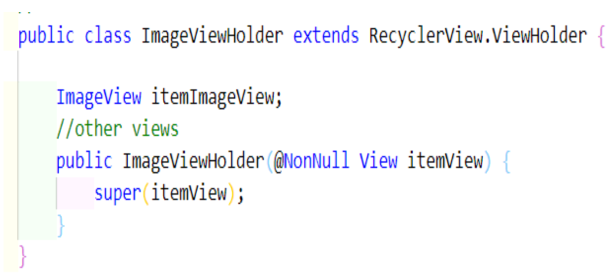
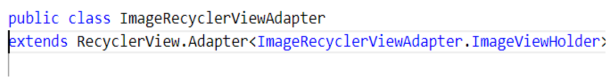
RecyclerView Adapter



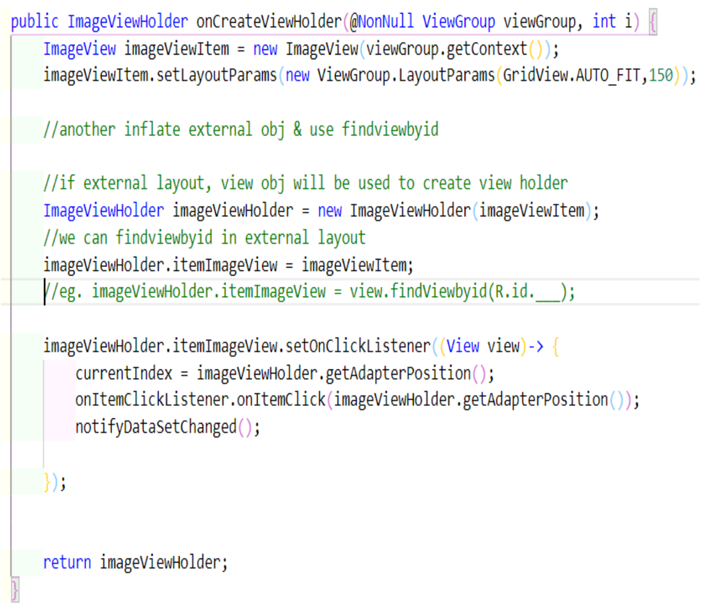
```java
public class ImageViewHolder extends RecyclerView.ViewHolder {

    ImageView itemImageView;
    //other views
    public ImageViewHolder(@NonNull View itemView) {
        super(itemView);
    }
}
```

Then extend the class and implement the methods, create constructors



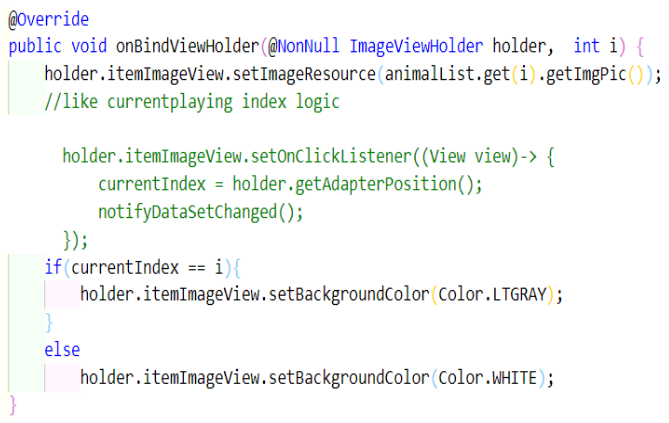
```java
public class ImageRecyclerViewAdapter
extends RecyclerView.Adapter<ImageRecyclerViewAdapter.ImageViewHolder>
```

Oncreate..



