Internet II: Communicating with APIs

CS 2046

Mobile Application Development

Fall 2010

Announcements

- HW2 due **tonight**, 11:59pm!
  - Office hours after class
  - Code for TaskFrontend released
- HW3 released shortly, probably due Friday, 11/19
  - Will hold office hours the week after class ends.
- Last lecture – Friday, 11/12.

Recap

- Displaying web pages using WebView
  - Browser settings – WebSettings
  - UI settings – WebChromeSettings
  - Page rendering settings – WebClientSettings
- Using URLConnection for reading/writing to the internet.
  - Today: Why this is often the wrong choice.

Problems with URLConnection

- As discovered while working on sample for today's lecture:

- Mentioned you could call getOutputStream() to upload data.
  - True, but you must call setDoOutput(true) first.

- In addition, at least on the emulator, URLConnection is very slow.

Alternate - HttpClient

- More robust HTTP API
  - Part of Apache Project – Apache runs ~60% of web servers (Netcraft)

- To obtain input stream:

```
String url = “http://www.google.com”;
HttpClient httpClient = new DefaultHttpClient();
HttpGet httpGet = new HttpGet(url);
InputStream is =
    httpClient.execute(httpGet).getEntity().getContent();
```

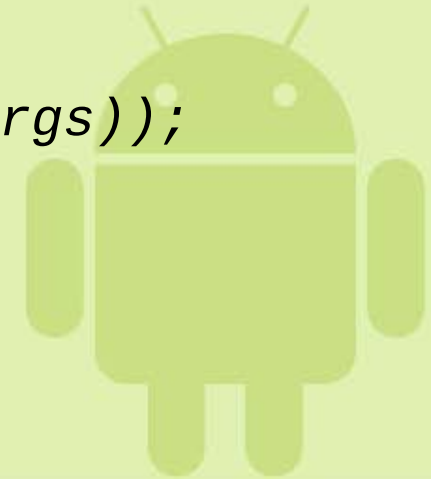
- Can reuse httpClient for multiple requests.

Upload with HttpClient

- Writing data is also simple:

```
List<BasicNameValuePair> args =
   new LinkedList<BasicNameValuePair>();
args.add(new BasicNameValuePair("val1", "data1"));
args.add(new BasicNameValuePair("val2", "data2"));

HttpClient httpClient = new DefaultHttpClient();
HttpPost httpPost = new HttpPost(url);
httpPost.setEntity(new UrlEncodedFormEntity(args));
httpClient.execute(httpPost);
```



Escaping URLs

- When communicating with an API, need to make web requests with special data.
- Data may have special characters (e.g. spaces, other URLs).
- Make sure to “escape” data:
  - Converts symbols to %NUM (space = %20)
  - Uri.escape(String) returns an escaped String.

JSON

- "JavaScript Object Notation"
  - Really, language-independent
  - Simple, human-readable (but lightweight) data interchange format.
    - Recall – need to transfer small amounts of data.

- Many APIs communicate with JSON.
  - Android provides parsing tools in org.json package.

Facebook Graph API

- Overview: http://developers.facebook.com/docs/api
- Represents the social graph:
  - Nodes – people, photos, events, pages
  - Connections – friends, shared content, photo tags
- Allows reading and writing of content.
  - Post on walls, upload photos
- Facebook does have an Android API
  - This lecture's approach is more general.

Example JSON Object

- From Facebook Graph API:

```
{
   "id": "143981188983163",
   "owner": {
      "name": "Jeff Davidson",
      "id": “1234567"
   },
   "name": "CS 2046 Example Event",
   "description": "Here's some event info!",
   "start_time": "2010-11-08T22:00:00+0000",
   "end_time": "2010-11-08T22:30:00+0000",
   "location": "Cornell",
   "privacy": "SECRET",
   "updated_time": "2010-11-08T10:35:00+0000"
}
```

Accessing JSON

- If we have an InputStream from our HTTP connection:

```
BufferedReader reader = new BufferedReader(
    new InputStreamReader(is), 512);
StringBuilder sb = new StringBuilder();
char[] buffer = new char[512];
int read;
while ((read = reader.read(buffer)) != -1) {
    sb.append(buffer, 0, readBytes);
}
JSONObject result = (JSONObject)
    new JSONTokener(sb.toString()).nextValue();
```

