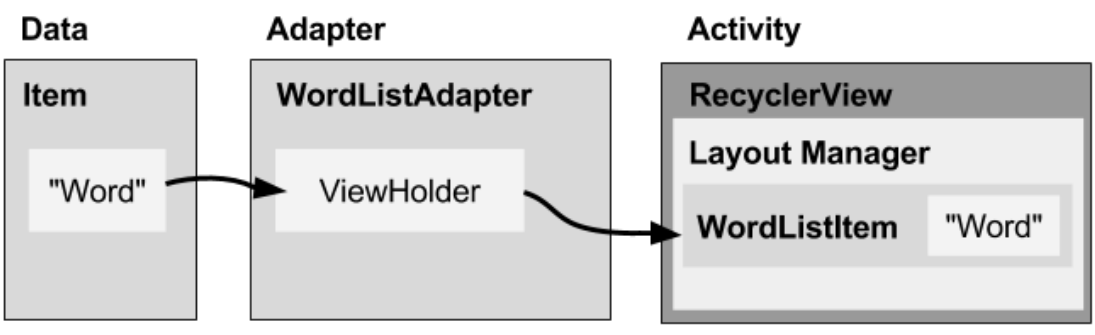
Task 2: Create a RecyclerView

In this practical, you **display data** in a **RecyclerView**. You need the following:

- **Data to display**: Use the **mWordList**.
- **A RecyclerView** for the scrolling list that contains the **list items**.
- **Layout for one item of data**. All list items look the **same**.
- **A layout manager**. **RecyclerView.LayoutManager** handles the hierarchy and layout of **View** elements.
  - **RecyclerView** requires an **explicit layout manager** to manage the arrangement of list items contained within it.
  - This layout could be **vertical, horizontal**, or a grid.
  - You will use **a vertical LinearLayoutManager**.
- **An adapter**.
  - **RecyclerView.Adapter connects your data to the RecyclerView**.
  - It **prepares the data** in a **RecyclerView.ViewHolder**.
  - You will create an adapter that inserts into and updates your generated words in your views.
- **A ViewHolder**.
  - **Inside your adapter**, you will **create a ViewHolder** that contains the **View information** for **displaying one item** from the item's layout.

- The diagram below shows the relationship between the **data**, the **adapter**, the **ViewHolder**, and the **layout manager**.



To implement these pieces, you will need to :

1. **Add** a **RecyclerView** **element** to the **MainActivity** **XML content layout** (**content_main.xml**) for the RecyclerView app.
2. **Create** an XML layout file (**wordlist_item.xml**) for **one list item**, which is **WordListItem**.
3. **Create an adapter** (**WordListAdapter**) with a **ViewHolder** (**WordViewHolder**). **Implement** the **method that takes the data**, places it in the **ViewHolder**, and **lets the layout manager know to display it**.
4. In the **onCreate() method** of **MainActivity**, **create** a **RecyclerView** and **initialize it with the adapter and a standard layout manager**.

Let's do these one at a time.

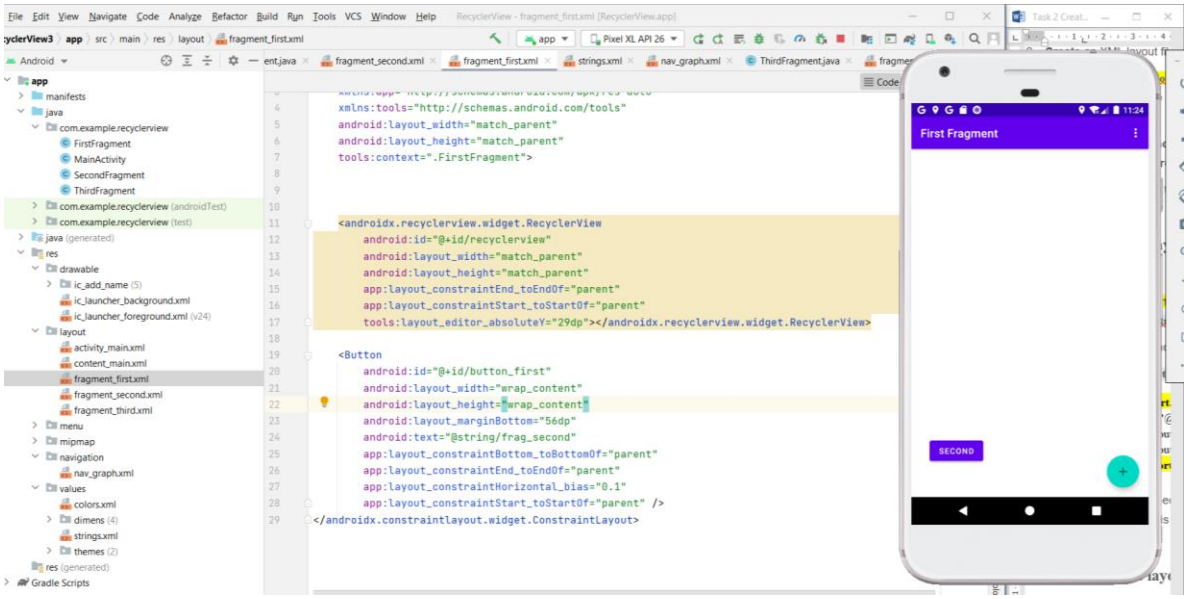
2**.1. Modify the layout in content_main.xml**

To **add a RecyclerView element to the XML layout**, follow these steps:

1. Open **content_main.xml** in your **RecyclerView app**. It shows a "Hello World" **TextView** at the center of a ConstraintLayout.
2. Click the **Text** tab to show the **XML code**.
3. **Replace** the entire **TextView** element with the following:

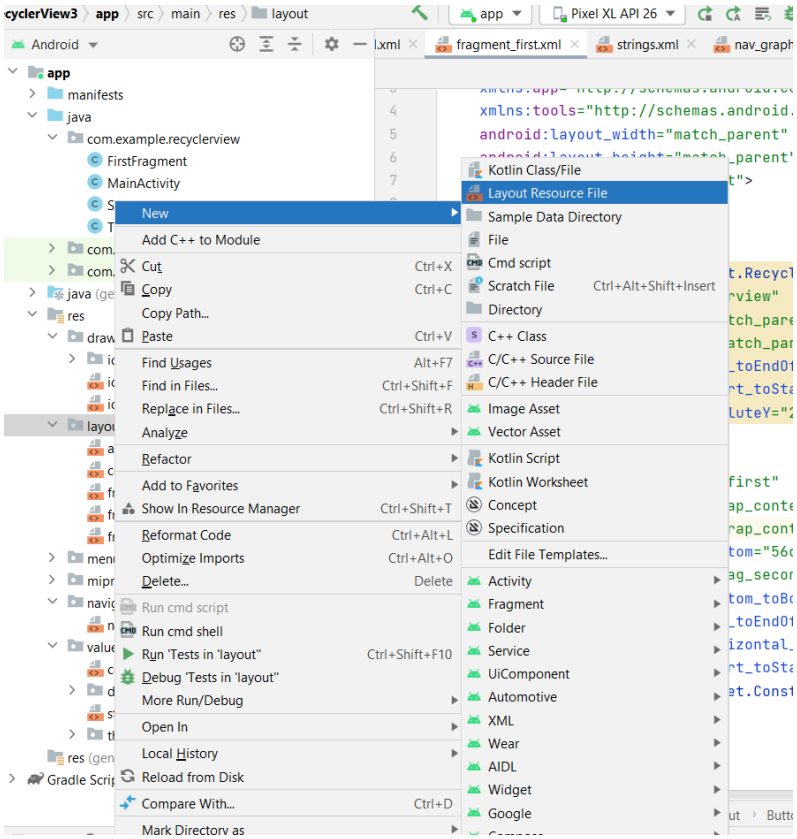
```
<android.support.v7.widget.RecyclerView
    android:id="@+id/recyclerview"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
</android.support.v7.widget.RecyclerView>
```

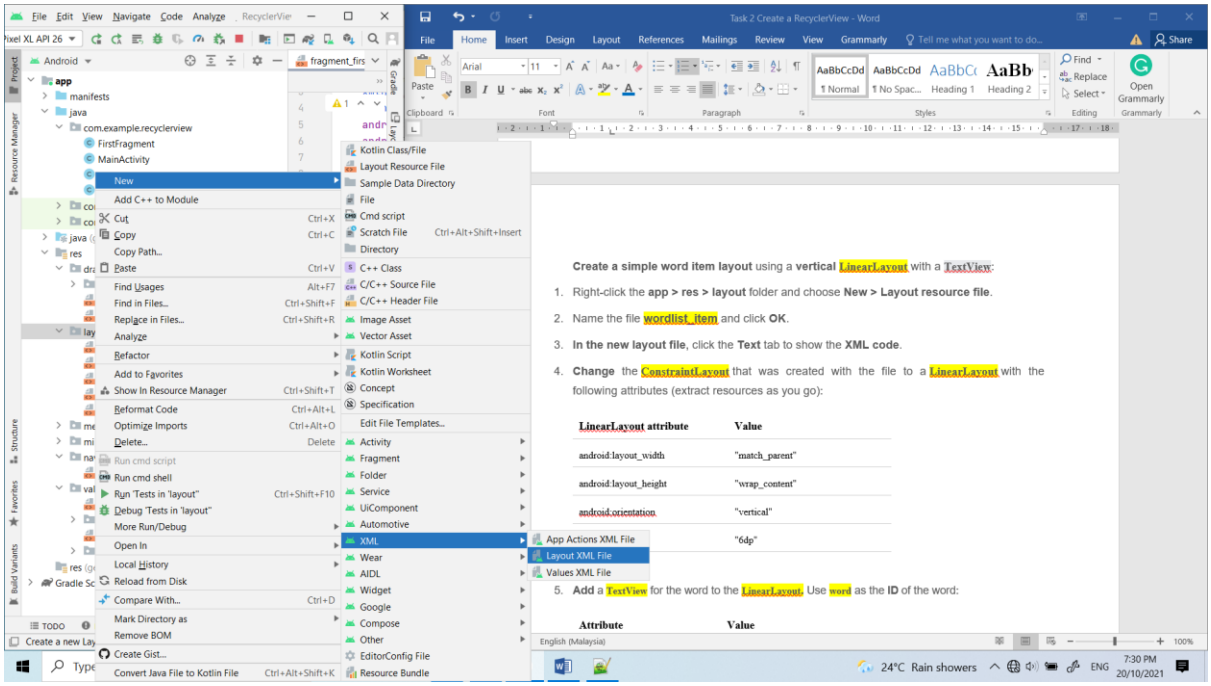
You need to specify the full path (**android.support.v7.widget.RecyclerView**), because RecyclerView is part of the Support Library.

