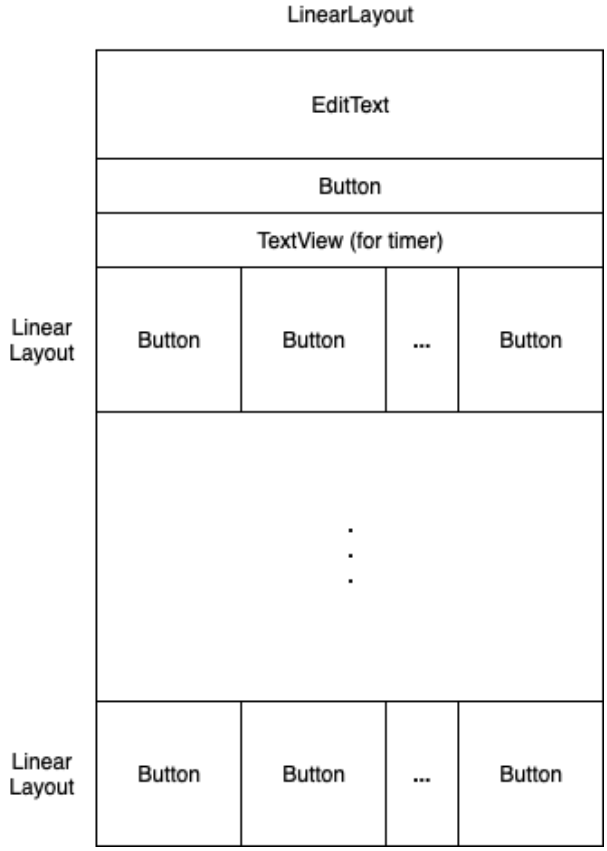
Embedded System Software
HW #4

20151623
한상구

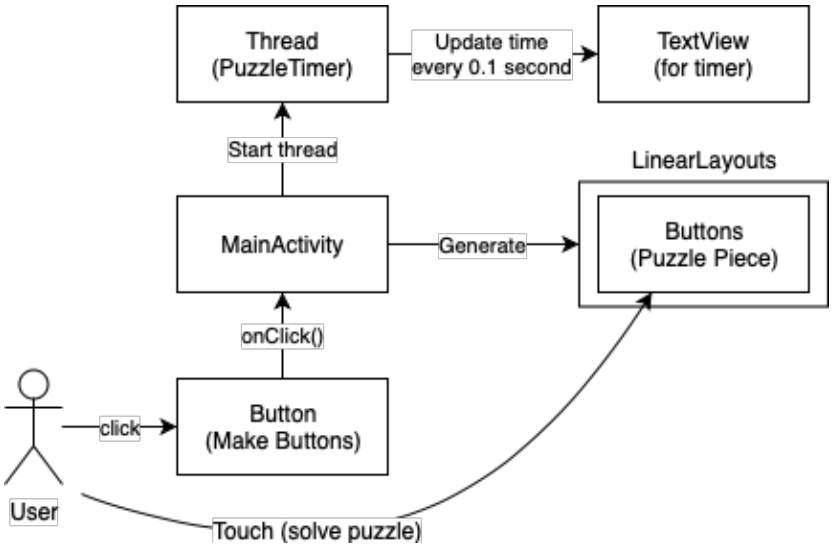
1. 프로젝트 목표

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

2. 설계



<그림 1 - Application View 구조도>



<그림 2 - Application flowchart>

위 그림 1, 2 와 같이 프로젝트를 설계, 구현하였습니다.

Touch button (Make buttons button) → Initiate thread & generate puzzle board → Clear puzzle board when solved 의 흐름이며, 세부 구현 내용은 아래 [3. 기능 구현] 에서 서술하도록 하겠습니다.

3. 기능 구현

a. Application

Main activity 가 생성될 때, “Make Buttons” 버튼에 OnClickListener 를 통해 퍼즐 초기화 및 타이머 시작을 하도록 했습니다.

입력된 값이 2 개 초과일 때, Row 와 Column 값이 범위 밖을 벗어나거나 숫자가 아닐 때에는 새로 게임을 시작하지 않도록 try catch 를 통해 구현했습니다.

퍼즐 보드를 만들 때에는, 1 차원 정수 배열인 `int[] puzzleStats` 에 랜덤하게 섞인 값을 저장하고, 이를 통해 초기 섞인 상태를 만들었습니다.