Accessing Example

```
{
   "id": "143981188983163",
   "owner": {
      "name": "Jeff Davidson",
      "id": “1234567"
   },
   "name": "CS 2046 Example Event",
   "description": "Here's some event info!",
   "start_time": "2010-11-08T22:00:00+0000",
   "end_time": "2010-11-08T22:30:00+0000",
   "location": "Cornell",
   "privacy": "SECRET",
   "updated_time": "2010-11-08T10:35:00+0000"
}
```

result.getString(“name”)
or
result.optString(“name”)

Accessing Example

```
{
   "id": "143981188983163",
   "owner": {
      "name": "Jeff Davidson",
      "id": “1234567"
   },
   "name": "CS 2046 Example Event",
   "description": "Here's some event info!",
   "start_time": "2010-11-08T22:00:00+0000",
   "end_time": "2010-11-08T22:30:00+0000",
   "location": "Cornell",
   "privacy": "SECRET",
   "updated_time": "2010-11-08T10:35:00+0000"
}
```

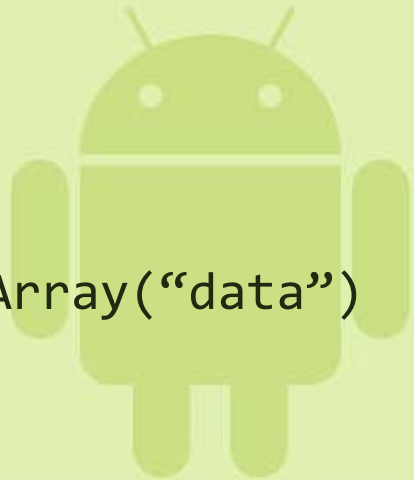
result.getJSONObject(“owner”)
or
result.optJSONObject(“owner”)

Another JSON Example - Arrays

```
{
   "data": [
      {
         "name": "Scott Rogoff",
         "id": “7654321",
         "rsvp_status": "attending"
      },
      {
         "name": "Jeff Davidson",
         "id": “1234567",
         "rsvp_status": "attending"
      }
   ]
}
```

Another JSON Example - Arrays

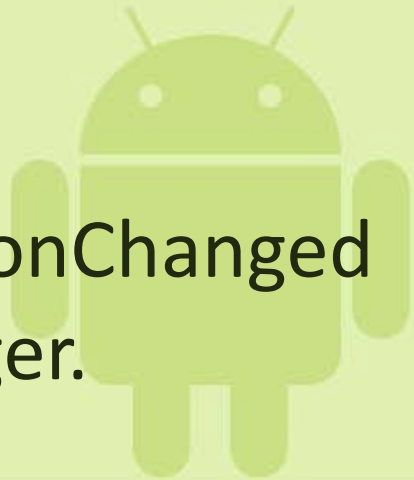
```
{
   "data": [
      {
         "name": "Scott Rogoff",
         "id": “7654321",
         "rsvp_status": "attending"
      },
      {
         "name": "Jeff Davidson",
         "id": “1234567",
         "rsvp_status": "attending"
      }
   ]
}
```



- Use `result.getJSONArray(“data”)` or `result.optJSONArray(“data”)` to obtain a JSONArray object.

Location Data

- Request location with LocationManager
- Unlike other managers we've used so far, Location is asynchronous
  - Location comes from GPS, Wifi, Cell Towers
  - Hardware may take time and cannot return result immediately.
- Define a LocationListener with onLocationChanged method and pass this to LocationManager.

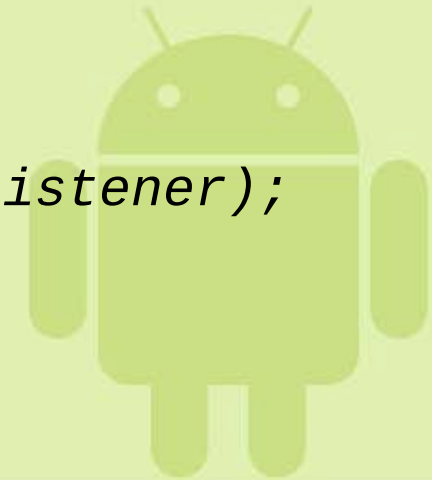


Geocoder

- Location is in Latitude/Longitude
  - What can we do with that?
- Geocoder class converts to a more reasonable location
  - Street address, city name, zip code
  - Uses the Maps API
- Unfortunately, not functioning in Android 2.2
  - But Android 2.1 emulator works

