

# Introduction to Android

## Http post

CS 436 Software Development on Mobile

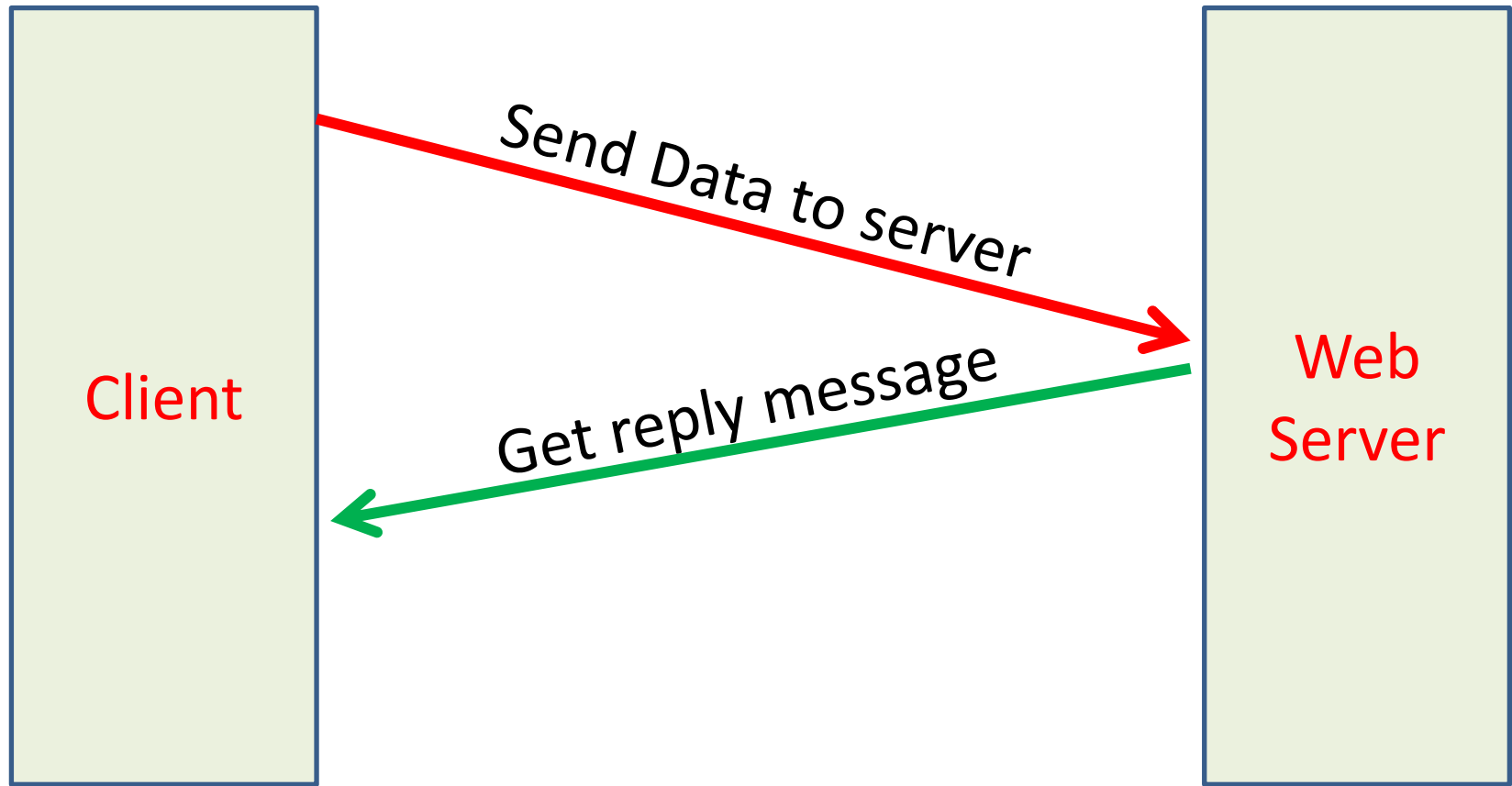
**Dr.Paween Khoenkaw**

Department of Computer Science  
Maejo University

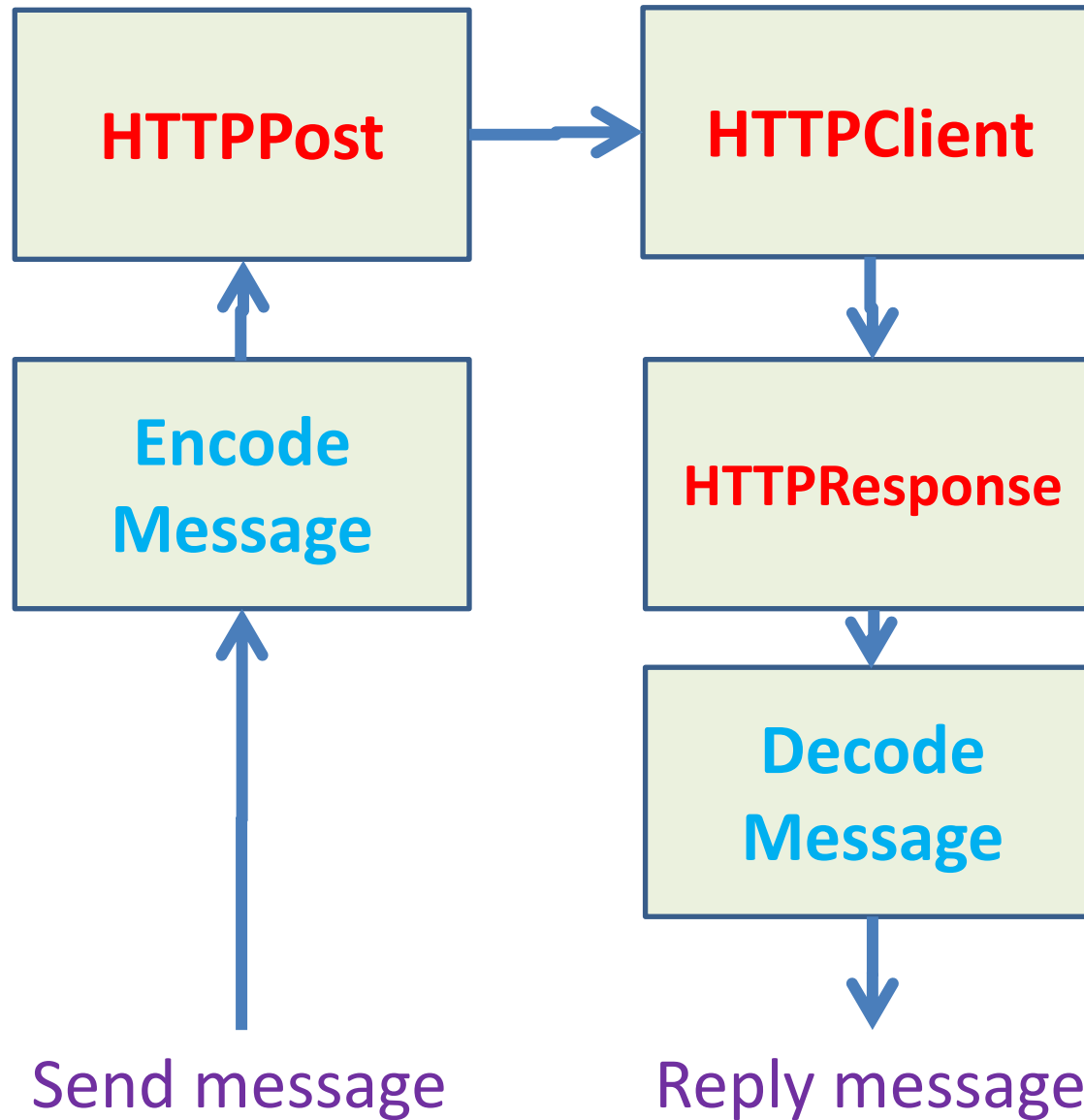


# HTTP Post

# HTTP Post



# HTTP Post



.....

