**<PROJECT 중간보고서>**

**REMOQON**QRemote를 활용한 Android 기반 리모컨

**B팀 정현호, 송시우, 최지헌**

**엄현상 교수님**

**전정은 담당자님 (LG전자, Technical Evangelist and Documentation Part)**

Table of Contents

1. Abstract 3

2. Introduction 3

3. Background Study 4

A. 관련 접근방법/기술 분석 4

B. 프로젝트 개발환경 4

4. Goal/Problem & Requirements 5

5. Approach 5

6. Project Architecture 6

A. Architecture Diagram 6

B. Architecture Description 6

7. Implementation Spec 7

A. Input/Output Interface 7

B. Inter Module Communication Interface 8

C. Modules 9

8. Current Status 9

9. Future Work 10

10. Division & Assignment of Work 10

11. Schedule 10

 [Appendix] Detailed Implementation Spec 11

A. MainActivity 11

B. BroadcastListActivity 12

C. DetailedTableActivity 13

# **Abstract**

이 프로젝트에서는 LG QRemote SDK를 이용해 QRemote의 특성을 잘 살린 독창적인 안드로이드 어플리케이션을 제작하는 것을 목표로 한다. 우리가 계획한 것은 TV 편성표를 인터넷에서 가져와 사용자에게 보여주고, 사용자가 선택한 방송 프로그램이 시작되기 전에 사용자에게 알려주고, QRemote 기능을 이용해 자동으로 채널을 전환하도록 하는 기능을 하는 어플리케이션이다. 또, 사용자가 직관적으로 리모컨의 버튼을 사용할 수 있도록 UI를 개선하고, 스마트 TV의 경우 Wi-Fi 연동을 통해 사용자가 직접 휴대전화를 TV 쪽으로 향할 필요 없이 같은 기능을 하도록 구현할 수 있으므로, 스마트 TV도 구현 과정에서 고려하도록 계획했다.

또, 기본적으로 탑재된 QRemote 어플리케이션에는 배경화면 위젯 기능이 없으므로 배경화면 위젯을 추가해 리모컨 기능에 대한 접근성을 향상시키려고 했다.

# **Introduction**

LG QRemote란 스마트폰의 IR(적외선) 센서를 이용하여 TV, DVD, 에어컨, 셋톱박스, 오디오 등의 각종 전자기기를 스마트폰 하나로 조작할 수 있도록 하는 LG 스마트폰의 새 기능이다. 3rd party application 제작자가 이 기능을 사용할 수 있도록 LG에서 QRemote SDK를 제공하고 있다.

우리가 제작하려고 하는 것은 편성표 기능을 TV 리모컨 기능과 연동시킨 어플리케이션이다. TV를 시청할 때 별다른 시청 계획을 정하지 않고 원하는 프로그램이 나올 때까지 채널을 돌리는 경우가 잦다. 또, IPTV 등의 기능으로 편성표가 제공되기는 하나 사용이 불편하고 리모컨을 통해 조작한다는 특성상 잘 쓰이지 못하고 있는 실정이다. 따라서 우리는 이런 점들에 착안해 휴대전화를 통해 편성표를 손쉽게 확인하고, 방송 프로그램을 스케쥴에 등록해 프로그램의 시작 시간이 되면 등록된 TV로 적외선 신호를 보낼 수 있는 어플리케이션을 만들어 이런 불편함을 해소하고자 한다.

스마트 TV에 대해서는 적외선 신호를 송출하지 않고 스마트 TV의 기능을 이용해 자동으로 채널 전환이 이루어지도록 할 것이다. 또, 스마트 TV의 리모컨을 이용해 문자 입력을 하기가 어렵다는 점을 개선하기 위해 스마트폰에서 문자를 입력할 수 있도록 키패드를 제공할 것이다.

즉, 우리가 중점을 둔 요소는 사용자의 편의이다. 따라서 UI 측면도 최대한 고려해 사용자가 더 나은 경험을 얻을 수 있도록 하는 것도 핵심적인 목표에 포함된다.

# **Background Study**

## **관련 접근방법/기술 분석**

QRemote 기능을 탑재한 LG 스마트폰에는 LG QRemote Library가 설치되어 있다. LG QRemote Library는 API를 제공하는데, 3rd party application 제작자는 이 API를 통해 LG QRemote Library에 접근한다. 라이브러리는 QRemote와 관련된 전반적인 기능을 총괄하는 역할을 한다. 앞서 말했듯 3rd party application에 API를 제공하며, 3rd party application의 요청에 따라 IR 서비스와 통신한다. IR 서비스에서는 실제로 IR 신호를 송출하는 모듈에 정보를 전달해 IR 신호를 송출한다.

