모바일 프로그래밍 및 실습

컴퓨터공학과
21720831 김준서

목차

MyShortCutApp

홈 화면에 배치된 위젯에서 패턴을 입력하면
패턴에 매핑된 어플리케이션을 실행

- 홈화면에 배치된 어플리케이션의 수를 줄일 수 있다.
- 앱 서랍에서 앱을 찾는 시간을 줄일 수 있다.

=> 사용자의 편의성을 증진

MyDBHelper.java

SQLite를 사용하기 위해 SQLiteOpenHelper를 상속.

void Insert와 void Delete를 구현하여
DB에 저장하고 삭제할 수 있도록 한다.

```java
public class MyDBHelper extends SQLiteOpenHelper{

    static final String DATABASE_NAME = "ShortCutApp.db";

    public MyDBHelper(Context context, int version) {
        super(context, DATABASE_NAME, factory: null, version);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("CREATE TABLE ShortCutAPP(appName TEXT, appPackage TEXT, pattern TEXT)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS ShortCutAPP");
        onCreate(db);
    }

    public void insert(String appName, String appPackage, String pattern) {
        SQLiteDatabase db = getWritableDatabase();
        db.execSQL("INSERT INTO ShortCutAPP VALUES('" + appName + "', '" + appPackage + "', '" + pattern + "')");
        db.close();
    }

    public void Delete(String appName) {
        SQLiteDatabase db = getWritableDatabase();
        db.execSQL("DELETE FROM ShortCutAPP WHERE APPNAME = "+ "'" + appName + "'");
        db.close();
    }
```

AppListItem.java

설치된 어플의 목록과
매핑된 어플리케이션의 객체를 생성하기 위한 클래스.

생성자와 getter & setter로 이루어져있다.

```java
public class AppListItem implements Serializable {
    private Drawable appIcon;
    private String appName;
    private String appPackageName;
    private String pattern;

    public AppListItem(String appName, String appPackageName, String pattern) {
        this.appName = appName;
        this.appPackageName = appPackageName;
        this.pattern = pattern;
    }

    public AppListItem(Drawable appIcon, String appName, String appPackageName) {
        this.appIcon = appIcon;
        this.appName = appName;
        this.appPackageName = appPackageName;
    }

    public String getAppPackageName() { return appPackageName; }

    public void setAppPackageName(String appPackageName) { this.appPackageName = appPackageName; }

    public Drawable getAppIcon() { return appIcon; }

    public void setAppIcon(String Drawable) { this.appIcon = appIcon; }

    public String getAppName() { return appName; }

    public void setAppName(String appName) { this.appName = appName; }

    public String getPattern() { return pattern; }

    public void setPattern(String pattern) {
        this.pattern = pattern;
    }
}
```

short_cut_widget.xml

홈 화면에 배치될 위젯의 디자인 xml .
총 4개의 버튼과 확인 버튼으로 이루어져있다.
상단 버튼은 왼쪽부터 값 1, 값2,
하단 버튼은 값 3, 값4 를 가지고 있다.

ShortCutWidget.java

Widget을 구현하기 위해 AppWidgetProvider를 상속.

홈 위젯에서 버튼이 클릭되면 onReceive에서
해당 버튼에 대한 기능을 수행한다.

패턴 버튼이 클릭되면 클릭된 시각을 기록하고
이전에 클릭된 버튼의 시각과 차이가 2초 이상이라면
StringBuillder pattern을 clear한다.
이후 버튼에 해당하는 값을 pattern 에 추가한다.

