

http://

PART 1: GOING DEEP ON HTTP
PART 2: HIGH LEVEL HTTP APIs

Jesse Wilson



Hi, I'm Jesse

- Infrastructure nerd
- Former Android core libraries lead
 - Maintained Android's HttpURLConnection and Apache HTTP client
- Today I'm at Square
 - Working on OkHttp & MockWebServer

Plan

- HTTP in 2013
- HTTP clients for Android
 - Apache HTTP client
 - HttpURLConnection
 - OkHttp
- Essential dev tools
 - Charles Web Debugging Proxy
 - MockWebServer

Plan

- Retrofit
- Picasso
- Volley

Follow along

http://tinyurl.com/androidhttp

The screenshot shows a web browser window displaying a GitHub repository page. The URL in the address bar is <https://github.com/swankjesse/android-http-examples/tree/master/src/main/java/com/publicobject/http>. The repository name is **swankjesse / android-http-examples**. The 'Code' tab is selected. The commit history shows the following entries:

Author	Date	Message
square-build-bot	6 minutes ago	Initial API examples [square-build-bot]
cache	6 minutes ago	Initial API examples [square-build-bot]
get	6 minutes ago	Initial API examples [square-build-bot]

HTTP in 2013

Demo

Port 80

```
$ telnet swank.ca 80
Trying 69.163.193.18...
Connected to swank.ca.
Escape character is '^]'.
GET / HTTP/1.1
Host: swank.ca

HTTP/1.1 200 OK
Date: Tue, 08 Jan 2013 00:16:56 GMT
Server: Apache
Last-Modified: Wed, 14 Sep 2011 19:12:02 GMT
ETag: "27b-4aceb88f6a480"
Accept-Ranges: bytes
Content-Length: 199
Vary: Accept-Encoding
Content-Type: text/html; charset=utf-8

<html>
  <head><title>swank.ca</title></head>
  <body align="center" bgcolor="#003399">
    
  </body>
</html>
```

Proxies

- HTTP proxies exist, annoy
- You speak HTTP to the proxy. It speaks HTTP to the world.
- HTTP and HTTPS

Caching

- Fundamental
- HTTP distinguishes between resources (GET) and actions (POST).

Caching example

Caching: for servers

- Use the Cache-Control & ETag headers.

HTTP/1.1 200 OK

Cache-Control: max-age=3600

Etag: v3

...

- Using modification dates is clumsy! The user agent is dumb and uses heuristics.

Caching: for servers

- Serving static resources like images?
Use a permanent URL and let it cache forever!
- When the resource is updated, just change the URL!

Content Delivery Networks (CDNs)

- Occasionally, you're bottlenecked by the speed of light



<http://www.igvita.com> Illya Grigorik

Thumbor



<http://im.squareup.com/stand.png>



<http://thumbor.squareup.com/unsafe/160x160/http://im.squareup.com/stand.png>

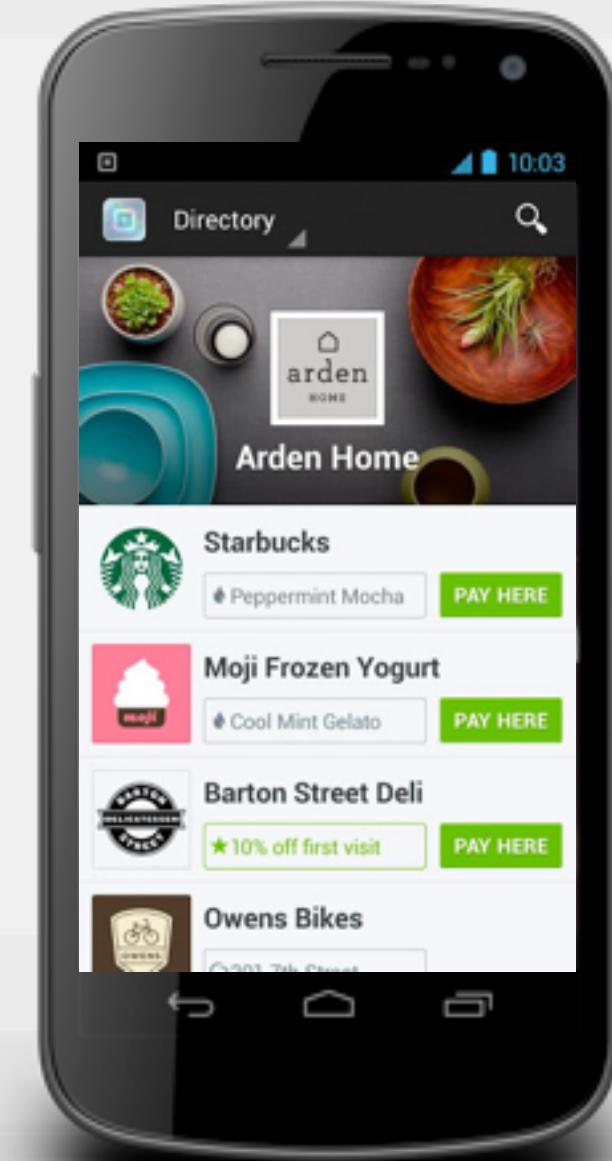
Gzip

JSON: **53** KiB

Proto: **31** KiB

gzipped JSON: **9.1** KiB

gzipped Proto: **8.5** KiB



Gzip: Clients

- Use the Accept-Encoding request header:

```
GET /directory.json HTTP/1.1  
Accept-Encoding: gzip
```

• • •

Gzip: Servers

- Using gzip on already-compressed content like images is wasteful. Don't double compress content types like image/png or image/jpeg

DNS

- Redundant datacenters?
- Supporting IPv6 and IPv4?
- Each host may have multiple IP addresses!