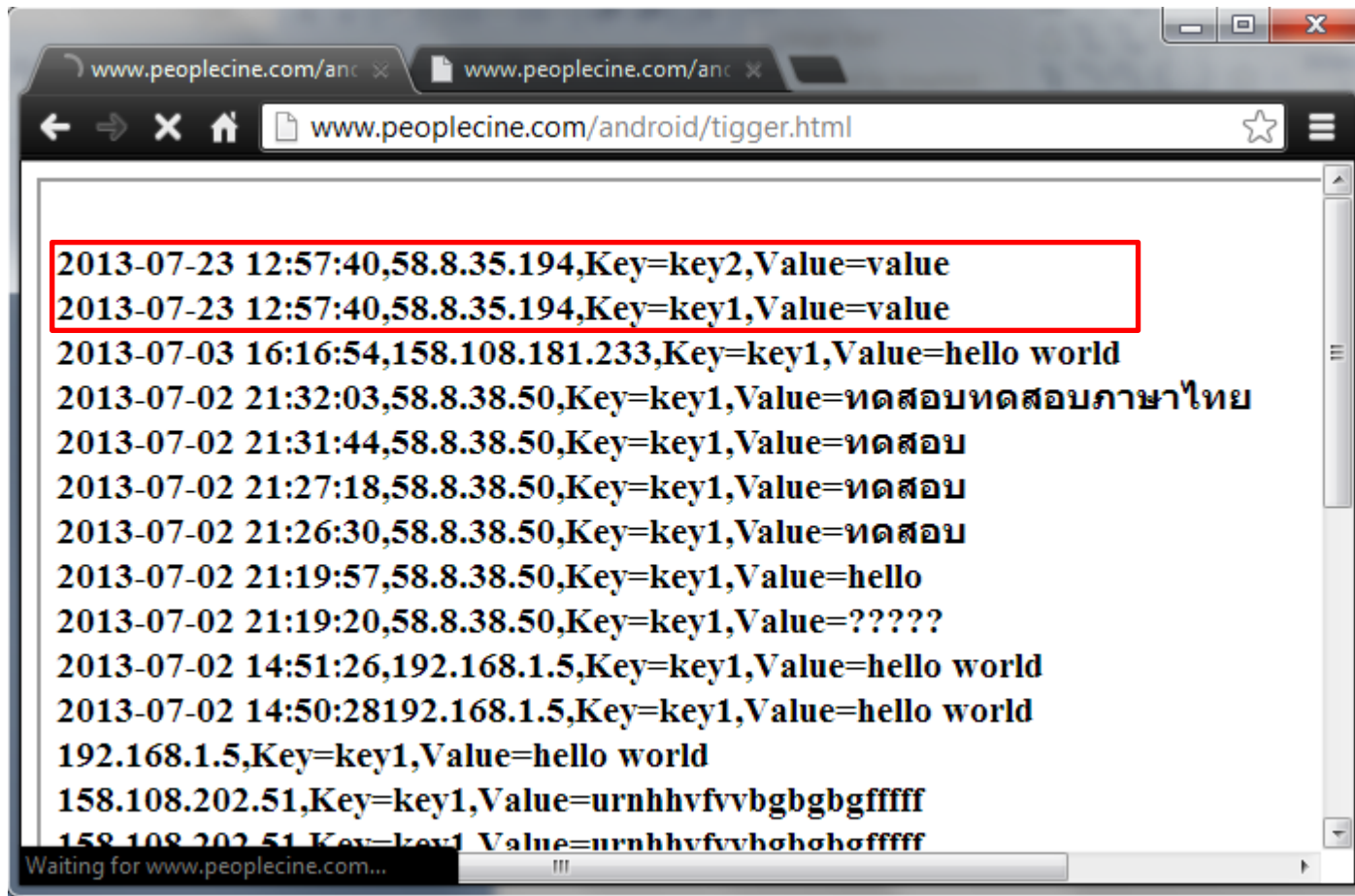
# HTTP Post

- 1) Prepare Message in **ArrayList** (key->value)
- 2) Create HttpClient and HttpResponse instant
- 3) Create HTTPPost instant and assign target URL
- 4) Encode Message in **UTF-8**
- 5) Set Message to HTTPPost
