Programación multimedia y dispositivos móviles

9

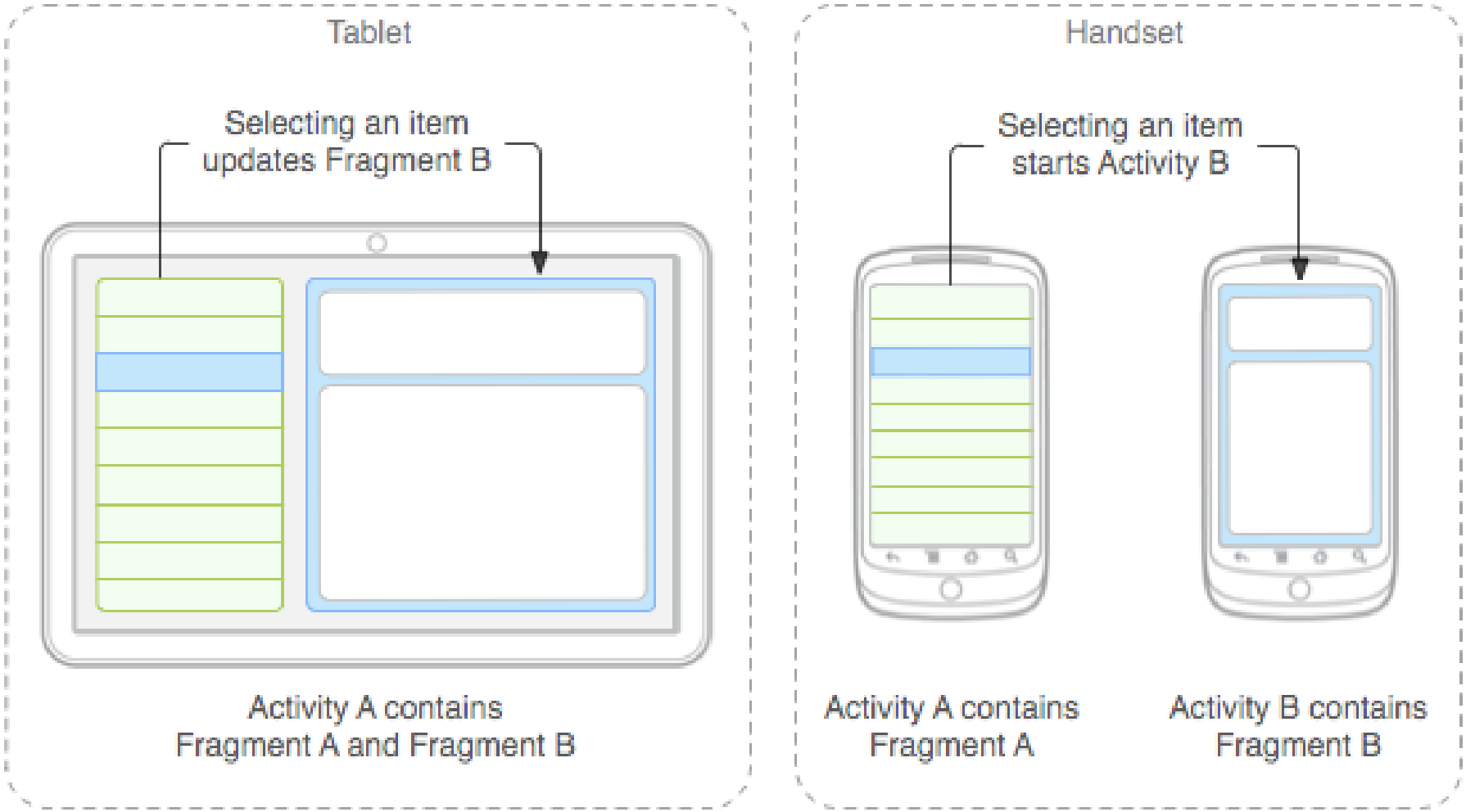
Fragments

IES Nervión
Miguel A. Casado Alías

Introducción

- Existen a partir de Android 3.0 (API 11)
- Se pueden ver como “subactividades”
- Tienen su propio ciclo de vida
- Se pueden agregar o quitar de la actividad en tiempo de ejecución
- Se pueden reutilizar en distintas actividades
- Permiten diseñar interfaces de usuario más flexibles y dinámicas en pantallas grandes (p.ej: Tablets)
- Podemos tener fragments sin interfaz gráfica