LG QRemote SDK를 사용할 때 특이한 점은 등록한 전자기기와 관련된 정보가 어플리케이션 외부에 저장된다는 점이다. 3rd party application에서 새로 기기를 등록하고자 한다면 LG QRemote 어플리케이션을 통해 기기를 등록해야 하고, 3rd party application에서 등록된 기기들에 대한 정보를 확인하고자 할 때에도 LG QRemote 어플리케이션과 통신을 통해 정보를 얻어야 한다. 이는 QRemote를 활용한 다양한 어플리케이션을 동시에 사용할 때 기기 정보가 분산되어 관리가 힘든 점을 막기 위한 것으로 보인다.

또, 우리는 스마트 TV에 대해서는 적외선 신호를 송출하지 않고 Wi-Fi 통신을 이용해 TV를 제어하려고 하므로, LG Smart TV SDK도 사용할 것이다. LG Smart TV SDK는 UDAP를 사용하는데, UDAP란 스마트 가전을 스마트폰 등의 기기로 제어하는 프로토콜이다. UDAP는 HTTP/1.1 기반으로 되어있고, 컨트롤러와 호스트가 같은 네트워크에 연결되어 있다면 컨트롤러를 통해 호스트를 제어할 수 있는 있는 기능을 지원한다. 우리는 스마트폰을 이용해 스마트 TV를 조작하고자 하므로, 컨트롤러는 스마트폰이 되고 호스트는 스마트 TV가 될 것이다. LG Smart TV SDK를 이용해 스마트 TV와 통신하는 과정은 크게 Discovery, Description, Pairing, Command/Event/Query의 4단계로 구분된다. Discovery 단계에서는 네트워크에 연결된 장치를 검색하고, Description 단계에서는 검색된 장치의 정보를 수집해 그 장치가 지원하는 서비스들을 파악한다. Pairing 단계에서는 호스트와 클라이언트가 연결되며, 마지막 Command/Event/Query 단계에서는 실제로 통신이 이루어진다. 이 SDK를 이용해 스마트 TV의 채널 전환 및 문자 입력 기능을 구현할 수 있다.

## **프로젝트 개발환경**

3인 1조로 구성된 조이므로, 협업이 중요한 요소가 될 것이라 생각해 버전 관리 도구로 git을 이용해 각각 개발을 수행할 수 있도록 한다. 원격 저장소로는 github을 사용하고, issue 및 wiki 기능을 이용해 서로 문제점 및 스펙을 공유할 것이다.

안드로이드 어플리케이션은 기본적으로 Java를 이용해 구현하게 되므로 eclipse 및 Android Development Kit을 이용해 개발을 진행한다. 개발 대상이 되는 안드로이드 OS 버전은 4.1 (API Level 16)이다. LG QRemote SDK가 지원하는 minimum API가 16이고, 또한 QRemote 기능이 처음으로 탑재된 Optimus G Pro의 기본 OS 버전이 4.1이기 때문에, 이전 버전에 대해서는 호환성을 생각할 필요가 없다.

또한 서버를 이용하여 TV 편성표를 사용자에게 제공한다. 서버는 학교나 회사 측에서 제공되지 않고, 또한 서버를 제공받아야 할 정도로 고성능의 서버가 당장 필요하지는 않으므로, 시연 용도로는 팀원이 개인적으로 사용하고 있는 서버를 활용하게 될 것이다. 편성표 수집을 위한 서버 개발은 Ruby on Rails를 이용하여 진행할 것이다.

# **Goal/Problem & Requirements**

이번 프로젝트의 목표는 편성표를 따로 찾아보지 않아도 리모컨 앱 내에서 편성표를 보고 프로그램을 바로 시청하거나 예약할 수 있는 Android 앱을 제작하는 것이다. 또한 LG Smart TV에 한해서는 QWERTY 키패드 기능을 제공하고 스마트폰이 TV를 향하고 있지 않더라도 예약된 프로그램을 실행할 수 있도록 하는 것도 이번 프로젝트의 목표이다.

