Understanding Heart and Body Status from a Smartphone

– Internet Programming on Android–

Guillaume Lopez
Living Environment Laboratory
The University of Tokyo, School of Engineering

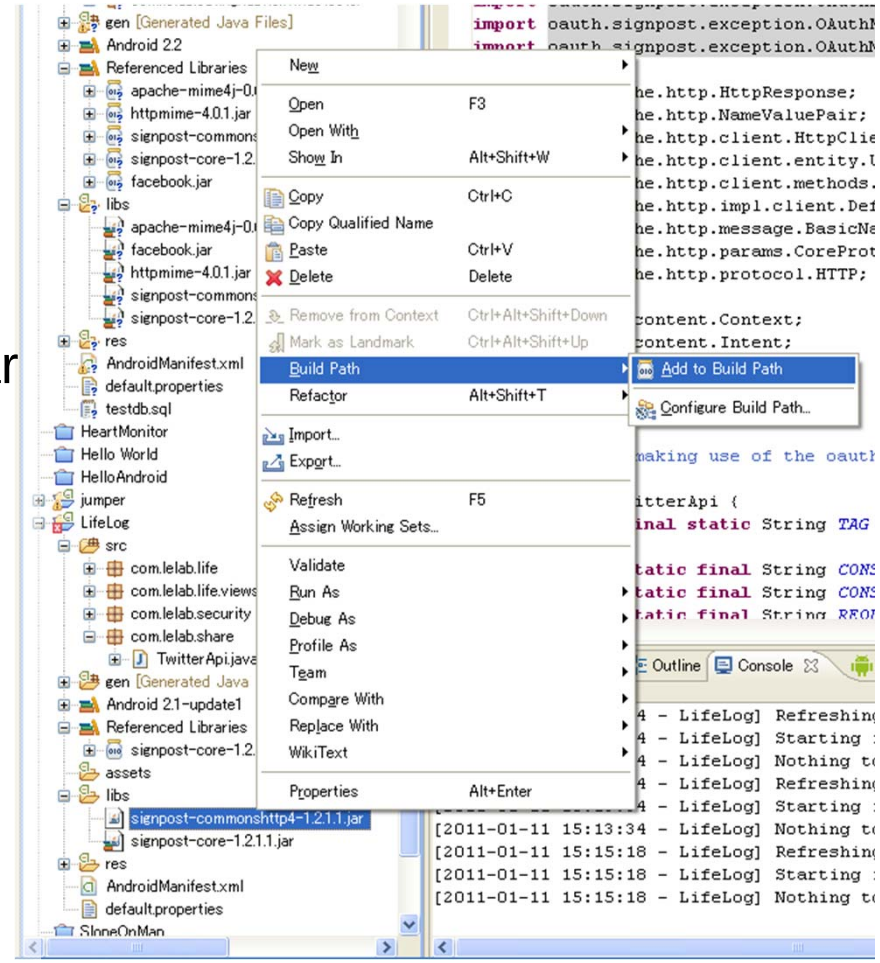
Internet Programming

- Access Web Pages
- Share Information on SNS
  - **Twitter,**
  - Facebook,
  - Mixi

