Friday Lunch Talks

Session 5

Async Processing

Do it

The Java Way

The Java Way

The Java Way

Thread

The Java Way

Thread

ThreadPool

The Java Way

Thread

ThreadPool

FutureTask

The Java Way

The Problem

Only the Main-Thread is allowed to manipulate views!!!

The Java Way

The Solution

```
MainActivity.this.runOnUiThread(new Runnable() {
	public void run() {
		Log.d(“LOG", "I am the UI thread");
	}
});
```

The Java Way

The Problem

The second Thread has no lifecycle information!!!

The Java Way

The Solution

Be creative :P

Do it

The Android Way

AsyncTask

AsyncTask

AsyncTask

Provides callbacks in second thread for processing

AsyncTask

Provides callbacks in second thread for processing

Provides callbacks in UI thread for view updates

AsyncTask

AsyncTask

```
AsyncTask<X, Y, Z>
```

AsyncTask

```
AsyncTask<Params, Y, Z>
```

AsyncTask

```
AsyncTask<Params, Progress, Z>
```

AsyncTask

```
AsyncTask<Params, Progress, Result>
```

AsyncTask

```
AsyncTask<Params, Progress, Result>
```

Step 1:

onPreExecute()

AsyncTask

```
AsyncTask<Params, Progress, Result>
```

Step 1:

```
onPreExecute()
```

--> executed in the UI Thread

AsyncTask

```
AsyncTask<Params, Progress, Result>
```

Step 2:

```
doInBackground(Params…)
```

AsyncTask

```
AsyncTask<Params, Progress, Result>
```

Step 2:

```
doInBackground(Params…)
```

--> executed in new Thread

AsyncTask

```
AsyncTask<Params, Progress, Result>
```

Step 3:

```
onProgressUpdate(Progress…)
```

AsyncTask

```
AsyncTask<Params, Progress, Result>
```

Step 3:

```
onProgressUpdate(Progress…)
```

--> executed in the UI Thread

AsyncTask

```
AsyncTask<Params, Progress, Result>
```

Step 4:

```
onPostExecute(Result)
```

AsyncTask

```
AsyncTask<Params, Progress, Result>
```

Step 4:

```
onPostExecute(Result)
```

--> executed in the UI Thread

AsyncTask

Example:

AsyncTask

Example:

```
ImageDownloadTask
extends
AsyncTask<?, ?, ?>
```

AsyncTask

Example:

```
ImageDownloadTask

extends

AsyncTask<URL, Void, Bitmap>
```

AsyncTask

Example:

```
private ImageView imageView;

public ImageDownloadTask(ImageView imageView) {
    this.imageView = imageView;
}
```

AsyncTask

Example:

```
public Bitmap doInBackground(URL... urls) {
    URL imageUrl = urls[0];
    Bitmap image = null;
    try {
        image = BitmapFactory.decodeStream(
                url.openConnection().getInputStream())
    } catch (Exception e) {
        e.printStackTrace();
    }
    return image;
}
```

AsyncTask

Example:

```
public void onPostExecute(Bitmap image) {
    imageView.setImageBitmap(image);
}
```

AsyncTask

Example:

```
new ImageDownloadTask(imageView)
        .execute(imageUrl);
```

AsyncTask

Problem using AsyncTask

AsyncTask

Problem using AsyncTask

Cannot by canceled

AsyncTask

Problem using AsyncTask

Cannot by canceled

Has no Lifecycle Information

Service

Service

Service

Can be shared between Activities (even Apps)

Service

Can be shared between Activities (even Apps)

Can perform operations in a seperate Thread

Service

Can be shared between Activities (even Apps)

Can perform operations in a seperate Thread

Can perform operations even if the user has closed the application

Service

BoundService

vs.

IntentService

Service

BoundService

Service

BoundService

Custom Service Implementation

Service

BoundService

Custom Service Implementation

Communication via IBinder

Service

IntentService

Service

IntentService

Implementation available in Android

Service

IntentService

Implementation available in Android

Communication via Intents

Service

IntentService

Implementation available in Android

Communication via Intents

Operations run in sequence in own Thread

You bubbled enough!

Let‘s start!!!