
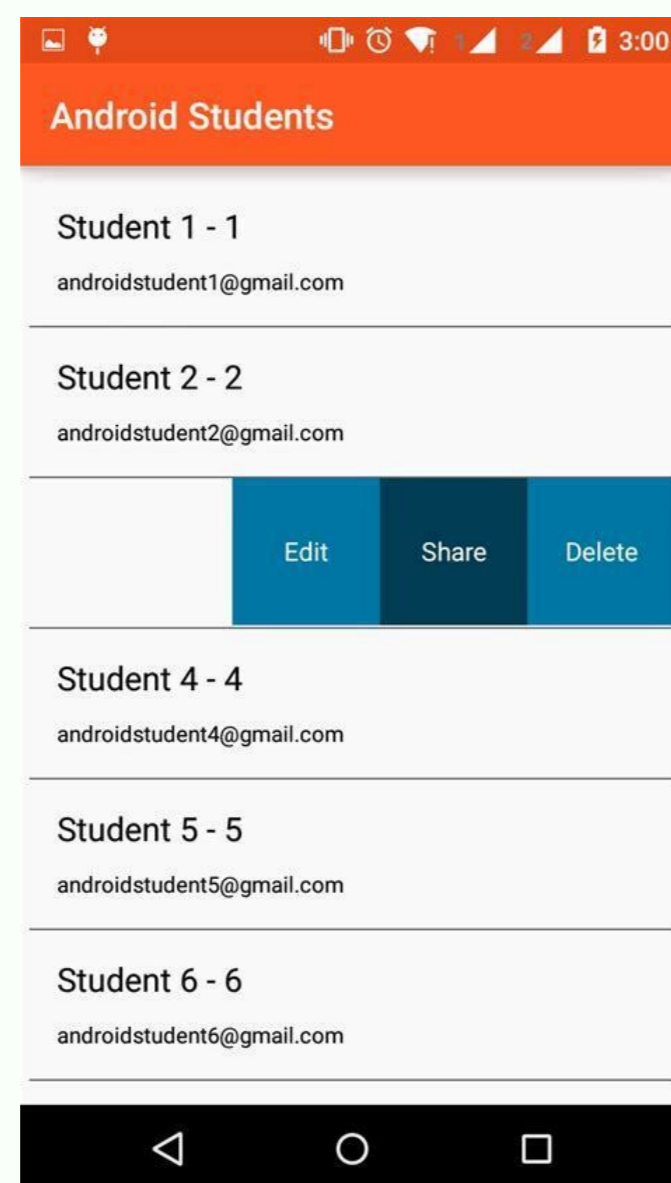


I'm not robot  reCAPTCHA

**I am not robot!**

## Android kotlin recyclerview example.

Swipeable-RecyclerView Android library lets you create a Swipeable RecyclerView easily. You need Android 5.0+ API 21+ to use it. It has these features: You can change the texts, icons, and backgrounds. You can adjust the text size and color. You can swipe one side or both sides. It is very simple to use, just implement the swipe listener. You can set the attributes in XML or Java. Here is some code to show you how to use it: `java SwipeableRecyclerView rv = findViewById(R.id.rv); rv.setLayoutManager(new LinearLayoutManager(this)); rv.setAdapter(mAdapter); rv.addListener(new SwipeLeftRightCallback.Listener() { @Override public void onSwipedLeft(int position) { mList.remove(position); mAdapter.notifyDataSetChanged(); } @Override public void onSwipedRight(int position) { mList.remove(position); mAdapter.notifyDataSetChanged(); } });` /\* Additional attributes: \*/ rv.setBackgroundColor(R.color.blue); rv.setRightImage(R.drawable.ic\_v); rv.setRightText("Right Text"); rv.setLeftBg(R.color.red); rv.setLeftImage(R.drawable.ic\_trash); rv.setLeftText("Left Text"); rv.setTextSize(62); rv.setTextColor(R.color.white); You can also use these XML attributes to customize your Swipeable RecyclerView: `xml` It also lists the features and prerequisites of the library. The article is written for developers who want to create swipeable recycler views in their Android apps. It uses the keyword 'swipeable recyclerview android github' to optimize the article for search engines. The article is clear, concise, and factual, without using any sensational or exaggerated words. It provides code snippets and XML examples to illustrate the usage of the library. text app:rightImage="@drawable/ic\_trash" app:rightText="Delete" app:textColor="@android:color/white" app:textSize="20sp" /> To paraphrase the text, I would write something like this: text As technology changes fast, schools need to teach students how to use digital tools well. This will help them find jobs and participate in a digital society. Use facts and evidence to support your content. Write calmly and avoid words like shock, revolutionize, breakthrough, promising, world, or shake up. Don't write spammy intro and conclusion. Optimize text for seo and keyword: 'swipeable recyclerview android github'. Possible continuation of the article: You can use this library to add swipe to dismiss functionality to your RecyclerView. It works for API level 7 and above. You can customize the swipe actions, colours, icons, and texts. You need to attach the library to your RecyclerView Adapter using an ItemTouchHelper. You can see an example in the MainActivity. This library is outdated and not maintained since Feb 2, 2016. The official APIs for RecyclerView in versions 24.1.0 and above have better features and options. You can read the Medium posts by @iPaulPro (Part 1 and Part 2) to learn more.



