

# Android Fragments Lifecycle and Specialization

Jim Wilson

[jimw@jwhh.com](mailto:jimw@jwhh.com)

@hedgehogjim

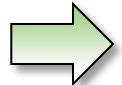


# Outline

- ➡ **Fragment lifecycle**
- ➡ **Types of Fragments**
- ➡ **ListFragment**
- ➡ **ListFragment customization**
- ➡ **WebViewFragment**
- ➡ **DialogFragment**
- ➡ **The Dialog within the DialogFragment**
- ➡ **Using DialogFragments as standard Fragments**

# Fragment & Activity setup/display lifecycle

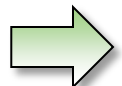
- **Activity lifecycle setup/display callbacks common to Fragment**



onCreate, onStart, onResume

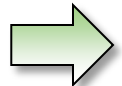
- Each Activity callback called before Fragment callback of the same name

- **Fragment-specific setup/display callbacks**



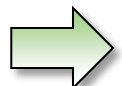
onAttach

- Receives a reference to the Activity which contains the Fragment



onCreateView

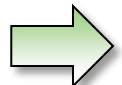
- Fragment should create its contents
- Must return a reference to a View



onActivityCreated

- Signals that the Activity which contains the Fragment has completed creation

- **Activity callbacks for Fragment setup**



onAttachFragment

- Called each time a Fragment is attached to the Activity

Sequence repeats  
for each Fragment

# Setup/display lifecycle sequence

## Activity

onCreate

onAttachFragment

onStart

onResume

## Fragment

onAttach

onCreate

onCreateView

onActivityCreated

onStart

onResume

1

2

3

4

5

6

7

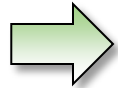
8

9

10

# Fragment and Activity teardown lifecycle

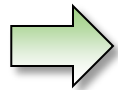
- **Activity lifecycle hide/teardown callbacks common to Fragment**



onPause, onStop, onDestroy

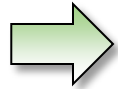
- Each Activity callback called after Fragment callback of the same name

- **Fragment-specific teardown callbacks**



onDestroyView

- The View returned by onCreateView is no longer attached to the Fragment



onDetach

- The Fragment is no longer attached to the View

# Types of Fragments

 **Android provides several Fragment-based classes**

 **Fragment class**

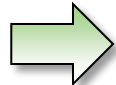
- You provide the entire UI

 **ListFragment class**

- Displays items in a list-like view

 **WebViewFragment**

- Displays a browser-like view

 **DialogFragment class**

- Displays a dialog window over the containing Activity

# ListFragment

## Displays a user-selectable list

### Automatically displays data contained in a ListAdapter

- Use SimpleAdapter for arrays and similar
- Use SimpleCursorAdapter for cursor-based data

### Specifying the data source

- Handle onActivityCreated
- Call setListAdapter

### If using default layout, no need to override onCreateView

- ListFragment provides a default view

```
public void onActivityCreated(Bundle savedInstanceState) {  
    super.onActivityCreated(savedInstanceState);  
  
    ArrayAdapter<String> dataAdapter =  
        new ArrayAdapter<String>( ... );  
  
    setListAdapter(dataAdapter);  
}
```

Introduction to Android  
Development

Android Programming with  
Intents

Android Async Programming  
and Services

Whats new in Android 4.0

# ListFragment row layout

## Specifying the layout of the list rows

### Define row layout using a layout resource

- Android provides several predefined row layout resources
- See list at <http://developer.android.com/reference/android/R.layout.html>

### Associate the layout resource with the list adaptor

- SimpleAdaptor & SimpleCursorAdaptor constructors that accept resource id
- Adaptor handles the details of applying the layout

```
String[] data = {"Apples", "Oranges", "Pears"};
```



# Customizing ListFragment appearance

## Customizing the layout of the ListFragment

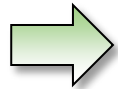
### Define fragment layout using a layout resource

- Include a ListView with an id of android:list
- Optionally include a TextView with an id of android:empty
  - TextView is displayed if list is empty

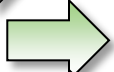
### Inflate the layout in onCreateView as in a regular Fragment-derived class

```
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <ListView android:id="@id/android:list"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#00FF00"/>
    <TextView android:id="@id/android:empty"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#FF0000"
        android:text="Empty List"/>
</LinearLayout>
```

