



Programming with Android: **Android Fragments**



Luca Bedogni

Marco Di Felice

Dipartimento di Scienze dell'Informazione

Università di Bologna



Outline

Android for Tablets: *A Case Study*

Android for Tablets: *Fragments Design Philosophy*

Android for Tablets: *Fragments Creation*

Android for Tablets: *Fragments Layout*

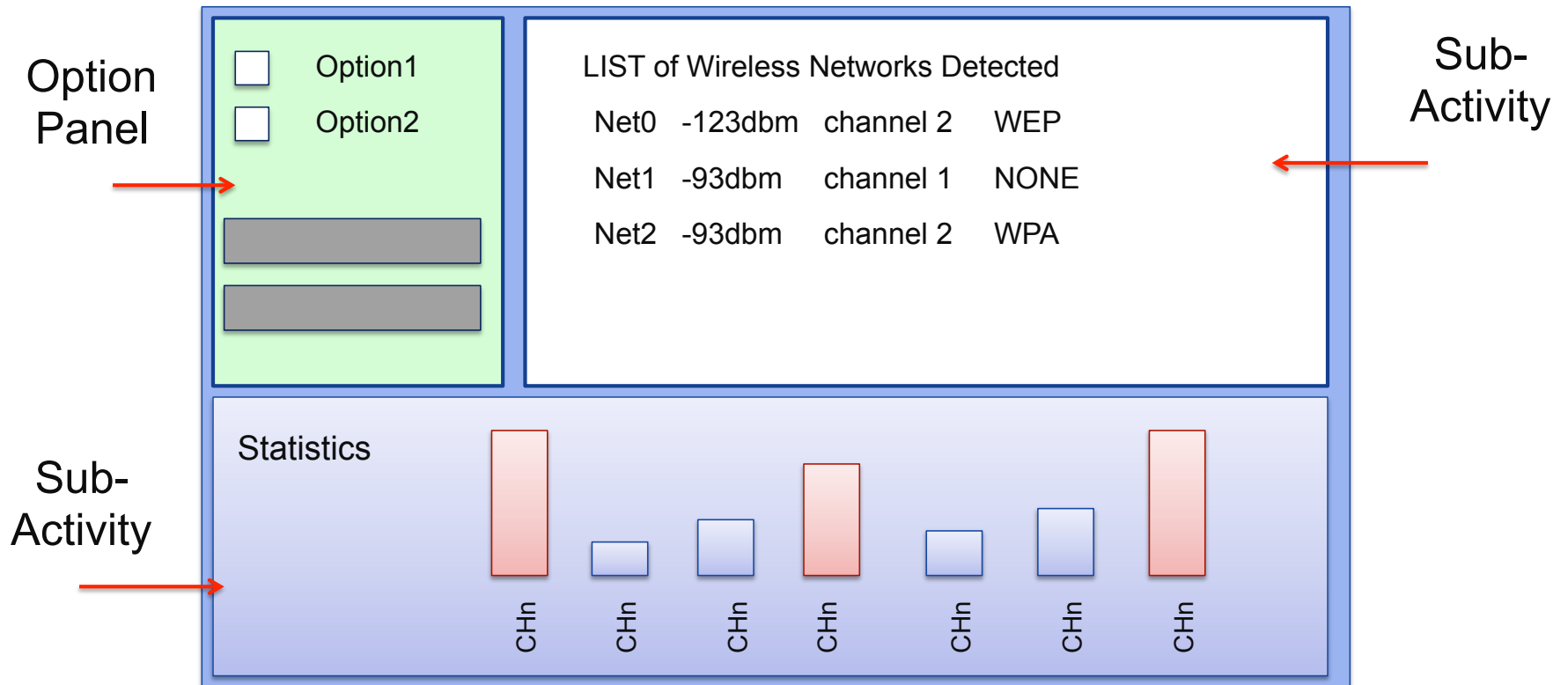
Android for Tablets: *Fragments Lifecycle*

Android for Tablets: *Fragments Transactions*

Android for Tablets: *Fragments Back State*



Android: Application Case Study





Android: **Fragments**

Fragment → A portion of the user interface in an Activity.