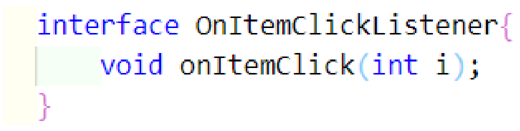
```java
public ImageViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int i) {
    ImageView imageViewItem = new ImageView(viewGroup.getContext());
    imageViewItem.setLayoutParams(new ViewGroup.LayoutParams(GridView.AUTO_FIT,150));

    //another inflate external obj & use findviewbyid

    //if external layout, view obj will be used to create view holder
    ImageViewHolder imageViewHolder = new ImageViewHolder(imageViewItem);
    //we can findviewbyid in external layout
    imageViewHolder.itemImageView = imageViewItem;
    //eg. imageViewHolder.itemImageView = view.findViewbyid(R.id.__);

    imageViewHolder.itemImageView.setOnClickListener((View view)-> {
        currentIndex = imageViewHolder.getAdapterPosition();
        onItemClickListener.onItemClick(imageViewHolder.getAdapterPosition());
        notifyDataSetChanged();

    });


    return imageViewHolder;
}
```

OnBind..



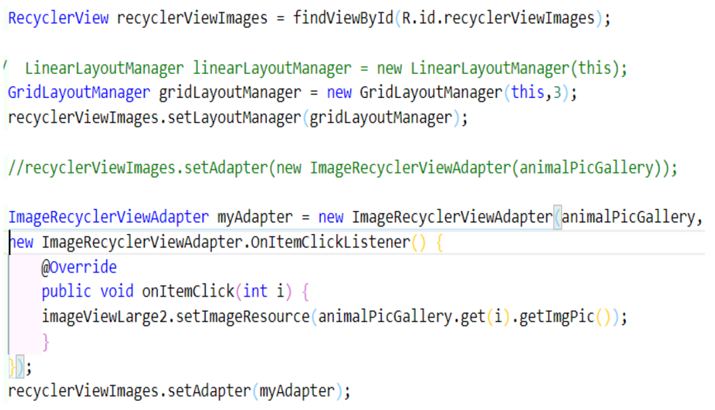
```java
@Override
public void onBindViewHolder(@NonNull ImageViewHolder holder,  int i) {
    holder.itemImageView.setImageResource(animalList.get(i).getImgPic());
    //like currentplaying index logic

     holder.itemImageView.setOnClickListener((View view)-> {
         currentIndex = holder.getAdapterPosition();
         notifyDataSetChanged();
     });
    if(currentIndex == i){
        holder.itemImageView.setBackgroundColor(Color.LTGRAY);
    }
    else
        holder.itemImageView.setBackgroundColor(Color.WHITE);
}
```

Don't forget Interface



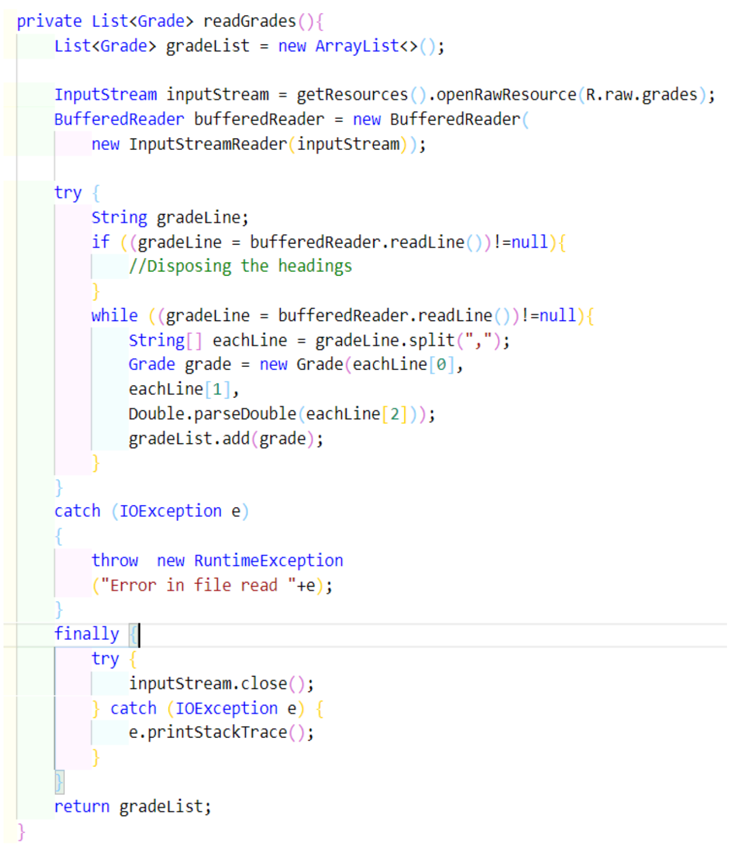
```java
interface OnItemClickListener{
    void onItemClick(int i);
}
```

