Home

PUBLIC

Stack Overflow

Tags

Users

Jobs

How to update RecyclerView Adapter Data?

Ask Question

Trying to figure out what is the issue with updating `RecyclerView` 's Adapter.

After I get a new List of products, I tried to:

1. Update the `ArrayList` from the fragment where `recyclerView` is created, set new data to adapter, then call `adapter.notifyDataSetChanged()` ; it did not work.
2. Create a new adapter, as others did, and it worked for them, but no change for me: `recyclerView.setAdapter(new RecyclerViewAdapter(newArrayList))`
3. Create a method in `Adapter` which updates the data as follows:

   ```
   public void updateData(ArrayList<ViewModel> viewModels) {
      items.clear();
      items.addAll(viewModels);
      notifyDataSetChanged();
   }
   ```

   Then I call this method whenever I want to update the data list; it did not work.
4. To check if I can modify the recyclerView in any way, and I tried to remove at least an item:

   ```
    public void removeItem(int position) {
       items.remove(position);
       notifyItemRemoved(position);
   }
   ```

   Everything remained as it was.

Here is my Adapter:

```
public class RecyclerViewAdapter extends
RecyclerView.Adapter<RecyclerViewAdapter.ViewHolder> implements View.OnClickListener {

    private ArrayList<ViewModel> items;
    private OnItemClickListener onItemClickListener;

    public RecyclerViewAdapter(ArrayList<ViewModel> items) {
        this.items = items;
    }


    public void setOnItemClickListener(OnItemClickListener onItemClickListener) {
        this.onItemClickListener = onItemClickListener;
    }

    @Override
    public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        View v = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_recycler,
parent, false);
        v.setOnClickListener(this);
        return new ViewHolder(v);
    }

    public void updateData(ArrayList<ViewModel> viewModels) {
        items.clear();
        items.addAll(viewModels);
        notifyDataSetChanged();
    }
    public void addItem(int position, ViewModel viewModel) {
        items.add(position, viewModel);
        notifyItemInserted(position);
    }

    public void removeItem(int position) {
        items.remove(position);
        notifyItemRemoved(position);
    }



    @Override
    public void onBindViewHolder(ViewHolder holder, int position) {
        ViewModel item = items.get(position);
        holder.title.setText(item.getTitle());
        Picasso.with(holder.image.getContext()).load(item.getImage()).into(holder.image);
        holder.price.setText(item.getPrice());
        holder.credit.setText(item.getCredit());
        holder.description.setText(item.getDescription());
```

```
    @Override
    public int getItemCount() {
        return items.size();
    }

    @Override
    public void onClick(final View v) {
        // Give some time to the ripple to finish the effect
        if (onItemClickListener != null) {
            new Handler().postDelayed(new Runnable() {
                @Override
                public void run() {
                    onItemClickListener.onItemClick(v, (ViewModel) v.getTag());
                }
            }, 0);
        }
    }

    protected static class ViewHolder extends RecyclerView.ViewHolder {
        public ImageView image;
        public TextView price, credit, title, description;

        public ViewHolder(View itemView) {
            super(itemView);
            image = (ImageView) itemView.findViewById(R.id.image);
            price = (TextView) itemView.findViewById(R.id.price);
            credit = (TextView) itemView.findViewById(R.id.credit);
            title = (TextView) itemView.findViewById(R.id.title);
            description = (TextView) itemView.findViewById(R.id.description);
        }
    }

    public interface OnItemClickListener {

        void onItemClick(View view, ViewModel viewModel);

    }
}
```

And I initiate `RecyclerView` as follows:

```
recyclerView = (RecyclerView) view.findViewById(R.id.recycler);
recyclerView.setLayoutManager(new GridLayoutManager(getActivity(), 5));
adapter = new RecyclerViewAdapter(items);
adapter.setOnItemClickListener(this);
recyclerView.setAdapter(adapter);
```

So, how do I actually update adapter data in order to display newly received items?

