Sprawozdanie Aplikacje Mobilne

Wojciech Chwaciński 151924

1.Opis klas wykorzystanych w projekcie:

a) AnimationActivity

Inicjalizacja: W metodzie onCreate są inicjalizowane dwa widoki (imageView i imageView2), które będą uczestniczyć w animacji.

Konfiguracja animacji: Dla każdego z widoków (imageView i imageView2) tworzone są animacje za pomocą ObjectAnimator, które zmieniają wartość TRANSLATION_X (przesunięcie po osi X) od wartości początkowej do końcowej, a następnie z powrotem do wartości początkowej dla imageView. Dla imageView2 animacja przesuwa widok w lewo o 500 jednostek. Czas trwania każdej animacji wynosi 2000 milisekund (2 sekundy).

AnimatorSet: Animacje są grupowane w AnimatorSet, co pozwala na ich jednoczesne uruchomienie. Dodatkowo, do AnimatorSet dodawany jest słuchacz (AnimatorListener), który reaguje na zakończenie animacji.

Słuchacz animacji: W słuchaczu (AnimatorListener), w metodzie onAnimationEnd, po zakończeniu animacji uruchamiana jest nowa aktywność (MainActivity) za pomocą Intent. Po przejściu do MainActivity, aktywność AnimationActivity jest kończona metodą finish(), co zapobiega powrotowi do niej przy użyciu przycisku cofnij.

Uruchomienie animacji: Na końcu metody onCreate, animacje są uruchamiane za pomocą wywołania animatorSet.start().

b) MainActivity

Inicjalizacja: W metodzie onCreate są inicjalizowane kluczowe elementy interfejsu użytkownika: DrawerLayout, NavigationView oraz ViewPager i TabLayout.

Nawigacja boczna: NavigationView jest skonfigurowany do obsługi wyboru elementów nawigacji. Metoda setNavigationItemSelectedListener ustawia MainActivity jako słuchacz wyborów nawigacji, co pozwala na reagowanie na wybór poszczególnych elementów menu nawigacji.

ViewPager i TabLayout: ViewPager jest używany do wyświetlania różnych fragmentów w zależności od wybranego zakładki w TabLayout. Adapter FragmentAdapter jest konfigurowany z trzema fragmentami (HomeFragment, TatryFragment, KarkonoszeFragment), które są przypisane do odpowiednich zakładek w TabLayout. TabLayout jest następnie skonfigurowany do współpracy z ViewPager, co umożliwia automatyczne przełączanie zakładek w zależności od aktualnie wyświetlanego fragmentu.

c) HomeFragment

Inflacja widoku: W metodzie onCreateView, fragment influje swój widok z pliku fragment_home.xml.

Dodawanie FragmentZegar: Jeśli savedInstanceState jest null, co oznacza, że fragment jest tworzony po raz pierwszy, wytwarzana jest nowa instancja FragmentZegar. Następnie, za pomocą childFragmentManager, rozpoczyna się transakcja fragmentu, która dodaje

FragmentZegar do kontenera o identyfikatorze R.id.fragment_container. Transakcja jest zatwierdzana za pomocą ft.commit(), co powoduje wyświetlenie FragmentZegar w widoku HomeFragment.

d)TatryFragment/KarkonoszeFragment

Konfiguracja RecyclerView: RecyclerView jest skonfigurowany do wyświetlania elementów w układzie siatki (GridLayoutManager) z dwoma kolumnami. To pozwala na wyświetlanie szlaków w atrakcyjny sposób, zgodnie z układem siatki.

Dane szlaków: Lista szlaków jest definiowana jako listOf z nazwami szlaków.

Adapter dla RecyclerView: Adapter jest tworzony z listą szlaków i implementacją interfejsu OnItemClickListener, który pozwala na obsługę kliknięć na poszczególne elementy listy. Adapter jest następnie przypisywany do RecyclerView, co umożliwia wyświetlanie szlaków.

Obsługa kliknięć: W metodzie onItemClick, po kliknięciu na szlak, tworzony jest nowy instancja SzczegolyFragment z nazwą wybranego szlaku jako argumentem. Następnie, za pomocą supportFragmentManager, rozpoczyna się transakcja fragmentu, która zastępuje aktualnie wyświetlany fragment (TatryFragment/KarkonoszeFragment), fragmentem SzczegolyFragment. Transakcja jest zatwierdzana za pomocą commit(), co powoduje wyświetlenie szczegółów wybranego szlaku.

e)TatryAdapter/KarkonoszeAdapter

Interfejs OnItemClickListener: Adapter definiuje własny interfejs OnItemClickListener, który pozwala na obsługę kliknięć na poszczególne elementy listy. Interfejs zawiera jedną metodę onItemClick, która jest wywoływana po kliknięciu na element listy.

