cellSize \* maze[0].length) / 2;

float offsetY = (float) (height - cellSize \* maze.length) / 2;

// Малювання лабіринту

for (int row = 0; row < maze.length; row++) {

for (int col = 0; col < maze[row].length; col++) {

float left = offsetX + col \* cellSize;

float top = offsetY + row \* cellSize;

float right = left + cellSize;

float bottom = top + cellSize;

if (maze[row][col] == 1) {

canvas.drawBitmap(wallBitmap, null, new RectF(left, top, right, bottom), null);

} else if (maze[row][col] == 0) {

} else if (maze[row][col] == 2) {

canvas.drawBitmap(trapBitmaps[trapFrame], null, new RectF(left, top, right, bottom), null);

}

}

}

for (int[] coin : coins) {

float cx = offsetX + coin[1] \* cellSize + (float) cellSize / 2;

float cy = offsetY + coin[0] \* cellSize + (float) cellSize / 2;

Bitmap currentCoinBitmap = coinBitmaps[coinFrame];

Bitmap scaledCoinBitmap = Bitmap.createScaledBitmap(currentCoinBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledCoinBitmap, cx - (float) scaledCoinBitmap.getWidth() / 2, cy - (float) scaledCoinBitmap.getHeight() / 2, null);

}

// Малювання персонажа

float playerX = offsetX + playerCol \* cellSize + (float) cellSize / 2;

float playerY = offsetY + playerRow \* cellSize + (float) cellSize / 2;

Bitmap currentPlayerBitmap = isBlinking ? playerBlinkBitmap : playerBitmap; // Якщо моргає, використовуємо моргаюче зображення

Bitmap scaledPlayerBitmap = Bitmap.createScaledBitmap(currentPlayerBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledPlayerBitmap, playerX - (float) scaledPlayerBitmap.getWidth() / 2, playerY - (float) scaledPlayerBitmap.getHeight() / 2, null);

}

private static final int SWIPE\_THRESHOLD = 50;

@SuppressLint("ClickableViewAccessibility")

@Override

public boolean onTouchEvent(MotionEvent event) {

switch (event.getAction()) {

case MotionEvent.ACTION\_DOWN:

startX = event.getX();

startY = event.getY();

return true;

case MotionEvent.ACTION\_UP:

float endX = event.getX();

float endY = event.getY();

float dx = endX - startX;

float dy = endY - startY;

if (Math.abs(dx) > SWIPE\_THRESHOLD || Math.abs(dy) > SWIPE\_THRESHOLD) {

if (Math.abs(dx) > Math.abs(dy)) {

if (dx > 0) movePlayerSmoothly(1, 0);

else movePlayerSmoothly(-1, 0);

} else {

if (dy > 0) movePlayerSmoothly(0, 1);

else movePlayerSmoothly(0, -1);

}

}

return true;

}

return super.onTouchEvent(event);

}

private void showToastMessage(String message) {

Toast.makeText(getContext(), message, Toast.LENGTH\_SHORT).show();

}

private void movePlayerSmoothly(final int dx, final int dy) {

handler.postDelayed(new Runnable() {

@Override

public void run() {

int nextRow = playerRow + dy;

int nextCol = playerCol + dx;

if (nextRow >= 0 && nextRow < maze.length && nextCol >= 0 && nextCol < maze[0].length && maze[nextRow][nextCol] != 1) {

playerRow = nextRow;

playerCol = nextCol;

collectCoin();

if (playerRow == exitRow && playerCol == exitCol) {

showLevelCompleteDialog();

}

if (maze[playerRow][playerCol] == 2) {

LayoutInflater inflater = LayoutInflater.from(getContext());

View layout = inflater.inflate(R.layout.custom\_toast, null);

TextView text = layout.findViewById(R.id.toast\_text);

text.setText("Ти потрапив у пастку! Обережніше!");

Toast toast = new Toast(getContext());

toast.setDuration(Toast.LENGTH\_SHORT);

toast.setView(layout);

toast.show();

resetGame();

return;

}

invalidate();

movePlayerSmoothly(dx, dy);

}

}

}, 5);

}

private void collectCoin() {

for (int i = 0; i < coins.size(); i++) {

if (coins.get(i)[0] == playerRow && coins.get(i)[1] == playerCol) {

coins.remove(i);

collectedCoins++;

if (coinSound != null) {

coinSound.start();

}

// Запуск моргання персонажа

isBlinking = true;

invalidate();

int BLINK\_TIME = 500;

blinkHandler.postDelayed(new Runnable() {

@Override

public void run() {

isBlinking = false;

invalidate();

}

}, BLINK\_TIME);

break;

}

}

}

@SuppressLint("SetTextI18n")

private void showLevelCompleteDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(getContext(), R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(getContext(), R.font.pixel);

TextView titleView = new TextView(getContext());

titleView.setText("Рівень пройдено!");

titleView.setTextSize(24);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.WHITE);

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

TextView messageView = new TextView(getContext());

messageView.setText("Кількість зібраних монет: " + collectedCoins + "/5");

messageView.setTextSize(18);

messageView.setTypeface(pixelFont);

messageView.setTextColor(Color.WHITE);

messageView.setGravity(Gravity.CENTER);

messageView.setPadding(20, 10, 20, 30);

LinearLayout containerLayout = new LinearLayout(getContext());

containerLayout.setOrientation(LinearLayout.VERTICAL);

containerLayout.setGravity(Gravity.CENTER);

containerLayout.setPadding(20, 20, 20, 20);

containerLayout.addView(titleView);

containerLayout.addView(messageView);

LinearLayout buttonLayout = new LinearLayout(getContext());

buttonLayout.setOrientation(LinearLayout.HORIZONTAL);

buttonLayout.setGravity(Gravity.CENTER);

buttonLayout.setPadding(20, 20, 20, 20);

int marginInDp = 2;

float scale = getContext().getResources().getDisplayMetrics().density;

int marginInPx = (int) (marginInDp \* scale + 0.5f);

int buttonHeightDp = 50;

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.WRAP\_CONTENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(marginInPx, 0, marginInPx, 0);

ImageButton menuButton = new ImageButton(getContext());

menuButton.setImageResource(R.drawable.menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setAdjustViewBounds(true);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

returnToMenu();

});

ImageButton restartButton = new ImageButton(getContext());

restartButton.setImageResource(R.drawable.restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

restartButton.setAdjustViewBounds(true);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

resetGame();

