

Département d'Informatique, L3, S6
Cours : Développement Mobile

Accès à une base de données distante
en utilisant le **REST (WEB)**



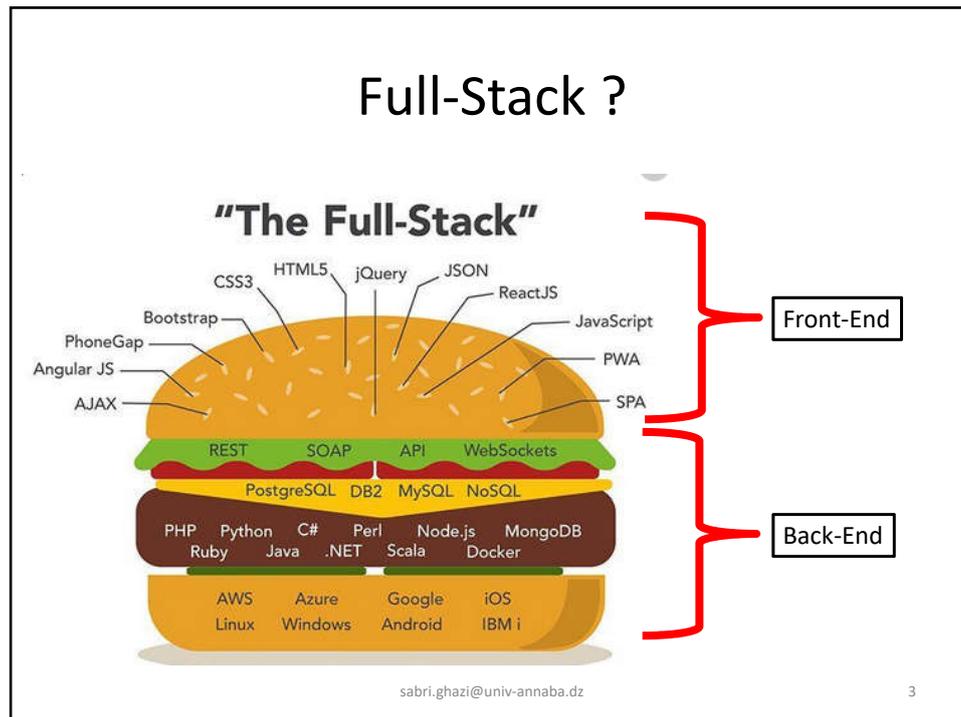
Université Badji Mokhtar Annaba

Email: sabri.ghazi@univ-annaba.dz

Accès à une base de données distante

- L'architecture d'une solution logicielle est généralement représentée sous forme d'une pile de couches :
 - Couche présentation , appelée aussi **Front-end**
 - **Couche Métier** et Persistance , appelée aussi **Back-end**.
- Il existent des technologies qui peuvent être utilisées dans chaque couche.

Full-Stack ?



Accès à une base de données via une API REST

- La plupart des applications mobile et Web utilisent une base de données centralisée dans un serveur.
- Comment accéder à une base de données distante (se trouve sur un serveur dans le réseau).
- L'accès directement à la base de données n'est pas possible, on ne peut pas stocker les mots de passe dans l'application (elle peut être décompilée).
 - **Solution** : Utiliser un **REST API**
 - C'est quoi ? : C'est un program qui permet d'effectuer des traitements et s'exécute dans un serveur et renvoi le résultat.

