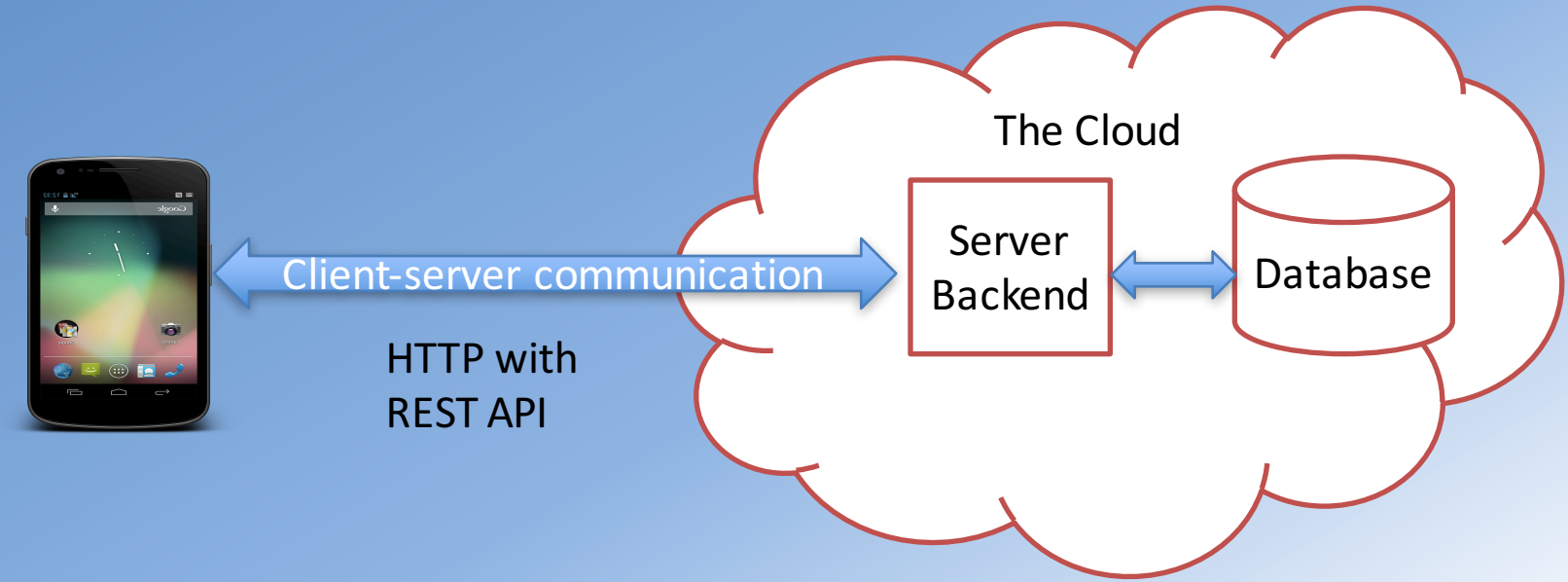
Communicating with a Server

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

香港科技大學
THE HONG KONG UNIVERSITY OF
SCIENCE AND TECHNOLOGY

Client and Server

- Most mobile applications are no longer stand-alone
- Many of them now have a “Cloud” backend



The Networking Alphabet Soup

Android Networking

- Network operations cause unexpected delays
- Always do network operations in the background
  - Use AsyncTask() or background thread
  - Android will cause exception if you do networking operations on the main UI thread

Off to the next exercise

- Processing a JSON string
- Mapping JSON string to Java Objects
- Using Gson library

Javascript Object Notation (JSON)

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

香港科技大學
THE HONG KONG UNIVERSITY OF
SCIENCE AND TECHNOLOGY

HTTP Response

- Server may send back information in a specific format:
  - eXtensible Markup Language (XML)
  - Javascript Object Notation (JSON)
- Android also includes three parsers for XML and a parser for JSON
  - the traditional W3C DOM parser (org.w3c.dom) – converts document to a tree of nodes
  - a SAX parser (org.xml.sax) – streaming with application callbacks
  - the XML pull parser – iterates over XML entries
  - JSON parser (org.json)
- Also consider the use of third-party libraries to deal with specific formats like RSS/Atom parser
- Gson libraries to convert from Java objects to JSON and back