});

buttonLayout.addView(menuButton);

buttonLayout.addView(restartButton);

containerLayout.addView(buttonLayout);

builder.setView(containerLayout);

builder.setCancelable(false);

levelCompleteDialog = builder.create();

levelCompleteDialog.show();

if (levelCompleteDialog.getWindow() != null) {

levelCompleteDialog.getWindow().setLayout(

dpToPx(400),

WindowManager.LayoutParams.WRAP\_CONTENT

);

levelCompleteDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private int dpToPx(int dp) {

float density = getContext().getResources().getDisplayMetrics().density;

return Math.round(dp \* density);

}

private void goToNextLevel() {

Context context = getContext();

Intent intent = new Intent(context, MazeActivity3.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void returnToMenu() {

Context context = getContext();

Intent intent = new Intent(context, LevelSelectionActivity.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void resetGame() {

playerRow = 42;

playerCol = 29;

collectedCoins = 0;

coins.clear();

coins.add(new int[]{40, 26});

coins.add(new int[]{9, 3});

coins.add(new int[]{20, 1});

coins.add(new int[]{6, 28});

coins.add(new int[]{27, 26});

if (levelCompleteDialog != null && levelCompleteDialog.isShowing()) {

levelCompleteDialog.dismiss();

}

invalidate();

}

}

Лістинг А.13 Layout-код файлу MazeView4

public class MazeView4 extends View {

private final MediaPlayer coinSound;

private AlertDialog levelCompleteDialog;

private final int[][] maze = {

{1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1},

{1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1},

{1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1},

{1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1},

{1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1},

{1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1},

{1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1},

{1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1},

{1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1},

{1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1},

{1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1},

{1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1},

{1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1},

{1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1},

{1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1},

{1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1},

{1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1},

{1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1},

{1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1},

{1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1},

};

private int playerRow = 0, playerCol = 7;

private final int exitRow = 29, exitCol = 7;

private float startX, startY;

private final Handler handler = new Handler();

private final List<int[]> coins;

private int collectedCoins = 0;

private Bitmap playerBitmap;

private final Bitmap playerBlinkBitmap;

private boolean isBlinking = false;

private final Handler blinkHandler = new Handler();

private final Bitmap[] coinBitmaps;

private final Bitmap wallBitmap;

private final Bitmap trapBitmap;

private int coinFrame = 0;

public MazeView4(Context context, AttributeSet attrs) {

super(context, attrs);

Paint exitPaint = new Paint();

exitPaint.setColor(0xFF00FF00);

playerBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.player);

playerBlinkBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.palyer1);

coinBitmaps = new Bitmap[6];

coinBitmaps[0] = BitmapFactory.decodeResource(getResources(), R.drawable.coin0);

coinBitmaps[1] = BitmapFactory.decodeResource(getResources(), R.drawable.coin1);

coinBitmaps[2] = BitmapFactory.decodeResource(getResources(), R.drawable.coin2);

coinBitmaps[3] = BitmapFactory.decodeResource(getResources(), R.drawable.coin4);

coinBitmaps[4] = BitmapFactory.decodeResource(getResources(), R.drawable.coin5);

coinBitmaps[5] = BitmapFactory.decodeResource(getResources(), R.drawable.coin0);

coins = new ArrayList<>();

coins.add(new int[]{1, 5});

coins.add(new int[]{1, 14});

coins.add(new int[]{24, 1});

animateCoin();

coinSound = MediaPlayer.create(getContext(), R.raw.click\_sound);

wallBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.wall\_texture\_4);

trapBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.trap);

}

private void animateCoin() {

int ANIMATION\_SPEED = 200;

handler.postDelayed(new Runnable() {

@Override

public void run() {

coinFrame++;

if (coinFrame >= coinBitmaps.length) {

coinFrame = 0;

}

invalidate();

animateCoin();

}

}, ANIMATION\_SPEED);

}

@SuppressLint("DrawAllocation")

@Override

protected void onDraw(@NonNull Canvas canvas) {

super.onDraw(canvas);

int width = getWidth();

int height = getHeight();

int cellSize = Math.min(width / maze[0].length, height / maze.length);

float offsetX = (float) (width - cellSize \* maze[0].length) / 2;

float offsetY = (float) (height - cellSize \* maze.length) / 2;

for (int row = 0; row < maze.length; row++) {

for (int col = 0; col < maze[row].length; col++) {

float left = offsetX + col \* cellSize;

float top = offsetY + row \* cellSize;

float right = left + cellSize;

float bottom = top + cellSize;

if (maze[row][col] == 1) {

canvas.drawBitmap(wallBitmap, null, new RectF(left, top, right, bottom), null);

} else if (maze[row][col] == 0) {

} else if (maze[row][col] == 2) {

canvas.drawBitmap(trapBitmap, null, new RectF(left, top, right, bottom), null);

}

}

}

for (int[] coin : coins) {

float cx = offsetX + coin[1] \* cellSize + (float) cellSize / 2;

float cy = offsetY + coin[0] \* cellSize + (float) cellSize / 2;

Bitmap currentCoinBitmap = coinBitmaps[coinFrame];

Bitmap scaledCoinBitmap = Bitmap.createScaledBitmap(currentCoinBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledCoinBitmap, cx - (float) scaledCoinBitmap.getWidth() / 2, cy - (float) scaledCoinBitmap.getHeight() / 2, null);

}

float playerX = offsetX + playerCol \* cellSize + (float) cellSize / 2;

float playerY = offsetY + playerRow \* cellSize + (float) cellSize / 2;

Bitmap currentPlayerBitmap = isBlinking ? playerBlinkBitmap : playerBitmap;

Bitmap scaledPlayerBitmap = Bitmap.createScaledBitmap(currentPlayerBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledPlayerBitmap, playerX - (float) scaledPlayerBitmap.getWidth() / 2, playerY - (float) scaledPlayerBitmap.getHeight() / 2, null);

}

private static final int SWIPE\_THRESHOLD = 50;

@SuppressLint("ClickableViewAccessibility")

@Override

public boolean onTouchEvent(MotionEvent event) {

switch (event.getAction()) {

case MotionEvent.ACTION\_DOWN:

startX = event.getX();

startY = event.getY();

return true;

case MotionEvent.ACTION\_UP:

float endX = event.getX();

float endY = event.getY();

float dx = endX - startX;

float dy = endY - startY;

if (Math.abs(dx) > SWIPE\_THRESHOLD || Math.abs(dy) > SWIPE\_THRESHOLD) {

if (Math.abs(dx) > Math.abs(dy)) {

if (dx > 0) movePlayerSmoothly(1, 0);

else movePlayerSmoothly(-1, 0);

