

기말-201421256-박세환

**목차**

개발배경 및 기대효과

개발중점

개발환경 및 사용도구

동작방법

프로그램 설명

1. 특징
2. 프로그램 구조
3. 소스코드

추후 개발

참조

**개발배경 및 기대효과**

기존의 메시지 앱은 발신자를 기준으로 정렬됩니다.

여러 발신자에서 넘어온 정보 중에 원하는 정보를 깔끔하게 정리하여서 확인할 필요성을 느꼈습니다. (예, 여러 은행에서 자동 이체되는 정보, 여러 기관에서 수집되는 정보 등)

이 어플리케이션을 통해서 분산되는 정보를 한 곳에서 확인하고 불필요한 정보를 필터링 할 수 있으며 메시지의 내용을 분석하는 기능을 추가하면 일정관리를 비롯한 개인 생활분야에 뛰어난 역할을 할 수 있게 됩니다.

**개발중점**

사용자의 편의를 최우선으로 한다.

**개발환경 및 사용도구**

Window 10 64bit

jdk1.8.0\_172

jre1.8.0\_172

Android Studio 3.1.2

AVD: Nexus 5 API 21 Android 5.0

DB browser for SQLite 3.10.1

**동작방법**

SMSParse는 원하는 단어를 입력하면 이후 해당 단어가 포함된 SMS를 수신할 때 그 SMS를 따로 저장하는 동작을 하는 어플리케이션입니다.

|  |  |
| --- | --- |
|  | 프로그램의 실행 화면입니다. 프로그램은 자동으로 SMS의 수신 권한 여부를 파악하고 권한이 필요할 경우 권한을 위한 요청이 진행됩니다. 이후 동작을 원할 때 좌측상단의 Draw Menu로 접근할 수 있습니다. |
|  | Draw Menu는 Database 메뉴와 SMSView 메뉴로 이루어져 있습니다.  Database 메뉴는 SMS를 분석할 때 사용할 단어를 추가하거나 삭제하는 기능을 담당하고 있습니다.  SMSView에서는 수집된 단어를 확인할 수 있습니다. |

|  |  |
| --- | --- |
|  | SMS분석에 사용할 단어를 입력하고 테이블생성버튼을 누르면 테이블이 생성되고 화면이 새로 고침 됩니다.  삭제하고자 하는 테이블은 아래의 Spinner에서 선택한 후 테이블삭제버튼을 누르면 테이블이 삭제되고 화면이 새로 고침 됩니다. |
|  | 원하는 단어가 포함된 SMS를 볼 수 있는 화면입니다. 메시지별로 배경색을 달리해서 구분하였으며 발신자와 내용, 수신시각을 확인할 수 있습니다.  화면 상단의 Spinner를 조작하면 다른 단어로부터 수집된 SMS를 확인할 수 있습니다.  또한, ScrollView를 사용하였기 때문에 아래로 스크롤하면 더 많은 내용을 확인할 수 있습니다. |

**프로그램 설명**

1. 특징

SMSParse는 Navigation Drawer Activity를 사용해서 익숙하고 직관적인 UI를 채택했습니다.

SMSParse는 많은 수의 Activity를 피하고 하나의 MainActivity에서 동작하며 Fragment를 교체하는 방식으로 화면 전환합니다. 이를 통해 효율적인 메모리 활용이 가능합니다.

SMSParse는 BradcastReceiver내부에서 문자메세지 분석 및 정렬이 이루어지기 때문에 사용자의 요구동작이 최소화됩니다.

SMSParse는 View를 동적으로 생성하여서 보여줄 View의 내용에 제한을 두지 않으며 ScrollView를 통해서 편안하게 확인할 수 있습니다.

1. 프로그램 구조

Activity: MainActivity  
onCreate()에서 IntroFragment를 띄우고 SMS수신 권한을 확인합니다.  
onNavigationItemSelected()에서 Fragment를 전환하는 코드를 정의했으며 FragmentManager 등이 이용되었습니다.