Introduced from **Android 3.0** (API Level 11)

Practically, a Fragment is a modular section of an Activity.

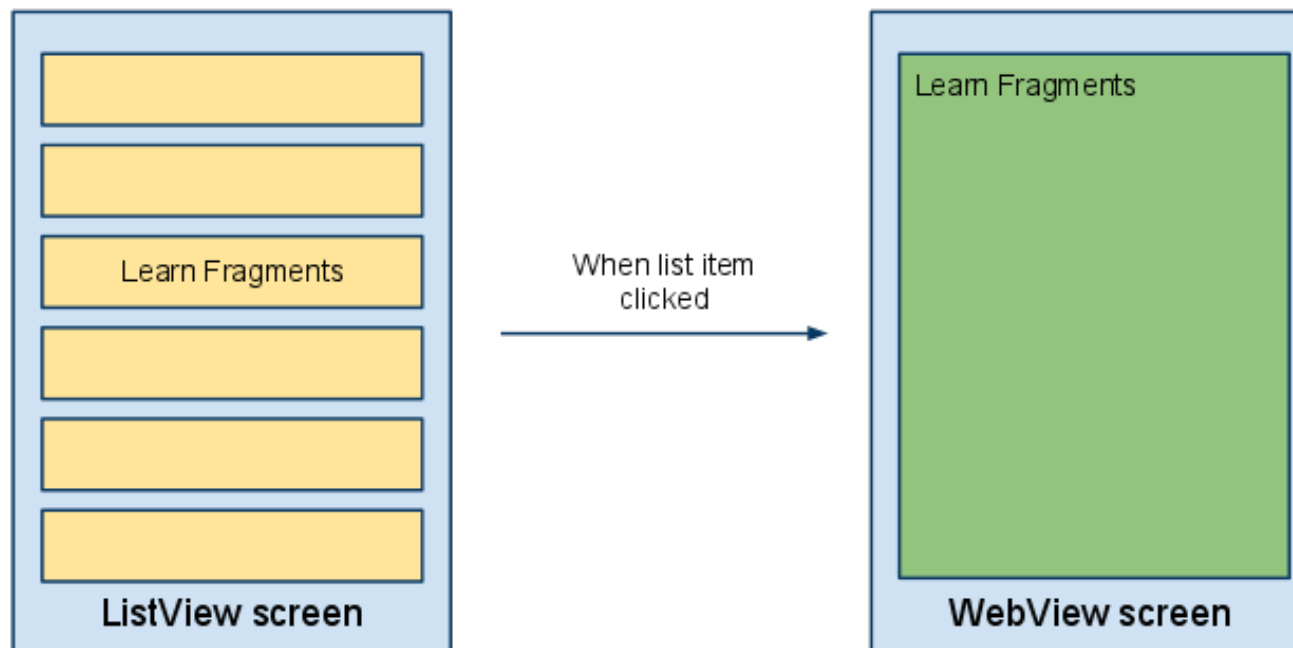
DESIGN PHILOSOPHY

- **Structure** an Activity as a collection of Fragments.
- **Reuse** a Fragment on different Activities ...