Ok 버튼이 클릭되면 pattern을 담아 StartApp.class을 실행한다.

```
public class ShortCutWidget extends AppWidgetProvider {
    private static final String MyOnClick1 = "myOnClickTag1";
    private static final String MyOnClick2 = "myOnClickTag2";
    private static final String MyOnClick3 = "myOnClickTag3";
    private static final String MyOnClick4 = "myOnClickTag4";
    private static final String MyOnClick_OK = "myOnClickTag_OK";


    private static StringBuilder pattern = new StringBuilder();

    private static long startTime = System.currentTimeMillis();
    private static long endTime = System.currentTimeMillis();
    private static long termTime;

    @Override
    public void onUpdate(Context context, AppWidgetManager appWidgetManager, int[] appWidgetIds) {
        for (int appWidgetsId : appWidgetIds) {
            RemoteViews views = new RemoteViews(context.getPackageName(), R.layout.short_cut_widget);
            views.setOnClickPendingIntent(R.id.widget_btn1, getPendingSelfIntent(context, MyOnClick1));
            views.setOnClickPendingIntent(R.id.widget_btn2, getPendingSelfIntent(context, MyOnClick2));
            views.setOnClickPendingIntent(R.id.widget_btn3, getPendingSelfIntent(context, MyOnClick3));
            views.setOnClickPendingIntent(R.id.widget_btn4, getPendingSelfIntent(context, MyOnClick4));
            views.setOnClickPendingIntent(R.id.widget_ok, getPendingSelfIntent(context, MyOnClick_OK));

            appWidgetManager.updateAppWidget(appWidgetsId, views);
        }
    }

    protected PendingIntent getPendingSelfIntent(Context context, String action) {
        Intent intent = new Intent(context, getClass());
        intent.setAction(action);
        return PendingIntent.getBroadcast(context,  requestCode: 0, intent,  flags: 0);
    }

    @Override
    public void onReceive(Context context, Intent intent) {
        super.onReceive(context, intent);

        if (MyOnClick1.equals(intent.getAction())) {
            endTime = System.currentTimeMillis();
            termTime = (endTime - startTime) / 100;

            if (termTime > 20) {
                pattern.setLength(0); //  pattern 비우기
            }

            pattern.append('1');
            startTime = System.currentTimeMillis();
        } else if (MyOnClick2.equals(intent.getAction())) {
            endTime = System.currentTimeMillis();
```

```
            startTime = System.currentTimeMillis();
        }else if(MyOnClick_OK.equals(intent.getAction())){
            Intent start_intent = new Intent(context, startApp.class);
            start_intent.putExtra( name: "pattern", pattern.toString());
            start_intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
            context.startActivity(start_intent);
        }
    }
}
```

