Android JetPack

mgr inż. Stanisław Lota

Wprowadzenie

Android Jetpack to zestaw bibliotek, narzędzi i wskazówek, które pomagają programistom postępować zgodnie z najlepszymi praktykami, redukować standardowy kod i pisać kod, który działa spójnie w wersjach i urządzeniach z Androidem.

Obecnie prawie 99% aplikacji obecnych w sklepie Google Play korzysta z bibliotek Android Jetpack.

Wszystkie nowe wersje i biblioteki dla Androida mają na celu poprawę zarówno doświadczenia programisty, jak i interakcji z użytkownikiem

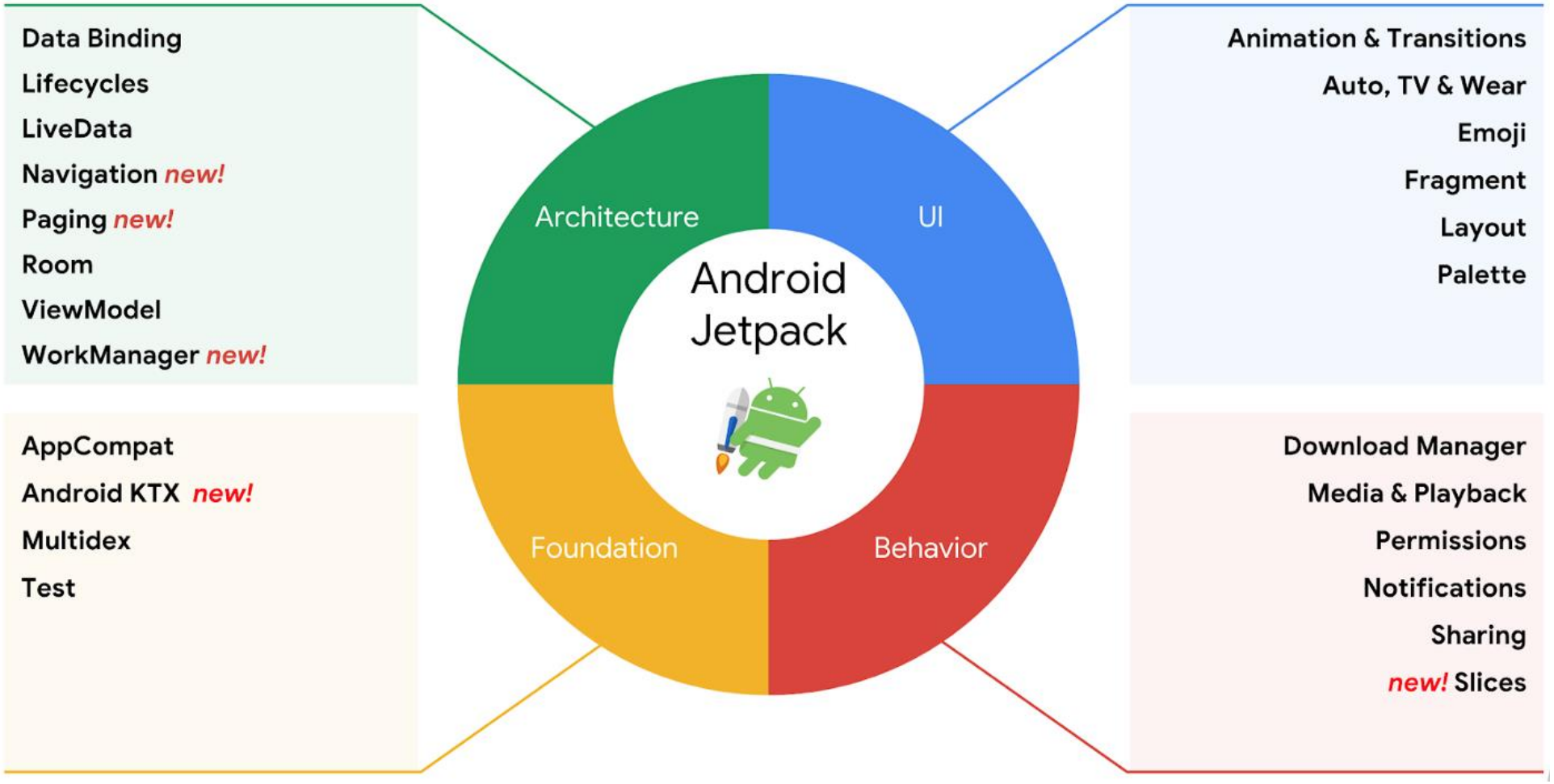
Wprowadzenie

Wszystkie komponenty Jetpack są dostępne w repozytorium Google – Maven, które należy dodać w swoim projekcie do : build.gradle().

Jetpack składa się z wcześniej istniejących bibliotek obsługi Androida, komponentów architektonicznych i biblioteki Android KTX (zestaw rozszerzeń Kotlin, które wykorzystują kilka funkcji języka Kotlin).

Według Google Jetpack jest obecnie używany przez 99% każdej aplikacji w Sklepie Play.

Architektura



Podstawowe komponenty

Podstawowe komponenty ułatwiają zgodność z poprzednimi wersjami, obsługę testów i obsługę języka Kotin.