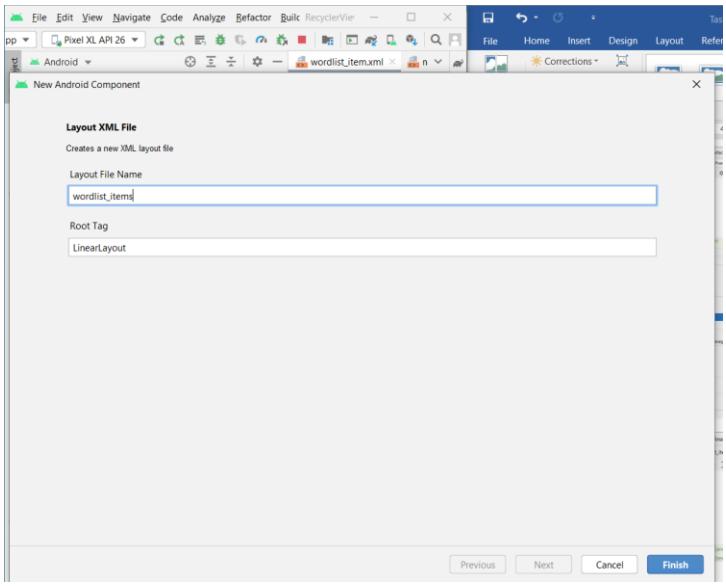


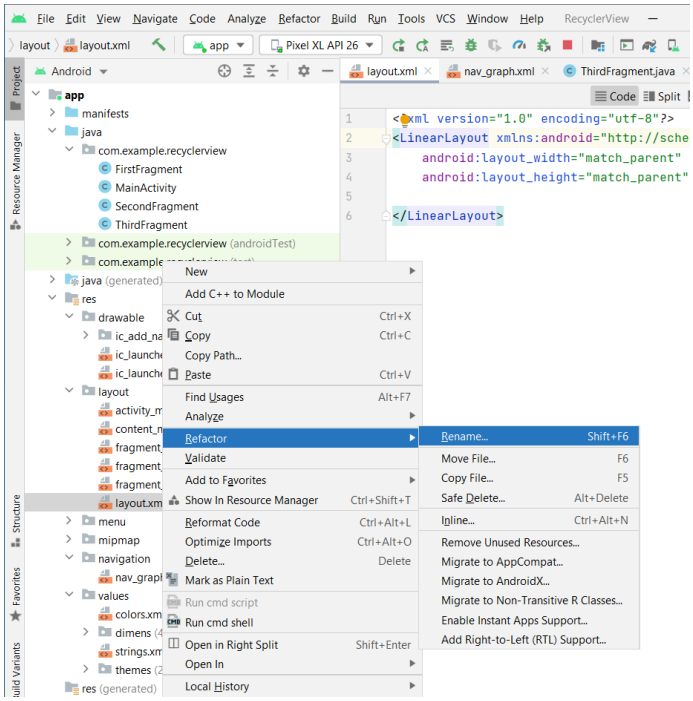
2**.2. Create the layout for one list item**

The adapter needs the layout for one item in the list. All the items use the **same layout**. You need to **specify that list item layout in a separate layout resource file**, because it is **used by the adapter**, separately from the RecyclerView.

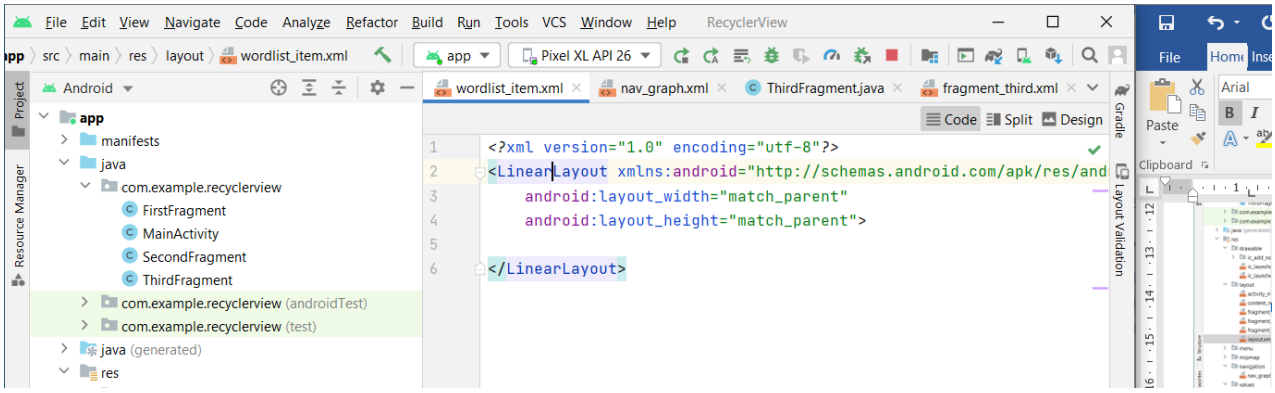



Task 2 Create a RecyclerView - Word
Create a simple word item layout using a vertical LinearLayout with a TextView:
1. Right-click the app > res > layout folder and choose New > Layout resource file.
2. Name the file wordlist_item and click OK.
3. In the new layout file, click the Text tab to show the XML code.
4. Change the ConstraintLayout that was created with the file to a LinearLayout with the following attributes (extract resources as you go):
LinearLayout attribute
Value
android:layout_width
"match_parent"
android:layout_height
"wrap_content"
android:orientation
"vertical"
"6dp"
5. Add a TextView for the word to the LinearLayout. Use word as the ID of the word:
Attribute
Value
New
Add C++ to Module
Cut
Copy
Copy Path...
Paste
Find Usages
Find in Files...
Replace in Files...
Analyze
Refactor
Add to Favorites
Show In Resource Manager
Reformat Code
Optimize Imports
Delete...
Run cmd script
Run cmd shell
Run 'Tests in 'layout''
Debug 'Tests in 'layout''
More Run/Debug
Open In
Local History
Reload from Disk
Compare With...
Mark Directory as
Remove BOM
Create Gist...
Convert Java File to Kotlin File
Kotlin Class/File
Layout Resource File
Sample Data Directory
File
Cmd script
Scratch File
Directory
C++ Class
C/C++ Source File
C/C++ Header File
Image Asset
Vector Asset
Kotlin Script
Kotlin Worksheet
Concept
Specification
Edit File Templates...
Activity
Fragment
Folder
Service
UiComponent
Automotive
XML
Wear
AIDL
Widget
Google
Compose
Other
EditorConfig File
Resource Bundle
App Actions XML File
Layout XML File
Values XML File


New Android Component
Layout XML File
Creates a new XML layout file
Layout File Name
wordlist_items
Root Tag
LinearLayout
Previous
Next
Cancel
Finish



If necessary

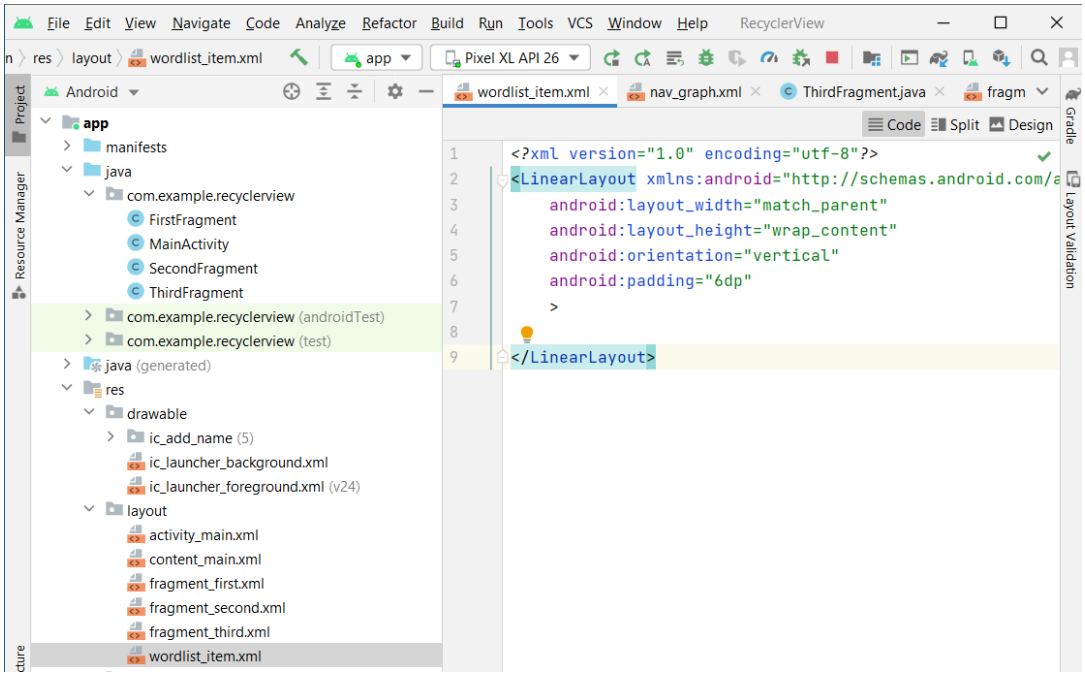


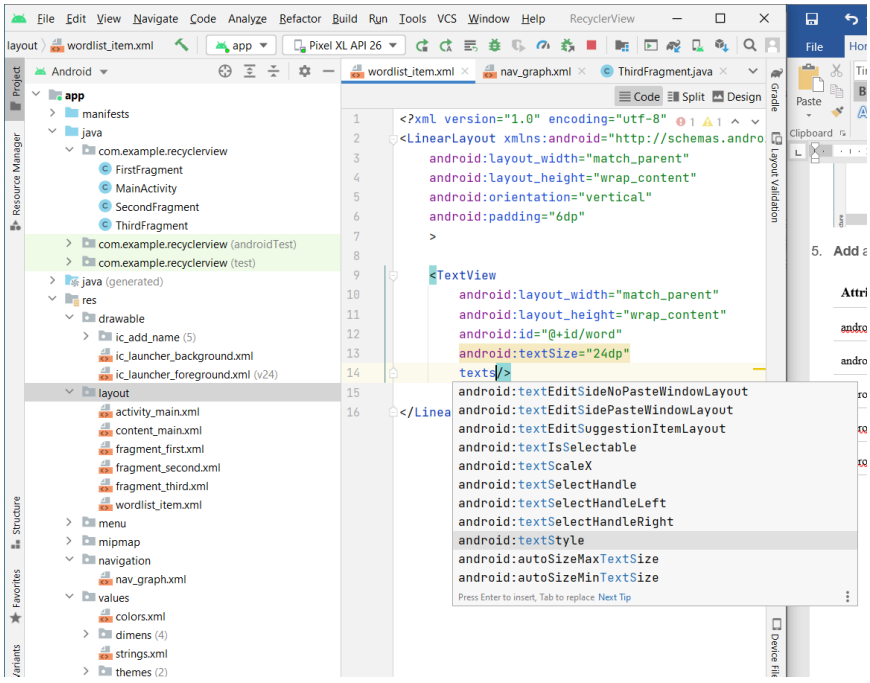
Create a simple word item layout using a **vertical LinearLayout** with a **TextView**:

1. Right-click the **app > res > layout** folder and choose **New > Layout resource file**.
2. Name the file **wordlist_item** and click **OK**.
3. **In the new layout file**, click the **Text** tab to show the **XML code**.
4. **Change** the **ConstraintLayout** that was created with the file to a **LinearLayout** with the following attributes (extract resources as you go):

LinearLayout attribute	Value
android:layout_width	"match_parent"

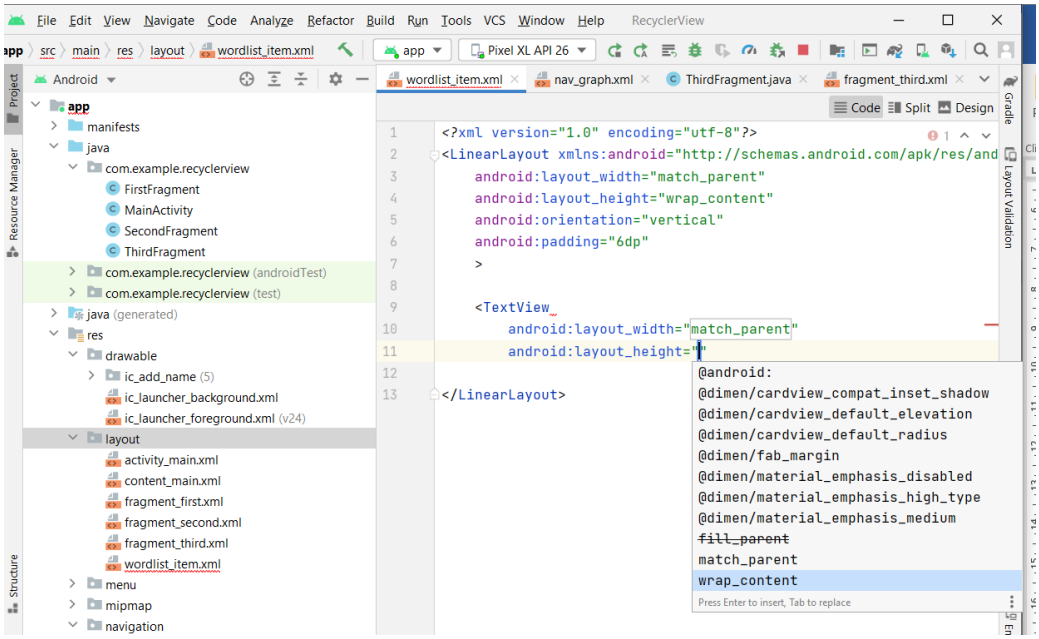
android:layout_height	"wrap_content"
android:orientation	"vertical"
android:padding	"6dp"

