
Volley
Easy, Fast Networking for Android
Ficus Kirkpatrick
Google, Inc.
Google

What is Volley?

volley (\ˈvä-lē\) n.:

the flight of the ball (as in volleyball or tennis) or its course before striking the ground

What is Volley?

volley (\ˈvä-lē\), n.:

a burst or emission of many things or a large amount at once

Everything you need

JSON, images, raw text

Memory and disk caching

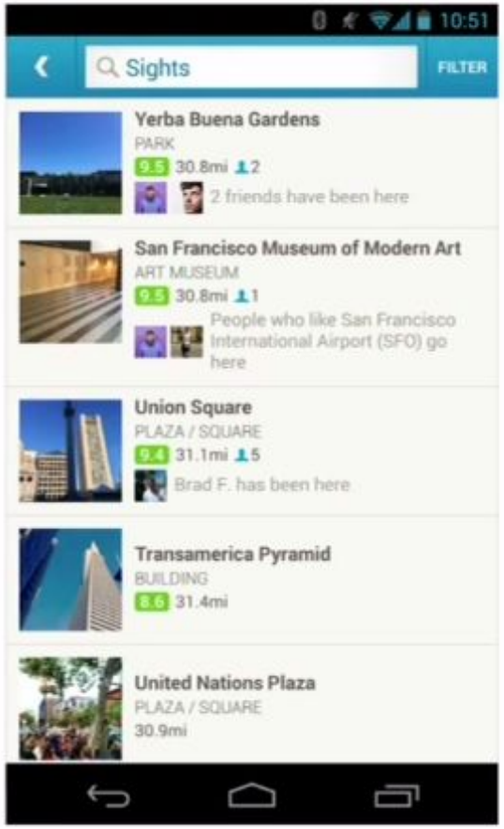
Powerful customization abilities

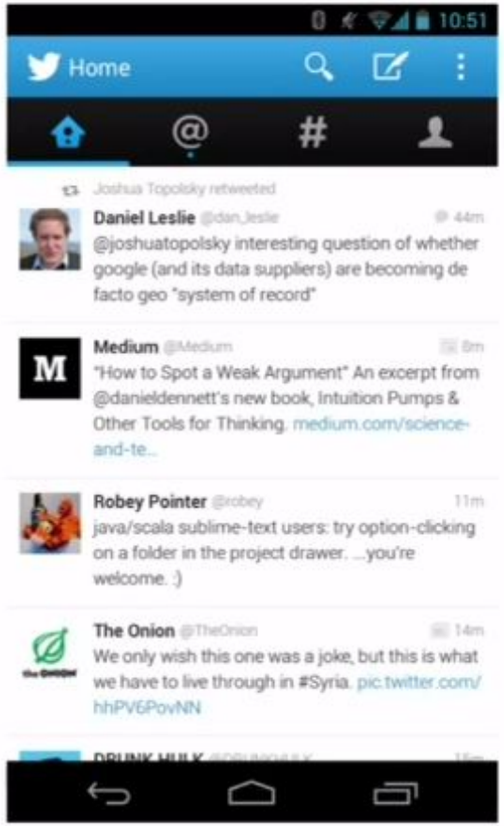
Debugging and tracing tools

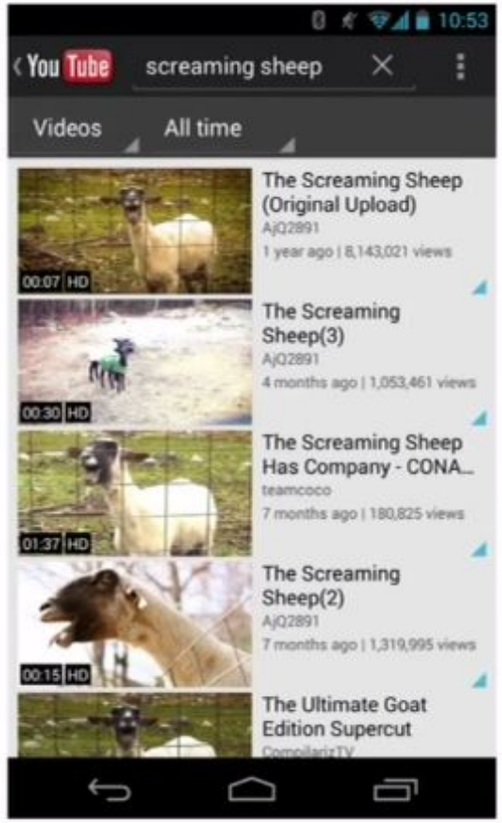
But why?

