Pokračovanie

View, Intent
Canvas, Menu
SurfaceView, Gestá

Peter Borovanský
KAI, I-18

MS-Teams: 2sf3ph4, List, github

borovan 'at' ii.fmph.uniba.sk

Hitparáda

(Hall of Fame)

DU-1

Michal S. – Hra15/anagramy

Radovan O. - Hra15/Analýza

ErikK. – Hra15/Anagramy

Adam O. - QR/Anagramy

CV-4

Kristián F.

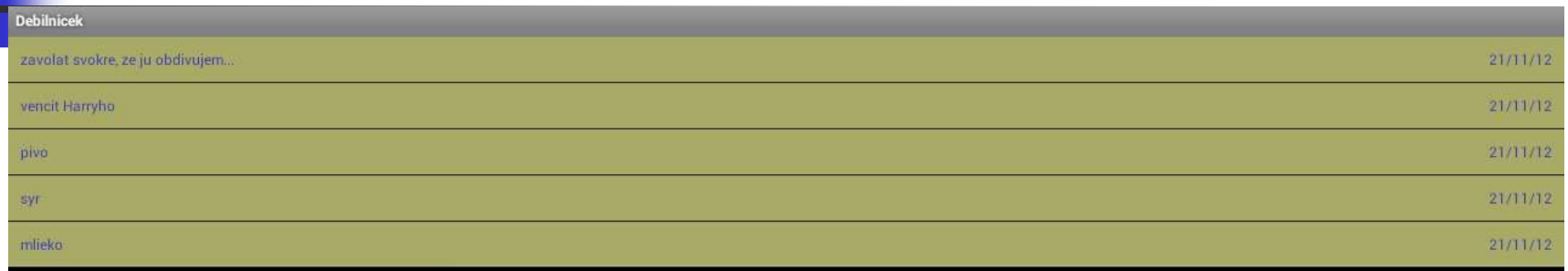
Ján M.

Adam O.

Radovan O.

Domáca úloha 2

(jedna z 3 alternatív)



Vytvorte (malú) aplikáciu zvanú Debilníček, resp. Nákupný košík:

- umožní poznamenať si, veci, predmety, činnosti do tzv. ToDo listu,
- dovolí nastaviť deadline na splnenie činnosti pomocou dátumu/času,
- ak to bude verzia nákupný košík, tak aj počet predmetov,
- umožní ich vymazať, resp. označiť za vybavené/nakúpené, resp. vymazať všetky vybavené,
- kontroluje deadline, a upozorní správou, zvukom na prešvihnutý deadline,
- pri vypnutí aplikácie si zoznam zapamätá, pri otvorení sa zoznam obnoví

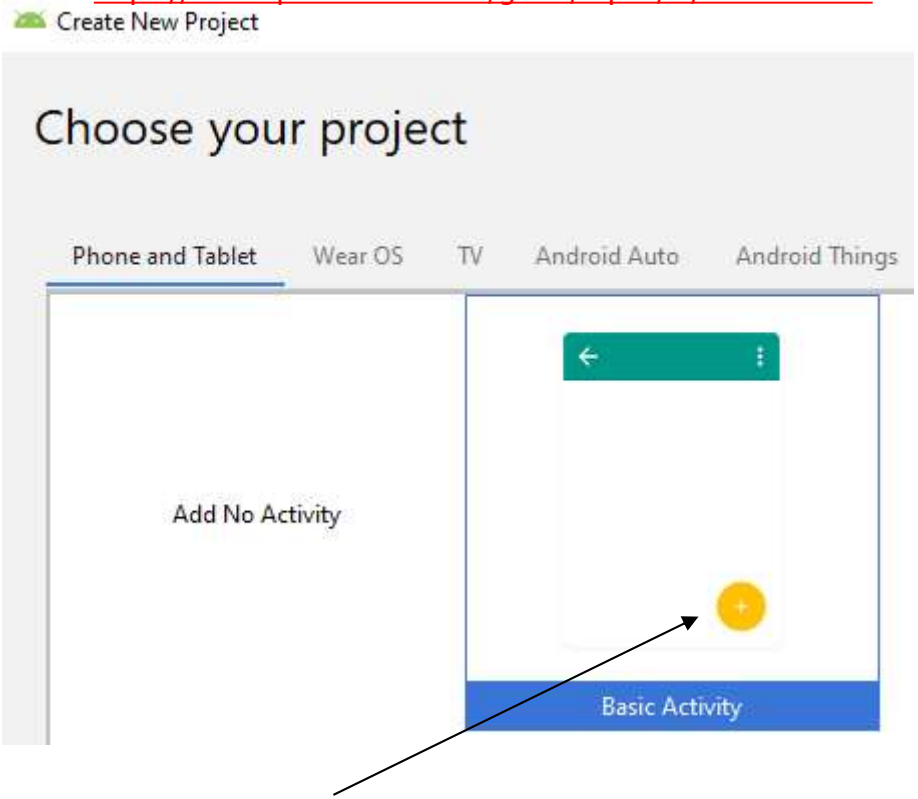
```
java.lang.Object
  ↳ android.view.View
    ↳ android.widget.ImageView
      ↳ android.widget.ImageButton
        ↳ com.google.android.material.floatingactionbutton.FloatingActionButton
```

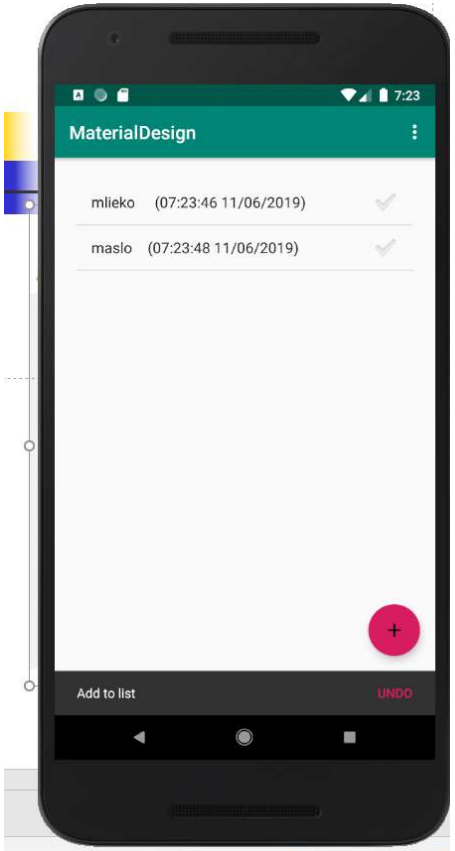
Material Design

(ako a kde začať https://material.io/)

Kotlin Android Fundamentals: 10.2
Material Design, dimens, and colors

- https://codelabs.developers.google.com/codelabs/kotlin-android-training-material-design-dimens-colors/index.html?index=..%2F..android-kotlin-fundamentals#0
- https://developer.android.com/guide/topics/ui/look-and-feel



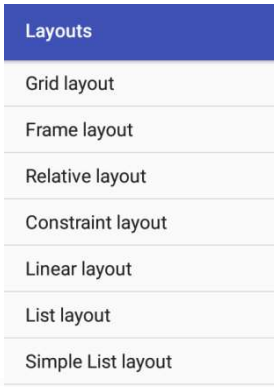


Floating Action Button

`<com.google.android.material.floatingactionbutton.FloatingActionButton`

Project: MaterialDesign.zip

Intent
(filter)



Pohľad do AndroidManifest: intent-filter hovorí, na aký intent aktivita reaguje

```
<activity android:name=".MainActivity">
   <intent-filter>
      <action android:name="android.intent.action.MAIN" />
      <category android:name="android.intent.category.LAUNCHER" />
   </intent-filter>
</activity>
```