Tworzenie widoku: W metodzie onCreateViewHolder, adapter influje widok z pliku fragment_tatry/karkonosze.xml dla każdego elementu listy. Widok jest następnie przekazywany do wewnętrznej klasy Tatry/KarkonoszeViewHolder, która zarządza widokiem.

Przygotowanie danych: W metodzie onBindViewHolder, adapter pobiera dane dla aktualnie wyświetlanego elementu listy (szlak) i ustawia zasoby obrazu oraz tekst dla ImageView i TextView odpowiednio. Dodatkowo, ustawia słuchacza kliknięć dla elementu listy, który wywołuje metodę onItemClick zdefiniowaną w interfejsie OnItemClickListener.

Licznik elementów: Metoda getItemCount zwraca liczbę elementów w liście szlaków, co informuje RecyclerView o liczbie elementów, które ma wyświetlić.

Wewnętrzna klasa : Tatry/KarkonoszeViewHolder jest wewnętrzną klasą adaptera, która przechowuje referencje do ImageView i TextView w widoku elementu listy. Pozwala to na łatwe ustawianie danych dla każdego elementu listy.

f)FragmentZegar

Stan fragmentu: Fragment przechowuje stan zegara, w tym liczbę sekund, czas, który upłynął, czy zegar jest aktualnie uruchomiony, czy był uruchomiony przed przejściem do stanu przejściowego, oraz nazwę wybranej ścieżki.

Restauracja stanu: W metodzie onCreate, fragment restauruje stan zegara z savedInstanceState, co pozwala na zachowanie czasu, który upłynął, oraz historii ścieżek, nawet po zmianie konfiguracji urządzenia.

Interakcje użytkownika: W metodzie onCreateView, fragment influje swój widok z pliku fragment_zegar.xml i konfiguruje interakcje z użytkownikiem, takie jak startowanie, stopowanie i resetowanie zegara, a także dodawanie nazwy ścieżki do historii. Użytkownik może wprowadzić nazwę ścieżki za pomocą EditText i dodać ją do historii za pomocą przycisku.

Zegar: Funkcja runStoper uruchamia zegar, który aktualizuje czas na ekranie co sekundę. Zegar jest synchronizowany z systemowym czasem urządzenia, co pozwala na precyzyjne śledzenie upływu czasu.

Historia ścieżek: Historia ścieżek jest przechowywana w MutableList<String>, do której dodawane są wpisy z nazwą ścieżki i czasem, gdy zegar został zresetowany. Historia jest aktualizowana na ekranie za pomocą metody updateHistoryTextView.

Zachowanie stanu: Metoda onSaveInstanceState zapewnia, że stan zegara zostanie zachowany, nawet gdy fragment zostanie zniszczony i ponownie utworzony, na przykład po zmianie orientacji ekranu.

g) SzczegolyFragment

Fragment prezentuje szczegółowe informacje o trasach turystycznych w Tatrach i Karkonoszach.

Komponenty: TextView służy do wyświetlania opisu wybranej trasy. HideButton do ukrywania szczegółów trasy. FloatingActionButton (FAB) otwiera kamerę telefonu. ScrollView służy do nawigacji po treści.

Funkcjonalność: Przetwarzanie argumentów przekazanych do fragmentu w onCreate i wyświetlanie szczegółów trasy w onCreateView w zależności od nazwy trasy przekazanej wcześniej. Obsługa zgody na dostęp do kamerę w openCamera i otworzenie aparatu. Dynamiczne tworzenie fragmentu za pomocą newInstance, który przyjmuje nazwę trasy jako argument.

Interakcje: Użytkownik może wybrać trasę, aby zobaczyć jej szczegółowe informacje. Możliwość ukrywania szczegółów trasy za pomocą przycisku. Możliwość otwierania kamery.

2.Zrealizowane wymagania dodatkowe:

- Kod napisany w Kotlinie

- Możliwość wglądu do zapamiętanych wyników