Volley源码解析

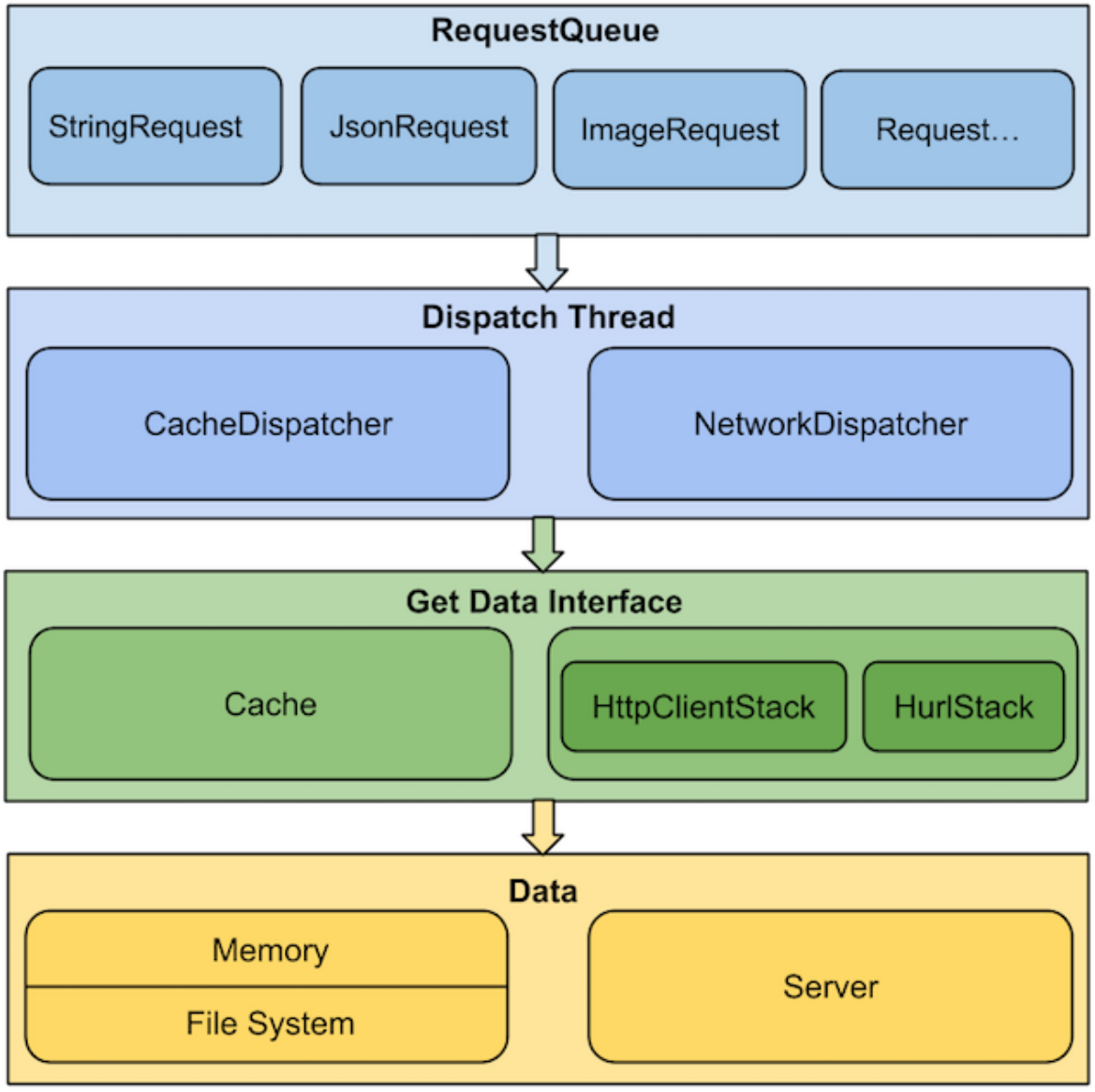
源码解析 Volley

功能介绍

- **简介**
  Volley 是 Google 推出的 Android 异步网络请求框架和图片加载框架，特别是适合`数据量小，通信频繁`的网络操作。
- **特点**
  1. 扩展性强。Volley 中大多是基于接口的设计，可配置性强。
  2. 一定程度符合 Http 规范，包括返回 ResponseCode(2xx、3xx、4xx、5xx) 的处理，请求头的处理，缓存机制的支持等。并支持重试及优先级定义。
  3. 默认 Android2.3 及以上基于 HttpURLConnection，2.3 以下基于 HttpClient 实现，这两者的区别及优劣在4.2.1 Volley中具体介绍。
  4. 提供简便的图片加载工具
     （ ImageLoader,ImageRequest,NetworkImageView ）。

总体设计

总体设计图



从Volley 的总体设计图看，主要是通过两种Dispatch Thread（调度线程）从RequestQueue中取出请求，根据是否已缓存调用Cache或Network这两类数据获取接口之一，从内存缓存或是服务器取得请求的数据，然后交由ResponseDelivery去做结果分发及回调处理。

Volley中的概念