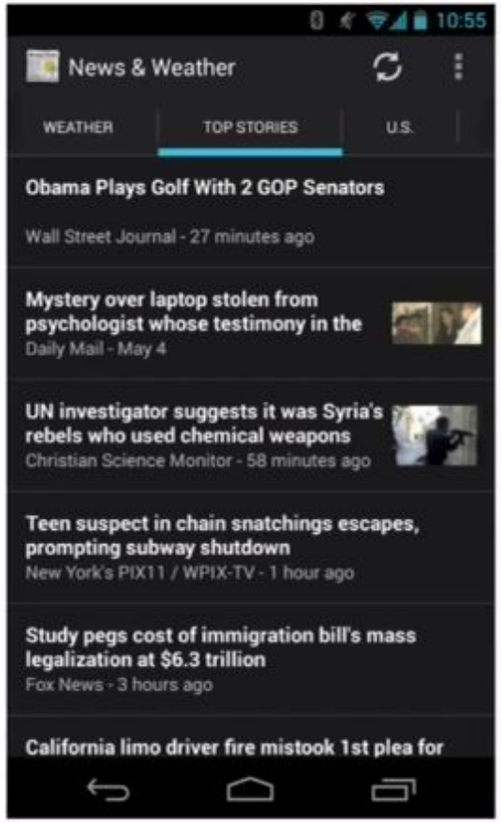
Android already has HTTP client support, right?

What do these all have in common?









Design tradeoffs

Great for RPC-style network operations that populate UI

Fine for background RPCs

Terrible for large payloads

A simple app

Paginated list of strings with thumbnail images

Simple JSON protocol

GET /api/list HTTP/1.1

```
{"items": [
    {
      "title": "Dollar Bill",
      "description": "Please. Mr. Y'all was my father.",
      "image_url": "/static/24.jpg"
    },
    {
      "title": "Tennis Ball",
      "description": "Every dog's favorite.",
      "image_url": "/static/60.jpg"
    },
   ...
 ],
  "next": "10_10" }
```

Simple JSON protocol

GET /api/list HTTP/1.1

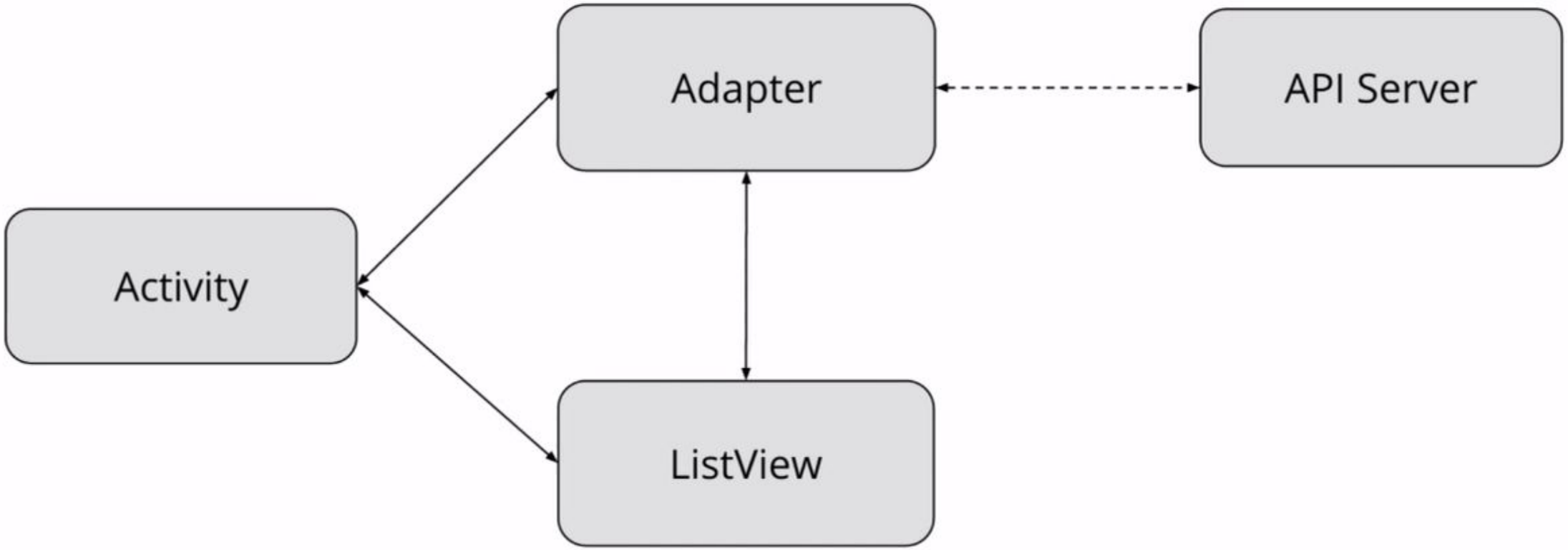
```
{"items": [
    {
      "title": "Dollar Bill",
      "description": "Please. Mr. Y'all was my father.",
      "image_url": "/static/24.jpg"
    },
    {
      "title": "Tennis Ball",
      "description": "Every dog's favorite.",
      "image_url": "/static/60.jpg"
    },
   ...
 ],
  "next": "10_10" }
```