} else {

if (dy > 0) movePlayerSmoothly(0, 1);

else movePlayerSmoothly(0, -1);

}

}

return true;

}

return super.onTouchEvent(event);

}

private void showToastMessage(String message) {

Toast.makeText(getContext(), message, Toast.LENGTH\_SHORT).show();

}

private void movePlayerSmoothly(final int dx, final int dy) {

handler.postDelayed(new Runnable() {

@Override

public void run() {

int nextRow = playerRow + dy;

int nextCol = playerCol + dx;

if (nextRow >= 0 && nextRow < maze.length && nextCol >= 0 && nextCol < maze[0].length && maze[nextRow][nextCol] != 1) {

playerRow = nextRow;

playerCol = nextCol;

collectCoin();

if (playerRow == exitRow && playerCol == exitCol) {

showLevelCompleteDialog();

}

if (maze[playerRow][playerCol] == 2) {

LayoutInflater inflater = LayoutInflater.from(getContext());

View layout = inflater.inflate(R.layout.custom\_toast, null);

TextView text = layout.findViewById(R.id.toast\_text);

text.setText("Ти потрапив у пастку! Обережніше!");

Toast toast = new Toast(getContext());

toast.setDuration(Toast.LENGTH\_SHORT);

toast.setView(layout);

toast.show();

resetGame();

return;

}

invalidate();

movePlayerSmoothly(dx, dy);

}

}

}, 5);

}

private void collectCoin() {

for (int i = 0; i < coins.size(); i++) {

if (coins.get(i)[0] == playerRow && coins.get(i)[1] == playerCol) {

coins.remove(i);

collectedCoins++;

if (coinSound != null) {

coinSound.start();

}

isBlinking = true;

invalidate();

int BLINK\_TIME = 500;

blinkHandler.postDelayed(new Runnable() {

@Override

public void run() {

isBlinking = false;

invalidate();

}

}, BLINK\_TIME);

break;

}

}

}

@SuppressLint("SetTextI18n")

private void showLevelCompleteDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(getContext(), R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(getContext(), R.font.pixel);

TextView titleView = new TextView(getContext());

titleView.setText("Рівень пройдено!");

titleView.setTextSize(24);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.WHITE);

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

TextView messageView = new TextView(getContext());

messageView.setText("Кількість зібраних монет: " + collectedCoins + "/3");

messageView.setTextSize(18);

messageView.setTypeface(pixelFont);

messageView.setTextColor(Color.WHITE);

messageView.setGravity(Gravity.CENTER);

messageView.setPadding(20, 10, 20, 30);

LinearLayout containerLayout = new LinearLayout(getContext());

containerLayout.setOrientation(LinearLayout.VERTICAL);

containerLayout.setGravity(Gravity.CENTER);

containerLayout.setPadding(20, 20, 20, 20);

containerLayout.addView(titleView);

containerLayout.addView(messageView);

LinearLayout buttonLayout = new LinearLayout(getContext());

buttonLayout.setOrientation(LinearLayout.HORIZONTAL);

buttonLayout.setGravity(Gravity.CENTER);

buttonLayout.setPadding(20, 20, 20, 20);

int marginInDp = 2;

float scale = getContext().getResources().getDisplayMetrics().density;

int marginInPx = (int) (marginInDp \* scale + 0.5f);

int buttonHeightDp = 50;

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.WRAP\_CONTENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(marginInPx, 0, marginInPx, 0);

ImageButton nextButton = new ImageButton(getContext());

nextButton.setImageResource(R.drawable.next);

nextButton.setBackgroundColor(Color.TRANSPARENT);

nextButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

nextButton.setAdjustViewBounds(true);

nextButton.setLayoutParams(buttonParams);

nextButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

goToNextLevel();

});

ImageButton menuButton = new ImageButton(getContext());

menuButton.setImageResource(R.drawable.menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setAdjustViewBounds(true);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

returnToMenu();

});

ImageButton restartButton = new ImageButton(getContext());

restartButton.setImageResource(R.drawable.restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

restartButton.setAdjustViewBounds(true);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

resetGame();

});

buttonLayout.addView(nextButton);

buttonLayout.addView(menuButton);

buttonLayout.addView(restartButton);

containerLayout.addView(buttonLayout);

builder.setView(containerLayout);

builder.setCancelable(false);

levelCompleteDialog = builder.create();

levelCompleteDialog.show();

if (levelCompleteDialog.getWindow() != null) {

levelCompleteDialog.getWindow().setLayout(

dpToPx(400),

WindowManager.LayoutParams.WRAP\_CONTENT

);

levelCompleteDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private int dpToPx(int dp) {

float density = getContext().getResources().getDisplayMetrics().density;

return Math.round(dp \* density);

}

private void goToNextLevel() {

Context context = getContext();

Intent intent = new Intent(context, MazeActivity2.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void returnToMenu() {

Context context = getContext();

Intent intent = new Intent(context, LevelSelectionActivity.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void resetGame() {

playerRow = 0;

playerCol = 7;

collectedCoins = 0;

coins.clear();

coins.add(new int[]{1, 5});

coins.add(new int[]{1, 14});

coins.add(new int[]{24, 1});

if (levelCompleteDialog != null && levelCompleteDialog.isShowing()) {

levelCompleteDialog.dismiss();

}

invalidate();

}

Лістинг А.14 Layout-код файлу MazeView5

public class MazeView5 extends View {

private final MediaPlayer coinSound;

private AlertDialog levelCompleteDialog;

private final int[][] maze = {

{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},

{1, 0, 2, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1},

{1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1},

{1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 2, 1, 0, 0, 1, 0, 1, 0, 0, 1},

{1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1},

{1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1},

{1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1},

{1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1},

{1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1},

{1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1},

{1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1},

{1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1},

{1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1},

{1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1},

{1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1},

{1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 1, 0, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1},

{1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1},

{1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1},

{1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1},

{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}

};

private int playerRow = 35, playerCol = 14;

private final int exitRow = 0, exitCol = 11;

private float startX, startY;

private final Handler handler = new Handler();

private final List<int[]> coins;

private int collectedCoins = 0;

private final Bitmap[] trapBitmaps = new Bitmap[8];

private final Bitmap[] coinBitmaps = new Bitmap[6];

private Bitmap playerBitmap;

private final Bitmap playerBlinkBitmap;

private boolean isBlinking = false;

private final Handler blinkHandler = new Handler();

private final Bitmap wallBitmap;

private int coinFrame = 0;

private int trapFrame = 0;

public MazeView5(Context context, AttributeSet attrs) {

super(context, attrs);