WiFi Hotspots



The image shows the McDonald's WiFi Hotspots landing page. At the top, there is a red navigation bar with four links: "Home" (with a house icon), "Connection Options" (with a signal icon), "Find Wi-Fi Locations" (with a location pin icon), and "Help" (with a question mark icon). To the right of the navigation bar is the text "delivered by" followed by the AT&T logo. The main content area features the McDonald's golden arches logo inside a red oval. Below the logo is a yellow button labeled "Free Connection". At the bottom of the page, there are three logos: "boingo CONNECT", "at&t WiFi", and "iPass connect". A copyright notice at the very bottom reads: "© 2010 McDonald's. All Rights Reserved. MCDONALD'S is a registered trademark of McDonald's Corporation. All other trademarks and service marks remain the sole property of their respective holders."

WiFi Hotspots

- Coffee shop?
- Airport?
- Your HTTP requests may be redirected.

HTTPS

- Transport Layer Security (TLS), the successor to Secure Sockets Layer (SSL).
- Reduces eavesdropping while you use the WiFi at Panera Bread or the airport

Certificate Authorities

- Lots of problems
- Governments want to spy

Certificate Authorities

- Browser vendors have two bad options:
 - Include a CA
 - Don't include a CA

Certificate Authorities

- Hardcode your cert in your app:

```
SSLocketFactory goodSslSocketFactory(Context context) {  
    KeyStore trusted = KeyStore.getInstance("BKS");  
    InputStream in = context.getResources().openRawResource(R.raw.truststore);  
    trusted.load(in, TRUST_STORE_PASSWORD);  
    SSLContext sslContext = SSLContext.getInstance("TLS");  
    TrustManagerFactory trustManagerFactory = TrustManagerFactory.getInstance(  
        TrustManagerFactory.getDefaultAlgorithm());  
    trustManagerFactory.init(trusted);  
    sslContext.init(null, trustManagerFactory.getTrustManagers(), null);  
    return sslContext.getSocketFactory();  
}
```

Hostname Verification

- Does *.squareup.com match beta.api.squareup.com ?
- Validate that the certificate matches the host.
- Rules followed inconsistently!

The Most Dangerous Code in the World: Validating SSL Certificates in Non-Browser Software

Martin Georgiev
The University of Texas
at Austin

Rishita Anubhai
Stanford University

Subodh Iyengar
Stanford University

Dan Boneh
Stanford University

Suman Jana
The University of Texas
at Austin

Vitaly Shmatikov
The University of Texas
at Austin

ABSTRACT

SSL (Secure Sockets Layer) is the de facto standard for secure Internet communications. Security of SSL connections against an active network attacker depends on correctly validating public-key certificates presented when the connection is established.

We demonstrate that SSL certificate validation is completely broken in many security-critical applications and libraries. Vulnerable software includes Amazon’s EC2 Java library and all cloud clients based on it; Amazon’s and PayPal’s merchant SDKs responsible for transmitting payment details from e-commerce sites to payment gateways; integrated shopping carts such as osCommerce, ZenCart, Ubercart, and PrestaShop; AdMob code used by mobile websites; Chase mobile banking and several other Android apps and libraries; Java Web-services middleware—including Apache Axis, Axis 2, Codehaus XFire, and Pusher library for Android—and *all* applications employing this middleware. Any SSL connection from any of

cations. The main purpose of SSL is to provide end-to-end security against an active, man-in-the-middle attacker. Even if the network is completely compromised—DNS is poisoned, access points and routers are controlled by the adversary, etc.—SSL is intended to guarantee confidentiality, authenticity, and integrity for communications between the client and the server.

Authenticating the server is a critical part of SSL connection establishment.¹ This authentication takes place during the SSL handshake, when the server presents its public-key certificate. In order for the SSL connection to be secure, the client must carefully verify that the certificate has been issued by a valid certificate authority, has not expired (or been revoked), the name(s) listed in the certificate match(es) the name of the domain that the client is connecting to, and perform several other checks [14, 15].

SSL implementations in Web browsers are constantly evolving through “penetrate-and-patch” testing, and many SSL-related vulnerabilities in browsers have been repaired over the years. SSL

Cookies

- Four specs:
Netscape, RFC 2109, RFC 2965, RFC 6265

“Prior to this document, there were at least three descriptions of cookies.”

“However, none of these documents describe how the Cookie and Set-Cookie headers are actually used on the Internet.”

Cookies

- Awkward on Android. No out-of-the-box place to persist across app launches.

Ranges

- Restarting a failed download sucks.
- If your app does large downloads (> 5 MiB) you should look into the Range header. You'll need to do this manually.