- 6) Execute HttpClient
- 7) Wait until done**
- 8) Get HttpResponse and decode to string
- 9) Step 6-8 Execute in AsyncTask Thread

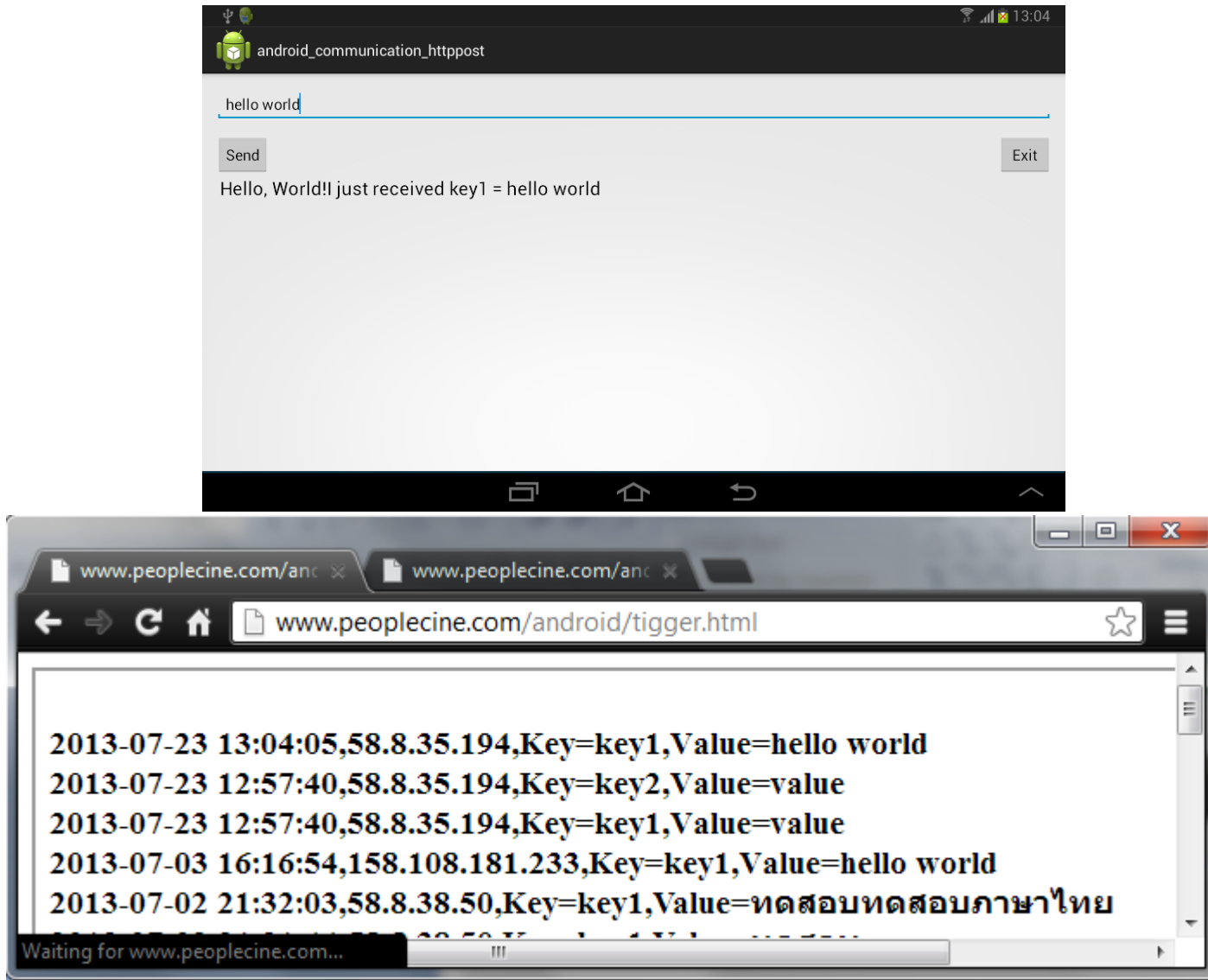
```
<uses-permission android:name="android.permission.INTERNET"/>
```

