Adding Swipe-to-Refresh To Your App

The swipe-to-refresh user interface pattern is implemented entirely within the [SwipeRefreshLayout](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html)widget, which detects the vertical swipe, displays a distinctive progress bar, and triggers callback methods in your app. You enable this behavior by adding the widget to your layout file as the parent of a [ListView](file:///G:\\Studio\\SDK\\docs\\reference\\android\\widget\\ListView.html) or [GridView](file:///G:\\Studio\\SDK\\docs\\reference\\android\\widget\\GridView.html), and implementing the refresh behavior that gets invoked when the user swipes.

This lesson shows you how to add the widget to an existing layout. It also shows you how to add a refresh action to the action bar overflow area, so that users who may be unable to use the swipe gesture can trigger a manual update with an external device.

Add the SwipeRefreshLayout Widget

To add the swipe to refresh widget to an existing app, add [SwipeRefreshLayout](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html) as the parent of a single [ListView](file:///G:\\Studio\\SDK\\docs\\reference\\android\\widget\\ListView.html) or [GridView](file:///G:\\Studio\\SDK\\docs\\reference\\android\\widget\\GridView.html). Remember that [SwipeRefreshLayout](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html) only supports a single [ListView](file:///G:\\Studio\\SDK\\docs\\reference\\android\\widget\\ListView.html)or [GridView](file:///G:\\Studio\\SDK\\docs\\reference\\android\\widget\\GridView.html) child.

The following example demonstrates how to add the [SwipeRefreshLayout](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html) widget to an existing layout file containing a [ListView](file:///G:\\Studio\\SDK\\docs\\reference\\android\\widget\\ListView.html):

<android.support.v4.widget.SwipeRefreshLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    android:id="@+id/swiperefresh"  
    android:layout\_width="match\_parent"  
    android:layout\_height="match\_parent">  
  
    <ListView  
        android:id="@android:id/list"  
        android:layout\_width="match\_parent"  
        android:layout\_height="match\_parent" />  
  
</android.support.v4.widget.SwipeRefreshLayout>

You can also use the [SwipeRefreshLayout](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html) widget with a [ListFragment](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\app\\ListFragment.html). If the layout contains a[ListView](file:///G:\Studio\SDK\docs\reference\android\widget\ListView.html) with the ID "@android:id/list", the swipe-to-refresh functionality is automatically supported. However, explicitly declaring the [ListView](file:///G:\\Studio\\SDK\\docs\\reference\\android\\widget\\ListView.html) in this way supersedes the default[ListFragment](file:///G:\Studio\SDK\docs\reference\android\support\v4\app\ListFragment.html) view structure. If you want to use the default view structure, you will have to override parts of the [SwipeRefreshLayout](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html) and [ListFragment](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\app\\ListFragment.html) behavior. For an example of how to do this, see the [SwipeRefreshListFragment](file:///G:\\Studio\\SDK\\docs\\samples\\SwipeRefreshListFragment\\index.html) sample app.

Add a Refresh Action to the Action Bar

You should add a refresh action to your app's action bar to ensure that users who may not be able to perform a swipe gesture can still trigger a manual update. For example, users with accessibility issues can trigger action bar actions using external devices, such as keyboards and D-pads.

You should add the refresh action as a menu item, rather than as a button, by setting the attributeandroid:showAsAction=never. If you display the action as a button, users may assume that the refresh button action is different from the swipe-to-refresh action. By making the refresh action less conspicuous in the action bar, you can encourage users to perform manual updates with the swipe gesture while still maintaining the accessible option in a place where D-pad users would look for it.

The following code demonstrates how to add the swipe-to-refresh action to the overflow area:

<menu xmlns:android="http://schemas.android.com/apk/res/android" >  
    <item  
        android:id="@+id/menu\_refresh"  
        android:showAsAction="never"  
        android:title="@string/menu\_refresh"/>  
</menu>

Responding to a Refresh Request

Respond to the Refresh Gesture

When the user makes a swipe gesture, the system displays the progress indicator and calls your app's callback method. Your callback method is responsible for actually updating the app's data.

