**Embedded System Software**

**<Project 4>**

과목명 : 임베디드 시스템 소프트웨어

담당교수 : 서강대학교 컴퓨터공학과 박 성 용

개발자 : 20151607 정재훈

개발기간 : 2019.06.20 ~ 2019.06.21

1. 개발 목표

실습 시간 때 배운 내용을 활용하여 간단한 java application 프로그램을 작성한다.

1. 개발 내용
2. 2개 Activity를 이용한 Application 구현

해당 Application은 총 두개의 Activity로 이루어져 있으며 처음 앱을 실행했을 때 main Activity는 학번과 Puzzle game버튼을 보여준다. 해당 버튼을 누를 시, Puzzle game을 할 수 있는 Activity이 생성되며 해당 Puzzle Activity에서는 Timer, Text Editor, Puzzle game Button으로 linear layout이 구성되어 있다. 만약 해당 Activity를 종료시키거나 puzzle game을 풀면, 다시 Main Activity로 돌아가게 된다.

1. Puzzle Game 구현

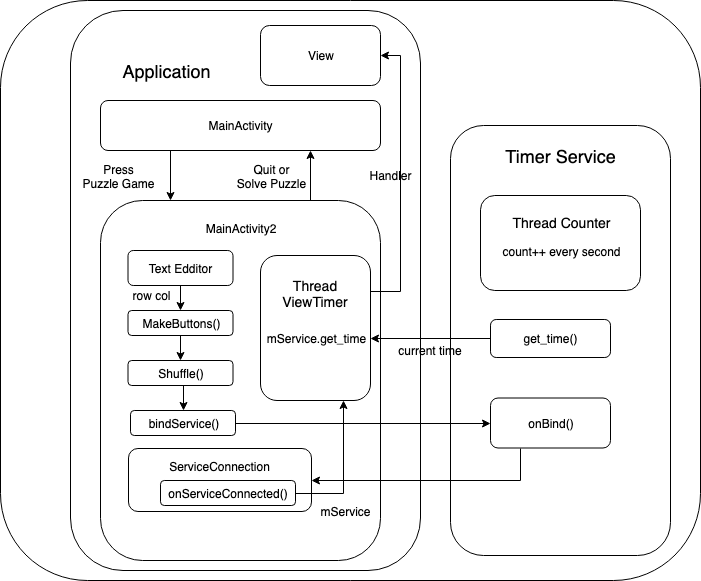
Puzzle Activity에 있는 Text Editor를 통해 row와 col을 입력 받는다. (형식 “row col”) Make Buttons이라고 표시되어 있는 버튼을 클릭하면 해당 row와 col에 따라 puzzle이 자동 생성되며 프로그램이 자동으로 섞어준다. 그때 timer service도 같이 수행되며 text editor 위에 현재 play시간이 “분: 초”로 표시된다. 사용자는 검은색으로 표시된 버튼 주위를 누를 수 있으며 해당 버튼을 누르면 검은색 버튼으로 해당 버튼이 이동한다. 이렇게 조작하여 버튼에 적힌 숫자를 순서대로 배열한다면 게임이 종료되며 다시 main Activity로 돌아가게 된다.

이미 Puzzle game이 진행중인 중에 Text Editor에 새로운 row col을 입력하게 된다면 새로운 게임이 시작되며 timer 역시 초기화된다.

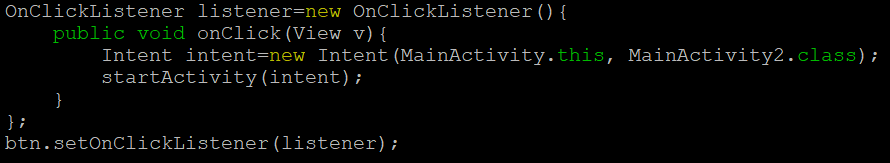
1. Timer Service 구현

Timer Service는 Activity와 상관없이 Background에서 돌아가는 안드로이드 서비스이다. Make Buttons를 누르면 해당 Service가 시작되며 Puzzle game을 풀거나, 해당 Puzzle Activity를 종료시켰을 때 서비스가 종료된다.