StartApp.java

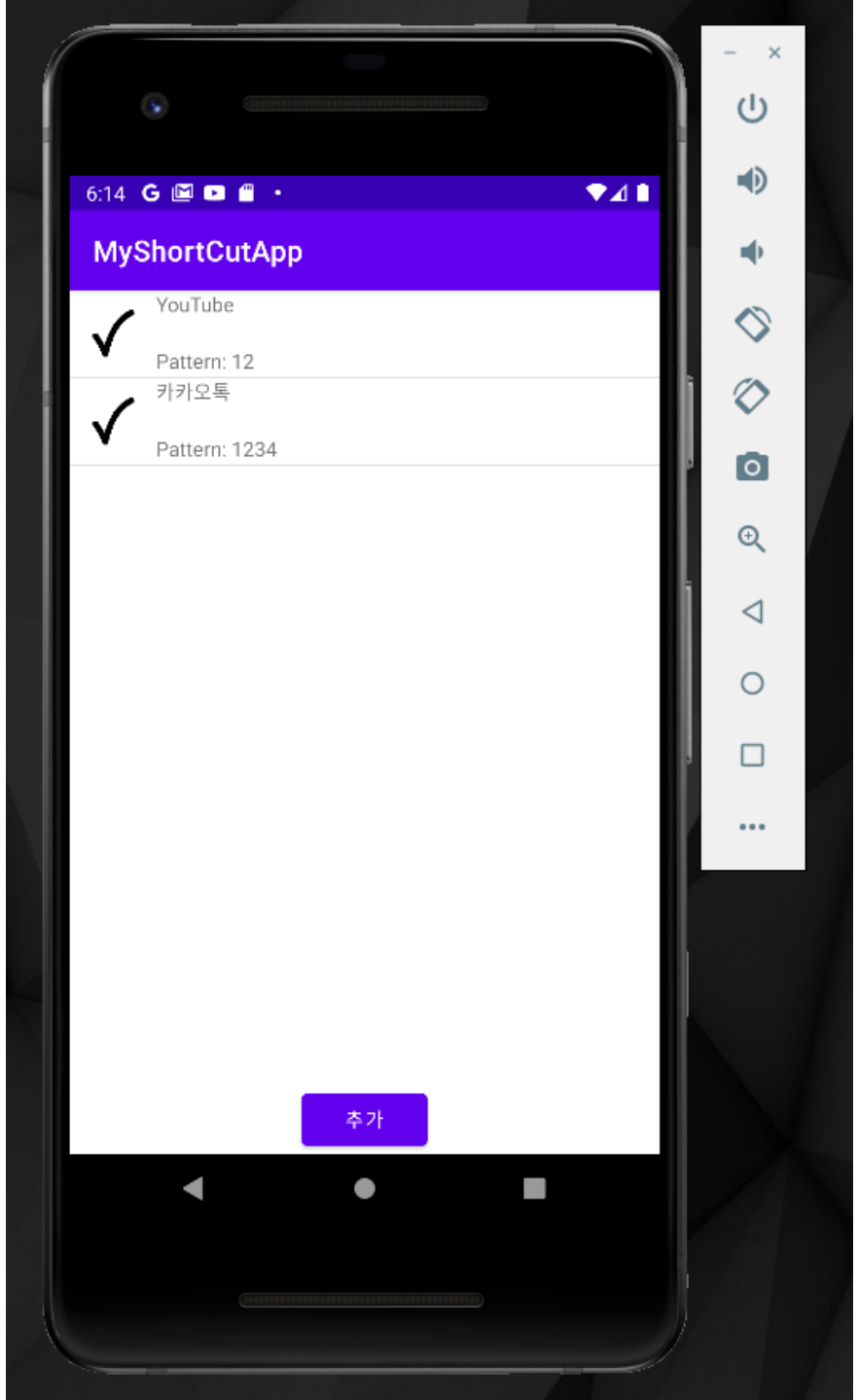
홈 화면에서 패턴을 입력했을 때,
어플리케이션을 실행하기 위한 java 파일.

ShortCutWidget.java에서 받아온 pattern을 가져와
그 pattern에 매핑된 어플리케이션을 실행하고 해당하는 어플리케이션의 이름을 메시지로 출력한다.

매핑 이후 앱을 삭제하는 등,
스마트폰에 어플이 존재 하지 않으면
'앱이 설치되어 있지 않습니다.'라는 메시지를 출력한다.

매핑 되어 있지 않은 패턴이라면 '패턴이 일치하지 않습니다.'라는 메시지를 출력한다.

```java
public class StartApp extends AppCompatActivity {

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Intent intent = getIntent();
        String pattern = intent.getStringExtra( name: "pattern");
        int index;
        for (index = 0; index < shortCut_apps.size(); index++) {
            if (pattern.equals(shortCut_apps.get(index).getPattern())) {
                Toast.makeText(getApplicationContext(),  text: shortCut_apps.get(index).getAppName() + "을 실행합니다.", Toast.LENGTH_SH

                try{
                    Intent appintent = getPackageManager().getLaunchIntentForPackage(shortCut_apps.get(index).getAppPackageName());
                    appintent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                    startActivity(appintent);
                    finish();
                    return;
                }
                catch (Exception e){

                    Toast.makeText(getApplicationContext(),  text: "앱이 설치되어 있지 않습니다.", Toast.LENGTH_SHORT).show();
                    finish();
                    return;
                }
            }
        }

        if(index == shortCut_apps.size()){
            Toast.makeText(getApplicationContext(),  text: "패턴이 일치하지 않습니다.", Toast.LENGTH_SHORT).show();
            Intent homeIntent = new Intent(Intent.ACTION_MAIN);
            homeIntent.addCategory(Intent.CATEGORY_HOME);
            homeIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
            startActivity(homeIntent);
        }
        finish();
    }
}
```



activity_main.xml
short_cut_app_list_item.xml

앱 실행 시 가장 먼저 보이는 화면 디자인 xml.
activity_main.xml에는 Listview와 Button이 있으며,
short_cut_app_list_item.xml는 Listview에 들어갈
매핑 객체를 담을 디자인 xml 이다.