Sockets

"A single-user client SHOULD NOT maintain more than 2 connections with any server or proxy."

- IE 10 uses 8
- Chrome uses 6
- Firefox uses 6

Sockets

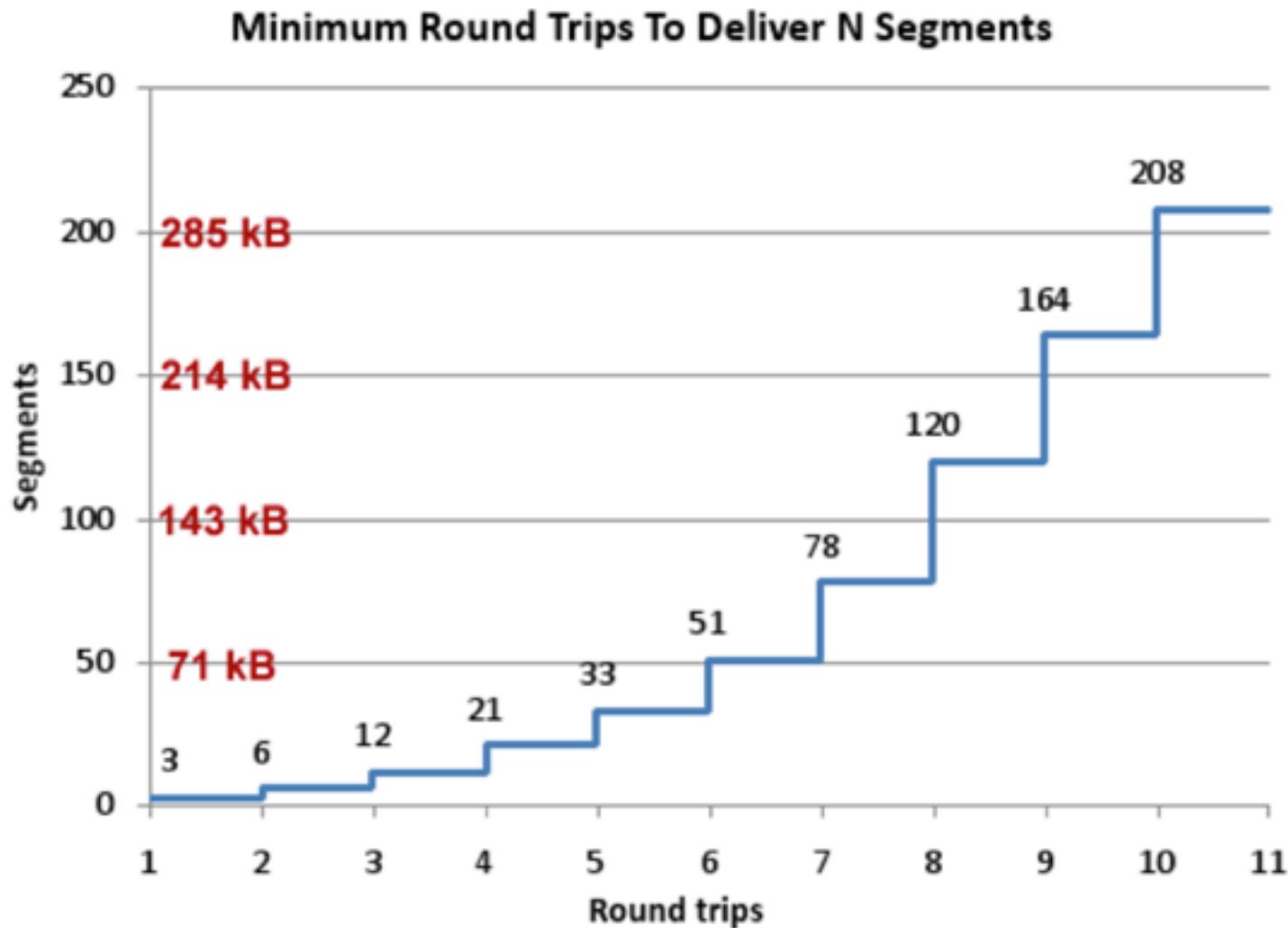
- Hacks!
 - Image spriting
 - Flattening CSS
 - Google serves content from multiple hostnames!
- These hacks hurt caching!