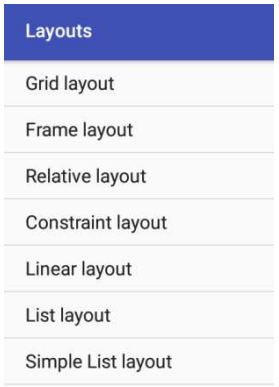
Spustí sa ako prvá

```
<activity android:name=".ListActivity">
   <intent-filter>
      <category android:name="android.intent.category.DEFAULT" />
   </intent-filter>
</activity>
```

nespustí sa

Project: Layouts2.zip

Intent
(startActivity)




```
listViewID.adapter = ArrayAdapter<String>(
    this, android.R.layout.simple_list_item_1,
    arrayListOf("Grid layout","Frame layout","Relative layout",
          "Constraint layout",)) // dáta nepatria do kódu, ale .xml
    resources.getStringArray(R.array.activities)
```

```
listViewID.setOnItemClickListener {
    adapterView, view, index, l ->
    Log.d("LISTPICK",
        "click: $index:${adapterView.getItemAtPosition(index)}")
    if (index < klasy.size)
       startActivity(Intent(this@MainActivity, klasy[index]))
}
```

```
val klasy:Array<Class<out AppCompatActivity>>= arrayOf(
    GridLayoutActivity::class.java,
    FrameLayoutActivity::class.java,
    ... .
    MainActivity::class.java
)
```

Project: Layouts2.zip

Reflection

- Reflection model v Kotline je iný ako v Jave

- ```
  val kotlinClass: KClass<GridLayoutActivity> =
      GridLayoutActivity::class
  ```

- na získanie Java class referencie, treba použiť property .java

- ```
  val javaClass: Class<GridLayoutActivity> =
      GridLayoutActivity::class.java
  val i = Intent(this@MainActivity, javaClass)
  ```

- viac o Kotlin reflection
  https://kotlinlang.org/docs/reference/reflection.html

List project

V ďalšom uvidíme sériu rôznych nezávislých aktivít, ktoré ilustrujú:

- intro_activity
  - logo, intent, CountDown/Timer, MediaPlayer
- email_activity
  - listView, intent.putExtra, startActivityForResult
- canvas_activity
  - canvas/view – Draw, MultiTouch, onTouch, Option & Context Menu
- pisky_activity
  - piškvorky, začiatok aj koniec jednoduchej hry
- login_activity
  - ukladanie informácie pomocou SharedPreferences

Project: List.zip

Intent - filter

- CATEGORY_BROWSABLE - ovláda web browser
- CATEGORY_LAUNCHER - ovláda spúšťač aplikácie
- *android.intent.action.MAIN – vstupný bod programu*

```
<intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

- CATEGORY_DEFAULT – startActivity/startActivityForResults

```
<intent-filter>
    <action android:name="com.example.list.CanvasActivity" />
    <category android:name="android.intent.category.DEFAULT" />
</intent-filter>
```

spustenie:

```
startActivity(
    Intent(this@IntroActivity, MainActivity::class.java))
```

ak máme:

```
class MainActivity : AppCompatActivity() {
```

Project: List.zip

Reflexivita

Aby sme nemuseli mať konštantu ako pole všetkých tried, trieda sa dá vyrobiť z mena triedy pomocou reflexívneho volania **`Class.forName`**

```
class MainActivity : AppCompatActivity() {
...
listView1.setOnItemClickListener {
  adapterView, view, index, l ->
    val hodnota = adapterView.getItemAtPosition(index)
    Log.d(TAG, "list item click: $index:$hodnota")
    val klasa: KClass<Any>
        = Class.forName("com.example.list.$hodnota").kotlin
    //val intent = Intent(this, IntroActivity::class.java)
    val qname: String? = klasa.qualifiedName
    Log.d(TAG, "class name: $qname")
    val intent = Intent(qname)
    startActivity(intent)
}
```

Project: List.zip

IntroActivity

(Intent, timer)

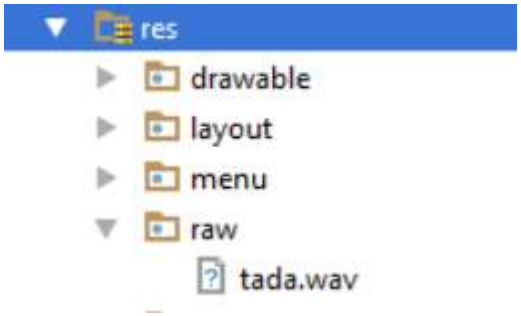
IntroActivity – CountDownTimer odpočítavajúci čas pre úvodné logo+.mp3

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

   // timer odpočítavajúci 4s.
   object: CountDownTimer(4000,1000) {
      override fun onTick(millisUntilFinished: Long) {}
      override fun onFinish() {
         Log.d(TAG, "go back to mainActivity")
         startActivity(
            Intent(this@IntroActivity, MainActivity::class.java)
         )
      }
   }.start()
```

Project: List.zip

MediaPlayer

(lokálne – adresár raw)



tada.mp3 [.wav] uložíme do project/res/raw
... a bude zakompilovaná do apky, a zazipovaná do zipky ☺

```
lateinit var mp : MediaPlayer          // globálna premenná
// ak je muzička lokálna, v res/raw/tada.wav
mp = MediaPlayer.create(this, R.raw.tada)
mp.isLooping = false
mp.start()
```

```
override fun onPause() {       // ak je IntroActivity pauzovaná, keď
   super.onPause() // odštartujeme com.example.actilist.MainActivity
   mp.release()    // uvoľníme MediaPlayer objekt, mp
   finish()        // akonáhle sa rozbehne MainActivy,
                   // IntroActivity  zanikne
                   // to rieši aj navigáciu, problém s back buttonom
}
```

Media player má problémy v niektorých emulátoroch
https://stackoverflow.com/questions/59073433/building-a-simple-mediaplayer-with-kotlin-in-android-studio-c

Project: List.zip

MediaPlayer
(onPreparedListener)

iná možnosť, tada.mp3 je prístupná niekde na sieti, dotiahneme ju a zahráme

```
// problém:apka musí deklarovať, že chce prístup na internet
lateinit var mp : MediaPlayer
try { // ak je muzička na webe, jej dotiahnutie može niečo trvať
    val uri = Uri.parse("http://dai.fmph.uniba.sk/courses/VMA/wave.mp3")
    mp.setAudioStreamType(AudioManager.STREAM_MUSIC)

    mp.setOnPreparedListener { mp.start() }

    mp.setDataSource(getApplicationContext(), uri)
    mp.prepare()     // tu sa spustí doťahovanie súboru
} catch (e:IOException) {
    e.printStackTrace()
    Toast.makeText(this, "file error", Toast.LENGTH_SHORT).show()
}
```

do AndroidManifest.xml

treba deklarovať povolenie aplikácie prístupu na internet

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

http://dai.fmph.uniba.sk/courses/VMA/wave.mp3

Project: List.zip

MediaPlayer

(na SD-karte, resp. v internej pamäti)

AndroidManifest.xml

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

Môžeme sa skúšať triafať do správnej cesty muziky, obrázku, či súboru:

```
mp.setDataSource("/mnt/sdcard/Music/tada.wav")
mp.setDataSource("/mnt/sdcard/Music/wave.mp3")
mp.setDataSource("/storage/sdcard0/Music/wave.mp3")
mp.setDataSource("/Removable/SD/Music/wave.mp3")

// ale správna cesta k prístupu k Music je cez root external storage
val filePath = Environment.getExternalStorageDirectory().toString()
                                + "/Music/tada.wav"
Log.d(TAG, filePath)   // vždy si zalogujte cestu,
                       // aby ste vedeli, kde súbor hľadá
mp = MediaPlayer()
mp.setDataSource(filePath) // hneď viete, prečo to nehrá..
mp.setOnPreparedListener { mp.start() }
mp.prepare()
```

Project: List.zip

SoundPool

```
class SoundPlayer(context: Context) {
    private val soundPool: SoundPool =
        SoundPool(10, AudioManager.STREAM_MUSIC, 0)
    init {
        try {
            val assetManager = context.assets
            var descriptor: AssetFileDescriptor  // adresár assets
                        = assetManager.openFd("shoot.ogg") //.mp3,.wav,
            shootID = soundPool.load(descriptor, 0) // toto trvá...
        } catch (e: IOException) {
            Log.e("SoundPlayer", "sound file not found")
        }

        soundPool.setOnLoadCompleteListener ({ // keď load skončil
            soundPool, sampleID, status ->     // môžeme zahrať
                soundPool.play(sampleID, 1f, 1f, 1, 1, 1f) }
        )
}  }
```