1. 구성



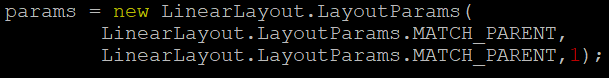
1. 제작 내용
2. 2개 Activity를 이용한 Application 구현

Application은 총 두개의 Activity로 이루어져 있으므로 MainActivity.java에 첫 시작 Activity를 작성하고 MainActivity2.java에 Puzzle game이 수행되는 Activity를 작성한다. MainActivity 첫 줄에 내 학번과 두번째 줄에 Puzzle game이라고 적힌 버튼을 생성한다. 이는 xml파일을 통해 Vertical LinearLayout를 구성하며 button의 listener는 MainActivity.java파일에 작성한다. 버튼이 눌려지면 StartActivity를 통해 MainActivity2를 불러온다. 

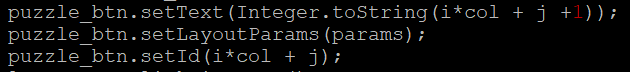
MainActivity2의 view는 첫 줄에 Timer 두번째 줄에는 TextEditor 세번째에는 MakeButons라고 적힌 버튼을 보여주도록 xml을 구성한다. 해당 버튼을 클릭하면 TextEditor로 부터 getText() 함수를 통해 적힌 값을 읽어온다. 적힌 값을 space bar로 split하고 row col을 int로 변환하여 적용한다. 아래에 Puzzle에 대한 버튼을 row \* col 만큼을 생성하고 Shuffle을 진행한다. 이때 Timer Service를 BindService로 수행한다.

1. Puzzle Game 구현
2. MakeButtons()

row만큼의 Horizontal한 Linear Layout을 동적으로 생성하고, 해당 Linear Layout 안에 Col만큼의 Button을 동적으로 생성한다. 이때, Linear Layout들과 해당 버튼은 크기가 모두 같아야 한다. 따라서 weight를 사용하여 크기를 맞추고, MATCH\_PARENT를 통해 나머지 화면을 채울 수 있도록 한다.



총 버튼은 row\*col만큼 생성되며 Button마다 TextView를 초기화해주고 button click action을 위해버튼마다 id를 부여한다. 1번째 버튼부터 TextView는 1부터 시작하며, Id는 0부터 시작한다.



해당 button마다 Listener를 달아 놓으며 Listener는 Swap\_Button()을 통해 버튼끼리 의 swap을 진행하고, Check\_Answer()를 통해 현 puzzle이 정답인지 확인한다. 만약 정답이면 finish()를 통해 해당 Activity를 종료한다.

마지막으로 마지막 Button은 SetBackGroundColor()를 통해 검은색으로 칠하며 LinearLayout과 Button의 addView를 통해 화면에 나오도록 한다.

1. shuffle()

MakeButtons()를 수행한 후 Puzzle은 정답 상태이며 이를 Shuffle한다.

Solvable를 유지하기 위해, 정답상태에서 가능한 움직임 안에서 섞어야 한다. 따라서 direction을 총 4개(동서남북)을 준 후 Black으로 칠해진 버튼 주위를 랜덤하게 하나를 고른다, 해당 버튼이 유효한 버튼이라면 해당 버튼을 누른 것처럼 Swap\_Button()을 동작 시켜 총 row \* col \* 10번 섞도록 한다.

1. Swap\_Button(int i, int j)

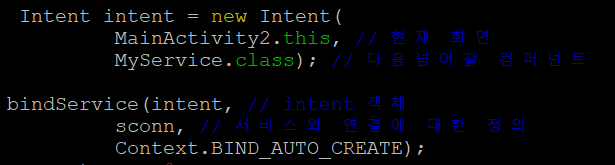
i행 j열의 버튼을 눌렀을 때 Puzzle의 동작을 수행하는 함수이다. 만약 해당 i행 j열이 puzzle의 유효 범위에서 벗어나 있거나 black button에 인접해 있지 않다면 아무것도 수행하지 않는다. 인접해 있다면, Black Button과 Text만 뒤바꾸고, 해당 Button은 까맣게 칠하고 Black Button은 색을 되돌린다. 이때 Black Button의 row와 col은 항상 전역으로 유지하고, text만 바꿀 뿐 id는 그대로 유지시킨다.

