Fragmenty w Android

mgr inż. Stanisław Lota

Fragmenty

Fragment to obiekt kontrolera, który może zostać wyznaczony przez aktywność do wykonania określonego zadania. Jest kontenerem dla danych elementów UI oraz dla funkcjonalności, które są im przypisane.

Zazwyczaj takie zadanie polega na zarządzaniu interfejsem użytkownika, który może zajmować cały dostępny obszar ekranu bądź tylko jego określoną część.

Fragmenty

W wersji Android 3.0 (Honeycomb) wprowadzono pojęcie fragmentu reprezentowanego przez klasę Fragment. Następnie obsługę fragmentów dodano do biblioteki wsparcia systemu Android, tak by można ich było używać w Androidzie, zaczynając od wersji 1.6 (API poziomu 4).

Fragment definiuje i zarządza własnym układem, ma własny cykl życia i może obsługiwać własne zdarzenia wejściowe. Dzięki wykorzystaniu takich niezależnych kontenerów, nasza aplikacja staje się modularna.

Fragmenty

Fragment zarządzający interfejsem aplikacji jest nazywany **fragmentem interfejsu użytkownika** (ang. UI fragment).

Każdy fragment interfejsu użytkownika posiada swój własny widok rozwijany z pliku układu.

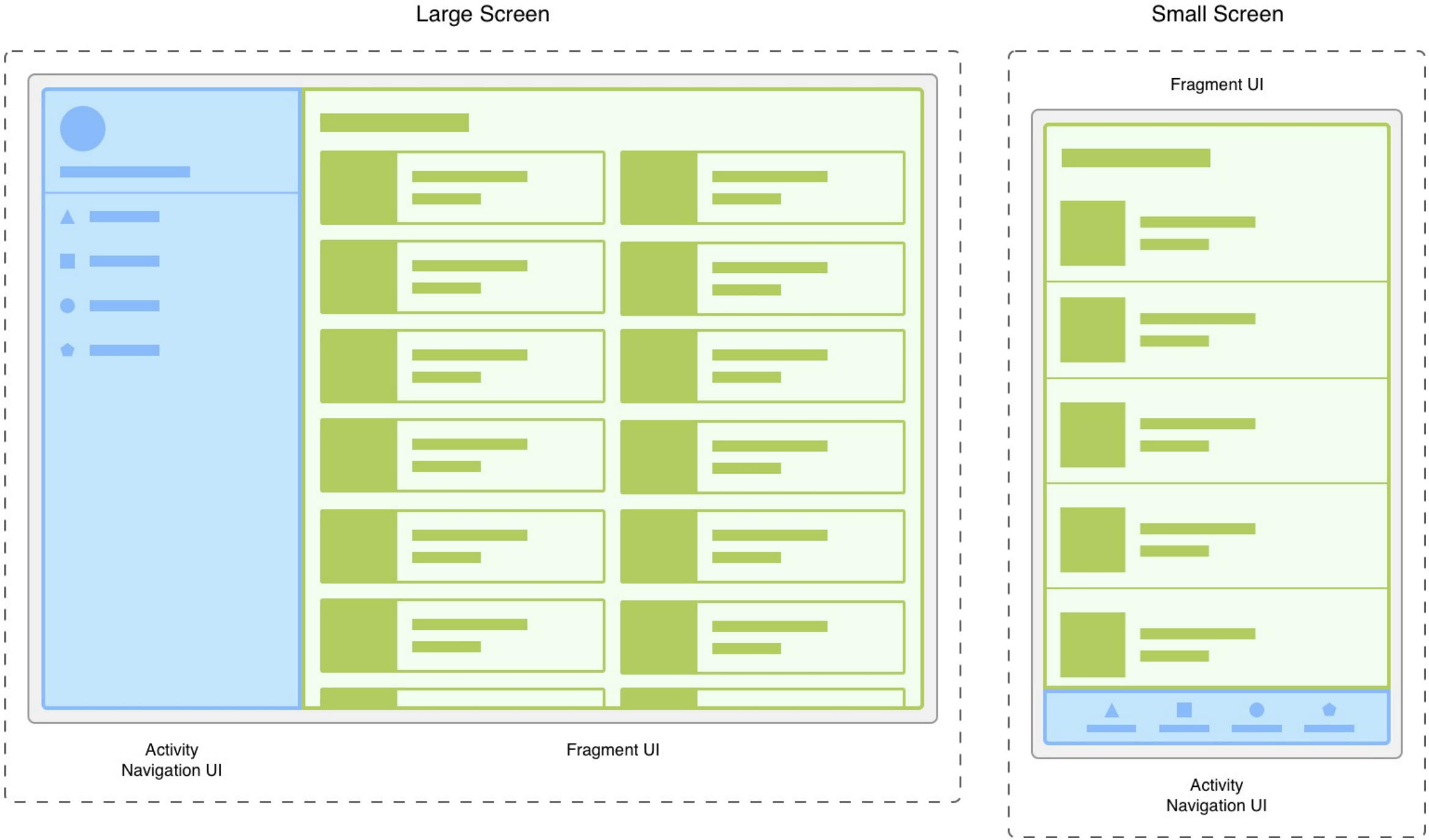
Widok fragmentu zawiera takie elementy interfejsu, które użytkownik będzie widział na ekranie i z którymi będzie prowadził interakcje.