AppListAdapter_ShortCutApp.java

매핑된 어플리케이션의 목록을 출력하기 위한
ListView Adapter 클래스.

```java
class AppListAdapter_ShorCutApp extends BaseAdapter {

    Context mContext = null;
    LayoutInflater mLayoutInflater = null;
    ArrayList<AppListItem> app;

    public AppListAdapter_ShorCutApp(Context context, ArrayList<AppListItem> data) {
        mContext = context;
        app = data;
        mLayoutInflater = LayoutInflater.from(mContext);
    }

    @Override
    public int getCount() { return app.size(); }

    @Override
    public long getItemId(int position) { return position; }

    @Override
    public AppListItem getItem(int position) { return app.get(position); }

    @Override
    public View getView(int position, View converView, ViewGroup parent) {
        View view = mLayoutInflater.inflate(R.layout.short_cut_app_list_item,  root: null);

        TextView appName = (TextView)view.findViewById(R.id.appName);
        TextView pattern = (TextView)view.findViewById(R.id.pattern);

        appName.setText(app.get(position).getAppName());
        pattern.setText("Pattern: " + app.get(position).getPattern());

        return view;
    }
}
```

MainActivity.java

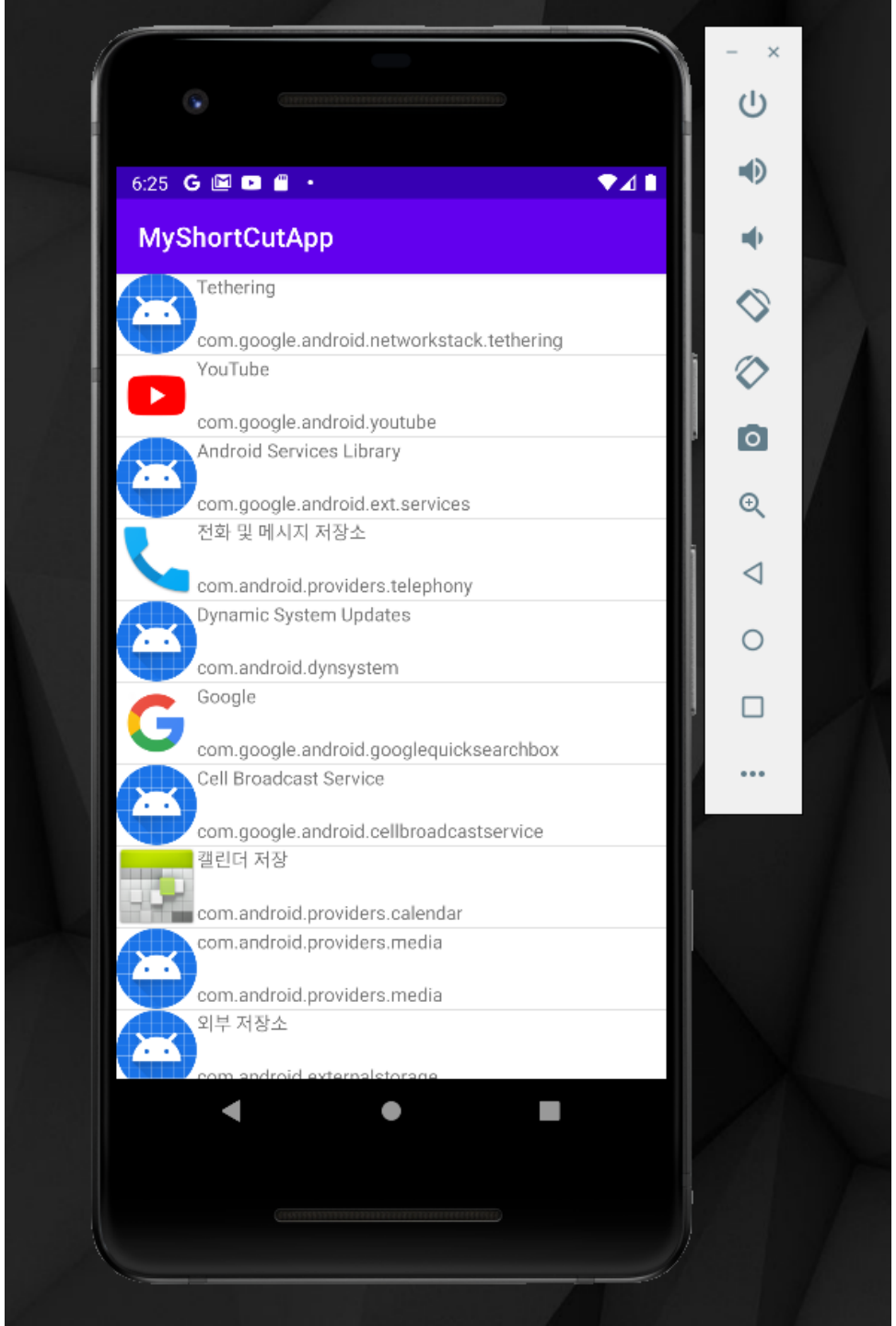
Activity_main.xml 에 대한 java 파일.