1. check\_answer()

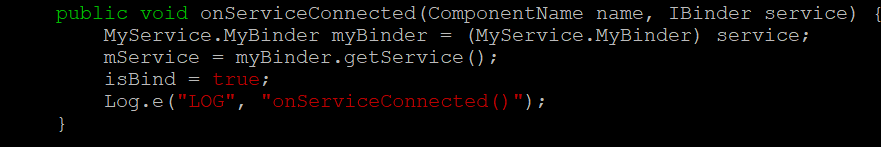
Puzzle이 정답이라는 뜻은 button의 id와 button의 text가 일치한다는 것이다. 따라서 총 row \* col개의 button을 조사하여 Text와 Id가 같은 값을 가지고 있는지 조사한다. 만약 가지고 있다면 true를 return하고 같지 않다면 false를 return한다.

1. Timer Service 구현

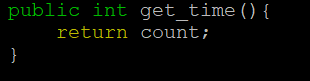
Timer Service는 Bind Service로 구현하며 MyService.java에 구현한다. MakeButtons 버튼을 눌렀을 때 Service를 시작하며 Intent를 생성하여 전달한다.



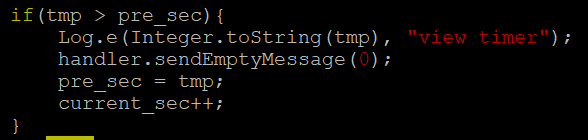
이때 현재 Activity에서는 onServiceConnected()함수가 수행되며 MyService의 Binder 객체를 받아오고 Binder객체의 Service객체를 참조가 가능해진다.

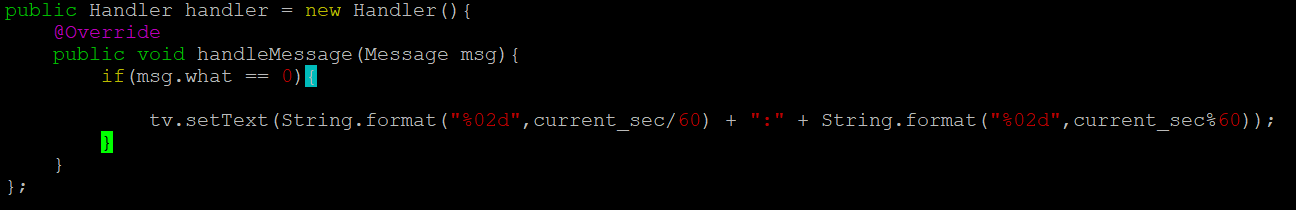


해당 Timer 서비스는 1초마다 count를 1씩 늘려주는데 Service가 시작되면 1초마다 1씩 증가시키는 Thread를 실행하고 이는 Service가 종료될 때까지 계속된다. 해당 count변수는 MyService class의 변수라 이 변수에 접근할 방법이 필요하다. 따라서 이를 Service function으로 구현한다.



