JÖNKÖPING UNIVERSITY

School of Engineering

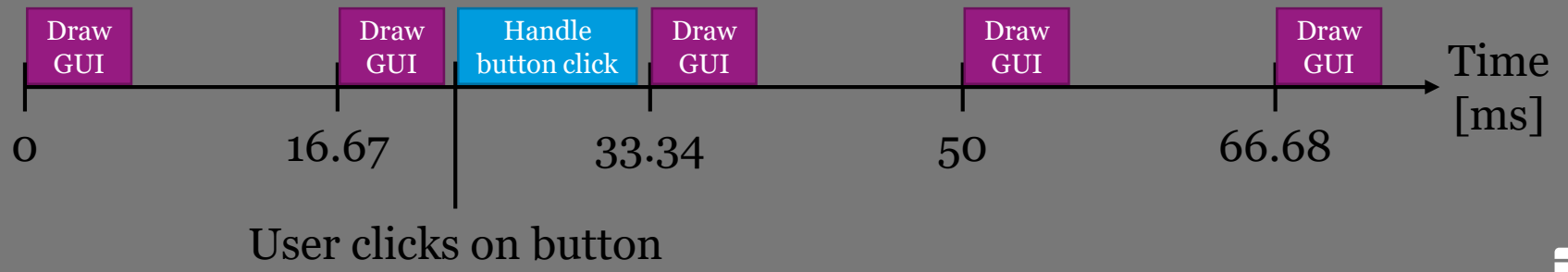
ANDROID ASYNC. OPERATIONS

Peter Larsson-Green
Jönköping University
Spring 2020

THE MAIN APPLICATION THREAD

By default, an Android application runs in a single thread.

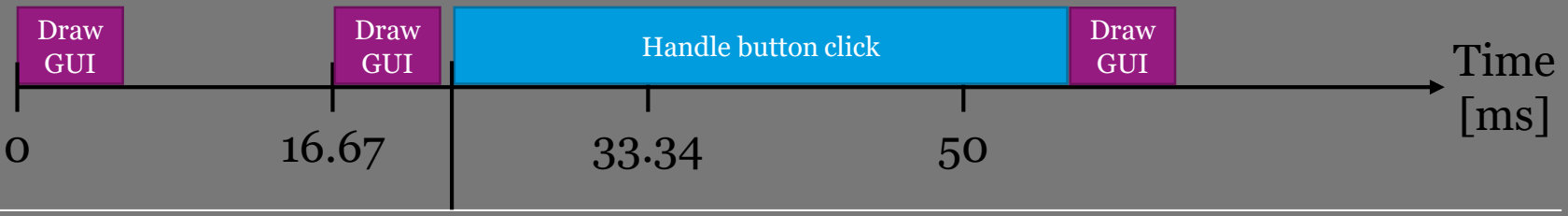
- Known as the main application thread.
  - Also known as the UI thread.
- Should re-draw the GUI 60 times a second.
  - Needs to re-draw each 1 sec/60 = 16,67 millisecond.
- Does by default everything else as well.
  - Such as handling clicks on buttons.



THE MAIN APPLICATION THREAD

By default, an Android application runs in a single thread.

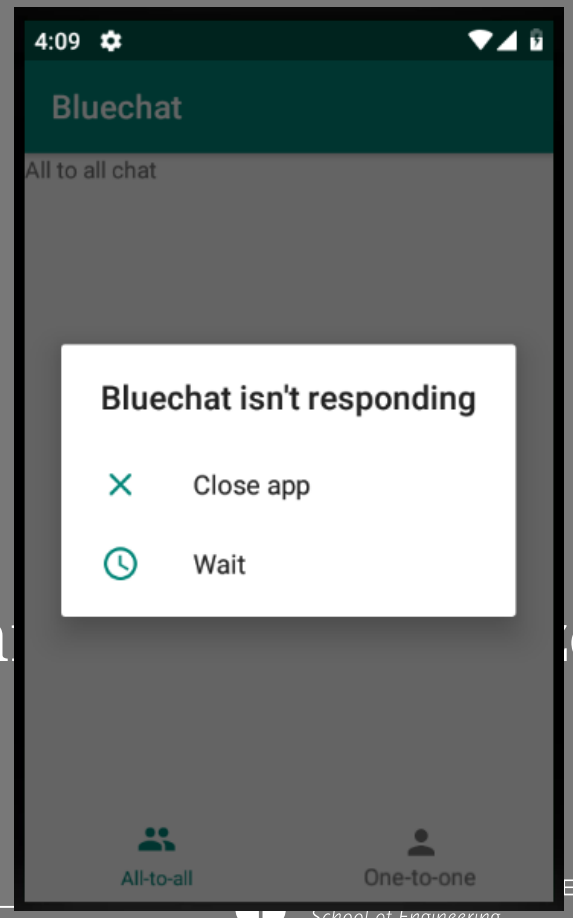
- Known as the main application thread.
  - Also known as the UI thread.
- Should re-draw the GUI 60 times a second.
  - Needs to re-draw each 1 sec/60 = 16,67 millisecond.
- Does by default everything else as well.
  - Such as handling clicks on buttons.
- Doesn't have time to re-draw the GUI → GUI unresponsive (freezes).



THE MAIN APPLICATION THREAD

By default, an Android application runs in a single thread.