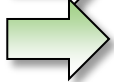
# WebViewFragment



**Provides a Fragment wrapper over a WebView**



Allows one to embed web-content within a Fragment



Has little functionality of its own

- Use `getWebView` method to access the contained `WebView`
- Populate in `onActivityCreated`

```
public class MyWebFragment extends WebViewFragment {  
    final String html =  
        "<html><body>" +  
        "<h2>This is a Title</h2>" +  
        "<p><strong>Hello</strong> <em>World</em></p>" +  
        "</body></html>";  
  
    public void onActivityCreated(Bundle savedInstanceState) {  
        super.onActivityCreated(savedInstanceState);  
    }  
}
```

# DialogFragment

➡ **DialogFragment makes dialog handling similar to other Fragments**

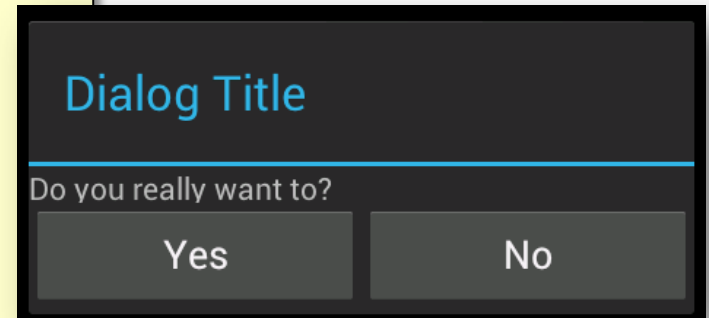
➡ Can specify the layout using a layout resource

- Override onCreateView as when directly inheriting from Fragment

➡ Use show method to display as a Dialog

- Will manage FragmentTransaction internally if passed FragmentManager
- Can be mixed as part of an application-managed FragmentTransaction
  - Must be last call in transaction because internally calls commit

```
<LinearLayout ... android:orientation="vertical" >
  <TextView ... android:layout_weight="1"
    android:text="Do you really want to?" />
  <LinearLayout ... android:orientation="horizontal"
    android:layout_weight="3">
    <Button ... android:id="@+id/btnYes"
      android:layout_weight="1"
      android:text="Yes" />
    <Button ... android:id="@+id/btnNo"
      android:layout_weight="1"
      android:text="No" />
  </LinearLayout>
</LinearLayout>
```



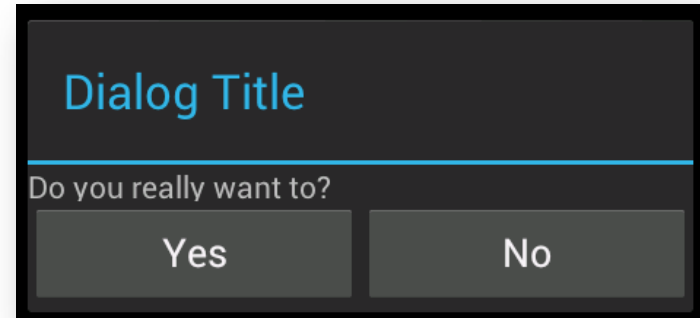
# DialogFragment appearance

➔ Use styles and themes to control appearance

➔ Must be set within onCreate callback

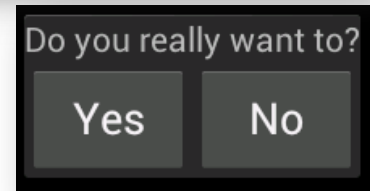
➔ Use `setStyle` to specify styles & themes

- ❑ Styles are incremental
- ❑ Themes come from standard theme resources

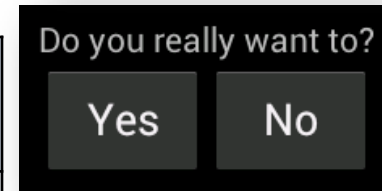


`android.R.style.Theme_Holo`

`android.R.style.Theme_Holo_Light_Dialog`



➔	STYLE_NORMAL	Dialog contains a title area and a default frame
➔	STYLE_NO_TITLE	Dialog contains no title area; has default frame
➔	STYLE_NO_FRAME	No title or frame; onCreateDialog is responsible to create entire dialog layout
➔	STYLE_NO_INPUT	Same as STYLE_NO_FRAME but Dialog



