09-Performing-Filtering-Operation-on-Room

Widget de recherche : `SearchView`

Le widget de recherche est une instance de SearchView.

Par défaut, ce widget se comporte comme un EditText

Processus d'implémentation d'une interface de recherche

1. Quand l'utilisateur exécute une recherche à partir de la boite de dialogue, le système crée un `Intent` et y stocke la requête de l'utilisateur

2. Le système démarre alors l'activité déclarée pour gérer les recherches et lui transmets le `Intent`

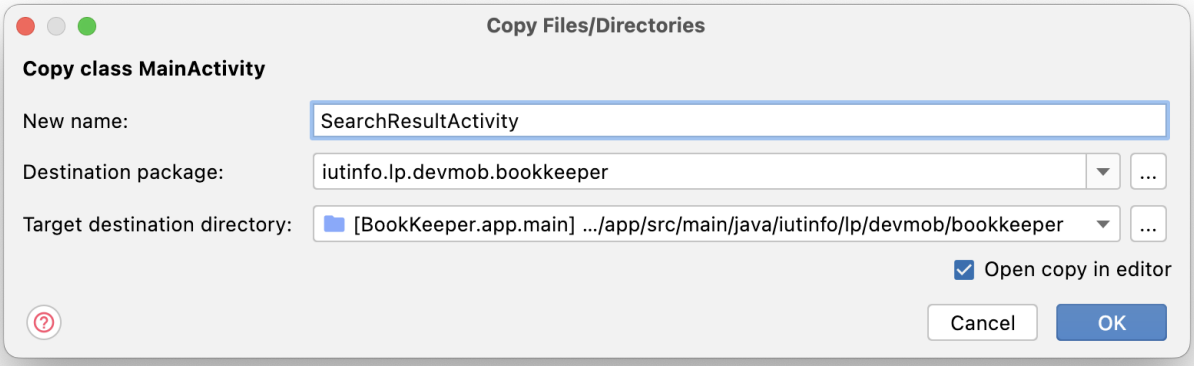
3. Pour config l'appli nous avons besoin de 3 éléments :

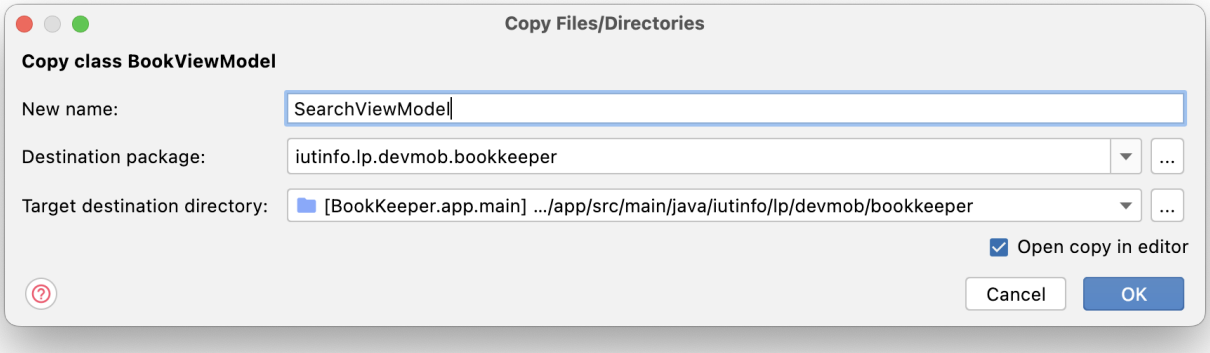
    - Configuration de la recherche (un fichier XML)
    - Une activité dédiée à la gestion de la recherche (ex: `Searchable Activity`)
    - Une interface de recherche (Boite de dialogue OU `SearchView`)

Le fichier de configuration doit inclure la balise `<searchable>`, exemple :

```
<?xml version="1.0" encoding="utf-8" ?>
<searchable
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:label="@string/app_name"
    android:hint="@string/search_int">
</searchable>
```

Création de l'activité





Modification de `SearchResultActivity` :

```
class SearchResultActivity : AppCompatActivity(),
BookListAdapter.OnDeleteClickListener {

    override fun OnDeleteClickListener(book: Book) {
        searchViewModel.delete(book)
        Toast.makeText(applicationContext, R.string.deleted,
Toast.LENGTH_LONG).show()
    }

    private lateinit var searchViewModel: SearchViewModel
    private var bookListAdapter: BookListAdapter? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        setSupportActionBar(toolbar)

        fab.visibility = View.INVISIBLE

        searchViewModel =
```

```
ViewModelProviders.of(this).get(SearchViewModel::class.java)

        bookListAdapter = BookListAdapter(this, this)
        recyclerview.adapter = bookListAdapter
        // pas obligatoire
        recyclerview.layoutManager = LinearLayoutManager(this)

        val bookDB = BookRoomDatabase.getDatabase(this);

        searchViewModel.allBooks.observe(this) { books ->
            books?.let {
                bookListAdapter!!.setBooks(books)
            }
        }    }
}
```

Ajouter une nouvelle activité dans le manifest :

```
<activity
    android:name=".SearchResultActivity"
    android:theme="@style/Theme.AppCompat.NoActionBar"
    android:launchMode="singleTop"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.SEARCH"/>
    </intent-filter>
    <meta-data
        android:name="android.app.searchable"
        android:resource="@xml/searchable">
    </meta-data>
</activity>
```

Dans le menu_main.xml :

```
<item
        android:id="@+id/search"
        android:title="@string/search"
        android:icon="@drawable/ic_search"
        app:showAsAction="collapseActionView|ifRoom"
        app:actionViewClass="android.widget.SearchView"/>
```

Mettre en place la recherche

`MainActivity`:

```
```

```
override fun onCreateOptionsMenu(menu: Menu): Boolean {
    var inflater = menuInflater
    // Inflate the menu; this adds items to the action bar if it is present.
    inflater.inflate(R.menu.menu_main, menu)

    val searchManager = getSystemService(Context.SEARCH_SERVICE) as
SearchManager
    val searchView = menu?.findItem(R.id.search)?.actionView as SearchView

    val componentName = ComponentName(this,
SearchResultActivity::class.java)
    val searchableInfo = searchManager.getSearchableInfo(componentName)
    searchView.setSearchableInfo(searchableInfo)

    return true
}
```

Dans le `SearchResultActivity` :

```
private fun handleIntent(intent:Intent) {
    if (Intent.ACTION_SEARCH == intent.action) {
        val searchQuery = intent.getStringExtra(SearchManager.QUERY)
        val TAG = "tag"
        Log.i(TAG, "Search Query : $searchQuery")
    }
}
```

`BookDAO` :

```

```