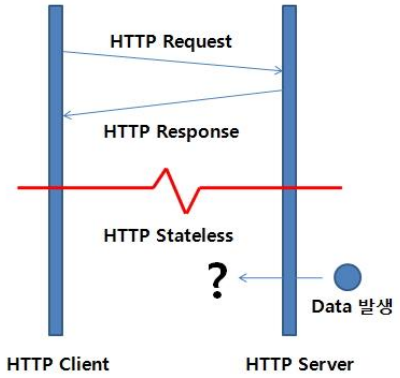
26장. 실시간 서버 푸시

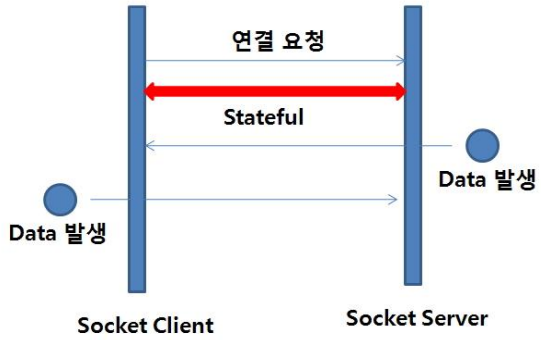
26.1 소켓 프로그램

26.1.1. 소켓 프로그램 작성 방법

- HTTP 통신



- 소켓 프로그램



26.1 소켓 프로그램

- 안드로이드에서 소켓 프로그램은 HTTP와 마찬가지로 Java API를 그대로 이용하여 구현

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

- 서버와 소켓 연결

```
Socket socket = new Socket();
SocketAddress remoteAddr = new InetSocketAddress(serverIp, serverPort);
socket.connect(remoteAddr, 10 * 1000);
```

- 데이터를 송수신 IO 객체를 생성

```
BufferedOutputStream bout = new BufferedOutputStream(socket.getOutputStream());
BufferedInputStream bin = new BufferedInputStream(socket.getInputStream());
```

- 데이터를 송신

```
bout.write(((String) msg.obj).getBytes());
bout.flush();
```

26.1 소켓 프로그램

- 데이터를 수신

```
String message = null;
int size = bin.read(buffer);
if (size > 0) {
    message = new String(buffer, 0, size, "utf-8");
}
```

26.1.2. 소켓 프로그램 작성 시 주의점

- Connection, Read, Write 모두 스레드로 처리
- Read 행위를 스레드로 처리

```
class ReadThread extends Thread {
    @Override
    public void run() {
        byte[] buffer = null;
        while (flagRead) {
            buffer = new byte[1024];
            try {
                String message = null;
                int size = bin.read(buffer);
                if (size > 0) {
                    //read 후 업무처리
                }
            } catch (IOException e) {
            }
        }
    }
}
```

26.1 소켓 프로그램

- 서버 연결 부분을 스레드로 처리

```
class SocketThread extends Thread {
    public void run() {
        while (flagConnection) {
            try {
                if (!isConnected) {
                    socket = new Socket();
                    SocketAddress remoteAddr = new InetSocketAddress(serverIp,
serverPort);
                    socket.connect(remoteAddr, 10 * 1000);
                    //...
                } else {
                    SystemClock.sleep(10000);
                }
            } catch (Exception e) {
            }
        }
    }
}
```