Project: List.zip

EmailActivity

(data do intentu, startActivityForResult s callbackom)

```
val emailString = edtEmail.text.toString() // dáta z formulára
val subjectString = edtSubject.text.toString()
val bodyString = edtBody.text.toString()

Toast.makeText(this@EmailActivity, "posielam mail",
               Toast.LENGTH_LONG).show()

val intent = Intent(android.content.Intent.ACTION_SEND) // SEND

intent.type = "text/plain"
intent.putExtra(android.content.Intent.EXTRA_SUBJECT,subjectString)
intent.putExtra(android.content.Intent.EXTRA_EMAIL,
                arrayOf(emailString) )       // pole adresátov
intent.putExtra(android.content.Intent.EXTRA_TEXT, bodyString)

// startActivity(intent)
startActivityForResult(intent, REQUEST_SEND_EMAIL)

private val REQUEST_SEND_EMAIL = 777
```

Project: List.zip

EmailActivity

(onActivityResult = callback)

```
private val REQUEST_SEND_EMAIL = 777

override fun onActivityResult( // CALLBACK pre startActivityForResult
        requestCode: Int,
        resultCode: Int,
        data: Intent?) {

    // Check which request we're responding to
    if (requestCode == REQUEST_SEND_EMAIL) {
        Toast.makeText(
            this@EmailActivity, "email poslany, alebo aj nie..",
            Toast.LENGTH_SHORT
        ).show();
    }
}
```

https://developer.android.com/reference/android/content/Intent.html#ACTION_SEND

Project: List.zip

Kto všetko chytá intent ?

android.content.Intent.*ACTION_SEND*

Nainštalujeme si ManifestViewer, resp. podobnú apku

https://play.google.com/store/apps/details?id=

PhotoActivity

(data z intentu)

Princíp intent-`startActivityForResult` spolu s `onActivityResult` ešte raz:

```
val takePictureIntent = Intent(MediaStore.ACTION_IMAGE_CAPTURE);
   // test, či existuje komponent, ktorý spracuje intent
if (takePictureIntent.resolveActivity(getPackageManager()) != null) {
   Toast.makeText(this@PhotoActivity,
           "smile ... taking picture", Toast.LENGTH_LONG).show();
   startActivityForResult(takePictureIntent, REQUEST_IMAGE_CAPTURE);
} else {
    Toast.makeText(this@PhotoActivity,
           "sorry ... no picture", Toast.LENGTH_LONG).show();
}

private val REQUEST_IMAGE_CAPTURE = 690
```

https://developer.android.com/reference/android/provider/MediaStore.html#ACTION_IMAGE_CAPTURE

Project: List.zip

PhotoActivity

(data z intentu)

Princíp intent-`startActivityForResult` spolu s `onActivityResult` ešte raz:

```
val takePictureIntent = Intent(MediaStore.ACTION_IMAGE_CAPTURE);
   // test, či existuje komponent, ktorý spracuje intent
if (takePictureIntent.resolveActivity(getPackageManager()) != null) {
   Toast.makeText(this@PhotoActivity,
           "smile ... taking picture", Toast.LENGTH_LONG).show();
   startActivityForResult(takePictureIntent, REQUEST_IMAGE_CAPTURE);
} else {
    Toast.makeText(this@PhotoActivity,
           "sorry ... no picture", Toast.LENGTH_LONG).show();
}

private val REQUEST_IMAGE_CAPTURE = 690
```

https://developer.android.com/reference/android/provider/MediaStore.html#ACTION_IMAGE_CAPTURE

Project: List.zip

Permissions

Povolenia (permissions) aplikácie slúžia na zabezpečenie:

- vašich privátnych dát (cez INTERNET, BLUETOOTH, ACCESS_WIFI)
- ochranu súkromia (ACCESS_FINE_LOCATION, [READ/WRITE]_CONTACTS)

Ak máte (Android <= 5.1 || target SDK < 23), tak

<uses-permissions /> sú staticky v AndroidManifest.xml,

Povolenia sa získavajú staticky pri inštalácii, ak ich užívateľ odmietne, aplikácia sa neinštaluje.

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

Inak (Android >= 6.0 && target SDK >= 23) aplikácia môže povolenia žiadať počas behu, podľa toho o akú službu práve ide (Runtime permissions).

Ak užívateľ odmietne, aplikácia beží ďalej.

https://developer.android.com/guide/topics/permissions/overview

Piškvorky

(logická hra v canvase)

Príklad logickej hry (hlavolamu), kde

- v pozadí nebeží žiadne vlákno,
- nehýbu sa postavičky,
- nemusíme pravidelne prekreslovať plochu, aby sme navodili ilúziu
  - ❑ hry
  - ❑ simulácie

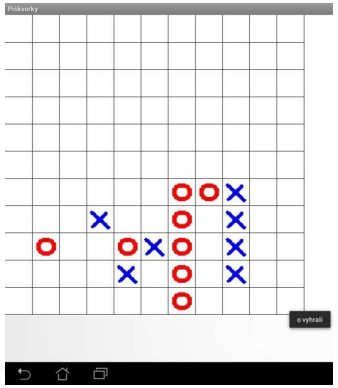
Na príklade Piškvorky ilustrujeme, že View má metódy:

- onTouch
- onDraw

Project:List.zip

onTouch vo View

(onTouchEvent)



```
class PiskyView(context: Context, attrs: AttributeSet) :
   View(context,attrs) {     // Piškvorky sú podtrieda View
                             // načítanie bitmapy obrázkov, postavičiek
    o_img = resources.getDrawable(R.drawable.o).toBitmap()
    x_img = resources.getDrawable(R.drawable.x).toBitmap()

override  fun onTouchEvent(e:MotionEvent): Boolean {
   if (e.action == MotionEvent.ACTION_DOWN) {
      val iX = (e.x / cellSize).toInt()              // transformácia
      val iY = (e.y / cellSize).toInt()              // pixlov na bunku
      if (iX >= SIZE || iY >= SIZE) return true      // mimo hraciu dosku
       if (playGround[iY][iX] == -1) {               // voľné políčko ?
          playGround[iY][iX] = onTurn                // polož značku hráča
          onTurn = 1 - onTurn                        // na ťahu, a ide súper
          invalidate()        // toto nakoniec prekreslí view
           val winner = check(iX, iY)                // vyhodnotenie víťazov...
           if (winner != -1)
              Toast.makeText(getContext(),"x vyhrali",Toast.LENGTH_LONG).show(
            } else
```

Project:List.zip

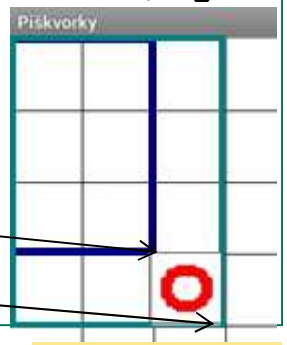
View/Activity

```
class XXX(internal var context: Context, attrs: AttributeSet) :
    View(context, attrs) {

    override fun onSizeChanged(w: Int, h: Int, oldw: Int, oldh: Int){
        super.onSizeChanged(w, h, oldw, oldh)
    }

    override fun onTouchEvent(event: MotionEvent?): Boolean {
        return super.onTouchEvent(event)
    }

    override fun onKeyDown(keyCode: Int, event: KeyEvent?):Boolean{
        return super.onKeyDown(keyCode, event)
    }

    override fun onDraw(canvas: Canvas?) {
        super.onDraw(canvas)
    }
}
```

onDraw vo View

(kreslenie do Canvas)


```
override protected fun onDraw(canvas: Canvas) { //  paint()
    minSize = Math.min(getWidth(), getHeight()) - 2
    cellSize = minSize / SIZE //  min.zrozmerov canvas/SIZE=10

    canvas.drawColor(Color.WHITE)
    val p = Paint()
    p.setColor(Color.BLACK)
    p.setStrokeWidth(1f)
    for (i in 1..SIZE) {
      canvas.drawLine(i*cellSize, 0f, i*cellSize, minSize,p)
      canvas.drawLine(0f,i*cellSize, minSize, i*cellSize, p)
    }
    for (y in 0 until SIZE) {
       for (x in 0 until SIZE) {
          canvas.drawBitmap(o_img,srcRect,
                                  destRect,
                                  p)
```