Android: **Fragments Design Philosophy**

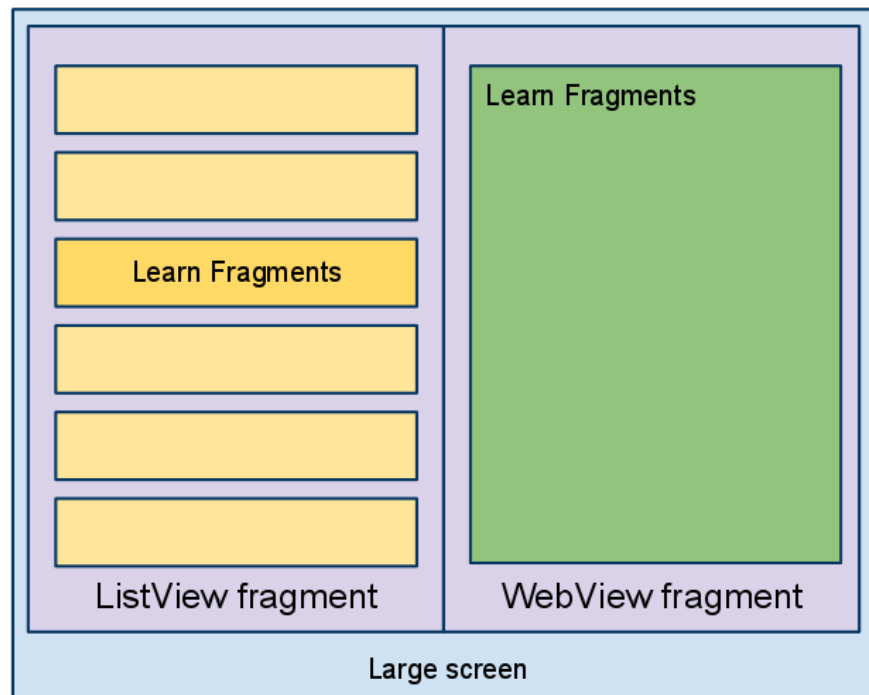
EXAMPLE: Using Fragments with Smartphone layout





Android: **Fragments Design Philosophy**

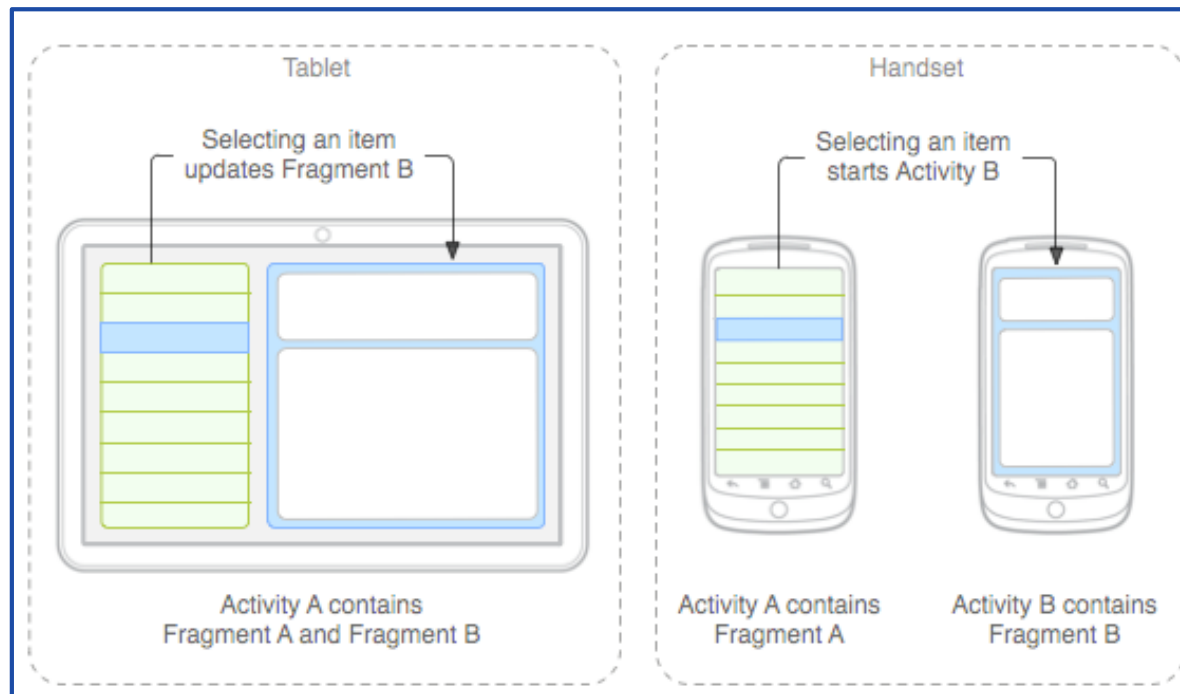
EXAMPLE: Using Fragments with Tablet layout