Simple JSON protocol

GET /api/list HTTP/1.1

```
{"items": [
    {
      "title": "Dollar Bill",
      "description": "Please. Mr. Y'all was my father.",
      "image_url": "/static/24.jpg"
    },
    {
      "title": "Tennis Ball",
      "description": "Every dog's favorite.",
      "image_url": "/static/60.jpg"
    },
   ...
 ],
  "next": "10_10" }
```

Application architecture



Typical implementation

Adapter loads data from getView()

Java

```java
@Override
public View getView(int position, View view, ViewGroup parent) {
  // Load more if we're close to the end.
  if (closeToEnd(position) && !mLoading) {
    loadMoreData();
  }

  // Make the views...
```

Typical implementation

loadMoreData()

Java

```java
// private class LoadItemsTask extends AsyncTask<URL, Void, JSONObject> {

protected JSONObject doInBackground(URL... params) {
  HttpURLConnection conn = (HttpURLConnection) params[0].openConnection();
  InputStream input = conn.getInputStream();
  ByteArrayOutputStream baos = new ByteArrayOutputStream();
  copy(input, baos);
  JSONObject jsonRoot = new JSONObject(baos.toString());
  return jsonRoot;
}
```

Typical implementation

loadMoreData()

Java

```java
// private class LoadItemsTask extends AsyncTask<URL, Void, JSONObject> {

protected JSONObject doInBackground(URL... params) {
  HttpURLConnection conn = (HttpURLConnection) params[0].openConnection();
  InputStream input = conn.getInputStream();
  ByteArrayOutputStream baos = new ByteArrayOutputStream();
  copy(input, baos);
  JSONObject jsonRoot = new JSONObject(baos.toString());
  return jsonRoot;
}
```

Typical implementation

loadMoreData()

Java

```
// private class LoadItemsTask extends AsyncTask<URL, Void, JSONObject> {

protected void onPostExecute(JSONObject jsonRoot) {
  List<Items> items = parseJson(jsonRoot);
  appendItemsToList(item);
  notifyDataSetChanged();
}
```

Typical implementation

Back in getView()

Java

```java
@Override
public View getView(int position, View view, ViewGroup parent) {
  // Load more if needed, make ViewHolder, etc.

  mTitleView.setText(item.title);
  mDescriptionView.setText(item.description);

  new LoadImageTask(holder.imageView).execute(
      new URL(BASE_URL + item.imageUrl));
```

Typical implementation

LoadImageTask

Java

```java
// private class LoadImageTask extends AsyncTask<URL, Void, Bitmap> {

public LoadImageTask(ImageView imageView) {
  mImageView = imageView;
}

protected Bitmap doInBackground(URL... params) {
  HttpURLConnection conn = (HttpURLConnection) params[0].openConnection();
  InputStream input = conn.getInputStream();
  return BitmapFactory.decodeStream(input);
}
```

Typical implementation

LoadImageTask

Java

```java
// private class LoadImageTask extends AsyncTask<URL, Void, Bitmap> {

protected void onPostExecute(Bitmap result) {
  mImageView.setImageBitmap(result);
}
```