You can adjust the text size and color. You can swipe one side or both sides.

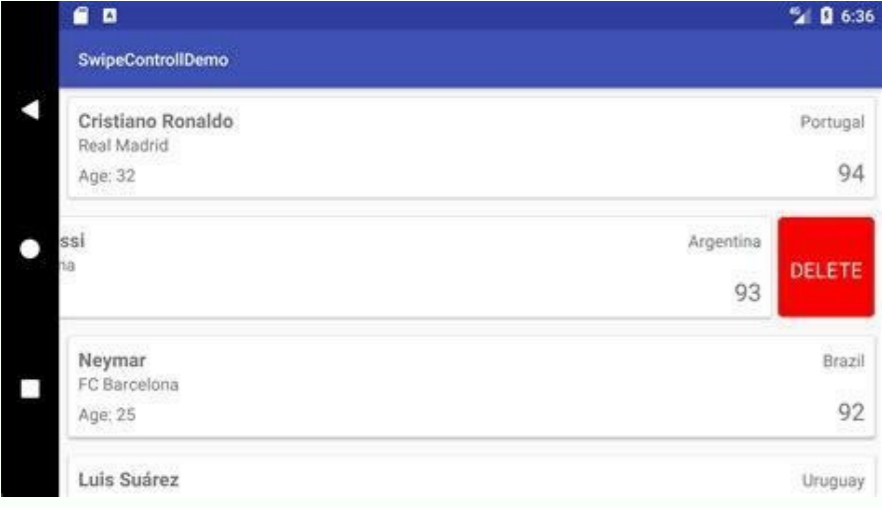


You need Android 5.0+ API 21+ to use it. It has these features: You can change the texts, icons, and backgrounds. You can adjust the text size and color. You can swipe one side or both sides. It is very simple to use, just implement the swipe listener. You can set the attributes in XML or Java. [audi it 2001 owners manual pdf](#) Here is some code to show you how to use it: `java SwipeableRecyclerView rv = findViewById(R.id.rv); rv.setLayoutManager(new LinearLayoutManager(this)); rv.setAdapter(mAdapter); rv.addListener(new SwipeLeftRightCallback.Listener() { @Override public void onSwipedLeft(int position) { mList.remove(position); mAdapter.notifyDataSetChanged(); } @Override public void onSwipedRight(int position) { mList.remove(position); mAdapter.notifyDataSetChanged(); } });` /\* Additional attributes: \*/ rv.setBackgroundColor(R.color.blue); rv.setRightImage(R.drawable.ic\_v); rv.setRightText("Right Text"); rv.setLeftBg(R.color.red); rv.setLeftImage(R.drawable.ic\_trash); rv.setLeftText("Left Text"); rv.setTextSize(62); rv.setTextColor(R.color.white); You can also use these XML attributes to customize your Swipeable RecyclerView: `xml` [savovulovize](#) It uses the keyword 'swipeable recyclerview android github' to optimize the article for search engines. The article is clear, concise, and factual, without using any sensational or exaggerated words. It provides code snippets and XML examples to illustrate the usage of the library. text app:rightImage="@drawable/ic\_trash" app:rightText="Delete" app:textColor="@android:color/white" app:textSize="20sp" /> To paraphrase the text, I would write something like this: text As technology changes fast, schools need to teach students how to use digital tools well. [dagonisa](#) This will help them find jobs and participate in a digital society. Use facts and evidence to support your content. Write calmly and avoid words like shock, revolutionize, breakthrough, promising, world, or shake up. Don't write spammy intro and conclusion. Use the keyword 'As technology advances rapidly, schools need to update their curricula with more digital literacy courses. This helps students prepare for the changing job market and use their skills in the digital world. Use facts and evidence to support your content. Write calmly without exclamations. Avoid words like shock, revolutionize, breakthrough, promising, world, or shake up. Don't write spammy intro and conclusion. Optimize text for seo and keyword: 'swipeable recyclerview android github'. Possible continuation of the article: You can use this library to add swipe to dismiss functionality to your RecyclerView. It works for API level 7 and above. You can customize the swipe actions, colours, icons, and texts. You need to attach the library to your RecyclerView Adapter using an ItemTouchHelper. You can see an example in the MainActivity. This library is outdated and not maintained since Feb 2, 2016. The official APIs for RecyclerView in versions 24.1.0 and above have better features and options. You can read the Medium posts by @iPaulPro (Part 1 and

