# NTI/PAA - PROGRAMOVÁNÍ MOBILNÍCH APLIKACÍ

## 6. Asynchronní zpracování, vlákna, Handler, AsyncTask

Ing. Igor Kopetschke – TUL, NTI

http://www.nti.tul.cz

# Android – práce na pozadí

- Aplikace v Androidu musí odpovědět na akci do 5 (resp. 10) sekund od zahájení požadavku
- Pokud není splněno, OS nabídne ukončení aplikace
- Co s úlohami, které běží korektně, ale odpověď trvá mnohem déle ?
- Typicky :
  - dotazy do sítě, čekání na odpověď
  - stahování
  - instalace
  - zpracování rozsáhlých informací (grafika…)
  - a další .. a další …

# Android – práce na pozadí

- Každá aplikace obsahuje hlavní vlákno – UI Thread
- V tomto vlákně běží veškeré UI procesy – Activity
- Dále v něm běží Services a Broadcast Receivers
- Na tomto vlákně je aplikováno již zmíněné časové omezení

# Android – podpora vláken

- Android poskytuje standardní Java mechanismy pro vytváření, běh a správu vláken
  - extends Thread
  - implements Runnable
- Pro připomenutí je na elearningu prezentace Java + vlákna
- Lze taktéž využívat ThreadPool, Executor + novější třídy z balíčku java.util.concurrent
- Pro takto vytvořená vlákna ovšem platí již zmíněné omezení
- Nikdy, ale opravdu NIKDY nesmí jakkoli zasahovat do UI
- .. a to včetně Toast
- Takže jsou vhodné hlavně pro backgroundové operace

# Android – Handler

- Možnost asynchronního vykonání operace + promítnutí změn do UI
- Funguje na principu zasílání a příjmu zpráv
- V hlavním (UI) vlákně vytvořena instance třídy Handler
  - překrytí metody public void handleMessage(Message msg)
  - kód metody může přepisovat UI – běží v hlavním vlákně
  - přijímá zprávu zaslanou jiným vláknem
  - Zpráva je instancí třídy Message – viz dále
- V jiném (ne UI) vlákně proces zaslání zprávy
  - vytvořena / získána instance Message
  - předána metodě sendMessage() instance Handler

# Android – Handler + Message

- Zpráva zasílána instanci Handler je datového typu Message
- Lze ji získat:
  - Message msg = new Message()
  - Message msg = handler.obtainMessage()
- Obsah zprávy
  - vlastnosti arg1, arg2 – pouze int
  - getData() / setData(Bundle b) – předání instance Bundle

# Android – Handler + Message

```
1.  private Handler mHandler = new Handler() {
2.      //nasledujici kod se spusti v hlavnim (GUI) vlakne
3.      public void handleMessage(Message msg) {
4.              Bundle bundle = msg.getData();
5.              myTextView.setText(bundle.getString("text"));
6.      };
7.  };
```

```
1.  new Thread() {
2.      public void run() {
3.          ...
4.          Message msg = mHandler.obtainMessage();
5.          Bundle b = new Bundle();
6.          b.putString("text", "novy text");
7.          msg.setData(b);
8.          mHandler.sendMessage(msg);
9.      }
10. }.start();
```

# Android – AsyncTask

- Zjednodušuje volání metod v hlavním vlákně
- Kombinace metod, z nichž jedna běží ve vedlejším vlákně a průběžné výsledky se spouští v hlavním
- `AsyncTask<Params, Progress, Result>` je definovaný třemi generickými typy:
  - Params – typ parametru nutný pro vykonání operace
  - Progress – typ parametru použitý při „zobrazování“ pokroku
  - Result – typ výsledku

# Android – AsyncTask

- `AsyncTask<Params, Progress, Result>` má 4 základní metody:
  - `void onPreExecute()` – provede kód v hlavním vlákně těsně před tím, než se spustí vedlejší
  - `result doInBackground(Params... params)` – zde se spustí kód na pozadí
  - `void onPostExecute(Result result)` – spustí se v hlavním vlákně, jakmile se `doInBackground` ukončí
  - `void onProgressUpdate(Progress... values)` – spustí se když, se v `doInBackground` zavolá `publishProgress(progress);`
- Spustí se zavoláním `asyncTask.execute(params);`
  - Musí provést hlavní vlákno

# Android – zrušení AsyncTask

- Může být zrušen kdykoliv pomocí `cancel()`
- Metoda `isCancelled()` začne vracet `true`
- Místo `onPostExecute()` je zavolána metoda `onCancelled()`
- Toto zavolání proběhne až po skončení `doInBackground()`
- Pokud je to možné, kontrolovat (např. ve smyčce) v `doInBackground()`, jestli nebyl AsyncTask zrušen

# Android – AsyncTask příklad

```
1. myTask = new AsyncTask<String, Integer, Double>() {
2.     @Override
3.     protected void onPreExecute() {
4.           myTextView.setText("Loading...");
5.           super.onPreExecute();
6.     }
7.     @Override
8.     protected Double doInBackground(String... params) {
9.           Double result;
10.          int progress;
11.          ...
12.          publishProgress(progress);//progress se preda do onProgressUpdate
13.          ...
14.          return result;//preda se do onPostExecute
15.    }
16.    @Override
17.    protected void onProgressUpdate(Integer... values) {
18.          myTextView.setText("Running, progress= " + values[0]);
19.          super.onProgressUpdate(values);
20.    }
21.    @Override
22.    protected void onPostExecute(Double result) {
23.          myTextView.setText("Finished, result= " + result);
24.          super.onPostExecute(result);
25.    }
26.};
27.myTask.execute("param");//param se preda do doInBackground
```

# Použité a doporučené zdroje

- http://developer.android.com/
- http://www.zdrojak.cz/serialy/vyvijime-pro-android/
- http://www.itnetwork.cz/java/android
- https://users.fit.cvut.cz/cermaond/dokuwiki
- Google…

## .. A to je pro dnešek vše

# DĚKUJI ZA POZORNOST