locale.png

icon.png

Sockets

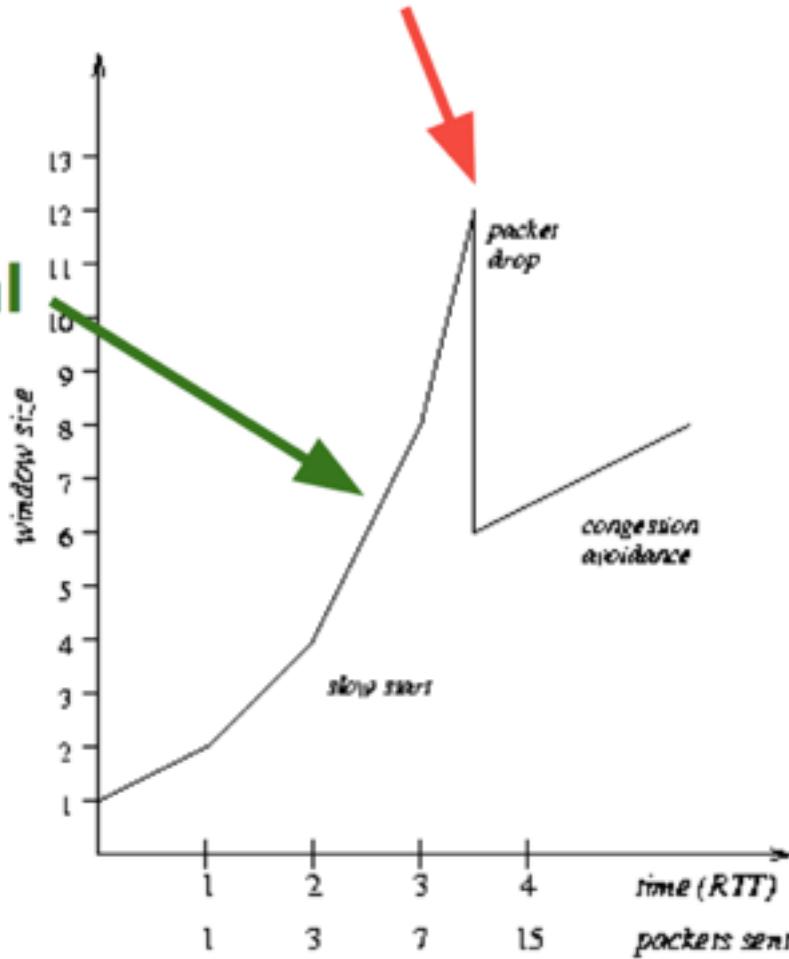


<http://www.igvita.com> Ilya Grigorik

Sockets

Exponential growth

Packet Loss



Sockets

- Connection pooling

SPDY

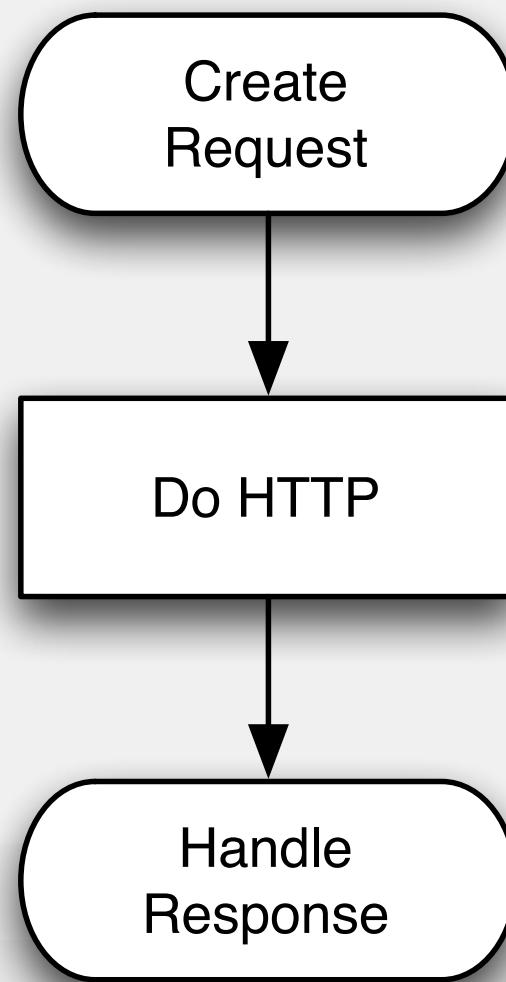
- Keep the good parts!
 - But fix the sockets problem
 - Compress more stuff
-
- Reasonable go-to-market strategy:
Chrome browser + google.com website + NPN