LocationManager Example

```
LocationManager mgr =
    (LocationManager)
    getSystemService(Context.LOCATION_SERVICE);
updateLocation(manager.getLastKnownLocation(
    LocationManager.NETWORK_PROVIDER));
LocationListener listener = new LocationListener() {
    public void onLocationChanged(Location loc) {
        updateLocation(loc);
    }
};
mgr.requestLocationUpdates(
    LocationManager.NETWORK_PROVIDER, 0, 0, listener);
```



updateLocation

```
Geocoder gcd = new Geocoder(context,
    Locale.getDefault());
List<Address> addresses = null;

try {
    addresses = gcd.getFromLocation(
        loc.getLatitude(), loc.getLongitude(), 1);
} catch (IOException e) {
    e.printStackTrace();
}

if (addresses != null && addresses.size() > 0) {
    mLocation = addresses.get(0).getLocality();
}
```

Using Location Services

- Need permission:
  - android.permission.ACCESS_*_LOCATION
    - COARSE: cell tower, wifi
    - FINE: gps
    - FINE implies COARSE
- If using Geocoder, need maps library
  - Build Library/Emulator: Google APIs
  - <uses-library android:name="*com.google.android.maps*" />
    - Careful – goes under <application>, not <manifest>, unlike other <uses-> tags we've seen.
- Can set location on emulator in Eclipse: Window → Show View → Android → Emulator Control