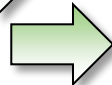
# There's a Dialog within

- ➡ **Framework automatically creates a Dialog object**
- ➡ Accessible with the `getDialog` method
  - Dialog created prior to `onCreateView` callback
- ➡ Most Dialog class behaviors available
  - Use Dialog `setTitle` method to set title text
  - Use `setCancelableOnTouchOutside(false)` to assure all input goes to the Dialog

```
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle state) {  
    View theView = inflater.inflate();  
  
    Dialog theDialog = getDialog();  
    if (theDialog != null) {  
        theDialog.setTitle("Dialog Title");  
        theDialog.setCanceledOnTouchOutside(false);  
    }  
    return theView;  
}
```

# Wrapping Dialogs in a DialogFragment

 Existing Dialog classes can be wrapped in DialogFragment

 Override onCreateDialog callback

- Return a reference to the Dialog class to wrap
- Called after onCreate and before onCreateView

 Most common use is to leverage the built-in AlertDialog class

```
public MyDialogFragment extends DialogFragment {  
1 public void onCreate( . . . ) { . . . }  
2 public Dialog onCreateDialog(Bundle state) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());  
    // set builder options  
  
    Dialog theDialog = builder.create();  
    theDialog.setCanceledOnTouchOutside(false);  
  
    return theDialog;  
}  
3 public View onCreateView( . . . ) { . . . }  
}
```

# When is a DialogFragment not a Dialog

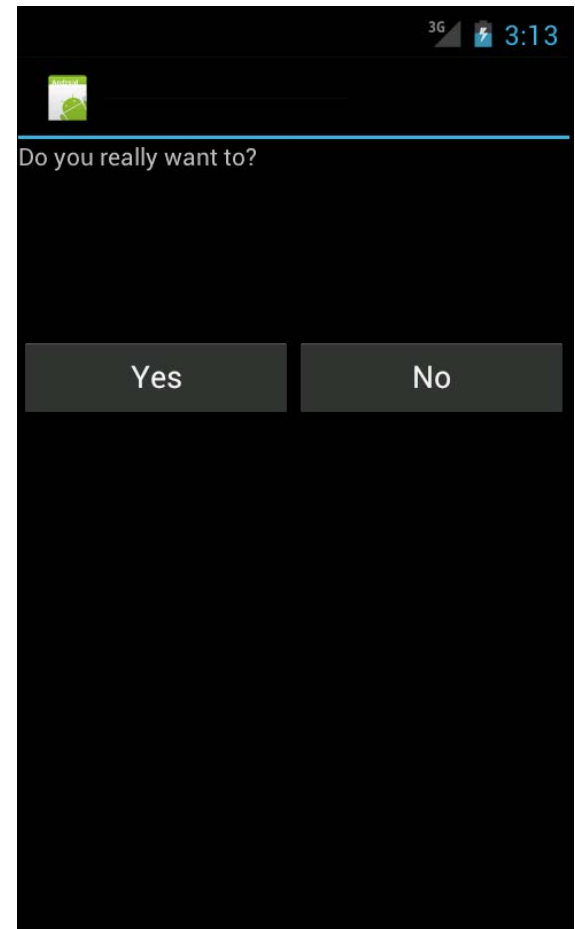
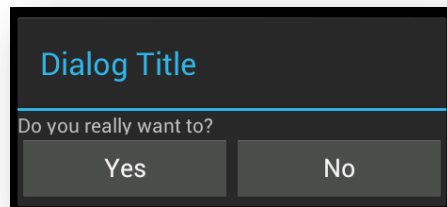
➡ **DialogFragments can be used as a regular Fragment**

➡ Does not have to be used as a Dialog

➡ Can use all standard Fragment behaviors

➡ Caveats

- ❑ Must use an onCreate View-based layout
- ❑ onCreateDialog is not called
- ❑ getDialog returns null



# Summary

- ➡ **Fragment & Activity lifecycle events are closely intertwined**
- ➡ **ListFragments simplify displaying lists of data**
  - Can optionally customize the layout by handling onCreateView
- ➡ **WebViewFragment allows web-based content within a Fragment**
- ➡ **DialogFragment provides a dialog behavior as a Fragment**
  - Allows more consistent programming model
- ➡ **DialogFragment automatically creates a Dialog around the Fragment**
  - Accessible with getDialog
  - Can create own Dialog by handling onCreateDialog
- ➡ **DialogFragments can be used as a standard Fragments**
  - Use FragmentTransaction add/replace rather than DialogFragment show
  - onCreateDialog call back not called and getDialog returns null