SPDY

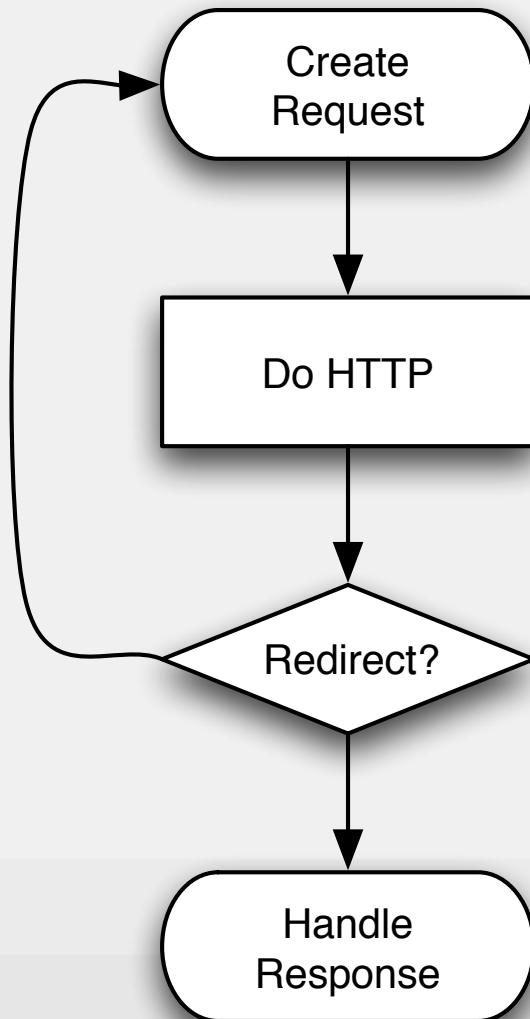
- Needs HTTPS
- Needs Android 4.1+

Android's HTTP Clients

HTTP

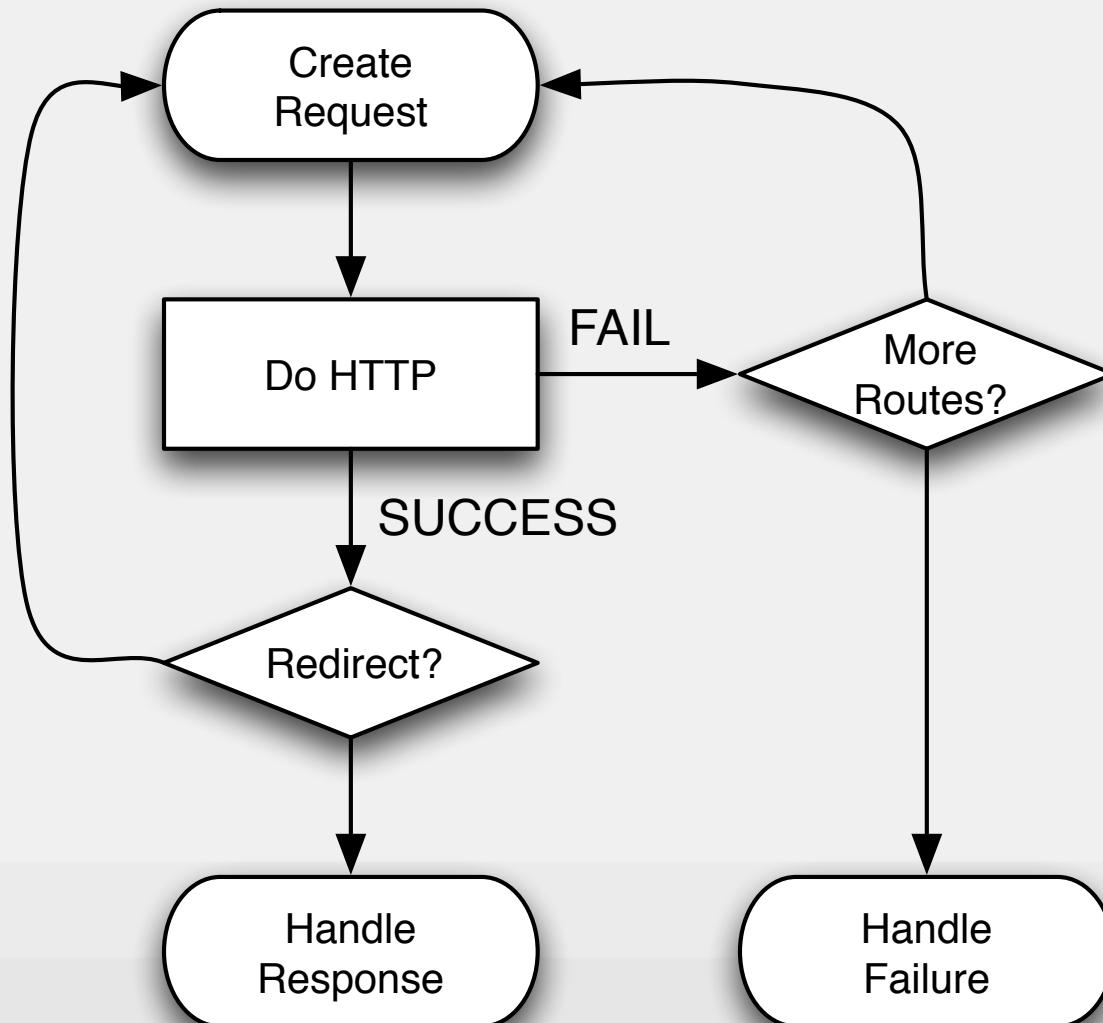


HTTP with redirects

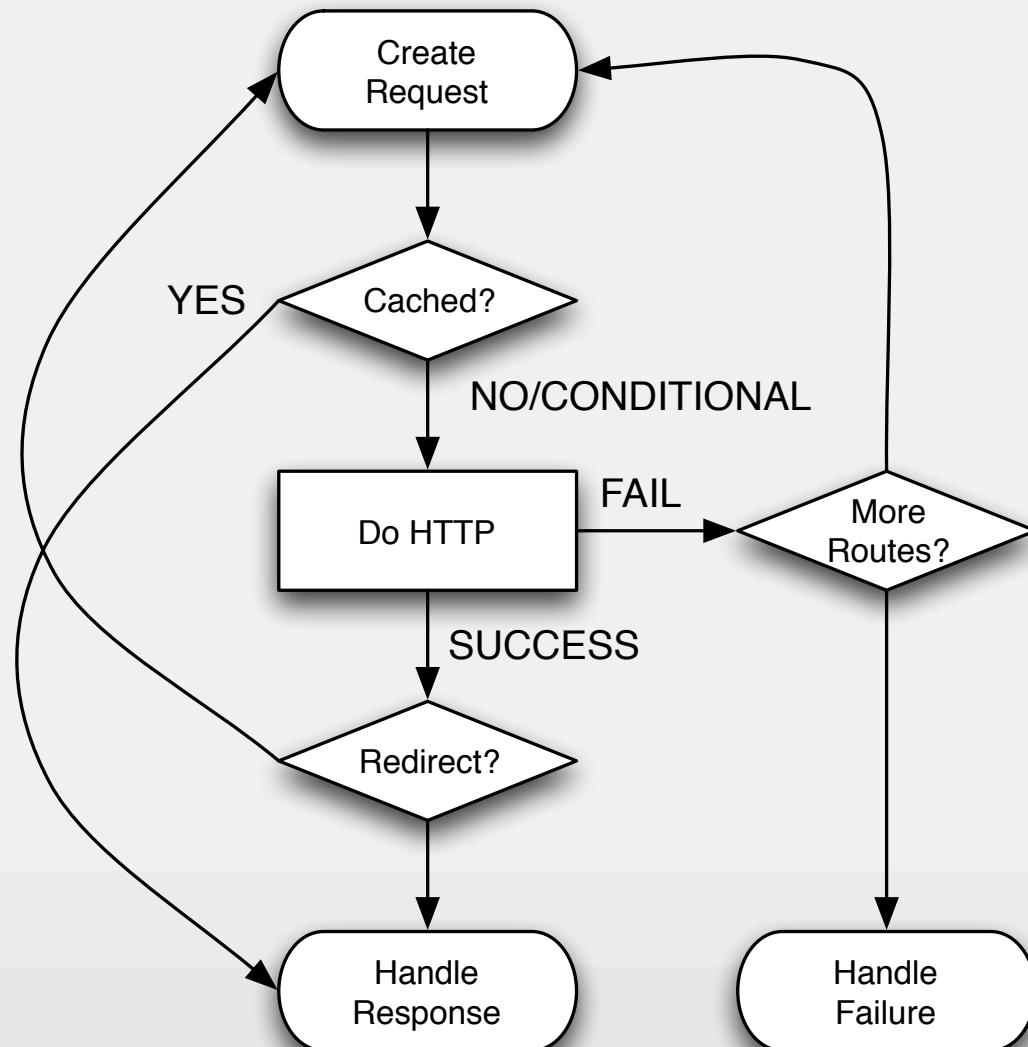


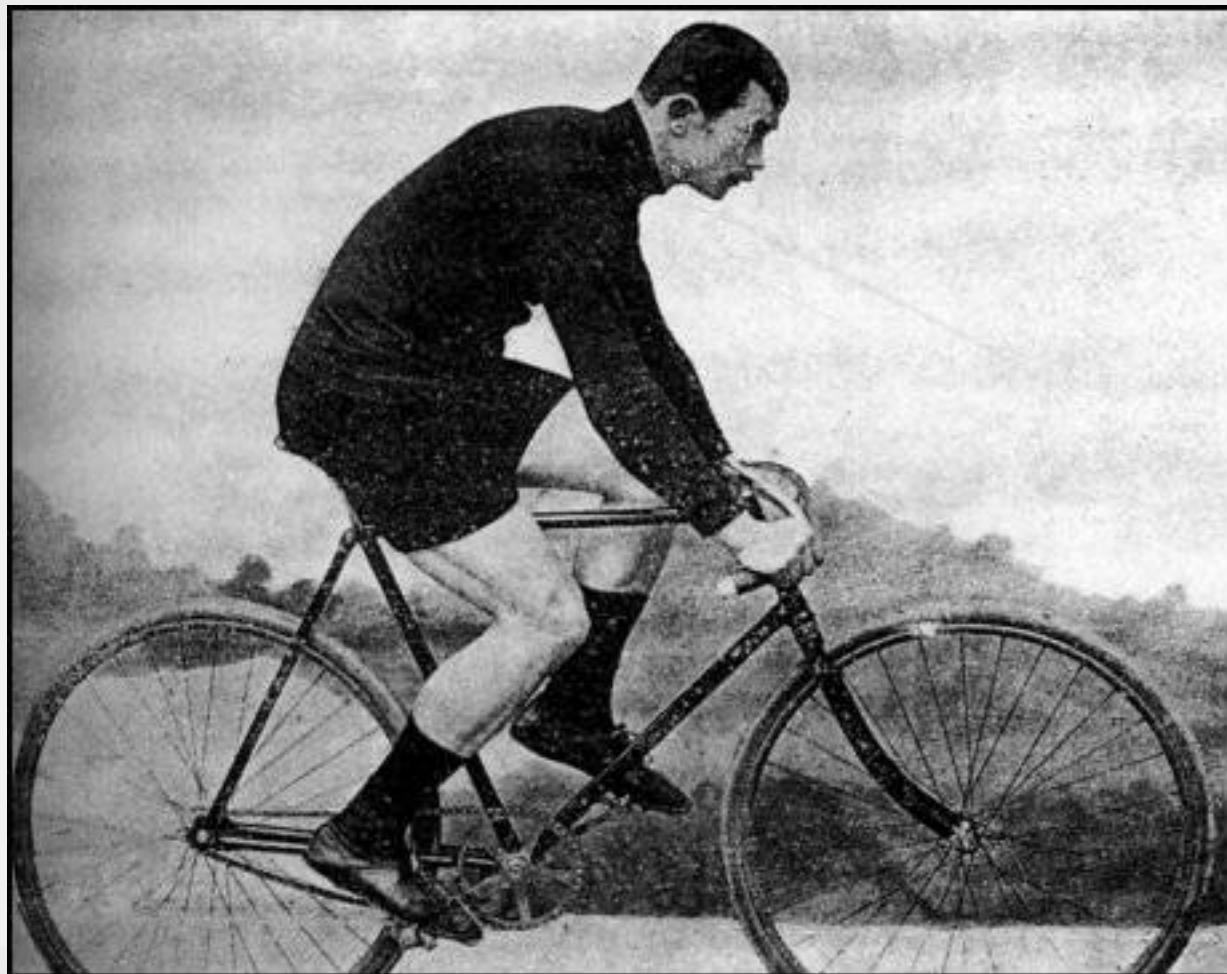
HTTP/1.1 302 Moved
Location: /account

HTTP with redirects and routing



HTTP with redirects, routing and caching





<http://www.flickr.com/photos/paukrus/>

HttpURLConnection

- Limited API, grown organically since the 1996
- ~ 9,000 lines of code

HttpURLConnection

- Problems
 - Connection pool configuration is gross system properties
 - Connection pool is buggy in Froyo and should be disabled

Demo



<http://www.flickr.com/photos/dhwright>

Apache HTTP Client

- Mature & capable
- Lots of knobs & configuration