이번 프로젝트가 성공적으로 완수된다는 것은 모든 채널에 대한 편성표를 사용자에게 빠짐없이 제공하고, 사용자가 시청을 원하는 프로그램을 예약하면 정해진 시간에 오차 없이 사용자에게 채널 변경 알림을 전달하며, 채널 변경 명령이 주어졌을 때 원하는 채널 번호에 해당하는 IR 신호를 완벽히 송신하는 앱을 제작하는 것을 의미한다. 또한 사용자의 편의를 위해 리모컨의 UI를 개선하여 사용자 경험(User eXperience)을 극대화하는 것도 이번 프로젝트의 요구 조건들 중 하나이다.

# **Approach**

앞서 언급했듯 이번 프로젝트에서는 TV 시청자가 스마트폰만을 이용하여 일반적인 TV 관련 작업을 수행할 수 있도록 한다. 이와 더불어 TV 편성표에 기반하여 시청하고자 하는 프로그램을 예약하여 자동으로 채널을 변경시켜주는 등 TV 시청 과정에서의 편의성을 도모하고자 한다.

이 목표를 달성하기 위하여 가장 먼저 필요한 리모컨 기능은 LG전자에서 제공한 QRemote SDK를 활용하여 구현한다. LG QRemote SDK를 이용하면 IR 하드웨어를 직접적으로 조작하지 않고도 TV 리모컨 기능을 구현할 수 있다. 또한 TV 편성표 기능을 구현할때 외부에서 편성표 정보를 가져와 앱에서 사용하기 좋은 형태로 가공해야 하는데, 현재 웹에서 제공되는 TV 편성표로는 정보를 가공하는 것이 힘들기 때문에 편성표 정보의 가공 작업을 수행하는 서버 또한 직접 구현하도록 한다. 앱에서 필요한 정보를 HTTP Query로 서버로 요청하면, 서버에서는 요청한 정보만을 편성표에서 추출해내어 반환한다. 이 과정에서 서버와 앱 사이의 간단한 통신 과정이 필요하다. 또한 앱에서는 반환된 정보를 바탕으로 TV 편성표를 자체 데이터베이스에 저장하고, 이를 사용자에게 출력한다. 데이터베이스에 편성표를 저장하는 이유는, 앱을 실행할 때마다 서버에 접근하는 것은 비효율적이기 때문이다.

# **Project Architecture**

## **Architecture Diagram**



## **Architecture Description**

이번 프로젝트는 크게 서버와 앱, 두 부분으로 나뉘어진다.

우선 서버에서는 TV 편성표 정보를 가공하여 안드로이드 앱에 전송해주는 역할을 담당한다. 서버는 네이버 TV 편성표 페이지로부터 당일의 편성표 정보를 텍스트 형태로 받아온다. 이 단계에서 수신하는 정보에는 당일에 방영되는 모든 프로그램의 이름과 시작 시간, 그리고 종료 시간이 포함된다. 수신된 정보를 바탕으로 서버는 국내의 모든 방송국 list를 추출하고, 방영되는 프로그램들을 방송사에 따라 분류, 가공하는 작업을 수행한다. 가공이 완료된 정보는 HTTP Query를 이용하여 접근이 가능하다. Query에 따라 모든 방송국의 list, 지상파 채널의 방송 목록, 특정 방송국의 프로그램 목록 등을 앱에서 얻어올 수 있다.

앱에서는 기본적인 리모컨 기능과 더불어 편성표를 활용한 자동 채널 변경 기능을 담당한다. 우선 LG QRemote SDK를 활용하여 IR 신호를 이용한 리모컨 기능을 구현한다. 리모컨 앱과 실제 TV를 페어링하는 과정은 스마트폰에 설치되어 있는 QRemote 앱이 담당한다. 그리고 TV와 페어링이 완료되면 SDK에 내장되어 있는 함수를 이용하여 TV의 각 기능에 해당하는 IR 신호를 송출하게 된다.

또한 위에서 구현한 서버와 통신하며 TV 편성표 정보를 수집하고, 이를 바탕으로 사용자가 시청하고자 하는 TV 프로그램 목록을 저장한다. 저장된 프로그램의 방영 시간이 되면 사용자에게 알림 메시지를 송출하고, 사용자가 채널 변경을 수락하면 채널을 변경하는 IR 신호를 자동으로 송출한다.

