Voici la méthode showList :

```
private void showList() {
    runOnUiThread(() -> {
        adapter.notifyDataSetChanged();
    });
}
```

Voici la méthode loadTraining :

```
public static ArrayList<TrainingBean> loadTraining () throws Exception {
    Gson g = new Gson();
    String json = OkHttpUtils.sendGetOkHttpRequest( url: "http://176.191.236.103/GetTraining");
    System.out.println("json : " + json);
    ArrayList<TrainingBean> listTraining = g.fromJson(json, new TypeToken<ArrayList<TrainingBean>>() {
    }.getType());
    return listTraining;
}
```

Voici la méthode sendGetOkHttpRequest :

```
public static String sendGetOkHttpRequest(String url) throws Exception {
    Log.w( tag: "tag",  msg: "url : " + url);
    OkHttpClient client = new OkHttpClient();
    //Création de la requete
    Request request = new Request.Builder().url(url).build();
    //Execution de la requête
    Response response = client.newCall(request).execute();
    System.out.println("reponse : " + response);
    //Analyse du code retour
    if (response.code() < 200 || response.code() >= 300) {
        throw new Exception("Réponse du serveur incorrect : " + response.code());
    }
    else {
        return response.body().string();
    }
}
```

Ces extraits de code permettent de visualiser comment depuis l'application cliente nous pouvons récupérer les données de notre serveur et les utiliser pour les afficher sur un élément graphique de l'application cliente.