Update: the issue was that the layout where gridView was looked as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:tag="catalog_fragment"
    android:layout_height="match_parent">

    <FrameLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <android.support.v7.widget.RecyclerView
            android:id="@+id/recycler"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:clipToPadding="false"/>

        <ImageButton
            android:id="@+id/fab"
            android:layout_gravity="top|end"
            style="@style/FabStyle"/>

    </FrameLayout>
</LinearLayout>
```

Then I just removed `LinearLayout` and made `FrameLayout` as parent layout.

android android-recyclerview

edited Sep 27 at 17:39

`items.clear(); items.addAll(newItems);` is an ugly pattern. If you really need defensive copying here, `items = new ArrayList(newItems);` would be less ugly. – Miha_x64 Oct 18 at 9:37

12 Answers

I'm working with RecyclerView and both the remove and the update work well.

1) REMOVE: There are 4 steps to remove an item from a RecyclerView

```
list.remove(position);
recycler.removeViewAt(position);
mAdapter.notifyItemRemoved(position);
mAdapter.notifyItemRangeChanged(position, list.size());
```

These line of codes work for me.

2) UPDATE THE DATA: The only things I had to do is

```
mAdapter.notifyDataSetChanged();
```

You had to do all of this in the Actvity/Fragment code not in the RecyclerView Adapter code.

edited Mar 23 '16 at 14:18
CaptJak
3,101 1 22 42

answered Jul 12 '15 at 13:39
nicopasso
3,580 2 18 28

2 Thank you. Please check the update. Somehow, linear layout does not let the RecylerView update list. – Filip Luchianenco Jul 12 '15 at 14:44

26 you only need these two lines: list.remove(position); mAdapter.notifyItemRemoved(position); – feisal Mar 10 '16 at 20:00

1 For me, the lines `recycler.removeViewAt(position);` (or rather `recycler.removeAllViews()`) were the crucial ones. – Mike Speed Mar 16 '16 at 11:20

2 An important consideration to avoid annoying glitch during the delete: You don't need to call this line recycler.removeViewAt(position); – Angelo Nodari Jul 15 '16 at 8:54

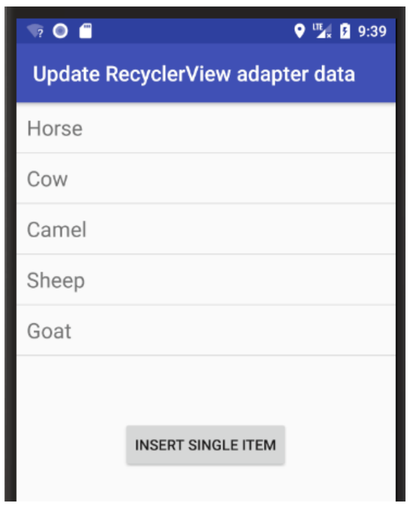
2 NotifyDataSetChanged is overkill, specially if you are dealing with VERY long lists. Reload the whole thing for a few items? No, thanks. Also, recycler.removeViewAt(position)? Not needed. And don't need to call both "mAdapter.notifyItemRemoved(position)" and "mAdapter.notifyItemRangeChanged(position, list.size())". Remove some, or remove one. Notify ONCE. Call one of them. – Vitor Hugo Schwaab Dec 11 '16 at 17:41

This is a general answer for future visitors. The various ways to update the adapter data are explained. The process includes two main steps every time:

1. Update the data set
2. Notify the adapter of the change

Insert single item

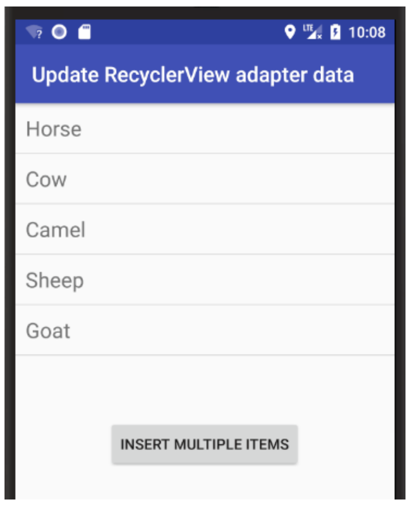
Add "Pig" at index `2`.



```
String item = "Pig";
int insertIndex = 2;
data.add(insertIndex, item);
adapter.notifyItemInserted(insertIndex);
```

Insert multiple items

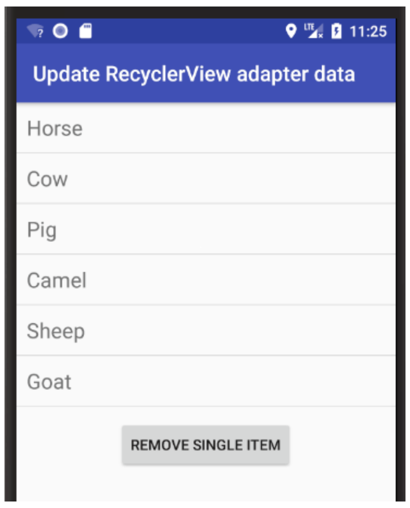
Insert three more animals at index `2`.