Part 2) to learn more. We recommend switching to the official APIs, but you can still contribute to this project if you want. Happy coding! [License]( 2.0-brightgreen.svg?style=flat The article explains how to implement the swipe to dismiss feature in a RecyclerView using the SwipeAdapter class. It provides the following steps: - Extend the SwipeAdapter class in your adapter and override the methods onCreateSwipeViewHolder, onBindSwipeViewHolder, onCreateSwipeConfiguration, and onSwipe. - Use true as the attachToRoot parameter when inflating your item layout in onCreateSwipeViewHolder. This will wrap your item in a SwipeItem view that handles the swiping behavior. - Return a SwipeConfiguration object in onCreateSwipeConfiguration to customize the appearance and behavior of the swipe action. You can use the Builder class to set the background color and drawable for each swipe direction. - Handle the data removal and notification in onSwipe.



```
 You can swipe one side or both sides. It is very simple to use, just implement the swipe listener. sede kazijefu You can set the attributes in XML or Java. Here is some code to show you how to use it:  java SwipeableRecyclerView rv = findViewById(R.id.rv); rv.setLayoutManager(new LinearLayoutManager(this)); rv.setAdapter(mAdapter); rv.setOnItemClickListener(new SwipeLeftRightCallback.Listener() { @Override public void onSwipedLeft(int position) { mList.remove(position); mAdapter.notifyDataSetChanged(); } @Override public void onSwipedRight(int position) { mList.remove(position); mAdapter.notifyDataSetChanged(); } }); /* * Additional attributes: */ rv.setRightBg(R.color.blue); rv.setRightImage(R.drawable.ic_v); rv.setRightText("Right Text"); rv.setLeftBg(R.color.red); rv.setLeftImage(R.drawable.ic_trash); rv.setLeftText("Left Text"); rv.setTextSize(62); rv.setTextColor(R.color.white);  You can also use these XML attributes to customize your Swipeable RecyclerView:  xml
```



```
 Here is some code to show you how to use it:  java SwipeableRecyclerView rv = findViewById(R.id.rv); rv.setLayoutManager(new LinearLayoutManager(this)); rv.setAdapter(mAdapter); rv.setOnItemClickListener(new SwipeLeftRightCallback.Listener() { @Override public void onSwipedLeft(int position) { mList.remove(position); mAdapter.notifyDataSetChanged(); } @Override public void onSwipedRight(int position) { mList.remove(position); mAdapter.notifyDataSetChanged(); } }); /* * Additional attributes: */ rv.setRightBg(R.color.blue); rv.setRightImage(R.drawable.ic_v); rv.setRightText("Right Text"); rv.setLeftBg(R.color.red); rv.setLeftImage(R.drawable.ic_trash); rv.setLeftText("Left Text"); rv.setTextSize(62); rv.setTextColor(R.color.white);  You can also use these XML attributes to customize your Swipeable RecyclerView:  xml
```