MyDBHelper를 생성하여 DB에 등록된 매핑된 어플리케이션 목록을 가져와 ListView에 출력한다.

하단에 위치하는 버튼인 main_button_add(추가 button)을 클릭하면 SelectApp.class를 출력한다.

목록 중 특정 항목을 롱 클릭하면 그 항목을 DB에서 삭제하고 MainAcitivity.java를 다시 출력한다.

```java
public class MainActivity extends AppCompatActivity {
    static ArrayList<AppListItem> shortCut_apps = new ArrayList<~>();
    Button main_btn_add;
    MyDBHelper dbHelper;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        main_btn_add = (Button) findViewById(R.id.main_btn_add);
        dbHelper = new MyDBHelper( context: MainActivity.this,  version: 1);

        dbHelper.readDB();
        ListView appList = (ListView) findViewById(R.id.main_appListView);
        final AppListAdapter_ShorCutApp myAdapter = new AppListAdapter_ShorCutApp( context: this, shortCut_apps);
        appList.setAdapter(myAdapter);

        main_btn_add.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(getApplicationContext(), SelectApp.class);
                startActivity(intent);
            }
        });

        appList.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {
            @Override
            public boolean onItemLongClick(AdapterView<?> parent, View view, int position, long id) {
                dbHelper.Delete(myAdapter.getItem(position).getAppName());

                Toast.makeText(getApplicationContext(),  text: "삭제되었습니다.", Toast.LENGTH_SHORT).show();

                Intent intent = new Intent(getApplicationContext(), MainActivity.class);
                intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                startActivity(intent);
                return false;
            }
        });

    }

}
```



Select_app.xml
App_list_item.xml

설치된 어플 목록을 보여주는 디자인 xml.

Select_app.xml에는 Listview가 있고,

App_list.xml에는 Listview에 담길 객체의 디자인 xml이며

어플의 이름과 Package Name을 가지고 있다.

AppListAdapter_SelectApp.java

설치된 어플리케이션의 목록을 출력하기 위한
ListView Adapter 클래스.

```java
public class AppListAdapter_SelectApp extends BaseAdapter {
    Context mContext = null;
    LayoutInflater mLayoutInflater = null;
    ArrayList<AppListItem> app;

    public AppListAdapter_SelectApp(Context context, ArrayList<AppListItem> data) {
        mContext = context;
        app = data;
        mLayoutInflater = LayoutInflater.from(mContext);
    }

    @Override
    public int getCount() { return app.size(); }

    @Override
    public long getItemId(int position) { return position; }

    @Override
    public AppListItem getItem(int position) { return app.get(position); }

    @Override
    public View getView(int position, View converView, ViewGroup parent) {
        View view = mLayoutInflater.inflate(R.layout.app_list_item,  root: null);

        ImageView appIcon = (ImageView)view.findViewById(R.id.appIcon);
        TextView appName = (TextView)view.findViewById(R.id.appName);
        TextView appPackageName = (TextView)view.findViewById(R.id.appPackageName);

        appIcon.setImageDrawable(app.get(position).getAppIcon());
        appName.setText(app.get(position).getAppName());
        appPackageName.setText(app.get(position).getAppPackageName());


        return view;
    }
}
```

SelectApp.java

Select_app.xml 에 대한 java 파일.