Wireless Programming on Android

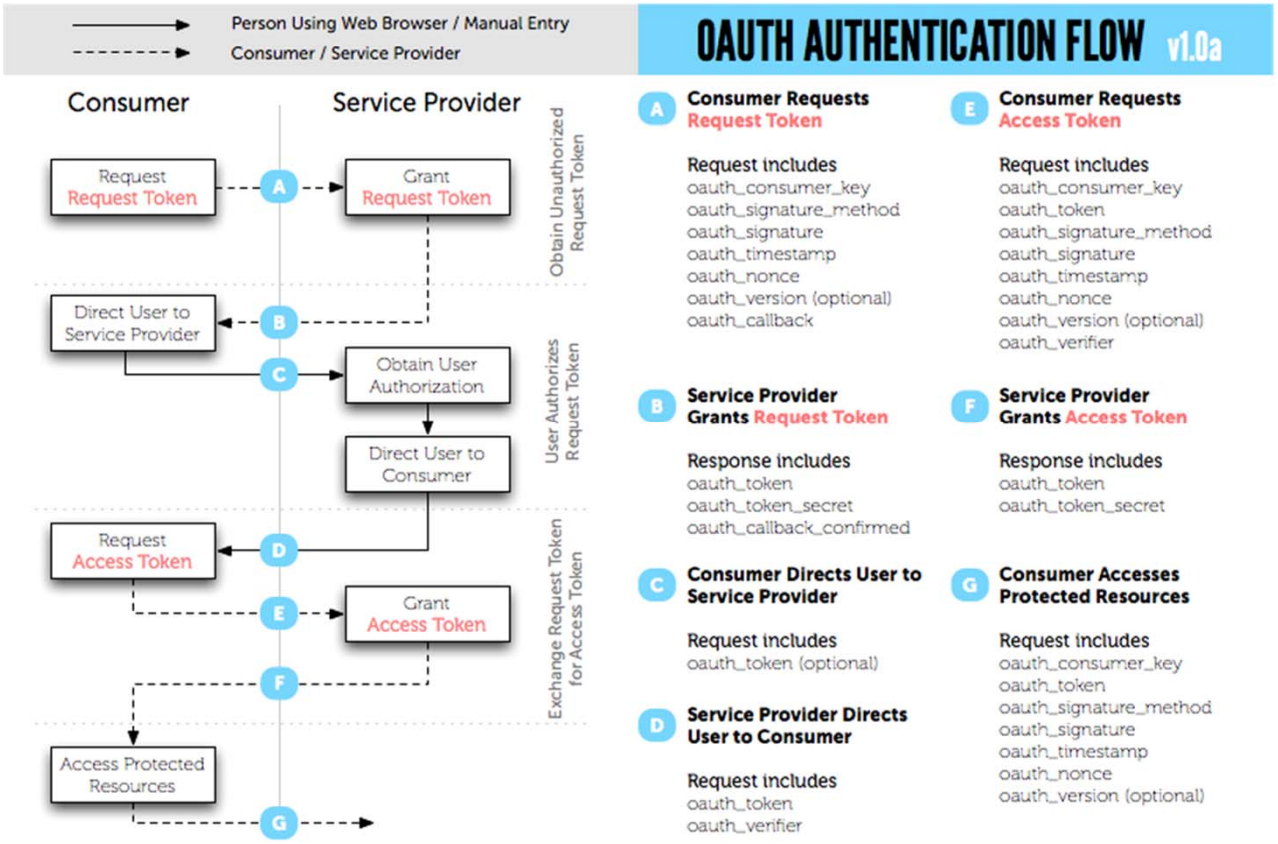
Fundamental Twitting Capabilities

- Connection
  - Authentication
  - Post
  - Read
  - Status
- Need some libraries <libs>
  - signpost-commonshttp4-1.2.1.1.jar
  - signpost-core-1.2.1.1.jar



Oauth Authentication Flow

- The Oauth request cycle is roughly
  1. Retrieve a request token
  2. Request user authorization by sending the user to a login page
  3. Exchange the request token for an access token



OAuth at Twitter

- Twitter uses the open authentication standard OAuth for authentication.
  - HTTP Header-based OAuth

    it separates concerns, makes debugging easier, and avoids common issues with under or over URL escaping parameters.
  - SSL for /oauth/* end points

    request_token, access_token, and authorize, use SSL
  - api.twitter.com

    use api.twitter.com as the hostname, not just "twitter.com"
  - An explicit oauth_callback

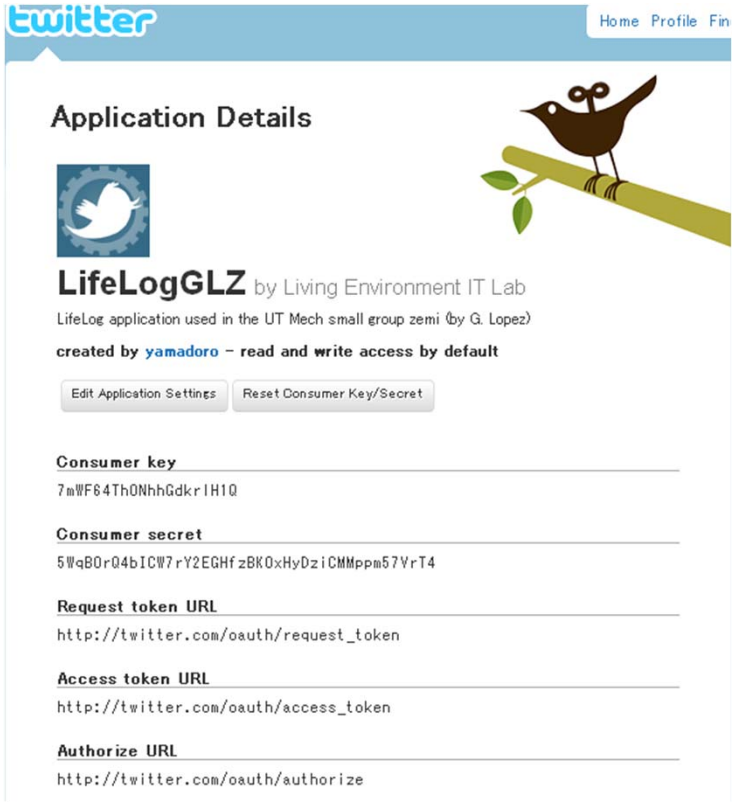
    By dynamically setting your oauth_callback, you can pass additional state information back to your application and control the experience best. If using the PIN code flow, specify your oauth_callback as "oob".