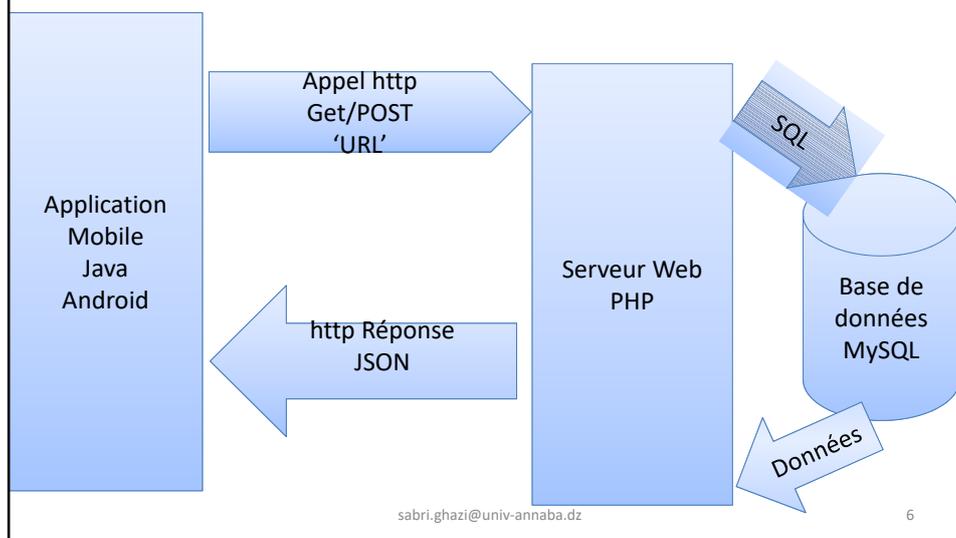
REST API

- **REpresentational State Transfer**
- On peut le considérer comme étant une fonction qui s'exécute dans le serveur.
- Invoqué par n'importe quel programme (mobile, desktop ou même un autre web service).
- Ils sont écrits en un langage **back-end** exemple : PHP, Python, Java, GO, Node.js etc

sabri.ghazi@univ-annaba.dz

5

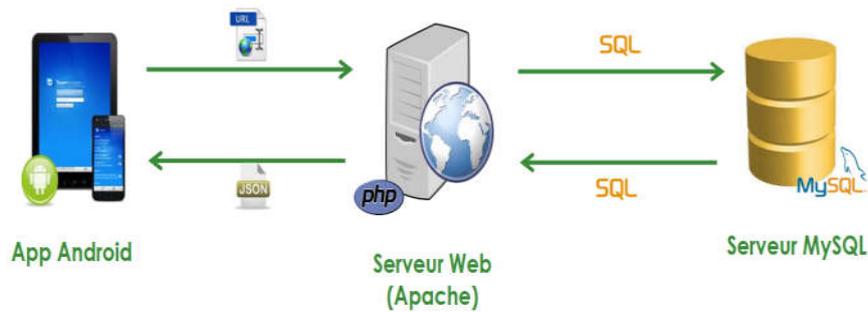
Schéma d'un accès **REST**



sabri.ghazi@univ-annaba.dz

6

Schéma d'un accès **REST**



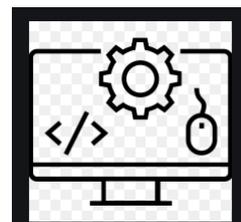
(crédit : Dr. CHAOUCHE A. C. , Université de Constantine).

sabri.ghazi@univ-annaba.dz

7

Exemple d'une application Mobile Android qui récupère des données d'une base de données distante

- Etapes **Back-End**:
 - Installation du serveur WAMP.
 - Création de la base de donnée
 - Création du code PHP Back-End.
- Etapes **Front-End**
 - Création de l'application Android
 - Création de l'activité
 - Création de la classe **AsyncTask**
 - Lecture des résultats et affichage,



sabri.ghazi@univ-annaba.dz

8

Etapes **Back-end**

Installation du serveur

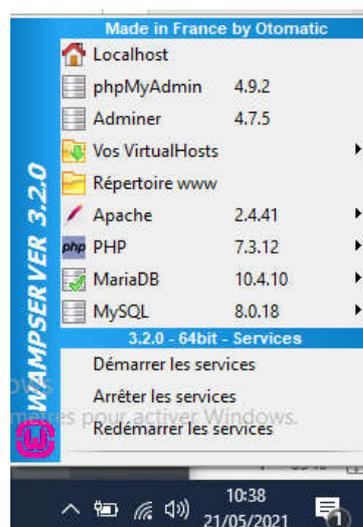
- **WAMP** est un ensemble d'outils qui permet de transformer votre machine en un serveur , il inclut
 - Un serveur **Apache**
 - Un serveur **Php**
 - Un serveur de base de données **MySQL**
 - Une application pour gérer **MySQL** nommé **PhpMyAdmin**
- Lien de téléchargement
 - <https://www.wampserver.com/>

sabri.ghazi@univ-annaba.dz

9

Etapes **Back-end**

Installation du serveur



sabri.ghazi@univ-annaba.dz

10

Etapas Back-end

Création de la base de données:

The screenshot shows the phpMyAdmin interface. On the left, the 'Structure de la table' (Table Structure) view is selected, showing the columns: code, nom, and population. A red arrow points from the label 'La Structure de la table' to this view. On the right, the 'Affichage des lignes' (Display Rows) view is selected, showing the data for the 'villes' table. A red bracket groups the data rows, with a label 'Le contenu de la table' pointing to it.

code	nom	population
23000	Annaba	50000
16000	Alger	100000
25000	Constantine	90000
31000	Oran	70000
36000	Taref	10000
10000	Adrar	4000
24000	Guelma	60000
04000	Ain Baïda	75000
05000	Batna	75000

sabri.ghazi@univ-annaba.dz 11

Etapas Back-end

Le code **php** en Back-End

- Cette table contient des informations concernant les villes algériennes:
 - Le nom, le code postal et la population.
- Premièrement on doit écrire un script **AfficherVilles.php** qui permet de lister les villes et leurs informations.
- Ce fichier est placé dans le serveur dans un répertoire nommé « **DevMob** »

Etapes Back-end

Le programme **AfficherVilles.php** qui permet de transformer les données

```

1  <?php
2  // Se connecter à la base de données.
3  $connection=mysqli_connect("localhost","root","","tpdevmob");
4  // envoyer la requetes SQL.
5  $response=mysqli_query($connection,"SELECT * FROM villes");
6
7  //Lire le résultat
8  $resultat=array();
9  while($data=mysqli_fetch_assoc($response))
10 {
11     array_push($resultat,$data);
12 }
13 //fermer la connection.
14 mysqli_close($connection);
15 header('Content-Type:application/json');
16 //envoi du résultat en JSON.
17 echo json_encode($resultat);
18 http_response_code(200);
19 ?>

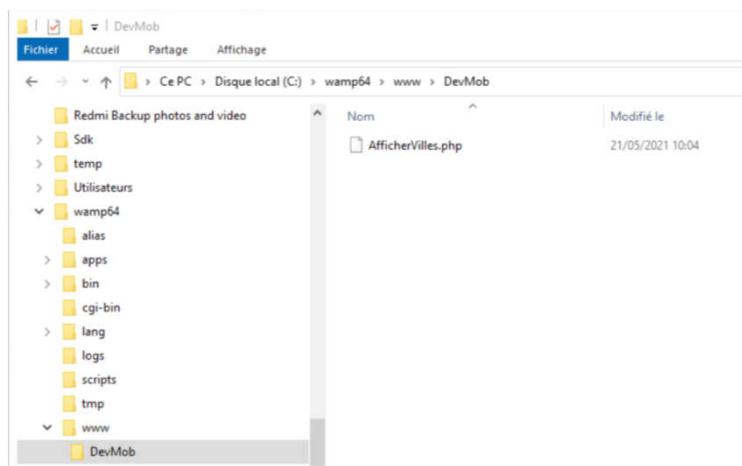
```

sabri.ghazi@univ-annaba.dz

13

Etapes Back-end

Configuration



sabri.ghazi@univ-annaba.dz

14

Etapas Back-end

Teste

- Si on lance la requête http dans notre navigateur web
- <http://localhost/DevMob/AfficherVilles.php>
- On obtient le résultat en format **JSON**

```
[
  {"code":"23000","nom":"Annaba","population":"50000"},
  {"code":"16000","nom":"Alger","population":"100000"},
  {"code":"13000","nom":"Telemcen","population":"14000"},
  {"code":"36000","nom":"Taref","population":"40000"},
  {"code":"10000","nom":"Adrar","population":"30000"},
  {"code":"25000","nom":"Constantine","population":"90000"}
]
```

