Android Push Notification System

Project Guide v0.4

2010.10

목 차

1 프로젝트 개요

1.1 프로젝트 목적

본 프로젝트는 안드로이드 개발자 커뮤니티에서 많은 필요성과 다양한 분야에 응용될 수 있는 안드로이드 기반의 푸시 통보 (Push Notification) 서비스를 위한 서버 시스템 및 클라이언트 솔루션을 오픈소스로 제공하여, 국내외 안드로이드 애플리케이션 개발자 커뮤니티 및 공개 SW 커뮤니티 활성화에 기여를 하고자 한다.

본 프로젝트는 다음과 같은 주요 모듈을 제공한다.

- 푸시 통보 서버 시스템
- 안드로이드 푸시 통보 클라이언트 모듈
- 안드로이드 샘플 애플리케이션

1.2 프로젝트 개발 배경

안드로이드 기반의 애플리케이션이 급격히 증가하고 있지만, 현재 Google에서 정식적인 안드로이드 기반 푸시 통보 서비스를 지원을 하고 있지 않은 상황이다. 안드로이드 2.2 (Froyo)부터 등장한 Cloud To Device Messaging (C2DM) 서비스는 현재 Lab 테스트 버전으로 정식 서비스가 되고 있지 않은 상황이며, 이전 안드로이드 OS 버전을 지원하지 않는다.
이에 반해 BlackBerry는 2002년 Push Email 서비스을 시작으로 푸시 통보 서비스를 시작했으며, Apple도 2009년 iPhone 3.0 이후부터 Apple Push Notification Service (APNS)를 지원하고 있다. 또한 Microsoft도 출시 예정인 Windows Phone 7에서 Microsoft Push Notification Service (MPNS) 지원과 함께 출시 예정이다.

2 안드로이드 푸시 통보 시스템

2.1 시스템 구성

안드로이드 푸시 통보 시스템은 XMPP 프로토콜을 지원하는 소켓 서버 시스템과 안드로이드 기반 클라이언트 애플리케이션으로 구성된다.

서버	Java 기반 소켓 서버로 XMPP 프로토콜을 지원하며, 관리자용 웹 콘솔 지원을 포함하고 있다. 관리자 웹 콘솔을 이용하여 클라이언트 접속 정보 모니터링 및 Push 메시지를 전송할 수 있다.
클라이언트	안드로이드 기반 애플리케이션으로 서버로부터 Push된 메시지를 수신하여 사용자에게 통보한다.

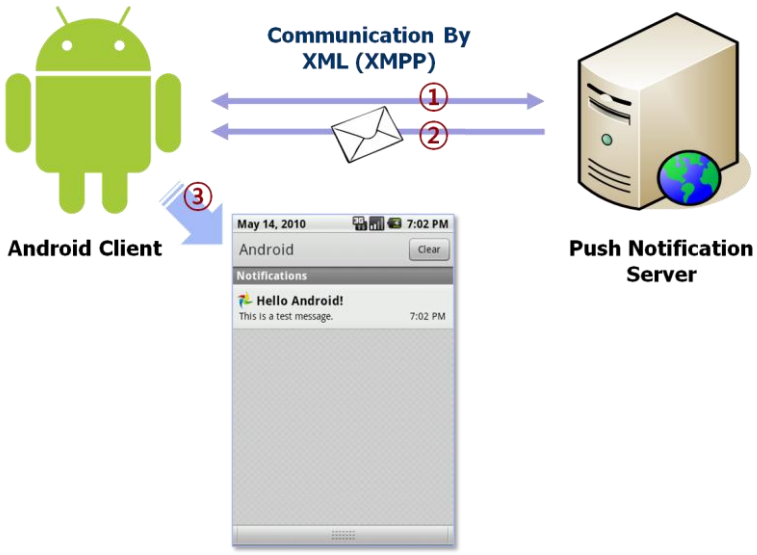
2.2 푸시 통보 과정

안드로이드 푸시 통보 시스템의 기본적인 푸시 통보 과정은 아래 그림과 같다.

① 클라이언트는 푸시 통보 서버에 접속하여, 단말 기본 정보를 서버에 등록하고 연결 상태를 유지한다.

② 서버는 메시징 이벤트가 발생한 경우, 클라이언트에 Push 방식으로 메시지를 전송한다.

③ 클라이언트는 서버로부터 전송된 메시지를 사용자에게 통보를 한다.



3 프로젝트 구성

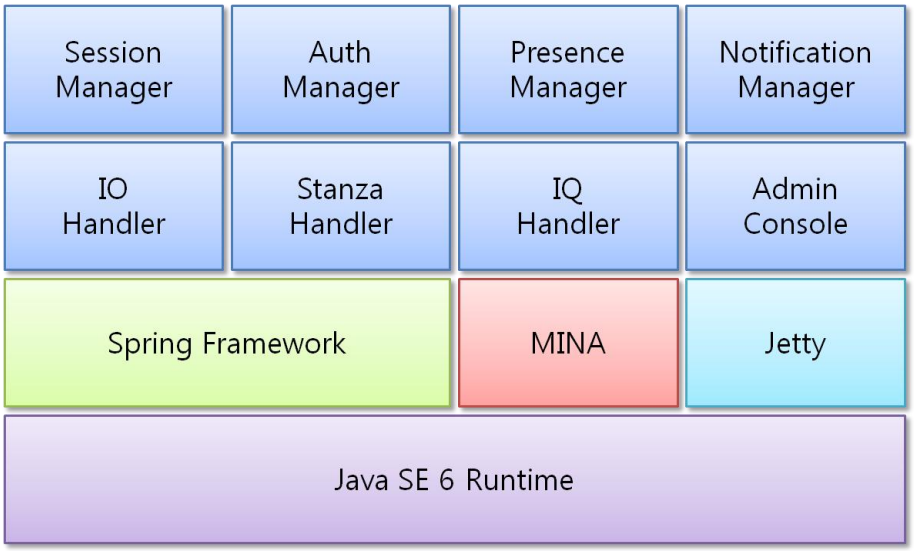
서버와 클라이언트 모듈은 모두 이클립스 기반의 자바 프로젝트이다.

구분	프로젝트명	설명	오픈소스 라이선스
서버	androidpn-server	Java 기반 소켓서버 형태의 푸시 통보 서버 애플리케이션	GPL 2.0
클라이언트	androidpn-client	안드로이드 플랫폼용 푸시통보 클라이언트 라이브러리 배포 모듈	Apache License 2.0
	androidpn-demoapp	개발자용 샘플 애플리케이션	

3.1 서버 (androidpn-server)

푸시 통보 서버는 구동을 위해 Java SE 6 런타임을 필요로 하며, Spring 프레임워크를 이용해 클래스간 의존성 주입을 하고 MINA 프레임워크를 이용하여 구성한 소켓 서버이다. 데이터 통신 방식은 XML 스트리밍 방식의 XMPP 프로토콜을 사용하고 있으며, XMPP 서버 스펙을 구현하고 있다. 또한 Jetty 서블릿 컨테이너를 내장하여 관리자 웹콘솔 애플리케이션을 포함하고 있다.