Fragment: Database, SMSView, IntroFragment, EmptyFragment

Database – 읽어들일 대상이 되는 단어를 추가하거나 삭제할 수 있습니다.  
onCreateView()에서 EditText와 Spinner, Button을 정의하였고 클래스 내부에서 createTable(), deleteTable(), createQuery(), deleteQuery()가 정의되어 있으며 FragmentManger, SQLiteDatabase.execSQL(), SharePreference 등이 이용되었습니다.

SMSView –수집된 parsesms.db를 확인할 수 있습니다.  
SharedPreference에 저장된 값을 Spinner에 맵핑하였고 해당 테이블의 내용을 동적으로 출력하는 viewTable()을 정의했으며 Cursor, ArrayAdapter ViewGroup.addView()등이 이용되었습니다.

IntroFragment – 프로그램에 대한 기본적인 정보가 담겨있으며 어플리케이션 실행 시 확인 가능합니다.

EmptyFragment – 아무런 역할도 하지 않는 빈 Fragment입니다.

Receiver: SMSReceiver  
SMS를 읽어들이고 받아올 단어와 SMS의 내용을 비교하여 그 단어가 포함되있는 경우 parse.db에 저장합니다. ContentValue와 SQLiteDatabase.insert() 등이 이용되었습니다.

SQLiteDatabase: parse.db  
읽어들일 단어를 테이블 이름으로 가지고 sender, contents, receivedDate의 열을 가지고 있는 데이터베이스 파일입니다. 프로그램 내부에서 SQL문으로 접근합니다.

SharedPreference: query.xml  
SMS에서 읽어들일 단어와 그 개수를 가지고 있는 메모리입니다.