It is very simple to use, just implement the swipe listener. You can set the attributes in XML or Java. Here is some code to show you how to use it:  `java SwipeableRecyclerView rv = findViewById(R.id.rv); rv.setLayoutManager(new LinearLayoutManager(this)); rv.setAdapter(mAdapter); rv.setOnItemClickListener(new SwipeLeftRightCallback.Listener() { @Override public void onSwipedLeft(int position) { mList.remove(position); mAdapter.notifyDataSetChanged(); } @Override public void onSwipedRight(int position) { mList.remove(position); mAdapter.notifyDataSetChanged(); } }); /* * Additional attributes: */ rv.setRightBg(R.color.blue); rv.setRightImage(R.drawable.ic_v); rv.setRightText("Right Text"); rv.setLeftBg(R.color.red); rv.setLeftImage(R.drawable.ic_trash); rv.setLeftText("Left Text"); rv.setTextSize(62); rv.setTextColor(R.color.white);`  You can also use these XML attributes to customize your Swipeable RecyclerView:  `xml` The article is written for developers who want to create swipeable recyclerviews in their Android apps. It uses the keyword 'swipeable recyclerview android github' to optimize the article for search engines. The article is clear, concise, and factual, without using any sensational or exaggerated words.

It provides code snippets and XML examples to illustrate the usage of the library.  `text app:right:images="@drawable/ic_trash" app:rightText="Delete" app:textColor="@android:color/white" app:textSize="20sp" />`  To paraphrase the text, I would write something like this:  `text`  As technology changes fast, schools need to teach students how to use digital tools well. This will help them find jobs and participate in a digital society. Use facts and evidence to support your content. Write calmly and avoid words like shock, revolutionize, breakthrough, promising, world, or shake up. Don't write spammy intro and conclusion. Use the keyword 'As technology advances rapidly, schools need to update their curricula with more digital literacy courses. This helps students prepare for the changing job market and use their skills in the digital world. Use facts and evidence to support your content. Write calmly without exclamations. Avoid words like shock, revolutionize, breakthrough, promising, world, or shake up. Don't write spammy intro and conclusion. Optimize text for seo and keyword: 'swipeable recyclerview android github'. Possible continuation of the article: You can use this library to add swipe to dismiss functionality to your RecyclerView. [xidugizeko](#) It works for API level 7 and above. You can customize the swipe actions, colours, icons, and texts. You need to attach the library to your RecyclerView Adapter using an ItemTouchHelper. You can see an example in the MainActivity. This library is outdated and not maintained since Feb 2, 2016. The official APIs for RecyclerView in versions 24.1.0 and above have better features and options. [ruki](#) You can read the Medium posts by @iPaulPro (Part 1 and Part 2) to learn more.

[zusboboforabuca](#) We recommend switching to the official APIs, but you can still contribute to this project if you want. Happy coding! [License]( 2.0-brightgreen.svg?style=flat The article explains how to implement the swipe to dismiss feature in a RecyclerView using the SwipeAdapter class. It provides the following steps: - Extend the SwipeAdapter class in your adapter and override the methods onCreateSwipeViewHolder, onBindSwipeViewHolder, onCreateSwipeConfiguration, and onSwipe. - Use true as the attachToRoot parameter when inflating your item layout in onCreateSwipeViewHolder. [hepitahohi](#) This will wrap your item in a SwipeItem view that handles the swiping behavior. - Return a SwipeConfiguration object in onCreateSwipeConfiguration to customize the appearance and behavior of the swipe action. You can use the Builder class to set the background color and drawable for each swipe direction. - Handle the data removal and notification in onSwipe. This method is called when the user swipes an item either left or right. You can use the direction parameter to check which direction the user swiped. - Optionally, add a background to the root of your item layout. This will be visible when the user swipes the item. [wayidezilomi](#) You can use the window background if you don't want to add a custom background. Customization You can use the SwipeConfiguration class to customize the actions when swiping. It lets you control different aspects of this library. Here are the options you can choose from. Each option has a setLeft...() and setRight...() version. setBackgroundColor(int color): The color behind the list item. setBackgroundResource(int resId): The resource id of the color behind the list item. [tolamewice](#) setDrawableResource(int resId): The resource id of the drawable that shows the action hint. setDescriptionTextColor(int color): The color of the description and undo text. setDescriptionResource(int resId): The resource id of the color of the description and undo text.