Volley 的调用比较简单，通过 newRequestQueue(…) 函数新建并启动一个请求队列RequestQueue后，只需要往这个RequestQueue不断 add Request 即可。

- Volley：Volley 对外暴露的 API，通过 newRequestQueue(…) 函数新建并启动一个请求队列RequestQueue。
  Request：表示一个请求的抽象类。StringRequest、JsonRequest、ImageRequest 都是它的子类，表示某种类型的请求。

- RequestQueue：表示请求队列，里面包含一个CacheDispatcher(用于处理走缓存请求的调度线程)、NetworkDispatcher数组(用于处理走网络请求的调度线程)，一个ResponseDelivery(返回结果分发接口)，通过 start() 函数启动时会启动CacheDispatcher和NetworkDispatchers。

- CacheDispatcher：一个线程，用于调度处理走缓存的请求。启动后会不断从缓存请求队列中取请求处理，队列为空则等待，请求处理结束则将结果传递给ResponseDelivery去执行后续处理。当结果未缓存过、缓存失效或缓存需要刷新的情况下，该请求都需要重新进入NetworkDispatcher去调度处理。

- NetworkDispatcher：一个线程，用于调度处理走网络的请求。启动后会不断从网络请求队列中取请求处理，队列为空则等待，请求处理结束则将结果传递给ResponseDelivery去执行后续处理，并判断结果是否要进行缓存。