Project:List.zip

Maľovátko

(MotionEvent actions)


```
private val mPath: Path // android.graphics.Path
override protected fun onDraw(canvas:Canvas){
   super.onDraw(canvas)
   canvas.drawPath(mPath, mPaint)
}
override fun onTouchEvent(event: MotionEvent): Boolean {
   val x = event.x
   val y = event.y
   when (event.action) {
      MotionEvent.ACTION_DOWN -> {
        startTouch(x, y) invalidate()   }
      MotionEvent.ACTION_MOVE -> {
        moveTouch(x, y)  invalidate()   }
      MotionEvent.ACTION_UP -> {
        upTouch() invalidate()          }
   }
   return true
}
```

Path je sekvencia
- úsečiek
- kvadratických a
- kubických kriviek

https://examples.javacodegeeks.com/android/core/graphics/canvas-graphics/android-canvas-example/

Project:List.zip/PaintActivity

AndroidCanvasExample

PB1

Maľovátko

(bezier vs. linear - nebezier)

```
private fun startTouch(x: Float, y: Float) {
  mPath.moveTo(x, y)
  lastX = x
  lastY = y
}
private val TOLERANCE = 5f // minimálne epsilon pre pohyb
private fun moveTouch(x: Float, y: Float) {
  val dx = Math.abs(x - lastX)
  val dy = Math.abs(y - lastY)
  if (dx >= TOLERANCE || dy >= TOLERANCE) { // ak zmena bola >=
     mPath.lineTo(x, y); // kreslíme úsečku, dostaneme kostrbaté
                         // úsečka z posledného bodu path do x,y
                         // alebo
     mPath.quadTo(lastX, lastY, (x+lastX)/2, (y+lastY)/2)
                         // kreslíme časť paraboly, aproximácia
     lastX = x           // kvadratickou krivkou...
     lastY = y           // to ale potrebujeme 3 body
  }
}
```

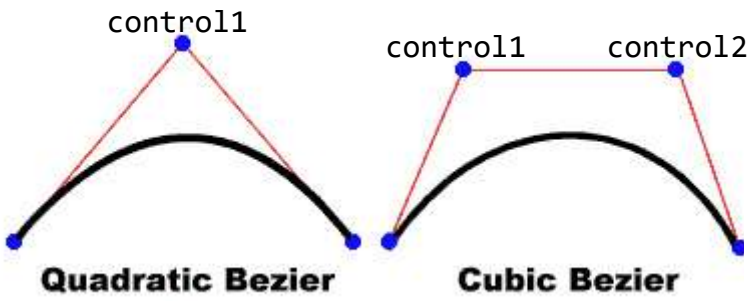
https://examples.javacodegeeks.com/android/core/graphics/canvas-graphics/android-canvas-example/

Project:List.zip/PaintActivity

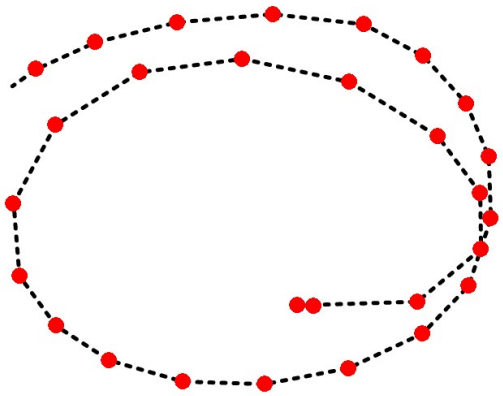
Snímka 27

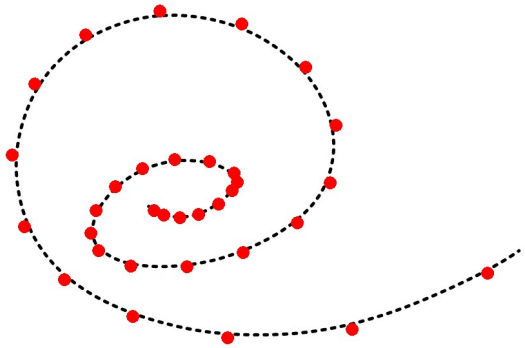
PB1 Peter Borovansky; 6. 11. 2019

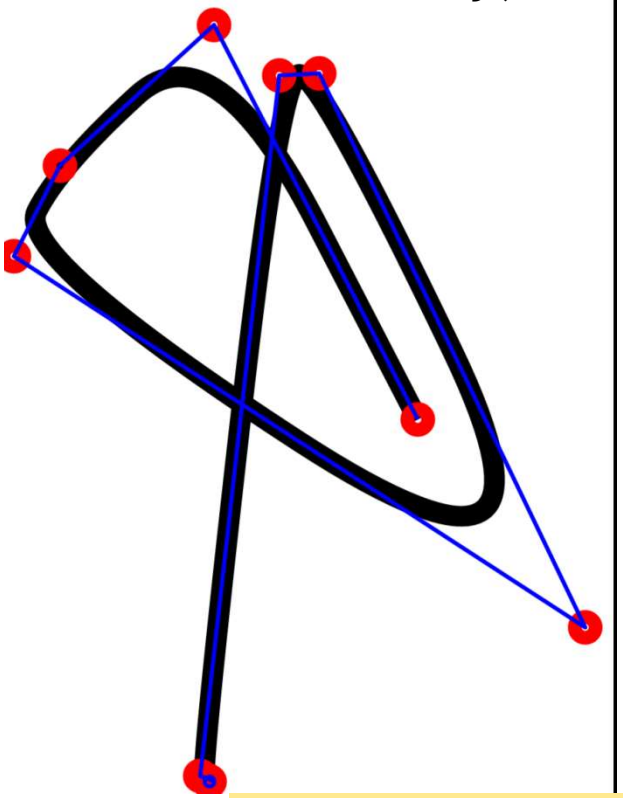
Bézier



- `lineTo(x,y)`
- `quadTo(controlX, controlY, x, y)`
- `cubeTo(controlX1, controlY1, controlX2, controlY2, x, y)`







https://www.desmos.com/calculator/ebdtbxgbq0

Project:List.zip/PaintActivity

Hierarchia

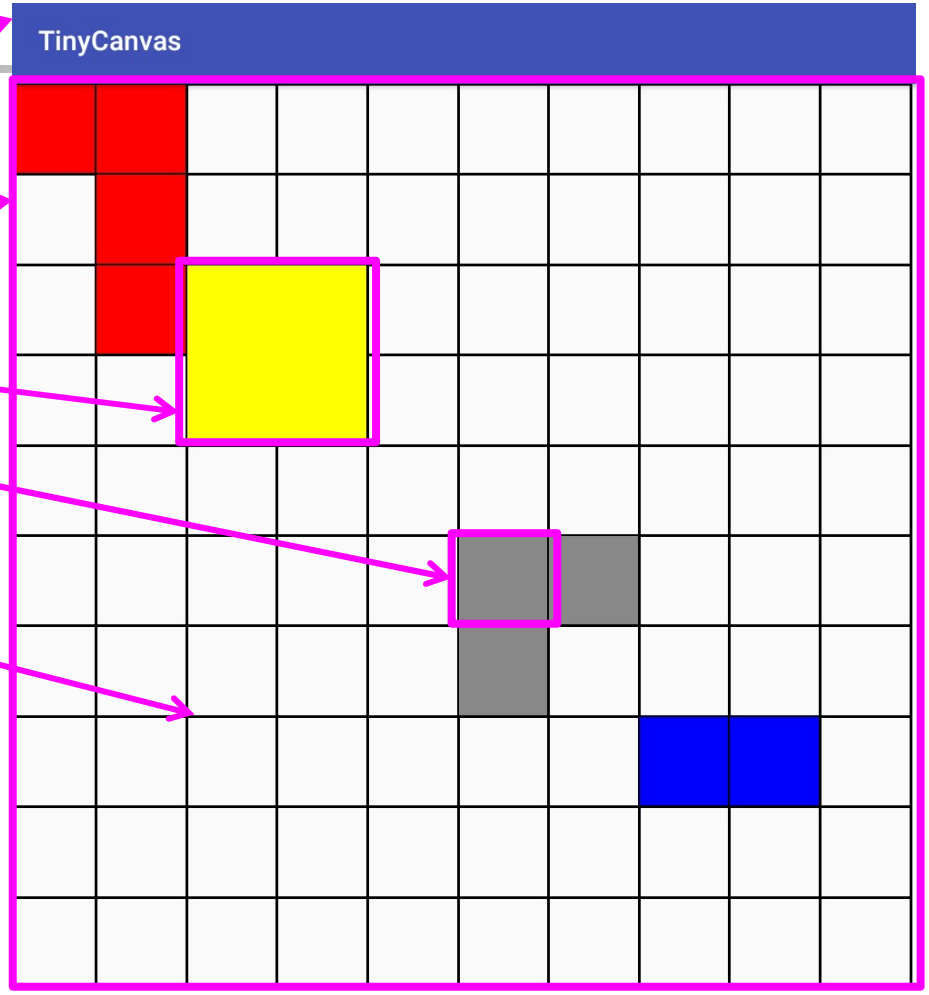
(je dôležitá pre poriadok)

