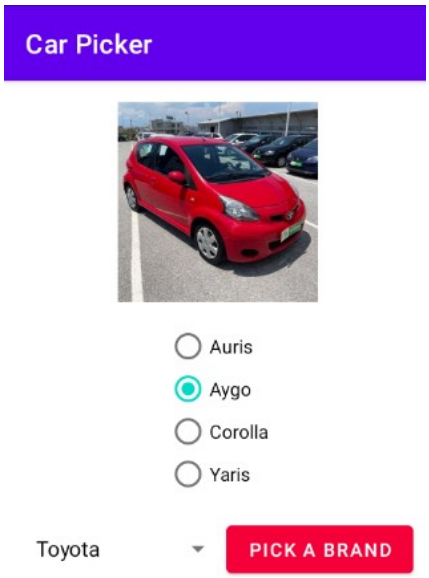
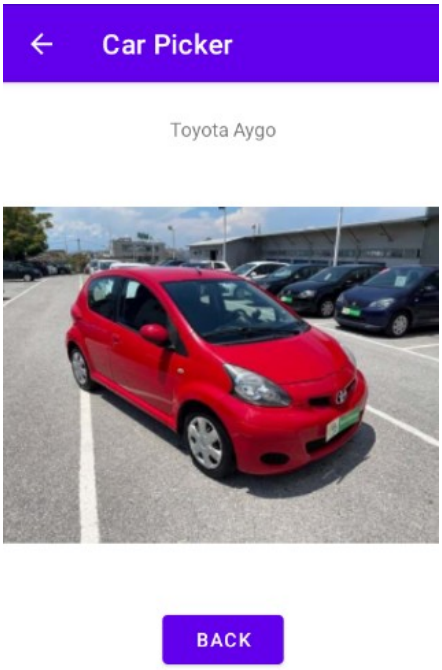
Walkthrough: Διάλεξη 10 – Χρήση πολυμέσων

Στόχος μας είναι η δημιουργία μίας εφαρμογής η οποία θα συνδέεται σε μία βάση δεδομένων μέσω υπηρεσιών και http, θα ανακτά δεδομένα, μεταξύ των οποίων και πολυμέσων, και θα τα εμφανίζει σε ένα νέο activity Χρησιμοποιώντας intents.

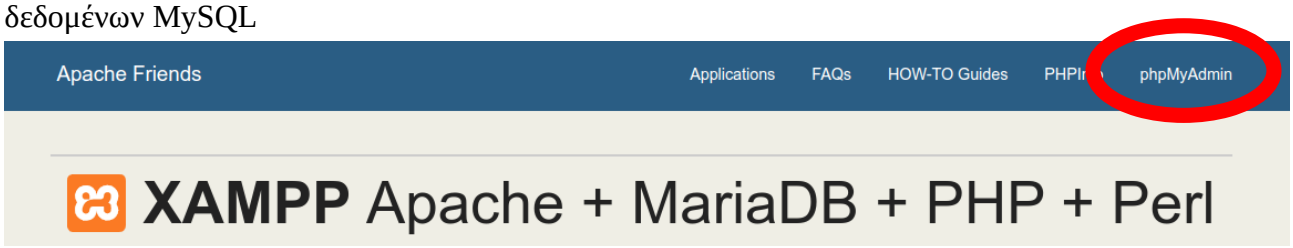
Το συγκεκριμένο μάθημα επεκτείνει αυτό της Διάλεξης 8. Για το λόγο αυτό θα βασιστούμε στην ενδεικτική επίλυση του συγκεκριμένου μαθήματος και θα συνεχίσουμε από εκεί.