# HTTP Post

<http://www.peoplecine.com/android/cgi1.php?key1=value1&key2=value2>



# HTTP Post



Project : android\_communication\_httppost

# HTTP Post

- 1) Prepare Message in **ArrayList** (key->value)
- 2) Create HttpClient and HTTPResponse instant
- 3) Create HTTPPost instant and assign target URL

```
String ServerAddr="http://www.peoplecine.com/android";  
HttpClient httpclient = new DefaultHttpClient();  
HttpPost httppost = new HttpPost(ServerAddr+"/cgi1.php");  
HttpResponse response;  
List<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>(2);  
nameValuePairs.add(new BasicNameValuePair("key1",  
edittext1.getText().toString()));
```



# HTTP Post

- 4) Encode Message in **UTF-8**
- 5) Set Message to HTTPPost
- 6) Execute HTTPClient
- 7) Wait until done**
- 8) Get HTTPResponse and decode to string

```
httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs,"UTF-8"));
response = httpclient.execute(httppost);
res= inputStreamToString(response.getEntity().getContent()).toString();
        } catch (ClientProtocolException e) {
        } catch (IOException e) {
        }

return res;
```

# HTTP Post

Read data in InputStream and return as string

```
private StringBuilder inputStreamToString(InputStream is) {  
    String line = "";  
    StringBuilder total = new StringBuilder();  
    BufferedReader rd = new BufferedReader(new InputStreamReader(is));  
    try {  
        while ((line = rd.readLine()) != null) {  
            total.append(line);  
        }  
    } catch (IOException e) {  
        e.printStackTrace();  
    }  
    return total;  
}
```

# HTTP Post

Multipart, MIME  
Send file to server

# HTTP Post

HTTPPost can only send plain text (7bit)

Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	\$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-`	63	3F	?	95	5F	_	127	7F	DEL

Dec	Hex	Char
128	80	Ç
129	81	ü
130	82	ë
131	83	â
132	84	ä
133	85	à
134	86	ã
135	87	ç
136	88	ê
137	89	è
138	8A	ë
139	8B	ì
140	8C	î
141	8D	ï
142	8E	Ä
143	8F	Å
144	90	É
145	91	æ
146	92	Ē
147	93	ô
148	94	ö
149	95	ò
150	96	ù
151	97	û
152	98	ÿ
153	99	ö
154	9A	Ü
155	9B	€
156	9C	£
157	9D	¥
158	9E	ℳ
159	9F	ƒ

Dec	Hex	Char
160	A0	á
161	A1	í
162	A2	ó
163	A3	ú
164	A4	ñ
165	A5	Ñ
166	A6	à
167	A7	o
168	A8	¿
169	A9	ˆ
170	AA	˜
171	AB	½
172	AC	¼
173	AD	ı
174	AE	«
175	AF	»
176	B0	⌘
177	B1	⌘
178	B2	⌘
179	B3	⌘
180	B4	⌘
181	B5	⌘
182	B6	⌘
183	B7	⌘
184	B8	⌘
185	B9	⌘
186	BA	⌘
187	BB	⌘
188	BC	⌘
189	BD	⌘
190	BE	⌘
191	BF	⌘

Dec	Hex	Char
192	C0	L
193	C1	⌞
194	C2	⌞
195	C3	⌞
196	C4	⌞
197	C5	⌞
198	C6	⌞
199	C7	⌞
200	C8	⌞
201	C9	⌞
202	CA	⌞
203	CB	⌞
204	CC	⌞
205	CD	⌞
206	CE	⌞
207	CF	⌞
208	D0	⌞
209	D1	⌞
210	D2	⌞
211	D3	⌞
212	D4	⌞
213	D5	⌞
214	D6	⌞
215	D7	⌞
216	D8	⌞
217	D9	⌞
218	DA	⌞
219	DB	⌞
220	DC	⌞
221	DD	⌞
222	DE	⌞
223	DF	⌞