```
ArrayList<String> items = new ArrayList<>();
items.add("Pig");
items.add("Chicken");
items.add("Dog");
int insertIndex = 2;
data.addAll(insertIndex, items);
adapter.notifyItemRangeInserted(insertIndex, items.size());
```

Remove single item

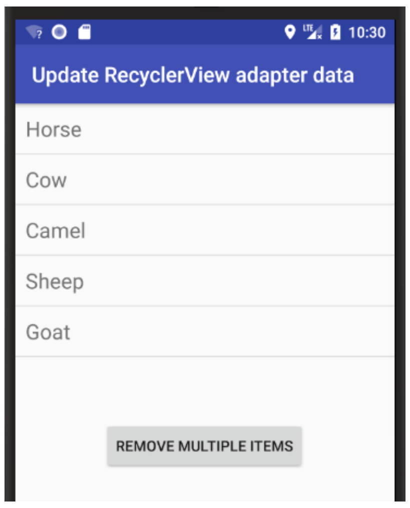
Remove "Pig" from the list.



```
int removeIndex = 2;
data.remove(removeIndex);
adapter.notifyItemRemoved(removeIndex);
```

Remove multiple items

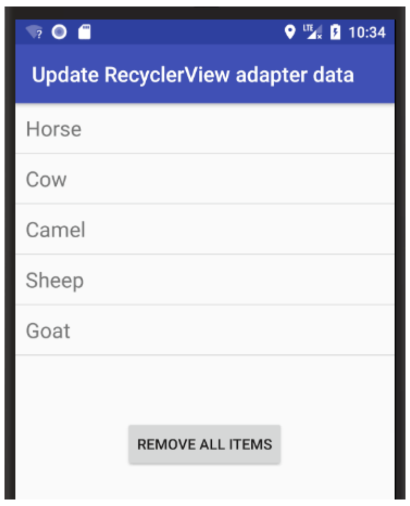
Remove "Camel" and "Sheep" from the list.



```
int startIndex = 2; // inclusive
int endIndex = 4;   // exclusive
int count = endIndex - startIndex; // 2 items will be removed
data.subList(startIndex, endIndex).clear();
adapter.notifyItemRangeRemoved(startIndex, count);
```

Remove all items

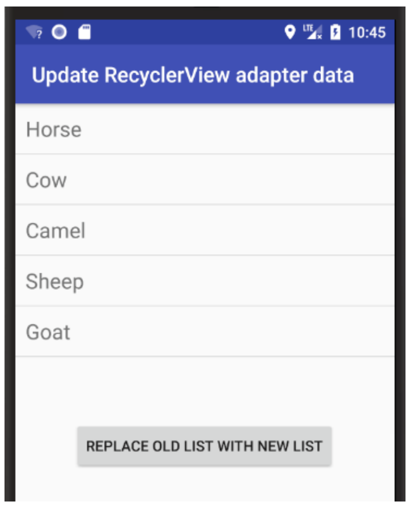
Clear the whole list.



```
data.clear();
adapter.notifyDataSetChanged();
```

Replace old list with new list

Clear the old list then add a new one.

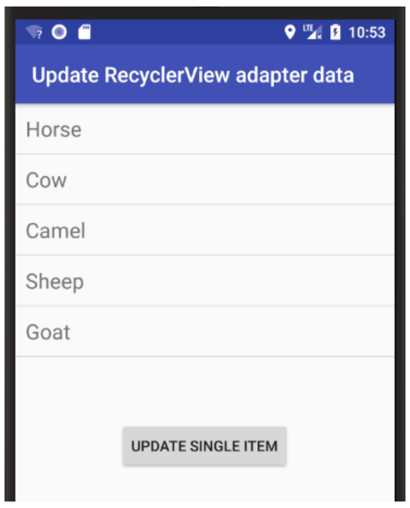

10:45
Update RecyclerView adapter data
Horse
Cow
Camel
Sheep
Goat
REPLACE OLD LIST WITH NEW LIST

```
// clear old list
data.clear();

// add new list
ArrayList<String> newList = new ArrayList<>();
newList.add("Lion");
newList.add("Wolf");
newList.add("Bear");
data.addAll(newList);