스마트 TV의 경우 LG Smart TV SDK를 활용해 IR 신호 대신 Wi-Fi 신호를 통한 리모컨 기능을 구현한다. SDK로부터 같은 네트워크에 연결되어 있는 스마트 TV를 찾고, IR 신호를 송출하는 경우와 마찬가지로 채널을 변경하는 Wi-Fi 신호를 송출하도록 한다. IR 신호의 경우와 달리 Wi-Fi 신호를 통하는 경우 사용자가 스마트폰을 TV 방향으로 향할 필요가 없으므로 좀 더 나은 사용자 경험을 제공할 수 있을 것이다. 또한 스마트 TV의 리모컨을 이용한 키보드 입력이 어렵다는 점에 착안해 qwerty 키보드를 제공한다. 사용자가 qwerty 키보드를 이용해 입력한 내용은 스마트 TV로 전송된다.

# **Implementation Spec**

## **Input/Output Interface**

1. **Input**

이번 프로젝트에서는 사용자와 리모컨 앱 사이의 상호작용 과정에서 대부분의 input이 발생할 것이다. 리모컨 기능을 이용하려면 TV와 앱을 페어링해야 하기 때문에, 가장 처음으로 발생하는 input은 앱에 사용자의 TV를 등록시키는 절차가 될 것이다. 하지만 device의 등록 과정을 3rd party application이 직접 담당하지 않도록 QRemote SDK가 구현되어 있기 때문에, 사용자의 TV를 등록하는 것은 이번 프로젝트의 고려 대상이 되는 input에서 제외된다.

다음 input은 기본적인 리모컨 기능을 사용하기 위한 사용자의 버튼 입력이 될 것이다. 이 input을 입력받기 위해서는 사용자의 터치를 인식하는 과정이 필요하다. 여기서 버튼을 클릭한다는 개념이 아니라 터치한다는 개념을 도입하였는데, 클릭의 경우 버튼에 손이 닿는 순간에만 일회성으로 이벤트가 발생하지만, 터치의 경우 손을 화면에 붙이고 있는 시간동안 계속 이벤트가 발생한다는 점에서 클릭과 터치의 개념이 구분된다. 리모컨의 버튼을 계속 누르고 있으면 계속해서 IR 신호가 송출되는 것이 사용자의 인식에도 당연하고, 또 여러 번 누를 필요가 없어 편리성 측면에서도 좋을 것이다. 따라서 사용자의 터치를 인식하기 위하여 이번 프로젝트에서는 Android에서 기본적으로 제공하는 OnTouchListener class를 사용한다. 또한 사용자가 터치하고 있는 시간동안 IR 신호를 발생시켜야 하기 때문에, 이 단계에서 LG QRemote SDK를 사용한다.

세 번째 input은 편성표 상에서 사용자가 시청하고자 하는 프로그램을 선택하는 과정에서 발생한다. 사용자가 프로그램을 선택하면, 앱에서는 해당 프로그램의 이름과 더불어 방송 시작 시간, 방송국 번호 등을 저장해 정해진 시간이 되면 알리도록 하는 데에 사용한다.

1. **Output**

이번 프로젝트의 output은 대부분 리모컨의 IR 신호 형태로 나타난다. 이 신호는 사용자의 버튼 입력에 의해 발생할 수도 있고, 채널 자동 전환을 위해 앱에서 스스로 발생시킬 수도 있다.

IR 신호는 QRemote SDK를 통하여 발생시킨다. 우선 SDK에서 getIRBlaster() function을 호출하여 IR 하드웨어를 작동시킨다. 정상적으로 하드웨어가 준비된다면 IRBlasterReady() callback function이 실행되며, 이후 getDevices() function을 실행하여 remoQon 앱에 등록되어 있는 기기 목록을 불러온다. 여기서 반환된 장치들 중 하나를 선택한 후 어떤 기능을 작동시킬 것인지를 인자에 담아 sendIR() function을 호출하면 IR 신호가 송출된다.

사용자가 원하는 프로그램을 등록하는 과정에서도 output이 발생한다. 만약 사용자가 등록하고자 하는 프로그램이 현재 방영 중인 프로그램일 경우에는 해당 프로그램은 예약 리스트에 등록하지 않고 바로 채널 전환이 이루어져야 할 것이다. 현재 시간과 프로그램의 시작 시간 및 끝나는 시간 정보를 이용하여 현재 시간이 시작 시간과 끝나는 시간 사이에 있다면, 즉 프로그램이 현재 방영 중이라면 그 결과로 채널을 전환하는 IR 신호를 output으로 내놓는다.