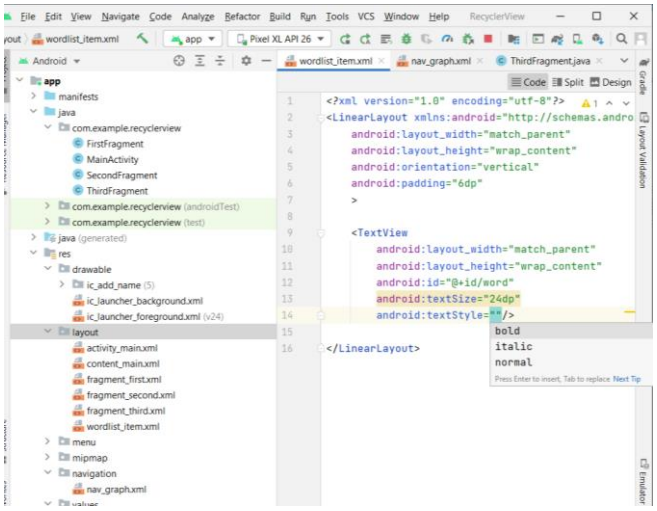




5. **Add** a **TextView** for the word to the **LinearLayout.** Use **word** as the **ID** of the word:

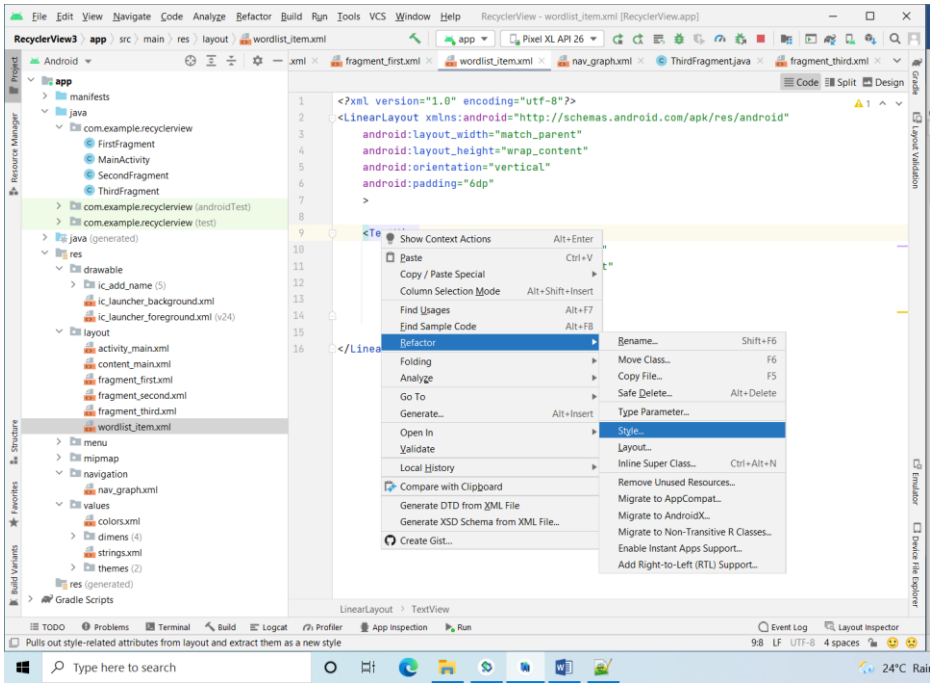
Attribute	Value
android:id	"@+id/word"
android:layout_width	"match_parent"
android:layout_height	"wrap_content"
android:textSize	"24sp"
android:textStyle	"bold"

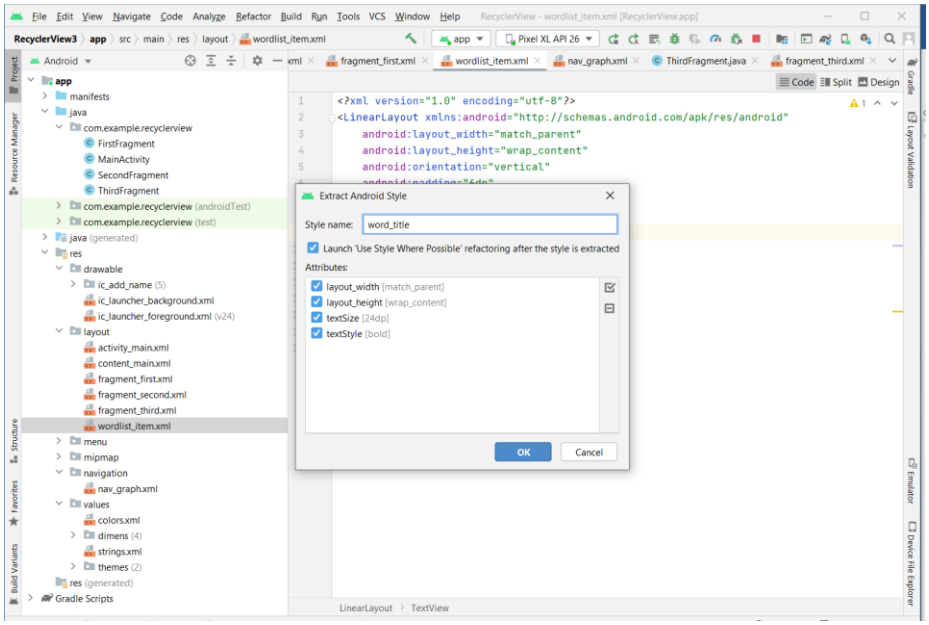


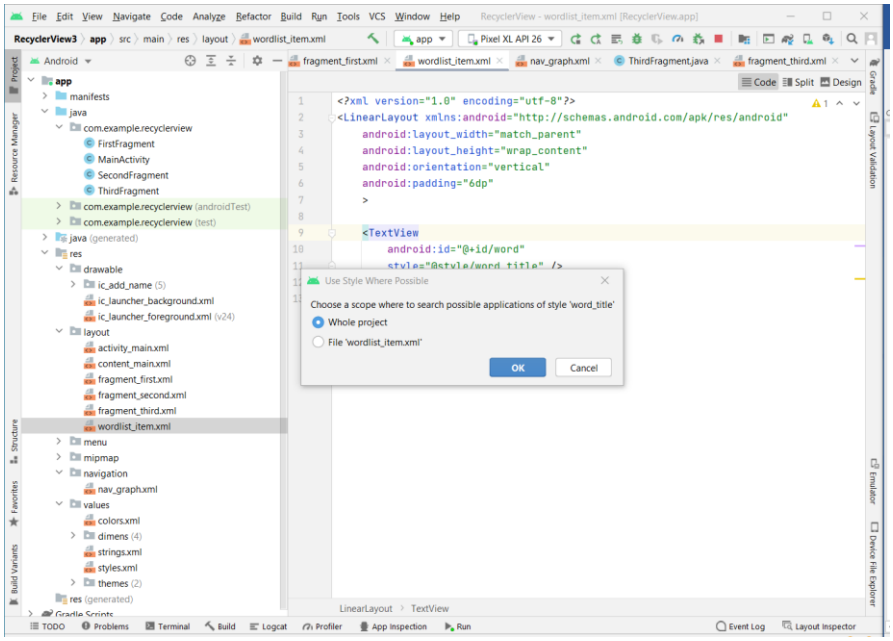


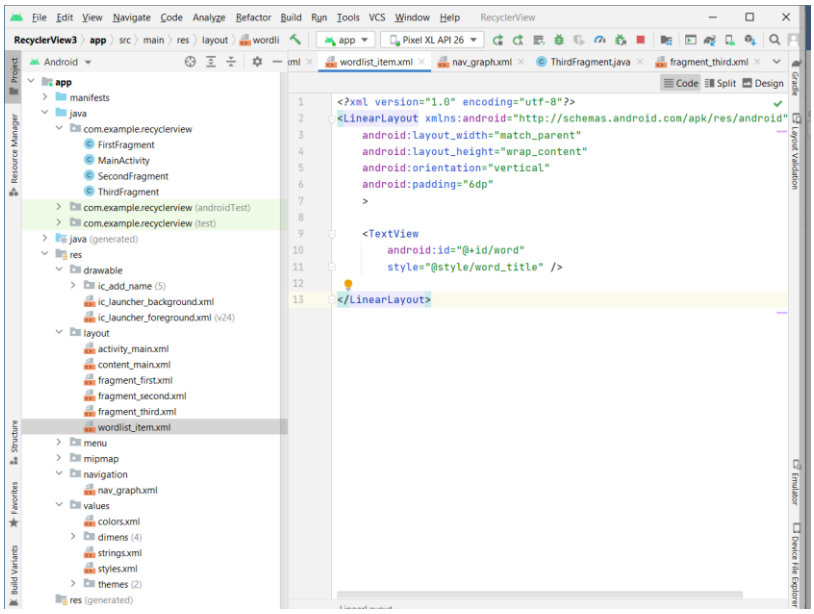
2.3 Create a style from the TextView attributes

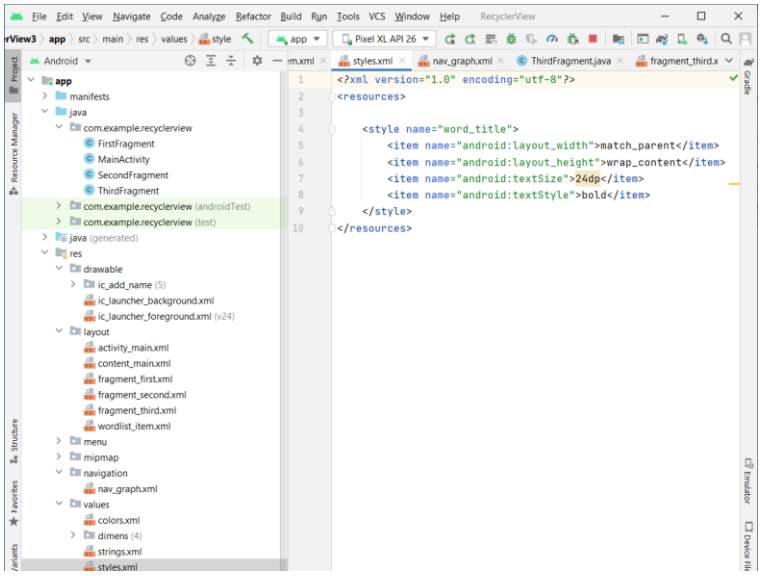
You can use **styles** to allow elements to share groups of display attributes. An easy way to create a style is to **extract the style of a UI element** that you already created. To extract the style information for the **word TextView** in **wordlist_item.xml**:






File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
RecyclerView - wordlist_item.xml [RecyclerView.app]
RecyclerView3 app src main res layout wordlist_item.xml
Pixel XL API 26
fragment_first.xml
wordlist_item.xml
nav_graph.xml
ThirdFragment.java
fragment_third.xml
Code Split Design
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="vertical"
android:padding="6dp"
>
<TextView
android:id="@+id/word"
Use Style Where Possible
Choose a scope where to search possible applications of style 'word_title'
Whole project
File 'wordlist_item.xml'
OK
Cancel
LinearLayout TextView
TODO Problems Terminal Build Logcat Profiler App Inspection Run Event Log Layout Inspector


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
RecyclerView
RecyclerView3 app src main res layout wordli
Pixel XL API 26
wordlist_item.xml
nav_graph.xml
ThirdFragment.java
fragment_third.xml
Code Split Design
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="vertical"
android:padding="6dp"
>
<TextView
android:id="@+id/word"
style="@style/word_title" />
</LinearLayout>