On main Activity



```java
RecyclerView recyclerViewImages = findViewById(R.id.recyclerViewImages);

  LinearLayoutManager linearLayoutManager = new LinearLayoutManager(this);
GridLayoutManager gridLayoutManager = new GridLayoutManager(this,3);
recyclerViewImages.setLayoutManager(gridLayoutManager);

//recyclerViewImages.setAdapter(new ImageRecyclerViewAdapter(animalPicGallery));

ImageRecyclerViewAdapter myAdapter = new ImageRecyclerViewAdapter(animalPicGallery,
new ImageRecyclerViewAdapter.OnItemClickListener() {
    @Override
    public void onItemClick(int i) {
    imageViewLarge2.setImageResource(animalPicGallery.get(i).getImgPic());
    }
});
recyclerViewImages.setAdapter(myAdapter);
```

File read from csv to List



```java
private List<Grade> readGrades(){
    List<Grade> gradeList = new ArrayList<>();

    InputStream inputStream = getResources().openRawResource(R.raw.grades);
    BufferedReader bufferedReader = new BufferedReader(
        new InputStreamReader(inputStream));

    try {
        String gradeLine;
        if ((gradeLine = bufferedReader.readLine())!=null){
            //Disposing the headings
        }
        while ((gradeLine = bufferedReader.readLine())!=null){
            String[] eachLine = gradeLine.split(",");
            Grade grade = new Grade(eachLine[0],
            eachLine[1],
            Double.parseDouble(eachLine[2]));
            gradeList.add(grade);
        }
    }
    catch (IOException e)
    {
        throw  new RuntimeException
        ("Error in file read "+e);
    }
    finally {
        try {
            inputStream.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    return gradeList;
}
```

Build Features Activate

```
buildFeatures {
    viewBinding true
}
```

In main activity

```
ActivityMainBinding binding;
```

then, after that all elements called binding.id

```
binding = ActivityMainBinding.inflate(getLayoutInflater());
View view = binding.getRoot();
setContentView(view);
```

External binding

```
LayoutStudentBinding binding = LayoutStudentBinding.inflate
        (LayoutInflater.from(viewGroup.getContext()),viewGroup,false);
TextView textViewStudentID = binding.textViewStudentID;
TextView textViewStudentName = binding.textViewStudentName;
TextView textViewStudentDept = binding.textViewStudentDept;
```

txtview.setText to set Data

return binding.getRoot(); at the end to return view

Simple Entity

```
@Entity(tableName = "STUDENTS")
public class Student {
    @NonNull
    @PrimaryKey
    @ColumnInfo(name = "studentid")
    String studentId;
    @ColumnInfo(name = "studentname")
    String studentName;
    @ColumnInfo(name = "studentdept")
    String studentDept;
```

Entity with Foreign Key

```
@Entity(tableName = "GRADES",primaryKeys = {"courseid","studentid"},
        foreignKeys = @ForeignKey(entity = Student.class,
                parentColumns = "studentid",
                childColumns = "studentid",
                onDelete = ForeignKey.CASCADE))
public class Grade {
    @ColumnInfo(name = "courseid")
    @NonNull
    String courseId;

    @ColumnInfo(name = "studentid")
    @NonNull
    String studentId;

    @ColumnInfo(name = "studentgrade")
    Double studentGrade;
```