Paint exitPaint = new Paint();

exitPaint.setColor(0xFF00FF00);

playerBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.player);

playerBlinkBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.palyer1);

for (int i = 0; i < coinBitmaps.length; i++) {

int resId = getResources().getIdentifier("coin" + i, "drawable", getContext().getPackageName());

coinBitmaps[i] = BitmapFactory.decodeResource(getResources(), resId);

}

for (int i = 0; i < trapBitmaps.length; i++) {

int resId = getResources().getIdentifier("trap" + i, "drawable", getContext().getPackageName());

trapBitmaps[i] = BitmapFactory.decodeResource(getResources(), resId);

}

coins = new ArrayList<>();

coins.add(new int[]{33, 9});

coins.add(new int[]{21, 7});

coins.add(new int[]{34, 24});

coins.add(new int[]{5, 15});

coins.add(new int[]{1, 1});

animateCoin();

animateTrap();

coinSound = MediaPlayer.create(getContext(), R.raw.click\_sound);

wallBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.wall\_texture\_5);

}

private void animateCoin() {

handler.postDelayed(() -> {

coinFrame = (coinFrame + 1) % coinBitmaps.length;

invalidate();

animateCoin();

}, 150);

}

private void animateTrap() {

handler.postDelayed(() -> {

trapFrame = (trapFrame + 1) % trapBitmaps.length;

invalidate();

animateTrap();

}, 150);

}

@SuppressLint("DrawAllocation")

@Override

protected void onDraw(@NonNull Canvas canvas) {

super.onDraw(canvas);

int width = getWidth();

int height = getHeight();

int cellSize = Math.min(width / maze[0].length, height / maze.length);

float offsetX = (float) (width - cellSize \* maze[0].length) / 2;

float offsetY = (float) (height - cellSize \* maze.length) / 2;

for (int row = 0; row < maze.length; row++) {

for (int col = 0; col < maze[row].length; col++) {

float left = offsetX + col \* cellSize;

float top = offsetY + row \* cellSize;

float right = left + cellSize;

float bottom = top + cellSize;

if (maze[row][col] == 1) {

canvas.drawBitmap(wallBitmap, null, new RectF(left, top, right, bottom), null);

} else if (maze[row][col] == 0) {

} else if (maze[row][col] == 2) {

canvas.drawBitmap(trapBitmaps[trapFrame], null, new RectF(left, top, right, bottom), null);

}

}

}

for (int[] coin : coins) {

float cx = offsetX + coin[1] \* cellSize + (float) cellSize / 2;

float cy = offsetY + coin[0] \* cellSize + (float) cellSize / 2;

Bitmap currentCoinBitmap = coinBitmaps[coinFrame];

Bitmap scaledCoinBitmap = Bitmap.createScaledBitmap(currentCoinBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledCoinBitmap, cx - (float) scaledCoinBitmap.getWidth() / 2, cy - (float) scaledCoinBitmap.getHeight() / 2, null);

}

float playerX = offsetX + playerCol \* cellSize + (float) cellSize / 2;

float playerY = offsetY + playerRow \* cellSize + (float) cellSize / 2;

Bitmap currentPlayerBitmap = isBlinking ? playerBlinkBitmap : playerBitmap;

Bitmap scaledPlayerBitmap = Bitmap.createScaledBitmap(currentPlayerBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledPlayerBitmap, playerX - (float) scaledPlayerBitmap.getWidth() / 2, playerY - (float) scaledPlayerBitmap.getHeight() / 2, null);

}

private static final int SWIPE\_THRESHOLD = 50;

@SuppressLint("ClickableViewAccessibility")

@Override

public boolean onTouchEvent(MotionEvent event) {

switch (event.getAction()) {

case MotionEvent.ACTION\_DOWN:

startX = event.getX();

startY = event.getY();

return true;

case MotionEvent.ACTION\_UP:

float endX = event.getX();

float endY = event.getY();

float dx = endX - startX;

float dy = endY - startY;

if (Math.abs(dx) > SWIPE\_THRESHOLD || Math.abs(dy) > SWIPE\_THRESHOLD) {

if (Math.abs(dx) > Math.abs(dy)) {

if (dx > 0) movePlayerSmoothly(1, 0);

else movePlayerSmoothly(-1, 0);

} else {

if (dy > 0) movePlayerSmoothly(0, 1);

else movePlayerSmoothly(0, -1);

}

}

return true;

}

return super.onTouchEvent(event);

}

private void showToastMessage(String message) {

Toast.makeText(getContext(), message, Toast.LENGTH\_SHORT).show();

}

private void movePlayerSmoothly(final int dx, final int dy) {

handler.postDelayed(new Runnable() {

@Override

public void run() {

int nextRow = playerRow + dy;

int nextCol = playerCol + dx;

if (nextRow >= 0 && nextRow < maze.length && nextCol >= 0 && nextCol < maze[0].length && maze[nextRow][nextCol] != 1) {

playerRow = nextRow;

playerCol = nextCol;

collectCoin();

if (playerRow == exitRow && playerCol == exitCol) {

showLevelCompleteDialog();

}

if (maze[playerRow][playerCol] == 2) {

LayoutInflater inflater = LayoutInflater.from(getContext());

View layout = inflater.inflate(R.layout.custom\_toast, null);

TextView text = layout.findViewById(R.id.toast\_text);

text.setText("Ти потрапив у пастку! Обережніше!");

Toast toast = new Toast(getContext());

toast.setDuration(Toast.LENGTH\_SHORT);

toast.setView(layout);

toast.show();

resetGame();

return;

}

invalidate();

movePlayerSmoothly(dx, dy);

}

}

}, 5);

}

private void collectCoin() {

for (int i = 0; i < coins.size(); i++) {

if (coins.get(i)[0] == playerRow && coins.get(i)[1] == playerCol) {

coins.remove(i);

collectedCoins++;

if (coinSound != null) {

coinSound.start();

}

isBlinking = true;

invalidate();

int BLINK\_TIME = 500;

blinkHandler.postDelayed(new Runnable() {

@Override

public void run() {

isBlinking = false;

invalidate();

}

}, BLINK\_TIME);

break;