To respond to the refresh gesture in your app, implement the [SwipeRefreshLayout.OnRefreshListener](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.OnRefreshListener.html)interface and its [onRefresh()](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.OnRefreshListener.html" \l "onRefresh()) method. The [onRefresh()](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.OnRefreshListener.html" \l "onRefresh()) method is invoked when the user performs a swipe gesture.

You should put the code for the actual update operation in a separate method, and call that update method from your [onRefresh()](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.OnRefreshListener.html" \l "onRefresh()) implementation. That way, you can use the same update method to perform the update when the user triggers a refresh from the action bar.

Your update method calls [setRefreshing(false)](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html" \l "setRefreshing(boolean)) when it has finished updating the data. Calling this method instructs the [SwipeRefreshLayout](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html) to remove the progress indicator and update the view contents.

For example, the following code implements [onRefresh()](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.OnRefreshListener.html" \l "onRefresh()) and invokes the methodmyUpdateOperation() to update the data displayed by the [ListView](file:///G:\\Studio\\SDK\\docs\\reference\\android\\widget\\ListView.html):

/\*  
 \* Sets up a SwipeRefreshLayout.OnRefreshListener that is invoked when the user  
 \* performs a swipe-to-refresh gesture.  
 \*/  
mySwipeRefreshLayout.setOnRefreshListener(  
    new SwipeRefreshLayout.OnRefreshListener() {  
        @Override  
        public void onRefresh() {  
            Log.i(LOG\_TAG, "onRefresh called from SwipeRefreshLayout");  
  
            // This method performs the actual data-refresh operation.  
            // The method calls setRefreshing(false) when it's finished.  
            myUpdateOperation();  
        }  
    }  
);

Respond to the Refresh Action

If the user requests a refresh by using the action bar, the system calls the[onOptionsItemSelected()](file:///G:\Studio\SDK\docs\reference\android\support\v4\app\Fragment.html#onOptionsItemSelected(android.view.MenuItem)) method. Your app should respond to this call by displaying the progress indicator and refreshing the app's data.

To respond to the refresh action, override [onOptionsItemSelected()](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\app\\Fragment.html" \l "onOptionsItemSelected(android.view.MenuItem)). In your override method, trigger the [SwipeRefreshLayout](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html) progress indicator by calling [setRefreshing()](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html" \l "setRefreshing(boolean)) with the valuetrue, then perform the update operation. Once again, you should be doing the actual update in a separate method, so the same method can be called whether the user triggers the update with a swipe or by using the action bar. When the update has finished, call [setRefreshing(false)](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html" \l "setRefreshing(boolean)) to remove the refresh progress indicator.

The following code shows how to respond to the request action:

/\*  
 \* Listen for option item selections so that we receive a notification  
 \* when the user requests a refresh by selecting the refresh action bar item.  
 \*/  
@Override  
public boolean onOptionsItemSelected(MenuItem item) {  
    switch (item.getItemId()) {  
  
        // Check if user triggered a refresh:  
        case R.id.menu\_refresh:  
            Log.i(LOG\_TAG, "Refresh menu item selected");  
  
            // Signal SwipeRefreshLayout to start the progress indicator  
            mySwipeRefreshLayout.setRefreshing(true);  
  
            // Start the refresh background task.  
            // This method calls setRefreshing(false) when it's finished.  
            myUpdateOperation();  
  
            return true;  
    }  
  
    // User didn't trigger a refresh, let the superclass handle this action  
    return super.onOptionsItemSelected(item);  
  
}

**Note:** When the user triggers a refresh with a swipe action as described in [Respond to the Refresh Gesture](file:///G:\Studio\SDK\docs\training\swipe\respond-refresh-request.html#RespondRefresh), you do not need to call [setRefreshing()](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html" \l "setRefreshing(boolean)). The [SwipeRefreshLayout](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html) widget takes care of displaying the progress indicator and removing it when the update has finished. However, if the update is triggered by any means *other than* a swipe gesture, you need to explicitly turn the progress indicator on with [setRefreshing()](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html" \l "setRefreshing(boolean)). The method which actually refreshes the data calls [setRefreshing(false)](file:///G:\\Studio\\SDK\\docs\\reference\\android\\support\\v4\\widget\\SwipeRefreshLayout.html" \l "setRefreshing(boolean)) to signal that the update is finished.