Dao Interface

```
@Dao
public interface StudentDao {
    @Insert(onConflict = OnConflictStrategy.IGNORE)
    void insertStudents(Student... students);

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    long[] insertStudentsFromList(List<Student> students);

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    void insertOneStudent(Student student);

    @Query("SELECT * FROM STUDENTS")
    List<Student> getAllStudents();

    @Query("SELECT * FROM STUDENTS WHERE studentid IN (:StdIds)")
    List<Student> getStudentsInfoFromIds(List<String> StdIds);

    @Delete
    int deleteOneStudent(Student student);

    @Delete
    int deleteStudentsFromList(List<Student> students);

    @Query("DELETE FROM STUDENTS")
    void deleteAllStudents();

    @Query("DELETE FROM STUDENTS WHERE studentid =:stdId")
    int deleteStudentWithId(String stdId);
}
```

```
@Query("UPDATE GRADES SET studentgrade = 1.1*studentgrade WHERE studentid
IN (:stdId) AND courseid = :courseId")
int increaseGradesForStudentsInCourse(List<String> stdId,String courseId);
```

Database with multiple Dao-StudentsDatabase.class

```
@Database(entities = {Student.class, Grade.class},version = 1,exportSchema = false)
public abstract class StudentsDatabase extends RoomDatabase {//Make the DB Class abstract
    public abstract StudentDao studentDao(); //just put the Daos in here
    public abstract GradeDao gradeDao(); //just put the Daos in here
    public abstract StudentGradeDao studentGradeDao();
}
```

Using db in activity

```
StudentsDatabase database;

database = Room.databaseBuilder(
        getApplicationContext(),StudentsDatabase.class,"Students.db"
).build();
```

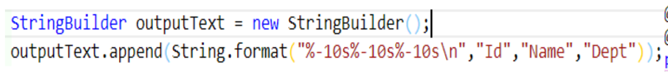
Run the queries inside executor service

```
ExecutorService executorService = Executors.newSingleThreadExecutor();
executorService.execute(()-> {
    int retValue = database.studentDao().deleteStudentWithId("312345");
    List<Student> students = database.studentDao().getAllStudents();
```

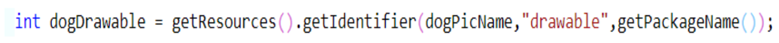
Run all ui inside this

```
runOnUiThread(()->{
    StudentAdapter studentAdapter = new StudentAdapter(allDbStudents);
    listViewStudents.setAdapter(studentAdapter);}
);
```

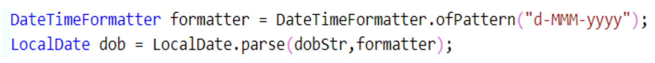
String Builder


```java
StringBuilder outputText = new StringBuilder();
outputText.append(String.format("%-10s%-10s%-10s\n","Id","Name","Dept"));
```

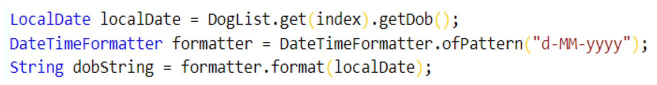
Getting resource using their names


```java
int dogDrawable = getResources().getIdentifier(dogPicName,"drawable",getPackageName());
```

String to LocalDate – dobStr is String

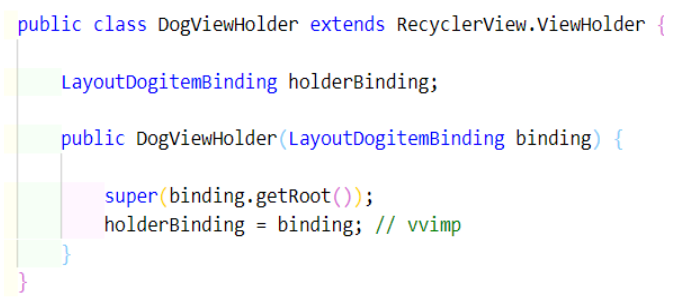

```java
DateTimeFormatter formatter = DateTimeFormatter.ofPattern("d-MMM-yyyy");
LocalDate dob = LocalDate.parse(dobStr,formatter);
```