Objektov/tried:

- `Canvas`
- `Scena`
- `Tvar`
- `Stvorcek`
- `Mreza`

Každý reaguje na:

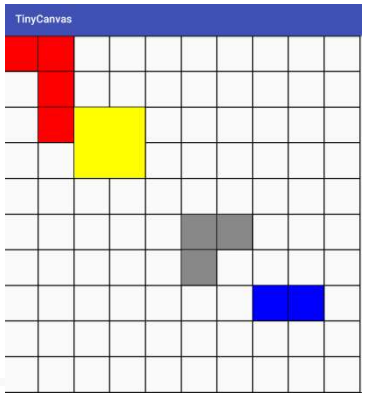
- `onTouch()`
- `onDraw()`



Project:List.zip/ParketyActivity

Hierarchia

(Tvar - Shape)

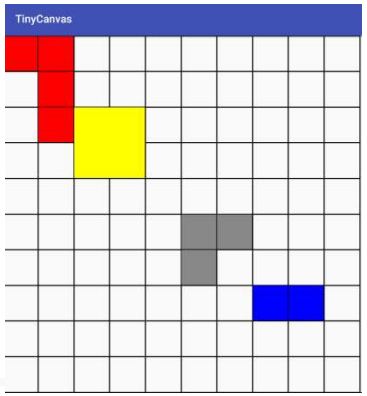



```
class Tvar(val stvoceky:List<Stvorcek>) {// Tvar je zoznam štvorčekov
    fun onDraw() {      // vykresli Tvar – vykresli každý jeho štvorček
        for (stvorcek in stvoceky) stvorcek.onDraw()
    }
    fun onTouched(motionEvent: MotionEvent): Boolean {
        if (isIn(motionEvent)) { // bol tvar zasiahnutý eventom ?
            var reDraw = false   // oznám všetkým prekresli sa
            for (stvorcek in stvoceky)
                reDraw = reDraw or stvorcek.onTouched(motionEvent)
            return reDraw        // true, ak treba invalidate()
        } else
            return false
    }
    private fun isIn(motionEvent: MotionEvent): Boolean {
        var isIn = false            // ak niektorý zo štvorčekov bol
        for (stvorcek in stvoceky) // zasiahnutý, tak Tvar bol tiež
            isIn = isIn or stvorcek.isIn(motionEvent)
        return isIn
    }
}
```

Project:List.zip/ParketyActivity

Hierarchia

(Stvorcek)

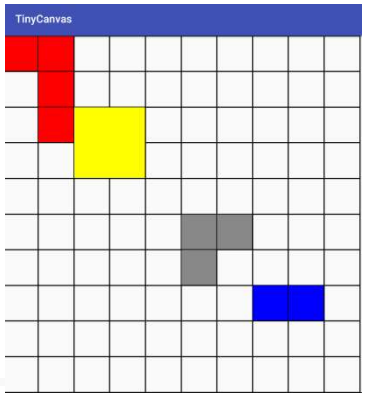



```
fun onDraw() {
     val r = Rect(x+1, y+1, x+sizeX-1, y+sizeY-1)
     CanvasView.c!!.drawRect(r, p) // chýbajú detaily, ako farba...
}
fun onTouched(event: MotionEvent): Boolean {
    int action = event.getAction()
    if (action == MotionEvent.ACTION_DOWN ) {
            ... START MOVE ... }    // začiatok dragovania štvorčeka
    else if (action == MotionEvent.ACTION_UP ||
        action == MotionEvent.ACTION_CANCEL ) {
            ... END MOVE...         // koniec dragovania, urobí 'hop'
    } else if (action == MotionEvent.ACTION_MOVE) {
            ... MOVE ... }          // počas dragovania, prekresľujeme
}
fun isIn(event: MotionEvent): Boolean { // event je v obdĺžniku
    return  x <= event.getX() && event.getX() <= x + sizeX
            &&
            y <= event.getY() && event.getY() <= y + sizeY
}
```

Project:List.zip/ParketyActivity

Hierarchia

(top level Canvas)




```
override fun onTouch(view: View,
                     motionEvent: MotionEvent): Boolean {
    if (scena.onTouched(motionEvent))
        invalidate()
    return true
}

override fun onDraw(canvas: Canvas) {
    super.onDraw(canvas)
    mreza.onDraw() // prekreslí mrežu
    scena.onDraw() // prekreslí scénu,
                   // scéna je List<Tvar>
                   // a každý Tvar je List<Stvorcek>
}
```

Objektov/tried:

- Canvas
- Scena
- Tvar
- Stvorcek
- Mreza

reagujú na:

- onTouch(event)
- onDraw()

príp.

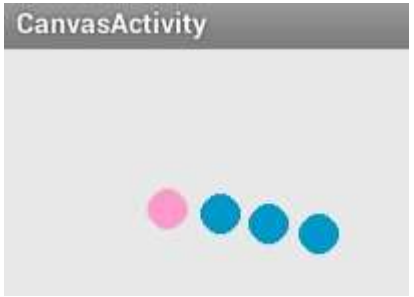
- isIn(event)

Project:List.zip/parkety/CanvasView

Vlákno (Thread) vo View

(dynamická hra v canvase, simulácia cez thread)

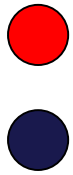
```
class CanvasView(context: Context, attrs: AttributeSet) :
    View(context,attrs), View.OnTouchListener, View.OnKeyListener {
    var touchX = 100f; var touchY = 100f        // interface
    var ballX = 200f;  var ballY = 200f

init {
    setOnTouchListener(this) setOnKeyListener(this)
    val th = object : Thread() {   // SAM - single abstract method
        override fun run() {       // život vlákna
            while (!stopped) {
                if (!paused) {     // simulácia
                    ballX += (touchX-ballX)/touches/50
                    ballY += (touchY-ballY)/touches/50
                    touchX = (ballX+50*touchX[i])/51
                    touchY = (ballY+50*touchY[i])/51
                    try {          // pozdržanie
                        Thread.sleep(100)
                        postInvalidate()   // prekreslenie v GUI vlákne
                    } catch (e: InterruptedException) {
    th.start()                     // spustenie vlákna
}
```



Project:List.zip

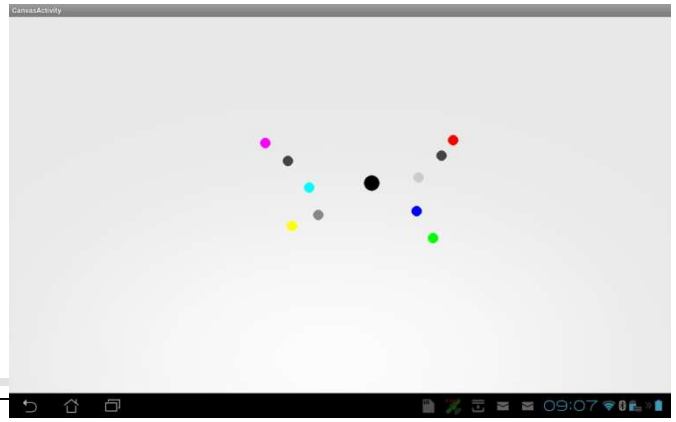
onDraw, onTouch vo View

```
override fun onDraw(canvas: Canvas?) { // paint z Appletov
    super.onDraw(canvas)
    if (canvas != null) {
        Paint p = Paint()                // kreslenie guličiek
        p.setColor(getResources().getColor(R.color.red))
        canvas.drawCircle(touchX, touchY, 10, p)
        p.setColor(getResources().getColor(R.color.blue))
        canvas.drawCircle(ballX, ballY, 10, p)
    } else
        Log.d("Canvas", "null")
}
override fun boolean onTouch(v:View, event:MotionEvent):Boolean {
    touchX = event.getX()            // netestujeme typ eventu
    touchY = event.getY()            // zoberieme len X,Y súradnice
    return true
}
```