3.1.1 서버 아키텍쳐



3.1.2 서버 주요 모듈

- **IO Handler**
  - XML 포맷의 스트리밍 바이너리 데이터를 핸들링
- **Stanza Handler**
  - XMPP 기본 통신 단위인 Stanza 및 클라이언트와의 연결 특징을 핸들링
- **IQ Handler**
  - IQ Stanza 데이터를 해석하고 타입별 데이터를 핸들링
- **Session Manager**
  - 클라이언트 연결 세션들을 전반적으로 관리
- **Auth Manager**
  - 클라이언트 사용자 등록 및 로그인 등 인증 관리
- **Presence Manager**
  - 클라이언트 연결 상태(Presence) 정보를 관리
- **Notification Manager**
  - 서버 이벤트에 따라 클라이언트로 Push 메시지 전송을 담당
- **Admin Console**
  - 클라이언트의 등록 정보, 접속 상태 모니터링 및 Push 메시지 전송 UI를 가진 관리자용 웹애플리케이션 모듈

3.1.3 서버 개발 플랫폼 및 도구

- **Java SE 6**
  - 자바 서버 애플리케이션 구동을 위한 런타임 엔진
- **Spring Framework**
  - 자바 클래스간 의존성을 연결해주는 DI (Dependency Injection) 컨테이너 프레임워크

- **MINA**
  - 자바 NIO 네트워크 서버 구현을 위한 네크워크 프레임워크
- **Jetty**
  - 관리자 웹콘솔 모듈을 위한 자바 내장형 서블릿 컨테이너
- **Hibernate**
  - 데이터 영속성을 위한 ORM (Object-Relational Mapping) 프레임워크
- **HSQLDB**
  - 사용자 데이터 저장을 위한 자바 내장형 DB (MySQL 등 다른 DB도 사용 가능)
- **Ant / Maven**
  - 프로젝트 소스 컴파일 및 빌드 도구
- **JUnit**
  - 자바 클래스 단위 테스트 프레임워크

3.1.4 디렉토리 및 자바 패키지 구성

- **프로젝트 디렉토리**

서버 프로젝트 소스는 Maven 2.0 기반의 멀티 모듈 프로젝트이며 다음과 같은 3개의 주요 모듈로 나눌 수 있으며, 디렉토리명과 동일하다.

디렉토리명	설명	빌드 파일
console	관리자 콘솔 모듈이며 웹애플리케이션이며, 스프링 MVC 기반의 구조로 되어있다.	console.war
Server	푸시 통보 서버 핵심 모듈이며, Spring + MINA 통합된 소켓 서버이며, XMPP 서버를 구현하고 있다.	androidpn-server-0.4.x.jar
starter	서버의 시작시키기 위한 모듈로, 클래스 로딩을 담당한다.	starter.jar

- **자바 패키지**

서버 프로젝트의 자바 클래스는 다음과 같은 주요 패키지로 구성되어 있다.

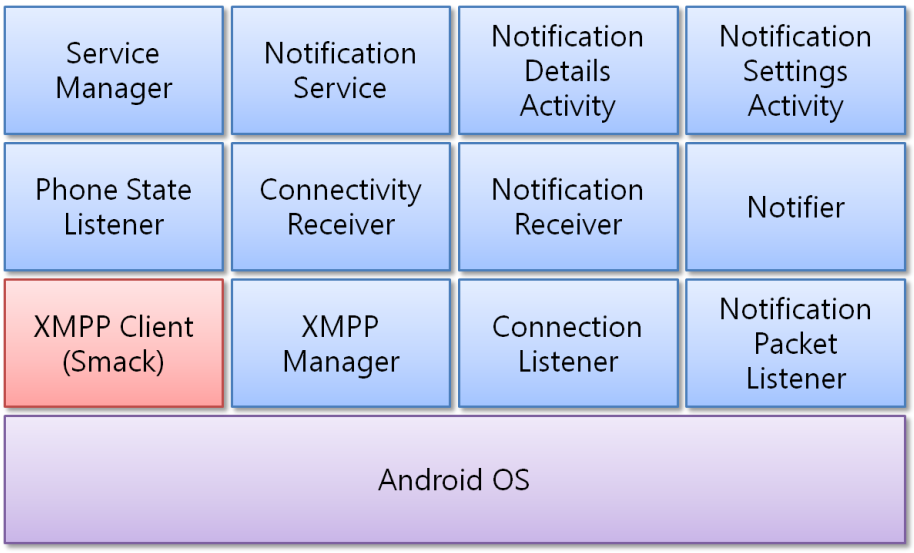
패키지명	설명
org.androidpn.server.container	관리자 콘솔 웹모듈을 시작하기 위한 컨테이너 클래스
org.androidpn.server.dao	Data Access Object 인터페이스
org.androidpn.server.dao.hibernate	Data Access Object 구현 클래스
org.androidpn.server.model	DB 엔티티를 표현하는 도메인 객체 모델 클래스
org.androidpn.server.service	비즈니스 로직을 담당하는 서비스 인터페이스
org.androidpn.server.service.impl	비즈니스 로직을 담당하는 서비스 구현 클래스
org.androidpn.server.starter	서버 시작을 위한 클래스
org.androidpn.server.util	유틸리티 클래스
org.androidpn.server.xmpp	XMPP 서버를 구현한 클래스를 담은 베이스 패키지
org.androidpn.server.xmpp.auth	사용자 인증을 담당하는 클래스
org.androidpn.server.xmpp.codec	MINA 기반의 XMPP 메시지 코덱 클래스

org.androidpn.server.xmpp.handler	XMPP Stanza를 처리하기 위한 각종 핸들러
org.androidpn.server.xmpp.net	MINA 기반의 핵심 소켓 네트워킹을 담당하는 클래스
org.androidpn.server.xmpp.presence	XMPP의 사용자 상태정보(Presence)를 처리하는 클래스
org.androidpn.server.xmpp.push	클라이언트로 푸시 메시지를 전송을 처리하는 클래스
org.androidpn.server.xmpp.router	수신된 Stanza 데이터를 적절한 핸들러로 보내는 클래스
org.androidpn.server.xmpp.session	클라이언트 연결 세션을 관리하기 위한 클래스

3.2 클라이언트 (androidpn-client)

푸시 통보 클라이언트 모듈을 안드로이드 OS 1.5 이상의 플랫폼을 필요로 하며, XMPP 클라이언트 형태로서 서버와 XML 스트리밍 기반 통신을 한다.

3.2.1 클라이언트 아키텍쳐



3.2.2 클라이언트 주요 모듈

- **Service Manager**
  - 클라이언트 환경 설정 로딩 및 통지 서비스를 시작/종료를 담당하는 기본 관리자 클래스
- **Notification Service**
  - 서버로부터 메시지 수신을 위해 사용자 프로그램이 종료되어도 백그라운드로 수행되는 안드로이드 Service를 상속한 서비스
- **XMPP Manager**
  - XMPP 클라이언트 모듈로 푸시 통보 서버로의 연결, 사용자 등록, 로그인 등을 담당
- **Connection Listener**
  - 클라이언트와 서버간 연결의 지속성을 유지하기 위한 연결 상태를 모니터링
- **Notification Packet Listener**
  - 서버로부터의 Notification 패킷을 수신을 담당하고, 통지를 위한 브로드캐스트 수행

- **Notification Receiver**
  - 서버로부터 수신 후 브로드캐스트된 Notification 메시지 정보를 받아 처리
- **Notifier**
  - Notification Manager를 이용하여 수신된 메시지를 사용자에게 통보
- **Connectivity Receiver**
  - 클라이언트와 서버간 연결의 지속성을 위해 네트워크 상태를 모니터링
- **Phone State Listener**
  - 클라이언트와 서버간 연결의 지속성을 위해 전화기 데이터 연결 상태를 모니터링
- **Notification Details Activity**
  - 사용자에게 통보된 메시지의 상세 정보 뷰를 위한 Activity
- **Notification Settings Activity**
  - 클라이언트의 메시지 수신, 소리, 진동 여부 등의 설정 뷰를 제공하는 Activity

3.2.3 클라이언트 개발 플랫폼 및 도구

- **Android SDK**
  - 안드로이드 애플리케이션 구동 플랫폼 (안드로이드 OS 1.5 이상 지원)
  - Service, NotificaitonManager, BroadcastReceiver 등
- **Smack**
  - XMPP 클라이언트 라이브러리
- **Ant**
  - 프로젝트 소스 컴파일 및 빌드 도구

3.2.4 디렉토리 및 자바 패키지 구성

- **프로젝트 디렉토리**

클라이언트 프로젝트는 이클립스 기반의 안드로이드 프로젝트로 기본 디렉토리 구조와 동일하다.

