Distributed Systems – Assignment 1
Aftermath

Matthias Kovatsch
kovatsch@inf.ethz.ch
Results

- Average grade of 5.6
- Many nice enhancements
- 6.0 means 6.0 – accordingly is the effort
- Assignments will count 20% of the grade
GENERAL HINTS

- Debugging
- UI and Feedback
- Starting the Service
- onCreate() and Persistence
- E-Mail
- Locking the phone (DeviceAdmin)
- Wake Lock
Debugging

- Many projects did not have `debuggable="true"`
- Use the `Log` functions `e()`, `w()`, `i()`, `d()` accordingly
- Filters can be also defined for «tag» and «pid»
UI and Feedback

- No course for UI design, however, ...
- Application should be usable

Now, wait that long:

- Nice design favors your work
  - Especially, if you present a demonstrator in a company
Starting the Service

// Service class
@Override
public IBinder onBind(Intent arg0) {
    // we do not need IPC, ignore
    return null;
}

@Override
public void onCreate() {
    super.onCreate();
    // initialize; is called once when created
}

@Override
public void onDestroy() {
    super.onDestroy();
    // clean up
}
Starting the Service

// Service class continued
@Override
public int onStartCommand(Intent i, int flags, int startId) {
    // is called with every startService(Intent)
    handleStart(i, startId);
    return START_STICKY; // explicitly started and stopped
}

private void handleStart(Intent i, int startId) {
    if (i==null) {
        // for START_STICKY
        // Service was restarted after memory shortage
        // handle that (e.g., restart work)
    } else {
        if (!running) { // static to check from Activity
            // start work (e.g., register to sensors)
        }
    }
}
Starting the Service

- **START_STICKY**
  - Service is re-scheduled after memory shortage
  - Must check for null Intents which are sent on restart
  - Explicitly start and stop Service

- **START_NOT_STICKY**
  - Service is not re-scheduled

- **REDELIVER_INTENT**
  - Service is re-scheduled and incomplete Intents will be re-delivered
  - Call `stopSelf(int startId)` when Intent has been processed
Starting the Service

- Foreground Service
  - Will not be killed on memory shortage
  - «Active» Service the user is aware of (e.g., media player)
  - Since 2.0 must provide ongoing notification

- Appropriate for AntiTheftService
- Call in Service class

```java
startForeground(NOTIFICATION_ID, notification);
```
onCreate() and Persistence

- Service state
  - Search the list

```java
ActivityManager mAM = (ActivityManager) getSystemService(ACTIVITY_SERVICE);
List<RunningServiceInfo> serviceInfos = activityManager.getRunningServices(Integer.MAX_VALUE);
for (RunningServiceInfo serviceInfo : serviceInfos) {
    if (serviceInfo.service.getClassName().equals(MY_SERVICE)) return true;
}
return false;
```

- Or use static flag in your Service

```java
public static boolean running = false; // -> handleStart() slide 7
```

- SharedPreferences

```java
SharedPreferences pref = getSharedPreferences(name, MODE_PRIVATE);
mailAddr = pref.getString(PREF_MAIL, ""); // string ID and default
...
SharedPreferences.Editor editor = pref.edit();
editor.putString(PREF_MAIL, mailAddr);
```
E-Mail

- Use a library for direct sending
  - javax.mail, etc.
  - GMailSender

- See
Locking the Phone (DeviceAdmin)

- See `DeviceAdminSample.java` of the `ApiDemos` sample

```java
mDPM = (DevicePolicyManager) getSystemService(Context.DEVICE_POLICY_SERVICE);
...
Intent i = new Intent(DevicePolicyManager.ACTION_ADD_DEVICE_ADMIN);
i.putExtra(DevicePolicyManager.EXTRA_DEVICE_ADMIN, mDeviceAdmin);
startActivityForResult(i, RESULT_ENABLE); // -> onActivityResult()
...
if (mDPM.isAdminActive(mDeviceAdmin)) {
    mDPM.lockNow();
}
```
Wake Lock

```java
wakeTimer.scheduleAtFixedRate(new TimerTask() {

    @Override
    public void run() {
        WakeLock lock = powerManager.newWakeLock(
            PowerManager.ACQUIRE_CAUSES_WAKEUP |
            PowerManager.SCREEN_DIM_WAKE_LOCK,
        );
        lock.acquire(ACQUISITION_TIME);
    }

}, 0, POWER_LOCK_INTERVAL);
```

- Phone is locked but still polls the accelerometer!