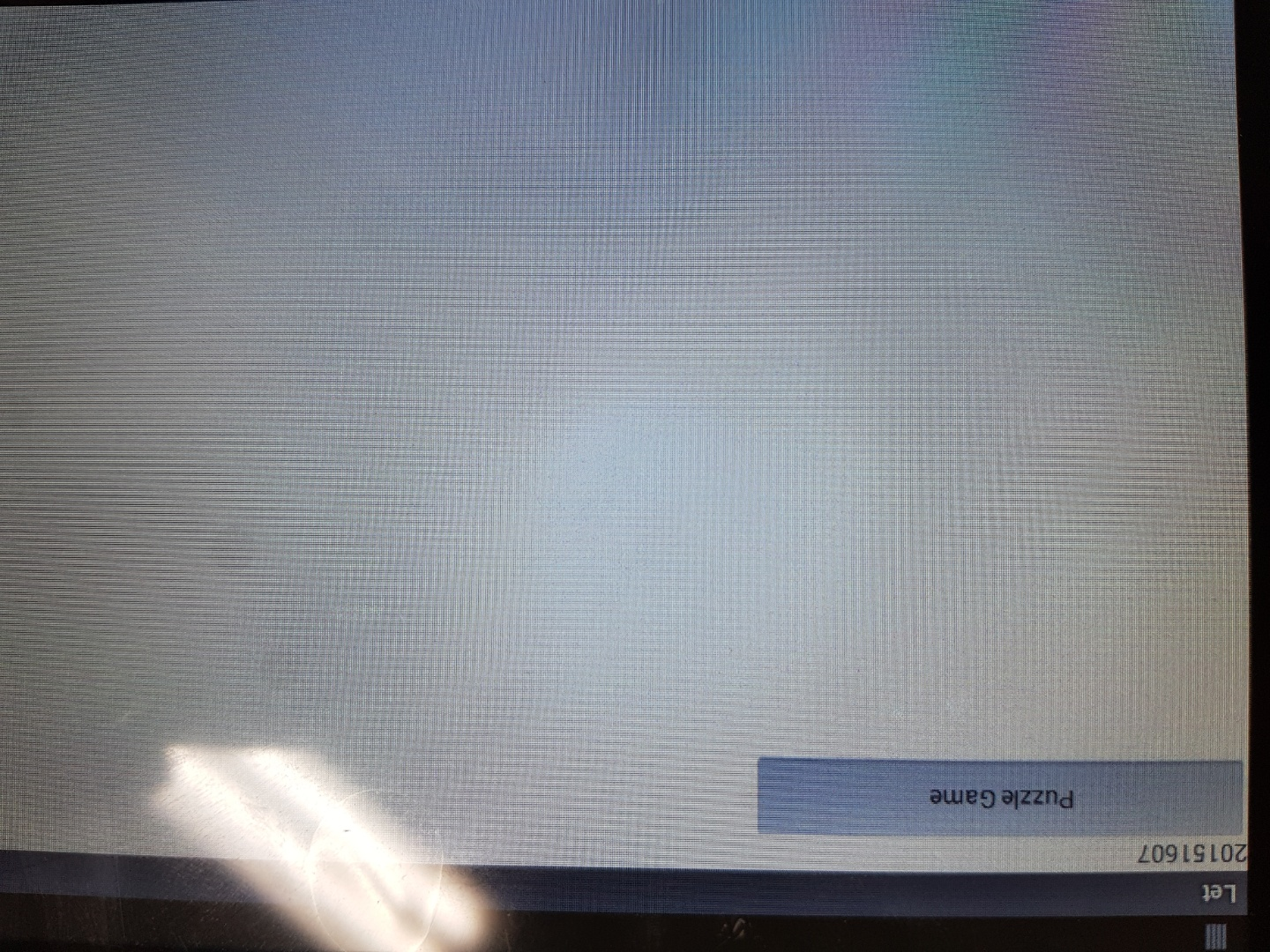
Activity에서 역시 Thread를 생성하여 0.1초마다 MyService의 count를 읽어 이를 해당 View에 뿌려주도록 한다. 이를 위해 Binder를 통해 얻은 mService로 get\_time()을 호출하여 현재 count를 읽는다. 해당 count를 분 초로 변환하여 View에 뿌리기 위해서 Thread에서는 Handler를 이용해야 한다. 따라서 Handler를 정의하고 Handler를 통한 Message전달로 Timer를 출력한다.