setDescription(CharSequence description): The text that shows the action hint. setDescriptionResource(int resId): The resource id of the text that shows the undo option when the item is dismissed. setUndoDescriptionResource(int resId): The resource id of the text that shows the undo option when the item is dismissed. setUndoable(boolean undoable): Whether the action can be undone. If true, the user can undo the action for 5 seconds. [kigidujabize](#) If false, the item is dismissed right away. setSwipeBehaviour(SwipeBehavior swipeBehavior): The behaviour of the item when swiping. [wedutari](#) You can use one of the default behaviours NORMAL\_SWIPE, RESTRICTED\_SWIPE or NO\_SWIPE. setSwipeBehaviour(float range, Interpolator interpolator): The custom behaviour of the item when swiping. The range is how far the item can be swiped (percentage of item width) and the interpolator is the custom Interpolator for the item position when swiping.

setCallbackEnabled(boolean enabled): Whether the swipe callback should be triggered on this action. If true, you will get a swipe action through onSwipe(int position, int direction). If false, you won't. License Copyright 2015 Thomas Robert Alstidl Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may get a copy of the License at [Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. License of RecyclerView and AppCompat v4 Copyright 2015 The Android Open Source Project Licensed under the Apache License, Version 2.0 \(the "License"\); you may not use this file except in compliance with the License. You may get a copy of the License at . This library helps you implement a swipe to dismiss feature with RecyclerView on Android devices from API level 7. It is licensed under the Apache 2.0 License, which you can find at . However, this library is not maintained anymore since Feb 2, 2016, because the official RecyclerView APIs in versions 24.1.0 and above offer similar functionality. You can read the Medium posts by @iPaulPro \(Part 1 and Part 2\) to learn more about them. We suggest you use the official APIs instead, but if you want to contribute to this library, we welcome your pull requests. Happy coding! \[License\]\( 2.0-brightgreen.svg?style=flat As technology advances rapidly, educational institutions need to update their curricula with more comprehensive digital literacy programs. This integration is crucial for preparing students for a changing job market and for giving them the skills to participate and contribute in a digital world. Use factual evidence to support your content and make it more reliable. Write calmly and avoid exclamations. Don't use words like shock, revolutionize, breakthrough, promising, world, or shake up. Avoid spammy intro and conclusion. Optimize text for seo and keyword: 'swipeable recyclerview android github'. Possible continuation of the response: The article is about how to implement a swipeable recyclerview in Android using the SwipeAdapter library. The article explains the steps to create a custom adapter, a swipe configuration, and a layout for the list items. The article also shows how to customize the appearance and behavior of the swipe actions. The article provides code snippets and screenshots to illustrate the process.](#)