Javascript Object Notation (JSON)

- http://www.json.org
- Lightweight data interchange format
- Language independent *
- *Self-describing* and easy to understand
- Data structured as:
  - A collection of name/value pairs
  - Ordered list of values
  - Example

```
{ "people": [
   { "id": 1, "name": "John”,   "statusMsg": "Imagine all the people ...”,   "imageURL": "John.png”    },
   { "id": 2, "name": "Paul”,   "statusMsg": "Let it be ...”,                 "imageURL": "Paul.png”    },
   { "id": 3, "name": "George”, "statusMsg": "Wait mister postman ...”,      "imageURL": "George.png"  },
   { "id": 4, "name": "Ringo”,  "statusMsg": "Yellow submarine ...”,         "imageURL": "Ringo.png”   }
]}
```

Off to the next exercise

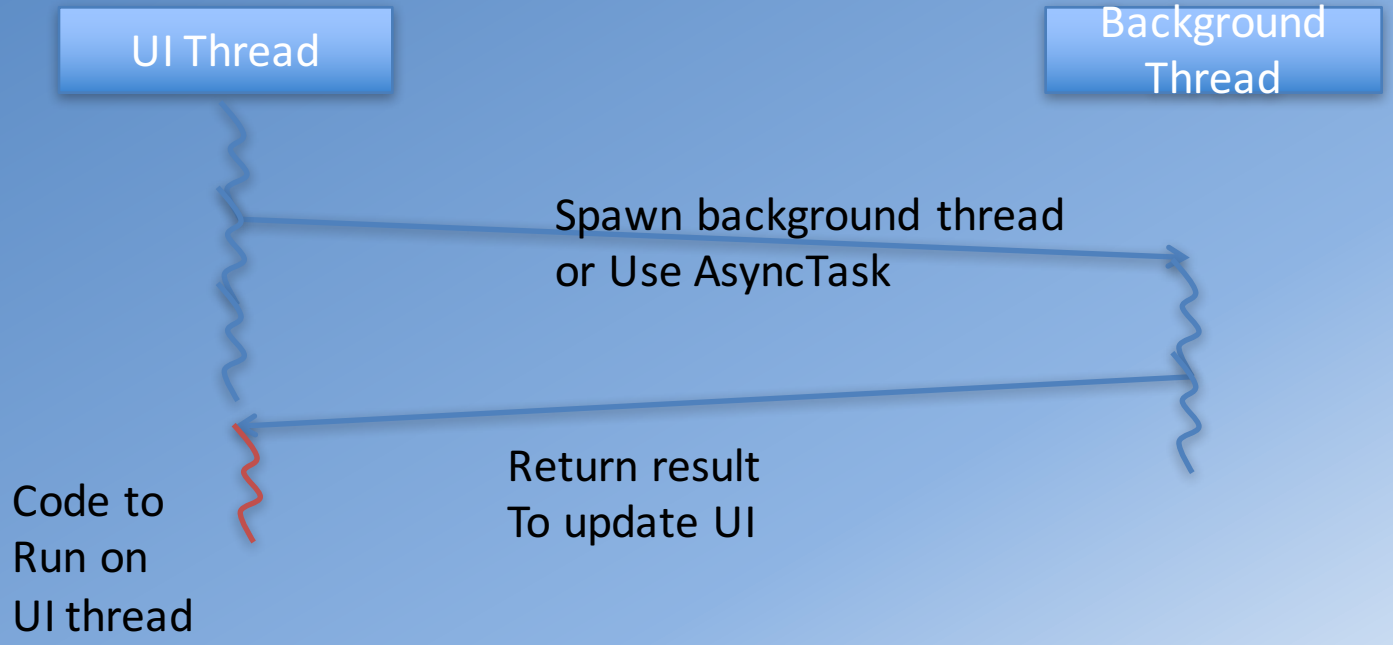
- Using an AsyncTask to offload work from the main UI thread
- Using Picasso image downloading library

AsyncTask: Doing Work in the Background

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Threads / AsyncTask



AsyncTask

- Threads provides a powerful framework
  - Code gets complicated and difficult to read
- AsyncTask: simplifies the creation of long-running tasks that need to communicate with the UI
  - Takes care of thread management
  - Has to be created on the UI thread