IR 신호 이외에도 편성표 정보를 불러와 사용자의 스마트폰 화면에 출력하는 것도 output에 해당한다. 편성표는 방송의 리스트로 볼 수 있고, 따라서 이를 효과적으로 사용자에게 표현하기 위하여 Android의 ListView를 활용해 리스트 형태로 표현하도록 구현하였다.

또한 스마트 TV의 경우 IR 신호 대신에 Wi-Fi 통신을 통해 채널 전환이 이루어지도록 하고, 스마트폰의 qwerty 키보드를 이용해 스마트 TV에 문자를 전달할 수 있도록 할 것이므로, 이 때의 output은 LG Smart TV SDK의 프로토콜을 통해 전달하는 정보가 될 것이다.

## **Inter Module Communication Interface**

Android app 내에서 module로 볼 수 있는 것은 각각의 activity들일 것이다. Android에서 activity 간의 이동이나 정보의 교환은 Intent class를 통하여 이루어진다. Intent class를 선언할 때는 전환하고자 하는 target activity를 명시한다. 그 후 intent.putExtra() 함수에 target activity에 전달하고자 하는 인자를 추가하고, startActivity(intent)를 실행하면 target activity로의 전환이 이루어진다.

또한 앱과 서버 사이의 통신도 module 간의 communication으로 볼 수 있다. App에 구현한 readUrl() 함수를 사용하여 App과 서버 사이에서의 통신이 수행된다. readUrl()의 인자는 http URL string이다. 인자로 넘겨받은 string을 url.openStream() 함수의 인자로 다시 전달하면 이는 서버에 HTTP GET Request를 전달한 것이 되고, 서버에서는 가공된 데이터를 JSON string의 형태로 App에 반환한다. App에서는 반환받은 JSON string을 StringBuffer를 이용하여 저장한 후, JSON parsing 과정을 수행하여 TV 편성표를 자체 데이터베이스에 저장하는 과정을 거치게 된다.

마지막으로 스마트 TV의 경우 앱과 TV 사이의 통신을 communication이라 볼 수 있다. LG Smart TV SDK에서 제공하는 프로토콜을 이용해 클라이언트(스마트폰)으로부터 서버(스마트 TV)로 정보를 전달하는 정보 교환이 이루어지기 때문이다. 이 때에는 같은 Wi-Fi 네트워크에 연결된 스마트 TV를 검색하고, HTTP 요청을 이용해 스마트 TV로 명령을 전달하는 과정을 거치게 된다.

## **Modules**

Android에서는 activity가 기본적인 기능 단위, 즉 module이라고 할 수 있다. 이번 프로젝트에서는 Activity를 크게 3가지로 구분한다. 첫 번째는 기본적인 리모컨 기능을 담은 MainActivity, 두 번째는 방송국의 리스트를 보여주는 BroadcastListActivity, 그리고 마지막 세 번째는 두 번째 단계에서 선택된 방송국에서 당일에 방영되는 프로그램의 리스트를 보여주는 DetailedTableActivity이다. 간단히 생각해 한 화면이 하나의 activity로 구성되고, 화면의 전환은 activity의 전환을 의미한다고 보면 좋을 것이다.

MainActivity는 이름 그대로 앱의 메인 화면을 담당하는 activity이다. remoQon의 메인 화면에는 숫자버튼, 볼륨 및 채널 조작 버튼, 전원 버튼, 음소거 버튼으로 구성된 기본적인 리모컨이 위치하게 된다. 이 기본 리모컨 기능을 수행하는 것이 MainActivity의 주 기능이다.

BroadcastListActivity에서는 서버로부터 채널의 목록을 받아와 데이터베이스에 저장하고 그 내용을 화면에 표시하는 것이 주된 기능이다. 사용자가 원하는 채널을 쉽게 찾을 수 있도록 채널을 카테고리에 따라 구분하고, 채널을 검색할 수 있도록 한다.

DetailedTableActivity는 BroadcastListActivity에서 선택된 방송국에서 당일 방영되는 프로그램의 리스트를 보여주게 된다. 여기서 프로그램을 예약할 수 있고, 예약된 프로그램은 만약 현재 방영 중이라면 즉시 채널이 전환되고, 그렇지 않다면 프로그램의 시작 시간에 맞춰 자동으로 채널을 전환하게 된다.