PackageManager를 이용하여 설치된 어플리케이션의
Icon과 이름, 패키지 명을 가지고와 ListView에 출력한다.

특정 어플리케이션을 클릭했을 때
이미 등록된 어플리케이션이면 메시지를 출력하고

그렇지 않으면 어플의 이름과 패키지명을 intent에 담아
SetPattern.class를 실행한다.

```
import static com.example.shortcut2.MainActivity.shortCut_apps;

public class SelectApp extends AppCompatActivity {

    ArrayList<AppListItem> apps = new ArrayList<~>();
    AppListItem app;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.select_app);


        // 설치된 어플리케이션 리스트 취득
        PackageManager pm = getPackageManager();
        List<ApplicationInfo> list = pm.getInstalledApplications( flags: 0);

        for (ApplicationInfo applicationInfo : list) {
            Drawable appIcon = applicationInfo.loadIcon(pm); // 앱 아이콘
            String appName = String.valueOf(applicationInfo.loadLabel(pm));    // 앱 이름
            String appPackageName = applicationInfo.packageName;   // 앱 패키지

            app = new AppListItem(appIcon, appName, appPackageName);
            apps.add(app);

        }

        ListView appList = (ListView) findViewById(R.id.listview);
        final AppListAdapter_SelectApp myAdapter = new AppListAdapter_SelectApp( context: this,apps);
        appList.setAdapter(myAdapter);

        appList.setOnItemClickListener(new AdapterView.OnItemClickListener(){
            @Override
            public void onItemClick(AdapterView parent, View v, int position, long id){
                for(int i = 0; i<shortCut_apps.size(); i++){
                    if(shortCut_apps.get(i).getAppName().equals(myAdapter.getItem(position).getAppName()) ){
                        Toast.makeText(getApplicationContext(), text: "등록되어 있는 어플입니다.", Toast.LENGTH_SHORT).show();
                        return;
                    }
                }


                Intent intent = new Intent(getApplicationContext(), SetPattern.class);
                intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                intent.putExtra( name: "appName", myAdapter.getItem(position).getAppName());
                intent.putExtra( name: "appPackageName", myAdapter.getItem(position).getAppPackageName());

                startActivity(intent);
            }
        });

    }
```

set_pattern.xml

어플리케이션에 매핑할 패턴을 등록할 때 출력할 디자인 xml.

Button상단은 왼쪽부터 값 1, 값2,

Button 하단은 왼쪽부터 값3, 값4 를 가지고 있다.

SetPattern.java

set_pattern.xml 에 대한 java 파일.

Button을 누르면 해당하는 값을 StringBuilder pattern에 append 한다.