Fragmenty

Fragmenty pomagają również podzielić kod na łatwe do zarządzania kawałki, bez potrzeby polegania na dużych i skomplikowanych klasach Activity.

Fragmenty pozwalają na łatwe poruszanie się w aplikacji i umożliwiają proste komunikowanie się między różnymi sekcjami aplikacji.

Klasy aktywności mogą dowolnie łączyć i stosować komponenty interfejsu użytkownika bądź zachowania, tworząc z ich pomocą znacznie bardziej elastyczny interfejs użytkownika.


Large Screen
Small Screen
Fragment UI
Activity
Navigation UI
Fragment UI
Activity
Navigation UI

Fragmenty

Na rysunku przedstawione są dwie wersje tego samego ekranu na różnych rozmiarach ekranu.

Po lewej stronie duży ekran zawiera szufladę nawigacji(Navigation Drawer), która jest kontrolowana przez aktywność, oraz listę siatki kontrolowaną przez fragment.

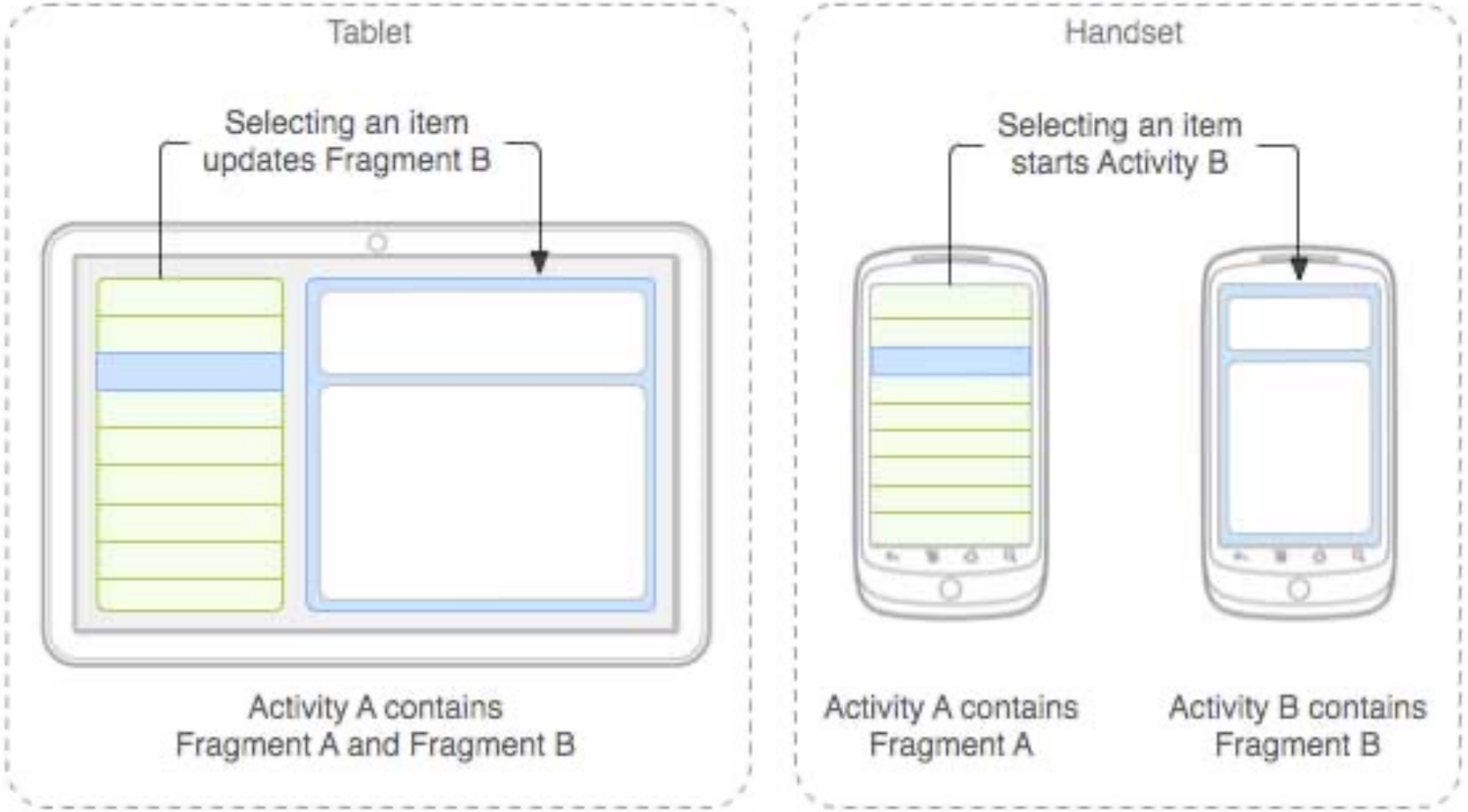
Po prawej stronie mały ekran zawiera dolny pasek nawigacji kontrolowany przez aktywność oraz liniową listę kontrolowaną przez fragment.