SMSParse

MainActivity

Database

SMSView

IntroFragment

EmptyFragment

SMSReceiver

parse.db

query

1. 소스코드

|  |
| --- |
| Database.java |
| package kr.ac.sangmyung.compeng.smsparse;  import android.app.FragmentTransaction; import android.content.Context; import android.content.SharedPreferences; import android.database.sqlite.SQLiteDatabase; import android.os.Bundle; import android.app.Fragment; import android.view.LayoutInflater; import android.view.View; import android.view.ViewGroup; import android.widget.AdapterView; import android.widget.ArrayAdapter; import android.widget.Button; import android.widget.EditText; import android.widget.Spinner;  import static android.content.Context.*MODE\_PRIVATE*;  public class Database extends Fragment {   SharedPreferences query;  /\*  query.xml  key value 쌍  "index" - 0부터 1씩 오르는 int값이며 다음으로 put 할 value의 index를 가지고 있다.  index - table의 이름과 SMS에서 분석하게 될 String값이며 하나씩 put 될 때 마다 String.valueof(getInt("index",0))의 값을 id로 저장하며 index가 하나씩 오르게 설계되었다.  \*/  SQLiteDatabase db;  /\*  smsparse.db  SMS에서 분석하게 될 단어를 테이블 이름으로 가지며 그 단어가 포함된  SMS의 발신자, 내용, 수신시각을 sender, contents, receivedDate 라는 엔트리로 접근한다.  \*/  EditText insertTableName;  Spinner selectTableName;  Button createTableBtn,deleteTableBtn;  String createTableName;  String deleteTableName;  String[] items;   public Database() {  // Required empty public constructor  }   @Override  public void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  }   @Override  public View onCreateView(LayoutInflater inflater, ViewGroup container,  Bundle savedInstanceState) {  final ViewGroup rootView = (ViewGroup) inflater.inflate(R.layout.*fragment\_database*, container, false);   //테이블과 query에 추가하고 싶은 단어를 추가하는 작업이다.  //EditText로 단어를 받아들이며 단어를 생성한 후에는 fragment를 refresh한다.  insertTableName = (EditText) rootView.findViewById(R.id.*inputTableName*);  createTableBtn = (Button) rootView.findViewById(R.id.*createTableBtn*);  createTableBtn.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  createTableName = insertTableName.getText().toString();  createTable(createTableName);  createQuery(createTableName);  insertTableName.setText("");  refresh();  }  });   //SMS에서 분석하게 될 단어를 삭제하는 작업이다.  //현재 분석하고 있는 단어들을 spinner를 활용해 보여주고 그 중 선택된 단어가 deleteTableBtn에서 onClick될 때 작업이 이루어진다.  query = getActivity().getSharedPreferences("query",*MODE\_PRIVATE*);  int index = query.getInt("index",0);  //index만큼의 배열을 가진 String[] items에 index와 value값을 맵핑한다.  items = new String[index];  for (int i=0;i<index;i++) {  items[i]=query.getString(String.*valueOf*(i),"");  }  //spinner에 items의 값을 입력하고 선택된 단어는 deleteTableName에 저장된다.  selectTableName = (Spinner) rootView.findViewById(R.id.*selectTableName*);  ArrayAdapter<String> adapter = new ArrayAdapter<String>(getActivity(),android.R.layout.*simple\_spinner\_item*,items);  selectTableName.setAdapter(adapter);  selectTableName.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {  @Override  public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {  deleteTableName = items[position];  }  @Override  public void onNothingSelected(AdapterView<?> parent) {   }  });  //앞서 spinner에서 설정된 deleteTableName을 이용해서 db와 query를 수정하는 작업이다.  //단어를 삭제한 이후에는 fragment를 refresh한다.  deleteTableBtn = (Button) rootView.findViewById(R.id.*deleteTableBtn*);  deleteTableBtn.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  deleteTable(deleteTableName);  deleteQuery(deleteTableName);  refresh();  }  });   return rootView;  }   @Override  public void onAttach(Context context) {  super.onAttach(context);  }   @Override  public void onDetach() {  super.onDetach();  }  //execSQL을 이용해서 db에 SQL질의문을 입력한다. 같은 이름의 테이블이 존재하지 않을 때 name이란 이름으로 테이블을 만들고 자동으로 증가하는 \_id값, 텍스트를 저장하는 sender, contents, receivedDate 열을 가진다.  public void createTable(String name) {  if(!name.isEmpty()) {//name 이 빈 문자열이 아닌 경우에만  dbOpen();  db.execSQL("create table if not exists " + name + "("  + " \_id integer PRIMARY KEY autoincrement, "  + " sender text, "  + " contents text, "  + " receivedDate text);");  }  }  public void deleteTable(String name) {  if(!name.isEmpty()) {//name 이 빈 문자열이 아닌 경우에만  dbOpen();  db.execSQL("DROP TABLE " + name);  }  }  void dbOpen() {  db = getActivity().openOrCreateDatabase("smsparse.db", *MODE\_PRIVATE* ,null); //SMSParse 이름의 db생성  }  void createQuery(String name) {  if(!name.isEmpty()) {//name 이 빈 문자열이 아닌 경우에만  query = getActivity().getSharedPreferences("query",*MODE\_PRIVATE*); // query.xml   SharedPreferences.Editor editor = query.edit();  //현재 index값을 가져오고 그 index에 추가하고자 하는 단어를 저장한 후 index의 값을 하나 올린다.  //현재 index를 key로 한 값에 name을 저장하고 index값을 1올린다.  int i = query.getInt("index",0);  editor.putString(String.*valueOf*(i),name);  i++;  editor.putInt("index",i);  editor.commit();  }  }  void deleteQuery(String name) {  SharedPreferences.Editor editor = query.edit();   //삭제하고자 하는 query의 value값을 받아오기 때문에 value값에 접근하기 위해서 탐색한다.  int i = query.getInt("index",0);  for(int a=0;a<i;a++) {  String target = query.getString(String.*valueOf*(a),"");  if (target.equals(name)) {//삭제하고자하는 name값을 발견하면  for(int b=a;b<i-1;b++) {//해당하는 index에 다음 index의 value값을 입력하는 것을 반복  editor.putString(String.*valueOf*(b),query.getString(String.*valueOf*(b+1),""));  }  i--;//맨 마지막 index를 삭제한다.  editor.putInt("index",i);  editor.commit();  }  }  }  void refresh(){  //refresh()가 작동하는 Fragment를 새로고침 한다. detach()와 attach()는 Fragment에서 발생하는 이벤트이다.  FragmentTransaction refresh = getFragmentManager().beginTransaction();  refresh.detach(this).attach(this).commit();  } } |