1. Open **wordlist_item.xml** if it is not already open.
2. **Right-click** (or Control-click) the **TextView** you just created in **wordlist_item.xml**, and choose **Refactor > Extract > Style**. The **Extract Android Style** dialog appears.
3. **Name** your **style** **word_title** and leave all other options selected. Select the **Launch 'Use Style Where Possible'** option. Then click **OK**.
4. When prompted, **apply the style** to the **Whole Project**.
5. Find and examine the **word_title** style in **values > styles.xml**.
6. Reopen **wordlist_item.xml** if it is not already open. The **TextView** now uses the style in place of individual styling properties, as shown below.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:padding="6dp">

    <TextView
        android:id="@+id/word"
        style="@style/word_title" />

</LinearLayout>
```

2**.4. Create an adapter**

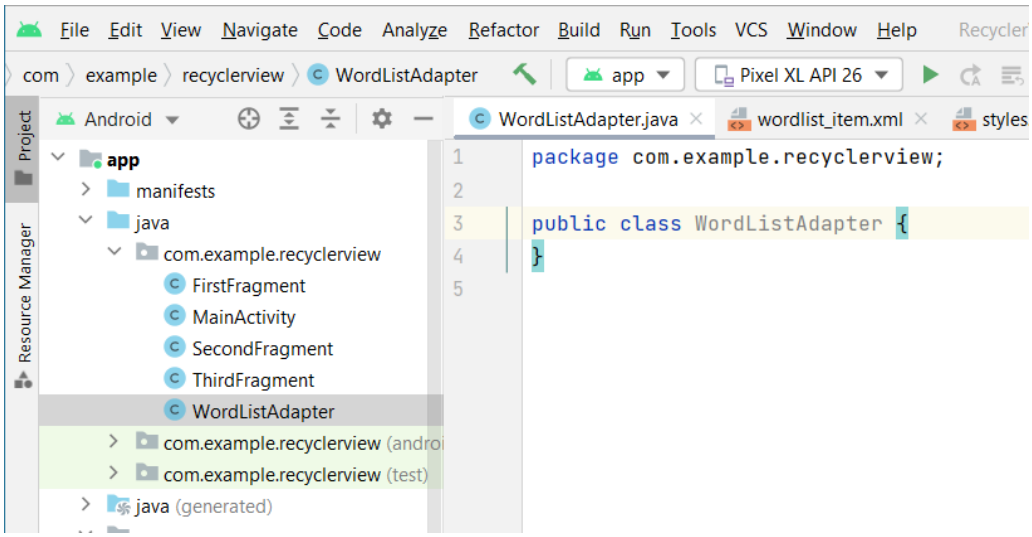
Android uses adapters (from the **Adapter class**) **to connect data with View items in a list.**

- There are many different kinds of adapters available, and you can also write custom adapters.
- In this task you will create an adapter that associates your list of words with word list **View** items.

To connect data with **View items**, the **adapter needs to know about the View items**. The adapter uses a **ViewHolder** that **describes a View item** and **its position** within the **RecyclerView**.

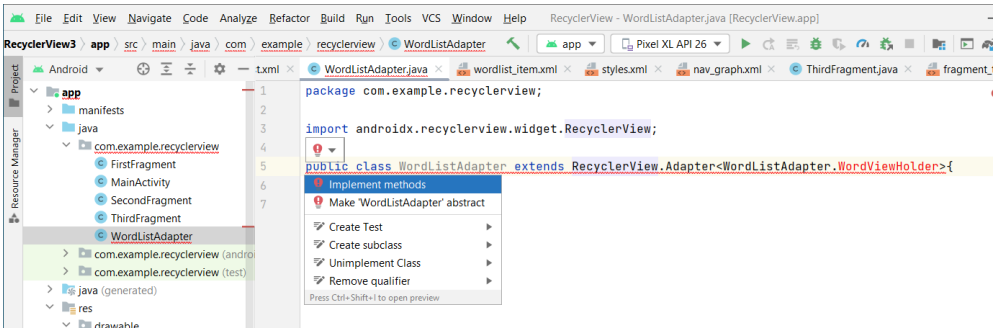
First, you will **build an adapter** that **bridges** the gap between the **data** in your **word list** and the **RecyclerView that displays it**:

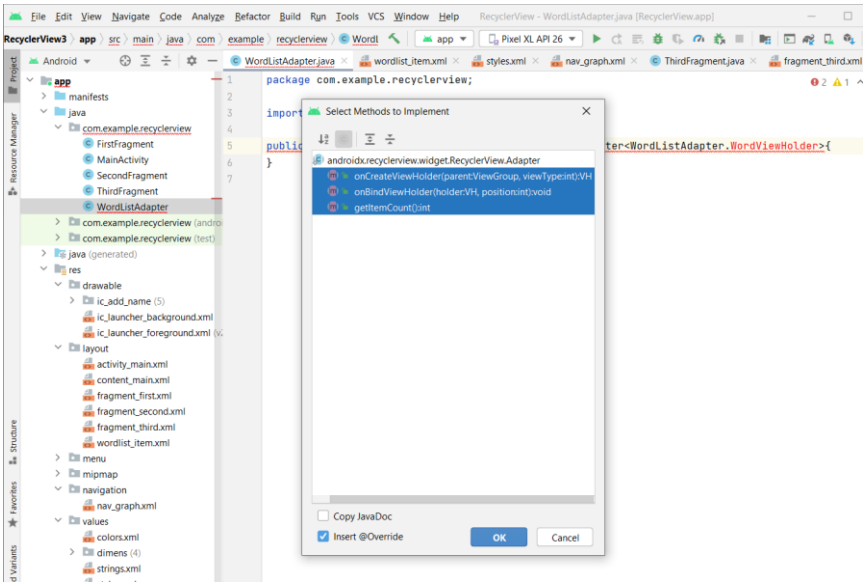
1. Right-click **java/com.android.example.recyclerview** and select **New > Java Class**.
2. Name the **class WordListAdapter**.

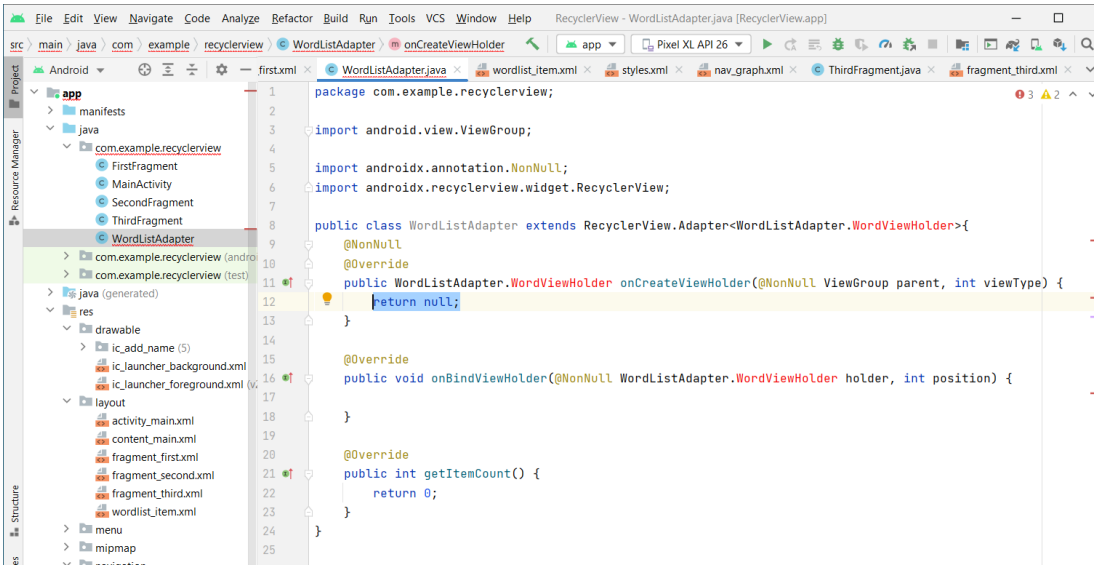


3. Give **WordListAdapter** the following signature:

```
public class WordListAdapter extends
    RecyclerView.Adapter<WordListAdapter.WordViewHolder> {}
```





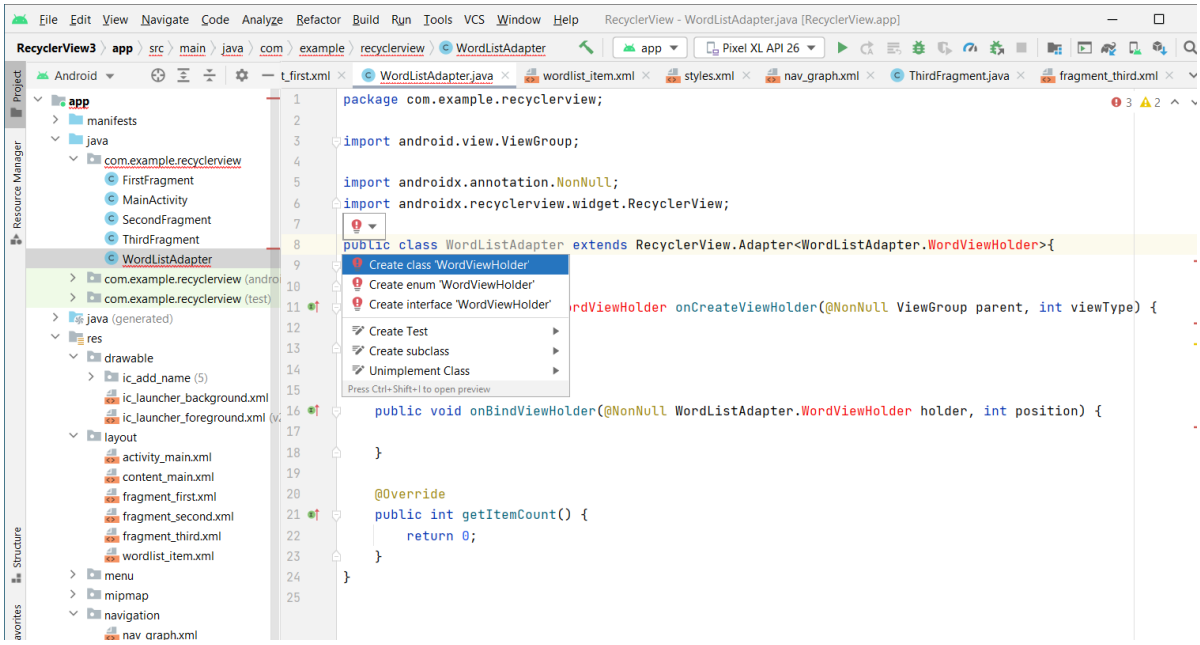


WordListAdapter extends a **generic adapter for RecyclerView** to use a **View holder** that is **specific for your app** and defined inside **WordListAdapter. WordViewHolder** shows an error, because you **have not yet defined it**.

4. Click the **class declaration** (**WordListAdapter**), then **click** the **red light bulb** on the left side of the pane. Choose **Implement methods**.

   A dialog appears that asks you to choose which methods to implement. Choose **all three methods** and click **OK**.

   Android Studio creates empty placeholders for all the methods. Note how **onCreateViewHolder** and **onBindViewHolder** both reference the **WordViewHolder**, which **hasn't been implemented yet**.



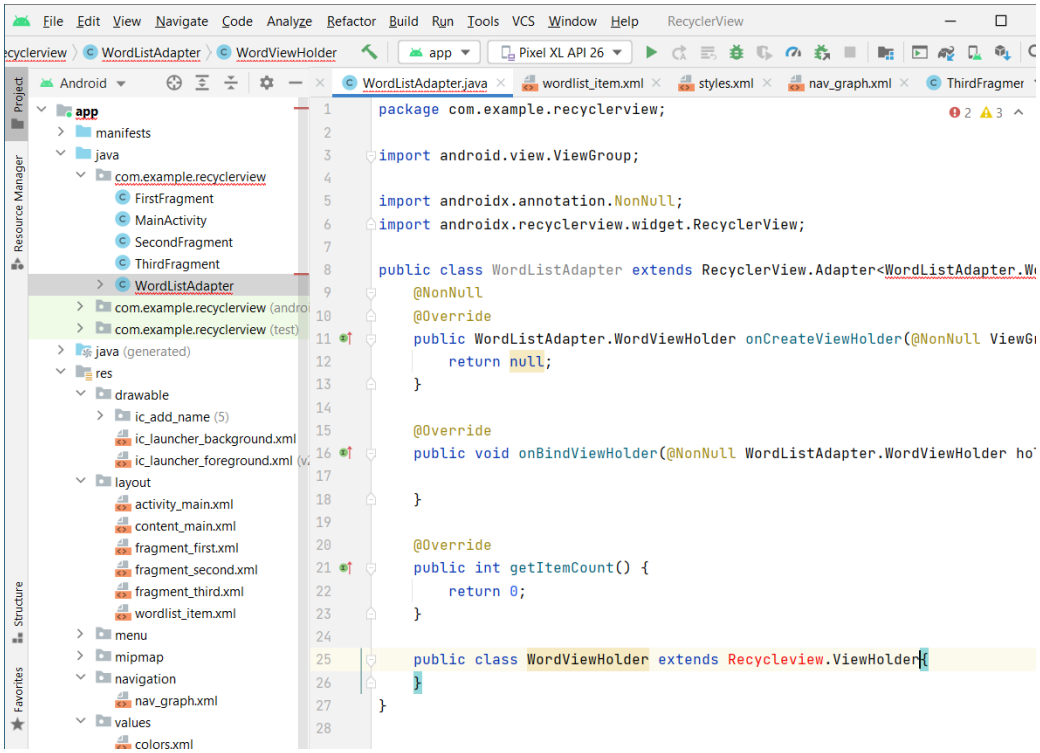
2.5 Create the ViewHolder for the adapter

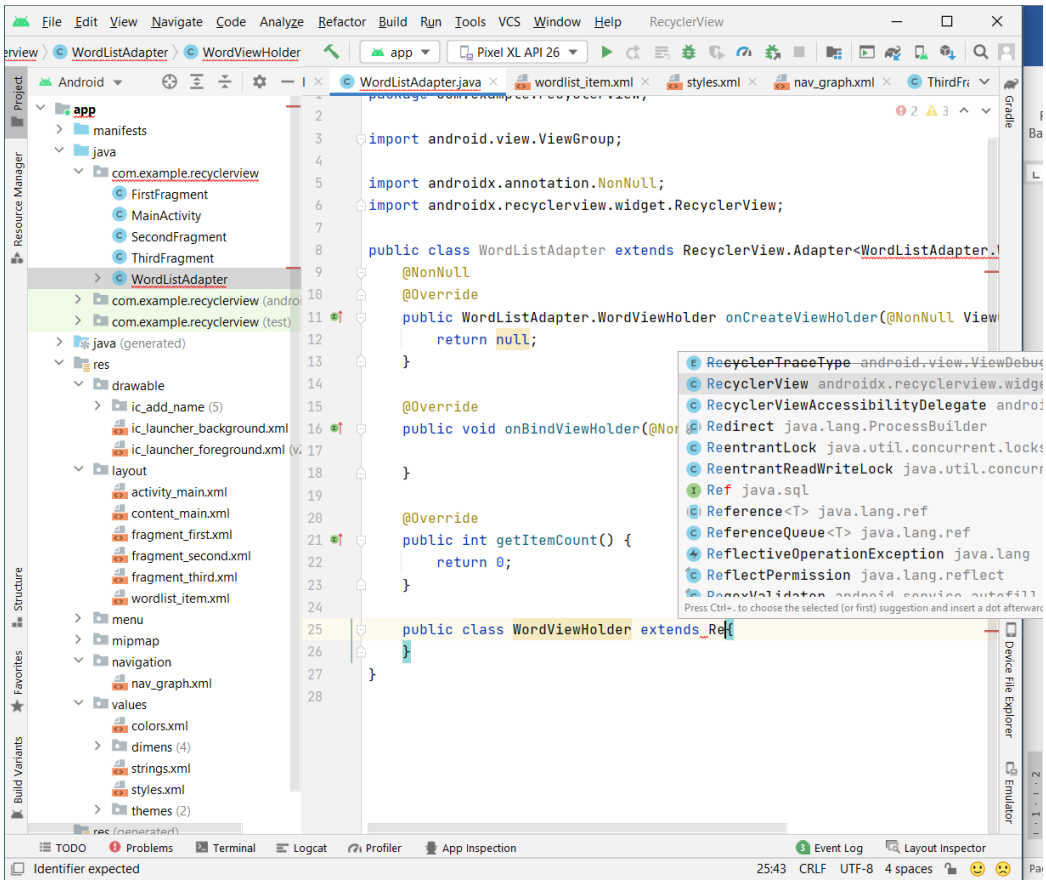
To **create the ViewHolder**, follow these steps:

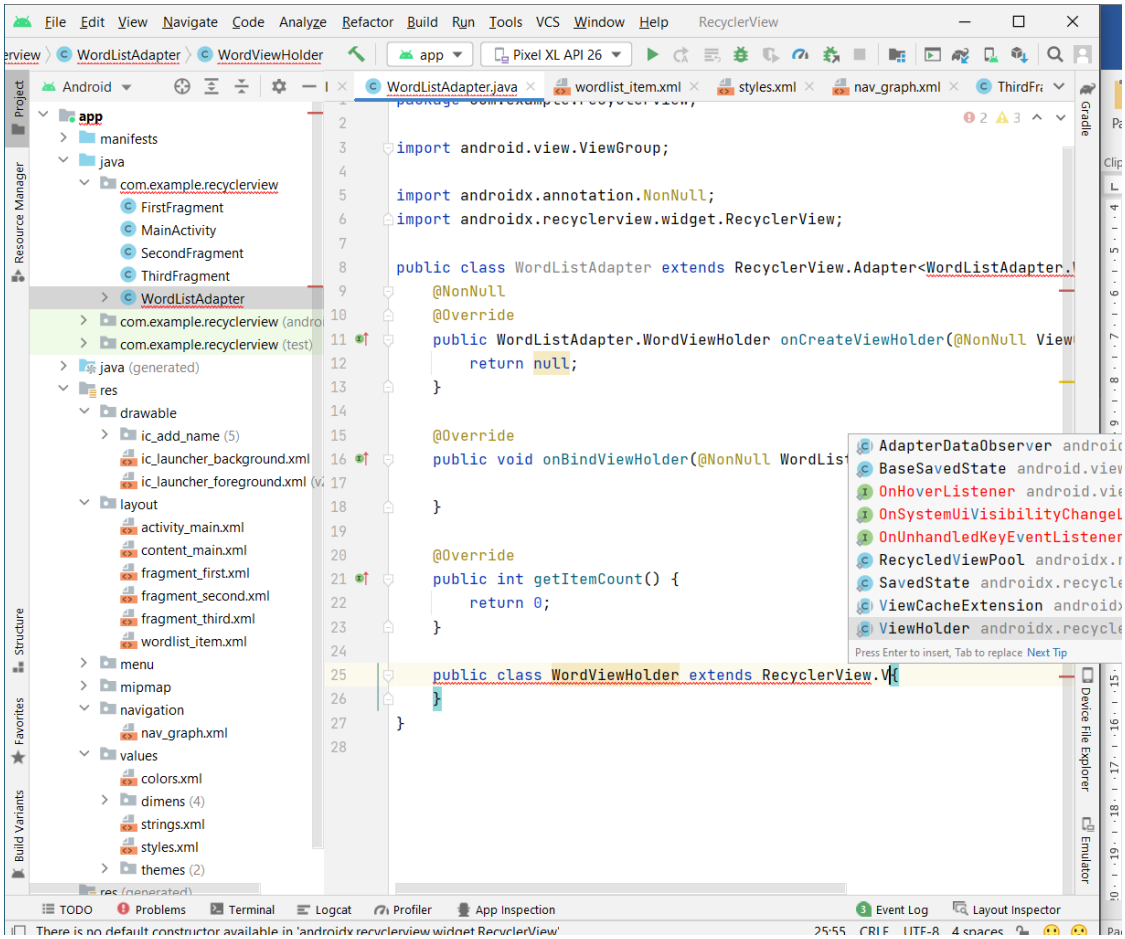
1. Inside the **WordListAdapter class**, **add a new WordViewHolder inner class** with this signature:

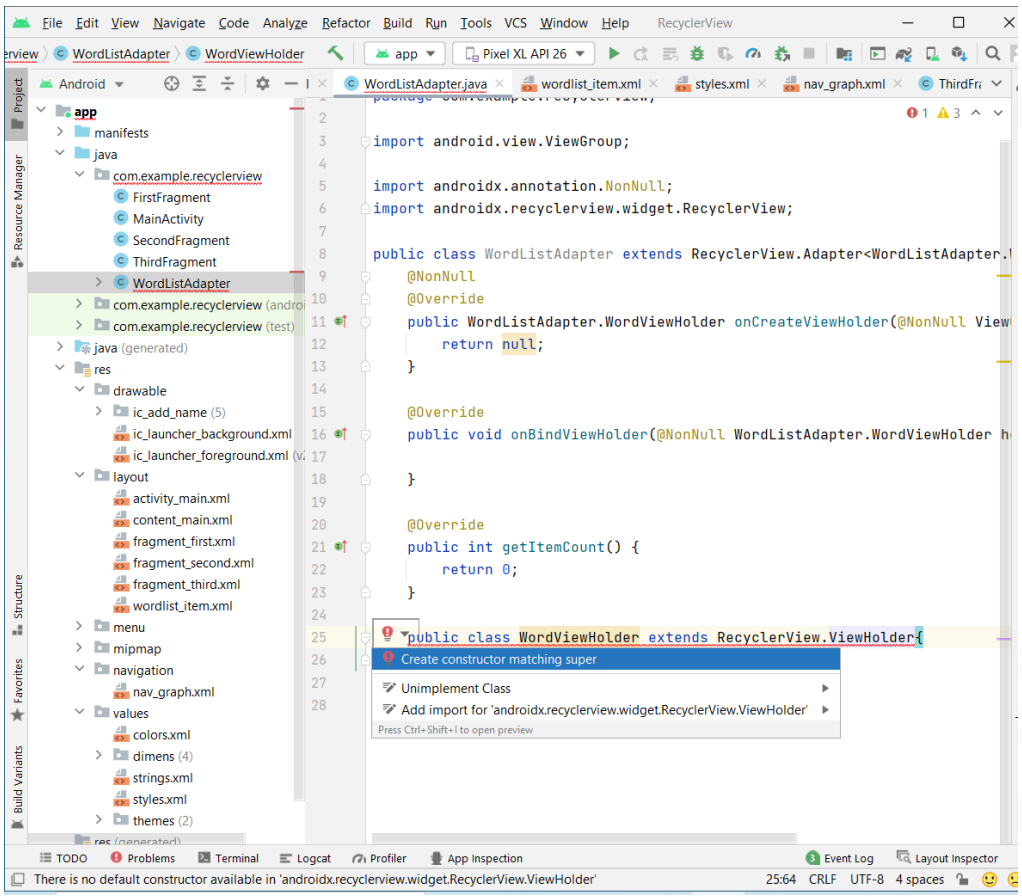
   class **WordViewHolder** extends **RecyclerView.ViewHolder** {}

   You will see an **error about a missing default constructor**. You can see details about the errors by hovering your mouse cursor over the red-underlined code or over any red horizontal line on the right margin of the editor pane.


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help RecyclerView
WordListAdapter WordViewHolder
app Pixel XL API 26
Android
WordListAdapter.java wordlist_item.xml styles.xml nav_graph.xml ThirdFragmer
app
manifests
java
com.example.recyclerview
FirstFragment
MainActivity
SecondFragment
ThirdFragment
WordListAdapter
com.example.recyclerview (andro
com.example.recyclerview (test)
java (generated)
res
drawable
ic_add_name (5)
ic_launcher_background.xml
ic_launcher_foreground.xml
layout
activity_main.xml
content_main.xml
fragment_first.xml
fragment_second.xml
fragment_third.xml
wordlist_item.xml
menu
mipmap
navigation
nav_graph.xml
values
colors.xml
package com.example.recyclerview;
import android.view.ViewGroup;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
public class WordListAdapter extends RecyclerView.Adapter<WordListAdapter.W
@NonNull
@Override
public WordListAdapter.WordViewHolder onCreateViewHolder(@NonNull ViewGr
return null;
}
@Override
public void onBindViewHolder(@NonNull WordListAdapter.WordViewHolder ho
}
@Override
public int getItemCount() {
return 0;
}
public class WordViewHolder extends Recycleview.ViewHolder{
}
}

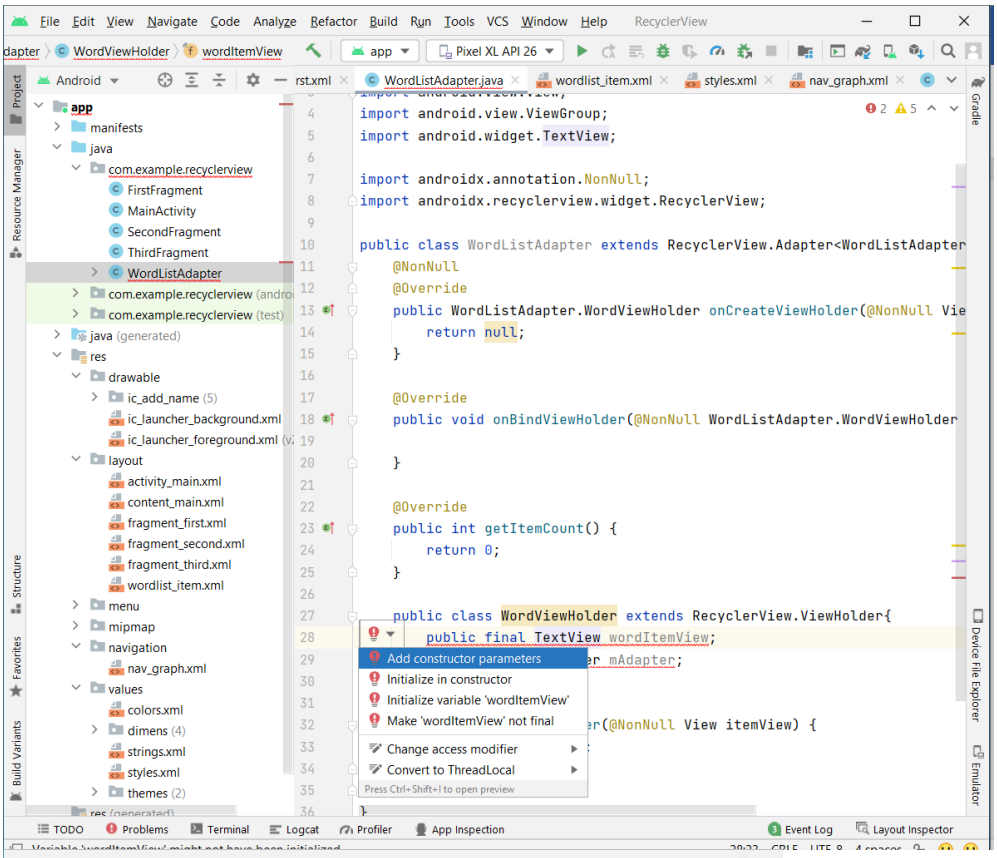

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help RecyclerView
WordListAdapter WordViewHolder
app Pixel XL API 26
WordListAdapter.java wordlist_item.xml styles.xml nav_graph.xml
import android.view.ViewGroup;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
public class WordListAdapter extends RecyclerView.Adapter<WordListAdapter.
@NonNull
@Override
public WordListAdapter.WordViewHolder onCreateViewHolder(@NonNull View
return null;
}
@Override
public void onBindViewHolder(@Nor
}
@Override
public int getItemCount() {
return 0;
}
public class WordViewHolder extends Ref{
}
}
RecyclerTraceType android.view.ViewDebug
RecyclerView androidx.recyclerview.widge
RecyclerViewAccessibilityDelegate androi
Redirect java.lang.ProcessBuilder
ReentrantLock java.util.concurrent.locks
ReentrantReadWriteLock java.util.concur
Ref java.sql
Reference<T> java.lang.ref
ReferenceQueue<T> java.lang.ref
ReflectiveOperationException java.lang
ReflectPermission java.lang.reflect
Press Ctrl+. to choose the selected (or first) suggestion and insert a dot afterwar
TODO Problems Terminal Logcat Profiler App Inspection Event Log Layout Inspector
Identifier expected
25:43 CRLF UTF-8 4 spaces

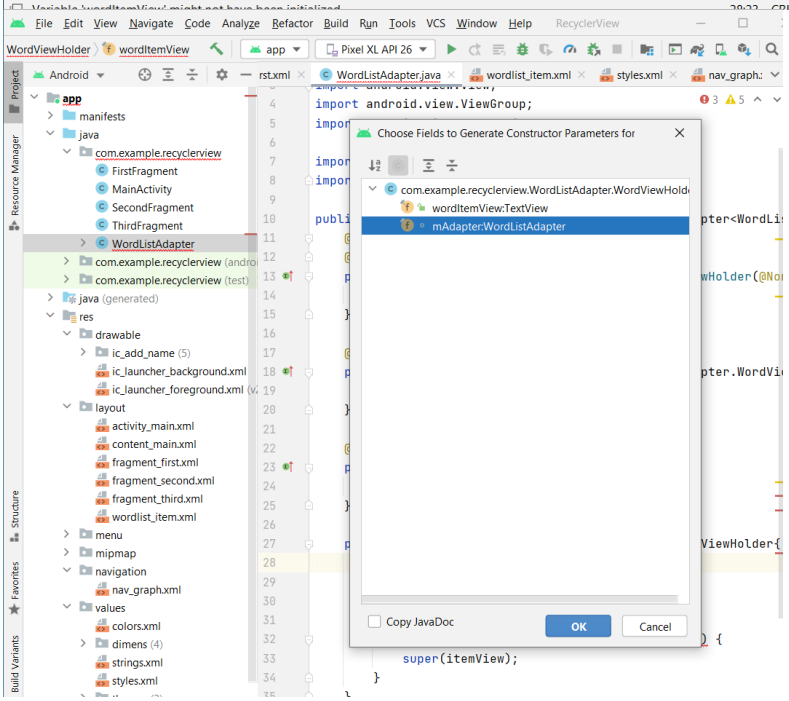

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help RecyclerView
WordListAdapter WordViewHolder
app Pixel XL API 26
Android
WordListAdapter.java wordlist_item.xml styles.xml nav_graph.xml
app
manifests
java
com.example.recyclerview
FirstFragment
MainActivity
SecondFragment
ThirdFragment
WordListAdapter
com.example.recyclerview (test)
java (generated)
res
drawable
ic_add_name (5)
ic_launcher_background.xml
ic_launcher_foreground.xml
layout
activity_main.xml
content_main.xml
fragment_first.xml
fragment_second.xml
fragment_third.xml
wordlist_item.xml