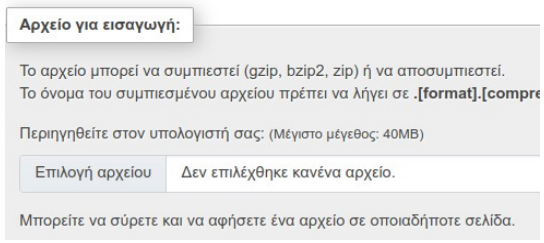
Εισαγωγή δεδομένων και σχήματος βάσης δεδομένων

Αφού εκκινήσουμε το xampp, και συγκεκριμένα τους διακομιστές Apache και MySQL, μεταβαίνουμε στη διεύθυνση http://localhost/ και επιλέγουμε το σύνδεσμο πάνω δεξιά, phpMyAdmin, ώστε να μεταβούμε στο ομώνυμο διαδικτυακό εργαλείο διαχείρισης βάσεων δεδομένων MySQL



Με την έναρξη του phpMyAdmin, επιλέγουμε "Βάσεις Δεδομένων" και στην επόμενη σελίδα που θα ανοίξει

επιλέγουμε **"Εισαγωγή"**. Στη σελίδα που θα ανοίξει, στο πεδίο "Αρχείο για εισαγωγή" επιλέγουμε το αρχείο **multimediacars.sql.txt**, που έχουμε μεταφέρει στον υπολογιστή μας από το openeclass (Αρχικός κατάλογος » Διάλεξη 10 – Χρήση Πολυμέσων » SQL Δημιουργίας ΒΔ). Έπειτα, επιλέγουμε το πλήκτρο **"Εισαγωγή"** από το κάτω μέρος της σελίδας. Το αποτέλεσμα είναι η δημιουργία της βάσης δεδομένων cars, που αποτελείται από δύο πίνακες: τον πίνακα history, όπου θα καταγράφεται το ιστορικό ενεργειών στην εφαρμογή, και τον πίνακα models, ο οποίος περιέχει πληροφορίες για τα οχήματα που θα εμφανίζονται στην εφαρμογή.



Δημιουργία υπηρεσιών

Επόμενο βήμα μας είναι η δημιουργία μίας επιπλέον υπηρεσίες σε http, οι οποία θα επικοινωνεί μέσω http request με την εφαρμογή, μεταφέροντας δεδομένα σε μορφή json και θα διαβάζει από τη βάση δεδομένων τις πληροφορίες πολυμέσων. Θα χρησιμοποιήσουμε PHP για τη δημιουργία υπηρεσίας και πάλι. Στο φάκελο htdocs αποσυμπιέζουμε τον φάκελο carsDBServices, τον μετονομάζουμε σε **multimediaDBServices** και μέσα σε αυτόν δημιουργούμε ένα αρχείο getMedia.php για τη νέα υπηρεσία.

```
<?php
        $data = array();

        $host="localhost";
        $uname="root";
        $pass="";
        $dbname="cars";

        $dbh = mysqli_connect($host,$uname,$pass) or die("cannot connect");
        mysqli_select_db($dbh, $dbname);

        $sql = "SELECT brand, GROUP_CONCAT(model) AS grouped_models,
GROUP_CONCAT(image) as images FROM models GROUP BY brand";
        $result = mysqli_query($dbh, $sql);
        while ($row = mysqli_fetch_array($result)) {
```

```
                $nested_data = array();
                $nested_data['grouped_models']= $row['grouped_models'];
                $nested_data['images']= $row['images'];
                $data[$row['brand']] = $nested_data;
        }
        mysqli_close($dbh);
        header("Content-Type: application/json");
        echo json_encode($data);
?>
```

ΠΡΟΣΟΧΗ! Πιθανόν να χρειαστεί να αλλάξετε τα στοιχεία χρήστη (`$uname`,`$pass`) για να λειτουργήσουν οι υπηρεσίες.

Για να ελέγξουμε ότι λειτουργούν οι υπηρεσίες, επισκεπτόμαστε είτε από browser είτε με χρήση Postman το http://localhost/multimediaDBServices/getMedia.php