|  |
| --- |
| SMSView.java |
| package kr.ac.sangmyung.compeng.smsparse;  import android.app.Activity; import android.content.Context; import android.content.SharedPreferences; import android.database.Cursor; import android.database.sqlite.SQLiteDatabase; import android.graphics.Color; import android.os.Bundle; import android.app.Fragment; import android.view.LayoutInflater; import android.view.View; import android.view.ViewGroup; import android.widget.AdapterView; import android.widget.ArrayAdapter; import android.widget.LinearLayout; import android.widget.Spinner; import android.widget.TextView;  import java.util.Random;  import static android.content.Context.*MODE\_PRIVATE*;  public class SMSView extends Fragment {   SharedPreferences query;  SQLiteDatabase db;  Spinner spinner;  String[] items;   public SMSView() {  // Required empty public constructor  }   @Override  public void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  }   @Override  public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {  ViewGroup rootView = (ViewGroup) inflater.inflate(R.layout.*fragment\_smsview*, container, false);   //단어를 통해 수집된 SMS를 보여주는 작업이다.  //현재 분석하고 있는 단어들을 spinner를 활용해 보여주고 onItemSelected되면 해당 테이블의 내용의 개수만큼 동적생성해서 화면에 출력한다.  query = getActivity().getSharedPreferences("query",*MODE\_PRIVATE*);  int index = query.getInt("index",0);  items = new String[index];  //index만큼의 배열을 가진 String[] items에 index와 value값을 맵핑한다.  for (int i=0;i<index;i++) {  items[i]=query.getString(String.*valueOf*(i),"");  }  //spinner에 items의 값을 입력하고 선택된 단어는 viewTable(String)의 파라미터로 넘겨진다.  spinner = (Spinner) rootView.findViewById(R.id.*spinner*);  ArrayAdapter<String> adapter = new ArrayAdapter<String>(getActivity(),android.R.layout.*simple\_spinner\_item*,items);  spinner.setAdapter(adapter);  spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {  @Override  public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {  viewTable(items[position]);  }   @Override  public void onNothingSelected(AdapterView<?> parent) {   }  });   return rootView;  }   @Override  public void onAttach(Context context) {  super.onAttach(context);  }   @Override  public void onDetach() {  super.onDetach();  }   void viewTable(String query){  db = getActivity().openOrCreateDatabase("smsparse.db", Activity.*MODE\_PRIVATE* ,null); //smsparse.db  //rawQuery()를 이용해서 query 테이블의 sender, contents, receivedDate값을 읽어들이며 그 값은 Cursor객체에 저장된다.  Cursor c1 = db.rawQuery("select sender, contents, receivedDate " + "from " + query,null);   LinearLayout dynamicLayout = (LinearLayout) getActivity().findViewById(R.id.*dynamicLayout*);  dynamicLayout.setOrientation(LinearLayout.*VERTICAL*);  dynamicLayout.removeAllViews(); //뷰를 동적생성할 레이아웃의 초기화작업   /\*  c1의 첫번째 값은 레코드의 갯수를 int값으로 저장하고  다음 값부터 레코드의 값을 읽어들인다. columnIndex를 이용해 접근하며 0부터 1씩 오른다.  다음값으로 넘어가고자 할 땐 Cursor.moveToNext()를 이용한다.  \*/  int recordCount = c1.getCount();  for (int i=0;i<recordCount;i++) {//레코드의 갯수만큼 반복   int R = new Random().nextInt(255);  int G = new Random().nextInt(255);  int B = new Random().nextInt(255);   c1.moveToNext();   String sender = c1.getString(0);  String contents = c1.getString(1);  String receivedDate = c1.getString(2);   TextView senderView = new TextView(getActivity());  senderView.setText(sender);  senderView.setBackgroundColor(Color.*argb*(60, R, G, B));//투명도가 설정된 랜덤으로 생성된 색상을 배경으로 설정한다.   TextView contentsView = new TextView(getActivity());  contentsView.setText(contents);  contentsView.setTextSize(20);  contentsView.setBackgroundColor(Color.*argb*(60, R, G, B));   TextView receivedDateView = new TextView(getActivity());  receivedDateView.setText(receivedDate);  receivedDateView.setBackgroundColor(Color.*argb*(60, R, G, B));   dynamicLayout.addView(senderView);  dynamicLayout.addView(contentsView);  dynamicLayout.addView(receivedDateView);  }  } } |