sabri.ghazi@univ-annaba.dz

15

Etapas Back-end

Configuration

The screenshot shows a web browser displaying the JSON output of a PHP script. The browser address bar shows the URL `localhost/DevMob/AfficherVilles.php`. The JSON data is displayed in a structured view with expandable items for each city:

```

JSON  Données brutes  En-têtes
Enregistrer Copier Tout réduire Tout développer Filtre le JSON
0:
  code: "23000"
  nom: "Annaba"
  population: "50000"
1:
  code: "16000"
  nom: "Alger"
  population: "100000"
2:
  code: "13000"
  nom: "Telemcen"
  population: "14000"
3:
  code: "36000"
  nom: "Taref"
  population: "40000"
4:
  code: "10000"
  nom: "Adrar"
  population: "30000"
5:
  code: "25000"
  nom: "Constantine"
  population: "90000"

```

5

Configuration **Front-end**

- Il faut vérifier que la machine client (le téléphone) arrive à communiquer avec le serveur.
- L'URL <http://localhost/DevMob/AfficherVilles.php>
- Dans l'émulateur elle devient
 - <http://10.0.2.2/DevMob/AfficherVilles.php>
 - L'émulateur peut communiquer avec la machine de développement en utilisant cette adresse.
- Si vous utiliser un téléphone réel,
 - <http://192.168.0.180/DevMob/AfficherVilles.php>
 - il faut mettre l'adresse IP du serveur, utilisé **IPConfig** pour savoir l'adresse IP de votre machine.
 - il faut que les deux font partie du même réseau

sabri.ghazi@univ-annaba.dz

17

Configuration **Front-End**

-lancer la ligne de commande et après lancer **IPConfig** pour savoir l'adresse **IP** de votre

```

ca) Sélection Administrateur: C:\WINDOWS\system32\cmd.exe
Statut du média. . . . . : Média déconnecté
Suffixe DNS propre à la connexion. . . . . : univ-annaba.dz

Carte Ethernet VirtualBox Host-Only Network :
Suffixe DNS propre à la connexion. . . . . :
Adresse IPv6 de liaison locale. . . . . : fe80::b116:8111:860c:f20e%13
Adresse d'autoconfiguration IPv4. . . . . : 169.254.242.14
Masque de sous-réseau. . . . . : 255.255.0.0
Passerelle par défaut. . . . . :

Carte réseau sans fil Connexion au réseau local* 1 :
Statut du média. . . . . : Média déconnecté
Suffixe DNS propre à la connexion. . . . . :

Carte réseau sans fil Connexion au réseau local* 2 :
Statut du média. . . . . : Média déconnecté
Suffixe DNS propre à la connexion. . . . . :

Carte réseau sans fil Wi-Fi :
Suffixe DNS propre à la connexion. . . . . :
Adresse IPv6 de liaison locale. . . . . : fe80::783d:568:2016:c2c8%17
Adresse IPv4. . . . . : 192.168.1.7
Masque de sous-réseau. . . . . : 255.255.255.0
Passerelle par défaut. . . . . : 192.168.1.1
  
```

sabri.ghazi@univ-annaba.dz

18

Configuration Front-End

Pour tester que tout fonctionne bien,
On peut ouvrir l'URL dans le navigateur du téléphone
Comme illustré dans la figure



sabri.ghazi@univ-annaba.dz

19

Front-End

L'application Android

- On cherche
 - À lancer une requête http, pour lancer le web service.
 - Récupérer le résultat.
 - Affiche ce résultat dans l'interface utilisateur.
- On aura besoin de
 - Donner les permissions a l'application pour se connecter.
 - Une classe Ville.java (contenant les attributs de même nom et type que la table de base de données).
 - écriture du programme qui lance la requête et récupère le résultat.

sabri.ghazi@univ-annaba.dz

20

Front-End

Donner les permissions

- Il faut modifier le fichier Manifest.xml de l'application afin qu'elle puisse se connecter à une machine distante.
- Si non :**java.lang.SecurityException: Permission denied (missing INTERNET**

```

1 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
2   package="com.androiddev.listviewjson"
3   android:versionCode="1"
4   android:versionName="1.0" >
5   ...
6   <uses-permission android:name="android.permission.INTERNET" />
7   <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
8   ...
9 </manifest>