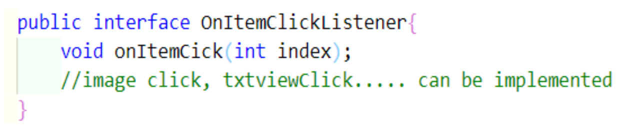
LocalDate to String


```java
LocalDate localDate = DogList.get(index).getDob();
DateTimeFormatter formatter = DateTimeFormatter.ofPattern("d-MM-yyyy");
String dobString = formatter.format(localDate);
```

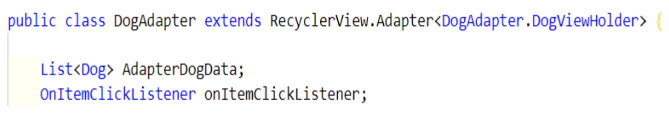
Recycler View using Binding

1) inner class/holder

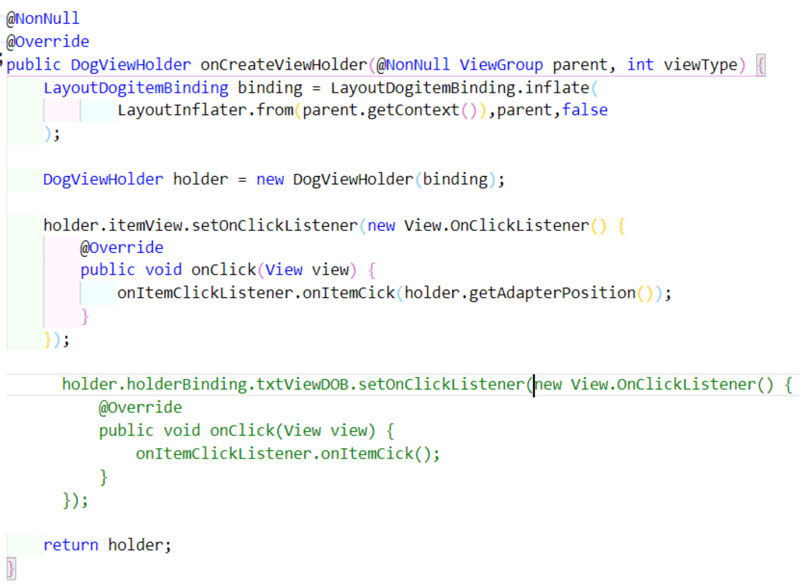

```java
public class DogViewHolder extends RecyclerView.ViewHolder {

    LayoutDogitemBinding holderBinding;

    public DogViewHolder(LayoutDogitemBinding binding) {

        super(binding.getRoot());
        holderBinding = binding; // vvimp
    }
}
```


```java
public interface OnItemClickListener{
    void onItemCick(int index);
    //image click, txtviewClick..... can be implemented
}
```