Dec	Hex	Char
224	E0	α
225	E1	β
226	E2	γ
227	E3	π
228	E4	Σ
229	E5	σ
230	E6	μ
231	E7	Υ
232	E8	ϕ
233	E9	θ
234	EA	Ω
235	EB	δ
236	EC	∞
237	ED	Φ
238	EE	ε
239	EF	∩
240	F0	≡
241	F1	±
242	F2	Σ
243	F3	≤
244	F4	∫
245	F5	∫
246	F6	÷
247	F7	≈
248	F8	◊
249	F9	•
250	FA	·
251	FB	√
252	FC	²
253	FD	³
254	FE	⁴
255	FF	⁵

# HTTP Post

HTTPPost can only send plain text (7bit)

We use Binary-to-text encoding algorithm to solve this

Encoding	Data type	Efficiency	Programming language implementations	Comments
<a href="#">Ascii85</a>	Arbitrary	$\frac{4}{5}$	<a href="#">awk</a> , <a href="#">C</a> , <a href="#">C#</a> , <a href="#">F#</a> , <a href="#">Java</a> , <a href="#">Perl</a> , <a href="#">Python</a> , <a href="#">Python (2)</a>	
<a href="#">Base16</a> (hexadecimal)	Arbitrary	$\frac{1}{2}$	Probably any language around	
<a href="#">Base32</a>	Arbitrary	$\frac{5}{8}$ (8 bits)	<a href="#">ANSI C</a> , <a href="#">Java</a>	
<a href="#">Base64</a>	Arbitrary	$\sim > 75\%$ (8 bits)	<a href="#">C</a> , <a href="#">C (2)</a> , many others	
<a href="#">BinHex</a>	Arbitrary	$\frac{3}{4}$ (BinHex $\geq 2.0$ )	<a href="#">Perl</a> , <a href="#">C</a> , <a href="#">C (2)</a>	Forgotten since the mid-1980s
<a href="#">Intel HEX</a>	Arbitrary	$\sim < 50\%$	<a href="#">C library</a> , <a href="#">C++</a>	Usually used for chip programming/flashing

# HTTP Post

HTTPPost can not handle multiple type data

String Part
Binary Part

# HTTP Post

**Multipurpose Internet Mail Extensions (MIME)** is an Internet standard that extends the format of email to support:

- Text in character sets other than ASCII
- Non-text attachments
- Message bodies with multiple parts
- Header information in non-ASCII character sets

Type multipart

For archives and other objects made of more than one part.

multipart/mixed: MIME Email; Defined in RFC 2045 and RFC 2046

multipart/alternative: MIME Email; Defined in RFC 2045 and RFC 2046

multipart/related: MIME Email; Defined in RFC 2387 and used by MHTML (HTML mail)

multipart/form-data: MIME Webform; Defined in RFC 2388

multipart/signed: Defined in RFC 1847

multipart/encrypted: Defined in RFC 1847

# HTTP Post

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary=frontier

This is a message with multiple parts in MIME format.
--frontier
Content-Type: text/plain

This is the body of the message.
--frontier
Content-Type: application/octet-stream
Content-Transfer-Encoding: base64

