Aplikacje mobilne. Projektowanie aplikacji Android. Animacje i inne komponenty

Animacje

Dzięki animacji możemy animować praktycznie każdy obiekt w aplikacji np. ImageView czy TextView. W aplikacji wykorzystamy komponent Spinner. Spinnery są jak rozwijane menu, które zawiera listę elementów do wyboru. Po wybraniu wartości Spinner powraca do stanu domyślnego z wybraną wartością.

1. Stwórz aplikację o nazwie PokeBallAnimationApp z RelativeLayout. Umieść na środku komponent Spinner oraz obrazek PokeBallu z Pokemenów. Spróbujemy go animować za pomocą funkcji Kotlina.
2. Przykładowy kod activity_main.xml wygląda tak:

```
?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Spinner
        android:id="@+id/anim_spinner"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="16dp"/>

    <ImageView
        android:layout_width="120dp"
        android:layout_height="120dp"
        android:src="@drawable/pokeball"
        android:id="@+id/ball"
        android:layout_centerInParent="true"/>

</RelativeLayout>
```

3. W pliku strings.xml zadeklaruj nowe stringi, które znajdą się w Spinerze.

```
    <string-array name="anim_options">
        <item>Select Animation</item>
        <item>Translate</item>
        <item>Rotate</item>
        <item>Fade</item>
        <item>Scale</item>
        <item>Animator Set</item>
        <item>ValueAnimator</item>
    </string-array>
```

4. Plik MainActivity.kt wygląda mniej więcej tak:

```
class MainActivity : AppCompatActivity(), AdapterView.OnItemSelectedListener {


    private var anims: Spinner? = null
    private var ball: ImageView? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        anims = findViewById<Spinner>(R.id.anim_spinner)
        ball = findViewById<ImageView>(R.id.ball);
        val adapter = ArrayAdapter.createFromResource(this,
                R.array.anim_options, android.R.layout.simple_spinner_item)
// Specify the layout to use when the list of choices appears
        adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)
// Apply the adapter to the spinner
        anims!!.setAdapter(adapter)
        anims!!.setOnItemSelectedListener(this);
    }

    override fun onNothingSelected(p0: AdapterView<*>?) {
    }

    override fun onItemSelected(p0: AdapterView<*>?, p1: View?, p2: Int, p3: Long) {
        if (p2 > 0) {
            if (p2 == 1) {
                translate()
            } else if (p2 == 2) {
                rotate()
            } else if (p2 == 3) {
                fade()
            } else if (p2 == 4) {
                scale()
            } else if (p2 == 5) {
                combin_anims()
            }else if (p2 == 6) {
              valueAnim()
            }
        }
```

```
}

private fun translate() {
    val ty1 = ObjectAnimator.ofFloat(ball, View.TRANSLATION_Y, 0f, 200f)
    ty1.setDuration(1000)
    ty1.interpolator = BounceInterpolator()
    ty1.start()
}

private fun rotate() {
    val rotate = ObjectAnimator.ofFloat(ball, View.ROTATION, -360f, 0f)
    rotate.setDuration(1000)
    rotate.interpolator = AccelerateInterpolator()
    rotate.start()
}

private fun fade() {
    val fade = ObjectAnimator.ofFloat(ball, View.ALPHA, 0.2f, 1.0f)
    fade.setDuration(1000)
    fade.start()
}

private fun scale() {
    val anims = AnimatorSet();
    val sX = ObjectAnimator.ofFloat(ball, View.SCALE_X, 0.2f, 1.0f)
    val sY = ObjectAnimator.ofFloat(ball, View.SCALE_Y, 0.2f, 1.0f)
    anims.playTogether(sX, sY)
    anims.interpolator = AccelerateInterpolator()
    anims.start()
}

private fun combin_anims() {

    val anims1 = AnimatorSet()
    val sX = ObjectAnimator.ofFloat(ball, View.SCALE_X, 0.2f, 1.0f)
    val sY = ObjectAnimator.ofFloat(ball, View.SCALE_Y, 0.2f, 1.0f)
    val fade = ObjectAnimator.ofFloat(ball, View.ALPHA, 0.2f, 1.0f)

    anims1.playTogether(sX, sY, fade)
    anims1.setDuration(600)

    val anims2 = AnimatorSet()

    val tx1 = ObjectAnimator.ofFloat(ball, View.TRANSLATION_Y, 0f, 200f)
    tx1.setDuration(1000)
    tx1.interpolator = BounceInterpolator()

    val rotate = ObjectAnimator.ofFloat(ball, View.ROTATION, -360f, 0f)
    rotate.setDuration(1000)
    rotate.interpolator = AccelerateInterpolator()
```

```
        anims2.playTogether(tx1, rotate)

        val final_anim = AnimatorSet();

        final_anim.play(anims1).before(anims2)
        final_anim.play(anims2)

        final_anim.addListener(object: Animator.AnimatorListener{
            override fun onAnimationRepeat(p0: Animator?) {
            }

            override fun onAnimationCancel(p0: Animator?) {
            }

            override fun onAnimationEnd(p0: Animator?) {
                Toast.makeText(applicationContext,"Animation Fininsed",Toast.LENGTH_SHORT).show();
            }

            override fun onAnimationStart(p0: Animator?) {
                Toast.makeText(applicationContext,"Animation Started",Toast.LENGTH_SHORT).show();
            }

        });

        final_anim.start()

    }

    private fun valueAnim(){
        val tx = ValueAnimator.ofFloat(200f, 0f)
        val mDuration = 1000 //in millis
        tx.duration = mDuration.toLong()
        tx.addUpdateListener { animation -> ball!!.setTranslationX(animation.animatedValue as Float)
}
        tx.start()
    }

}
```