// notify adapter
adapter.notifyDataSetChanged();
```

The `adapter` has a reference to `data`, so it is important that I didn't set `data` to a new object. Instead I cleared the old items from `data` and then added the new ones.

Update single item

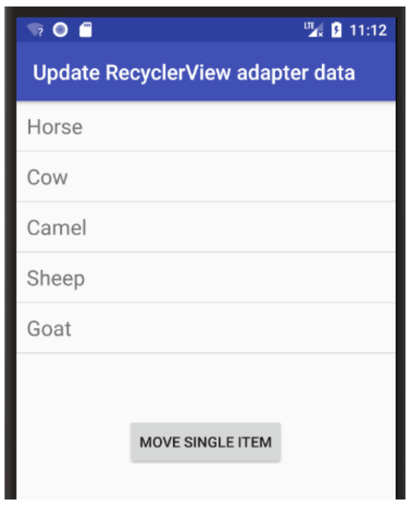
Change the "Sheep" item so that it says "I like sheep."



```
String newValue = "I like sheep.";
int updateIndex = 3;
data.set(updateIndex, newValue);
adapter.notifyItemChanged(updateIndex);
```

Move single item

Move "Sheep" from position `3` to position `1`.



```
int fromPosition = 3;
int toPosition = 1;

// update data array
String item = data.get(fromPosition);
data.remove(fromPosition);
data.add(toPosition, item);

// notify adapter
adapter.notifyItemMoved(fromPosition, toPosition);
```

Code

Here is the project code for your reference. The RecyclerView Adapter code can be found at this answer.

MainActivity.java

```
public class MainActivity extends AppCompatActivity implements
MyRecyclerViewAdapter.ItemClickListener {

    List<String> data;
    MyRecyclerViewAdapter adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // data to populate the RecyclerView with
        data = new ArrayList<>();
        data.add("Horse");
        data.add("Cow");
        data.add("Camel");
        data.add("Sheep");
        data.add("Goat");

        // set up the RecyclerView
        RecyclerView recyclerView = findViewById(R.id.rvAnimals);
        LinearLayoutManager layoutManager = new LinearLayoutManager(this);
        recyclerView.setLayoutManager(layoutManager);
        DividerItemDecoration dividerItemDecoration = new
DividerItemDecoration(recyclerView.getContext(),
                layoutManager.getOrientation());
        recyclerView.addItemDecoration(dividerItemDecoration);
        adapter = new MyRecyclerViewAdapter(this, data);
        adapter.setClickListener(this);
        recyclerView.setAdapter(adapter);
    public void onItemClick(View view, int position) {
        Toast.makeText(this, "You clicked " + adapter.getItem(position) +
" + position, Toast.LENGTH_SHORT).show();
    }

    public void onButtonClick(View view) {
        insertSingleItem();
    }

    private void insertSingleItem() {
        String item = "Pig";
        int insertIndex = 2;
        data.add(insertIndex, item);
        adapter.notifyItemInserted(insertIndex);
    }

    private void insertMultipleItems() {
        ArrayList<String> items = new ArrayList<>();
        items.add("Pig");
        items.add("Chicken");
        items.add("Dog");
        int insertIndex = 2;
        data.addAll(insertIndex, items);
        adapter.notifyItemRangeInserted(insertIndex, items.size());
    }

    private void removeSingleItem() {
        int removeIndex = 2;
        data.remove(removeIndex);
        adapter.notifyItemRemoved(removeIndex);
    }

    private void removeMultipleItems() {
        int startIndex = 2; // inclusive
        int endIndex = 4;   // exclusive
        int count = endIndex - startIndex; // 2 items will be removed
        data.subList(startIndex, endIndex).clear();
        adapter.notifyItemRangeRemoved(startIndex, count);
    }

    private void removeAllItems() {
        data.clear();
        adapter.notifyDataSetChanged();
    }

    private void replaceOldListWithNewList() {
        // clear old list
        data.clear();

        // add new list
        ArrayList<String> newList = new ArrayList<>();
        newList.add("Lion");
        newList.add("Wolf");
        newList.add("Bear");
        data.addAll(newList);

        // notify adapter
        adapter.notifyDataSetChanged();
    }

    private void updateSingleItem() {
        String newValue = "I like sheep.";
        int updateIndex = 3;
        data.set(updateIndex, newValue);
        adapter.notifyItemChanged(updateIndex);
    }

    private void moveSingleItem() {
        int fromPosition = 3;
        int toPosition = 1;

        // update data array
        String item = data.get(fromPosition);
        data.remove(fromPosition);
        data.add(toPosition, item);

