12-Type-Converter

Ajouter un nouveau string dans `strings.xml` :

```
<string name="last_updated">Last Updated</string>
```

Dans `list_item`, ajout d'un TextView :

```
<TextView
    android:id="@+id/tvLastUpdated"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="7dp"
    android:text="@string/last_updated"
    android:textColor="#616161"
    android:textSize="18sp" />
/>
```

On exécute et on voit un nouveau champ

Créer une classe `DateTypeConverter` :

```
class DateTypeConverter {

    @TypeConverter
    fun toDate(value: Long?) : Date? {
        return if (value == null) null else Date(value)
    }

    @TypeConverter
    fun toLong(value: Date?): Long? {
        return value?.time
    }
}
```

Ajout d'un nouveau champ dans l'entité `Book` :

```
@ColumnInfo(name = "last_updated")
val lastUpdated: Date
```

Ajout d'une nouvelle migration (version 3) :

```
@Database(entities = [Book::class], version = 3, exportSchema = false)
@TypeConverters(DateTypeConverter::class)

...

val MIGRATION_2_3: Migration = object : Migration(2, 3) {
    override fun migrate(database: SupportSQLiteDatabase) {
        database.execSQL(
            "ALTER TABLE book " +
                    "ADD COLUMN last_updated INTEGER DEFAULT NULL"
        )
    }
}
```

Ajout de notre nouvelle migration :

```
fun getDatabase(context: Context): BookRoomDatabase? {
    if (bookRoomInstance == null) {
        synchronized(BookRoomDatabase::class.java) {
            if (bookRoomInstance == null) run {
                bookRoomInstance = Room.databaseBuilder<BookRoomDatabase>(
                    context.applicationContext,
                    BookRoomDatabase::class.java,
                    "book_room_db.sqlite"
                )
                    .addMigrations(MIGRATION_1_2, MIGRATION_2_3)  // AJOUT
ICI
                    .build()
            }
        }    }
    return bookRoomInstance!!
}
```

Modification dans le MainActivity :

```
private fun updateBook(data: Intent?) {
    val id : String ? = data?.getStringExtra(EditBookActivity.ID)
    val author: String? =
data?.getStringExtra(EditBookActivity.UPDATED_AUTHOR)
    val book : String ? =
data?.getStringExtra(EditBookActivity.UPDATED_BOOK)
    val description: String? =
data?.getStringExtra(EditBookActivity.UPDATED_DESCRIPTION)
    val currentTime: Date? = Calendar.getInstance().time
    val updated_book = Book(id!!, author!!, book!!, description!!,
currentTime!!)
currentTime!!)
    bookViewModel.update(updated_book)
}

private fun insertNewBook(data: Intent?) {
    val id = UUID.randomUUID().toString()
    val authorName: String? =
data?.getStringExtra(NewBookActivity.NEW_AUTHOR)
    val bookName : String? = data?.getStringExtra(NewBookActivity.NEW_BOOK)
    val description : String? =
data?.getStringExtra(NewBookActivity.NEW_DESCRIPTION)
    val currentTime: Date? = Calendar.getInstance().time
    val book = Book(id, authorName!!, bookName!!, description!!,
currentTime!!)

    bookViewModel.insert(book)

    Toast.makeText(applicationContext, R.string.saved,
Toast.LENGTH_LONG).show()
}
```

Dans `SearchResultActivity`, on ajoute :

```
private fun updateBook(data: Intent?) {
    val id : String ? = data?.getStringExtra(EditBookActivity.ID)
    val author: String? =
data?.getStringExtra(EditBookActivity.UPDATED_AUTHOR)
    val book : String ? =
data?.getStringExtra(EditBookActivity.UPDATED_BOOK)
    val description: String? =
data?.getStringExtra(EditBookActivity.UPDATED_DESCRIPTION)
    val currentTime: Date? = Calendar.getInstance().time
    val updated_book = Book(id!!, author!!, book!!, description!!,
currentTime!!)
    searchViewModel.update(updated_book)
}
```

Préciser dans le NewBookActivity qu'on ne veut pas montrer le champ lastupdated :

```
txvLastUpdated.visibility = View.INVISIBLE
```

Modification dans le BookListAdapter :

```
override fun onBindViewHolder(holder: BookViewHolder, position: Int) {
    val book = bookList[position]
    holder.setData(book.author, book.book, book.lastUpdated,position)
    holder.setListeners()
}
```

Modifier le setData et ajout d'une nouvelle fonction :

```
fun setData(author: String, book : String, lastUpdated: Date, position: Int)
{
    itemView.tvAuthor.text = author
    itemView.tvBook.text = book
    itemView.tvLastUpdated.text =  getFormattedDate(lastUpdated)
    this.pos = position
}

private fun getFormattedDate(lastUpdated: Date?): String {
    var time = "Last updated: "
    time += lastUpdated?.let {
        val sdf = SimpleDateFormat("HH:mm d MM, yyyy", Locale.getDefault())
        sdf.format(lastUpdated)
    } ?: "Not found"
    return time
}
```

Puis, ajout dans le setListeners :

```
intent.putExtra("lastUpdated", bookList[pos].lastUpdated)
```

Dans `EditBookActivity` , modification de onCreate :

```
bundle?.let {
    id = bundle.getString("id")
    val author: String? = bundle.getString("author")
    val book: String? = bundle.getString("book")
    val description: String? = bundle.getString("description")
    val lastUpdated: String? = bundle.getString("lastUpdated")

    etAuthorName.setText(author)
    etBookName.setText(book)
    etDescription.setText(description)
    txvLastUpdated.setText(lastUpdated)
}
```

ET VOILA !!!!