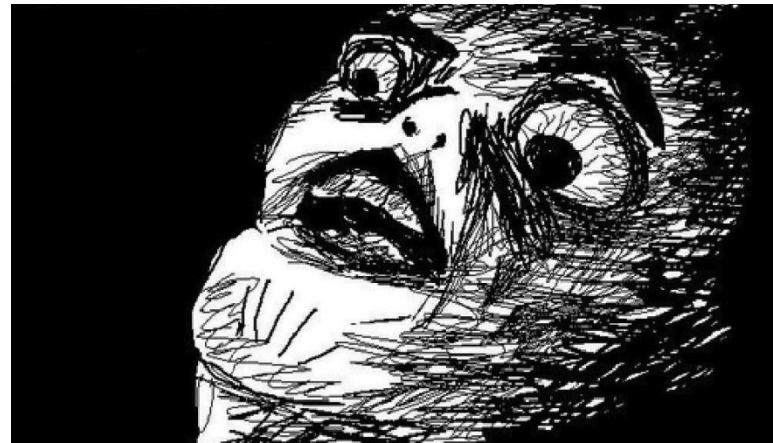
PGh0bWw+CiAgPGhlYWQ+CiAgPC9oZWFKPgogIDxib2R5PgogICAgPHA+VGhpcyBpcyB0aGUg
Ym9keSBvZiB0aGUgbWVzc2FnZS48L3A+CiAgPC9ib2R5Pgo8L2h0bWw+Cg==
--frontier--
```

Example of MIME encode data



# HTTP Post

Most platform can handle MIME in HTTPPost  
But Android don't



# HTTP Post

Apache HttpComponents Http MIME module library



# HTTP Post

## Apache HttpComponents Http MIME module library

<http://hc.apache.org/downloads.cgi>

[HttpCore \(DEV\)](#)  
[HttpAsyncClient \(DEV\)](#)

### Legacy

[Commons HttpClient](#)

### Project

[Status](#)  
[Charter](#)  
[Bylaws](#)  
[Goals](#)

### Project

### Documentation

► [Project Information](#)