디렉토리명	설명
src	클라이언트 라이브러리 소스와 데모 애플리케이션 소스 포함
lib	프로젝트 참조용 라이브러리로 XMPP 클라이언트 smack 라이브러리 포함
res	안드로이드 리소스 파일들로 클라이언트 설정파일인 android.properties 도 포함
Assets	안드로이드 assets 디렉토리 (사용되지 않음)

- **자바 패키지**

클라이언트 프로젝트의 자바 클래스는 다음과 같은 주요 패키지로 구성되어 있다.

패키지명	설명
org.androidpn.client	푸시 통보 클라이언트 클래스. XMPP 클라이언트 및 안드로이드 Service, Activity 등을 포함
org.androidpn.demoapp	데모용 애플리케이션 클래스

3.3 데모 애플리케이션 (androidpn-demoapp)

푸시 통보 클라이언트 모듈을 이용하여 안드로이드 클라이언트 애플리케이션을 개발을 위한 가이드용 샘플 안드로이드 애플리케이션이다. 클라이언트 프로젝트에서 생성된 라이브러리 모듈(androidpn-client.jar)을 포함하고 있다.

4 프로젝트 컴파일 및 빌드

4.1 서버 컴파일 및 빌드

서버 애플리케이션 소스를 컴파일하기 위해서는 JDK 1.5 이상과 Maven 2.0을 필요로 한다. 다음과 같은 절차로 수행한다.

① 프로젝트 서버 소스 배포 파일(androidpn-server-0.4.x.zip 형태)을 준비하고, 압축을 푼다.

② 쉘 프로프트에서 **mvn install** 명령을 수행하여 기본 모듈들을 컴파일, 테스트 및 빌드한다. 다음과 유사한 결과를 볼 수 있다.

```
C:₩projects₩androidpn-server-0.4.5>mvn install
[INFO] Scanning for projects...
[INFO] Reactor build order:
[INFO]   Andorid Push Notification
[INFO]   Unnamed - org.androidpn:server:jar:0.4.5
[INFO]   Unnamed - org.androidpn:starter:jar:0.4.5
[INFO]   Unnamed - org.androidpn:console:war:0.4.5
[INFO] ------------------------------------------------------------------------
... 중략 ...
 [INFO] Reactor Summary:
[INFO] ------------------------------------------------------------------------
[INFO] Andorid Push Notification ..... SUCCESS [1.562s]
[INFO] Unnamed - org.androidpn:server:jar:0.4.5 ..... SUCCESS [9.109s]
[INFO] Unnamed - org.androidpn:starter:jar:0.4.5 ..... SUCCESS [0.812s]
[INFO] Unnamed - org.androidpn:console:war:0.4.5 ..... SUCCESS [2.109s]
[INFO] ------------------------------------------------------------------------
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESSFUL
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 13 seconds
[INFO] Finished at: Wed Oct 13 02:15:18 KST 2010
[INFO] Final Memory: 47M/114M
[INFO] ------------------------------------------------------------------------
```

③ 쉘 프로프트에서 **mvn assembly:assembly** 명령을 수행하여, 서버 프로그램 바이너리 및 소스 배포 파일을 생성한다.

```
C:₩projects₩androidpn-server-0.4.5>mvn assembly:assembly
[INFO] Scanning for projects...
[INFO] Reactor build order:
[INFO]   Andorid Push Notification
[INFO]   Unnamed - org.androidpn:server:jar:0.4.5
[INFO]   Unnamed - org.androidpn:starter:jar:0.4.5
[INFO]   Unnamed - org.androidpn:console:war:0.4.5
[INFO] Searching repository for plugin with prefix: 'assembly'.
[INFO] ------------------------------------------------------------------------
 ... 중략 ...
[INFO] Building zip: C:₩projects₩androidpn-server-0.4.5₩target₩androidpn-server-0.4.5-bin.zip
[INFO] Building zip: C:₩projects₩androidpn-server-0.4.5₩target₩androidpn-server-0.4.5-src.zip
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESSFUL
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 14 seconds
[INFO] Finished at: Wed Oct 13 02:23:30 KST 2010
[INFO] Final Memory: 27M/64M
[INFO] ------------------------------------------------------------------------
```

④ 정상적으로 수행이 되면 target 디렉토리에 2개의 zip 파일이 생성된다. 이중 서버 바이너리 실행 프로그램은 **androidpn-server-0.4.x-bin.zip** 형태로 빌드된 파일이다.

4.2 클라이언트 컴파일 및 빌드

클라이언트 애플리케이션 소스를 컴파일하기 위해서는 JDK 1.5 이상과 Android 1.5 이상 SDK 라이브러리 및 빌드를 위해서 Ant 1.7 이상을 필요로 한다. 다음과 같은 절차로 수행한다.

① 프로젝트 클라이언트 소스 배포 파일(androidpn-client-0.4.x.zip 형태)을 준비하고, 압축을 푼다.

② Ant 빌드시 사용되는 build.properties 파일을 열어, Android SDK 라이브러리의 위치를 환경에 맞게 수정해준다.

```
# Android SDK
android.sdk.jar=c:/Android/android-sdk/platforms/android-3/android.jar
```

③ 쉘 프로프트에서 **ant build** 명령을 수행하여, 클라이언트 바이너리 라이브러리 및 소스 배포 파일을 생성한다.

```
C:₩projects₩androidpn-client-0.4.5>ant build
Buildfile: build.xml

clean:

compile:
```

```
    [mkdir] Created dir: C:₩projects₩androidpn-client-0.4.5₩target₩classes
    [javac] Compiling 16 source files to C:₩projects₩androidpn-client-0.4.5₩target₩classes

build:
      [jar] Building jar: C:₩projects₩androidpn-client-0.4.5₩target₩androidpn-client-0.4.5.jar
    [mkdir] Created dir: C:₩projects₩androidpn-client-0.4.5₩target₩androidpn-client-0.4.5
     [copy] Copying 33 files to C:₩projects₩androidpn-client-0.4.5₩target₩androidpn-client-0.4.5
     [copy] Copied 15 empty directories to 1 empty directory under C:₩projects₩androidpn-
client-0.4.5₩target₩androidpn-client-0.4.5
      [zip] Building zip: C:₩projects₩androidpn-client-0.4.5₩target₩androidpn-client-0.4.5.zip
   [delete] Deleting directory C:₩projects₩androidpn-client-0.4.5₩target₩androidpn-client-0.4.5

BUILD SUCCESSFUL
Total time: 1 second
```

④ 정상적으로 수행이 되면 target 디렉토리에 1개의 jar 파일과 1개의 zip 파일이 생성된다. 이중 클라이언트 바이너리 라이브러리는 **androidpn-client-0.4.x.jar** 형태로 빌드된 파일이며, 이는 안드로이드 애플리케이션을 개발시 사용된다.

5 서버 설치 및 구동

5.1 서버 설치

서버 애플리케이션은 빌드 시 하나의 zip 형태의 압축파일로 생성이 되며, 원하는 위치에 풀면 설치가 된다.

5.1.1 디렉토리 구조

서버 애플리케이션은 다음과 같은 디렉토리 구조를 가지고 있다.

디렉토리	설명
bin	서버 실행 스크립트가 있으며, Windows 및 Unix 용 2개의 스크립트가 있다.
conf	각종 환경 설정파일들이 있으며, 일부 파일들은 사용자 서버 환경에 맞게 수정을 해줘야 한다.
console	관리자 콘솔 웹애플리케이션 모듈이다.
lib	핵심 anroidpn 서버 모듈을 비롯하여, 각종 참조용 라이브러리들이 위치한다.