Αν όλα λειτουργούν σωστά, θα πρέπει να εμφανίζεται το παρακάτω:


```
{"Nissan":{"grouped_models":"Sunny","images":"https:\/\/www.autogreeknews.gr\/
wp-content\/uploads\/2020\/09\/Nissan-Sunny-1981-B11-1.jpg"},"Toyota":
{"grouped_models":"Auris,Aygo,Corolla,Yaris","images":"https:\/\/
upload.wikimedia.org\/wikipedia\/commons\/6\/6d\/
Toyota_Auris_5dr._front.jpg,https:\/\/www.stock-center.gr\/sites\/default\/
files\/styles\/car_listing\/public\/car\/2021-06\/AYGO%20AUTO%201.jpg?itok=llS6-
EvV,https:\/\/s1.cdn.autoevolution.com\/images\/gallery\/TOYOTACorolla-5doors--
623_5.jpg,https:\/\/www.autogreeknews.gr\/wp-content\/uploads\/2022\/03\/Toyota-
Yaris-GR-Sport.jpg"},"VW":{"grouped_models":"Golf,Polo","images":"https:\/\/
upload.wikimedia.org\/wikipedia\/commons\/thumb\/6\/65\/
Volkswagen_Golf_VIII_IMG_3829.jpg\/1920px-
Volkswagen_Golf_VIII_IMG_3829.jpg,https:\/\/www.allcarphotos.net\/wp-content\/
uploads\/2020\/09\/VW-Polo-2021-01.jpg"}}
```


Δημιουργία frontend

Για τη δημιουργία του frontend θα βασιστούμε σε προηγούμενη υλοποίηση την οποία βρίσκουμε στην εξής τοποθεσία:

Αρχικός κατάλογος » Διάλεξη 08 - Κλήση απομακρυσμένων λειτουργιών (DB) » Ενδεικτική Λύση

Προετοιμασία

Σε αυτή την υλοποίηση θα προσπελάσουμε το διαδίκτυο και θα χρησιμοποιήσουμε μία εξωτερική βιβλιοθήκη για να πραγματοποιήσουμε τις κλείσεις http. Για το λόγο αυτό θα πρέπει να προχωρήσουμε σε αλλαγές στα παρακάτω αρχεία:

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.carpicker">

    <uses-permission android:name="android.permission.INTERNET" />
```

```
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.CarPicker"
        android:networkSecurityConfig="@xml/network_security_config"
        android:usesCleartextTraffic="true">

        <meta-data
            android:name="com.google.android.actions"
            android:resource="@xml/network_security_config" />

        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

network_security_config.xml

Δημιουργία αρχείου network_security_config.xml. Δημιουργούμε φάκελο xml εντός του φακέλου res και εντός του φακέλου xml δημιουργούμε αρχείο network_security_config.xml. Δεξί κλικ: New-→ Folder → XML Resources Folder

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="true">
        <domain includeSubdomains="true">192.168.1.69</domain>
    </domain-config>
</network-security-config>
```

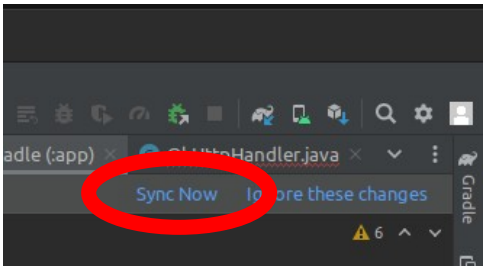
Προσοχή! Η διεύθυνση IP που βλέπετε παραπάνω θα πρέπει να αντικατασταθεί από τη διεύθυνση ip του υπολογιστή στο οποίο τρέχουν οι υπηρεσίες στις οποίες θα συνδεθούμε. Αν τρέχουν στον ίδιο υπολογιστή, βρίσκουμε την IP διεύθυνσή του (π.χ. μέσω ipconfig από γραμμή εντολών).

build.gradle

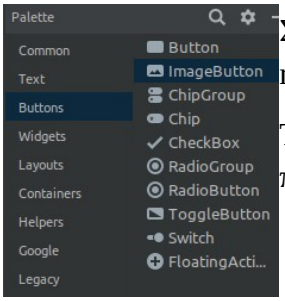
Στο build.gradle (Module App) εκτός από το okhttp, κάνουμε την παρακάτω προσθήκη:

```
implementation 'com.squareup.picasso:picasso:2.5.2'
```

Δεν ξεχνάμε να επιλέξουμε Sync Now.



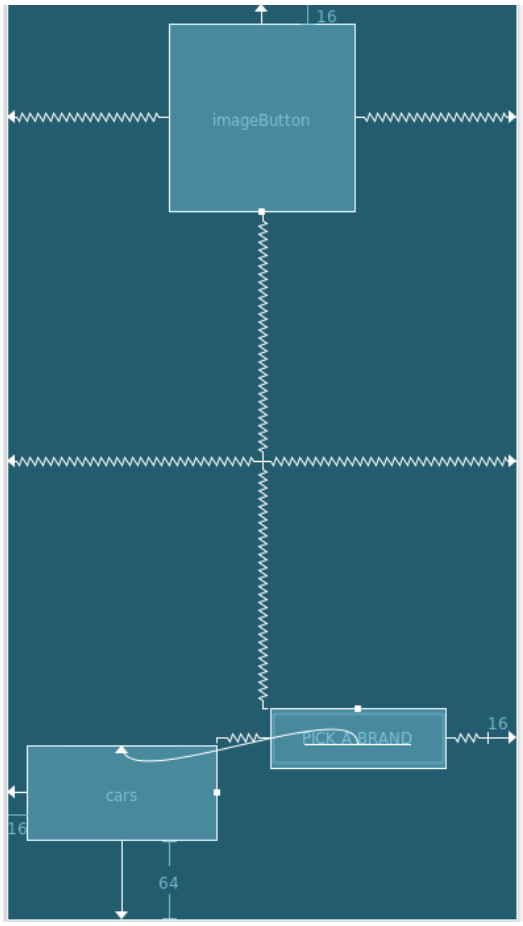
Δημιουργία layouts



Στο activity_main.xml προσθέτουμε ένα ImageButton ανάμεσα στο radiogroup και το πάνω πλαίσιο και ρυθμίζουμε τους περιορισμούς ανάλογα.

Το ImageButton θα έχει το ευφάνταστο id imageButton, θα έχει ύψος και πλάτος 150dp και background #FFFFFF.

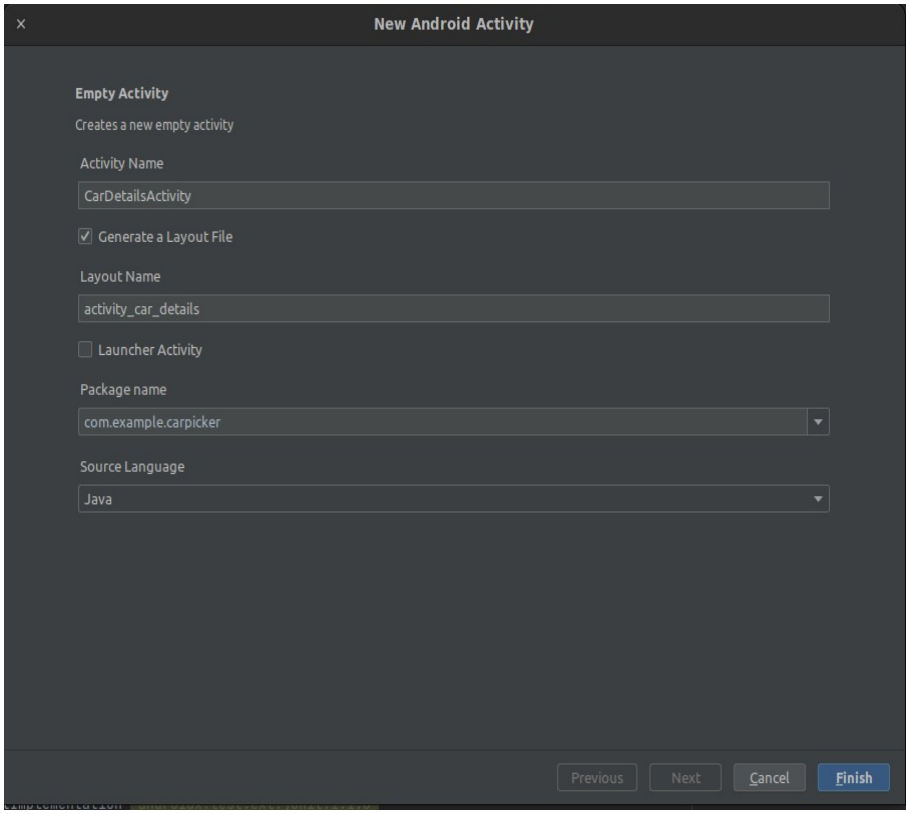
Το συνολικό layout μπορείτε να το δείτε στην παρακάτω εικόνα:



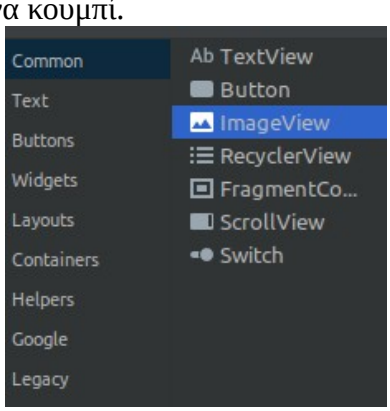
Επιπλέον, θα προσθέσουμε ένα νέο activity το οποίο θα συνδέσουμε με το υφιστάμενο.

res > xml > δεξί κλικ > New > Activity > Empty Activity

και αλλάζουμε το όνομα του layout (Layout Name) σε activity_car_details. Το Activity Name συμπληρώνεται μόνο του από το όνομα του layout. Στη συνέχει επιλέγουμε Finish.



Ανοίγουμε το νέο activity που φτιάξαμε (activity_car_details.xml) και εισάγουμε ένα TextView, ένα ImageView και, κάτω από αυτό, ένα κουμπί.



Το textView θα λέγεται fullNameTXT, θα έχει απόσταση 24dp από το πάνω περιθώριο και θα έχει περιορισμούς πάνω, δεξιά και αριστερά.

Το imageView θα λέγεται myImage θα έχει περιορισμούς: πάνω με το textView, κάτω με το buttonView, καθώς και αριστερά και δεξιά. Θα έχει layout_width: match_constraint (0dp) και layout height 550dp.

Το κουμπί θα λέγεται backBtn και θα είναι wrap_content τόσο οριζόντια όσο και κατακόρυφα. Θα έχει απόσταση 24dp από το κάτω περιθώριο και θα έχει περιορισμούς κάτω, δεξιά και αριστερά.

Η διάταξη που θα υπάρχει θα είναι η παρακάτω:

Δημιουργία κλάσεων και κώδικα Java

Θα δημιουργήσουμε μία νέα κλάση και θα τροποποιήσουμε αυτές που υπάρχουν ήδη:

Δημιουργία νέας κλάσης: Media.java