AppCompat: Umożliwia dostęp do nowych interfejsów API w starszych wersjach interfejsu API platformy z obsługą implementacji interfejsu użytkownika do material design.

Android KTX: Jest to zestaw rozszerzeń Kotlin zawartych w Android Jetpack i innych bibliotekach Androida.

Multidex: Wykonuje podział systemowy pliku .dex aplikacji na wiele plików .dex i zapewnia obsługę wielu plików .dex.

Test: Zawiera strukturę testowania Espresso UI dla środowiska uruchomieniowego UI i AndroidJUnitRunner do testowania jednostkowego w systemie Android.

Składniki architektoniczne

Data binding: Format deklaratywny służący do łączenia elementów interfejsu użytkownika w Layoucie ze źródłem danych w aplikacji.

LiveCycles: Składniki uwzględniające cykl życia działają w odpowiedzi na zmianę stanu cyklu życia innego składnika, takiego jak aktywność i fragment.

LiveData: klasa, która aktualizuje obserwatorów składników aplikacji, którzy są w aktywnym stanie cyklu życia.

Navigation: obsługuj wszystkie aspekty nawigacji w aplikacji w tym projektowanie interfejsu za pomocą grafów bez użycia intentów.

Składniki architektoniczne

Pagging: Stopniowo ładuje i wyświetla małe fragmenty danych na żądanie ze źródła danych.

Room: Zapewnia płynny dostęp do bazy danych, jednocześnie wykorzystując SQLite.

ViewModel: Przechowuje i zarządza danymi związanymi z interfejsem użytkownika w sposób uwzględniający cykl życia i pozwala danym przetrwać zmiany konfiguracji, takie jak obracanie ekranu.

WorkManager: interfejs API, który ułatwia planowanie niezawodnych, asynchronicznych zadań, które mają zostać uruchomione, nawet jeśli aplikacja zostanie wycofana lub urządzenie uruchomi się ponownie.

Składniki behawioralne

Uprawnienia: interfejsy API do sprawdzania i żądania uprawnień aplikacji.

Preferencje: Umożliwia tworzenie interaktywnych ekranów ustawień

Udostępnianie: Udostępnia akcję udostępniania odpowiednią dla paska akcji aplikacji do udostępniania prostych danych (obrazu, tekstu lub pliku).

Slices: Tworzy elastyczne elementy użytkownika, które mogą wyświetlać dane aplikacji poza aplikacją.

Składniki behawioralne

Menedżer pobierania: Planowanie i zarządzanie długotrwałymi pobraniami HTTP.

Media & playback: Interfejsy API do odtwarzania i routingu multimediów, które są wstecznie kompatybilne.

Powiadomienia: Zapewnia wstecznie kompatybilny interfejs API powiadomień z obsługą Wear i Auto.

Komponenty interfejsu użytkownika

Animacja i przejścia: Umożliwia animowanie przejścia interfejsu użytkownika, widżetów dotyczących działań użytkownika.

Auto: Pomaga opracować aplikacje na Android Auto

Emoji: Pomaga aktualizować urządzenia z Androidem o najnowsze emotikony.

Fragment: Jednostka wielokrotnego użytku UI.

Układ: pomaga tworzyć strukturę układu aplikacji przy użyciu widżetów układu z różnymi algorytmami.

Komponenty interfejsu użytkownika

Paleta: wybieranie i stosowanie przydatnych informacji z palet kolorów, aby aplikacja była atrakcyjna wizualnie.

TV: interfejs API ułatwia tworzenie aplikacji na Android TV.

Wear: komponenty do tworzenia aplikacji Wear, które działają na inteligentniejszych ekranach, takich jak smartwatche.

Navigation

Do już istniejących Android Architecture Components została dodana biblioteka Navigation. Pozwala ona na tworzenie grafów z ekranami aktwyności definiujących nawigację w naszej aplikacji. Przy wykorzystaniu tej biblioteki nie musimy korzystać z intencji.

Grafy nawigacji tworzy się w graficznym edytorze do złudzenia przypominającym Interface Buildera dobrze znanego programistom korzystającym z Xcode.

Compose

Jetpack Compose to nowoczesny zestaw narzędzi do budowania natywnego systemu Android UI. Jetpack Compose upraszcza i przyspiesza tworzenie interfejsu użytkowników na Androida dzięki mniejszej ilości kodu i zaawansowanym narzędziom.

Nie trzeba w nim edytować żadnych układów XML ani używać Edytora układów. Zamiast tego można wywołać funkcje Jetpack Compose i wskazać jakie elementy chcemy, a kompilator Compose zrobi resztę za nas.

Android Cars

Nowa biblioteka Android for Cars, która została uruchomiona w najnowszej aktualizacji Jetpack, Do tej pory jedynymi dwiema aplikacjami nawigacyjnymi dostępnymi dla samochodów z Android Auto były Mapy Google i Waze, ale od końca zeszłego roku zaczęło się to zmieniać.Większość programistów przenosi swoje aplikacje do nowej biblioteki, aby dotrzeć do ponad 500 modeli pojazdów zgodnych z Android Auto.

Zadanie

Android Navigation

https://developer.android.com/codelabs/android-navigation

Android Compose

https://developer.android.com/codelabs/jetpack-compose-basics