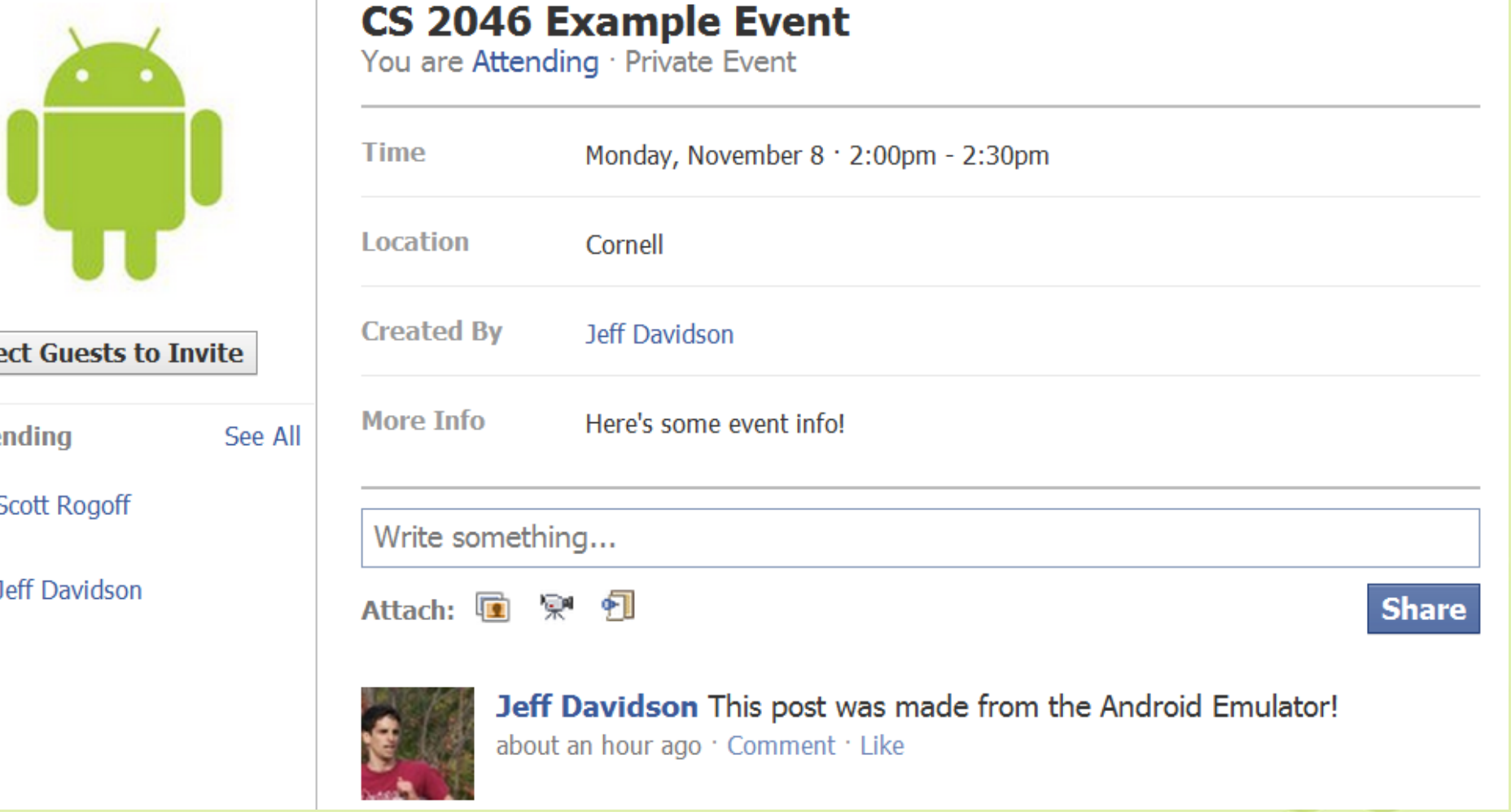
Involved Example: FacebookEvent

- Suppose you're holding an event, and you want to create an app for people attending.
  - View basic info about event
  - Post your thoughts or suggestions
  - View other people attending
  - See pictures or upload your own pictures

- Facebook's Events let you do all of this already.
  - Why duplicate all the work?

Facebook Event

http://www.facebook.com/event.php?eid=**143981188983163**



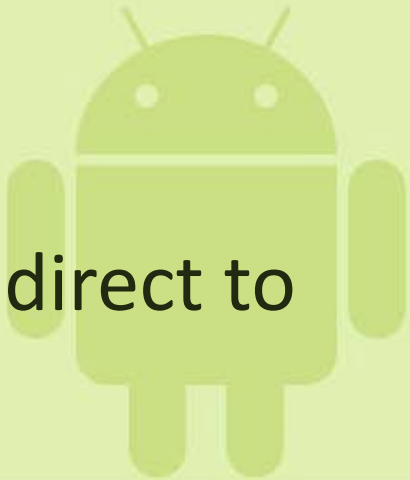
FacebookEvent Architecture

- Activities
  - AuthActivity – Handles authentication with Facebook
  - FacebookTabActivity – Shell to display tabs.
    - FacebookEventActivity – event info
    - AttendingActivity – who is attending

- AsyncTasks
  - APIFetcherTask – calls API, returns JSONObject
  - APIPusherTask – pushes data to API
  - ImgDownloadTask – downloads an image

Facebook API Authentication

- See http://developers.facebook.com/docs/authentication/
- Request authentication by loading page at:
  https://graph.facebook.com/oauth/authorize?
  client_id=...
  &redirect_uri=http://www.example.com/oauth_redirect
- client_id (and client_secret, later) assigned when you register app with facebook.
- redirect_uri is a URI your browser will redirect to once authentication is successful



Facebook API Authentication

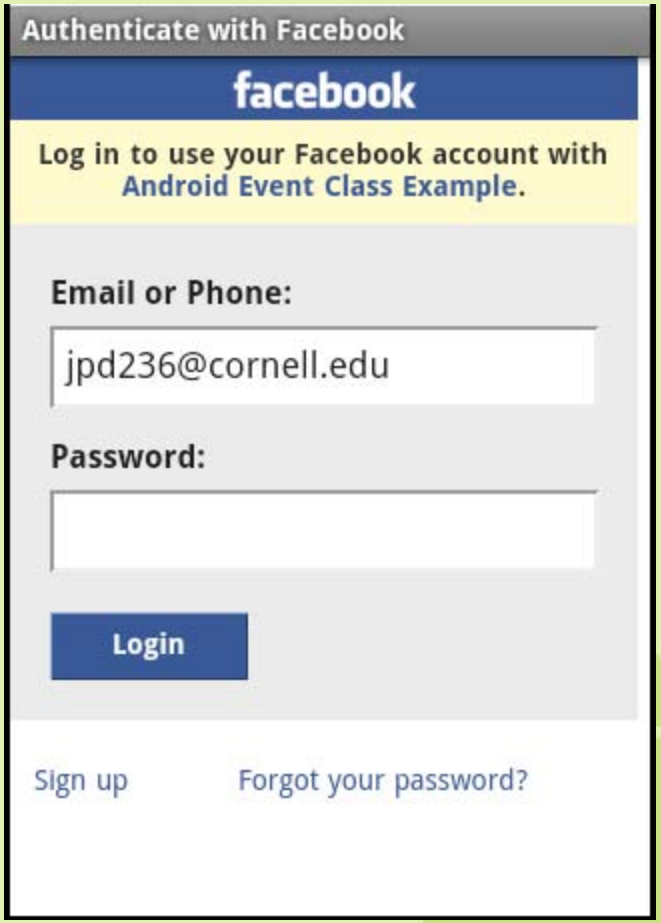
- If authorized (and redirected), get argument "code" added to redirect_uri.