5.1.2 서버 환경 설정

설치된 서버를 정상적으로 구동하기 위해서는 몇 가지 환경 설정이 필요하다. 환경설정 파일은 conf 디렉토리 안에 있으며, 다음과 같은 설정이 필요하다.

- 서버 기본 설정 (config.properties)
  서버 관리자 콘솔을 정상적으로 구동하기 위해서 서버 호스트명과 포트번호를 지정한다.

```
admin.console.host=127.0.0.1
admin.console.port=7070
```

- DB 정보 설정 (jdbc.properties, hibernate.xml)
  기본적으로 개발 환경을 위해 HSQLDB를 이용하나, 사용 서비스를 위해서 MySQL, Oracle 등으로 바꿀 수 있다.

```
# JDBC Configuration
jdbcDriverClassName=org.hsqldb.jdbcDriver
jdbcUrl=jdbc:hsqldb:db/androidpn;shutdown=true
jdbcUsername=sa
jdbcPassword=
```

DB 정보가 바뀔 경우, ORM 프레임워크인 Hibernate 설정파일의 DB 정보도 맞춰준다.

```
<!-- Database connection settings -->
<property name="hibernate.connection.driver_class">org.hsqldb.jdbcDriver</property>
<property
name="hibernate.connection.url">jdbc:hsqldb:db/androidpn;shutdown=true</property>
<property name="hibernate.connection.username">sa</property>
<property name="hibernate.connection.password"></property>
```

5.2 서버 구동

서버 시작을 위한 시작 스크립트는 bin 디렉토리에 윈도우 및 유닉스용 스크립트가 있다. 서버 구동을 위해서는 Java 런타임 6.0 이상이 시스템에 설치되어 있어야 한다. JAVA_HOME 값을 스크립트 안에 맞게 설정해 주거나 시스템 환경정보에 설정해 주어야 한다.

아래는 윈도우 환경에서 서버 시작의 구동 예이다.

```
C:₩androidpn-server-0.4.5₩bin>run
{DEBUG} [2010-10-14 13:00:06,187] <org.androidpn.server.xmpp.XmppServer> : base.dir=..
{DEBUG} [2010-10-14 13:00:06,312] <org.apache.commons.configuration.ConfigurationUtils> :
ConfigurationUtils.locate(): b
ase is C:₩androidpn-server-0.4.5₩bin, name is config.xml
{DEBUG} [2010-10-14 13:00:06,343] <org.apache.commons.configuration.ConfigurationUtils> :
Loading configuration from the context classpath (config.xml)

...중략...

{INFO } [2010-10-14 13:00:14,562] <org.mortbay.log> : Started
SelectChannelConnector@127.0.0.1:7070
```

```
{DEBUG} [2010-10-14 13:00:14,562] <org.androidpn.server.container.AdminConsole> : Admin console started.
{INFO } [2010-10-14 13:00:14,562] <org.androidpn.server.xmpp.XmppServer> : Admin console listening at http://127.0.0.1:7070
{INFO } [2010-10-14 13:00:14,562] <org.androidpn.server.xmpp.XmppServer> : XmppServer started: 127.0.0.1
{INFO } [2010-10-14 13:00:14,562] <org.androidpn.server.xmpp.XmppServer> : Androidpn Server v0.4.5
```

서버가 정상적으로 구동되었으며, 웹브라우저를 통해서 http://호스트명:포트번호 주소를 접속하여 관리자 콘솔을 볼 수 있을 것이다. (스크린샷 참조).

6 클라이언트 애플리케이션 개발

안드로이드 푸시 통보 클라이언트 라이브러리를 포함한 애플리케이션을 개발하기 위해서는 샘플 애플리케이션(androidpn-demoapp) 프로젝트를 보면 빠르게 참조할 수 있다.

6.1 클라이언트 라이브러리 추가

이클립스 IDE에서 신규 안드로이드 프로젝트를 생성한 후, 푸시 통보 클라이언트 프로젝트(androidpn-client)에서 빌드한 클라이언트용 라이브러리(androidpn-client.jar)를 프로젝트 라이브러리에 추가한다.

6.2 클라이언트 환경 설정 파일

안드로이드 프로젝트의 기본 폴더인 res 폴더 밑에 raw 디렉토리를 생성 후 그곳에 **adroidpn.peoperties** 파일을 생성한다. 이 프로퍼티 파일에 다음과 같이 접속할 서버의 호스트와 포트번호(기본:5222) 정보를 입력하여 편집한다.

```
xmppHost=192.168.0.5
xmppPort=5222
```

6.3 클라이언트 서비스 시작 코드

- 클라이언트 메인 Activity 클래스 파일의 onCreate() 메소드 안에 다음과 같은 코드를 작성한다.

```
public void onCreate(Bundle savedInstanceState) {
        Log.d("DemoAppActivity", "onCreate()...");

        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
```

```
        // Start the service
        ServiceManager serviceManager = new ServiceManager(this);
        serviceManager.setNotificationIcon(R.drawable.notification);
        serviceManager.startService();
    }
```

- 푸시 통보 수신 여부 및 사운드, 진동을 설정하는 Settings Activity를 표시하기 위해서는 버튼 등의 클릭 이벤트에 다음과 같은 코드를 삽입하면 된다.

```
Button okButton = (Button) findViewById(R.id.btn_settings);
okButton.setOnClickListener(new View.OnClickListener() {
    public void onClick(View view) {
        ServiceManager.viewNotificationSettings(DemoAppActivity.this);
    }
});
```

- AndroidManifest.xml 파일에 다음과 같이 서비스 및 Activity 클래스를 등록해 준다.

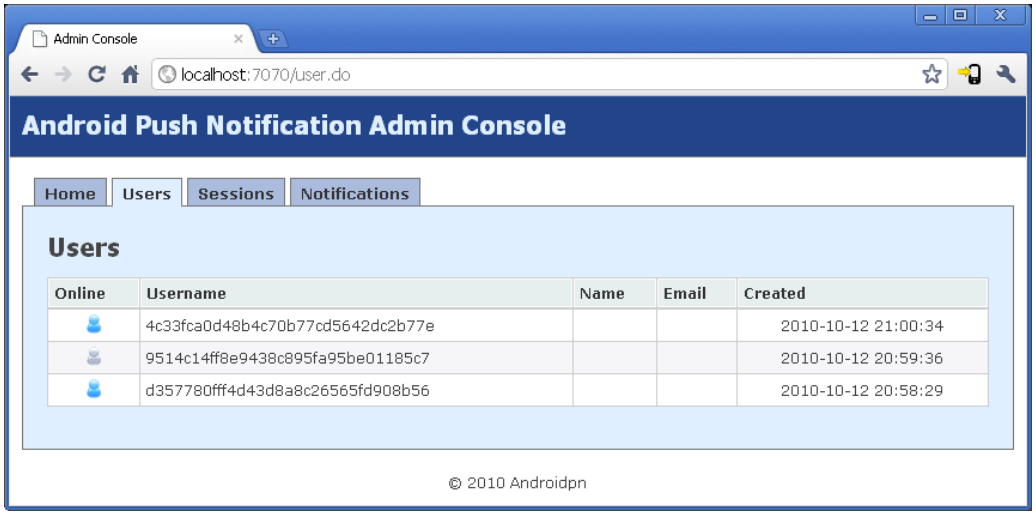
```
<activity android:name="org.androidpn.client.NotificationDetailsActivity"
        android:label="Notification Details">
</activity>

<activity android:name="org.androidpn.client.NotificationSettingsActivity"
        android:label="Notification Settings">
</activity>

<service android:enabled="true"
        android:name="org.androidpn.client.NotificationService"
        android:label="NotificationService">
        <intent-filter>
                <action android:name="org.androidpn.client.NotificationService" />
        </intent-filter>
</service>
```