|  |
| --- |
| MainActivity.java |
| package kr.ac.sangmyung.compeng.smsparse;  import android.Manifest; import android.content.pm.PackageManager; import android.os.Bundle; import android.support.annotation.NonNull; import android.support.design.widget.FloatingActionButton; import android.support.design.widget.Snackbar; import android.support.v4.app.ActivityCompat; import android.support.v4.content.ContextCompat; import android.view.View; import android.support.design.widget.NavigationView; import android.support.v4.view.GravityCompat; import android.support.v4.widget.DrawerLayout; import android.support.v7.app.ActionBarDrawerToggle; import android.support.v7.app.AppCompatActivity; import android.support.v7.widget.Toolbar; import android.view.Menu; import android.view.MenuItem; import android.widget.Toast;  public class MainActivity extends AppCompatActivity implements NavigationView.OnNavigationItemSelectedListener {   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);  Toolbar toolbar = (Toolbar) findViewById(R.id.*toolbar*);  setSupportActionBar(toolbar);   DrawerLayout drawer = (DrawerLayout) findViewById(R.id.*drawer\_layout*);  ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(  this, drawer, toolbar, R.string.*navigation\_drawer\_open*, R.string.*navigation\_drawer\_close*);  drawer.addDrawerListener(toggle);  toggle.syncState();   NavigationView navigationView = (NavigationView) findViewById(R.id.*nav\_view*);  navigationView.setNavigationItemSelectedListener(this);   //프로젝트를 소개하는 프래그먼트를 띄웁니다.  getFragmentManager().beginTransaction().replace(R.id.*container*, new IntroFragment()).commit();   //SMS수신 권한을 체크합니다.  int permissionCheck = ContextCompat.*checkSelfPermission*(this, Manifest.permission.*RECEIVE\_SMS*);  if (permissionCheck == PackageManager.*PERMISSION\_GRANTED*) {  Toast.*makeText*(this, "SMS 수신 권한 있음", Toast.*LENGTH\_LONG*).show();  } else {  Toast.*makeText*(this, "SMS 수신 권한 없음", Toast.*LENGTH\_LONG*).show();  if (ActivityCompat.*shouldShowRequestPermissionRationale*(this, Manifest.permission.*RECEIVE\_SMS*)) {  Toast.*makeText*(this, "SMS 권한 설명 필요함.", Toast.*LENGTH\_LONG*).show();  } else {  ActivityCompat.*requestPermissions*(this, new String[]{Manifest.permission.*RECEIVE\_SMS*}, 1);  }  }  }   @Override  public void onBackPressed() {  DrawerLayout drawer = (DrawerLayout) findViewById(R.id.*drawer\_layout*);  if (drawer.isDrawerOpen(GravityCompat.*START*)) {  drawer.closeDrawer(GravityCompat.*START*);  } else {  super.onBackPressed();  }  }   @SuppressWarnings("StatementWithEmptyBody")  @Override //draw menu의 이벤트를 정의합니다.  public boolean onNavigationItemSelected(MenuItem item) {  // Handle navigation view item clicks here.  int id = item.getItemId();   //프래그먼트를 선택합니다. item의 id는 menu/activity\_main\_drawer.xml에서 설정합니다.  //프래그먼트매니저를 호출해서 새로운 프래그먼트의 생성자를 통해 접근합니다. container는 content\_main.xml의 fregment id 입니다.  if (id == R.id.*nav\_database*) {  getFragmentManager().beginTransaction().replace(R.id.*container*, new Database()).commit();  } else if (id == R.id.*nav\_smsview*) {  getFragmentManager().beginTransaction().replace(R.id.*container*, new SMSView()).commit();  }   DrawerLayout drawer = (DrawerLayout) findViewById(R.id.*drawer\_layout*);  drawer.closeDrawer(GravityCompat.*START*);  return true;  }    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {  switch (requestCode) {  case 1: {  if (grantResults.length > 0 && grantResults[0] == PackageManager.*PERMISSION\_GRANTED*) {  Toast.*makeText*(this, "SMS 권한을 사용자가 승인함", Toast.*LENGTH\_LONG*).show();  } else {  Toast.*makeText*(this, "SMS 권한 거부됨", Toast.*LENGTH\_LONG*).show();  }  return;  }  }  } |