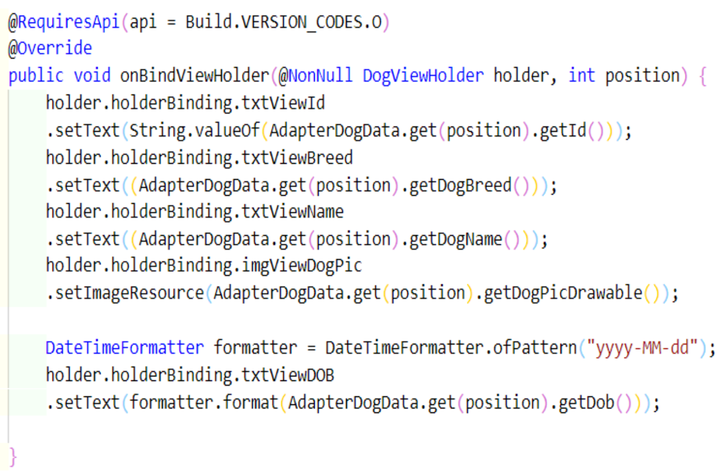
2)Adapter extension,create constructors


```java
public class DogAdapter extends RecyclerView.Adapter<DogAdapter.DogViewHolder> {

    List<Dog> AdapterDogData;
    OnItemClickListener onItemClickListener;
```

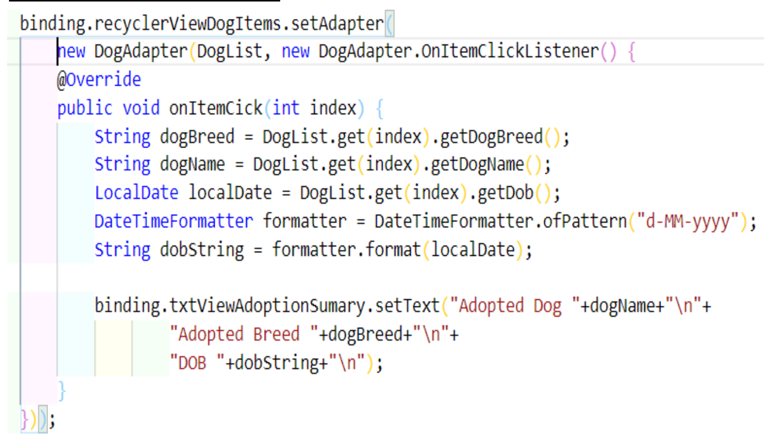
3)Oncreate


```java
@NonNull
@Override
public DogViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    LayoutDogitemBinding binding = LayoutDogitemBinding.inflate(
            LayoutInflater.from(parent.getContext()),parent,false
    );

    DogViewHolder holder = new DogViewHolder(binding);

    holder.itemView.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            onItemClickListener.onItemCick(holder.getAdapterPosition());
        }
    });

     holder.holderBinding.txtViewDOB.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View view) {
             onItemClickListener.onItemCick();
         }
     });

    return holder;
}
```

4)On Bind


```java
@RequiresApi(api = Build.VERSION_CODES.O)
@Override
public void onBindViewHolder(@NonNull DogViewHolder holder, int position) {
    holder.holderBinding.txtViewId
    .setText(String.valueOf(AdapterDogData.get(position).getId()));
    holder.holderBinding.txtViewBreed
    .setText((AdapterDogData.get(position).getDogBreed()));
    holder.holderBinding.txtViewName
    .setText((AdapterDogData.get(position).getDogName()));
    holder.holderBinding.imgViewDogPic
    .setImageResource(AdapterDogData.get(position).getDogPicDrawable());

    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");
    holder.holderBinding.txtViewDOB
    .setText(formatter.format(AdapterDogData.get(position).getDob()));

}
```

5)Main Activity On Click Display, <u>don't forget to add layoutManager</u>


```java
binding.recyclerViewDogItems.setAdapter(
    new DogAdapter(DogList, new DogAdapter.OnItemClickListener() {
    @Override
    public void onItemCick(int index) {
        String dogBreed = DogList.get(index).getDogBreed();
        String dogName = DogList.get(index).getDogName();
        LocalDate localDate = DogList.get(index).getDob();
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("d-MM-yyyy");
        String dobString = formatter.format(localDate);

        binding.txtViewAdoptionSumary.setText("Adopted Dog "+dogName+"\n"+
                "Adopted Breed "+dogBreed+"\n"+
                "DOB "+dobString+"\n");
    }
}));
```