4. Spróbuj to samo zrobić z TextView.

Shake Animation - https://www.youtube.com/watch?v=YIq0OKIhLa0
Więcej : https://developer.android.com/training/animation/overview

VideoView

Wszystkie nowe aplikacje społecznościowe mają odtwarzacz wideo, a za pomocą VideoView i MediaController możemy stworzyć własny odtwarzacz wideo do odtwarzania filmów. Spróbujmy to zrobić!

1. Stwórz nową aplikację o nazwie VideoApp z LinearLayout. Dodaj w pliku activity_main.xml komponent VideoView.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:id="@+id/relativeLayout"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  tools:context=".MainActivity">
  <VideoView
    android:id="@+id/video_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
</LinearLayout>
```

2. W Manifeście dodaj poniższe zezwolenie:

```
<uses-permission android:name="android.permission.INTERNET" />
```

3. Aby odtwarzać filmy z pamięci wewnętrznej, utwórz folder w res a następnie w tym folder raw. Umieść w folderze raw film, który zamierzamy odtworzyć, w formacie .mp4
4. W pliku MainActivity.kt umieść

```
import android.net.Uri
import android.net.Uri.parse
import android.os.Bundle
import android.widget.MediaController
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {
  override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    //Setting MediaController
    val mediaController = MediaController(this)
    mediaController.setAnchorView(video_view)
    //Setting video path in the URI
    val uri: Uri =
      parse("android.resource://" + getPackageName() + "/" + R.raw.video)
    //Setting MediaController and URI, then starting the videoView
    video_view.setMediaController(mediaController)
    video_view.setVideoURI(uri)
```

```
        video_view.requestFocus()
        video_view.start()
    }
}
```

5. W klasie MediaController można ustawić przyciski m.in. dla pauzy i innych niezbędnych przycisków.
6. W Uri ustawiamy ścieżkę wideo, która jest zapisywana w folderze raw. W tym wypadku folder raw i nazwa pliku video.
7. W przypadku kiedy chcemy uruchomić film z zewnętrznego serwera należy wprowadzić tylko jedną zmianę w powyższym kodzie, a jest to zastąpienie ścieżki URI ścieżką wideo przy użyciu **setVideoPathmetody.**