Project:List.zip/CanvasActivity

MultiTouch



```
override fun onTouch(v: View, event: MotionEvent): Boolean {
    Log.d("Canvas", "counts:" + event.pointerCount)
    val maskedAction = event.actionMasked

    if (maskedAction == MotionEvent.ACTION_DOWN ||
        maskedAction == MotionEvent.ACTION_POINTER_DOWN) {

        touches = event.pointerCount
        for (i in 0 until event.pointerCount) {
            Log.d("Canvas", "X:" + event.getX(i))
            Log.d("Canvas", "Y:" + event.getY(i))
            touchX[i] = event.getX(i)
            touchY[i] = event.getY(i)
        }
        return true
```

Žiadne dva prsty sa nedotknú naraz

```
class CanvasView(…) : View(…),
      View.OnTouchListener,
      View.OnKeyListener {

override fun onTouch(…):Boolean
override fun onKey(...):Boolean
```

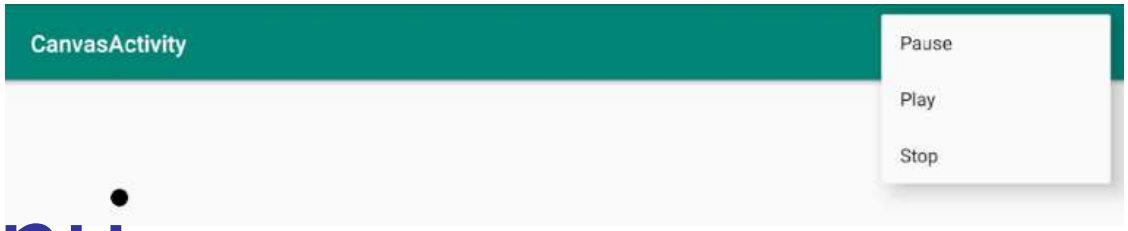
Project:List.zip/CanvasActivity

```
class CanvasView(…) : View(…),
      View.OnTouchListener,
      View.OnKeyListener {

override fun onTouch(…):Boolean
override fun onKey(...):Boolean
```

onKey vo View

```
override fun onKey(arg0: View, arg1: Int, arg2: KeyEvent):
                Boolean {
   val rnd = Random()
   when (arg1) {
      KeyEvent.KEYCODE_DPAD_LEFT -> ballX -= rnd.nextInt(50)
      KeyEvent.KEYCODE_DPAD_RIGHT -> ballX += rnd.nextInt(50)
      KeyEvent.KEYCODE_DPAD_UP -> ballY -= rnd.nextInt(50)
      KeyEvent.KEYCODE_DPAD_DOWN -> ballY += rnd.nextInt(50)
      KeyEvent.KEYCODE_SPACE -> {
         ballX += rnd.nextInt(100) - 50
         ballY += rnd.nextInt(100) - 50
      }
      else -> return false
   }
   invalidate()
   return true // event handled
}
```

Project:List.zip/CanvasActivity



Option Menu

(onCreateOptionMenu)

```
<menu
   xmlns:android="http://schemas.android.com/apk/res/android">
   <item android:id="@+id/pause" android:icon="@drawable/pause"
                 android:title="Pause">
   </item>
   <item android:id="@+id/play"  android:icon="@drawable/play"
         android:title="Play">
   </item>
   <item android:id="@+id/stop" android:icon="@drawable/stop"
         android:title="Stop">
   </item>
</menu>
```

```
override fun onCreateOptionsMenu(menu: Menu): Boolean {
   val inflater = menuInflater
   inflater.inflate(R.menu.activity_canvas, menu)
   return super.onCreateOptionsMenu(menu)
}
```

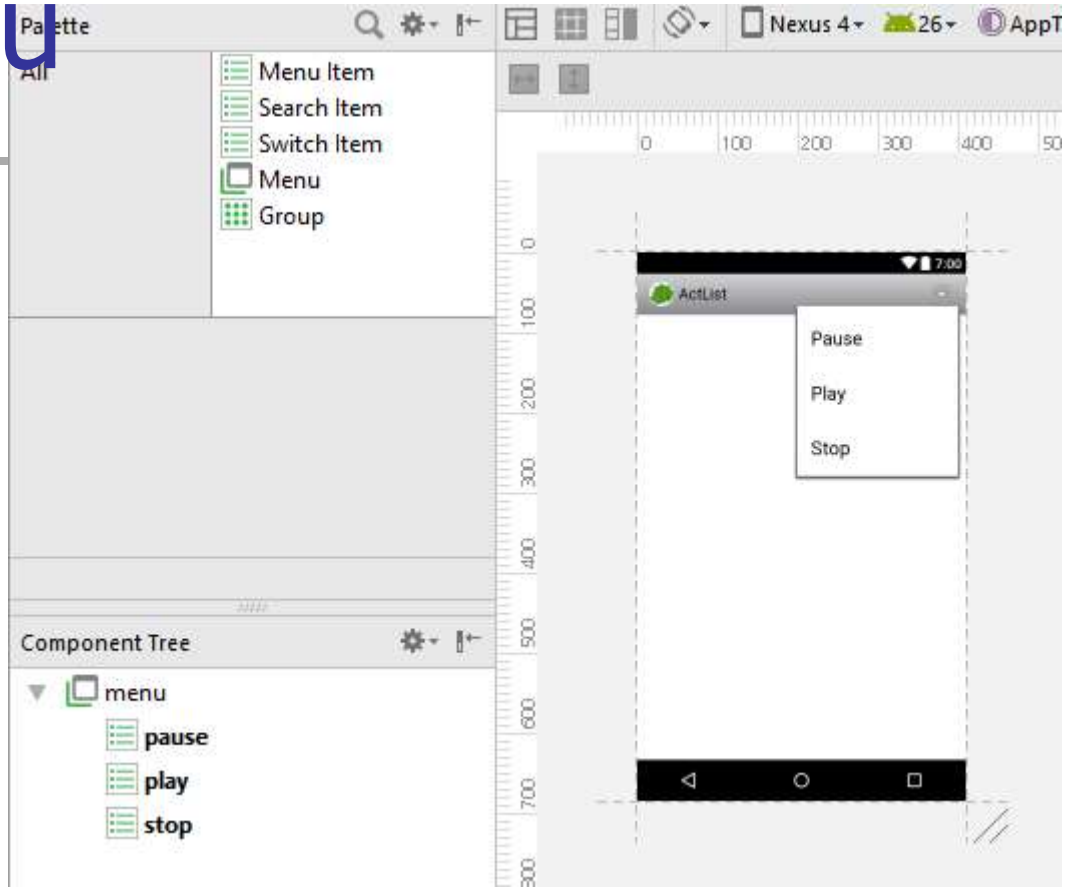
Project:List.zip

Option Menu

(onCreateOptionMenu)

Rovnako dobre to môžete navrhovať v editore

Spôsob zobrazenia a renderovania závisí na API level zariadenia




```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
   <item android:id="@+id/pause" android:icon="@drawable/pause" android:title="Pause"> </item>
   <item android:id="@+id/play"  android:icon="@drawable/play"  android:title="Play">  </item>
   <item android:id="@+id/stop"  android:icon="@drawable/stop"  android:title="Stop">  </item>
</menu>
```

