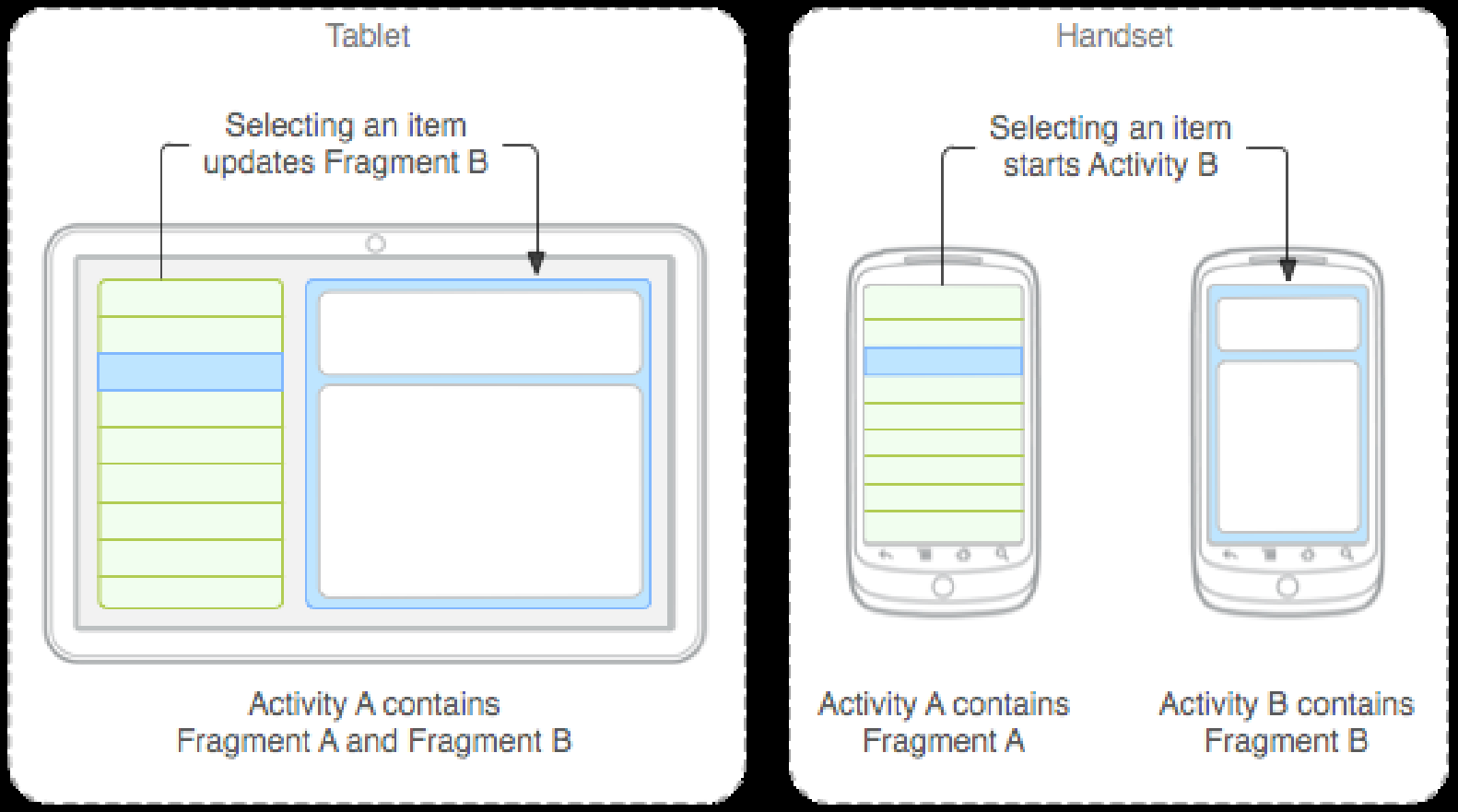
O fragmentach w Androidzie

Czym jest fragment?

- Modularna część aktywności
- Ma swój własny cykl życia
- Jest zawsze zagnieżdżony w aktywności i jego cykl życia jest powiązany z hostującą aktywnością

Po co fragmenty?



W jaki sposób zarządzać fragmentami?