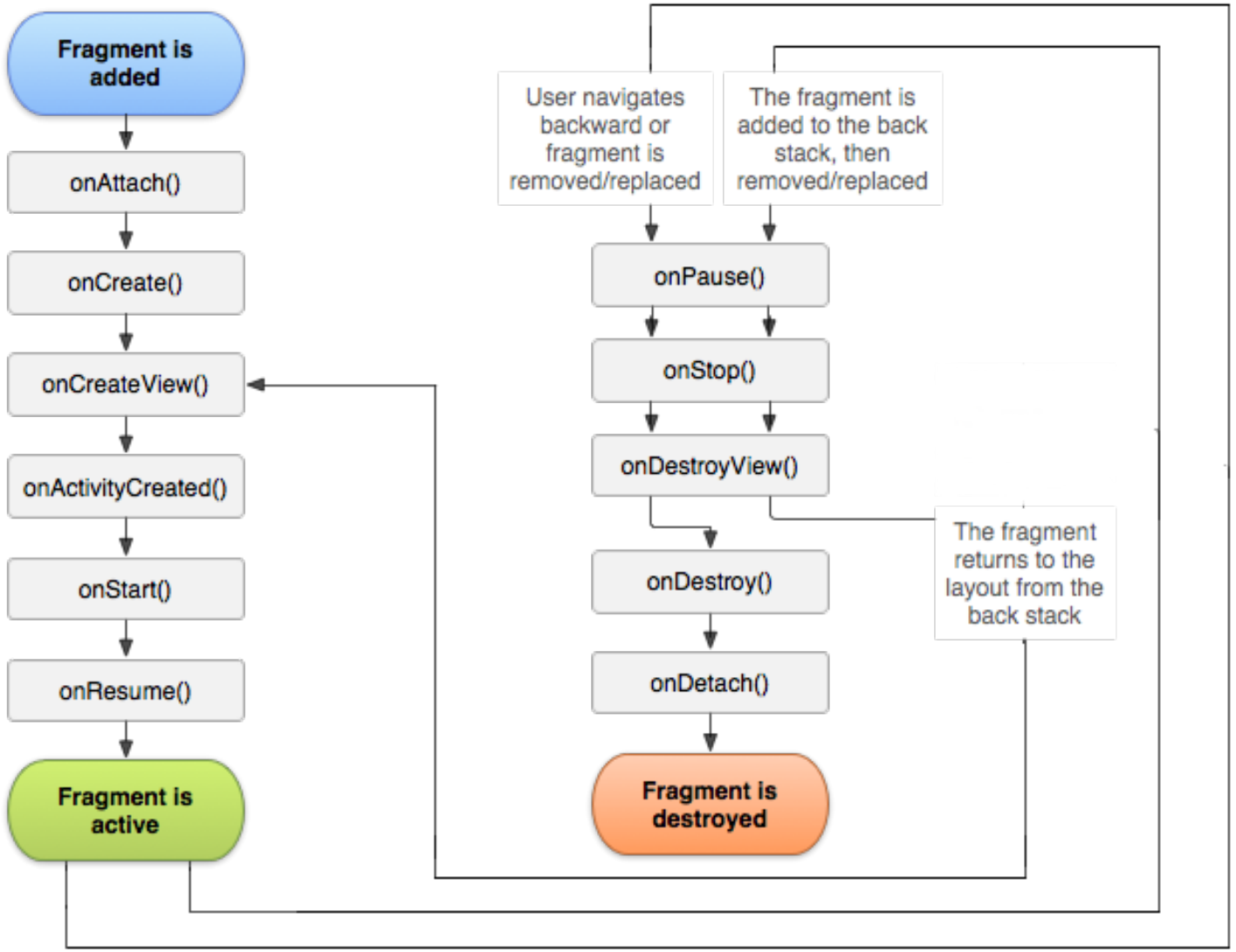
Filosofía de diseño



Filosofía de diseño (II)