### ASF

[ASF Home Page](#)   
[Foundation](#) 





## HttpClient 4.2.5 (GA)

[KEYS](#)  [Release Notes](#) 

### Binary

- [4.2.5.tar.gz](#) [[md5](#) pgp - [4.2.5.zip](#) [[md5](#) pgp - [4.2.5.tar.gz \(OSGi bundle\)](#) [[md5](#) pgp - [4.2.5.zip \(OSGi bundle\)](#) [[md5](#) pgp 

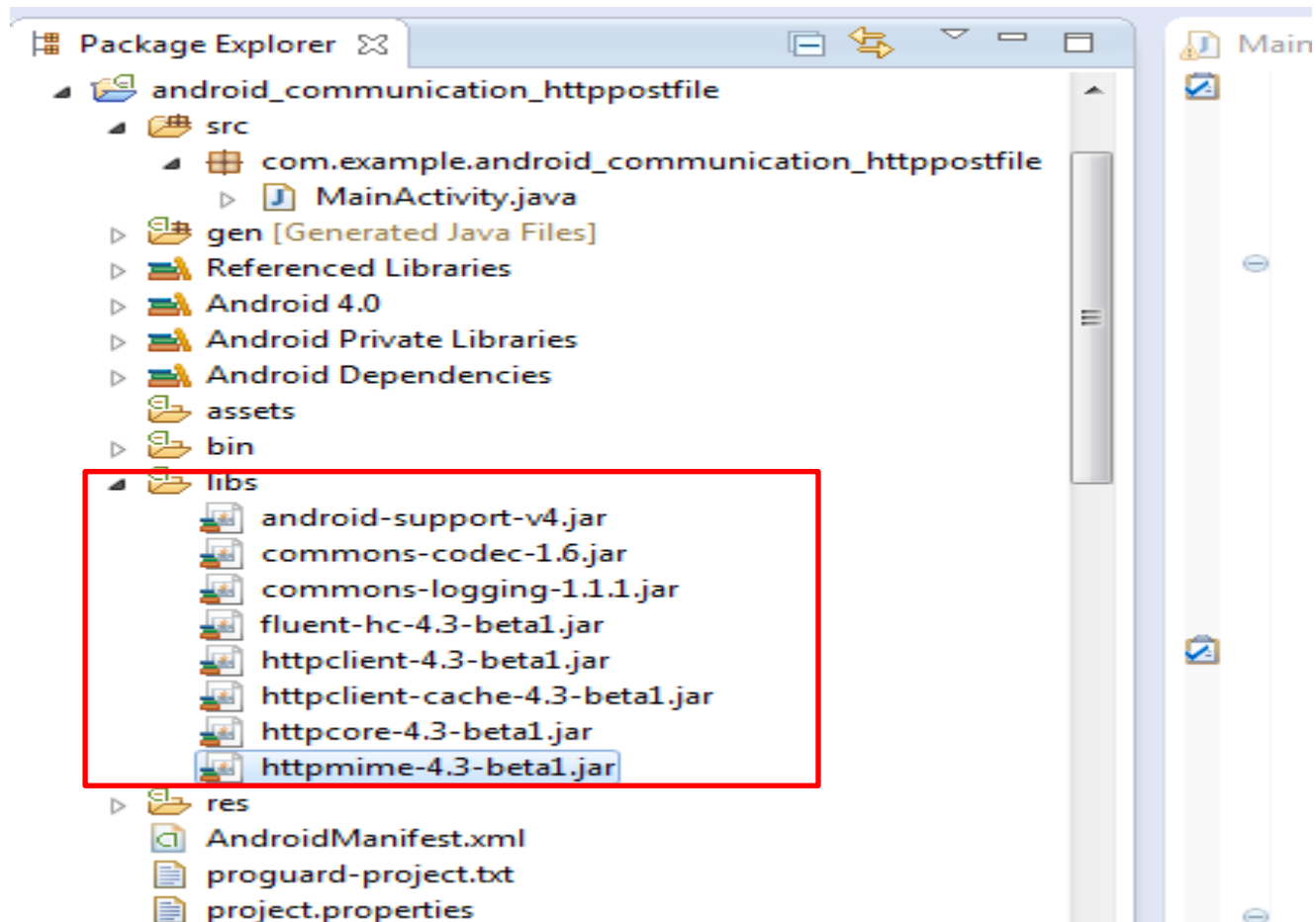
### Source

- [4.2.5.tar.gz](#) [[md5](#) pgp - [4.2.5.zip](#) [[md5](#) pgp 

# HTTP Post

## Apache HttpComponents Http MIME module library

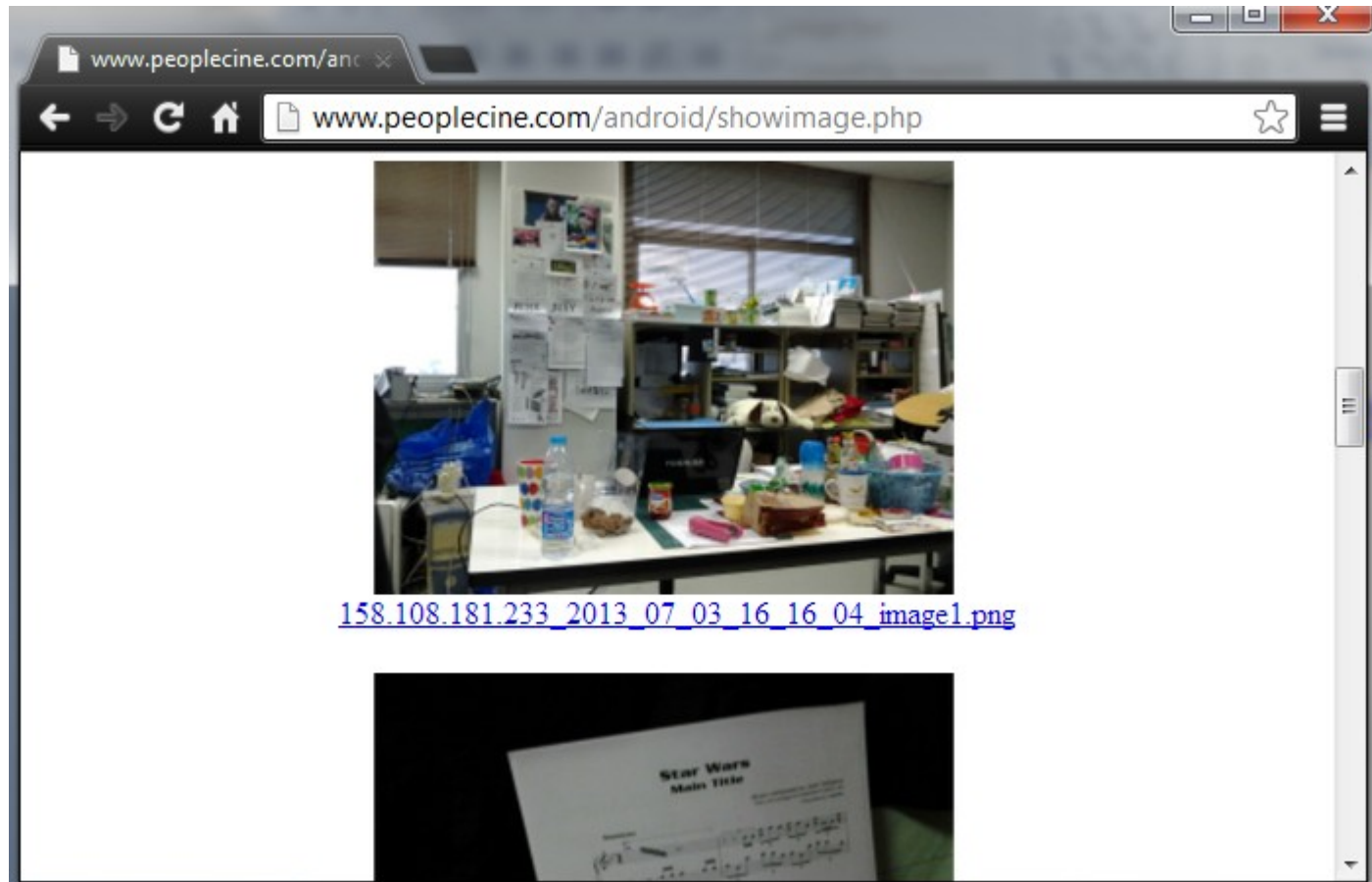
<http://hc.apache.org/downloads.cgi>



# HTTP Post

Send file to Server

<http://www.peoplecine.com/android/showimage.php>



# HTTP Post

