KIV / MKZ

Cvičení 5, 2016,v2

L. Pešička

Téma cvičení

- Rozbor cvičné hry
- Ošetření UI z jiného vlákna
  - Asynchronní task
  - Handler
  - Post
- Vibrace
- Přehrávání hudby
- Animace (točící se text)

HERNÍ APLIKACE

- Pohybující se postavy (sprity)
- Zpracování akce hráče (dotyková událost)
- Detekce kolize (hráč zasáhl sprite)
- Dočasný sprite (grafické znázornění kolize)

- Různé množství pohybujících se postav

- Hra je popsána na: http://www.edu4java.com/en/androidgame/androidgame1.html
- Projekt s danou hrou: mkz_hra2016.zip

HERNÍ APLIKACE

Cílem je seznámit se s následující problematikou psaní her:

- Použití SurfaceView
- Pokud hra bez časování – proměnlivá rychlost animace
- **Jak správně časovat** – čekáme sleep() do zbytku požadovaného intervalu není tedy sleep pevný čas, ale sleep (interval – čas_posledního vykreslení), pokud by vycházela záporná hodnota (nestihli jsme), potom sleep(pevný čas)
- **Sprite** a jeho vlastnosti, více rozfázovaných obrázků v rámci jednoho png souboru
- **Pohyb spritu** – goniometrické funkce
- **Více spritů** najednou
- Obsluha **dotykové události** (hráč), detekce kolize se spritem
- Přidání **efektu po kolizi** (dočasné sprity)

HRA - MAIN ACTIVITY

- requestWindowFeature(Window.FEATURE_NO_TITLE);
  - Pro lepší dojem ze hry
- setContentView(new GameView(this));
  - Nepoužijeme XML, celá obrazovka pro naše View
- MediaPlayer mp = MediaPlayer.create(getBaseContext(), R.raw.scream9);
- mp.start();
  - Ukázka přehrání zvuku na začátku hry

HRA - GAME VIEW I.

- Od SurfaceView
  - Vlastní rychlé překreslování
- Seznam spritů
  - Pohybující se postavy
- Seznam dočasných spritů
  - Statické skvrny po kolizi, dočasně zobrazené

- getHolder().addCallback(new SurfaceHolder.Callback() …
  - Budeme reagovat na různé stavy:
  - SurfaceDestroyed – ukončíme hru
  - SurfaceCreated – pustíme hru
  - SurfaceChanged

HRA - GAME VIEW II.

- Metoda onDraw
  - canvas.drawColor(Color.BLACK);
  - Vykreslíme statické skvrny
  - Vykreslíme pohyblivé objekty

```
protected void onDraw(Canvas canvas) {

    canvas.drawColor(Color.BLACK);

    for (int i = temps.size() - 1; i >= 0; i--) {

        temps.get(i).onDraw(canvas);

    }

    for (Sprite sprite : sprites) {

        sprite.onDraw(canvas);

    }

}
```

HRA - GAMEVIEW III.

- public boolean onTouchEvent(MotionEvent event)

- if (System.currentTimeMillis() - lastClick > 300)
  - Pokud byl poslední klik nedávno, nezapočítáme
  - Rozdíl musí být alespoň 0.3sekundy
  - Proti švindlování uživatele

- float x = event.getX();
  float y = event.getY();
  - Souřadnice dotykové události

- Pokud je kolize -> uživatel zasáhl objekt
  - Odebereme sprite
  - Přidáme skvrnu