Project:List.zip

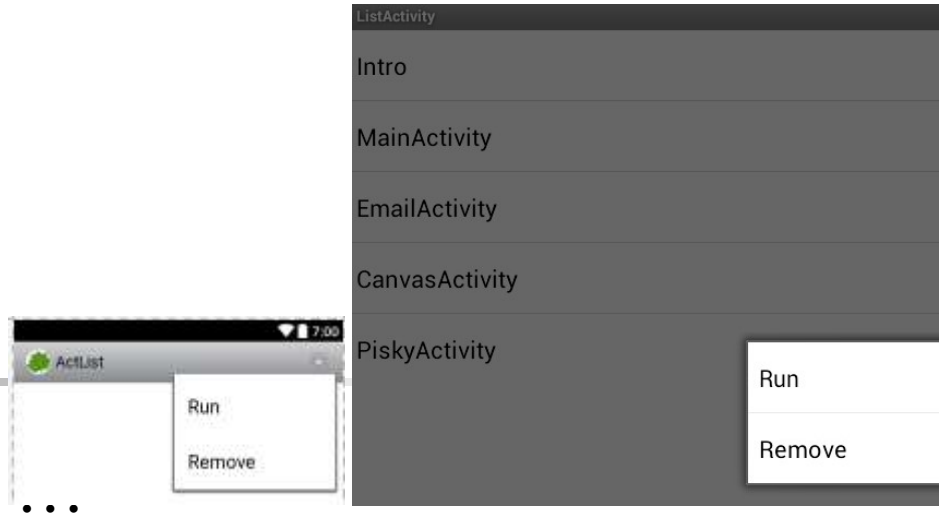
Option Menu

```
Thread th = new Thread() {
fun run() {
    while (!stopped) {
        if (!paused) {
            ...
```

```
override fun onOptionsItemSelected(item: MenuItem): Boolean {
    when (item.getItemId()) {
        R.id.pause -> {
            canvasView1?.paused = true
            return true
        }
        R.id.play -> {
            canvasView1?.paused = false
            return true
        }
        R.id.stop -> {
            canvasView1?.stopped = true
            return true
        }
        else -> return super.onOptionsItemSelected(item)
    }
}
```

Project:List.zip

Context Menu




```
override fun onCreate(
   savedInstanceState:Bundle?) {  ...
  registerForContextMenu(listView1)  // rozdiel od OptionMenu
} ContextMenu (oproti OptionMenu) treba registrovať k príslušnému view

override fun onCreateContextMenu(menu: ContextMenu?, v: View?,
                menuInfo: ContextMenu.ContextMenuInfo? ) {
     getMenuInflater().inflate(R.menu.list_menu, menu)
}   // v je View, na ktoré bolo spáchané ContextMenu Action

override fun onContextItemSelected(item: MenuItem): Boolean {
  val info = item.getMenuInfo() as AdapterContextMenuInfo
  val className = actList.get(info.id.toInt())
  when (item.getItemId()) {
     R.id.remove -> {
         actList.removeAt(info.id.toInt())
         la.notifyDataSetChanged()
         return true
     }
```

Project:List.zip

invalidate() vs. postInvalidate()

(sumár poznatkov)

vo **View**, ak chceme modifikovať obsah, používame:

- `view.invalidate()` v GUI vlákne, t.j. v event handleroch onKey, onTouch
- `view.postInvalidate()` v iných (non-GUI) vláknach, ktoré chcú view modifikovať, alternatíva `Activity.runOnUiThread` (z minulej prednášky)

toto však nenastane hneď (podobne, ako Event Dispatch Thread vo JavaFx)

nastane to po VSYNC (vertical synchronization), 40 fps ~ každých 25 ms

Všetky View sú kreslené v jednom GUI vlákne. Preto, ak

- chceme lepšie kontrolovať renderovanie (veľa) objektov,

resp.

- renderovanie objektov trvá dlho

používame triedu **SurfaceView**. To je však náročnejšie

- na cpu
- programovanie.

SurfaceView

(podtrieda View, nadtrieda ako GLSurfaceView, VideoView)

SurfaceView je typicky renderované iným vláknom pomocou SurfaceHolder.Callback

```
class GamePanel(context:Context) : SurfaceView(context),
                                   SurfaceHolder.Callback {

lateinit var thread : GameThread            // vlákno hry
init {
   getHolder().addCallback(this) //kto implementuje SurfaceHolder
   thread = GameThread(this)
   setFocusable(true)
}
override fun surfaceCreated(holder: SurfaceHolder?) {
   thread.start()                    // entry point pre SurfaceView
}

override fun surfaceDestroyed(holder: SurfaceHolder?) {
   // exit point SfV-treba zastaviť vlákno hry a počkať kým skončí
   // viď priložený projekt...
```

https://developer.android.com/reference/android/view/SurfaceView.html

Project:List.zip

GameThread

(čo robí vlákno hry - alternatíva k invalidate)

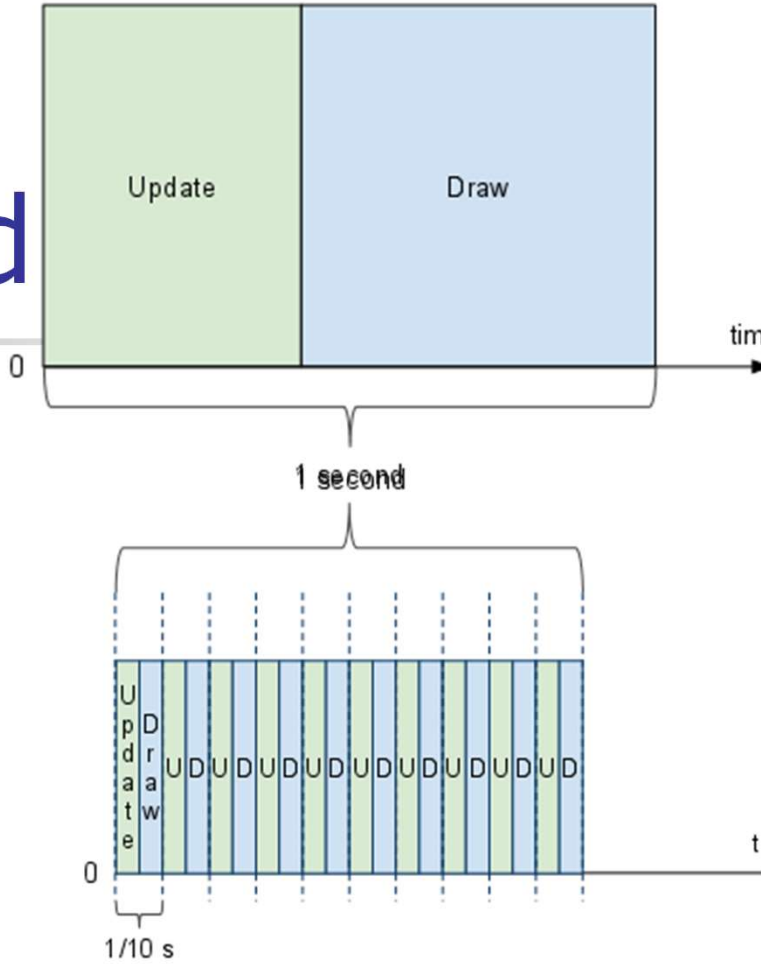
```
class GameThread(val gamePanel: GamePanel) : Thread() {
                                // zapamätáme v konštruktore GameTread
override fun run() {  // hlavný cyklus vlákna, hry, simulácie
   val surfaceHolder = gamePanel.holder
   while (running) {
        try {
           canvas = surfaceHolder.lockCanvas()
           synchronized (surfaceHolder) {
                for (pika in gamePanel.pikaList)
                    pika.update(gamePanel.getWidth(),
                                gamePanel.getHeight())
                gamePanel.showPika(canvas) // draw
                running = gamePanel.killed < gamePanel.pika.length
           try {Thread.sleep(FRAME_PERIOD-elapsedTime)} catch (){}
           }
        } finally {
           surfaceHolder.unlockCanvasAndPost(canvas) }
```