```
private class PostFileHttp extends AsyncTask<String,Void,String>{
@Override
protected String doInBackground(String... arg0) {
HttpClient httpclient = new DefaultHttpClient();
HttpPost httppost = new HttpPost(ServerIP+"/android/cgi2.php");
HttpResponse response;
String res="";
try {
    MultipartEntity entity = new MultipartEntity();
    File myFile = new File( Environment.getExternalStorageDirectory(), "001.jpg" );
    FileBody fileBody = new FileBody(myFile);
    entity.addPart("file", fileBody);
    httppost.setEntity(entity);
    response = httpclient.execute(httppost);
    Log.v("result",String.format("%s",
    inputStreamToString(response.getEntity().getContent()).toString()));

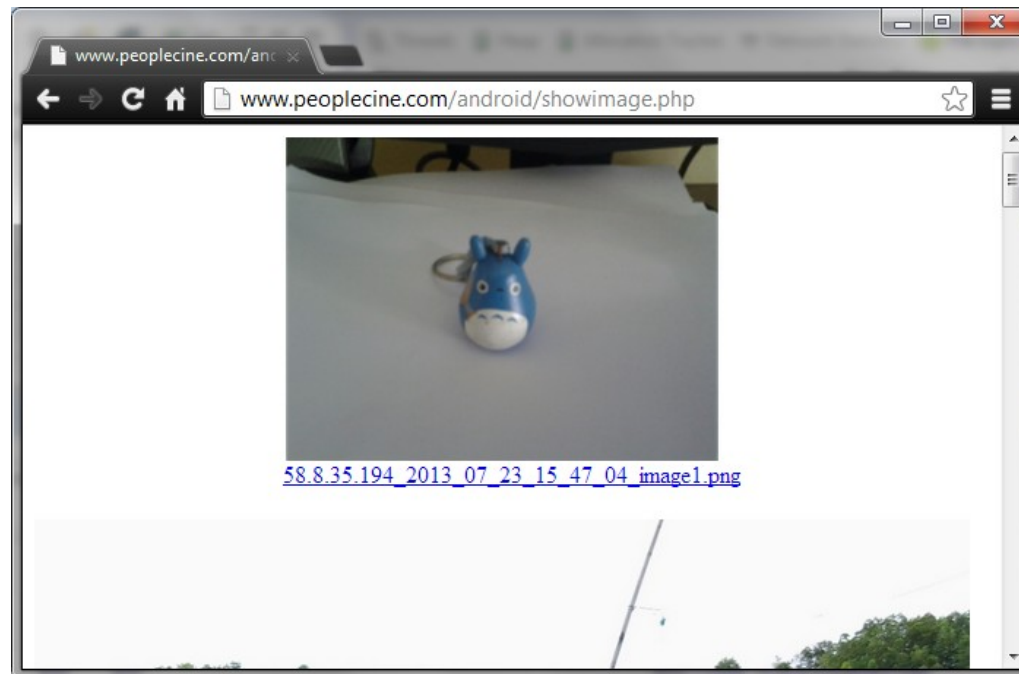
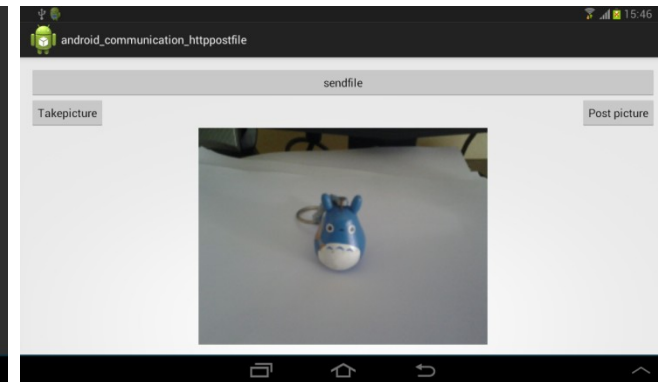
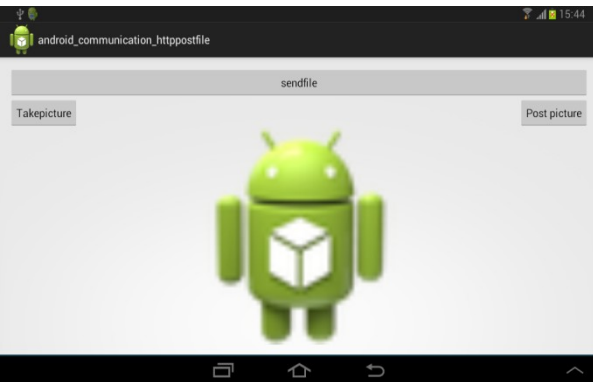
                } catch (ClientProtocolException e) {
                } catch (IOException e) {
                }

return res;
}
```

# HTTP Post

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

# HTTP Post



Project : android\_communication\_httppostfile



# HTTP Post

## Image processing step

```
imageView1.buildDrawingCache(true);  
bitmap=imageView1.getDrawingCache(true).copy(Config.ARGB_8888,false);  
imageView1.destroyDrawingCache();  
ByteArrayOutputStream bos = new ByteArrayOutputStream();  
bitmap.compress(CompressFormat.PNG, 99, bos);  
byte[] data = bos.toByteArray();
```

Create ByteArray ready to post data from image view

# HTTP Post

```
private class PostImageHttp extends AsyncTask<String,Void,String>{
@Override
protected String doInBackground(String... arg0) {
HttpClient httpclient = new DefaultHttpClient();
HttpPost httppost = new HttpPost(ServerIP+"/android/cgi2.php");
HttpResponse response;
String res="";
try {
.. Image Processing step ....
ByteArrayBody bab = new ByteArrayBody(data, "image1.png");
MultipartEntity entity = new MultipartEntity( HttpMultipartMode.BROWSER_COMPATIBLE);
entity.addPart("file", bab);
httppost.setEntity(entity);
response = httpclient.execute(httppost);
Log.v("result",String.format("%s",
InputStreamToString(response.getEntity().getContent()).toString()));
} catch (ClientProtocolException e) {
} catch (IOException e) {
}return res;
    }
}
```

# HTTP Post

**Upload progress checking**