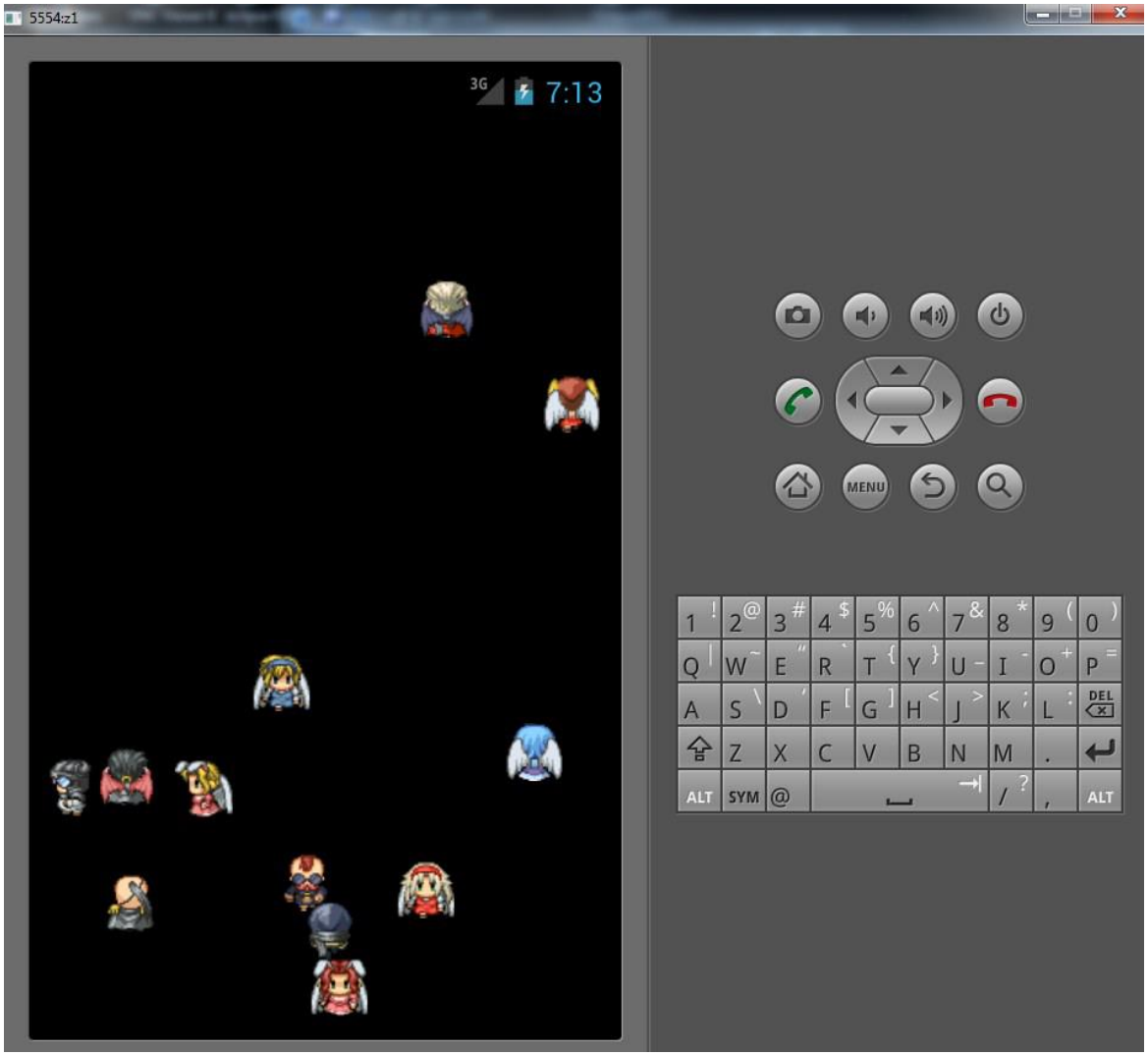
HRA – GAME LOOP THREAD

- FPS – počet snímků za sekundu
- Stav zda má běžet

Opakuje se:

- Zobraz a vyhodnoť kolize
- Čekej po zbytek intervalu

SCREENSHOT APLIKACE



Vyzkoušejte hru pokud možno v emulátoru i reálném zařízení

POUŽITÍ ASYNCHRONNÍHO TASKU

- potřeba dělat dlouhé činnosti na pozadí v novém vlákně
- interakce s UI po skončení práce na pozadí
- možnost informovat průběžně, že se něco děje

- projekt: 05-AsyncTask

ASYNCHRONNÍ TASK

- Spuštění:
  new LongOperation().execute("");
- private class LongOperation extends AsyncTask<String, Void, String>
- doInBackground()
  - může volat publishProgress()
  - běží v samostatném vláknu
- onPostExecute()
  - když práce na pozadí doběhne úspěšně
  - na UI vlákně
- onProgressUpdate()
  - Na UI vlákně, průběžně informovat uživatele

```
private class LongOperation extends AsyncTask<String, Void, String> {

    @Override
    protected String doInBackground(String... params) {
        for(int i=0;i<5;i++) {
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }

        return "Executed";
    }

    @Override
    protected void onPostExecute(String result) {
        TextView txt = (TextView) findViewById(R.id.textView1);
        // txt.setText("Executed");
        txt.setText(result);
        //might want to change "executed" for the returned string passed into onPostExecute() but that is upto you
    }

    @Override
    protected void onPreExecute() {
    }

    @Override
    protected void onProgressUpdate(Void... values) {
    }
} }
```