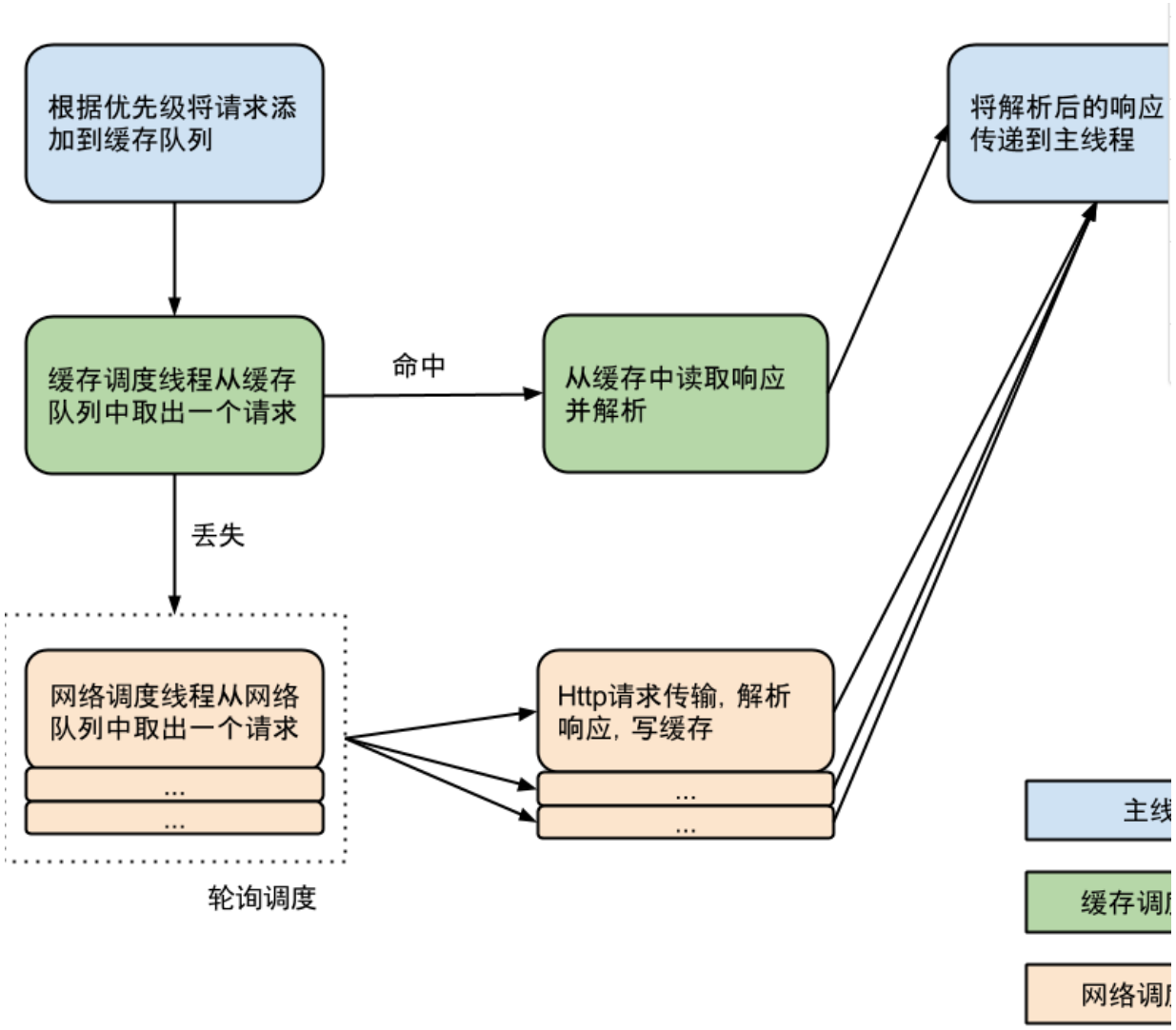
- ResponseDelivery：返回结果分发接口，目前只有基于ExecutorDelivery的在入参 handler 对应线程内进行分发。

- HttpStack：处理 Http 请求，返回请求结果。目前 Volley 中有基于HttpURLConnection 的HurlStack和 基于 Apache HttpClient 的HttpClientStack。

- Network：调用HttpStack处理请求，并将结果转换为可被ResponseDelivery处理的NetworkResponse。

- Cache：缓存请求结果，Volley 默认使用的是基于 sdcard 的DiskBasedCache。NetworkDispatcher得到请求结果后判断是否需要存储在 Cache，CacheDispatcher会从 Cache 中取缓存结果。

Volley请求流程图



由图可以看出当请求发出时，先有缓存调度线程来查找缓存队列，若命中则直接从缓存中获取响应并解析最终将响应传递到主线程；若在缓存队列中没有命中，则由网络调度线程从网络来获取请求，然后通过HttpStack来处理请求传输、解析、响应和写缓存，最终将解析后的响应传递到主线程。

HttpURLConnection 和 AndroidHttpClient(HttpClient 的封装)如何选择及原因：

在 Froyo(2.2) 之前，HttpURLConnection 有个重大 Bug，调用 close() 函数会影响连接池，导致连接复用失效，所以在 Froyo 之前使用 HttpURLConnection 需要关闭 keepAlive。

另外在 Gingerbread(2.3) HttpURLConnection 默认开启了 gzip 压缩，提高了 HTTPS 的性能，Ice Cream Sandwich(4.0) HttpURLConnection 支持了请求结果缓存。

再加上 HttpURLConnection 本身 API 相对简单，所以对 Android 来说，在 2.3 之后建议使用 HttpURLConnection，之前建议使用 AndroidHttpClient。