- ~ 50,000 lines of code

Apache HTTP Client

- Problems:
 - Where does the work happen? How do I get the behavior I want?
 - Enterprisey: `HttpAbstractParamBean`

Demo



<http://www.flickr.com/photos/turbulentflow>

OkHttp

- Forked from Android 4.0's HttpURLConnection
- New features:
 - SPDY
 - Fault recovery
- ~ 12,000 lines of code



OkHttp

- Problems:
 - It's new. No deep archive of support on stackoverflow.com!
 - Still uses the old API



Demo

Essential Dev Tools

Charles

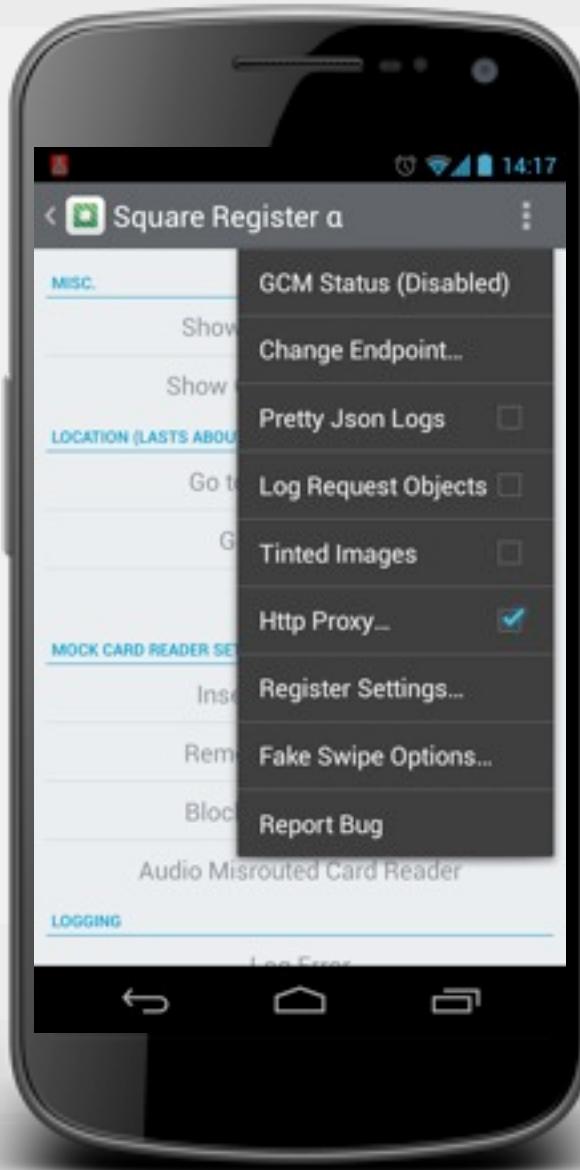
- Web Debugging Proxy
- \$50



<http://www.charlesproxy.com>

Demo

Charles



Charles

Charles 3.6.5 – Session 1 *

The Charles interface displays a network request for the URL `https://api.broadway.squareup.com/preferences`. The left pane shows the request structure, and the right pane shows the detailed response headers and body.

Request Structure:

- https://api.broadway.squareup.com
- 1.0/
- items/
 - sync
 - sync
 - sync
 - sync
- account/
 - status/
 - preferences

Response Headers:

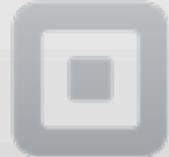
```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Keep-Alive: timeout=300
Status: 200
Cache-Control: private, max-age=0, must-revalidate
ETag: "bc70ceba41a22a5e4c5ea6be5c9ba114"
X-Frame-Options: DENY
X-Square: S=stage02.mtv.squareup.com
Strict-Transport-Security: max-age=1296000
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Encoding: gzip
```

Response Body:

```
{"tax_enabled":false,"tipping_use_manual_tip_entry":false,"skip_sig":true}
```

MockWebServer

- Does my HTTP code work?