# **Current Status**

QRemote 앱을 통해 TV를 등록하는 기능을 구현하고, 리모컨의 기본 버튼들(숫자버튼 0~9, 볼륨버튼, 채널버튼, 전원, 조용히)과 그 동작을 구현하였다. 서버에서는 네이버 TV편성표(<http://tvguide.naver.com/>)를 읽어 방송국 정보를 추출하고 그에 따라 프로그램들을 채널 유형(드라마, 다큐멘터리, 교육 등) 및 방송 유형(지상파, 종합편성, 케이블)에 따라 분류하여 저장한다. 앱에서는 서버에서 채널 정보를 받아와 DB에 저장하고, 저장된 채널 정보를 이용해 서버에 편성표 정보에 대한 요청을 보낸 결과를 읽어 원하는 방송국의 편성표를 찾아 볼 수 있다.

# **Future Work**

편성표에서 특정 프로그램을 선택하였을 때 현재 방영 중인 프로그램이라면 즉시 채널을 전환하고 그렇지 않다면 프로그램을 예약한 후 시작 시간이 임박하면 채널을 자동으로 전환하는 기능을 구현해야 한다. 또, 스마트 TV와 연동해서 할 수 있는 작업들인 qwerty 키보드 입력과 자동 채널 변경 기능을 구현해야 한다. 그리고 리모컨 앱을 실행하지 않아도 배경화면에서 기본 리모컨 기능을 사용할 수 있도록 배경화면 위젯을 구현해야 한다. 마지막으로 리모컨 버튼의 배치나 편성표 화면을 수정해 UI를 개선하여야 한다.

# **Division & Assignment of Work**

|  |  |
| --- | --- |
| 항목 | 담당자 |
| 편성표 서버 구현 | 최지헌 |
| QRemote SDK 활용 리모컨 구현 | 정현호 |
| App과 서버 사이의 통신 기능 구현 | 최지헌 |
| 시청하고자 하는 프로그램 리스트 등록 및 관련 기능 구현 | 송시우, 최지헌 |
| 위젯 구현 | 송시우 |
| Smart TV 관련 기능 구현 | 정현호 |
| UI 구현 | 정현호, 송시우, 최지헌 |

# **Schedule**

|  |  |  |
| --- | --- | --- |
| 내용 | 11월 | 12월 |
| 1 | 2 | 3 | 4 | 1 | 2 |
| 프로그램 등록 기능 구현 |  | **v** |  |  |  |  |
| 위젯 기능 구현 |  |  | **v** | **v** |  |  |
| Smart TV 기능 구현 |  |  |  | **v** | **v** |  |
| 앱 구동 테스트 |  |  |  |  | **v** |  |
| 피드백 반영 |  |  |  |  | **v** |  |
| LG전자 행사에서 발표 |  |  |  |  |  | **v** |
| 최종 발표 |  |  |  |  |  | **v** |

# **[Appendix] Detailed Implementation Spec**

## **MainActivity**

### **protected void onCreate(Bundle savedInstanceState)**

Android에서 activity의 상태는 크게 active, paused, stopped의 세 가지로 나눌 수 있다. Active는 현재 화면에 activity가 표시되고 있어 사용자와의 interaction이 가능한 상태이고, paused는 현재 화면에 activity가 표시되고는 있지만 다른 activity에 의해 화면의 일부가 가려져 사용자와의 interaction이 불가능한 상태이다. Stopped는 다른 activity에 의해 화면이 완전히 가려진 상태이다.

onCreate 메소드는 activity를 처음 실행할 때 실행되는 메소드이다. 인자로 받는 savedInstanceState는 초기화를 위하여 사용되는 값으로, 실제 앱의 구동과는 큰 관계가 없다. 여기서는 리모컨 작동을 위한 파워, 채널, 볼륨, 숫자 버튼 등을 등록, 화면에 표시하고 버튼을 눌렀을 때 QRemote SDK를 통해 IR 하드웨어를 작동시키는 역할을 한다.

