# Accelerating Information Technology Innovation

http://aiti.mit.edu

India Summer 2012

Lecture 9 – Multithreading on Android

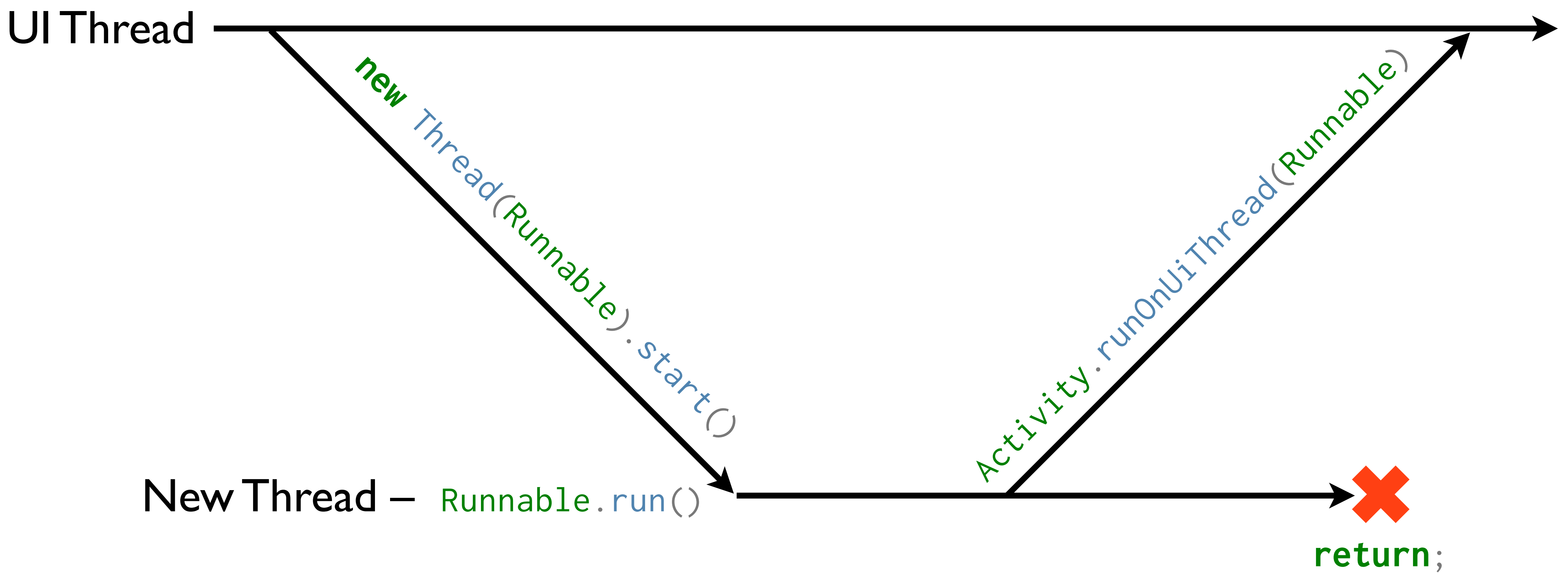
## What is multithreading?

- Like human multi-tasking
  - While waiting on input, your phone can do something else.
- Allows multiple actions to be happening simultaneously

## Why do multithreading?

- Make your app more responsive
  - Push heavy computation into a separate thread
  - Don’t halt your app with “waiting” actions
- Android 4.0 requires that you run networking code in a separate thread

# How multithreading works


UI Thread
new Thread(Runnable).start()
New Thread – Runnable.run()
Activity.runOnUiThread(Runnable)
return;

# Running a Thread

- `Thread` class creates a new thread.
- `Runnable` interface is used to run code in a separate thread.
    - `public void Runnable.run()` contains code to be run in the thread.
- `public void Thread.start()` starts the thread and calls `run()` in it.

## Running a Thread

```
public void onClick(View view) {
    Thread t = new Thread(new Runnable() {
        public void run() {
            System.out.println("I am in another thread!");
        }
    });
    t.start();
}
```

# Multithreading Pitfalls

- Network code *MUST* be in a separate thread, but...
- UI (Buttons, TextViews, EditTexts, etc.) *CAN'T* be accessed outside the UI thread!
  - Can run UI code with:
    - `Activity.runOnUiThread(Runnable);`
    - `View.post(Runnable);`

# Sample Networking Code

```
public void onClick(View view) {
    String url = "http://www.example.com/";
    new Thread(new Runnable() {
        public void run() {
            // downloadStates(url) downloads state data.
            ArrayList<String> states = downloadStates(url);
            MyActivity.this.runOnUiThread(new Runnable() {
                public void run() {
                    // populateList() populates the ListView
                    populateList(states);
                }
            });
        }
    });
}
```

# Android Multithreading

- This is very tricky!
- Android provides a convenience class: `AsyncTask<Params, Progress, Result>`
- Subclass `AsyncTask` to do *asynchronous* tasks like network code.

# Android Multithreading

- Create a subclass of `AsyncTask`
  - `Params` – the class of the task arguments
  - `Progress` – the class of the progress arguments (can be `void`)
  - `Result` – the class of the return value

# Android Multithreading

- Create a subclass of `AsyncTask`
  - `Result doInBackground(Params... params)`
    The code to run in the background (e.g. networking code)
  - `void doPostExecute(Result result)`
    The code to run on the UI thread when done (e.g. changing the `ListView`)

## Sample Networking Code

```
public void onClick(View view) {
    new StateDownloader().execute("http://www.example.com/");
}

private class StateDownloader
    extends AsyncTask<String, void, ArrayList<String>> {
    public ArrayList<String> doInBackground(String... urls) {
        return downloadStates(urls[0]);
    }

    public void doPostExecute(ArrayList<String> states) {
        populateList(states);
    }
}
```

## Multithreading Exercises

## Multithreading Exercises

Create a new Android project and add a button that says “Get my IP!” below the TextView

## Multithreading Exercises

Create a class named `IPFetcher` that is a subclass of `AsyncTask` that takes a `URL` argument and returns a `String` when the background task is done.

## Multithreading Exercises

Make `IPFetcher` connect to the `URL` and return the contents of the `URL` in the background.

# Multithreading Exercises

Make `IPFetcher` set the text of the `TextView` to the response from the `URL` once it returns.

## Multithreading Exercises

Make the button cause `IPFetcher` to execute when it is clicked.

# References

- “Processes and Threads” on the Android Developer Site: <http://developer.android.com/guide/components/processes-and-threads.html>
- “Concurrency” Java Tutorial: <http://docs.oracle.com/javase/tutorial/essential/concurrency/>
- “Writing Multithreaded Applications” on IBM developerWorks: <http://www.ibm.com/developerworks/java/library/j-thread/index.html>