Національний технічний університет України
“Київський політехнічний інститут ім І. Сікорського”

Звіт

з лабораторноі роботи №5
на тему “Робота з мережею”

Виконав: студент групи
ІС-42 Коростильов Євгеній

Київ 2017 р

Принципи роботи з мережею

Багато речей є в сучасних додатках відбувається через підключення до мережі. Тому це є не незмінною частиною багатьох додатків. Але є деякі моменти які слід пам'ятати під час проектування мережевої частини додатку.

- Перше - це трафік. Не завжди є можливість працювати з безкоштовним Wi-Fi-з'єднанням, а мобільний інтернет все ще дорогий, і про це потрібно пам'ятати, тому що трафік - це гроші користувача.
- Друге - це ліміт батарейки. Мобільні пристрої необхідні користувачеві для якихось повсякденних справ, нарад, прогулянок, бізнесу, і коли батарейка сідає в самий невідповідний момент, користувач обурюється.
- Третє - це безпека. Так як все-таки мова йде про мобільні клієнтів, і дані гуляють по мережі від клієнта до сервера і назад, то їх необхідно захищати.

Підходи щодо реалізації мережевої взаємодії

Існують різні підходи дореалізації взаємодії. Наведемо деякі з них.

Перший підхід - на основі сокетів. Він часто використовується в додатках, де важлива швидкість доставки повідомлення, важливий порядок доставки повідомлень і необхідно тримати стабільне з'єднання з сервером. Такий спосіб часто реалізується в месенджерах і іграх.

Другий підхід - це часті опитування (polling): клієнт посилає запит на сервер і каже йому: «Дай мені свіжі дані»; сервер відповідає на запит клієнта і віддає все, що у нього накопичилося до цього моменту. Мінус такого підходу в тому, що клієнт не

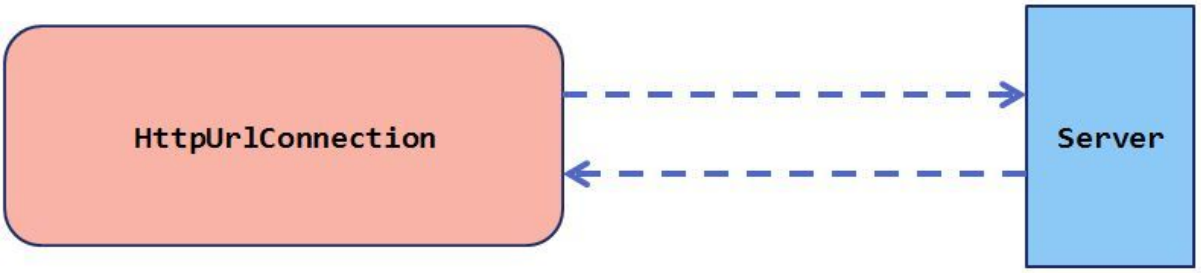
знає, чи з'явилися свіжі дані на сервері. За мережі зайвий раз ганяється трафік, в першу чергу через часті установок з'єднань з сервером.

Третій підхід - довгі опитування (long polling) - полягає в тому, що клієнт посилає «очікує» запит на сервер. Сервер дивиться, чи є свіжі дані для клієнта, якщо їх немає, то він тримає з'єднання з клієнтом до тих пір, поки ці дані не з'являться. Як тільки дані з'явилися, він «пушіт» їх назад клієнтові. Клієнт, отримавши дані від сервера, тут же посилає наступний «очікує» запит і т.д. Реалізація цього підходу досить складна на мобільному клієнті в першу чергу через нестабільність мобільного з'єднання. Зате при цьому підході трафіку витрачається менше, ніж при звичайному polling'е, тому що скорочується кількість установок з'єднань з сервером.