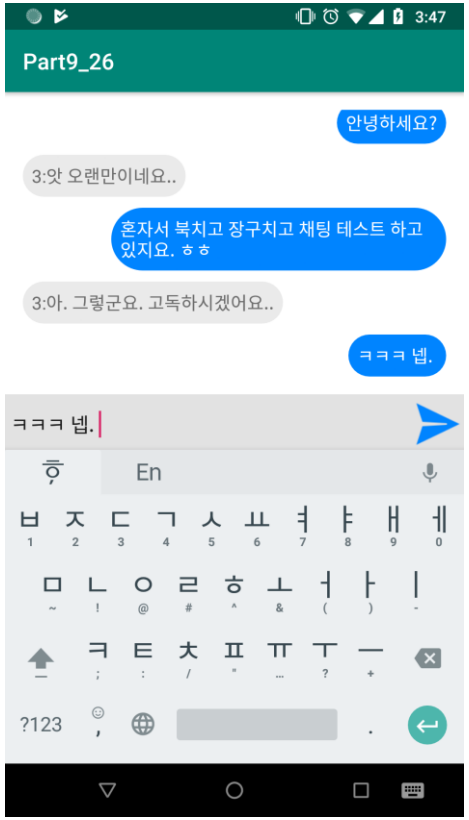
26.1 소켓 프로그램

- Write 스레드

```
class WriteThread extends Thread {
    @Override
    public void run() {
        Looper.prepare();
        writeHandler = new Handler(){
            @Override
            public void handleMessage(Message msg) {
                try {
                    bout.write(((String) msg.obj).getBytes());
                    bout.flush();
                    //...                }catch (Exception e){                }
            }
        };
        Looper.loop();
    }
}
```

Step by Step 26-1 – 소캣 통신

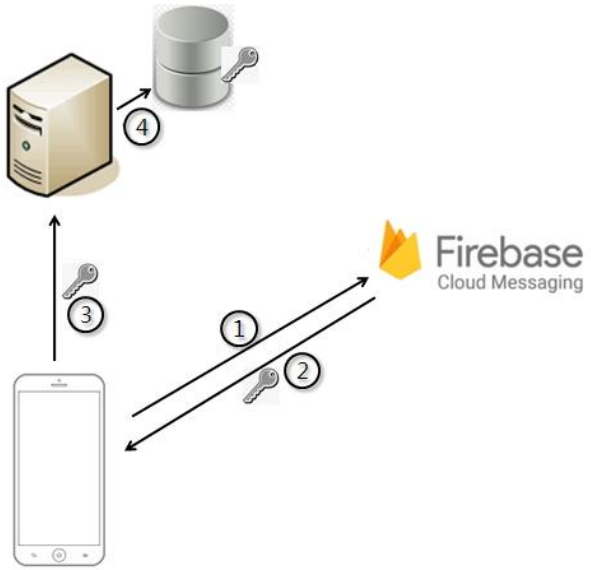
- 모듈 생성
- 파일 복사
- AndroidManifest.xml 작업
- MainActivity 작성
- 서버 준비
- 실행



26.2 Firebase Cloud Message

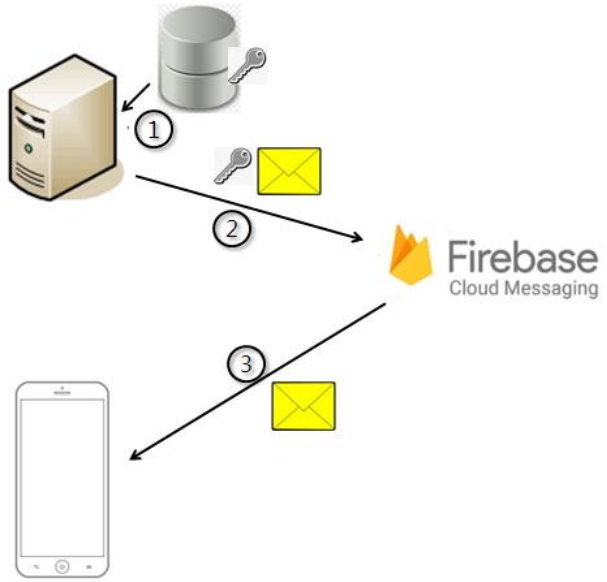
- Firebase는 구글의 모바일 앱 개발 통합 플랫폼이며, Firebase에서 제공하는 여러 가지 서비스 중 하나가 FCM
- FCM은 2016년 구글 I/O 행사에서 Firebase를 개선하여 GCM(Google Cloud Message)을 대체하기 위해 선보인 서비스

- FCM의 동작 원리
- 첫 번째 단계가 앱을 위한 키를 FCM 서버를 통해 얻는 단계



26.2 Firebase Cloud Message

- 두 번째 단계로 서버에서 데이터를 스마트폰에 전달



Step by Step 26-2 – Firebase 클라우드 메시지

- SHA1 지문 획득
- Firebase 콘솔 작업
- 그레이들 설정
- FirebaseMessagingService 작성
- FirebaseInstanceIDService 작성
- 앱 실행
- Node.js 로 FCM 데이터 전달