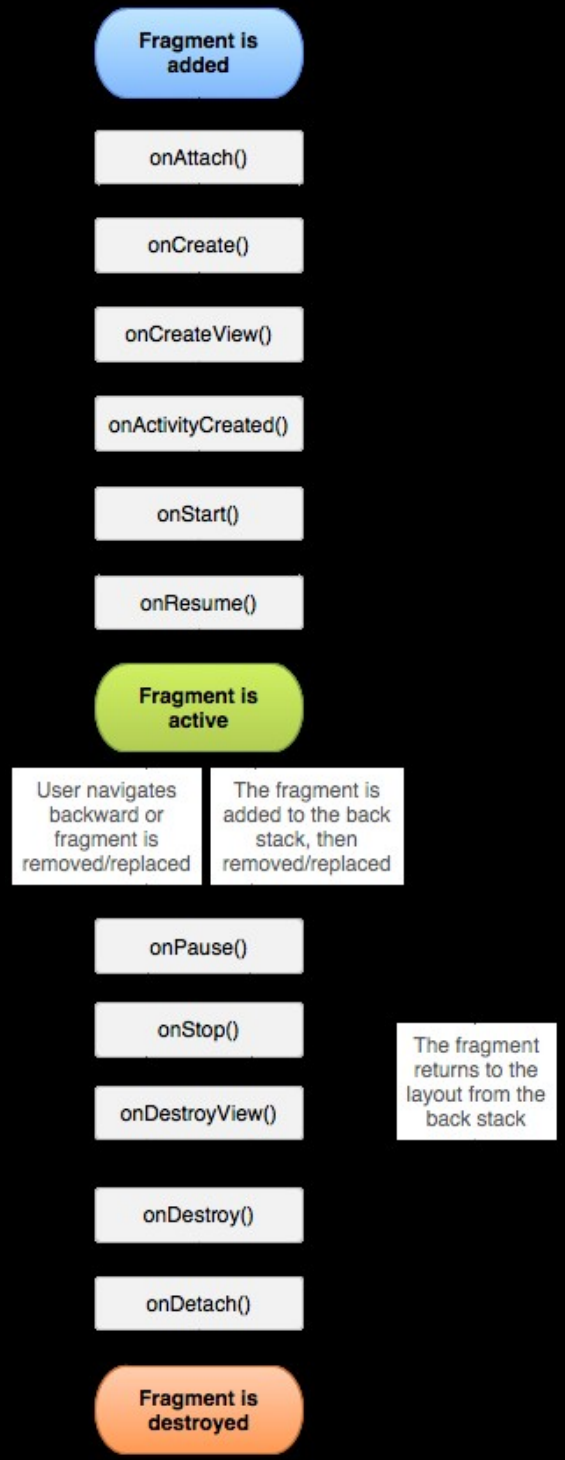
- Fragment można dodawać/podmieniać/usuwać w ramach danej aktywności
- Możemy traktować fragmenty jako komponenty, z którymi możemy kontaktować się poprzez udostępniony przez nie interfejs
- Zdecydowanie nie w ten sposób: ((TextView)fragment.getView().findViewByID(R.id.name_fragment_container)).setText(“Ala”), lepiej przez interfejs fragmentu: fragment.setName(“Ala”)

Jak otrzymać instancję fragmentu?

- Można po prostu utworzyć instancję fragment poprzez jego metodę statyczną newInstance(...), można też po prostu skorzystać z new jeśli nie ma argumentów
- Można też skorzystać z jakiegoś sposobu na luźne wiązanie, np. Dagger (czy inny kontener DI) lub skorzystać z antycznego wzorca fabryki abstrakcyjnej
- Niestety, aby dodać fragment musimy dokonać rzutowania (jeśli gadamy tylko przez jego interfejs), ze względu na politykę Androida (niewiele rzeczy dzieje się tam za pośrednictwem interfejsów, a dziedziczenie jest powszechną (złą?) praktyką)

Fragmenty z perspektywy początkującego programisty Androida

- Trzeba bardzo uważać na cykl życia Androida, np. może się zdarzyć, że coś może nie być zainicjalizowane
- Android automatycznie próbuje odtworzyć cały stan fragmentu, więc wiele rzeczy może się popsuć, jeśli tego stanu nie jesteśmy w stanie łatwo zapisać (obiekty złożone/serializacja/referencje)
- Poza tym, Android próbuje wykonać powyższy krok zawsze, nawet jeśli nie umieściliśmy jawnie naszego fragmentu w metodzie onCreate hostującej aktywności


Fragment is added
onAttach()
onCreate()
onCreateView()
onActivityCreated()
onStart()
onResume()
Fragment is active
User navigates backward or fragment is removed/replaced
The fragment is added to the back stack, then removed/replaced
onPause()
onStop()
onDestroyView()
The fragment returns to the layout from the back stack
onDestroy()
onDetach()
Fragment is destroyed

Komunikacja z aktywnością

- Komunikacja z hostującą aktywnością zazwyczaj odbywa się przez interfejs aktywności, np. aktywność może zdecydować, że chce zostać powiadomiony, gdy wewnątrz fragmentu kliknięty zostanie button
- Aktywność może albo jawnie się zarejstrować, albo fragment może “po cichu” założyć, że aktywność implementuje odpowiedni interfejs

Komunikacja między fragmentami

- Najczęściej za pośrednictwem hostującej aktywności
- Zazwyczaj nie ma potrzeby, aby fragmenty jawnie o sobie wiedziały

Zapisywanie stanu fragmentu

- Nie jest to przyjemne ani proste (Parcelable, JSON)
- SetRetainInstance(boolean) // często nadużywana (uważać na zmiany w lifecycle, w szczególności na różnice między onCreate() a onCreateView())
- onSavedInstanceState(Bundle)

Odtwarzanie stanu fragmentu

- W metodzie onCreateView(...) bądź onCreate(...) odczytując stan z argumentu Bundle
- Deserializacja
- Odtwarzanie z jakiegoś zewnętrznego źródła
- Bądź po prostu całkowite tworzenie fragmentu od nowa z przekazanymi parametrami (należy uwazać na fakt, że Android i tak spróbuje na siłę przeprowadzić cały cykl fragmentu)
- Można spróbować obejść ten problem za pomocą wywołania setRetainInstance(true)

Trochę więcej o Daggerze

- Prosty kontener DI, dobry na potrzeby Androida
- Wykorzystuje graf zależności zdefiniowany przez Moduły
- Statyczna (!) analiza grafu zależności
- Dzięki temu szybki
- Ten graf zależności powinien trafić do globalnego (ble, ale Android) obiektu Application (musi zostać zdefiniowany w Manifeście)