Fragmenty

Można tworzyć układy jednopanelowe dla telefonów i układy wielopanelowe dla tabletów. Możemy także użyć fragmentów do obsługi innego układu dla orientacji poziomej i pionowej na smartfonie.

Obecnie stosowane są nie tylko smartfony, lecz także urządzenia z większymi ekranami, takie jak tablety i telewizory, także działające pod kontrolą Androida. Urządzenia te dysponują znacznie większą powierzchnią ekranu, którą mogą wykorzystać programiści.

Fragmenty


Tablet
Selecting an item updates Fragment B
Activity A contains Fragment A and Fragment B
Handset
Selecting an item starts Activity B
Activity A contains Fragment A
Activity B contains Fragment B

Fragmenty

Użycie fragmentu w aplikacji na Androida całkowicie zależy od rozmiaru ekranu urządzenia, na którym aplikacja jest używana. Jeśli rozmiar ekranu jest duży, możemy łatwo pokazać 2 lub może więcej fragmentów na ekranie, ale jeśli rozmiar ekranu jest mniejszy, zaleca się użycie Fragmentów w osobnych czynnościach.

Fragmenty

Ponieważ możliwe jest dynamiczne dodawanie i usuwanie fragmentów czynności wykorzystanie fragmentów pozwala na projektowanie bardzo elastycznych interfejsów użytkownika.

Elastyczność interfejsu użytkownika to zdolność do dynamicznej zmiany widoku aktywności w czasie działania aplikacji w zależności od wymagań użytkownika bądź możliwości urządzenia.

Fragmenty a Aktywności

Aktywności nie są zbudowane w sposób ułatwiający wprowadzenie takiej elastyczności.

Widoki aktywności mogą się co prawda zmieniać w czasie działania programu, ale cały kod sterujący widokami musi funkcjonować wewnątrz takiej aktywności.

W rezultacie takiego rozwiązania poszczególne aktywności są bardzo ściśle powiązane z rozmiarem i rozdzielczością ekranu aktualnie używanego urządzenia.

Fragmenty

Fragmenty Androida mają swój własny cykl życia, bardzo podobny do aktywności Androida.