```
//Setting MediaController
    val mediaController = MediaController(this)
    mediaController.setAnchorView(video_view)
    //Setting video path in the URI
    val uri: Uri =
       parse("android.resource://" + getPackageName() + "/" + R.raw.elephant_video)
    //Setting MediaController and URI, then starting the videoView
    video_view.setMediaController(mediaController)
    video_view.setVideoURI(uri)
    video_view.requestFocus()
    video_view.start()
```

8. Spróbuj sprawdzić te metody w MediaController.

stopPlayback() - zatrzymanie odtwarzania video
pause() - wstrzymanie video
suspend() – wstrzymanie odtwarzania video.
resume() - wznowienie odtwarzania
seekTo(int millis) – uruchamia film od danej sekundy.

9. Dostosuj VideoView do trybu poziomego, pionowego oraz full screen.

Simple AudioPlayer

MediaPlayer to klasa używana do sterowania odtwarzaniem plików audio / wideo i strumieni, dostęp do wbudowanych usług odtwarzacza multimediów, takich jak odtwarzanie audio, wideo itp. Aby użyć klasy MediaPlayer , musimy wywołać instancję create jej, wywołując metodę create () tej klasy.

1. Stwórzmy aplikację o nazwie MediaPlayer z LinearLayout oraz komponentami ImageView, Seekbar, Button.
2. Plik activity_main.xml powinien wyglądać tak:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
```

```
   android:layout_height="match_parent"
   tools:context=".MainActivity"
   android:orientation="vertical"
   android:gravity="center">

<ImageView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:src="@drawable/image"/>

<SeekBar
      android:id="@+id/positionBar"
      android:layout_width="300dp"
      android:layout_height="wrap_content"
      android:layout_marginTop="30dp"/>

<LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content">
   <TextView
         android:id="@+id/elapsedTimeLabel"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="0:11"
         android:layout_marginLeft="40dp"/>

   <TextView
         android:id="@+id/remainingTimeLabel"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="-1:11"
         android:layout_marginLeft="240dp"/>
</LinearLayout>

<Button
      android:id="@+id/playBtn"
      android:layout_width="30dp"
      android:layout_height="30dp"
      android:background="@drawable/play"
      android:layout_marginTop="40dp"
      android:onClick="playBtnClick"/>

<LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="horizontal"
      android:layout_marginTop="40dp"
      android:gravity="center">
   <ImageView
         android:layout_width="18dp"
         android:layout_height="18dp"
         android:src="@drawable/sound"/>
```

```
    <SeekBar
        android:id="@+id/volumeBar"
        android:layout_width="300dp"
        android:layout_height="wrap_content"
        android:progress="50"
        android:max="100"/>
    <ImageView
        android:layout_width="26dp"
        android:layout_height="26dp"
        android:src="@drawable/sound2"/>
  </LinearLayout>
</LinearLayout>
```

3. Plik MainActivity.kt

```
import android.annotation.SuppressLint
import android.media.MediaPlayer
import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.os.Handler
import android.os.Message
import android.view.View
import android.widget.SeekBar
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

  private lateinit var mp: MediaPlayer
  private var totalTime: Int = 0

  override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)

    mp = MediaPlayer.create(this, R.raw.music)
    mp.isLooping = true
    mp.setVolume(0.5f, 0.5f)
    totalTime = mp.duration

    // Volume Bar
    volumeBar.setOnSeekBarChangeListener(
      object : SeekBar.OnSeekBarChangeListener {
        override fun onProgressChanged(seekbar: SeekBar?, progress: Int, fromUser:
Boolean) {
          if (fromUser) {
            var volumeNum = progress / 100.0f
            mp.setVolume(volumeNum, volumeNum)
          }
        }
        override fun onStartTrackingTouch(p0: SeekBar?) {
        }
        override fun onStopTrackingTouch(p0: SeekBar?) {
```

```
            }
        }
    )

    // Position Bar
    positionBar.max = totalTime
    positionBar.setOnSeekBarChangeListener(
        object : SeekBar.OnSeekBarChangeListener {
            override fun onProgressChanged(seekBar: SeekBar?, progress: Int, fromUser:
Boolean) {
                if (fromUser) {
                    mp.seekTo(progress)
                }
            }
            override fun onStartTrackingTouch(p0: SeekBar?) {
            }
            override fun onStopTrackingTouch(p0: SeekBar?) {
            }
        }
    )

    // Thread
    Thread(Runnable {
        while (mp != null) {
            try {
                var msg = Message()
                msg.what = mp.currentPosition
                handler.sendMessage(msg)
                Thread.sleep(1000)
            } catch (e: InterruptedException) {
            }
        }
    }).start()
}

@SuppressLint("HandlerLeak")
var handler = object : Handler() {
    override fun handleMessage(msg: Message) {
        var currentPosition = msg.what

        // Update positionBar
        positionBar.progress = currentPosition

        // Update Labels
        var elapsedTime = createTimeLabel(currentPosition)
        elapsedTimeLabel.text = elapsedTime

        var remainingTime = createTimeLabel(totalTime - currentPosition)
        remainingTimeLabel.text = "-$remainingTime"
    }
}
```

```
    fun createTimeLabel(time: Int): String {
        var timeLabel = ""
        var min = time / 1000 / 60
        var sec = time / 1000 % 60

        timeLabel = "$min:"
        if (sec < 10) timeLabel += "0"
        timeLabel += sec

        return timeLabel
    }

    fun playBtnClick(v: View) {

        if (mp.isPlaying) {
            // Stop
            mp.pause()
            playBtn.setBackgroundResource(R.drawable.play)

        } else {
            // Start
            mp.start()
            playBtn.setBackgroundResource(R.drawable.stop)
        }
    }

}
```

Więcej : https://www.youtube.com/watch?v=U95WuzlgOUQ

Dodatkowe zadania:
RecyclerView z CardView - https://www.youtube.com/watch?v=j29YnF-mPd8
Weather App - https://www.youtube.com/watch?v=8Yt6EibGAa8