}

}

}

@SuppressLint("SetTextI18n")

private void showLevelCompleteDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(getContext(), R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(getContext(), R.font.pixel);

TextView titleView = new TextView(getContext());

titleView.setText("Рівень пройдено!");

titleView.setTextSize(24);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.WHITE);

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

TextView messageView = new TextView(getContext());

messageView.setText("Кількість зібраних монет: " + collectedCoins + "/5");

messageView.setTextSize(18);

messageView.setTypeface(pixelFont);

messageView.setTextColor(Color.WHITE);

messageView.setGravity(Gravity.CENTER);

messageView.setPadding(20, 10, 20, 30);

LinearLayout containerLayout = new LinearLayout(getContext());

containerLayout.setOrientation(LinearLayout.VERTICAL);

containerLayout.setGravity(Gravity.CENTER);

containerLayout.setPadding(20, 20, 20, 20);

containerLayout.addView(titleView);

containerLayout.addView(messageView);

LinearLayout buttonLayout = new LinearLayout(getContext());

buttonLayout.setOrientation(LinearLayout.HORIZONTAL);

buttonLayout.setGravity(Gravity.CENTER);

buttonLayout.setPadding(20, 20, 20, 20);

int marginInDp = 2;

float scale = getContext().getResources().getDisplayMetrics().density;

int marginInPx = (int) (marginInDp \* scale + 0.5f);

int buttonHeightDp = 50;

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.WRAP\_CONTENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(marginInPx, 0, marginInPx, 0);

ImageButton nextButton = new ImageButton(getContext());

nextButton.setImageResource(R.drawable.next);

nextButton.setBackgroundColor(Color.TRANSPARENT);

nextButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

nextButton.setAdjustViewBounds(true);

nextButton.setLayoutParams(buttonParams);

nextButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

goToNextLevel();

});

ImageButton menuButton = new ImageButton(getContext());

menuButton.setImageResource(R.drawable.menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setAdjustViewBounds(true);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

returnToMenu();

});

ImageButton restartButton = new ImageButton(getContext());

restartButton.setImageResource(R.drawable.restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

restartButton.setAdjustViewBounds(true);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

resetGame();

});

buttonLayout.addView(nextButton);

buttonLayout.addView(menuButton);

buttonLayout.addView(restartButton);

containerLayout.addView(buttonLayout);

builder.setView(containerLayout);

builder.setCancelable(false);

levelCompleteDialog = builder.create();

levelCompleteDialog.show();

if (levelCompleteDialog.getWindow() != null) {

levelCompleteDialog.getWindow().setLayout(

dpToPx(400),

WindowManager.LayoutParams.WRAP\_CONTENT

);

levelCompleteDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private int dpToPx(int dp) {

float density = getContext().getResources().getDisplayMetrics().density;

return Math.round(dp \* density);

}

private void goToNextLevel() {

Context context = getContext();

Intent intent = new Intent(context, MazeActivity6.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void returnToMenu() {

Context context = getContext();

Intent intent = new Intent(context, LevelSelectionActivity.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void resetGame() {

playerRow = 35;

playerCol = 14;

collectedCoins = 0;

coins.clear();

coins.add(new int[]{33, 9});

coins.add(new int[]{21, 7});

coins.add(new int[]{34, 24});

coins.add(new int[]{5, 15});

coins.add(new int[]{1, 1});

if (levelCompleteDialog != null && levelCompleteDialog.isShowing()) {

levelCompleteDialog.dismiss();

}

invalidate();

}

}

Лістинг А.15 Layout-код файлу MazeView6

public class MazeView6 extends View {

private final MediaPlayer coinSound;

private AlertDialog levelCompleteDialog;

private final int[][] maze = {

{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},

{1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1},

{1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 2, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1},

{1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1},

{1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1},

{1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 1, 2, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 1, 0, 0, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1},

{1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1},

{1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1},

{0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0},

{1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1},

{1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1},

{1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1},

{1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1},

{1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1},

{1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 2, 1, 0, 0, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1},

{1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1},

{1, 0, 0, 0, 1, 0, 0, 2, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1},

{1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1},

{1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1},

{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}

};

private int playerRow = 20, playerCol = 0;

private final int exitRow = 21, exitCol = 24;

private float startX, startY;

private final Handler handler = new Handler();

private final List<int[]> coins;

private int collectedCoins = 0;

private final Bitmap[] trapBitmaps = new Bitmap[8];

private final Bitmap[] coinBitmaps = new Bitmap[6];

private Bitmap playerBitmap;

private final Bitmap playerBlinkBitmap;

private boolean isBlinking = false;

private final Handler blinkHandler = new Handler();

private final Bitmap wallBitmap;

private int coinFrame = 0;

private int trapFrame = 0;

public MazeView6(Context context, AttributeSet attrs) {

super(context, attrs);

Paint exitPaint = new Paint();

exitPaint.setColor(0xFF00FF00);

playerBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.player);

playerBlinkBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.palyer1);

for (int i = 0; i < coinBitmaps.length; i++) {

int resId = getResources().getIdentifier("coin" + i, "drawable", getContext().getPackageName());

coinBitmaps[i] = BitmapFactory.decodeResource(getResources(), resId);

}

for (int i = 0; i < trapBitmaps.length; i++) {

int resId = getResources().getIdentifier("trap" + i, "drawable", getContext().getPackageName());

trapBitmaps[i] = BitmapFactory.decodeResource(getResources(), resId);

}

coins = new ArrayList<>();

coins.add(new int[]{37, 13});

coins.add(new int[]{25, 5});

coins.add(new int[]{1, 13});

coins.add(new int[]{7, 7});

coins.add(new int[]{1, 1});

animateCoin();

animateTrap();

coinSound = MediaPlayer.create(getContext(), R.raw.click\_sound);

wallBitmap = BitmapFactory.decodeResource(getResources(), R.drawable.wall\_texture\_6);

}

private void animateCoin() {

handler.postDelayed(() -> {

coinFrame = (coinFrame + 1) % coinBitmaps.length;

invalidate();

animateCoin();

}, 150);

}

private void animateTrap() {

handler.postDelayed(() -> {

trapFrame = (trapFrame + 1) % trapBitmaps.length;

invalidate();

animateTrap();

}, 150);

}

@SuppressLint("DrawAllocation")