Android: **Fragment Transactions**

EXAMPLE: Using Fragments on Different Devices (Smartphone/Tab)





Android: Adding a Fragment to the UI

Specify layout properties for the Fragment as it were a View.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="horizontal" >
    <fragment android:name="it.cs.android30.FragmentOne"
        android:id="@+id/f1"
        android:layout_width="wrap_content"
        android:layout_height="fill_parent"
        />
    <fragment android:name="it.cs.android30.FragmentTwo"
        android:id="@+id/f2"
        android:layout_width="wrap_content"
        android:layout_height="fill_parent"
        />
</LinearLayout>
```



Android: **Fragment Creation**

To define a new Fragment → create a subclass of Fragment.

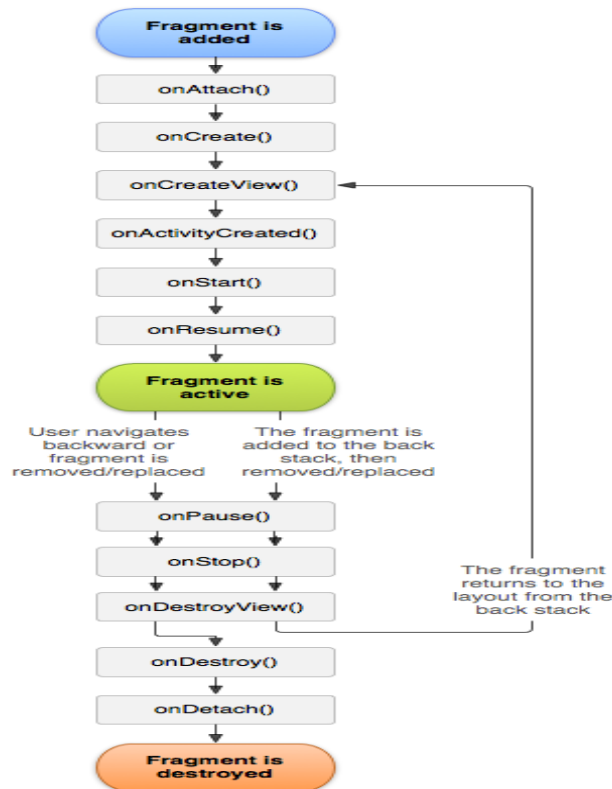
```
public class MyFragment extends Fragment { ... }
```

PROPERTY of a Fragment:

- Has its own **lifecycle** (partially connected with the Activity lifecycle)
- Has its own **layout** (or may have)
- Can receive its own **input events**
- Can be added or removed while the Activity is running.



Android: **Fragment Lifecycle**



Several **callback methods** to handle various stages of a Fragment lifecycle:

`onCreate()` → called when creating the Fragment.

`onCreateView()` → called when it is time for the Fragment to draw the user interface the first time.



Android: **Fragment Lifecycle**

CALLED when FRAGMENT is CREATED

**onAttach()
onCreate()
onCreateView()
onActivityCreated()
onStart()
onResume()**

CALLED when FRAGMENT is DESTROYED

**onPause()
onStop()
onDestroyView()
onDestroy()
onDetach()**



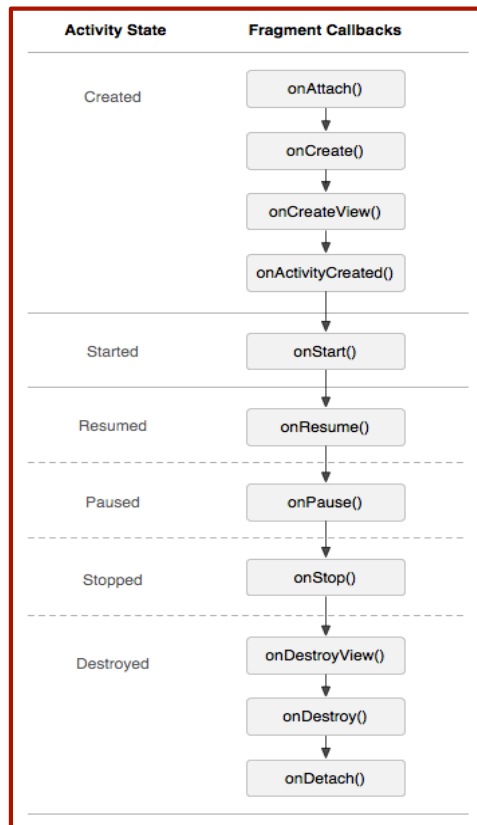
Android: **Fragment Creation**

onCreateView() → must return the **View** associated to the UI of the Fragment (if any) ...