btn_ok(확인) 버튼을 클릭했을 때,
사용 중인 패턴이라면 메시지를 출력하고

그렇지 않으면
어플리케이션의 이름과 패키지명, 입력한 패턴을 intent에 담아
RegPattern.class를 실행한다.

```
public class SetPattern extends AppCompatActivity {
    private static StringBuilder pattern = new StringBuilder();
    Button btn1, btn2, btn3, btn4, btn_ok;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        pattern.setLength(0);
        super.onCreate(savedInstanceState);
        setContentView(R.layout.set_pattern);

        btn1 = (Button) findViewById(R.id.setPattern_btn_btn1);
        btn2 = (Button) findViewById(R.id.setPattern_btn_btn2);
        btn3 = (Button) findViewById(R.id.setPattern_btn_btn3);
        btn4 = (Button) findViewById(R.id.setPattern_btn_btn4);
        btn_ok = (Button) findViewById(R.id.setPattern_btn_ok);

        btn1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) { pattern.append('1'); }
        });

        btn2.setOnClickListener(new View.OnClickListener() {
```

```
        btn_ok.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                for(int i = 0; i<shortCut_apps.size(); i++){
                    if(shortCut_apps.get(i).getPattern().equals(pattern.toString()) ){
                        Toast.makeText(getApplicationContext(), text: "등록되어 있는 패턴입니다.", Toast.LENGTH_SHORT).show();
                        pattern.setLength(0);
                        return;
                    }
                }
                Intent preIntent = getIntent();
                Intent intent = new Intent(getApplicationContext(), RegPattern.class);
                intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                intent.putExtra( name: "appName",preIntent.getStringExtra( name: "appName"));
                intent.putExtra( name: "appPackageName",preIntent.getStringExtra( name: "appPackageName"));
                intent.putExtra( name: "firstPattern", pattern.toString());

                startActivity(intent);
            }
        });
```

reg_pattern.xml

set_pattern에서 입력한 패턴을 확인하기 위한 디자인 xml.

Button 상단은 왼쪽부터 값 1, 값2,

Button 하단은 왼쪽부터 값3, 값4 를 가지고 있다.

RegPattern.java

reg_pattern.xml 에 대한 java 파일.

Button을 누르면 해당하는 값을 StringBuilder pattern에 append 한다.

btn_reg(등록) 버튼을 클릭하면
SetPattern.java에서 입력한 pattern(firstPattern)을 가져와
pattern과 비교하여 같으면 DB에 Insert 하고
MainActivity.class 를 실행한다.

같지 않으면 패턴이 일치하지 않는다는 메시지를 출력하고
SeleteApp.class를 실행한다.

```java
public class RegPattern extends AppCompatActivity {
    private static StringBuilder pattern = new StringBuilder();
    Button btn1, btn2, btn3, btn4, btn_reg;
    MyDBHelper dbHelper;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        pattern.setLength(0);
        super.onCreate(savedInstanceState);
        setContentView(R.layout.reg_pattern);

        btn1 = (Button) findViewById(R.id.regPattern_btn_btn1);
        btn2 = (Button) findViewById(R.id.regPattern_btn_btn2);
        btn3 = (Button) findViewById(R.id.regPattern_btn_btn3);
        btn4 = (Button) findViewById(R.id.regPattern_btn_btn4);
        btn_reg = (Button) findViewById(R.id.regPattern_btn_reg);

        dbHelper = new MyDBHelper(getApplicationContext(), version: 1);

        btn1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) { pattern.append('1'); }
        });

        btn2.setOnClickListener(new View.OnClickListener() {
```

```java
        btn_reg.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent preIntent = getIntent();

                if (preIntent.getStringExtra( name: "firstPattern").equals(pattern.toString())) {

                    Toast.makeText(getApplicationContext(), text: "등록이 완료되었습니다.", Toast.LENGTH_SHORT).show();
                    Intent intent = new Intent(getApplicationContext(), MainActivity.class);

                    dbHelper.insert(preIntent.getStringExtra( name: "appName"), preIntent.getStringExtra( name: "appPackageName"), pa
                    Toast.makeText(getApplicationContext(), text: "DB등록완료", Toast.LENGTH_SHORT).show();
                    intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                    startActivity(intent);
                } else {
                    Toast.makeText(getApplicationContext(), text: "패턴이 일치하지 않습니다..", Toast.LENGTH_SHORT).show();
                    Intent intent = new Intent(getApplicationContext(), SelectApp.class);
                    intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                    startActivity(intent);

                }
            }
        });
    }
}
```

시연 영상

문제점 및 향후 개선 사항

1. 백그라운드에서 반드시 실행이 되고 있어야 함.
2. 홈 위젯에 확인 버튼을 없앨 것.
3. 전화 걸기, 알람 설정 등 각종 매크로도 매핑 가능하도록
4. 디자인 개선

감사합니다