        // notify adapter
        adapter.notifyItemMoved(fromPosition, toPosition);
    }
}
```

Notes

- If you use `notifyDataSetChanged()` , then no animation will be performed. This can also be an expensive operation, so it is not recommended to use `notifyDataSetChanged()` if you are only updating a single item or a range of items.
- Check out DiffUtil if you are making large or complex changes to a list.

Further study

- CodePath: Using the RecyclerView

answered Feb 24 at 3:50

Suragch
190k 113 656 719

14 Man, your answer is amazing. – Mauker Mar 8 at 4:59

2 *answerS ... he is writing many new and very clear answers to older topics. Thanks! – hamena314 May 6 at 17:42

1 By far the best answer to solve common issues revolving RecyclerView. +1 – Zhi Kai Jun 16 at 10:03

Awesome answer! What would be the steps when replacing a in item, which would result in a different view type? Is it just notifyItemChanged or a combined remove followed by an insert? – Semaphor Sep 25 at 7:11

With your minimal example, it looks like notifyItemChanged is sufficient. It will call onCreateViewHolder and onBindViewHolder for the changed item and the different view is shown. I had a problem in an application updating an item which would result in a different view, but there seems to be another problem. – Semaphor Oct 1 at 14:43

This is what worked for me:

```
recyclerView.setAdapter(new RecyclerViewAdapter(newList));
recyclerView.invalidate();
```

After creating a new adapter that contains the updated list (in my case it was a database converted into an ArrayList) and setting that as adapter, I tried `recyclerView.invalidate()` and it worked.

answered Jan 22 '16 at 14:02

martinpkmn
550 5 7

10 wouldn't this refresh the entire view instead of just updating the items that have changed? – TWilly Feb 25 '16 at 18:44

11 Instead of making a new adapter each time, what I did was to create a setter method in my custom adapter class to set the new list. After that, just call `YourAdapter adapter = (YourAdapter) recyclerView.getAdapter(); adapter.yourSetterMethod(newList); adapter.notifyDataSetChanged();` That being said, it sounds like what the OP tried first (just adding this on as a comment so that it may help someone else, as that worked in my case). – Kevin Lee Apr 15 '16 at 11:50

2 Reader, don't settle for this. This answer works, but there's a more efficient way. Instead of reloading *all data again*, the most efficient way is just add the new data to the adapter. – Sebastialonso Aug 15 '16 at 3:30

Yes I agree with @TWilly it will redraw complete View on UI. – Rahul Aug 24 '17 at 17:47

you have 2 options to do this: refresh UI from the adapter:

```
mAdapter.notifyDataSetChanged();
```

or refresh it from recyclerView itself:

```
recyclerView.invalidate();
```

answered Aug 1 '16 at 13:08

ediBersh
634 7 14

Another option is to use diffutil . It will compare the original list against the new list and use the new list as the update if there is a change.

Basically, we can use DiffUtil to compare the old data vs new data and let it call notifyItemRangeRemoved, and notifyItemRangeChanged and notifyItemRangeInserted on your behalf.

A quick example of using diffUtil instead of notifyDataSetChanged:

```
DiffResult diffResult = DiffUtil
```

```
diffResult.dispatchUpdatesTo(this);

//and then, if necessary
items.clear()
items.addAll(newItems)
```

I do the calculateDiff work off the main thread in case it's a big list.

edited 2 days ago
MatPag
11.8k 6 36 51

answered Dec 5 '16 at 3:29
j2emanue
20.9k 22 135 244

1 although this is correct, how do update the data inside your adapter? Let's say I am displaying List<Object> in the adapter and I get a new update with List<Object>. DiffUtil will calculate a difference and dispatches it in the adapter, but it is doing nothing with the original or new list (in your case `getItems()` . How do you know which data from original list need to be removed/added/updated? – vanomart Jan 12 '17 at 10:17

1 Maybe this article can help. .. medium.com/@nullthemall/… – j2emanue Jan 12 '17 at 13:25

@vanomart you only have to directly replace your local list field with the new list value as you would normally do, the only difference in this approach is that instead of notifyDataSetChanged() you use DiffUtil to update the view. – carrizo Mar 6 '17 at 3:24

Update Data of listview, gridview and recyclerview

```
mAdapter.notifyDataSetChanged();
```

or

```
mAdapter.notifyItemRangeChanged(0, itemList.size());
```

answered Jan 20 '17 at 12:21
Pankaj Talaviya
1,218 10 22

This doesn't seem to update my data until the user starts scrolling. I've looked all over the place and can't figure why.. – SudoPlz Mar 19 at 16:35

@SudoPlz you can use custom scroll listener for detect scroll and perform your notify operation like stackoverflow.com/a/31174967/4401692 – Pankaj Talaviya Mar 20 at 4:21

Thanks Pankaj but that's exactly what I'm trying to avoid. I want to update everything WITHOUT having the user touch the screen. – SudoPlz Mar 20 at 10:18

I've solved the same problem in a different way. I don't have data I waiting for it from the background thread so start with an emty list.

