МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

Звіт

з лабораторної роботи № 4 з дисципліни

Програмування під Android

Виконала: Перевірив:

ст. гр. ПЗПІ-23-5 ст. викладач кафедри ПІ

Євсєєва Олександра Сокорчук Ігор Петрович

Харків 2024

ЗАВДАННЯ

Розробити застосунок для ведення нотаток. Застосунок повинен дозволяти переглядати список нотаток, створювати нову нотатку, редагувати нотатку та видаляти нотатку. Кожна нотатка повинна мати назву, опис, важливість (три класи), час і дату призначення і картинку (вибирається з галереї). У списку нотатки відображаються з іконкою з галереї і часом створення (праворуч зверху більш дрібним шрифтом), а також графічним відображенням класу важливості (іконка з множини).

Потрібно реалізувати функції додавання, видалення, редагування (без збереження між перезапуск додатку) нотаток.

Видалення/редагування викликаються з контекстного меню при довгому натисненні, додавання - з основного меню в ActionBar. У ActionBar реалізувати пошук / фільтрація заміток за змістом тексту, а також фільтрацію за важливістю. Текстові елементи користувацького інтерфейсу повинні бути локалізовані українською та англійською мовами.

ХІД РОБОТИ

Cтворюємо новий проєкт та обираємо Empty Views Activity.

Починаємо із розділення на фрагменти: для списку нотаток, для перегляду нотатки, редагування нотатки. Вони зроблені подібні один до одного для полегшення досвіду користувачу.

В MainActivity прив’язуємо toolbar для виконання навігації між фрагментами:

1.navController.addOnDestinationChangedListener((navController, navDestination, 2.bundle) -> {  
3. switch (navDestination.getId()) {  
4. case R.id.*noteListFragment*:  
5. toolbar.setTitle(R.string.*fragmentList*);  
6. break;  
7. case R.id.*noteEditFragment*:  
8. toolbar.setTitle(R.string.*fragmentEdit*);  
9. break;  
10. case R.id.*galleryFragment*:  
11. toolbar.setTitle(R.string.*fragemntGallery*);  
12. break;  
13. case R.id.*noteDetailsFragment*:  
14. toolbar.setTitle(R.string.*fragmentDetails*);}

А також прив’язуємо viewModel для відстеження подій:

15.viewModel = new ViewModelProvider(this).get(MainViewModel.class);

Переходимо до створення деталей нотатки. Першим кроком – завантажуємо. Потім зробимо функцію для збереження. Стан та поля нотатки знаходиться у MutableLiveData для того, щоб відстежувати зміни та динамічною виводити користувачеві.

Список нотаток показується за RecyclerView та Adapter відповідно. onBindViewHolder використовує holder.binding для встановлення значень у список. Далі для дій редагування треба натиснути на елемент списку:

16.holder.binding.itemRoot.setOnLongClickListener(view -> {  
17. PopupMenu menu = new PopupMenu(holder.binding.itemRoot.getContext(),holder.binding.itemRoot);  
18. menu.getMenu().add(Menu.*NONE*, 0, 0, R.string.*delete*);  
19. menu.getMenu().add(Menu.*NONE*, 1,1, R.string.*edit*);  
20. menu.setOnMenuItemClickListener(v -> {  
21. switch (v.getItemId()) {  
22. case 0:  
23. callback.onDelete(list.get(position).getTitle());  
24. break;  
25. case 1:  
26. callback.onEdit(list.get(position).getTitle());  
27. break;

Реалізуємо BindingAdapter для використання даних як тип пріоритету, дат та картинок [ДОДАТОК А].

Створюємо фрагмент та Adapter для відображення штучної галереї.

28.public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {  
29. binding = DataBindingUtil.*inflate*(inflater, R.layout.*fragment\_gallery*, container, false);  
30. binding.setLifecycleOwner(getViewLifecycleOwner());  
31. return binding.getRoot();}

32.binding.gallery.setAdapter(new GalleryAdapter(list,image -> {  
33. Bundle result = new Bundle();  
34. result.putInt("image", image);  
35. getParentFragmentManager().setFragmentResult("image", result);  
36. NavHostFragment.*findNavController*(this).popBackStack();}

Розмітка для головного меню для додатку:

37.<menu xmlns:android="http://schemas.android.com/apk/res/android">  
38. <item  
39. android:id="@+id/add"  
40. android:title="@string/add"/>  
41. <item  
42. android:id="@+id/search"  
43. android:title="@string/search"/>  
44. <item  
45. android:id="@+id/filter"  
46. android:title="@string/filter"/>  
47.</menu>

Додаємо локалізації через values.strings. Для прикладу, різниця між однією й тією же змінною:

48.<string name="titleLabel">Назва</string>  
49.<string name="descLabel">Опис</string>

50.<string name="titleLabel">Title</string>  
51.<string name="descLabel">Description</string>

ВИСНОВКИ

Під час виконання лабораторної роботи я набула навичок створення дійсного повноцінного проєкту з Android. Я навчилася використовувати різні елементи інтерфейсу та архітектурі застосунку. Отримала досвід з структури MVVM.

Відео-звіт на YouTube: <https://youtu.be/SI6AMXzQwJs>

ВИКОРИСТАНІ ДЖЕРЕЛА

1. Документація JavaRush switch-case. URL: <https://javarush.com/en/groups/posts/en..switch-statement-in-java> (дата звернення: 14.11.2024)
2. Документація JavaRush replacing strings in java. URL: <https://javarush.com/en/groups/posts/en.2845.replace-string-in-java> (дата звернення: 14.11.2024)

ДОДАТОК А

Код програми

Увесь код програми можно знайти за посиланням у GitHub:

<https://github.com/NureIevsieievaOleksandra/android-pzpi-23-5-ievsieieva-oleksandra>.