Persistence

In computer science, persistence refers to the characteristic of state that outlives the process that created it.

Saving to Disk

- Use Gson to convert Object to JSON String
- Create OutputStrem from Context's `openFileOutput(String fileName, int mode)` method
- Use a Writer to write the JSON String to disk
- Close Writer

openFileOutput

Open a private file associated with this Context's application package for writing.

Retrieving Data from Disk

- Create InputStream from Context's `openFileInput(String fileName)` method
- Use a reader to read contents of file and create a JSON String
- Use Gson to convert JSON String to object(s)

openFileInput

Open a private file associated with this Context's application package for reading.

Code

Save

```
Car[] cars = {car1, car2, car3, car4};

        String carsString = sGson().toJson(cars, Car[].class);// sGson is a static instance of Gson
        Writer writer = null;

        try {
            OutputStream out = mContext.openFileOutput(mFileName, Context.MODE_PRIVATE);
            writer = new OutputStreamWriter(out);
            writer.write(peopleString);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (writer != null){
                writer.close();
            }
        }
```

Load

```
List<Car>cars = new ArrayList<Car>();

    BufferedReader reader = null;
    try{
        InputStream inputStream = mContext.openFileInput(mFileName);
        reader = new BufferedReader(new InputStreamReader(inputStream));
        StringBuilder jsonString = new StringBuilder();
        String line = null;
        while((line = reader.readLine()) != null){
            jsonString.append(line);
        }
        Car[] carArr = sGson().fromJson(jsonString.toString(), Car[].class);

    }catch (FileNotFoundException e){

    }finally {
        if (reader != null){
            reader.close();
        }
    }
    return carArr;
```

Refactor: Anonymous Inner Class -> Implementing Interface