7 프로젝트 스크린샷

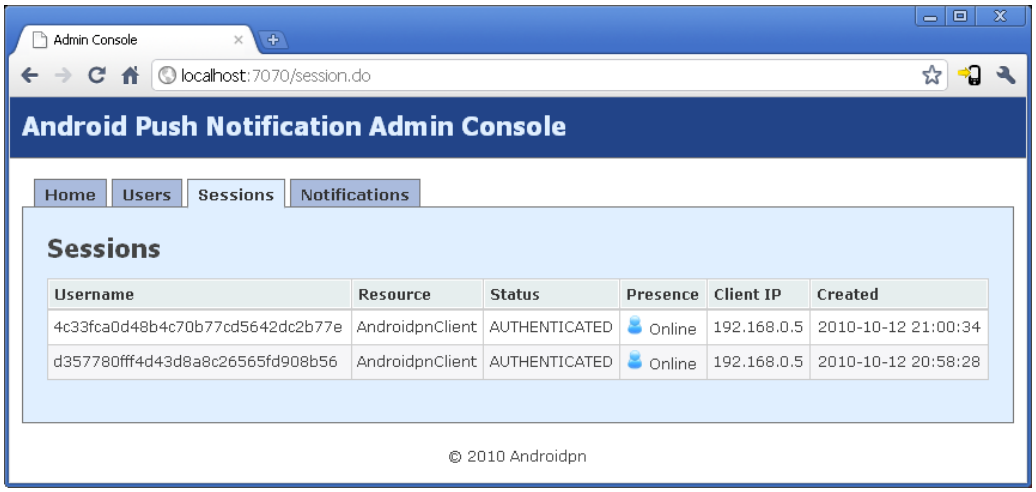
7.1 서버 (관리자 웹 콘솔)

- 푸시 통보 서버의 관리자 콘솔 초기화면이다.

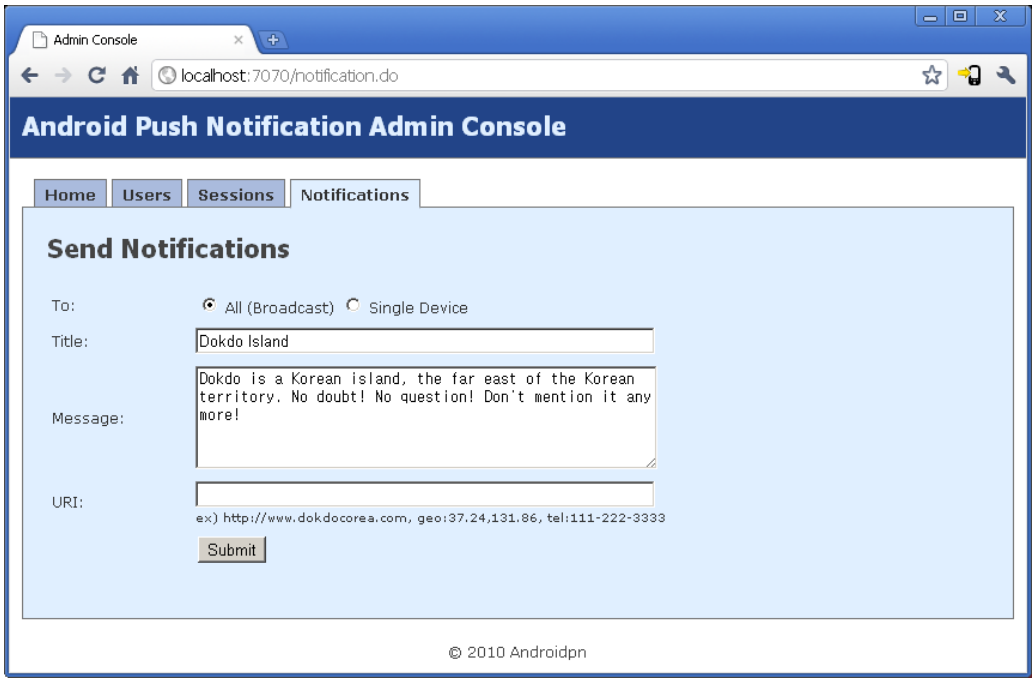
- 서버에 등록된 클라이언트 사용자 목록이다.



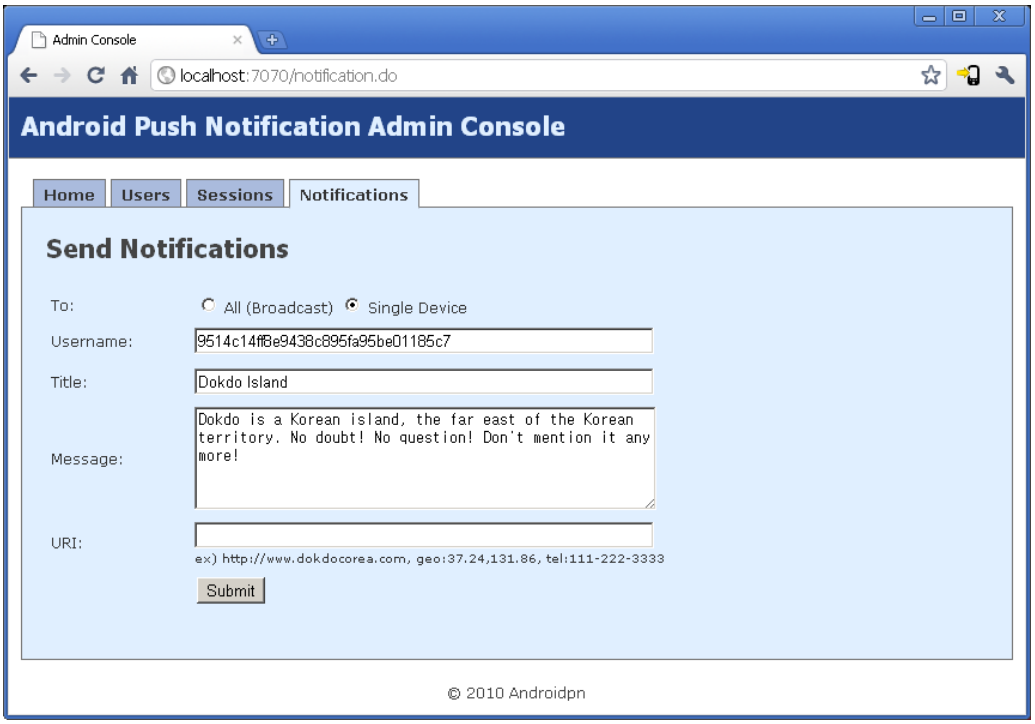
- 현재 접속된 클라이언트 세션 목록이다.



- 모든 클라이언트에 푸시 메시지 브로드캐스트를 할 수 있다.