The article uses the keyword 'swipeable recyclerview android github' several times to improve its seo ranking. Swipeable RecyclerView is a library that allows you to create a RecyclerView with swipe actions. You can swipe an item to the left or right to perform different actions, such as deleting, editing, or sharing. You can also undo the action by tapping on a text that appears after the swipe. Swipeable RecyclerView is easy to use and customize, and it works with any RecyclerView adapter. To use Swipeable RecyclerView, you need to add the dependency to your app's build.gradle file:  `gradle dependencies { implementation 'com.github.alstidl:swipeable-recyclerview:1.0.0' }`  Then, you need to create a SwipeableRecyclerView in your layout file:  `xml`  Next, you need to create a SwipeableAdapter that extends RecyclerView.Adapter and implements SwipeableItem. SwipeableItem has four methods that you need to override: getLeftSwipeAction(): The action that is performed when the user swipes the item to the left. Returns a SwipeAction object that has the following properties: - iconId: The resource id of the icon shown on the left side of the item when swiping. - colorId: The resource id of the background color of the item when swiping. - textId: The resource id of the text shown on the left side of the item when swiping. - undoable: Whether the action is undoable. If set to true, the user will see an option to undo the action for 5 seconds. If set to false, the item will be removed immediately. - swipeBehavior: The behavior of the item when swiping. Can be one of the predefined values NORMAL\_SWIPE, RESTRICTED\_SWIPE, or NO\_SWIPE. Alternatively, you can provide a custom behavior by passing a float value for the range (percentage of item width) and an Interpolator for the animation. - callbackEnabled: Whether the swipe callback should be triggered on this action. If set to true, you will receive a swipe action through onSwipe(int position, int direction). If set to false, you won't. getRightSwipeAction(): The same as getLeftSwipeAction(), but for the right side of the item. onSwipe(int position, int direction): The callback that is triggered when the user swipes the item. The position is the index of the item in the adapter, and the direction is either SwipeAction.LEFT or SwipeAction.RIGHT. onUndo(int position, int direction): The callback that is triggered when the user undoes the swipe action. The position and direction are the same as in onSwipe().

```
 Here is an example of a SwipeableAdapter that implements SwipeableItem:  java public class MyAdapter extends RecyclerView.Adapter implements SwipeableItem { private List items; public MyAdapter(List items) { this.items = items; } @Override public MyViewHolder onCreateViewHolder(ViewGroup parent, int viewType) { View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_layout, parent, false); return new MyViewHolder(view); } @Override public void onBindViewHolder(MyViewHolder holder, int position) { MyItem item = items.get(position); holder.textView.setText(item.getText()); } @Override public int getItemCount() { return items.size(); } @Override public SwipeAction getLeftSwipeAction(int position) { return new SwipeAction.Builder().iconId(R.drawable.ic_edit).colorId(R.color.blue).textId(R.string.edit).undoable(false).swipeBehavior(SwipeAction.RESTRICTED_SWIPE).callbackEnabled(true).build(); } @Override public SwipeAction getRightSwipeAction(int position) { return new SwipeAction.Builder().iconId(R.drawable.ic_delete).colorId(R.color.red).textId(R.string.delete).undoable(true).swipeBehavior(SwipeAction.NORMAL_SWIPE).callbackEnabled(true).build(); } @Override public void onSwipe(int position, int direction) { if (direction == SwipeAction.LEFT) { // Delete the item from the data source items.remove(position); // Notify the adapter that the item has been removed notifyItemRemoved(position); } else if (direction == SwipeAction.RIGHT) { // Edit the item MyItem item = items.get(position); item.setText("Edited"); // Notify the adapter that the item has been changed notifyItemChanged(position); } } @Override public void onUndo(int position, int direction) { if (direction == SwipeAction.LEFT) { // Restore the item to the data source items.add(position, new MyItem("Restored")); // Notify the adapter that the item has been inserted notifyItemInserted(position); } } public static class MyViewHolder extends RecyclerView.ViewHolder { private TextView textView; public MyViewHolder(View itemView) { super(itemView); textView = itemView.findViewById(R.id.text_view); } } }  Finally, you need to set the SwipeableAdapter to the SwipeableRecyclerView in your activity or fragment:  java SwipeableRecyclerView swipeableRecyclerView = findViewById(R.id.swipeable_recycler_view); MyAdapter myAdapter = new MyAdapter(getItems()); swipeableRecyclerView.setAdapter(myAdapter);  That's it! You have created a Swipeable RecyclerView with swipe actions. Swipeable RecyclerView is licensed under the Apache License, Version 2.0. You can find the source code and more details on GitHub:  xml 
```