# Quick Start Guide

**I'm a wrapper!**

The first thing to know about this library is that it is a wrapper around the existing View classes. I try not to extend and change any standard Android View code (where possible). So if you use this library and want to get access to the internal ListView/GridView/etc then simply callgetRefreshableView(). You can then call all of your usual methods such as setOnClickListener()etc.

**Layout**

The first thing to do is to modify your layout file to reference one of the PullToRefresh Views instead of an Android platform View (such as ListView), as so:

<!--

The PullToRefreshListView replaces a standard ListView widget.

The ID CAN NOT be @+id/android:list

-->

<com.handmark.pulltorefresh.library.PullToRefreshListView

android:id="@+id/pull\_to\_refresh\_listview"

android:layout\_height="fill\_parent"

android:layout\_width="fill\_parent" />

**Activity**

Now we can add the function so that the your application knows when a user has completed a 'PullToRefresh'.

// Set a listener to be invoked when the list should be refreshed.

PullToRefreshListView pullToRefreshView = (PullToRefreshListView) findViewById(R.id.pull\_to\_refresh\_listview);

pullToRefreshView.setOnRefreshListener(new OnRefreshListener<ListView>() {

@Override

public void onRefresh(PullToRefreshBase<ListView> refreshView) {

// Do work to refresh the list here.

new GetDataTask().execute();

}

});

private class GetDataTask extends AsyncTask<Void, Void, String[]> {

...

@Override

protected void onPostExecute(String[] result) {

// Call onRefreshComplete when the list has been refreshed.

pullToRefreshView.onRefreshComplete();

super.onPostExecute(result);

}

}

And that's it! I would now recommend having a look at the [Customisation](https://github.com/chrisbanes/Android-PullToRefresh/wiki/Customisation) page for details of how to change the way the View looks/behaves, as well as the sample code.