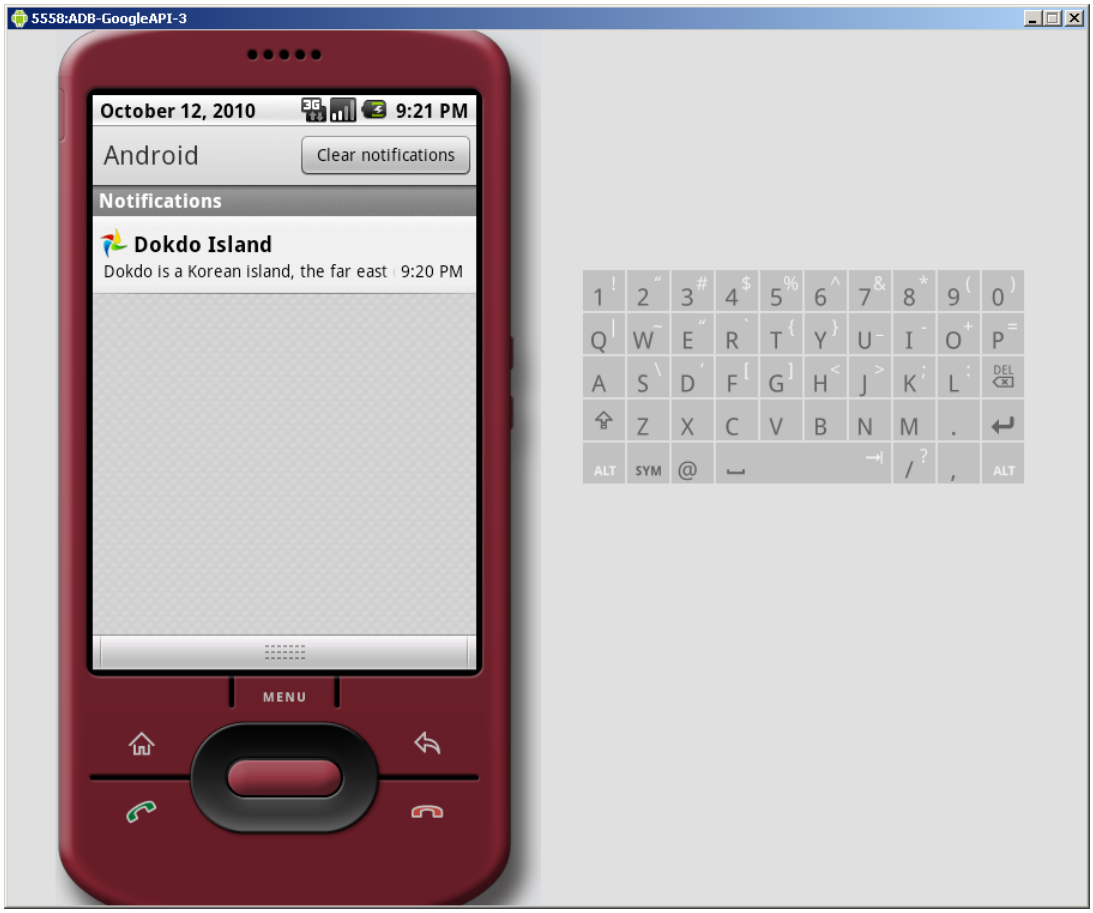


- Username을 지정하여 특정 클라이언트에만 푸시 메시지 전송할 수 있다.

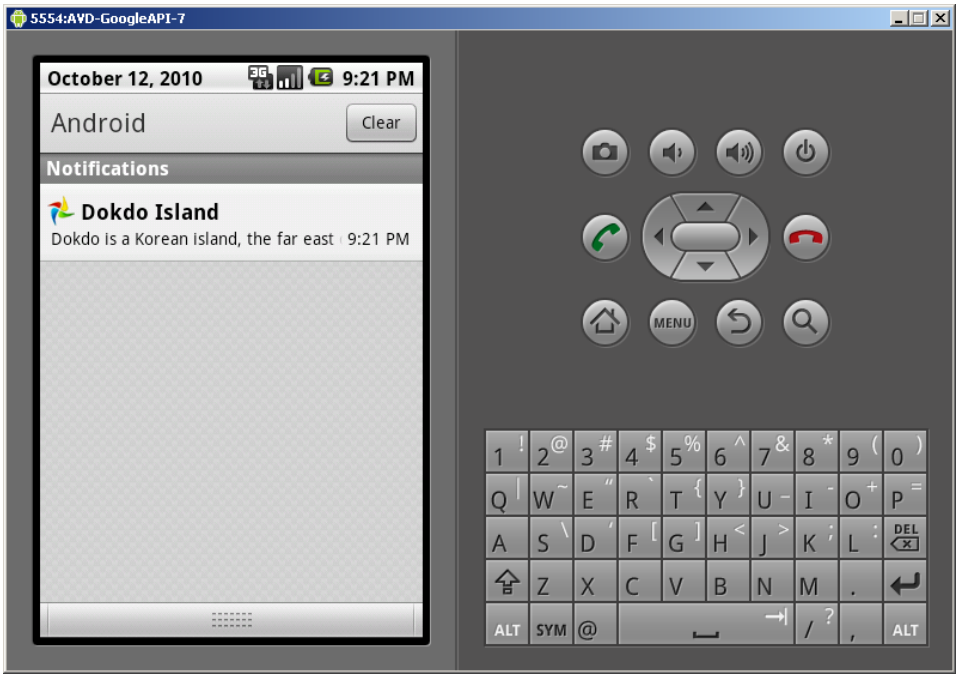


7.2 클라이언트 (안드로이드 애플리케이션)

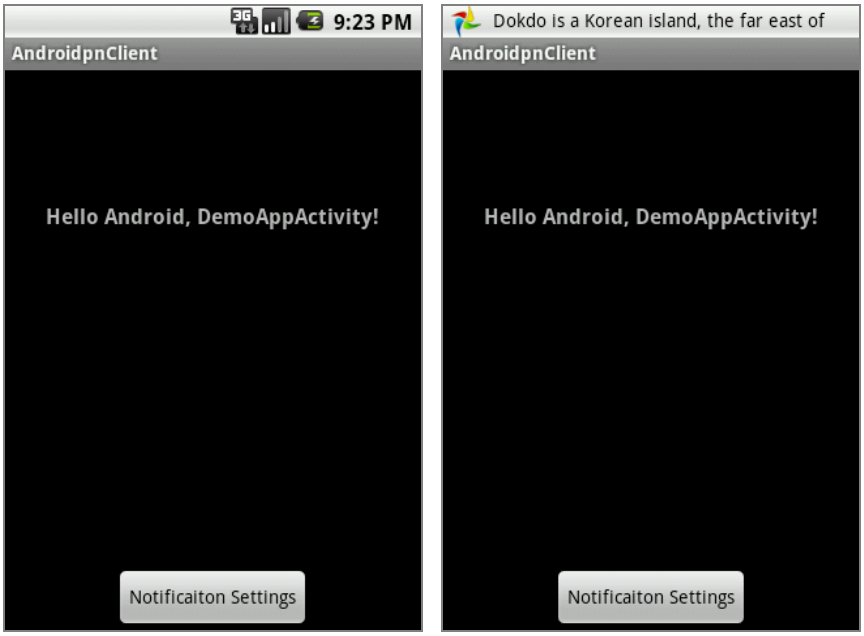
- Android 1.5 (Cupcake) 에뮬레이터에서 푸시 메시지를 수신한 화면이다.