```
        mAdapter = new ModelAdapter(getContext(),new ArrayList<Model>());
    // then when i get data

        mAdapter.update(response.getModelList());
  //  and update is in my adapter

        public void update(ArrayList<Model> modelList){
            adapterModelList.clear();
            for (Product model: modelList) {
                adapterModelList.add(model);
            }
           mAdapter.notifyDataSetChanged();
        }
```

That's it.

edited Oct 7 '17 at 14:12
user8690908

answered Oct 4 '16 at 12:09
Ahmed Ozmaan

The best and the coolest way to add new data to the present data is

```
ArrayList<String> newItems = new ArrayList<String>();
newItems = getList();
int oldListItemscount = alcontainerDetails.size();
alcontainerDetails.addAll(newItems);
recyclerview.getAdapter().notifyItemChanged(oldListItemscount+1, al_conta
```

edited Jul 30 '17 at 4:08
CoDe
5,627 11 54 130

answered Jun 9 '16 at 11:34
Rushi Ayyappa
1,954 1 10 21

1 I see people using the "notifyDataSetChanged" for EVERYthing, isn't that overkill? I mean, reload the whole list because a few changed or were added... I like this approach, was implementing right now, with the "notifyItemRangeInserted" method. – Vitor Hugo Schwaab Dec 11 '16 at 17:37

I found out that a really simple way to reload the RecyclerView is to just call

```
recyclerView.removeAllViews();
```

This will first remove all content of the RecyclerView and then add it again with the updated values.

answered Feb 28 at 20:30
Jonathan Persgarden
61 2

i got the answer after a long time

```
  SELECTEDROW.add(dt);
                notifyItemInserted(position);
                SELECTEDROW.remove(position);
                notifyItemRemoved(position);
```

answered Jul 12 '17 at 13:17
roufal
11 1

If nothing mentioned in the above comments is working for you. It might mean the problem lies somewhere else.

One place I found the solution was in the way I was setting the list to the adapter. In my activity the list was a instance variable and I was changing it directly when any data changed. Due to it being a reference variable there was something weird going on. So I changed the reference variable to a local one and used another variable to update data and then pass to `addAll()` function mentioned in above answers.

answered Nov 1 '17 at 21:26
bitSmith
13 3

These methods are efficient and good to start using a basic `RecyclerView`.

```
private List<YourItem> items;

public void setItems(List<YourItem> newItems)
{
    clearItems();
    addItems(newItems);
}

public void addItem(YourItem item, int position)
```

```
    items.add(item);
    notifyItemInserted(position);
}

public void addMoreItems(List<YourItem> newItems)
{
    int position = items.size() + 1;
    newItems.addAll(newItems);
    notifyItemChanged(position, newItems);
}

public void addItems(List<YourItem> newItems)
{
    items.addAll(newItems);
    notifyDataSetChanged();
}

public void clearItems()
{
    items.clear();
    notifyDataSetChanged();
}

public void addLoader()
{
    items.add(null);
    notifyItemInserted(items.size() - 1);
}

public void removeLoader()
{
    items.remove(items.size() - 1);
    notifyItemRemoved(items.size());
}

public void removeItem(int position)
{
    if (position >= items.size()) return;

    items.remove(position);
    notifyItemRemoved(position);
}

public void swapItems(int positionA, int positionB)
{
    if (positionA > items.size()) return;
    if (positionB > items.size()) return;

    YourItem firstItem = items.get(positionA);

    videoList.set(positionA, items.get(positionB));
    videoList.set(positionB, firstItem);

    notifyDataSetChanged();
}
```

You can implement them inside of an Adapter Class or in your Fragment or Activity but in that case you have to instantiate the Adapter to call the notification methods. In my case I usually implement it in the Adapter.

answered Feb 13 at 8:59

Teocci
1,471 1 11 21

protected by Hovercraft Full Of Eels Nov 3 '17 at 19:34

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?