UKÁZKA PUBLIKOVÁNÍ PROGRESSU

Execute your task on main/UI thread:

```
new MyAsyncTask().execute("Test");
```

Asynctask:

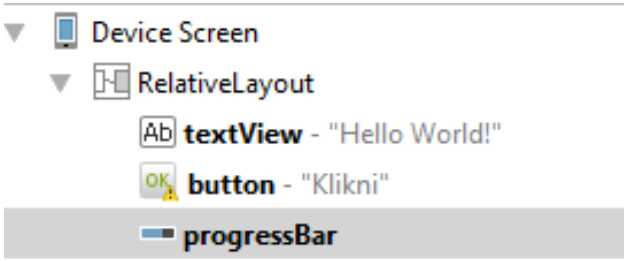
```
public class MyAsyncTask extends AsyncTask<String, Integer, String> {

    protected void onPreExecute() {
        Log.d("MyAsyncTask","MyAsyncTask Started");
    }

    protected String doInBackground(String... params) {
        Log.d("MyAsyncTask",params[0] + " in background.");
        publishProgress(params[0]);
        return null;
    }

    protected void onProgressUpdate(Integer... values) {
        Log.d("MyAsyncTask","onProgressUpdate - " + values[0]);
    }

    protected void onPostExecute(String result) {
        Log.d("MyAsyncTask","onPostExecute " + result);
    }
}
```

http://stackoverflow.com/questions/17246379/android-asynctask-onprogressupdate-and-onpostexecute-not-being-called

HANDLER, POST

- Další možnost jak pracovat v jiném vlákně a předávat informaci do UI

- Vytvoříme si UI
  - Tlačítko
  - ProgressBar Horizontal
    - max hodnota 100



HANDLER - TLAČÍTKO

```
Handler ha = new Handler() {
    public void handleMessage(Message msg) {
        ProgressBar pb1 = (ProgressBar)
findViewById(R.id.progressBar);
        pb1.incrementProgressBy(10);

    }

};


public void klikni_TlacitkoPB(View v) {
    new Thread (new Runnable()
    {
        public void run() {ha.sendMessage(ha.obtainMessage());}
    }).start();


}
```

POST

```
public void klikni_B1(View v) {
  new Thread (new Runnable()
  {
    public void run() {
      iv1.post(new Runnable() {

        public void run() {
          //iv1.setAlpha(50);
          iv1.setBackgroundColor(Color.RED);
          }  // konec run
      }

          ); // konec post

    }

  }
      ).start();

}  // konec klikni_B1
```

POST (KOPIROVATELNE)

```
public void klikni_B1(View v) {
    new Thread (new Runnable()
    {
        public void run() {
            iv1.post(new Runnable() {

                        public void run() {
                            //iv1.setAlpha(50);
                            iv1.setBackgroundColor(Color.RED);
                        }  // konec run
                    }

            ); // konec post

        }

    }
    ).start();

}  // konec klikni_B1
```

VIBRACE

- právo v manifestu:

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

- kód:

```
public void tlacitko_Vibruj (View v) {

        Vibrator vibrator = (Vibrator)
        getSystemService(Context.VIBRATOR_SERVICE);

        long []  vzor = {1000, 2000, 1000, 2000, 1000 };
        vibrator.vibrate(vzor, 0);
        vibrator.vibrate(1000);
}
```

VIBRACE - POZNÁMKA

- Testování na telefonu
  - S právem v manifestu – OK
  - Bez práva v manifestu – pád aplikace

PŘEHRÁNÍ HUDBY

- dáme mp3 do resourců:
  res/raw/pisen1.mp3

- kód:

```
MediaPlayer mp;
mp = MediaPlayer.create(this, R.raw.pisen1);
mp.start();
```

zastavení: mp.stop()

lze pustit jedno přehrávání vícekrát souběžně

ANIMACE

- ukázka rotace textu
- akcelerátor – animace zrychluje, zpomaluje

- projekt: 05-AnimaceTextu

UKÁZKA