@Override

protected void onDraw(@NonNull Canvas canvas) {

super.onDraw(canvas);

int width = getWidth();

int height = getHeight();

int cellSize = Math.min(width / maze[0].length, height / maze.length);

float offsetX = (float) (width - cellSize \* maze[0].length) / 2;

float offsetY = (float) (height - cellSize \* maze.length) / 2;

for (int row = 0; row < maze.length; row++) {

for (int col = 0; col < maze[row].length; col++) {

float left = offsetX + col \* cellSize;

float top = offsetY + row \* cellSize;

float right = left + cellSize;

float bottom = top + cellSize;

if (maze[row][col] == 1) {

canvas.drawBitmap(wallBitmap, null, new RectF(left, top, right, bottom), null);

} else if (maze[row][col] == 0) {

} else if (maze[row][col] == 2) {

canvas.drawBitmap(trapBitmaps[trapFrame], null, new RectF(left, top, right, bottom), null);

}

}

}

for (int[] coin : coins) {

float cx = offsetX + coin[1] \* cellSize + (float) cellSize / 2;

float cy = offsetY + coin[0] \* cellSize + (float) cellSize / 2;

Bitmap currentCoinBitmap = coinBitmaps[coinFrame];

Bitmap scaledCoinBitmap = Bitmap.createScaledBitmap(currentCoinBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledCoinBitmap, cx - (float) scaledCoinBitmap.getWidth() / 2, cy - (float) scaledCoinBitmap.getHeight() / 2, null);

}

float playerX = offsetX + playerCol \* cellSize + (float) cellSize / 2;

float playerY = offsetY + playerRow \* cellSize + (float) cellSize / 2;

Bitmap currentPlayerBitmap = isBlinking ? playerBlinkBitmap : playerBitmap;

Bitmap scaledPlayerBitmap = Bitmap.createScaledBitmap(currentPlayerBitmap, cellSize, cellSize, false);

canvas.drawBitmap(scaledPlayerBitmap, playerX - (float) scaledPlayerBitmap.getWidth() / 2, playerY - (float) scaledPlayerBitmap.getHeight() / 2, null);

}

private static final int SWIPE\_THRESHOLD = 50;

@SuppressLint("ClickableViewAccessibility")

@Override

public boolean onTouchEvent(MotionEvent event) {

switch (event.getAction()) {

case MotionEvent.ACTION\_DOWN:

startX = event.getX();

startY = event.getY();

return true;

case MotionEvent.ACTION\_UP:

float endX = event.getX();

float endY = event.getY();

float dx = endX - startX;

float dy = endY - startY;

if (Math.abs(dx) > SWIPE\_THRESHOLD || Math.abs(dy) > SWIPE\_THRESHOLD) {

if (Math.abs(dx) > Math.abs(dy)) {

if (dx > 0) movePlayerSmoothly(1, 0);

else movePlayerSmoothly(-1, 0);

} else {

if (dy > 0) movePlayerSmoothly(0, 1);

else movePlayerSmoothly(0, -1);

}

}

return true;

}

return super.onTouchEvent(event);

}

private void showToastMessage(String message) {

Toast.makeText(getContext(), message, Toast.LENGTH\_SHORT).show();

}

private void movePlayerSmoothly(final int dx, final int dy) {

handler.postDelayed(new Runnable() {

@Override

public void run() {

int nextRow = playerRow + dy;

int nextCol = playerCol + dx;

if (nextRow >= 0 && nextRow < maze.length && nextCol >= 0 && nextCol < maze[0].length && maze[nextRow][nextCol] != 1) {

playerRow = nextRow;

playerCol = nextCol;

collectCoin();

if (playerRow == exitRow && playerCol == exitCol) {

showLevelCompleteDialog();

}

if (maze[playerRow][playerCol] == 2) {

LayoutInflater inflater = LayoutInflater.from(getContext());

View layout = inflater.inflate(R.layout.custom\_toast, null);

TextView text = layout.findViewById(R.id.toast\_text);

text.setText("Ти потрапив у пастку! Обережніше!");

Toast toast = new Toast(getContext());

toast.setDuration(Toast.LENGTH\_SHORT);

toast.setView(layout);

toast.show();

resetGame();

return;

}

invalidate();

movePlayerSmoothly(dx, dy);

}

}

}, 5);

}

private void collectCoin() {

for (int i = 0; i < coins.size(); i++) {

if (coins.get(i)[0] == playerRow && coins.get(i)[1] == playerCol) {

coins.remove(i);

collectedCoins++;

if (coinSound != null) {

coinSound.start();

}

isBlinking = true;

invalidate();

int BLINK\_TIME = 500;

blinkHandler.postDelayed(new Runnable() {

@Override

public void run() {

isBlinking = false;

invalidate();

}

}, BLINK\_TIME);

break;

}

}

}

@SuppressLint("SetTextI18n")

private void showLevelCompleteDialog() {

AlertDialog.Builder builder = new AlertDialog.Builder(getContext(), R.style.CustomDialogTheme);

Typeface pixelFont = ResourcesCompat.getFont(getContext(), R.font.pixel);

TextView titleView = new TextView(getContext());

titleView.setText("Рівень пройдено!");

titleView.setTextSize(24);

titleView.setTypeface(pixelFont);

titleView.setTextColor(Color.WHITE);

titleView.setGravity(Gravity.CENTER);

titleView.setPadding(20, 20, 20, 20);

TextView messageView = new TextView(getContext());

messageView.setText("Кількість зібраних монет: " + collectedCoins + "/5");

messageView.setTextSize(18);

messageView.setTypeface(pixelFont);

messageView.setTextColor(Color.WHITE);

messageView.setGravity(Gravity.CENTER);

messageView.setPadding(20, 10, 20, 30);

LinearLayout containerLayout = new LinearLayout(getContext());

containerLayout.setOrientation(LinearLayout.VERTICAL);

containerLayout.setGravity(Gravity.CENTER);

containerLayout.setPadding(20, 20, 20, 20);

containerLayout.addView(titleView);

containerLayout.addView(messageView);

LinearLayout buttonLayout = new LinearLayout(getContext());

buttonLayout.setOrientation(LinearLayout.HORIZONTAL);

buttonLayout.setGravity(Gravity.CENTER);

buttonLayout.setPadding(20, 20, 20, 20);

int marginInDp = 2;

float scale = getContext().getResources().getDisplayMetrics().density;

int marginInPx = (int) (marginInDp \* scale + 0.5f);

int buttonHeightDp = 50;

LinearLayout.LayoutParams buttonParams = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.WRAP\_CONTENT,

(int) (buttonHeightDp \* scale)