Problems and solutions

Typical approach vs. Volley approach

Problems

All network requests happen serially

Problems

Rotating the screen will reload everything from the network

Problems

AsyncTasks stomp on recycled views

Problems

Compatibility problems on Froyo

Volley implementation

Setup

Java

```java
// Somewhere common; app startup or adapter constructor

mRequestQueue = Volley.newRequestQueue(context);
mImageLoader = new ImageLoader(mRequestQueue, new BitmapLruCache());
```

Volley implementation

loadMoreData()

Java

```java
mRequestQueue.add(new JsonObjectRequest(Method.GET, url, null,
   new Listener<JSONObject>() {
      public void onResponse(JSONObject jsonRoot) {
         mNextPageToken = jsonGet(jsonRoot, "next", null);
         List<Items> items = parseJson(jsonRoot);
         appendItemsToList(item);
         notifyDataSetChanged();
      }
   }
}
```

Volley implementation

Retrieving images with ImageLoader

Java

```
if (holder.imageRequest != null) {
   holder.imageRequest.cancel();
}

holder.imageRequest = mImageLoader.get(BASE_URL + item.image_url,
        holder.imageView, R.drawable.loading, R.drawable.error);
```

Volley implementation

Using NetworkImageView

XML

```
- <ImageView

+ <com.android.volley.NetworkImageView
```

Java

```
mImageView.setImageUrl(BASE_URL + item.image_url, mImageLoader);
```

Easy to write custom requests

Java

```java
@Override
protected Response<T> parseNetworkResponse(NetworkResponse response) {
    try {
        String json = new String(
                response.data, HttpHeaderParser.parseCharset(response.headers));
        return Response.success(
                gson.fromJson(json, clazz), HttpHeaderParser.parseCacheHeaders(response));
    } catch (UnsupportedEncodingException e) {
        return Response.error(new ParseError(e));
    } catch (JsonSyntaxException e) {
        return Response.error(new ParseError(e));
    }
}
```

https://gist.github.com/ficusk/5474673

Gson implementation

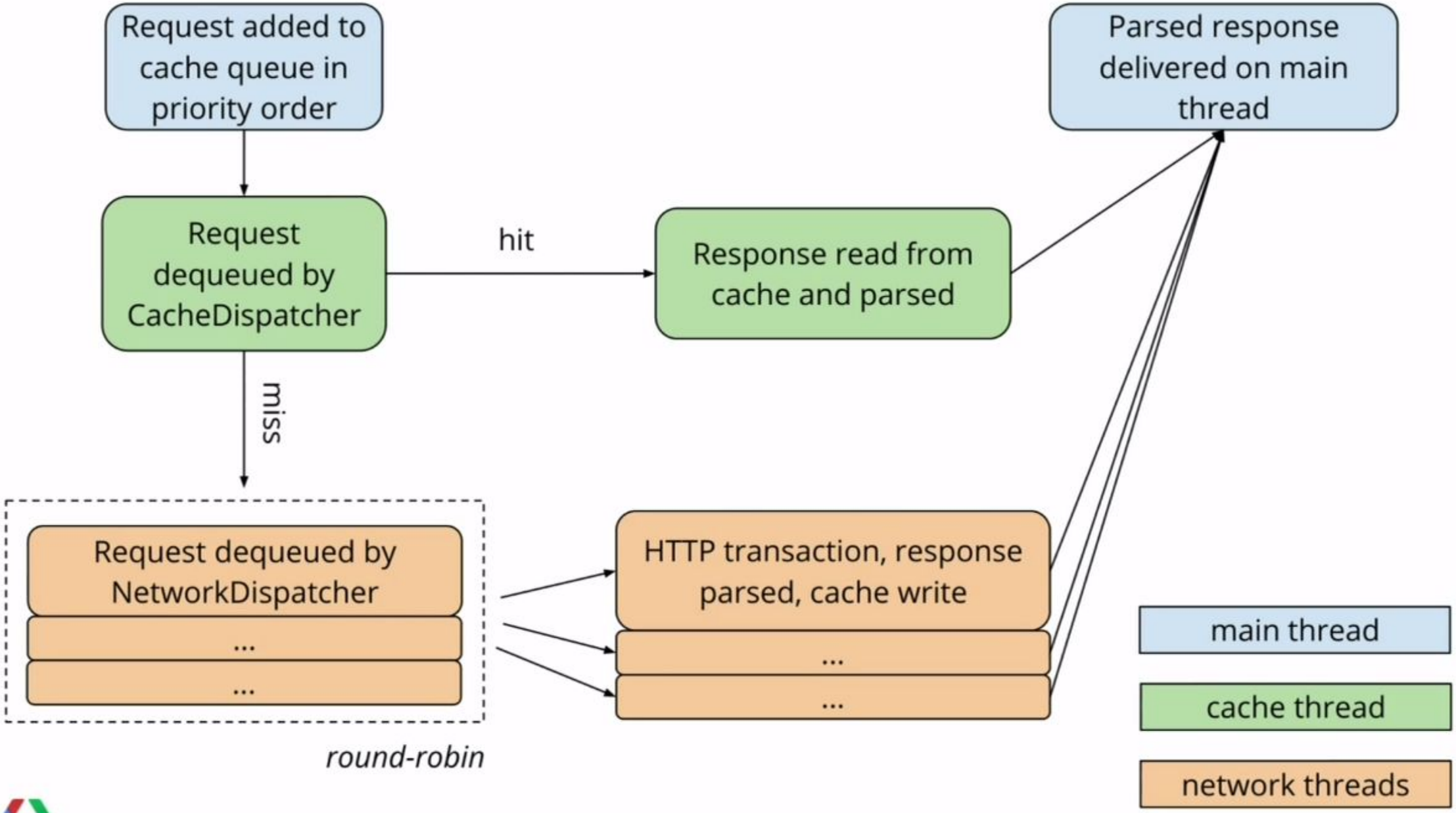
loadMoreData()

