```
package com.example.carpicker;

import android.os.*;
import org.json.*;
import java.util.*;
import okhttp3.*;

public class OkHttpHandler {

    /*
    Αυτός είναι ένας κατασκευαστής για την κλάση OkHttpHandler.
    Ορίζει μια αυστηρή πολιτική νημάτων για να επιτρέπει τις λειτουργίες δικτύου στο κύριο νήμα.
    Αυτό είναι απαραίτητο επειδή οι λειτουργίες δικτύου εκτελούνται συνήθως σε ξεχωριστά νήματα
    για να αποφευχθεί ο αποκλεισμός του νήματος διεπαφής χρήστη.
    Φαίνεται ότι σε αυτήν την περίπτωση, ο κώδικας επιτρέπει ρητά τις λειτουργίες δικτύου στο κύριο νήμα χρησιμοποιώντας το StrictMode.
     */
    public OkHttpHandler() {
        // Δημιουργεί ένα νέο αντικείμενο ThreadPolicy.Builder για να ορίσει την πολιτική για λειτουργίες δικτύου στο κύριο νήμα χρησιμοποιώντας το StrictMode.
        // .permitAll(): Αυτή η μέθοδος διαμορφώνει το ThreadPolicy.Builder ώστε να επιτρέπει όλες τις λειτουργίες δικτύου στο κύριο νήμα, ανεξάρτητα από τον πιθανό αντίκτυπό τους στην απόδοση ή την απόκριση.
        // .build(): Δημιουργεί το αντικείμενο ThreadPolicy με την καθορισμένη διαμόρφωση πολιτικής.
        StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();

        // Ορίζει το καθορισμένο ThreadPolicy για το τρέχον νήμα, επιτρέποντας λειτουργίες δικτύου στο κύριο νήμα.
        StrictMode.setThreadPolicy(policy);
    }

    /*
    Αυτή η μέθοδος λαμβάνει ένα url παραμέτρου συμβολοσειράς που αντιπροσωπεύει ένα τελικό σημείο διεύθυνσης URL.
    Εκτελεί ένα αίτημα HTTP POST στην καθορισμένη διεύθυνση URL και ανακτά τα δεδομένα απόκρισης.
    Τα δεδομένα απόκρισης αναμένεται να είναι σε μορφή JSON, που θα περιέχουν πληροφορίες επωνυμίας και μοντέλων.
    Τα δεδομένα JSON αναλύονται για την εξαγωγή της επωνυμίας και των μοντέλων και δημιουργείται και επιστρέφεται
    μια λίστα αντικειμένων CarBrand.
     */

    // Αυτή η μέθοδος λαμβάνει ένα url παραμέτρου συμβολοσειράς που αντιπροσωπεύει ένα τελικό σημείο
    // διεύθυνσης URL. Εκτελεί ένα αίτημα HTTP POST στην καθορισμένη διεύθυνση URL και ανακτά τα δεδομένα
    // απόκρισης. Επιστρέφει ένα αντικείμενο ArrayList<CarBrand>.
    ArrayList<CarBrand> populateDropDown(String url) throws Exception {
        //Αρχικοποιεί μια κενή ArrayList αντικειμένων CarBrand για την αποθήκευση των ανακτημένων δεδομένων.
```

```
        ArrayList<CarBrand> cbList = new ArrayList<>();

        // Δημιουργεί ένα αντικείμενο της κλάσης OkHttpClient, το οποίο είναι υπεύθυνο για την υποβολή
        // αιτημάτων HTTP.
        OkHttpClient client = new OkHttpClient().newBuilder().build();

        // Δημιουργεί ένα κενό σώμα αιτήματος χρησιμοποιώντας τη μέθοδο RequestBody.create().
        // Η μέθοδος MediaType.parse() χρησιμοποιείται για τον καθορισμό του τύπου μέσου του σώματος αιτήματος,
        // που σε αυτήν την περίπτωση είναι "κείμενο/απλό".
        RequestBody body = RequestBody.create("", MediaType.parse("text/plain"));

        // Δημιουργεί ένα αίτημα HTTP POST χρησιμοποιώντας το Request.Builder. Η διεύθυνση URL ορίζεται στο παρεχόμενο url
        // και η μέθοδος αιτήματος ορίζεται σε "POST". Το κενό σώμα αιτήματος που δημιουργήθηκε στο προηγούμενο βήμα
        // περιλαμβάνεται στο αίτημα.