```
public class ExampleFragment extends Fragment {  
  
    @Override  
    public View onCreateView(LayoutInflater inflater,  
        ViewGroup container, Bundle savedInstanceState) {  
  
        return  
inflater.inflate(R.layout.example_fragment,  
        container, false);  
  
    }  
}
```



Android: **Fragment Lifecycle**



The lifecycle of the Activity in which the Fragment lives directly affects the lifecycle of the Fragment.

onPause (Activity) → onPause (Fragment)

onStart (Activity) → onStart (Fragment)

onDestroy (Activity) → onDestroy (Fragment)

Fragments have also extra lifecycle callbacks to enable runtime creation/destroy.



Android: Fragment Lifecycle

ACTIVITY]: onCreate() ---
before setContentView
[FRAGMENT]: onAttach()
[FRAGMENT]: onCreate()
[FRAGMENT]: onCreateView()
[ACTIVITY]: onCreate() --- after
setContentView
[FRAGMENT]: onActivityCreated()
[ACTIVITY]: onStart()
[FRAGMENT]: onStart()
[ACTIVITY]: onResume()
[FRAGMENT]: onResume()

[FRAGMENT]: onPause()
[ACTIVITY]: onPause()
[FRAGMENT]: onStop()
[ACTIVITY]: onStop()
[FRAGMENT]: onDestroyView()
[FRAGMENT]: onDestroy()
[FRAGMENT]: onDetach()
[ACTIVITY]: onDestroy()



Android: Managing Fragments

A **Fragment** can get a reference to the Activity ...

```
Activity getActivity()
```

An **Activity** can get a reference to the Fragment ...

```
ExampleFragment fragment=(ExampleFragment)  
getFragmentManager().findFragmentById(R.id.example_f  
ragment)
```

The **FragmentManager** manages the Fragment associated to the current Activity.



Android: Managing Fragments

In some cases, a Fragment must share an event with the Activity ... how to do it?

1. Define a **callback** interface inside the Fragment

```
public interface OnArticleSelectedListener {  
    public void onArticleSelected(Uri uri);  
    ...  
}
```

2. Require that the host Activity implements it



Android: **Fragment Transactions**

- Fragments can be added/removed/replaced while the Activity is running ...
- Each set of changes to the Activity is called a **Transaction**.
- **Transaction** can be saved in order to allow a user to navigate backward among Fragments when he clicks on the “Back” button.



Android: **Fragment Transactions**

1. **ACQUIRE** an instance of the FRAGMENT MANAGER

```
FragmentManager man=getFragmentManager();  
FragmentTransaction transaction=man.beginTransaction();
```

2. **CREATE** new Fragment and Transaction

```
FragmentExample newFragment=new FragmentExample();  
transaction.replace(R.id.fragment_container, newFragment);
```

3. **SAVE** to backStack and **COMMIT**

```
transaction.addToBackStack(null);  
transaction.commit();
```



Android: **Fragment Transactions**

- A Transaction is not performed till the **commit** ...
- If **addToBackStack()** is not invoked → the Fragment is destroyed and it is not possible to navigate back.
- If **addToBackStack()** is invoked → the Fragment is stopped and it is possible to resume it when the user navigates back.
- **popBackStack()** → simulate a Back from the user.