Java

```java
mRequestQueue.add(
  new GsonRequest<ListResponse>(url, ListResponse.class, null,
  new Listener<ListResponse>() {
    public void onResponse(ListResponse response) {
      appendItemsToList(response.items);
      notifyDataSetChanged();
    }
  }
}
```

Under the hood

Architecture and semantics


Request added to cache queue in priority order
Parsed response delivered on main thread
Request dequeued by CacheDispatcher
hit
Response read from cache and parsed
miss
Request dequeued by NetworkDispatcher
...
...
HTTP transaction, response parsed, cache write
...
...
round-robin
main thread
cache thread
network threads

Debugging and tracing

```
adb shell setprop log.tag.Volley VERBOSE
```

```
D/Volley  ( 6027): [1] MarkerLog.finish: (443  ms) [ ] http://ficus.me:8080/static/05.jpg LOW 11
D/Volley  ( 6027): [1] MarkerLog.finish: (+0   ) [ 1] add-to-queue
D/Volley  ( 6027): [1] MarkerLog.finish: (+68  ) [15] cache-queue-take
D/Volley  ( 6027): [1] MarkerLog.finish: (+1   ) [15] cache-hit-expired
D/Volley  ( 6027): [1] MarkerLog.finish: (+136 ) [19] network-queue-take
D/Volley  ( 6027): [1] MarkerLog.finish: (+127 ) [19] network-http-complete
D/Volley  ( 6027): [1] MarkerLog.finish: (+101 ) [19] network-parse-complete
D/Volley  ( 6027): [1] MarkerLog.finish: (+9   ) [19] network-cache-written
D/Volley  ( 6027): [1] MarkerLog.finish: (+0   ) [19] post-response
D/Volley  ( 6027): [1] MarkerLog.finish: (+1   ) [ 1] done
```

Debugging and tracing

```
adb shell setprop log.tag.Volley VERBOSE
```

```
D/Volley  ( 6027): [1] MarkerLog.finish: (443  ms) [ ] http://ficus.me:8080/static/05.jpg LOW 11
D/Volley  ( 6027): [1] MarkerLog.finish: (+0   ) [ 1] add-to-queue
D/Volley  ( 6027): [1] MarkerLog.finish: (+68  ) [15] cache-queue-take
D/Volley  ( 6027): [1] MarkerLog.finish: (+1   ) [15] cache-hit-expired
D/Volley  ( 6027): [1] MarkerLog.finish: (+136 ) [19] network-queue-take
D/Volley  ( 6027): [1] MarkerLog.finish: (+127 ) [19] network-http-complete
D/Volley  ( 6027): [1] MarkerLog.finish: (+101 ) [19] network-parse-complete
D/Volley  ( 6027): [1] MarkerLog.finish: (+9   ) [19] network-cache-written
D/Volley  ( 6027): [1] MarkerLog.finish: (+0   ) [19] post-response
D/Volley  ( 6027): [1] MarkerLog.finish: (+1   ) [ 1] done
```