        Request request = new Request.Builder().url(url).method("POST", body).build();

        // Στέλνει το αίτημα HTTP και λαμβάνει την απάντηση χρησιμοποιώντας το OkHttpClient.
        // Η μέθοδος execute() καλείται για να εκτελέσει τη σύγχρονη εκτέλεση της αίτησης.
        Response response = client.newCall(request).execute();

        // Ανακτά το σώμα απόκρισης ως συμβολοσειρά.
        String data = response.body().string();
        //System.out.println("My Response: " + data);
        try {
            // Δημιουργεί ένα JSONObject αναλύοντας τα δεδομένα απόκρισης που ανακτήθηκαν.
            JSONObject json = new JSONObject(data);

            // Ανακτά έναν επαναλήπτη πάνω από τα κλειδιά του JSONObject.
            Iterator<String> keys = json.keys();

            // Επαναλαμβάνεται πάνω από κάθε κλειδί στο JSONObject.
            while(keys.hasNext()) {
                // Ανακτά το επόμενο κλειδί από τον επαναλήπτη, που αντιπροσωπεύει την επωνυμία.
                String brand = keys.next();

                // Ανακτά την αντίστοιχη τιμή για το κλειδί επωνυμίας από το JSONObject και τη μετατρέπει σε συμβολοσειρά.
                String models = json.get(brand).toString();

                // Δημιουργεί ένα νέο αντικείμενο CarBrand χρησιμοποιώντας την επωνυμία και μοντελοποιεί τιμές και το προσθέτει στη cbList.
                cbList.add(new CarBrand(brand, models));
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }

        // Επιστρέφει τη συμπληρωμένη ArrayList<CarBrand> που περιέχει τα δεδομένα επωνυμίας και μοντέλων που ανακτήθηκαν.
        return cbList;
```

```
    }

    /*
    Αυτή η μέθοδος λαμβάνει ένα url παραμέτρου συμβολοσειράς που αντιπροσωπεύει ένα τελικό σημείο διεύθυνσης URL.
    Εκτελεί ένα αίτημα HTTP POST στην καθορισμένη διεύθυνση URL για την καταγραφή ορισμένων δεδομένων.
    Το αίτημα υποβάλλεται χωρίς κανένα φορέα αιτήματος. Η απάντηση από τον διακομιστή καταγράφεται στην κονσόλα.
     */
    public void logHistory(String url) throws Exception {
        // Δημιουργεί αντικείμενο της κλάσης OkHttpClient, η οποία είναι υπεύθυνη για την υποβολή αιτημάτων HTTP.
        OkHttpClient client = new OkHttpClient().newBuilder().build();

        // Δημιουργεί ένα κενό σώμα αιτήματος χρησιμοποιώντας τη μέθοδο RequestBody.create().
        // Η μέθοδος MediaType.parse() χρησιμοποιείται για τον καθορισμό του τύπου μέσου του σώματος αιτήματος,
        // που σε αυτήν την περίπτωση είναι "κείμενο/απλό".
        RequestBody body = RequestBody.create("", MediaType.parse("text/plain"));

        // Δημιουργεί ένα αίτημα HTTP POST χρησιμοποιώντας το Request.Builder. Η διεύθυνση URL ορίζεται στο παρεχόμενο url
        // και η μέθοδος αιτήματος ορίζεται σε "POST". Το κενό σώμα αιτήματος που δημιουργήθηκε στο προηγούμενο βήμα
        // περιλαμβάνεται στο αίτημα.
        Request request = new Request.Builder().url(url).method("POST", body).build();

        // Στέλνει το αίτημα HTTP και λαμβάνει την απάντηση χρησιμοποιώντας το OkHttpClient.
        // Η μέθοδος execute() καλείται για να εκτελέσει τη σύγχρονη εκτέλεση της αίτησης.
        Response response = client.newCall(request).execute();

        // Καταγράφει το αντικείμενο απόκρισης στην κονσόλα. Το αντικείμενο απόκρισης περιέχει πληροφορίες σχετικά με την
        // απόκριση HTTP, όπως τον κωδικό κατάστασης, τις κεφαλίδες και το σώμα.
        System.out.println("My Response: " + response);
    }
}
```