|  |
| --- |
| SMSReceiver.java |
| package kr.ac.sangmyung.compeng.smsparse;  import android.content.BroadcastReceiver; import android.content.ContentValues; import android.content.Context; import android.content.Intent; import android.content.SharedPreferences; import android.database.sqlite.SQLiteDatabase; import android.os.Build; import android.os.Bundle; import android.telephony.SmsMessage;  import java.text.SimpleDateFormat; import java.util.Date;  import static android.content.Context.*MODE\_PRIVATE*;  public class SMSReceiver extends BroadcastReceiver {  public SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");  SQLiteDatabase db;  SharedPreferences query;   @Override  public void onReceive(Context context, Intent intent) {  // 인텐트 안에 들어있는 SMS 메시지를 파싱합니다.  Bundle bundle = intent.getExtras();  SmsMessage[] messages = parseSmsMessage(bundle);   if (messages != null && messages.length > 0) {  // SMS 발신 번호 확인  String sender = messages[0].getOriginatingAddress();  // SMS 메시지 확인  String contents = messages[0].getMessageBody().toString();  // SMS 수신 시간 확인  Date receivedDate = new Date(messages[0].getTimestampMillis());   insertSMS(context, sender, contents, receivedDate);  }  }   private SmsMessage[] parseSmsMessage(Bundle bundle) {  Object[] objs = (Object[]) bundle.get("pdus");//"pdus"라는 key값으로 저장된 SMS를 object객체로 읽는다.  SmsMessage[] messages = new SmsMessage[objs.length];   int smsCount = objs.length;  for (int i = 0; i < smsCount; i++) {  // PDU 포맷으로 되어 있는 메시지를 복원합니다.  if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*M*) { // API 23 이상  String format = bundle.getString("format");  messages[i] = SmsMessage.*createFromPdu*((byte[]) objs[i], format);  } else {  messages[i] = SmsMessage.*createFromPdu*((byte[]) objs[i]);  }  }  return messages;  }    void insertSMS(Context context, String sender, String contents, Date receivedDate) {  db = context.openOrCreateDatabase("smsparse.db", *MODE\_PRIVATE* ,null); //SMSParse 이름의 db생성  query = context.getSharedPreferences("query",*MODE\_PRIVATE*); // query.xml qeury문 모음   int i;  int index = query.getInt("index",0);   for(i=0;i<index;i++) {  String a = query.getString(String.*valueOf*(i), "");  if (contents.contains(a)) {  ContentValues recordValues = new ContentValues();  recordValues.put("sender", sender);  recordValues.put("contents", contents);  recordValues.put("receivedDate", format.format(receivedDate));  //recordValues에 sender,contents,receivedDate 키값으로 저장된 SMS의 내용을 db에 insert()  db.insert(a, null, recordValues);  }  }  } } |

**추후 개발**

MMS 분석기능 추가

Message 내부의 날짜 분석해서 Google Calendar API의 이벤트추가기능 결합(Naver Calendar)

NotificationListenerService를 이용해서 다른 어플의 push 불러오기 (예: 은행관리어플의 출금알림)

**참조**

Do it 안드로이드 앱 프로그래밍 5판 / 이지스퍼블리싱

Developer.android.com

<https://developer.android.com/reference/java/lang/String>

Youtube

https://youtu.be/7L17qFN2I6k

<http://nexthops.tistory.com/29>

<https://rain483.blog.me/220812579755>