Assignment FOUR — Content providers

Xuefan Wu

PART2: Chat App

entity classes (entities package) for Messages and Peers

```
public class Message implements Parcelable {
    public long id;
    public String messageText;
    public String sender;

    public Message(String messageText,String sender){
        this.messageText = messageText;
        this.sender = sender;
    }

    protected Message(Parcel in) {
        id = in.readLong();
        messageText = in.readString();
        sender = in.readString();
    }

    public Message(Cursor cursor){
        this.id = MessageContract.getId(cursor);
        this.messageText = MessageContract.getMessage(cursor);
```

```
public class Peer implements Parcelable {
    public long id;
    public String name;
    public InetAddress address;
    public int port;

    public Peer(String name,InetAddress address,int port){
        this.name = name;
        this.address = address;
        this.port = port;
    }

    protected Peer(Parcel in) {
        id = in.readLong();
        name = in.readString();
        port = in.readInt();
        byte[] ipAddress = new byte[in.readInt()];
```

contract class (contracts package)

```
ublic class MessageContract {

    public MessageContract() {
    }

    public static final String ID = "_id";
    public static final String MESSAGE = "message";
    public static final String TXT = "txt";
    public static final String PEER_FK = "peer_fk";
    public static final String AUTHORITY = "edu.stevens.cs522.cha
    public static final String PATH ="MessageTable";
    public static final Uri CONTENT_URI = CONTENT_URI(AUTHORITY,P
    public static final String CONTENT_PATH = CONTENT_PATH(CONTEN
    public static final String CONTENT_PATH_ITEM = CONTENT_PATH(C


    public static Uri CONTENT_URI(String authority,String path){
        return new Uri.Builder().scheme("content")
                .authority(authority)
                .path(path)
                .build();
    }


    public static Uri withExtendedPath(Uri uri, String... path){
```

```
ublic class PeerContract {

    public static final String ID = "_id";
    public static final String NAME = "name";
    public static final String ADDRESS = "address";
    public static final String PORT = "port";
    public static final String AUTHORITY = "edu.stevens.cs522.chat.oneway.server";
    public static final String PATH ="PeerTable";
    public static final Uri CONTENT_URI = CONTENT_URI(AUTHORITY,PATH);
    public static final String CONTENT_PATH = CONTENT_PATH(CONTENT_URI);
    public static final String CONTENT_PATH_ITEM = CONTENT_PATH(CONTENT_URI("#"));


    public static Uri CONTENT_URI(String authority,String path){
        return new Uri.Builder().scheme("content")
                .authority(authority)
                .path(path)
                .build();
    }
```

content provider (providers package)

```
static final UriMatcher uriMatcher;
static{
    uriMatcher = new UriMatcher(UriMatcher.NO_MATCH);
    uriMatcher.addURI(PeerContract.AUTHORITY, PeerContract.CONTENT_PATH, PEER);
    uriMatcher.addURI(PeerContract.AUTHORITY, PeerContract.CONTENT_PATH_ITEM, P
    uriMatcher.addURI(MessageContract.AUTHORITY, MessageContract.CONTENT_PATH,
    uriMatcher.addURI(MessageContract.AUTHORITY, MessageContract.CONTENT_PATH_I
}
@Override
public boolean onCreate() {
    Context context = getContext();
    dbHelper = new DatabaseHelper(context);
    db = dbHelper.getWritableDatabase();
    pDb = new PeerDbAdapter(context);
    return (db == null)? false:true;
}
```

manager class (managers package)

PeerManager.java

```
public class PeerManager extends Manager<Peer>{


    public PeerManager(Context context, IEntityCreator creator, int loadID) {
        super(context, creator, loadID);
    }

    public boolean checkPeer(final Peer peer) { return PeerProvider.checkPeerExist(

    public void persistAsync(final Peer peer){
        if(checkPeer(peer)) {
            ContentValues contentValues = new ContentValues();
            peer.writeToProvider(contentValues, peer);
            getAsyncContentResolver().insertAsync(PeerContract.CONTENT_URI,
                    contentValues, null);
        }

    }
```

MessageManager.java

```
*/
public class MessageManager extends Manager<Message>{


    public MessageManager(Context context, IEntityCreator creator, int loadID)
        super(context, creator, loadID);
    }

    public void persistAsync(final Peer peer,final Message message){
        int flag=0;
        while (flag ==0){
            flag = (int) PeerProvider.getPeer_fk(peer);

    }
    ContentValues contentValues = new ContentValues();
    MessageContract.putMessage(contentValues, message.messageText);
    MessageContract.putPeer_fk(contentValues, flag);
    getAsyncContentResolver().insertAsync(MessageContract.CONTENT_URI,
                                          contentValues, null);}
//    public void removeAsync(long ID) {
//        getAsyncContentResolver().deleteAsync(BookContract.CONTENT_URI(String
```