Before starting register yourself

- The first thing you have to do is register a client application. Each client application you register will be provisioned
  - A consumer **key**
  - A consumer **secret**.

  (key and secret scheme is similar to the public and private keys used in SSH)

Used, to *sign* every request you make to the API, to ensure it is you.



In the code

TwitterAPI.java

```
public static void register(Context context){
    /**************************************************
     * The following steps should only be performed ONCE
     **************************************************/
    // we do not support callbacks, thus pass OOB
    String authUrl;
    try {
          authUrl = provider.retrieveRequestToken(consumer, OAuth.OUT_OF_BAND);
          // bring the user to authUrl, e.g. open a web browser and note the PIN code
        context.startActivity(new Intent(Intent.ACTION_VIEW, Uri.parse(authUrl)));
    } catch (OAuthMessageSignerException e) {
          Log.e(TAG, e.getMessage());
    } catch (OAuthNotAuthorizedException e) {
          e.printStackTrace();
    } catch (OAuthExpectationFailedException e) {
          e.printStackTrace();
    } catch (OAuthCommunicationException e) {
          e.printStackTrace();
    }
}
```

In the code

```
public static boolean postUpdate(String status){
    OAuthConsumer cons = new CommonsHttpOAuthConsumer(CONSUMER_KEY, CONSUMER_SECRET);
          cons.setTokenWithSecret(TOKEN, TOKEN_SECRET);

          // create a request that requires authentication
        HttpPost post = new HttpPost("http://twitter.com/statuses/update.xml");

        final List<NameValuePair> nvps = new ArrayList<NameValuePair>();
        // 'status' here is the update value you collect from UI
        nvps.add(new BasicNameValuePair("status", status));
        post.setEntity(new UrlEncodedFormEntity(nvps, HTTP.UTF_8));
        // set this to avoid 417 error (Expectation Failed)
        post.getParams().setBooleanParameter(CoreProtocolPNames.USE_EXPECT_CONTINUE, false);
        cons.sign(post);
        HttpClient client = new DefaultHttpClient();
        // send the request
        final HttpResponse response = client.execute(post);
        // response status should be 200 OK
        int statusCode = response.getStatusLine().getStatusCode();
        final String reason = response.getStatusLine().getReasonPhrase();
        // release connection
        response.getEntity().consumeContent();
        if (statusCode != 200) {
            Log.e(TAG, reason);
            throw new OAuthNotAuthorizedException();
        }
}
```

In the code

LifeLog.java

```
if(TwitterApi.twitterPin.equals("")){
    String pin = etTwitPIN.getText().toString();
    //registering a new account if not existing
    if(pin.equals("")){
          TwitterApi.register(this);
          tbTwit.setChecked(false);
    }
    else if( TwitterApi.setPin(pin) ){
          etTwitPIN.setText("");
          etTwitPIN.setEnabled(false);
          ctrlBtnLayout.removeView(etTwitPIN);
          ctrlBtnLayout.refreshDrawableState();
          SettingsManager.setTwitterTokens(this, TwitterApi.TOKEN,
    TwitterApi.TOKEN_SECRET);
          twitMotion = true;
    }
}
else if( TwitterApi.setPin(TwitterApi.twitterPin) ){
    etTwitPIN.setEnabled(false);
    ctrlBtnLayout.removeView(etTwitPIN);
    ctrlBtnLayout.refreshDrawableState();
    SettingsManager.setTwitterTokens(this, TwitterApi.TOKEN,
    TwitterApi.TOKEN_SECRET);
    twitMotion = true;
}
```

Oauth Authentication Flow

Oauth Authentication Flow