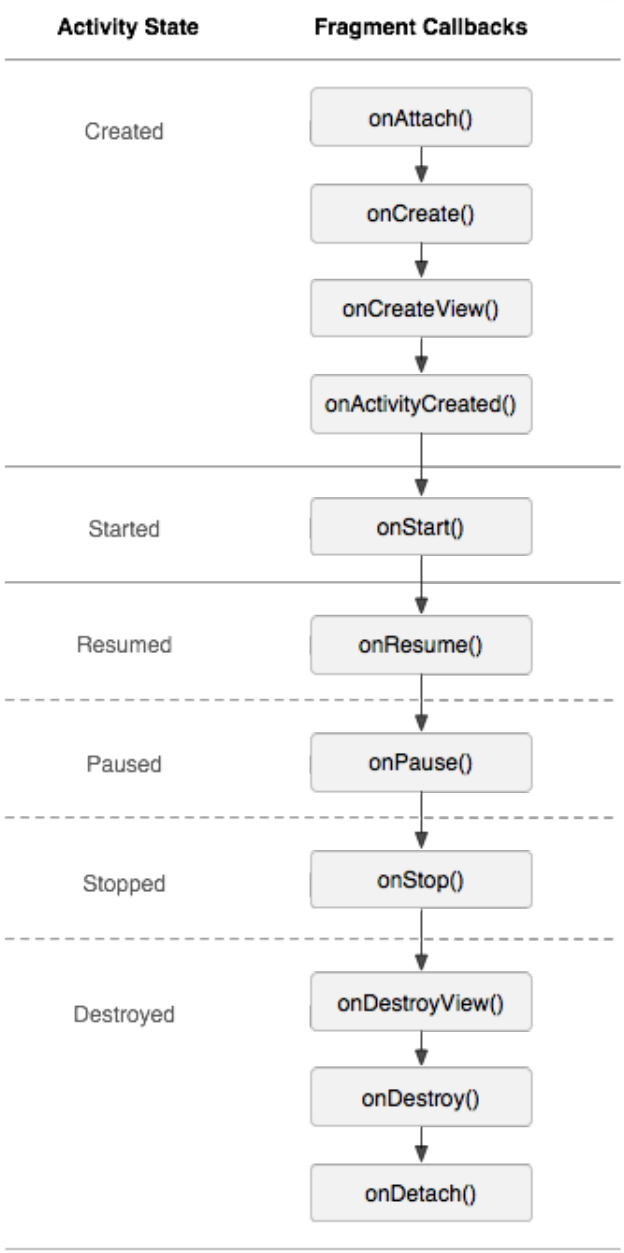
- Cada fragment define su comportamiento y su interfaz de usuario, por lo tanto, el mismo fragment se debe poder usar en distintas actividades
- Cada fragment debe ser independiente de los demás, para permitir modularidad y reusabilidad
- **NO SE DEBE MANIPULAR UN FRAGMENT DESDE OTRO**
- Al dividir la interfaz de una actividad en distintos fragments, podemos cambiar su apariencia fácilmente en tiempo de ejecución (intercambiando unos fragments por otros)
- Podemos guardar esos intercambios de fragments en una pila de retroceso (back stack), para así permitir al usuario navegar hacia atrás.

Ciclo de vida



Ciclo de Vida (II)

- El ciclo de vida de la actividad en la que reside el fragmento afecta directamente al fragmento
- Cada callback del ciclo de vida para la actividad genera un callback similar para cada fragmento de la misma
- Los fragmentos tienen algunos callbacks del ciclo de vida adicionales que abordan la interacción única con la actividad para poder realizar acciones como crear y destruir la IU del fragmento.
- Cuando una actividad está en “resumed” se pueden agregar y quitar fragmentos, así que el ciclo de vida de estos fragmentos cambiaría independientemente del de la actividad



Callbacks adicionales de los Fragments

- onAttach: llamado cuando el fragmento se ha asociado con la actividad
- onCreateView: llamado para crear la jerarquía de vistas asociada con el fragmento. Debe devolver una vista (o null)
- onActivityCreated(): llamado cuando el método onCreate() de la actividad finaliza
- onDestroyView(): llamado cuando la jerarquía de vistas del fragmento se va a destruir
- onDetach(): llamado cuando se desasocia el fragmento de la actividad

Pila de retroceso (Back Stack)

- Cuando una actividad se para, el sistema la añade automáticamente a una pila de retroceso. Cuando el usuario presiona el botón “Atrás” el sistema navega hacia atrás sacando actividades de dicha pila
- Para los fragments existe una pila de retroceso gestionada por la actividad a la que pertenecen
- Debemos añadir manualmente los fragments a la pila cuando los reemplazamos por otro si queremos que el usuario pueda navegar hacia un fragmento previo
- addToBackStack() es el método para añadir un fragment a la pila. Se le pasa el nombre que le queremos dar al estado o null

Obtener el contexto

- Para obtener el contexto desde un fragment podemos hacer uso de varios métodos, como por ejemplo:
  - getContext()
  - getApplication()
- No obstante, si hacemos la llamada cuando el fragmento todavía no ha sido asociado a la actividad (onAttach) o después de que ha sido desasociado (onDetach), esos métodos devolverán *null*

Salvar el estado de la instancia

- Se puede salvar el estado de la instancia de un fragmento de forma similar a como se hace con una actividad, es decir con ayuda de onSaveInstanceState y ViewModel
- Lo que guardemos en onSaveInstanceState lo podemos recuperar en:
  - onCreate
  - onCreateView
  - onActivityCreated