Debugging and tracing

```
adb shell setprop log.tag.Volley VERBOSE
```

```
D/Volley  ( 6027): [1] MarkerLog.finish: (443  ms) [ ] http://ficus.me:8080/static/05.jpg LOW 11
D/Volley  ( 6027): [1] MarkerLog.finish: (+0   ) [ 1] add-to-queue
D/Volley  ( 6027): [1] MarkerLog.finish: (+68  ) [15] cache-queue-take
D/Volley  ( 6027): [1] MarkerLog.finish: (+1   ) [15] cache-hit-expired
D/Volley  ( 6027): [1] MarkerLog.finish: (+136 ) [19] network-queue-take
D/Volley  ( 6027): [1] MarkerLog.finish: (+127 ) [19] network-http-complete
D/Volley  ( 6027): [1] MarkerLog.finish: (+101 ) [19] network-parse-complete
D/Volley  ( 6027): [1] MarkerLog.finish: (+9   ) [19] network-cache-written
D/Volley  ( 6027): [1] MarkerLog.finish: (+0   ) [19] post-response
D/Volley  ( 6027): [1] MarkerLog.finish: (+1   ) [ 1] done
```

Debugging and tracing

```
adb shell setprop log.tag.Volley VERBOSE
```

```
D/Volley  ( 6027): [1] MarkerLog.finish: (443  ms) [ ] http://ficus.me:8080/static/05.jpg LOW 11
D/Volley  ( 6027): [1] MarkerLog.finish: (+0   ) [ 1] add-to-queue
D/Volley  ( 6027): [1] MarkerLog.finish: (+68  ) [15] cache-queue-take
D/Volley  ( 6027): [1] MarkerLog.finish: (+1   ) [15] cache-hit-expired
D/Volley  ( 6027): [1] MarkerLog.finish: (+136 ) [19] network-queue-take
D/Volley  ( 6027): [1] MarkerLog.finish: (+127 ) [19] network-http-complete
D/Volley  ( 6027): [1] MarkerLog.finish: (+101 ) [19] network-parse-complete
D/Volley  ( 6027): [1] MarkerLog.finish: (+9   ) [19] network-cache-written
D/Volley  ( 6027): [1] MarkerLog.finish: (+0   ) [19] post-response
D/Volley  ( 6027): [1] MarkerLog.finish: (+1   ) [ 1] done
```

The main thread

Or, how I learned to stop using *synchronized*

The main thread

Ever written this block of code?

Java

```java
@Override
public void onPostExecute(Result r) {
    if (getActivity() == null) {
        return;
    }

    // ...
```

The main thread

All responses are delivered to the main thread

If you cancel from the main thread, Volley guarantees your response will not be delivered

Java

```java
@Override
public void onStop() {
    for (Request <?> req : mInFlightRequests) {
        req.cancel();
    }
    ...
```

The main thread

All responses are delivered to the main thread

If you cancel from the main thread, Volley guarantees your response will not be delivered

Java

```java
@Override
public void onStop() {
    for (Request <?> req : mInFlightRequests) {
        req.cancel();
    }
    ...
```

The main thread

All responses are delivered to the main thread

If you cancel from the main thread, Volley guarantees your response will not be delivered

Java

```
@Override
public void onStop() {
    mRequestQueue.cancelAll(this);
    ...
```

The main thread

All responses are delivered to the main thread

If you cancel from the main thread, Volley guarantees your response will not be delivered

Java

```java
@Override
public void onStop() {
   mRequestQueue.cancelAll(
       new RequestFilter() { ...
```

Wrapping up

What does it all mean?

How to get started

1. Clone the Volley project

2. Import the code into your project

3. `Volley.newRequestQueue(context)`

```
git clone https://android.googlesource.com/platform/frameworks/volley
```

Thanks!

http://google.com/+FicusKirkpatrick
http://twitter.com/ficus