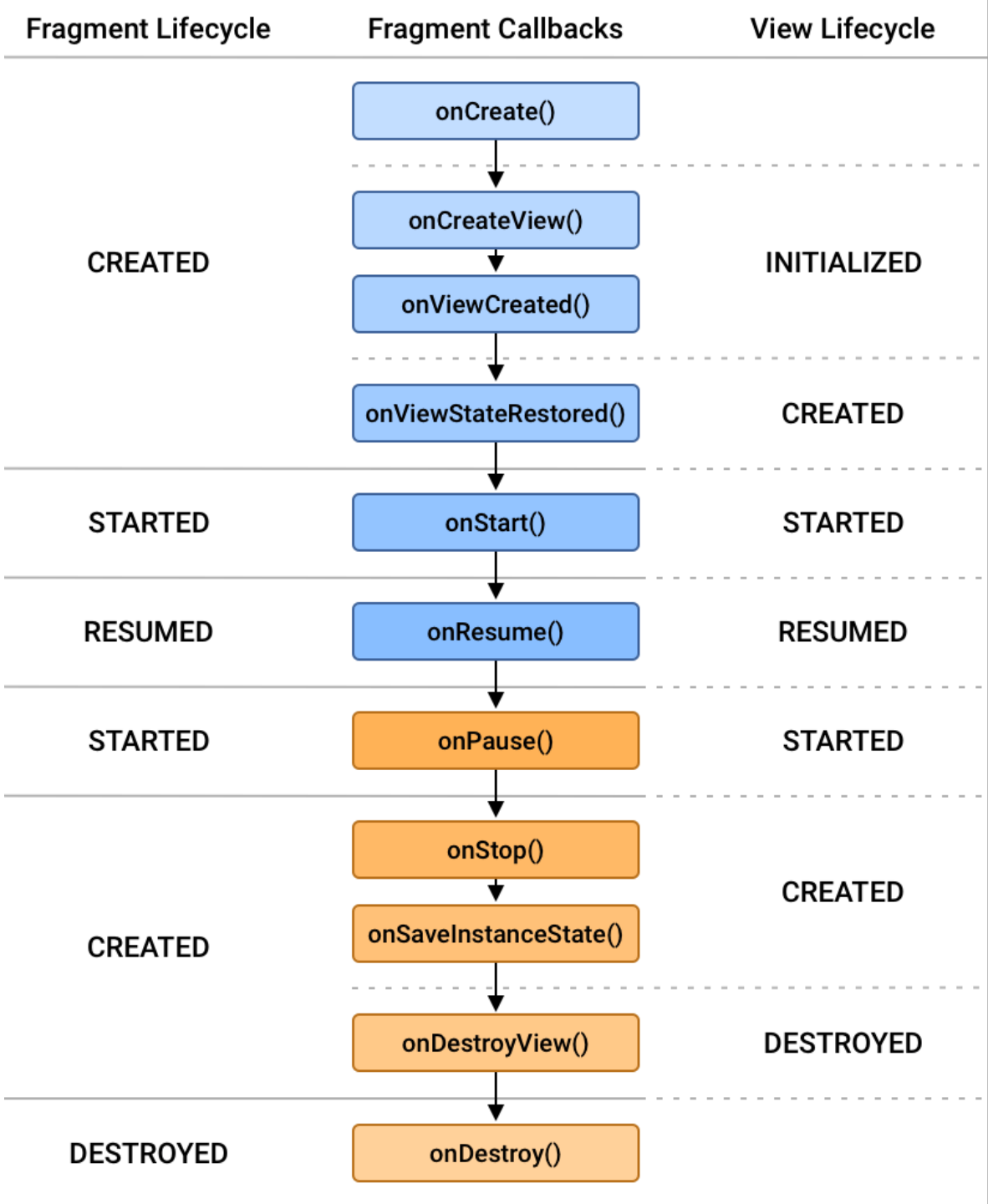
Posiadanie przez fragment stanów cyklu życia odpowiadających stanom aktywności jest bardzo ważne. Jest to spowodowane tym, że fragmenty są hostowane przez określone aktywności, a zatem muszą posiadać odpowiednie metody cyklu życia, za pomocą których mogą współpracować z określonymi stanami cyklu życia aktywności.

Cykl życia fragmentów

Jedna z najważniejszych różnic pomiędzy cyklem życia fragmentu a cyklem życia aktywności polega na tym, że metody cyklu życia fragmentu są wywoływane przez aktywność hostującą dany fragment, a nie przez system operacyjny urządzenia, na którym działa aplikacja.

System operacyjny nie ma żadnej wiedzy na temat fragmentów, których aktywność używa do zarządzania elementami wyświetlanymi na ekranie. Inaczej mówiąc, używanie fragmentów to wewnętrzna sprawa danej aktywności.

Cykl życia fragmentów



Cykl życia fragmentów

Metoda zwrotna onAttach() jest wywoływana podczas pierwszego dołączania fragmentu do aktywności.

Metoda zwrotna onCreate() jest wywoływana w momencie pierwszego tworzenia fragmentu.

Metoda zwrotna onCreateView() jest wywoływana w chwili, gdy powinien być utworzony układ bądź hierarchia kontrolek View zawierająca dany fragment.

Cykl życia fragmentów

Metoda zwrotna onActivityCreated() jest wywoływana w celu „poinformowania” fragmentu o zakończeniu wykonywania metody onCreate() nadrzędnej aktywności, do której został on dołączony.

Metoda zwrotna onStart() jest wywoływana, kiedy interfejs użytkownika fragmentu został już utworzony, lecz jeszcze nie jest aktywny.

Cykl życia fragmentów

Metoda zwrotna onResume() jest wywoływana, kiedy interfejs użytkownika fragmentu stanie się aktywny i gotowy do prowadzenia interakcji z użytkownikiem po wcześniejszym wznowieniu aktywności lub aktualizacji fragmentu przy użyciu obiektu FragmentTransaction.