Mocks?

EASYMock

jMock



Mocks!

It's a script!

1. Describe a scenario.
2. Exercise your code.
3. Verify.

Demo

Retrofit



Retrofit

- Manually calling HTTP services requires a lot of code
- There's pain!
 - Error handling
 - Async
 - Testing

Retrofit

- Adapts an HTTP service to a Java interface



<http://macklemore.com>

Demo

Picasso

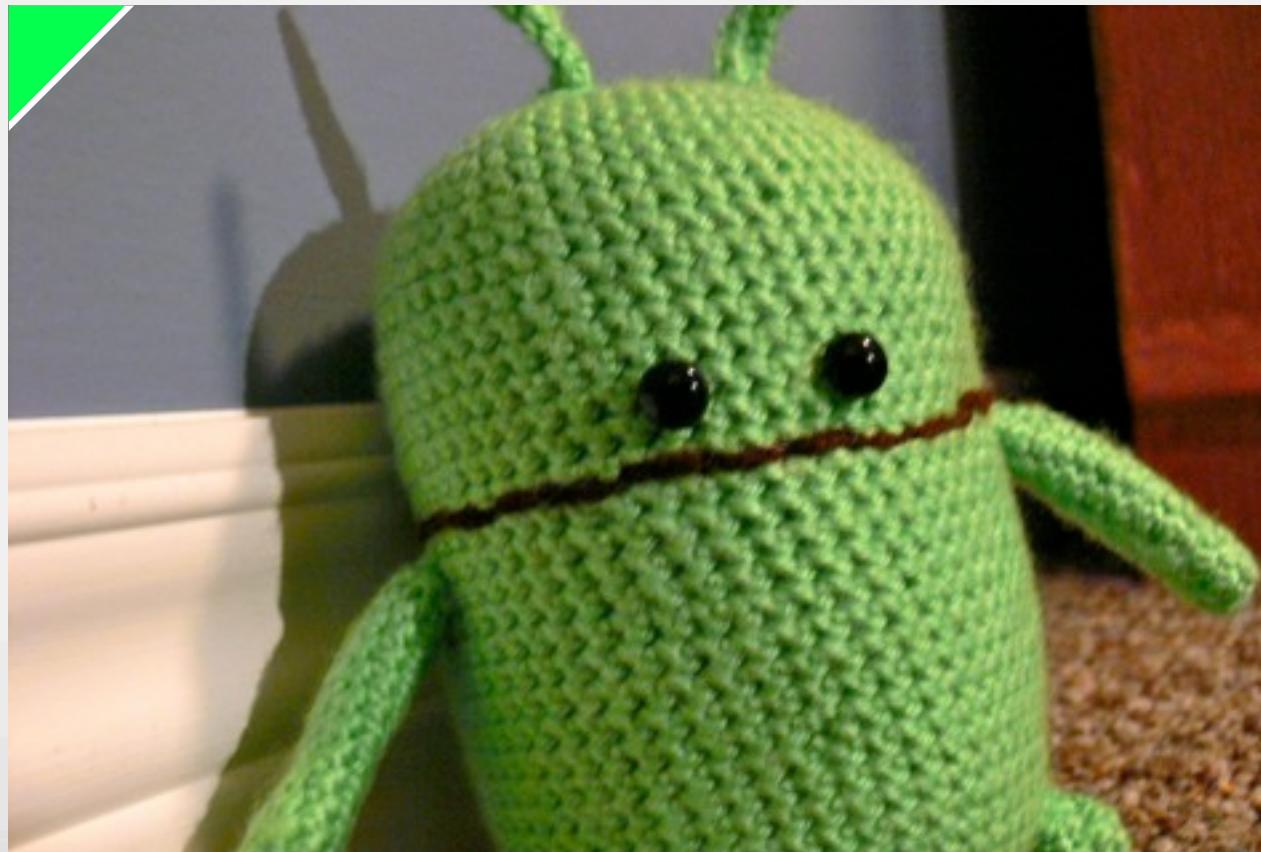


Picasso

- Manually showing images on Android requires a lot of code
- There's pain!
 - Threading
 - View recycling
 - Memory caching
 - Disk caching
 - Transforming images

Picasso

- Makes it foolishly easy to show images in your Android app



<http://www.flickr.com/photos/jumpyjodes>

Demo

Volley

Volley

- An HTTP library from the guys who make the Play Store app
- Sophisticated features
 - Request canceling
 - Request scheduler
 - Pluggable

Demo

Wrapping Up

Advice: HTTP

- HTTP has knobs
- Use them

Advice: HTTP

- Don't forget about the server!

Advice: Development

- Don't fly blind!
 - Charles Web Debugging Proxy
 - MockWebServer

Advice: Android

- High level libraries exist. Use them.
 - Retrofit, Picasso, Volley

Questions?

squareup.com/careers