);

buttonParams.setMargins(marginInPx, 0, marginInPx, 0);

ImageButton nextButton = new ImageButton(getContext());

nextButton.setImageResource(R.drawable.next);

nextButton.setBackgroundColor(Color.TRANSPARENT);

nextButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

nextButton.setAdjustViewBounds(true);

nextButton.setLayoutParams(buttonParams);

nextButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

goToNextLevel();

});

ImageButton menuButton = new ImageButton(getContext());

menuButton.setImageResource(R.drawable.menu);

menuButton.setBackgroundColor(Color.TRANSPARENT);

menuButton.setAdjustViewBounds(true);

menuButton.setLayoutParams(buttonParams);

menuButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

returnToMenu();

});

ImageButton restartButton = new ImageButton(getContext());

restartButton.setImageResource(R.drawable.restart);

restartButton.setBackgroundColor(Color.TRANSPARENT);

restartButton.setScaleType(ImageView.ScaleType.FIT\_CENTER);

restartButton.setAdjustViewBounds(true);

restartButton.setLayoutParams(buttonParams);

restartButton.setOnClickListener(v -> {

ButtonSoundManager.playButtonClickSound(getContext());

levelCompleteDialog.dismiss();

resetGame();

});

buttonLayout.addView(nextButton);

buttonLayout.addView(menuButton);

buttonLayout.addView(restartButton);

containerLayout.addView(buttonLayout);

builder.setView(containerLayout);

builder.setCancelable(false);

levelCompleteDialog = builder.create();

levelCompleteDialog.show();

if (levelCompleteDialog.getWindow() != null) {

levelCompleteDialog.getWindow().setLayout(

dpToPx(400),

WindowManager.LayoutParams.WRAP\_CONTENT

);

levelCompleteDialog.getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_LAYOUT\_STABLE

);

}

}

private int dpToPx(int dp) {

float density = getContext().getResources().getDisplayMetrics().density;

return Math.round(dp \* density);

}

private void goToNextLevel() {

Context context = getContext();

Intent intent = new Intent(context, MazeActivity3.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void returnToMenu() {

Context context = getContext();

Intent intent = new Intent(context, LevelSelectionActivity.class);

context.startActivity(intent);

if (context instanceof android.app.Activity) {

((android.app.Activity) context).finish();

}

}

private void resetGame() {

playerRow = 20;

playerCol = 0;

collectedCoins = 0;

coins.clear();

coins.add(new int[]{37, 13});

coins.add(new int[]{25, 5});

coins.add(new int[]{1, 13});

coins.add(new int[]{7, 7});

coins.add(new int[]{1, 1});

if (levelCompleteDialog != null && levelCompleteDialog.isShowing()) {

levelCompleteDialog.dismiss();

}

invalidate();

}

}

Лістинг А.16 Layout-код файлу MusicServic

public class MusicService extends Service {

private static MediaPlayer mediaPlayer;

private static boolean isMusicEnabled = true; // за замовчуванням увімкнено

@Override

public void onCreate() {

super.onCreate();

if (mediaPlayer == null) {

mediaPlayer = MediaPlayer.create(this, R.raw.background\_music);

mediaPlayer.setLooping(true);

}

loadMusicSetting();

}

@Nullable

@Override

public IBinder onBind(Intent intent) {

return null;

}

private void loadMusicSetting() {

SharedPreferences prefs = PreferenceManager.getDefaultSharedPreferences(this);

isMusicEnabled = prefs.getBoolean("music\_enabled", true);

}

public static void pauseMusic() {

if (mediaPlayer != null && mediaPlayer.isPlaying()) {

mediaPlayer.pause();

}

}

public static void resumeMusic() {

if (mediaPlayer != null && !mediaPlayer.isPlaying() && isMusicEnabled) {

mediaPlayer.start();

}

}

public static void setVolume(float volume) {

if (mediaPlayer != null) {

mediaPlayer.setVolume(volume, volume);

}

}

public static void setMusicEnabled(Context context, boolean enabled) {

isMusicEnabled = enabled;

SharedPreferences prefs = PreferenceManager.getDefaultSharedPreferences(context);

prefs.edit().putBoolean("music\_enabled", enabled).apply();

if (mediaPlayer != null) {

if (enabled) {

mediaPlayer.start();

} else {

mediaPlayer.pause();

}

}

}

public static boolean isMusicEnabled() {

return isMusicEnabled;

}

}

Лістинг А.17 Layout-код файлу SystemUIHelper

public class SystemUIHelper {

public static void hideSystemUI(Activity activity) {

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.KITKAT) {

View decorView = activity.getWindow().getDecorView();

int uiOptions = View.SYSTEM\_UI\_FLAG\_FULLSCREEN

| View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

| View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY;

decorView.setSystemUiVisibility(uiOptions);

}

}

public static void showSystemUI(Activity activity) {

View decorView = activity.getWindow().getDecorView();

decorView.setSystemUiVisibility(View.SYSTEM\_UI\_FLAG\_VISIBLE);

}

}

Лістинг А.18 Layout-код файлу activity\_level\_selection.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:id="@+id/linearLayout"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="#000000"

android:gravity="center">

<ImageButton

android:id="@+id/soundToggleButton"

android:layout\_width="48dp"

android:layout\_height="48dp"

android:background="@null"

android:src="@drawable/sound\_on"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

android:scaleType="fitCenter"

android:layout\_marginTop="16dp"

android:layout\_marginEnd="16dp"/>

<ImageButton

android:id="@+id/level3Button"

android:layout\_width="125dp"

android:layout\_height="125dp"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/wall\_texture\_3"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintHorizontal\_bias="0.821"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintVertical\_bias="0.759" />

<TextView

android:id="@+id/level3Label"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:fontFamily="@font/pixel"

android:text="Level 6"

android:textColor="#FFFFFF"

android:textSize="24sp"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="@id/level3Button"

app:layout\_constraintHorizontal\_bias="0.558"

app:layout\_constraintStart\_toStartOf="@id/level3Button"

app:layout\_constraintTop\_toBottomOf="@id/level3Button"

app:layout\_constraintVertical\_bias="0.041" />

<ImageButton

android:id="@+id/level2Button"

android:layout\_width="125dp"

android:layout\_height="125dp"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/wall\_texture\_2"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintHorizontal\_bias="0.216"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintVertical\_bias="0.45" />