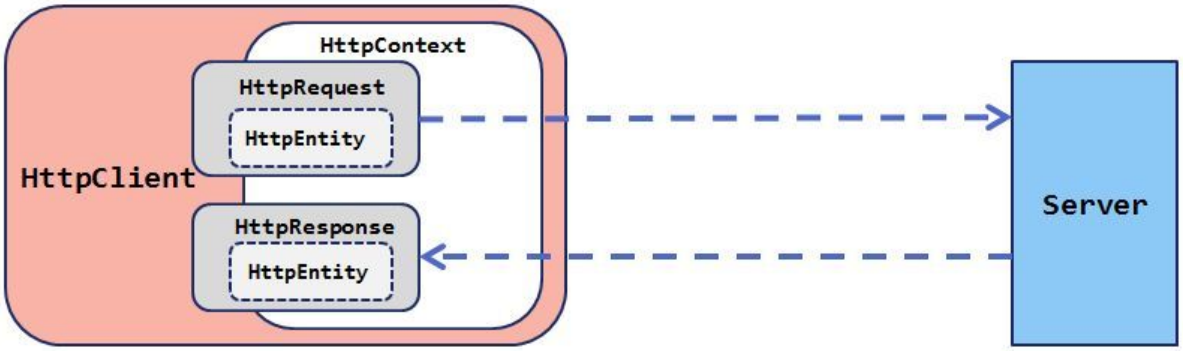
Методи роботи з мережею

В Android-розробника є два класи для роботи з цими протоколами. Перший - це java.net.HttpURLConnection, другий - org.apache.http.client.HttpClient. Обидві ці бібліотеки включені в Android SDK.

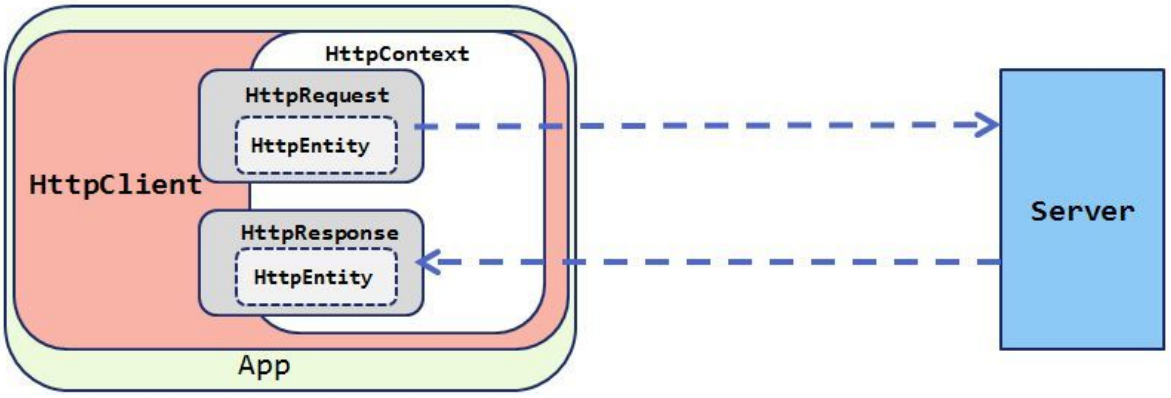
В HttpURLConnection один клас і все. Це пояснюється тим, що батьківський клас URLConnection був спроектований для роботи не тільки по HTTP-протоколу, а ще по таким, як file, mailto, ftp і т.п.



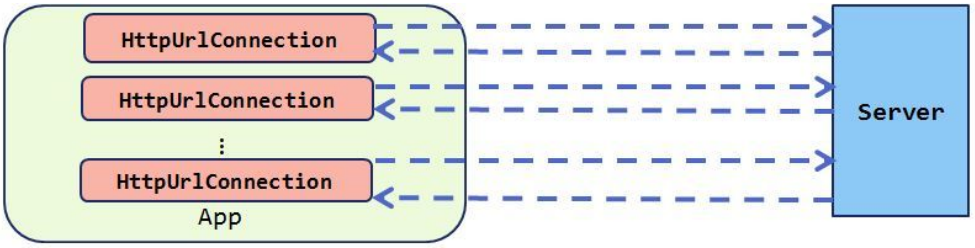
HttpClient спроектований більш об'єктно-орієнтовано. У ньому є чіткий поділ абстракцій. У найпростішому випадку ми будемо працювати з п'ятьма різними інтерфейсами: HttpRequest, HttpResponse, HttpEntity і HttpContext. Тому апачевскій клієнт набагато великовагові HttpUrlConnection.



Як правило, на все додаток існує всього один екземпляр класу HttpClient. Це обумовлено його ваговитістю. Використання окремого примірника на кожен запит буде марнотратним. Ми можемо, наприклад, зберігати примірник HTTP-клієнта в спадкоємця класу Application.



У разі HttpUrlConnection слід створювати на кожен запит новий екземпляр клієнта.



Висновки

В цій лабораторній роботі я розібрався з роботою додатку через інтернет-мережу. Після обробки великого об’єму даних стало зрозуміло, що розробники платформи Android рекомендують в нових додатках використовувати HttpUrlConnection, тому що він простий у використанні, його розвиватимуть далі і адаптувати під платформу. HttpClient слід використовувати на платформах нижче Android 2.3, в першу чергу через серйозне бага з keep alive-з'єднанням.