Android HttpPost message won't send its payload across the wire

Asked 13 years, 4 months ago Modified 11 years, 9 months ago Viewed 5k times

Part of Mobile Development Collective

5

I'm trying to send a simple string as the contents of a HttpPost message.

The problem is, the body of the HttpPost message never makes it to the wire. (Says the Wireshark capture). The header looks just fine though (including the correctly calculated Content-Length.

Here's what the code looks like:

```
String url = "http://1.2.3.4/resource";
HttpClient client = new DefaultHttpClient();
String cmd = "AT+AVLPOS\r\n";
StringEntity se = new StringEntity(cmd);
se.setContentType("text/plain");

HttpPost request = new HttpPost(url);
request.setHeader("Content-Type","text/plain");
request.setEntity(se);

HttpResponse response = client.execute(request);
[...]
```

The string should be ASCII-encoded, but that's a detail at this point.

This is what shows up in WireShark: -> note that lines marked with + are what's sent, and - is what's received.

```
+POST /resource HTTP/1.1
+Content-Type: text/plain
+Content-Length: 11
+Host: 1.2.3.4
+Connection: Keep-Alive
+User-Agent: Apache-HttpClient/UNAVAILABLE (java 1.4)
+Expect: 100-Continue

-HTTP/1.1 200 OK
-Content-Type: text/plain
-Transfer-Encoding: chunked

-4
-OK
```

This is what *should* show up (wrote a very simple console app in C# to do this, it just works):

```
+POST /resource HTTP/1.1
+Content-Type: text/plain
+Host: 1.2.3.4
+Content-Length: 11
+Expect: 100-continue
+Connection: Keep-Alive
+
-HTTP/1.1 200 OK
-Content-Type: text/plain
-Transfer-Encoding: chunked
-
+AT+AVLPOS
+
-4
-OK
-
-48
-$AVTMR,99999999,204810,A,1234.2218,N,0123.1051,E,0,20,150811,0,REQ*69
-
-0
-
```

Any suggestions?

MD java android network-programming http-post

Share Improve this question Follow

asked Aug 16, 2011 at 12:50

Cristian Diaconescu
35.6k ● 32 ● 148 ● 239

This should answer your question :) – flob Aug 16, 2011 at 13:58

@flob I think I've read all SO questions related to Android Http Post, including that one - that's how I got this far, but it still doesn't work. – Cristian Diaconescu Aug 16, 2011 at 21:36

1 Answer

Sorted by: Highest score (default)

13

I've figured it out, AND I've learned something today.

Long story short: disable the HttpClient's `HTTP Post expect-continue` handshake, by setting one of its parameters, this will send the whole request message in one chunk.

```
//set up HttpPost request as before
HttpClient client = new DefaultHttpClient();
```

```
client.getParams().setBooleanParameter("http.protocol.expect-continue", false);
HttpResponse response = client.execute(request);
[...]
```

Now here's how I got there, maybe this will help someone someday.

First I derived from a `HttpEntityWrapper` and used that as my request entity to see what gets called when, and found out that the `Entity`'s `writeTo(OutputStream)` method was never called at all.

Then I started to look at why, in the case of the "correct" behaviour, the POST request wasn't sent all at once, and instead, the request headers were sent, then the response header is received, THEN the request body is sent.

IT's all got to do with the `HTTP Post expect-continue` handshake. Read more about it on Haacked. If the expect-continue header is sent in a request, the Http server SHOULD reply with a `100 Continue` message signifying "OK, I will accept your message", or with an error, stopping the possibly long POST message in its tracks.

Unfortunately, the web server I run against is a bare bones implementation that runs on a chip, and it sends the wrong reply (`200 OK` instead of `100 Continue`).
The default implementation of the .NET Http Client seems to be more forgiving here: it treats the `200` message as `100 Continue`, *shrugs*, and gets on its way to send the request body.

Not so with the Http client implementation of Android (API level 7).

Next thing I tried was to disable the `expect-continue` handshake completely, in order to make the HttpClient send the whole request. To my surprise and joy, this was handled fine by the web server, which replied with the information I wanted. Yay!

Share

Improve this answer

Follow

edited Mar 28, 2013 at 11:31

answered Aug 16, 2011 at 21:32

Cristian Diaconescu

35.6k ● 32 ● 148 ● 239

1 Thank you!! I had a POST that was just hanging on client.execute no matter what I tried. You fix has saved the day. – ShibbyUK Jan 25, 2012 at 12:50