Metoda zwrotna onPause() jest wywoływana po wstrzymaniu wykonywania nadrzędnej aktywności lub po aktualizacji fragmentu z wykorzystaniem obiektu FragmentTransaction. Jej wywołanie oznacza, że fragment nie jest już dłużej aktywny bądź że jest obecnie wyświetlany w tle.

Cykl życia fragmentów

Metoda zwrotna onStop() jest wywoływana po zatrzymaniu wykonywania nadrzędnej aktywności lub po aktualizacji fragmentu za pomocą obiektu FragmentTransaction. Jej wywołanie oznacza, że fragment nie jest już widoczny.

Metoda zwrotna onDestroyView() jest wywoływana w celu wyczyszczenia zasobów skojarzonych z danym fragmentem, używanych przez układ lub hierarchię kontrolek View.

Cykl życia fragmentów

Metoda zwrotna onDestroy() jest wywoływana w celu wyczyszczenia wszelkich pozostałych zasobów używanych przez fragment.

Metoda zwrotna onDetach() jest wywoływana bezpośrednio przed odłączeniem fragmentu od aktywności.

Fragment

FragmentManager to klasa odpowiedzialna za wykonywanie działań na fragmentach aplikacji, takich jak dodawanie, usuwanie lub zastępowanie ich oraz dodawanie ich do stosu.

Jeżeli mamy zamiar użyć aktywności do hostowania fragmentów, mamy do wyboru dwa sposoby postępowania.

Fragmenty

W aktywności definiujemy FrameLayout.

Jest to kontener na fragmenty. W zależności co chce zobaczyć użytkownik, dynamicznie podmieniamy fragment, który jest aktualnie wyświetlany (zamiast tworzyć nową aktywność).

Tworzymy fragmenty statycznie.

Przykładowo w jednej aktywności wyświetlamy dwa fragmenty obok siebie jeśli jest wystarczająco miejsca. Jeśli nie, tworzymy oddzielne aktywności dla każdego fragmentu.

Fragmenty

Pierwszy sposób jest określany jako używanie **fragmentów układu** i jest bardzo prosty, ale jednocześnie mało elastyczny.

Jeżeli dodajemy fragment do układu aktywności, to dokonujemy sztywnego powiązania fragmentu i jego widoku z tym układem, dlatego w czasie cyklu życia aktywności nie będziemy mieć możliwości podmiany takiego fragmentu.

Fragmenty

Drugie podejście, czyli dodawanie fragmentów do kodu aktywności, jest znacznie bardziej złożone, ale jest jedynym sposobem dającym pełną kontrolę nad fragmentami podczas działania aplikacji.

Dzięki takiemu rozwiązaniu możemy w dowolny sposób decydować o tym, kiedy dany fragment jest dodawany do aktywności i co dzieje się później.

Możemy na przykład usunąć dany fragment, zamienić go na inny, a później ponownie dodać do aktywności ten pierwszy fragment.

Podklasy fragmentów

Ponieważ fragmenty zostały opracowane w celu odseparowania tej możliwości funkcjonalnej od klas aktywności, obecnie można znaleźć klasy pochodne klasy Fragment, odpowiadające wyspecjalizowanym klasom aktywności.

ListFragment ułatwia zarządzanie kontrolką ListView.

PreferenceFragment ułatwia zarządzanie preferencjami użytkownika.

WebViewFragment zawiera kontrolkę WebView, dzięki czemu bez trudu można w nim wyświetlać treści z internetu.

Podklasy fragmentów

MapFragment umożliwia wyświetlenie komponentu MapView z usługą Mapy Google.

BrowseFragment i DetailsFragment obie te klasy Fragment są dostępne jako część bibliotek wsparcia dla Android TV i obsługują interfejs użytkownika i podstawową funkcjonalność przy tworzeniu aplikacji dla telewizorów.

DialogFragment - do wyświetlania ruchomego okna dialogowego

Fragmenty

Fragmenty znacznie zwiększają możliwości wielokrotnego stosowania kodu, upraszcza proces testowania aplikacji oraz ułatwia publikowanie pakietu aplikacji oraz zarządzanie nim.

Android udostępnia Fragment API zapewniający dwa sposoby używania efektów ruchu i transformacji do wizualnego łączenia fragmentów podczas nawigacji co skutkuje dodaniem animacji w trakcie przechodzenia między fragmentami.

Zadanie

Wykonaj oficjalne ćwiczenie z Google Developer :

https://developer.android.com/codelabs/kotlin-android-training-create-and-add-fragment#0