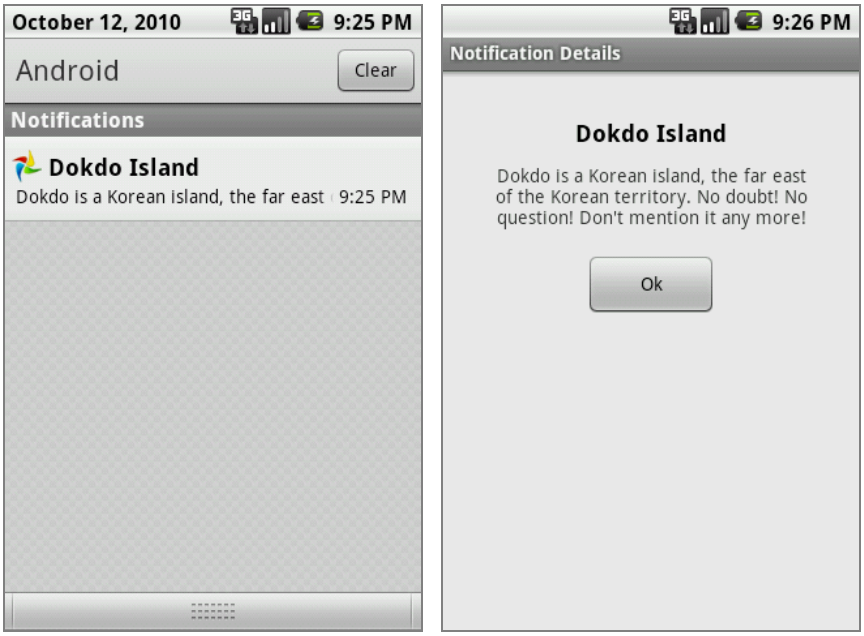
- Android 2.1 (Éclair) 에뮬레이터에서 푸시 메시지를 수신한 화면이다.



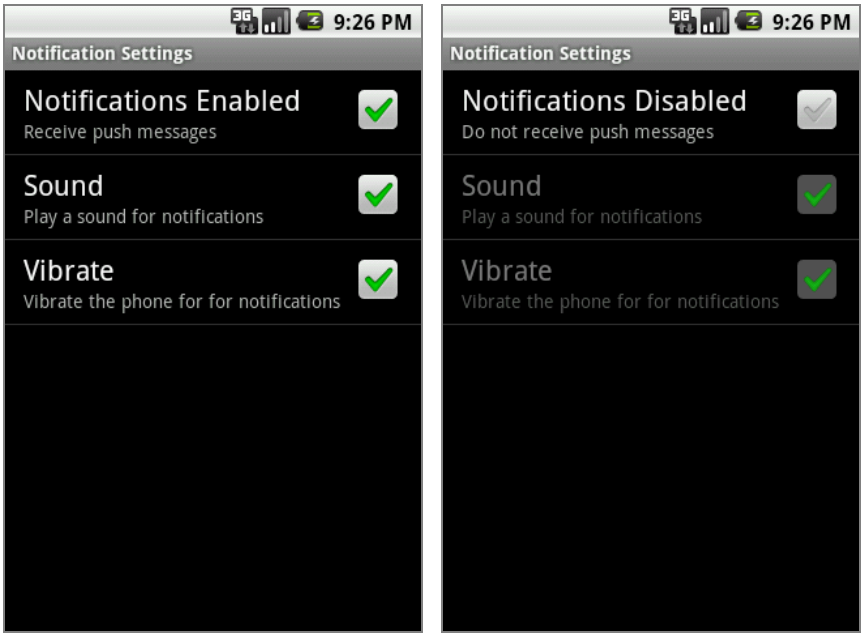
- 데모 애플리케이션 구동 초기화면 및 메시지 수신시 상태 Ticker 활성 화면이다.



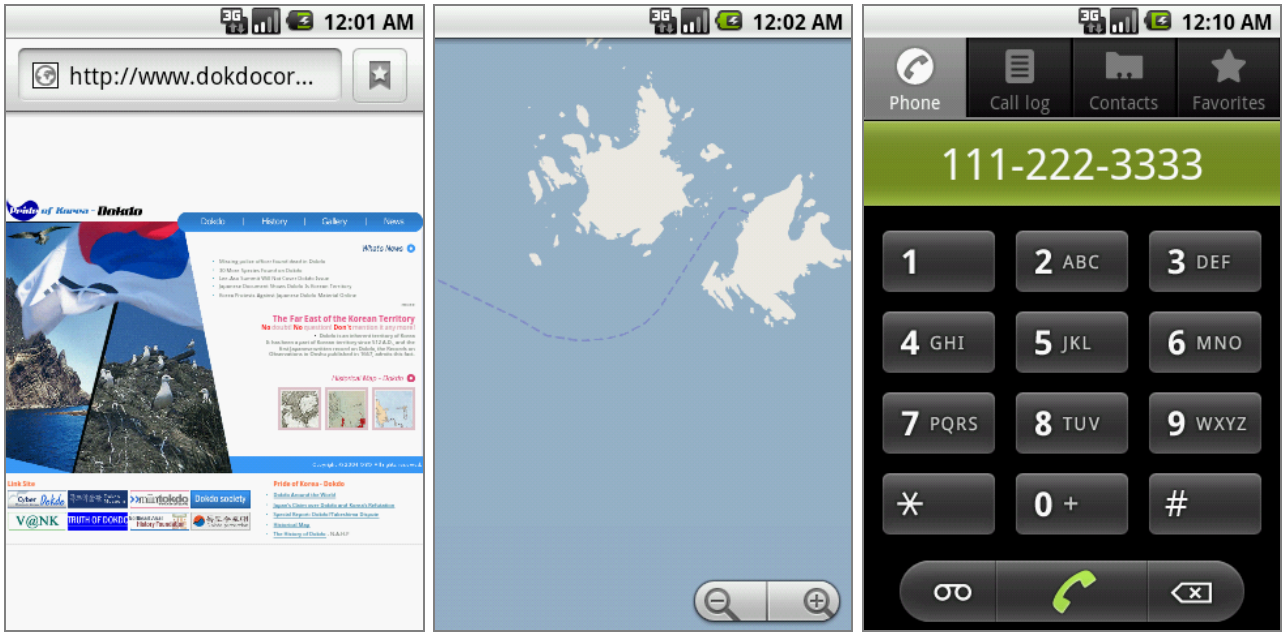
- 메시지 수신 목록 화면 및 이를 터치했을 경우 전체 메시지 상세 내용을 볼 수 있다.



- 푸시 통보 설정화면에서 메시지 수신여부 및 사운드, 진동 여부 설정할 수 있다.



- 서버에서 URI 정보를 추가로 전송했을 경우, 메시지 내용 확인 후 특정 웹페이지 및 지도 위치, 걸화걸기 등을 표시할 수 있다. 아래 화면은 URI 정보에 http://www.dokdocorea.com, geo:37.24,131.86, tel:111-222-333 값을 각각 전송한 경우이다.



8 주요 관련 기술

8.1 푸시 통보 (Push Notification)란?

8.1.1 푸시 통보란?

- 푸시 통보란 단말 시스템에서 애플리케이션이 동작 중이지 않은 상태에서도 서버로부터 "Push"되는 메시지를 Listen하여 수신할 수 있는 기술이다.
- 모바일 애플리케이션에서 발생하는 주요 뉴스 및 이벤트 등을 사용자에게 전달해 줄 수 있는 수단이다.
- 자원을 많이 소비하는 클라이언트에서 서버로 지속적인 Polling을 하는 Pull 기술에 비해 효율적인 기술이다.

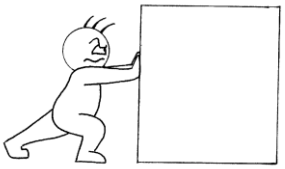
8.1.2 모바일 푸시 통보 서비스

모바일 플랫폼	업체	현황
BlackBerry	RIM	2002년부터 Push Email 서비스를 시작으로 자체적으로 푸시 통보 서비스를 지원
iPhone OS (iOS)	Apple	2009년 iPhone OS 3.0부터 Apple Push Notification Service (APNS)를 지원
Android	Google	Android 2.2 (Froyo)부터 Cloud To Device Messaging (C2DM)을 통해 지원 예정
Windows Phone 7	Microsoft	Microsoft Push Notification Service (MPNS) 지원과 함께 출시 예정

8.2 Push vs. Pull

네트워크 상의 서버와 클라이언트 간의 통신 방식에서 요청하는 주체를 기준으로 다음과 같이 Push 와 Pull 방식으로 나눌 수 있다.