- Known as the main application thread.
  - Also known as the UI thread.
- Should re-draw the GUI 60 times a second.
  - Needs to re-draw each 1 sec/60 = 16,67 millisecond.
- Does by default everything else as well.
  - Such as handling clicks on buttons.
- Doesn't have time to re-draw the GUI → GUI un es).
  - Unresponsive for a few seconds → Android Not Responding dialog:



HANDLING LONG RUNNING OPERATIONS

Anything that might take more than a few milliseconds to execute should not run on the main application thread, including:

- Network communication.
- Database communication.
- Reading/Writing from/to files.

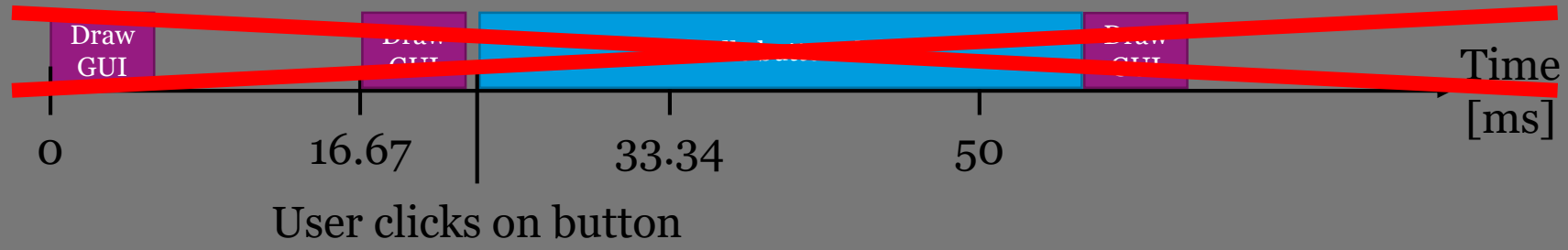
Instead:

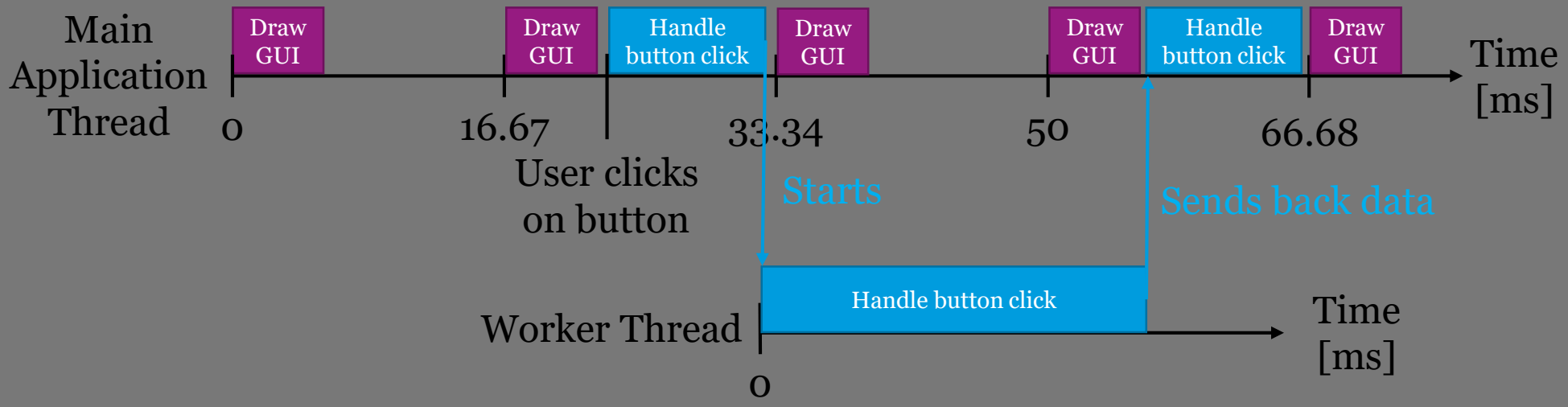
- Start new threads handling the long running operations.

Possible complication:

- Only the main application thread may change the GUI.
  - Thread communication needed.

HANDLING LONG RUNNING OPERATIONS





CREATING THREADS

```
Runnable runnable = new Runnable(){
  @Override
  public void run(){
    // Code to be executed in new the thread.
    int result = computeSomethingThatTakesLongTime();
    // How to send back result to the main app thread?
  }
};
Thread thread = new Thread(runnable);
thread.start();
```

THREAD COMMUNICATION

Handlers are used to communicate between threads.

```
final Handler handler = new Handler();
new Thread(new Runnable(){
  public void run(){
    final int result = compute();
    handler.post(new Runnable(){
      public void run(){
        // Runs on the thread that
        // created the handler.
      }
    });
  }
}).start();
```

THREAD COMMUNICATION

Android provides two simple way to execute a `Runnable` on the main application thread.

- Call `runOnUiThread` on an `Activity`:

```
anActivity.runOnUiThread(theRunnable);
```

- Call `post` on a `View`:

```
aView.post(theRunnable);
```

ASYNCTASK

Android's class `AsyncTask` makes thread handling even simpler.

```
AsyncTask t = new AsyncTask<Params, Progress, Result>(){
    protected void onPreExecute(){
    }
    protected Result doInBackground(Params[] params){
                                    publishProgress(new Progress())
    }
    protected void onProgressUpdate(Progress[] values){
    }
    protected void onPostExecute(Result result){
    }
};
t.execute(new Params(), new Params());
```

Called by main thread.