### **protected void onResume()**

onResume() 메소드는 paused 상태에서 active 상태로 복귀될 때 호출되는 메소드이다. Paused 상태가 되었을 때 디바이스(TV)에 대한 정보가 사라졌을 수 있으므로, 이 메소드에서는 만약 디바이스에 대한 정보가 유실되었다면 디바이스에 대한 정보를 다시 받아오는 작업을 수행한다.

### **protected void onDestroy()**

onDestroy() 메소드는 activity가 종료될 때 호출되는 메소드이다. 만약 IR Blaster가 어떤 디바이스와 연결되어 있다면 종료되기 전에 그 연결을 끊어주어야 하므로 onDestroy() 메소드에서는 이 작업을 수행하게 된다.

### **public boolean onOptionsItemSelected(MenuItem item)**

onOptionsItemSelected 메소드는 Android 기기의 기본 버튼들 중 메뉴 버튼을 눌렀을 때 호출된다. 인자로 받는 MenuItem은 사용자가 메뉴에서 선택한 항목에 대한 정보이다. 메뉴의 구성은 res/menu/main.xml 파일에 정의되어 있으며, 지금까지 구현된 바로는 만약 메뉴 탭에서 새로운 디바이스를 추가하는 버튼을 눌렀을 경우 IR Blaster의 addDevice 메소드를 호출해 QRemote 앱을 통해 새로운 디바이스를 추가하도록 한다.

### **public void getDevices()**

getDevices() 메소드는 디바이스 정보를 가져오는 역할을 한다. 여기에서는 IR Blaster의 getDevices 메소드를 호출해 어떤 디바이스들이 등록되었는지를 확인하고, 디바이스를 불러오는 역할을 한다.

### **public boolean onTouch(View v, MotionEvent event)**

onTouch는 onTouchListener의 onTouch 메소드를 오버라이드한 것으로, activity 내의 버튼을 터치했을 때 어떤 동작을 할지를 결정한다. MainActivity에서는 기본 리모컨 기능을 수행하므로, 숫자 버튼이나 전원, 음소거, 채널, 음량 버튼을 클릭했을 때 적절한 IR 신호를 보내도록 구현한다. MotionEvent event라는 인자로 넘어오게 되는 터치 이벤트의 종류로는 ACTION\_DOWN, ACTION\_OUTSIDE, ACTION\_CANCEL, ACTION\_UP 등이 있는데, ACTION\_DOWN의 경우 터치를 시작했음을 의미한다. 터치 시작이 감지되면 View v로 받은 인자를 이용하여 현재 발생한 터치가 유효한 터치인지(버튼을 터치한 것인지), 어떤 버튼을 터치했는지를 판별한 후 어떤 버튼을 터치했는지에 대한 정보를 getFunctionKeyCode 메소드에 넘겨 준다. 계속 터치하고 있을 때는 신호를 여러 번 보내도록 한다. ACTION\_UP의 경우 터치가 종료되었음을 의미한다. 터치가 종료되면 신호를 보내는 것을 중지한다.

### **private int getFunctionKeyCode(String funcLabel)**

여기에서는 어떤 버튼을 터치했는지에 대한 정보를 통해 IR 신호에 담을 값을 결정한다. LG QRemote SDK에서는 개발 편의를 위해 모든 버튼에 대한 label을 붙여놓았다. 이 메소드에서는 이 label을 인자로 받아 실제로 이 label이 의미하는 동작을 해석하여 int 값으로 반환한다. 이 메소드의 리턴값과 디바이스 정보를 이용해 IR Blaster에서는 디바이스에 IR 신호를 보내게 된다.

### **public void onClick(View v)**

버튼이 클릭되었음을 감지하는 메소드이다. 위에서도 서술했지만 클릭과 터치는 다른 동작이며, 따라서 서로 다른 메소드를 통하여 구현된다. 만약 채널 정보 보기 버튼이 클릭되었다면 BroadcastListActivity를 실행한다.

## **BroadcastListActivity**

### **protected void onCreate(Bundle savedInstanceState)**

MainActivity의 경우와 마찬가지로 savedInstanceState 값을 이용해 초기화를 수행한다. 그 후 layout을 설정하고, DB를 초기화한 후 서버에서 채널 정보를 받아 오도록 한다.