<TextView

android:id="@+id/level2Label"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="12dp"

android:fontFamily="@font/pixel"

android:text="Level 3"

android:textColor="#FFFFFF"

android:textSize="24sp"

app:layout\_constraintEnd\_toEndOf="@id/level2Button"

app:layout\_constraintHorizontal\_bias="0.522"

app:layout\_constraintStart\_toStartOf="@id/level2Button"

app:layout\_constraintTop\_toBottomOf="@id/level2Button" />

<ImageButton

android:id="@+id/level1Button"

android:layout\_width="125dp"

android:layout\_height="125dp"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/wall\_texture\_1"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintHorizontal\_bias="0.216"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintVertical\_bias="0.145" />

<TextView

android:id="@+id/level1Label"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="12dp"

android:fontFamily="@font/pixel"

android:text="Level 1"

android:textColor="#FFFFFF"

android:textSize="24sp"

app:layout\_constraintEnd\_toEndOf="@id/level1Button"

app:layout\_constraintHorizontal\_bias="0.489"

app:layout\_constraintStart\_toStartOf="@id/level1Button"

app:layout\_constraintTop\_toBottomOf="@id/level1Button" />

<ImageButton

android:id="@+id/level4Button"

android:layout\_width="125dp"

android:layout\_height="125dp"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/wall\_texture\_4"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintHorizontal\_bias="0.821"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintVertical\_bias="0.145" />

<TextView

android:id="@+id/level4Label"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="12dp"

android:fontFamily="@font/pixel"

android:text="Level 2"

android:textColor="#FFFFFF"

android:textSize="24sp"

app:layout\_constraintEnd\_toEndOf="@id/level4Button"

app:layout\_constraintHorizontal\_bias="0.523"

app:layout\_constraintStart\_toStartOf="@id/level4Button"

app:layout\_constraintTop\_toBottomOf="@id/level4Button" />

<ImageButton

android:id="@+id/level5Button"

android:layout\_width="125dp"

android:layout\_height="125dp"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/wall\_texture\_5"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintHorizontal\_bias="0.821"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintVertical\_bias="0.45" />

<TextView

android:id="@+id/level5Label"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="12dp"

android:fontFamily="@font/pixel"

android:text="Level 4"

android:textColor="#FFFFFF"

android:textSize="24sp"

app:layout\_constraintEnd\_toEndOf="@id/level5Button"

app:layout\_constraintHorizontal\_bias="0.523"

app:layout\_constraintStart\_toStartOf="@id/level5Button"

app:layout\_constraintTop\_toBottomOf="@id/level5Button" />

<ImageButton

android:id="@+id/level6Button"

android:layout\_width="125dp"

android:layout\_height="125dp"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/wall\_texture\_6"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintHorizontal\_bias="0.216"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintVertical\_bias="0.759" />

<TextView

android:id="@+id/level6Label"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="12dp"

android:fontFamily="@font/pixel"

android:text="Level 5"

android:textColor="#FFFFFF"

android:textSize="24sp"

app:layout\_constraintEnd\_toEndOf="@id/level6Button"

app:layout\_constraintHorizontal\_bias="0.523"

app:layout\_constraintStart\_toStartOf="@id/level6Button"

app:layout\_constraintTop\_toBottomOf="@id/level6Button" />

Лістинг А.19 Layout-код файлу activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:id="@+id/linearLayout3"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="@drawable/background"

android:gravity="center">

<ImageView

android:id="@+id/imgTitle"

android:layout\_width="350dp"

android:layout\_height="150dp"

android:src="@drawable/game\_name"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

android:layout\_marginTop="134dp" />

<ImageButton

android:id="@+id/btnStart"

android:layout\_width="250dp"

android:layout\_height="125dp"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/start"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintVertical\_bias="0.882" />

</androidx.constraintlayout.widget.ConstraintLayout>

Лістинг А.20 Layout-код файлу activity\_maze\_level\_1.xml

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:background="#000000">

<com.example.mazegame.MazeView1

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"/>

<ImageButton

android:id="@+id/btnSettings"

android:layout\_width="32dp"

android:layout\_height="24dp"

android:layout\_gravity="top|end"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/settings"

android:padding="0dp" />

</FrameLayout>

Лістинг А.21 Layout-код файлу activity\_maze\_level\_2.xml

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="#000000">

<com.example.mazegame.MazeView2

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"/>

<ImageButton

android:id="@+id/btnSettings"

android:layout\_width="32dp"

android:layout\_height="24dp"

android:layout\_gravity="top|end"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/settings2"

android:padding="0dp" />

</FrameLayout>

Лістинг А.22 Layout-код файлу activity\_maze\_level\_3.xml

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="#000000">

<com.example.mazegame.MazeView3

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"/>

<ImageButton

android:id="@+id/btnSettings"

android:layout\_width="32dp"

android:layout\_height="24dp"

android:layout\_gravity="top|end"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/settings3"

android:padding="0dp" />

</FrameLayout>

Лістинг А.23 Layout-код файлу activity\_maze\_level\_4.xml

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="#000000">

<com.example.mazegame.MazeView4

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"/>

<ImageButton

android:id="@+id/btnSettings"

android:layout\_width="32dp"

android:layout\_height="24dp"

android:layout\_gravity="top|end"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/settings4"

android:padding="0dp" />

</FrameLayout>

Лістинг А.24 Layout-код файлу activity\_maze\_level\_5.xml

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="#000000">

<com.example.mazegame.MazeView5

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"/>

<ImageButton

android:id="@+id/btnSettings"

android:layout\_width="32dp"

android:layout\_height="24dp"

android:layout\_gravity="top|end"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/settings5"

android:padding="0dp" />

</FrameLayout>

Лістинг А.25 Layout-код файлу activity\_maze\_level\_6.xml

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="#000000">

<com.example.mazegame.MazeView6

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"/>

<ImageButton

android:id="@+id/btnSettings"

android:layout\_width="32dp"

android:layout\_height="24dp"

android:layout\_gravity="top|end"

android:background="@null"

android:scaleType="fitCenter"

android:src="@drawable/settings6"

android:padding="0dp" />

</FrameLayout>

Лістинг А.26 Layout-код файлу custom\_toast.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="horizontal"

android:padding="12dp"

android:background="#DD000000"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="center">

<TextView

android:id="@+id/toast\_text"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:textColor="#FFFFFF"

android:textSize="18sp"

android:fontFamily="@font/pixel"

android:maxLines="2"

android:ellipsize="end"

android:textAlignment="center"

android:gravity="center"

android:text="Custom Toast" />

</LinearLayout>