menu
mipmap
navigation
nav_graph.xml
values
colors.xml
dimens (4)
strings.xml
styles.xml
themes (2)
import android.view.ViewGroup;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
public class WordListAdapter extends RecyclerView.Adapter<WordListAdapter.
@NonNull
@Override
public WordListAdapter.WordViewHolder onCreateViewHolder(@NonNull View
return null;
@Override
public void onBindViewHolder(@NonNull WordList
@Override
public int getItemCount() {
return 0;
public class WordViewHolder extends RecyclerView.V
AdapterDataObserver
BaseSavedState
OnHoverListener
OnSystemUiVisibilityChangeL
OnUnhandledKeyEventListener
RecycledViewPool
SavedState
ViewCacheExtension
ViewHolder
Press Enter to insert, Tab to replace Next Tip
TODO Problems Terminal Logcat Profiler App Inspection Event Log Layout Inspector
There is no default constructor available in 'androidx.recyclerview.widget.RecyclerView'
25:55 CRLF UTF-8 4 spaces

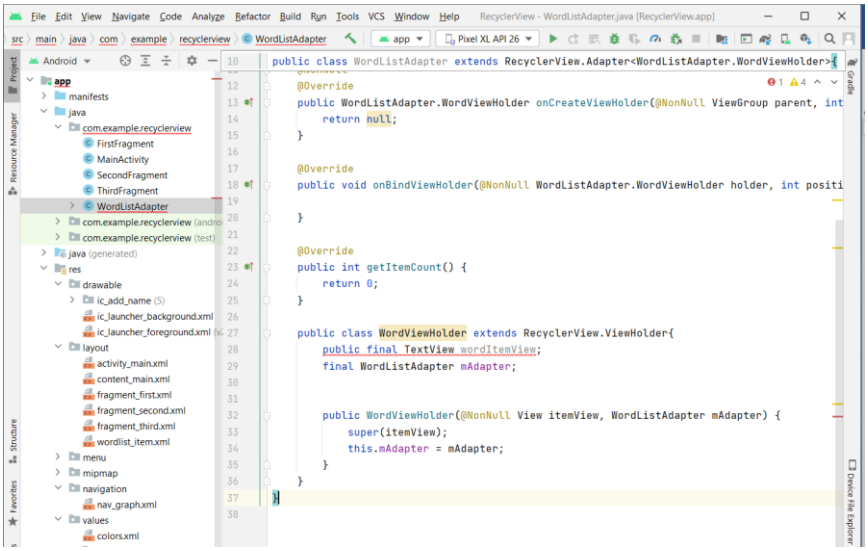

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help RecyclerView
WordListAdapter WordViewHolder
app Pixel XL API 26
Android
WordListAdapter.java wordlist_item.xml styles.xml nav_graph.xml
app
manifests
java
com.example.recyclerview
FirstFragment
MainActivity
SecondFragment
ThirdFragment
WordListAdapter
com.example.recyclerview (test)
java (generated)
res
drawable
ic_add_name (5)
ic_launcher_background.xml
ic_launcher_foreground.xml
layout
activity_main.xml
content_main.xml
fragment_first.xml
fragment_second.xml
fragment_third.xml
wordlist_item.xml
menu
mipmap
navigation
nav_graph.xml
values
colors.xml
dimens (4)
strings.xml
styles.xml
themes (2)
import android.view.ViewGroup;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
public class WordListAdapter extends RecyclerView.Adapter<WordListAdapter.
@NonNull
@Override
public WordListAdapter.WordViewHolder onCreateViewHolder(@NonNull View
return null;
@Override
public void onBindViewHolder(@NonNull WordListAdapter.WordViewHolder h
@Override
public int getItemCount() {
return 0;
public class WordViewHolder extends RecyclerView.ViewHolder{
Create constructor matching super
Unimplement Class
Add import for 'androidx.recyclerview.widget.RecyclerView.ViewHolder'
Press Ctrl+Shift+I to open preview
TODO Problems Terminal Logcat Profiler App Inspection Event Log Layout Inspector
There is no default constructor available in 'androidx.recyclerview.widget.RecyclerView.ViewHolder'
25:64 CRLF UTF-8 4 spaces

2. **Add variables** to the **WordViewHolder inner class** for the **TextView** and the adapter:

public final **TextView** **wordItemView**;
final **WordListAdapter** **mAdapter**;

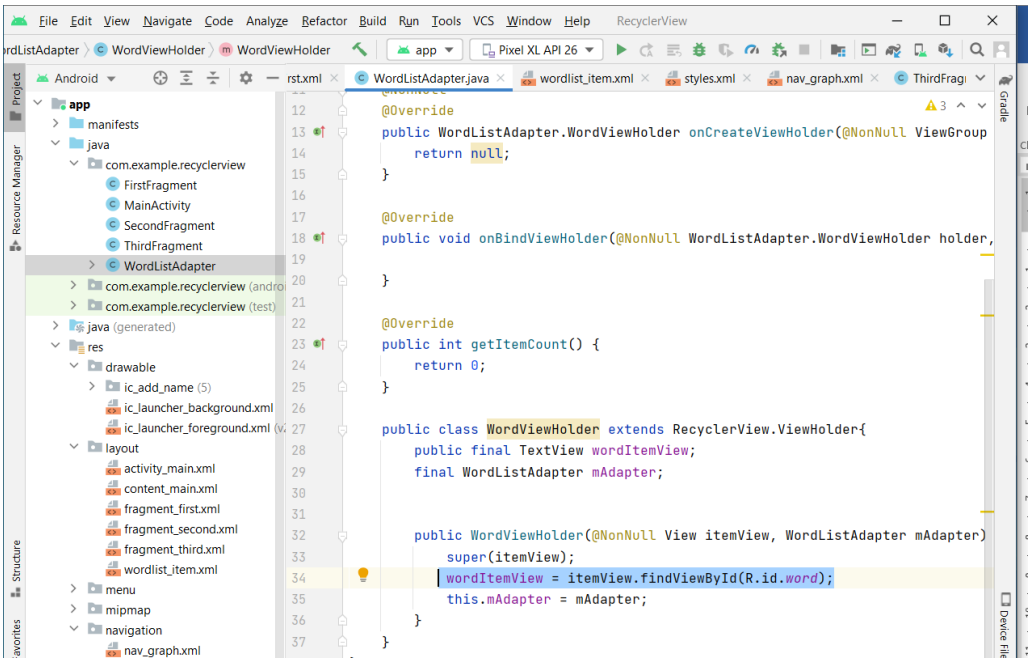


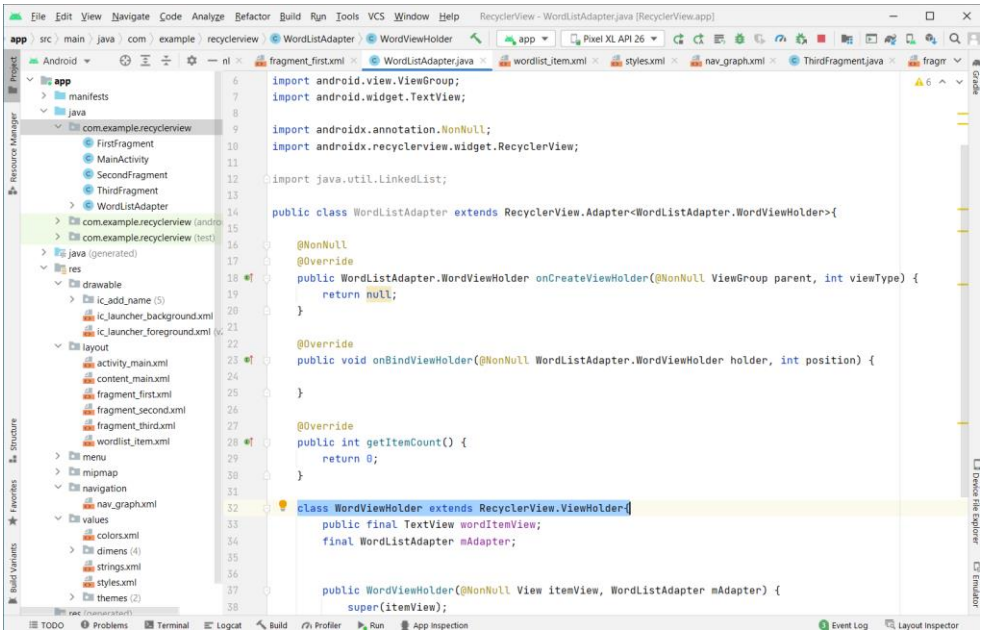




3. In the **inner class WordViewHolder**, **add** a **constructor** that initializes the **ViewHolder TextView** from the **word** XML resource, and sets its adapter:

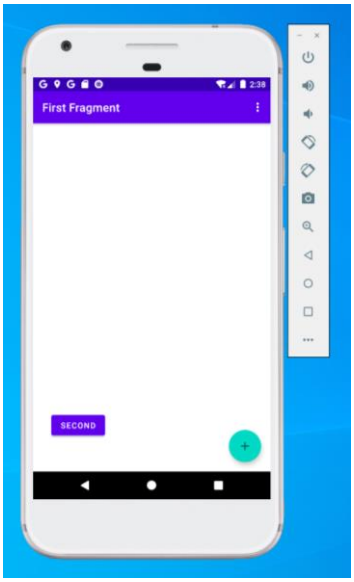
```
public WordViewHolder(View itemView, WordListAdapter adapter) {
  super(itemView);
  wordItemView = itemView.findViewById(R.id.word);
  this.mAdapter = adapter;
}
```

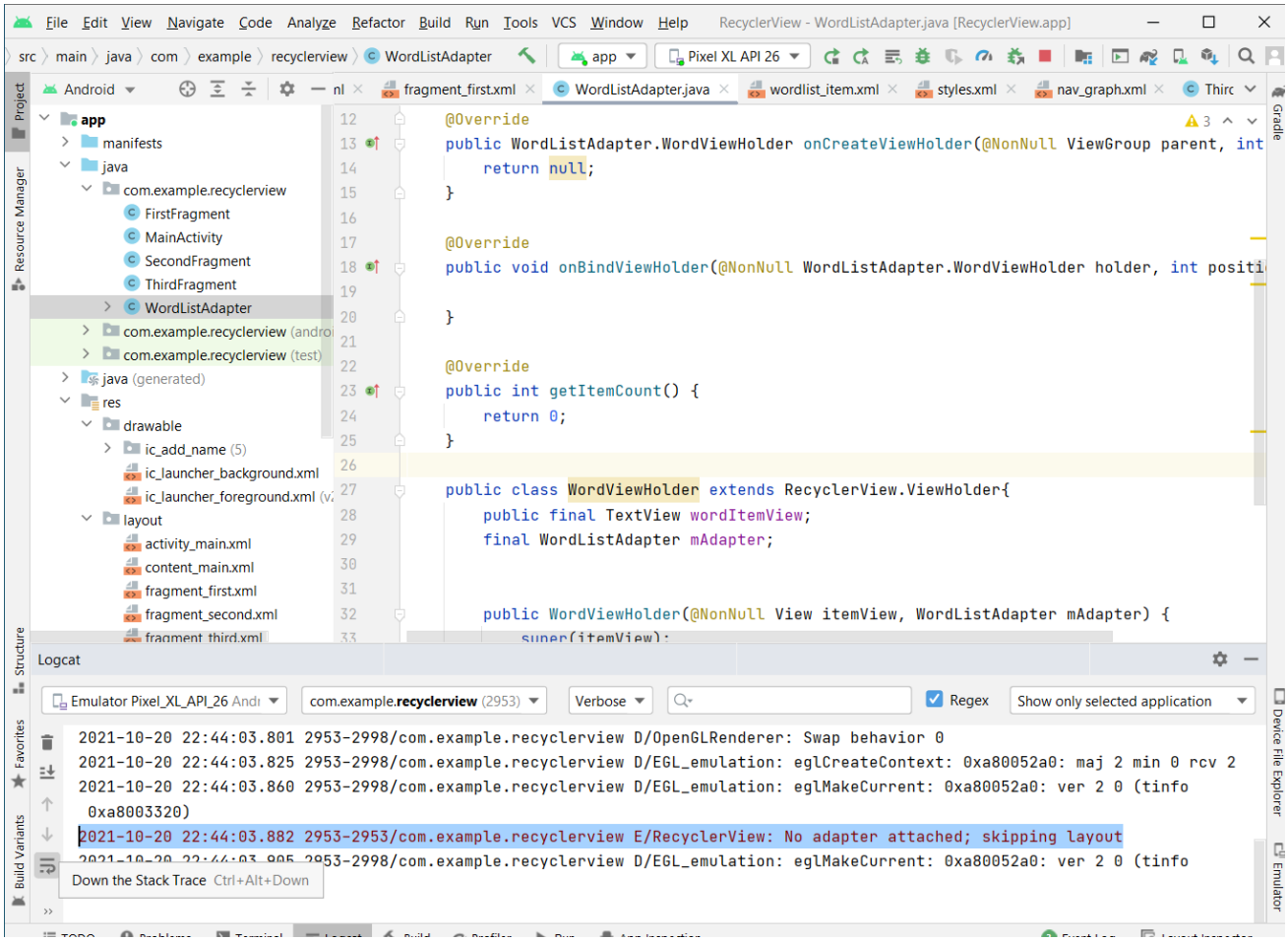




4. Run your app to make sure that you have **no errors**. Your will still see only a **blank view**.

5. Click the **Logcat** tab to see the **Logcat** pane, and note the **E/RecyclerView: No adapter attached;** skipping layout warning. **You will attach the adapter to the** `RecyclerView` **in another step.**





2.6 Storing your data in the adapter

You need to **hold your data in the adapter**, and **WordListAdapter needs a constructor that initializes the word list** from the **data**. Follow these steps:

1. To **hold your data in the adapter**, create a **private linked list of strings** in **WordListAdapter** and call it **mWordList**.

   private final LinkedList<String> **mWordList**;

2. You can now **fill in the getItemCount() method** to **return the size** of **mWordList**:

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

3. **WordListAdapter** needs a **constructor** that **initializes the word list from the data**.

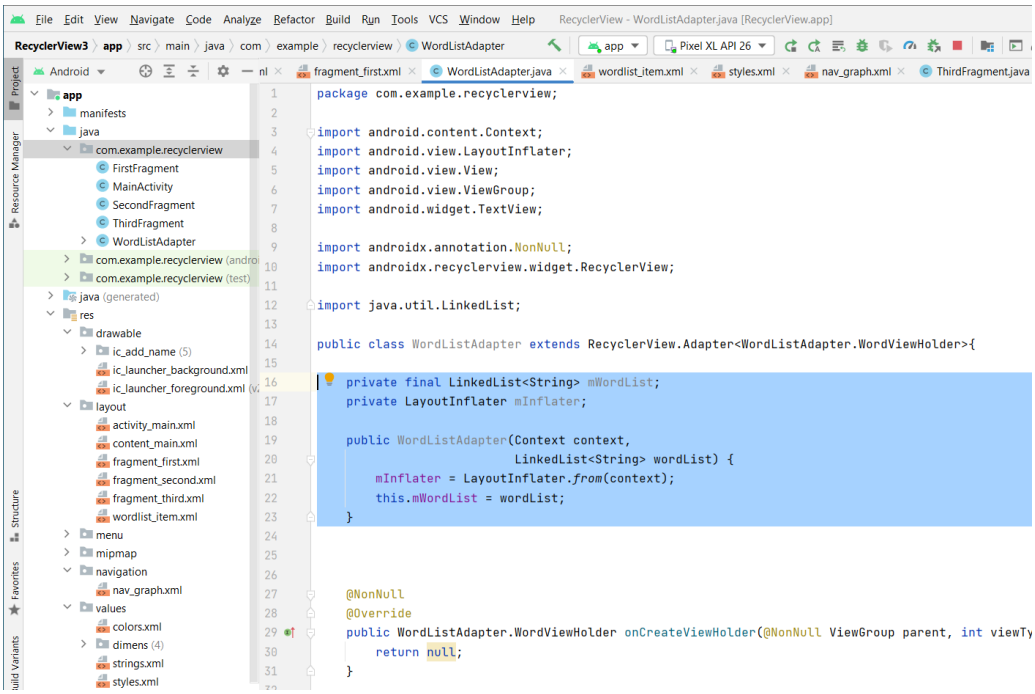
   - To create a `View` for a list item, the `WordListAdapter` needs to inflate the XML for a list item.

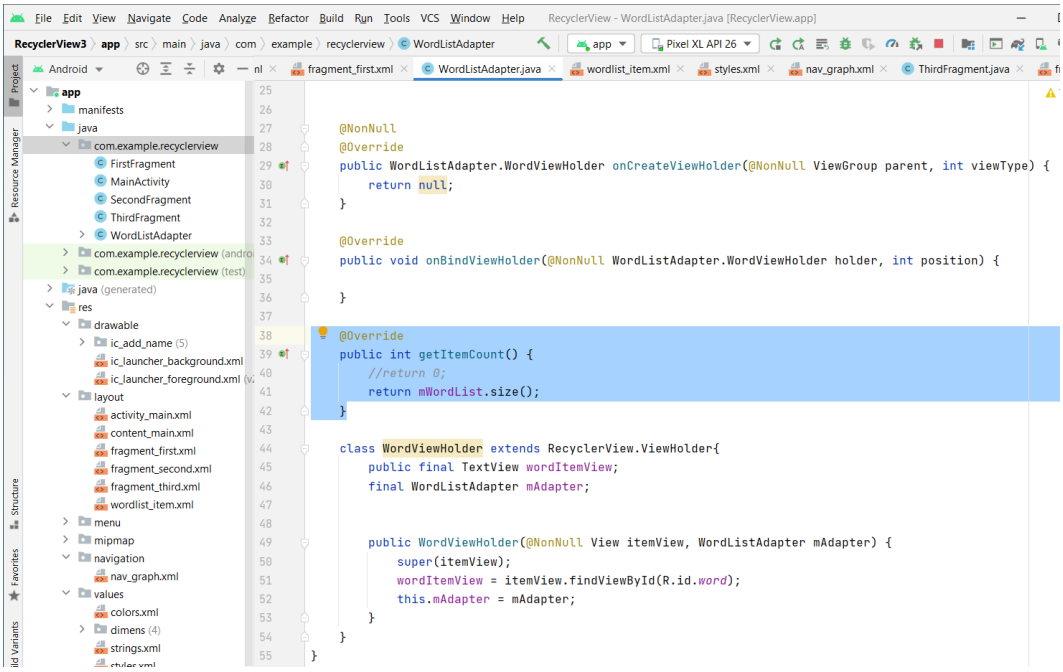
- You use a *layout inflator* for that job. **LayoutInflator** reads a layout XML description and converts it into the corresponding `View` items.
- Start by creating a **member variable for the inflater** in **WordListAdapter**:

```
private LayoutInflater mInflater;
```

4. **Implement the constructor** for `WordListAdapter`.
   - The constructor needs to have a **context parameter**, and a **linked list of words** with the app's data.
   - The method needs to **instantiate a LayoutInflator** for **mInflater** and set **mWordList** to the **passed in data**:

```
public WordListAdapter(Context context,
                       LinkedList<String> wordList) {
  mInflater = LayoutInflater.from(context);
  this.mWordList = wordList;
}
```





5. Fill out the **onCreateViewHolder() method** with this code:

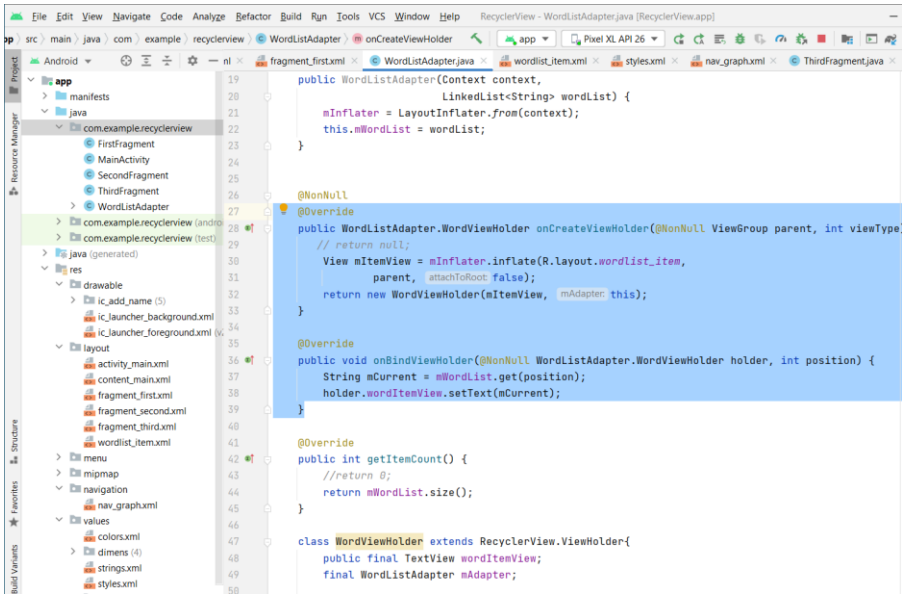
```
@Override
public WordViewHolder onCreateViewHolder(ViewGroup parent,
                                         int viewType) {
  View mItemView = mInflater.inflate(R.layout.wordlist_item,
                                     parent, false);
  return new WordViewHolder(mItemView, this);
}
```

- The **onCreateViewHolder()** method is similar to the onCreate() method.
- It **inflates the item layout**, and **returns** a **ViewHolder with the layout** and the **adapter**.

6. Fill out the onBindViewHolder() method with the code below:

```
@Override
public void onBindViewHolder(WordViewHolder holder, int position) {
  String mCurrent = mWordList.get(position);
  holder.wordItemView.setText(mCurrent);
}
```

The **onBindViewHolder() method connects** your **data to the view holder**.



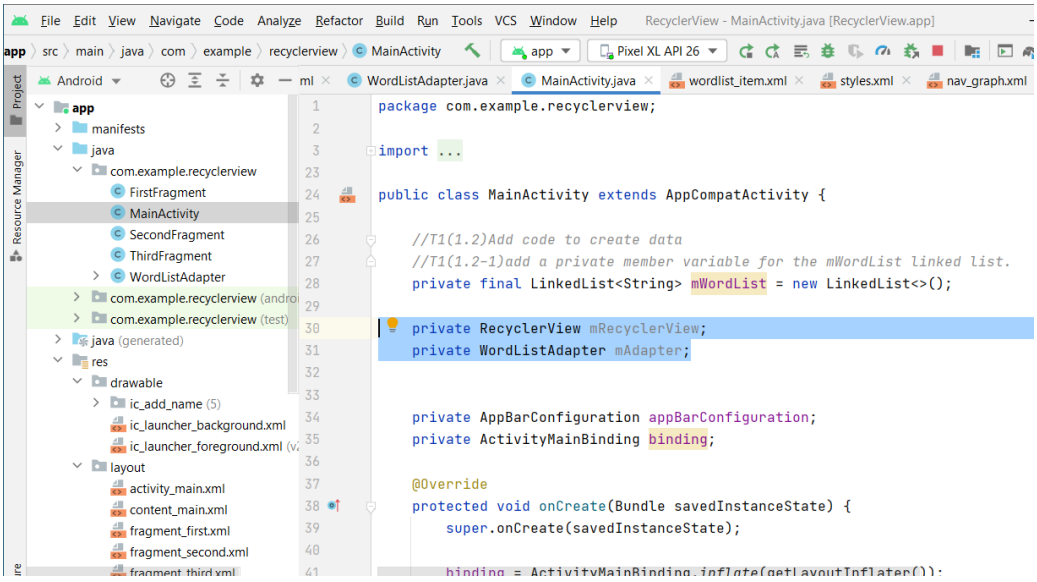
7. Run your app to make sure that there are no errors.

2**.7. Create the RecyclerView in the Activity**

Now that you have an adapter with a ViewHolder, you can **finally create a RecyclerView and connect all the pieces to display your data.**

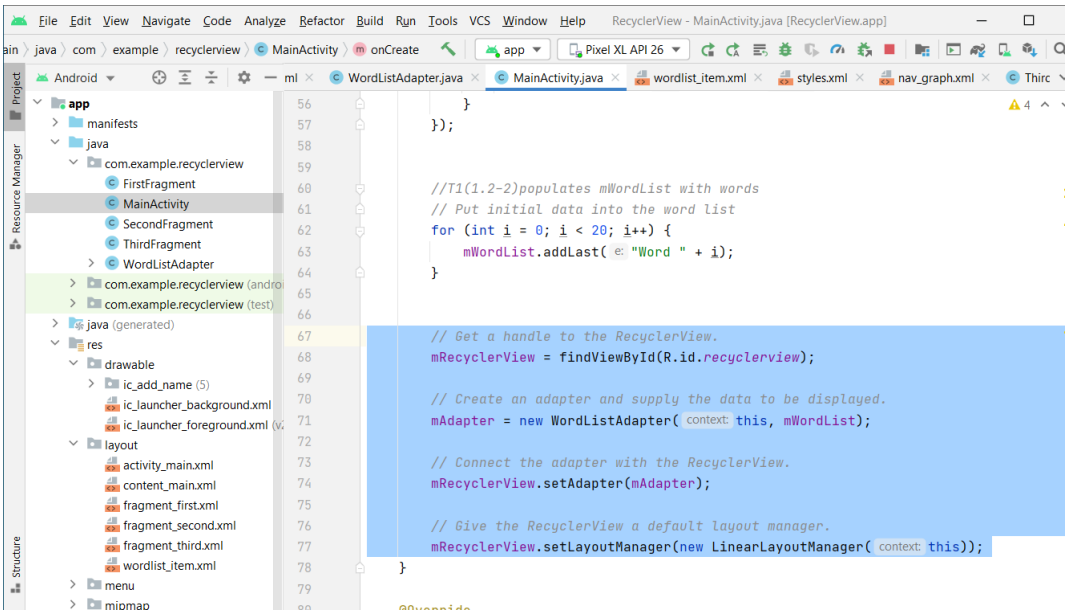
1. Open **MainActivity**.
2. **Add member variables** for the **RecyclerView** and the **adapter**.

```
private RecyclerView mRecyclerView;
private WordListAdapter mAdapter;
```



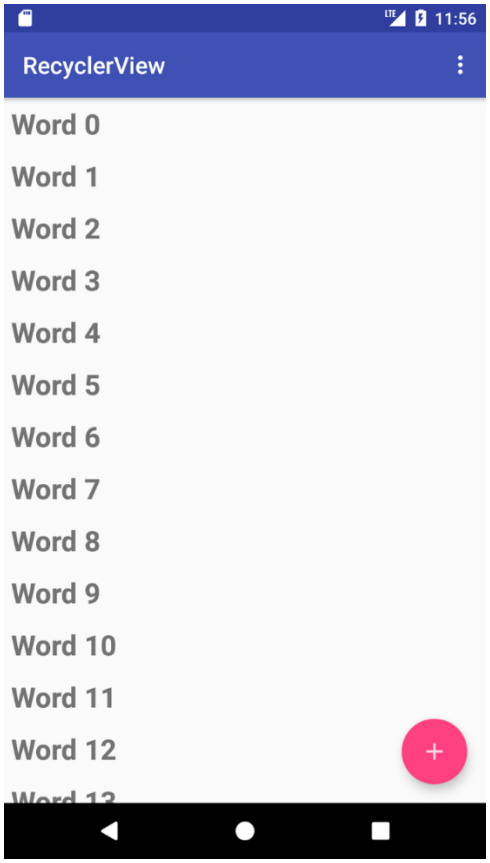
3. In the **onCreate() method** of **MainActivity**, add the following code that **creates the RecyclerView and connects it with an adapter and the data**. The comments explain each line. You must **insert this code *after* the `mWordList` initialization**.

```
// Get a handle to the RecyclerView.
mRecyclerView = findViewById(R.id.recyclerview);

// Create an adapter and supply the data to be displayed.
mAdapter = new WordListAdapter(this, mWordList);

// Connect the adapter with the RecyclerView.
mRecyclerView.setAdapter(mAdapter);

// Give the RecyclerView a default layout manager.
mRecyclerView.setLayoutManager(new LinearLayoutManager(this));
```



4. Run your app.

   You should see your **list of words displayed**, and you **can scroll the list**.


11:56
RecyclerView
Word 0
Word 1
Word 2
Word 3
Word 4
Word 5
Word 6
Word 7
Word 8
Word 9
Word 10
Word 11
Word 12