- Request access_token from:

https://graph.facebook.com/oauth/access_token?
client_id=...
&redirect_uri=http://www.example.com/oauth_redirect
&client_secret=...
&code=...

AuthActivity

- Called as a sub Activity when authorization is needed.
- Displays a WebView to allow user to log into Facebook and authenticate application.
- Once authorized, uses AsyncTask and HttpClient to fetch the access token to be used by other Activities.
  - Stored in SharedPreferences



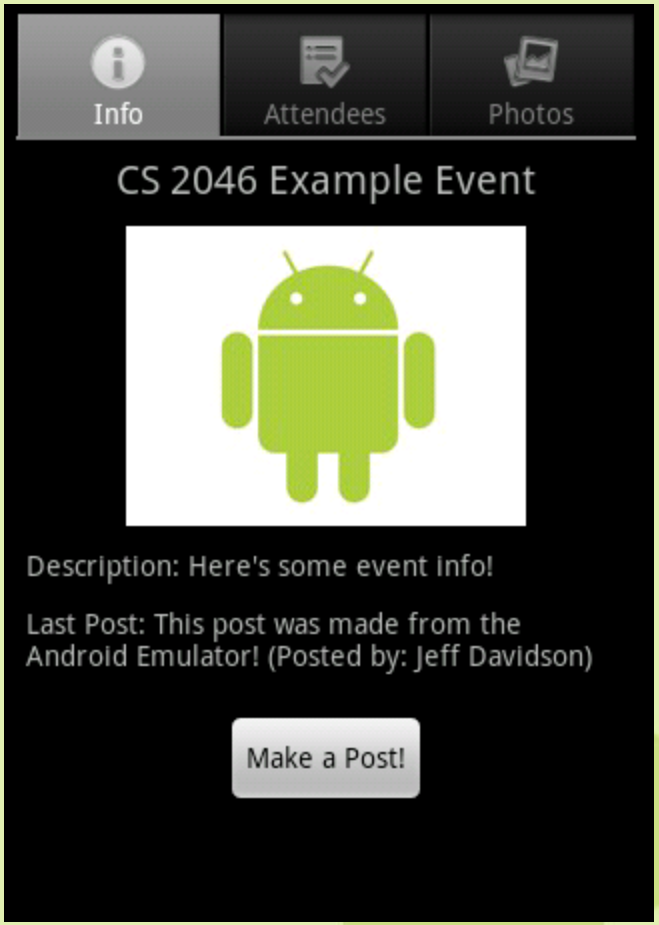
APIFetcherTask

- AsyncTask which handles calling the Facebook API to obtain a JSON object
  - Uses HttpClient

- Supports a callback interface to allow caller to access the JSON object once the download is complete.

FacebookEventActivity

- Displays Event info
- Uses existing access token, or requests new one if this fails or none exists.
- Calls APIFetcherTask and ImgDownloadTask to fetch data in background.
- Calls APIPusherTask to make posts to event feed.



AttendingActivity

- Similar access pattern to FacebookEventActivity.
- Parses JSON object into a List<Map<String, String>> containing the data.
- Uses SimpleAdapter to bind data to ListView.

AttendingActivity List Data

```
List<Map<String, String>> listData = new ArrayList<Map<String, String>>();
JSONArray attendees = obj.optJSONArray("data");
if (attendees != null) {
    Map<String, String> attendee;
    for (int i = 0; i < attendees.length(); i++) {
        attendee = new HashMap<String, String>();
        try {
            attendee.put("name", attendees.getJSONObject(i).optString("name"));
        } catch (JSONException e) {
            // Should never occur since array indices are bounded.
        }
        listData.add(attendee);
    }
}

SimpleAdapter adapter = new SimpleAdapter(this, listData,
    android.R.layout.simple_list_item_1,
    new String[] { "name" }, new int[] { android.R.id.text1 });

lv.setAdapter(adapter);
```

Next Class

- Multimedia!
  - Camera – extend FacebookEvent to take and upload pictures to the event.
  - 3D graphics, audio/video