vlákno nemusí byť jediné

elapsedTime

Project:List.zip

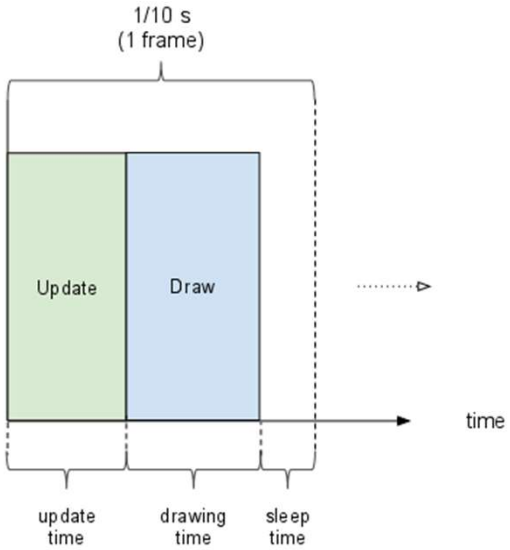
Frame per second

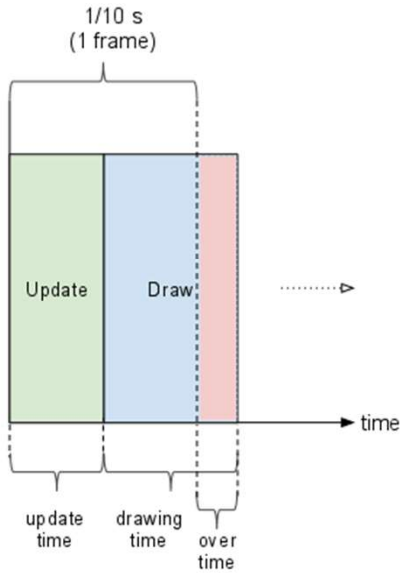


- 1 Frame per Second

Chceli by sme viac, napr. 10 fps

```
FRAME_PERIOD = 1000 / 10 //10 fps
Môže sa nám stať, že to
    stihneme  alebo   nestihneme
```

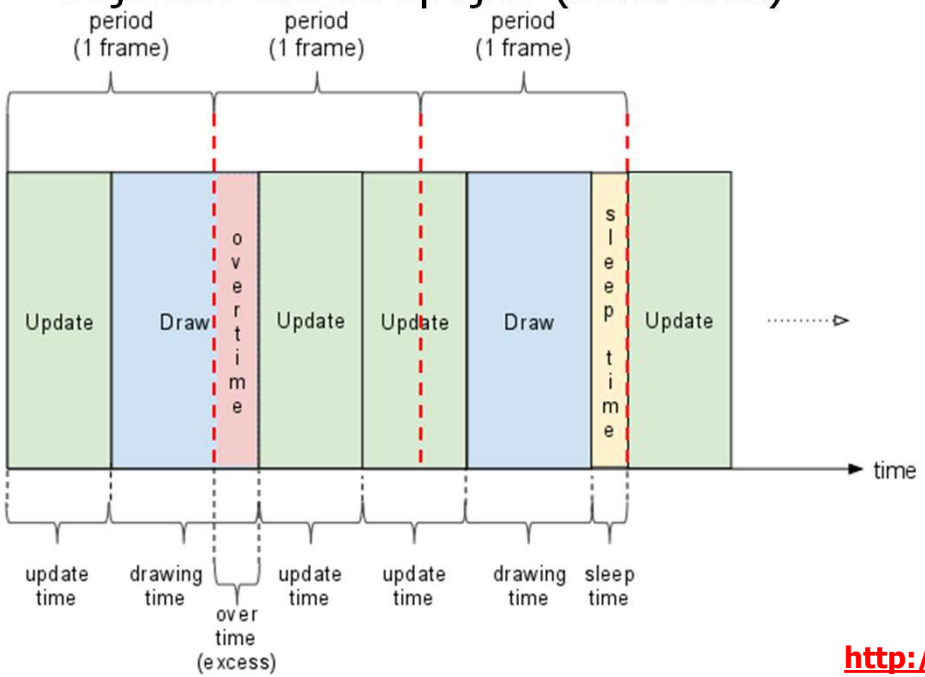




http://obviam.net/index.php/the-android-game-loop/

Čo ak nestíhame vykreslovať

- ak nestíhame vykreslovať, nemali by sme zmenšiť rýchlosť hry,
- rýchlosť hry nie je rýchlosť vykreslovania,
- radšej niektoré prekreslenia scény vynecháme, sústredíme sa na update stavu hry,
- výsledkom je hra, ktorá sa nespomaluje kvôli vykreslovaniu, ale pohyby objektov nie sú spojité (seká to...)

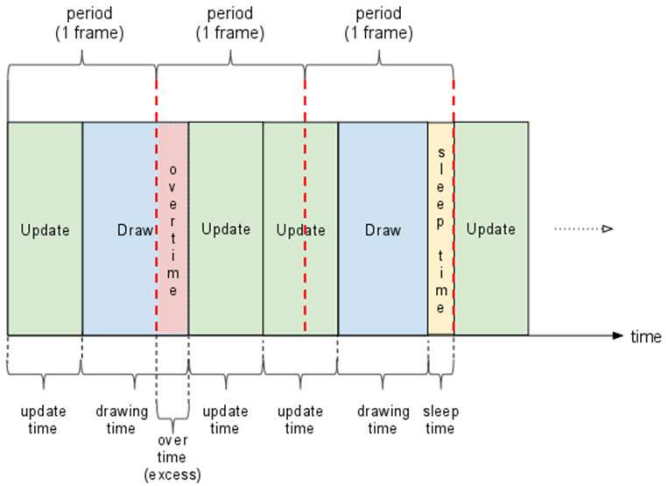


http://obviam.net/index.php/the-android-game-loop/

```
FRAME_PERIOD = 1000/50; //50 fps
```

Preskočíme pár vykreslovaní

```
if (elapsedTime <= FRAME_PERIOD) { // lepší prípad, stíhame
   try {                           // počkáme zvyšný čas
       Thread.sleep(FRAME_PERIOD - elapsedTime)
   } catch (InterruptedException e) {}
}
while (elapsedTime > FRAME_PERIOD) { // nestíhame
   for (pika in gamePanel.pikaList)
       pika.update(r.getWidth(), r.getHeight())
   elapsedTime -= FRAME_PERIOD
   skippedInPeriod++
}
framesInPeriod++
```



http://obviam.net/index.php/the-android-game-loop/

GLSurfaceView

- openGL renderer
- detaily v kóde pre tých, čo sú 3D...

- Prémia: Prezentácia bakalárky
- ak tvoríte android appku
- môžete jú vymeniť za jednu DÚ
- ak budete niečo prezentovať v Dec

Gestá
(štandardné)

```
class GesturesActivity : AppCompatActivity(),
    GestureDetector.OnGestureListener,
    GestureDetector.OnDoubleTapListener {
    lateinit var gDetector: GestureDetectorCompat

GestureDetector.OnDoubleTapListener:
override fun onDoubleTap(event: MotionEvent): Boolean
override fun onDoubleTapEvent(event: MotionEvent): Boolean
override fun onSingleTapConfirmed(event: MotionEvent): Boolean

GestureDetector.OnGestureListener:
override fun onDown(event: MotionEvent): Boolean
override fun onFling(event1: MotionEvent, event2: MotionEvent,
                     velocityX: Float, velocityY: Float):Boolean
override fun onLongPress(event: MotionEvent)
override fun onScroll(e1: MotionEvent, e2: MotionEvent,
                      distanceX: Float, distanceY: Float):Boolean
override fun onShowPress(event: MotionEvent)
override fun onSingleTapUp(event: MotionEvent): Boolean
```

Project:List.zip

Gestá
(vlastné)


```
class GesturesActivity : AppCompatActivity(),
        OnGesturePerformedListener {
   lateinit var gLibrary: GestureLibrary
...
gLibrary = GestureLibraries.fromRawResource(this,
        R.raw.gestures2          // tento súbor si
        )    // vyrobíte v Gesture Editore, uložíte do raw/
if (gLibrary.load() == false) {
    finish()
}
gOverlay.addOnGesturePerformedListener {
  overlay: GestureOverlayView, gesture: Gesture ->
    val predictions = gLibrary.recognize(gesture)
    predictions?.let {
      if (it.size > 0 && it[0].score > 1.0) {
        val action = it[0].name
        Toast.makeText(this, action, Toast.LENGTH_SHORT).show()
      }
    }
}
```

Project:GestureBuilder.zip