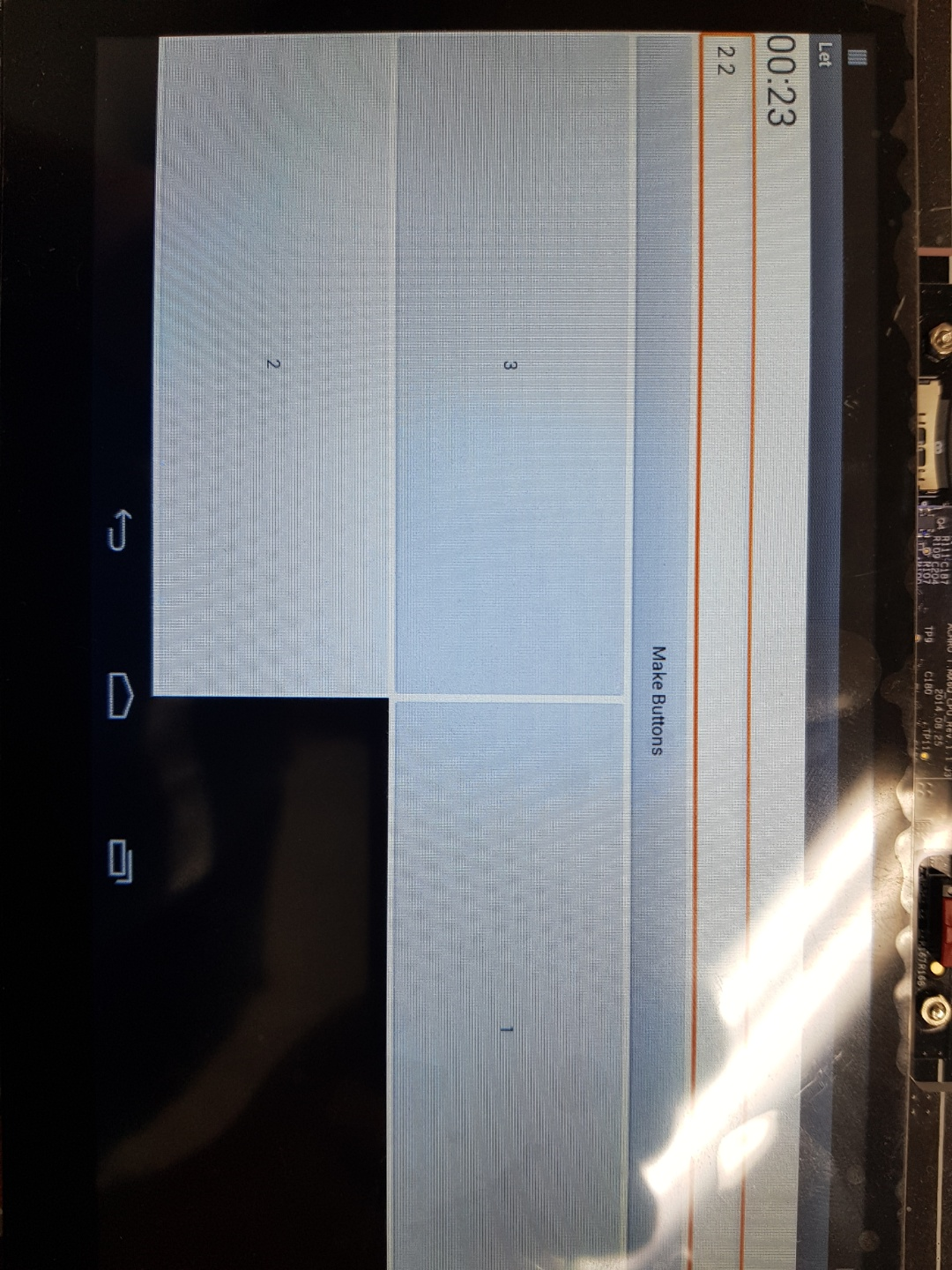
 Thread의 Message 전달



Handler로 받아 Text View로 출력

1. 실행화면





1. 기타

**느낀 점**  
Android 단에서 Linear Layout이 어떻게 동작하는지 정확히 알 수 있었고, 안드로이드에서의 Service 동작방식을 알 수 있었다. Java와 Android를 전혀 몰랐지만 계속 구글링하여 마침내 프로젝트를 완성하니 뿌듯함을 느낄 수 있었다.