### **private void initDB()**

DB를 초기화하는 메소드이다. 여기에서는 만약 테이블이 이미 존재한다면 테이블을 불러 오고, 그렇지 않다면 새로 테이블을 만든다. 테이블은 id, broadcast\_type, channel\_group, channel\_id, name, user\_no라는 칼럼으로 구성되어 있다. id는 primary key에 해당하는 칼럼이고, broadcast\_type은 방송 유형이 어떻게 되는지, 즉 지상파, 종합편성, 케이블 중 어느 것에 해당하는지를 저장하고, channel\_group은 드라마, 뉴스, 다큐멘터리 등 채널 유형을 저장한다. channel\_id는 각 채널을 관리하기 위해 부여한 고유한 숫자이며, name은 채널의 이름이다. user\_no는 이후에 사용자가 지정하게 될 채널 번호를 의미한다. 지역이나 방송 사업자에 따라 같은 방송일지라도 채널 번호가 다를 수 있기 때문에, user\_no에는 사용자에게 특화된 채널 번호를 저장한다.

### **private static String readUrl(String urlString) throws Exception**

readUrl 메소드는 인자로 주어진 urlString에 해당하는 주소에 요청을 보내 응답을 받아오는 역할을 한다. URL을 연 후 BufferedReader를 이용해 StringBuffer에 서버의 응답을 저장한 후 String으로 변환한 값을 리턴한다.

### **protected Map<String, Cursor> doInBackground(String... urls)**

doInBackground 메소드는 AsyncTask의 doInBackground 메소드를 오버라이드한 것으로, background에서 수행되는 부분이다. 서버에 요청을 보내고 응답을 받은 후 파싱하는 것을 main thread에서 수행하게 되면 작업이 끝날 때까지 사용자가 기다려야 하므로 사용자 경험에 나쁜 영향을 준다. 이 때문에 Android 버전 3.0 이상에서는 main thread에서 서버에 요청을 보내는 것을 금지하고 있다. 따라서 background에서 해당 작업을 수행한 후 작업이 끝나면 알리도록 해 asynchronous하게 서버와 통신하도록 한다. background에서는 readUrl을 통해 읽은 서버 응답의 내용을 JSONParser를 이용해 파싱하고 DB에 저장하는 작업을 한다. 그 후 onPostExecute 메소드에서 ListView에 출력을 수행하도록 적절히 채널 목록을 카테고리화해 리턴한다.

### **protected void onPostExecute(Map<String, Cursor> result)**

onPostExecute 메소드는 background에서의 작업이 끝나면 호출되는 메소드이다. doInBackground 메소드의 리턴값에는 카테고리의 이름과 select 문의 where 절에 사용할 clause들이 저장되어 있는데, 이를 onPostExecute의 인자로 받아 이를 DB에 쿼리로 보내 값을 읽어온 후 화면에 출력하는 작업을 수행한다.

## **DetailedTableActivity**

### **protected void onCreate(Bundle savedInstanceState)**

onCreate 메소드에서는 다른 activity의 경우와 마찬가지로 savedInstanceState 값을 이용해 초기화를 수행하고, DB를 초기화한 후 서버에서 프로그램 정보를 받아오도록 한다.

### **private void initDB()**

BroadcastListActivity에서와 마찬가지로 DB를 초기화하는 작업을 수행하는 메소드이다. 정상적인 사용 환경이라면 DetailedTableActivity는 BroadcastListActivity에서 특정 채널을 선택한 경우 시작되는 activity이므로, 여기에서는 채널 정보와 관련된 테이블이 이미 있다고 가정한다. 따라서 테이블이 없을 경우 예외를 호출한다.

### **private static String readUrl(String urlString) throws Exception**

채널 정보를 서버에 보낼 쿼리의 인자로 담아 서버에 요청을 보낸 후 응답을 받아오는 메소드이다. 동작하는 방식은 BroadcastListActivity의 경우와 동일하다.

### **protected List<ProgramInfo> doInBackground(String... urls)**

BroadcastListActivity의 경우와 같이, background에서 서버에 요청을 보낸 후 파싱하는 작업을 담당하는 메소드이다. 다만 프로그램 정보의 경우 DB에 모두 저장하지는 않으므로 DB에 쓸 필요 없이 단순히 파싱 결과를 리턴한다.

### **protected void onPostExecute(List<ProgramInfo> result)**

background에서의 요청 및 파싱 작업이 끝나면 그 파싱 결과를 화면에 출력하는 역할을 담당하는 메소드이다. 여기에서 프로그램을 선택하면 채널을 전환하거나 예약하는 작업을 하도록 구현해야 한다.