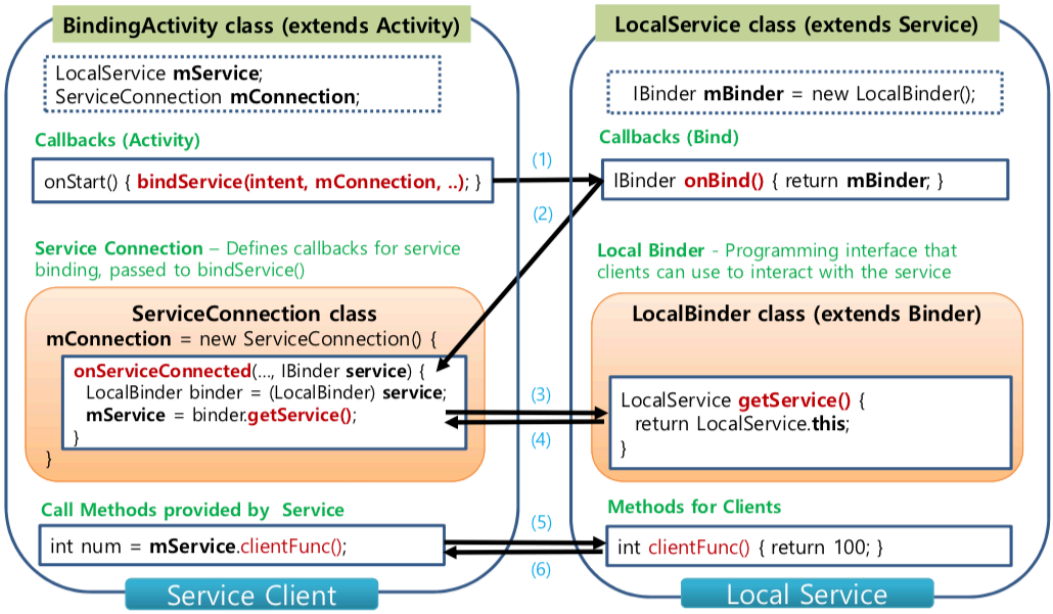
퍼즐 보드 상의 버튼이 클릭 될 때마다, 주변 4 방향에 이동할 수 있는 곳이 있는지 확인하고 이동할 수 있다면 `puzzleStats` 에 상태를 업데이트 합니다. 이후 퍼즐이 풀렸는지 확인한 다음, 풀렸다면 보드를 지우고 아직 풀리지 않은 상태라면 업데이트된 상태를 기반으로 보드를 다시 그리도록 했습니다.

그림 2 에서 Thread 가 TextView 에 표시할 시간은 아래 **b. Service(timer)**에서 가져오며, 0.1 초마다 시간을 업데이트 할 수 있도록 했습니다.

b. Service (timer)

그저 시간을 재기만 한다면 Main Activity 와 상호작용이 없기 때문에 Unbounded service 로 작성하려 했으나 TextView 를 업데이트 해야하는 작업이 존재하기 때문에 (inflater 등을 이용해 여러 시도를 해보았으나 성공하지 못했습니다) Bounded service 로 작성, Main Activity 에서 시간 정보를 가져와 업데이트 하는 방향으로 구현했습니다.

시간 출력은 그림 1 에서 TextView (for timer)가 위치한 곳에 하도록 구현했습니다.



<그림 3 - Binding Process 흐름도[1]>

위 흐름도와 같은 순서로 Main activity (client) - Service 간의 바인딩 및 호출이 이루집니다.

아래는 위 흐름도 각 구성에 해당하는 실제 구현 코드를 가져왔습니다.

```
Intent intent = new Intent(
                MainActivity.this,
                TimerService.class
                );
bindService(intent, conn, Context.BIND_AUTO_CREATE);
```

<Callbacks (Activity)>

```
@Override
public IBinder onBind(Intent intent) {
        System.out.println("onBind");
        return mBinder;
}
```

<Callbacks (Bind)>

```
class TimerBinder extends Binder {
        TimerService getService() {
                return TimerService.this;
        }
}
```

<Local Binder>

[1] 강의자료 `BindService Example.pdf` p.29

```
ServiceConnection conn = new ServiceConnection() {
        @Override
        public void onServiceConnected(ComponentName name, IBinder service) {
                TimerService.TimerBinder mb = (TimerService.TimerBinder) service;
                ms = mb.getService();
                isService = true;
        }

        @Override
        public void onServiceDisconnected(ComponentName name) {
                isService = false;
        }
};
```

<Service Connection>

```
        public int getElapsed() {
                return elapsed;
        }
```

<Method for client>

```
int elapsed = ms.getElapsed();
```

<Call Methods provided by Service>

Service 에서는 `elapsed` 에 시간이 얼마나 지났는지 지속적으로 업데이트 하며 (unit time: 0.1s), Client 가 Service bind 이후 시간이 얼마나 흘렀는지 확인할 수 있도록 `getElapsed()`를 구현했습니다.

Client 인 main activity 에서는 서비스 바인딩 이후 Thread 를 통해 0.1 초마다 `getElapsed()`를 이용 시간 정보를 가져오고, 게임 진행중에 지속적으로 시간 정도를 TextView 에 업데이트 하도록 구현했습니다.