AsyncTask

- AsyncTask has the following excellent features:
  - Ability to **return values** of custom type to the UI thread when the task is finished
  - Ability to execute some code **in the UI thread** before the background task begins execution and after it is finished
  - Ability to **push updates** to the UI thread during the execution of the background task
  - **Automatic** under-the-hood thread management

AsyncTask Example

```
private class WorkerTask extends AsyncTask<Object, String, Boolean> {

    // Initialize the progress bar and the status TextView
    @Override
    protected void onPreExecute() {
        completed = 0;
        // This will result in a call to onProgressUpdate()
        publishProgress();
    }

    @Override
    // This method updates the main UI, refreshing the progress bar and TextView.
    protected void onProgressUpdate(String... values) {

    }

    // Do the main computation in the background and update the UI using publishProgress()
    @Override
    protected Boolean doInBackground(Object... params) {

        return null;
    }
}
```

AsyncTask

- Several methods are part of AsyncTask:
  - doInBackground() executes automatically on a worker thread
  - onPreExecute(), onPostExecute(), and onProgressUpdate() are all invoked on the UI thread
  - Call publishProgress() at any time in doInBackground() to execute onProgressUpdate() on the UI thread
  - The value returned by doInBackground() is sent to onPostExecute()
  - You can cancel the task at any time, from any thread

Off to the next exercise

- Connecting to a server using HTTPURLconnection
- Getting JSON string from the server

Android Networking

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

香港科技大學
THE HONG KONG UNIVERSITY OF
SCIENCE AND TECHNOLOGY

Android Permissions

- Set permission in the Manifest file:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
```

Android Connectivity Manager

- Android provides the ConnectivityManager class to monitor network connectivity state, setting preferred network connections and managing connectivity failover
- Get access to the ConnectivityManager by:

```
ConnectivityManager myNetMan = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
```

Android Connectivity Manager

- Connectivity Manager provides methods like getNetworkInfo(), getActiveNetworkInfo(), and getAllNetworkInfo() etc.
  - These methods return the NetworkInfo object
  - Can use methods within this object like isAvailable(), isConnected, isConnectedorConnecting(), getState(), etc.

Android Connectivity Manager

- If you are using the network access in your application, it is always a good idea to check if the network connectivity exists, and take action accordingly
- Example

```
public boolean isOnline() {
    ConnectivityManager connMgr = (ConnectivityManager)
      getSystemService(Context.CONNECTIVITY_SERVICE);

    NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();

    if (networkInfo != null && networkInfo.isConnected()) {
       return true;
    } else {
       return false;
    }
}
```

Android HTTP Support

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

香港科技大學
THE HONG KONG UNIVERSITY OF
SCIENCE AND TECHNOLOGY

Android and HTTP

- Android provides HTTPURLConnection client to send and receive data over the web
- Android also has Apache HTTP components library built into the framework
  - HttpClient component enables handling of HTTP requests on your behalf, issuing HTTP requests and dealing with the response
  - Not the preferred choice any more
- You can layer a SOAP/XML-RPC layer atop this library or use it "straight" for accessing REST-style web services

Android HTTPURLConnection Example

```
final String site= "http://<your IP address>:3000/people";

// get the URL connection
URL url = new URL(site);
HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();

// set up the URL request parameters
urlConnection.setReadTimeout(10000 /* milliseconds */);
urlConnection.setConnectTimeout(15000 /* milliseconds */);
urlConnection.setDoInput(true);

/* optional request header */
urlConnection.setRequestProperty("Content-Type", "application/json");
urlConnection.setRequestProperty("Accept", "application/json");

/* for Get request */
urlConnection.setRequestMethod("GET");
```

Android HTTPURLConnection Example

```
// Starts the query
int statusCode = urlConnection.getResponseCode();
Log.d(DEBUG_TAG, "The response is: " + statusCode);

// get the input stream for the response body of the URL response
InputStream inputStream = new
        BufferedInputStream(urlConnection.getInputStream());

// parse the response
String response = convertInputStreamToString(inputStream);
parseResult(response);
```

Assignment 5: Asynchronous Download of Images

- Use Picasso library to download the images asynchronously and update the ListView avatars of the users