```
package com.example.carpicker;

class Media{
    private String image;

    public Media(String image) {
        this.image = image;
    }

    public String getImage() {
        return image;
    }

}
```

Η παρακάτω κλάση δημιουργήθηκε αυτόματα με την εισαγωγή του νέου activity. Εμείς θα συμπληρώσουμε τον κώδικά της. Σημεία που πρέπει να προσέξουμε είναι η λήψη intent από το προηγούμενο activity καθώς και η εμφάνιση της εικόνας με Picasso.

CarDetailsActivity:

```
package com.example.carpicker;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;

import com.squareup.picasso.Picasso;

public class CarDetailsActivity extends AppCompatActivity {
    ImageView myImage;
    Button returnButton;
    TextView fname;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_car_details);
        myImage = (ImageView) findViewById(R.id.myImage);
        returnButton = findViewById(R.id.backBtn);
        fname = findViewById(R.id.fullNameTXT);

        myImage.setVisibility(View.VISIBLE);
        Intent intent = getIntent();
        String imageUri = intent.getStringExtra("IMAGE");
        String full_name = intent.getStringExtra("FULL_NAME");

        fname.setText(full_name);
        Picasso.with(getApplicationContext()).load(Uri.parse(imageUri)).resize(300,
0).into(myImage);

        returnButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                finish();
            }
        });

    }
}
```

CarBrand.java

Προσθέτουμε ένα attribute:

```
private HashMap<String, Media> media = new HashMap<>();
```

Προσθέτουμε τον παρακάτω κατασκευαστή ώστε να μπορούμε να εισάγουμε το media:

```
public CarBrand(String brand, String models, String images) {
    name = brand;
    this.models = Arrays.asList(models.split(","));
    String[] imageArray = images.split(",");
    for (int i=0; i<this.models.size(); i++) {
```

```
            media.put(this.models.get(i),new Media(imageArray[i]));
        }
    }
```

Τέλος, προσθέτουμε τον παρακάτω getter:

```
public Media getMedia(String key){
    return this.media.get(key);
}
```

CarBrandList.java

Αλλάζουμε το url ώστε να δείχνει στη νέα υπηρεσία που φτιάξαμε:

```
String url= "http://"+ip+"/multimediaDBServices/getMedia.php";
```

Προσθέτουμε την παρακάτω μέθοδο:

```
/**
 * Method added to return CarBrand.Media object when model is provided
 * */
public Media lookup(String brand, String model) {
    for (int i = 0; i < cbList.size(); i++) {
        if (cbList.get(i).hasName(brand)) {
            return cbList.get(i).getMedia(model);
        }

    }
    return null;
}
```

MainActivity.java

Πραγματοποιούμε τροποποιήσεις στο MainActivity.java ώστε να χρησιμοποιήσουμε τη νέα κλάση:

Προσέχουμε το παρακάτω (Προσοχή! Βάζετε τη διεύθυνση του διακομιστή όπου εκτελούνται οι υπηρεσίες):

```
private final String myIP = "192.168.1.69";
```

Προσθέτουμε τα παρακάτω attributes στην κλάση:

```
private ImageButton myImage;
private String imageUri;
private String full_name;
```

Στην onCreate, πριν το spinner, δίνουμε τιμή στο ImageButton:

```
myImage = findViewById(R.id.imageButton);
```

Στο τέλος της onCreate προσθέτουμε τα παρακάτω για να ενεργοποιήσουμε το κουμπί:

```
myImage.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        sendMessage(view);
```

```
    }
});
```

Προσθέτω την παρακάτω μέθοδο:

```
public void sendMessage(View view){
    Intent intent = new Intent(this, CarDetailsActivity.class);
    intent.putExtra("IMAGE", imageUri);
    intent.putExtra("FULL_NAME", full_name);
    startActivity(intent);
}
```

Στην onCheckedChange προσθέτω τα παρακάτω ανάμεσα στο String url και στο try:

```
imageUri = cbl.lookup(brand, rb.getText().toString()).getImage();
full_name = brand + " " +  rb.getText().toString();
myImage.setVisibility(View.VISIBLE);
System.out.println(imageUri);
Picasso.with(getApplicationContext()).load(Uri.parse(imageUri)).resize(300, 0).into(myImage);
```

OkHttpHandler.java:

Αντικαθιστώ την populateDropDown ώστε να μπορώ να διαβάζω από το json αρχείο που λαμβάνω από την υπηρεσία και τις πληροφορίες εικόνας.

```
/**
 * Changed this method to also parse image urls
 * Warning! URLs should not contain commas!
 * */
ArrayList<CarBrand> populateDropDown(String url) throws Exception {
    ArrayList<CarBrand> cbList = new ArrayList<>();
    OkHttpClient client = new OkHttpClient().newBuilder().build();
    RequestBody body = RequestBody.create("", MediaType.parse("text/plain"));
    Request request = new Request.Builder().url(url).method("POST", body).build();
    Response response = client.newCall(request).execute();
    String data = response.body().string();
    System.out.println("My Response: " + data);
    try {
        JSONObject json = new JSONObject(data);
        Iterator<String> keys = json.keys();
        while(keys.hasNext()) {
            String brand = keys.next();
            String models = json.getJSONObject(brand).getString("grouped_models").toString();
            String images = json.getJSONObject(brand).getString("images").toString();
            cbList.add(new CarBrand(brand, models, images));
        }
    } catch (JSONException e) {
        e.printStackTrace();
    }
    return cbList;
}
```

Extra:

Στο manifest.xml προσθέτω το παρακάτω στο πεδίο Application. Το αποτέλεσμα είναι να δημιουργηθεί backbutton στο πλαίσιο της εφαρμογής:

```
android:parentActivityName=".MainActivity"
```