8.2.1 Push Technology (Server Push)



- 메시지 전송 트랜잭션이 서버에서 시작되어 클라이언트로 전송되는 통신 방식이다.
- XMPP, SMTP, HTTP Server Push (HTTP Streaming), Long Polling 등

8.2.2 Pull Technology (Client Pull)

- 클라이언트에서 데이터 전송 요청이 시작되어 서버로부터 응답을 받는 형태의 통신 방식이다.
- HTTP, POP3, IMAP, RSS Feeds 등

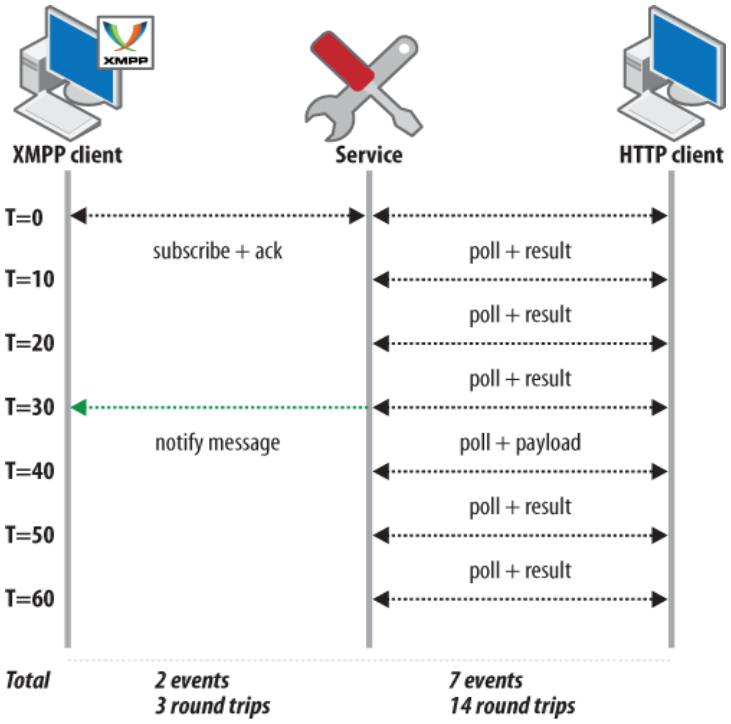
8.3 XMPP (eXtensible Message and Presence Protocol)

8.3.1 XMPP란?

- 인터넷 상의 두 지점 간에 확장 가능한 메시지와 상태정보(Presence)를 실시간으로 통신하기 위한 XML 기반의 오픈 표준 기술이다.
- 1999년 Jabber 오픈소스 커뮤니티에 의해 개발 되었고, 2000년~2004년에 걸쳐 IETF에 의해 표준화 되었다.
- XMPP Standard Foundation (XSF)의 표준화 작업에 의해 지속적 확장되고 있으며, XMPP Extension Protocols (XEP)로 별도로 확장 스펙을 관리하고 있다.
- http://xmpp.org/ 싸이트에서 정보를 얻을 수 있다.

8.3.2 XMPP vs. HTTP

XMPP 프로토콜은 Push 기술의 통신방식으로 클라이언트가 서버로부터의 정보를 업데이트하고자 할 때, HTTP 기반의 Polling 방식에 비해 훨씬 클라이언트의 자원 소모가 적고 효율적이다.



8.3.3 XMPP는 다향한 분야에서 응용되는 프로토콜

- Instant Messaging (IM)
- Multi-User Chat
- Voice and Video Conferencing
- Real-time Collaboration
- Social Networking
- Microblogging
- Lightweight Middleware
- Content Syndication

8.4 안드로이드(Android)

8.4.1 안드로이드란?

- 운영체제, 미들웨어 및 핵심 응용프로그램을 포함하는 휴대기기용 소프트웨어 스택
- Google에 의해 개발되고 오픈 소스화 (2008년)
- Linux 커널 기반의 운영체제
- Dalvik 가상머신과 Java 언어 기반의 SDK 지원
- Android Market을 통해 애플리케이션 배포

8.4.2 안드로이드는 진화 중

- Cupcake(v1.5) → Donut(v1.6) → Éclair(v2.0,v2.1) → Froyo(v2.2) → Gingerbread(3.0)
- 현재 Andorid 2.2 (Froyo)까지 발표되었으며, 이 OS 버전부터 C2DM(Cloud to Device Messaging) 기술이 적용되었으며, 조만간 구글에서 정식 서비스 예정이다.

9 개발 환경

프로젝트 기본 개발 환경은 서버와 클라이언트 모두 Java 언어를 사용하고 있으며 JDK 5.0 이상과 Eclipse (3.5 권장)를 필요로 한다. 상세 개발 환경은 다음과 같다.

9.1 서버 개발 환경

구분	설명
시스템 플랫폼	Linux 또는 Windows
프로그래밍 언어	Java (JDK 1.5 이상)
IDE	Eclipse 3.5
빌드 도구	Ant 1.7.1, Maven 2.0
Database	HSQLDB 또는 MySQL
Servlet Container	Jetty 6.1
J2EE 스펙	Servlet 2.4, JSP 2.0
핵심 프레임워크 및 라이브러리	Spring 2.5, MINA 2.0, Hibernate 3.3.1, SiteMesh 2.4

9.2 클라이언트 개발 환경

구분	설명
디바이스 단말기	삼성 갤럭시S
모바일 플랫폼	Android SDK r07
프로그래밍 언어	Java (JDK 1.5 이상)
IDE	Eclipse 3.5
빌드 도구	Ant 1.7.1
핵심 프레임워크 및 라이브러리	Smack 3.1

10 발전 방향

10.1 소프트웨어 품질 향상

- 클라이언트와 서버 간의 SSL/TSL 보안 채널 적용으로 상용화 서비스가 가능한 수준으로 품질 향상. (XMPP 프로토콜에서 권장하는 방식을 채택).

- Wi-Fi, 3G, HSDPA 망 구분과 수시로 변할 수 모바일 네트워크 환경에서 클라이언트와 서버간의 연결 지속성을 위한 품질 향상 및 다양한 환경에서 테스트.

10.2 서버 및 클라이언트 시스템 확장

- 서버 시스템을 확장하여 여러 개의 안드로이드 애플리케이션들을 동시에 지원하는 플랫폼 서비스에 응용될 수 있음. 여러 개의 애플리케이션을 동시에 지원하기 위한 개별 애플케이션 인증과 여러 개의 애플리케이션을 효율적으로 관리할 수 있는 클라이언트 라이브러리 향상 필요.
- 클라이언트는 안드로이드 OS 이외의 타 모바일 OS 애플리케이션을 지원을 위해 클라이언트 모듈을 확장할 수 있음.

10.3 다양한 분야에 응용

- 메시지 프로토콜 확장을 통해 이미지, 사운드, 비디오 등 멀티미디어 메시지 전달하는 등 더욱동 생동감 있는 뉴스 전달하는 애플리케이션에 응용될 수 있음.
- 사용자의 위치에 따른 지역 광고 메시지를 전달하는 위치기반 서비스(LBS, Location Based Service)를 제공하는 애플리케이션에 이용될 수 있음.
- SMS(Short Message Service)를 대신하는 가입자가 실시간으로 단문 메시지 전달하는 서비스 등에 응용될 수 있음.

10.4 오픈소스 커뮤니티 확대에 기여

- SourceForge, Google Code, KLDP 등 다양한 오픈소스 커뮤니티를 통해 국내외 다양한 개발자 참여 유도 및 이를 통한 품질 향상.
- 국내 개발자들 개발자들에게 오픈소스를 통한 개발자 역량 향상을 꽤하고, 다양한 안드로이드 애플리케이션에 응용될 수 있는 기회를 제공.