```

sabri.ghazi@univ-annaba.dz 21

Front-End

Autoriser l'utilisation du HTTP au lieu de HTTPS

- Pour des mesures de sécurité, les version récente d'Android n'autorise pas a une application de faire des communication en Texte clair (non crypté), toute communication doit être en **Https**.
- Pour simplifier cette démonstration, on va utiliser juste une communication **http**.
- Cette autorisation se fait au niveau du fichier Manifest.xml comme suit :
- Dans la balise application on utilise l'attribut

```
android:usesCleartextTraffic="true"
```

sabri.ghazi@univ-annaba.dz

22

Front-End Ville.java

```
public class Ville {
    String code;
    String nom;
    int population;

    public Ville(String code, String nom, int population) {
        this.code = code;
        this.nom = nom;
        this.population = population;
    }

    public String getCode() {
        return code;
    }

    public void setCode(String code) {
        this.code = code;
    }

    public String getNom() {
        return nom;
    }

    public void setNom(String nom) {
        this.nom = nom;
    }

    public int getPopulation() {
        return population;
    }

    public void setPopulation(int population) {
        this.population = population;
    }
}
```

sabri.ghazi@univ-annaba.dz

23

Front-End

Dans notre application Android

- On doit développer la classe qui permet de récupérer les résultats d'un web service.
- Cette classe doit hériter la classe **AsyncTask**.
- L'invocation et la récupération des résultats d'un web service se fait en arrière plan, c'est à dire sans bloquer le Thread principal (Celui de l'interface utilisateur).

sabri.ghazi@univ-annaba.dz

24

Front-End

La classe **AsyncTask**

- Elle permet de lancer des traitements **asynchrone**, c'est-à-dire dans un Thread autre que le Thread principal.
 - Pour ne pas bloquer l'interface utilisateur.
- Elle possède la méthode **doInBackground** :
 - fait le traitement en arrière plan.
- Est une méthode **onPostExecute** :
 - qui s'exécute après la fin du traitement. Elle est utilisé par récupérer le résultat.

sabri.ghazi@univ-annaba.dz

25

```

public class ListerVilleNom extends AsyncTask<String, Integer, String> {
    ProgressDialog dialog;
    public static DisplayVille context;

    protected void onPreExecute() {
        dialog = ProgressDialog.show(context, "Please wait", "Loading Data from server");
        dialog.setCancelable(true);
    }

    @Override
    protected String doInBackground(String... params) {
        HttpURLConnection urlConnection = null;
        BufferedReader reader = null;
        StringBuffer strb = new StringBuffer();
        try {
            URL url = new URL("http://192.168.1.7/DevMob/AfficherVilles.php");
            urlConnection = (HttpURLConnection) url.openConnection();
            urlConnection.setRequestMethod("GET");
            urlConnection.connect();
            InputStream inputStream = urlConnection.getInputStream();
            if (inputStream != null) {
                reader = new BufferedReader(new InputStreamReader(inputStream));
                String line;
                while ((line = reader.readLine()) != null) {
                    strb.append(line + "\n");
                }
            }
        } catch (Exception e) {
            Log.e("Error", e.getMessage());
        } finally {
            if (urlConnection != null)
                urlConnection.disconnect();
            if (reader != null) {
                try {
                    reader.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
        return strb.toString();
    }
}

```

Diagram illustrating the flow of data and actions in the `AsyncTask` class:

- Afficher un ProgressDialog**: Corresponds to the `onPreExecute()` method.
- Adresse IP Du serveur**: Corresponds to the URL `"http://192.168.1.7/DevMob/AfficherVilles.php"` used in `doInBackground()`.
- Ouverture de la connexion L'URL**: Corresponds to `url.openConnection()`.
- Récupère la réponse du serveur**: Corresponds to the `while` loop that reads the response from the server.
- Fermer la connexion**: Corresponds to the `finally` block that disconnects the `HttpURLConnection` and closes the `BufferedReader`.
- La réponse Du serveur**: Corresponds to the `return strb.toString();` statement.

26

Front-End

From JSON to java Object

```
public void DisplayResult(String contents){
```

```
    JSONArray jsonarray = null;
    ArrayList<Ville> ret= new ArrayList<Ville>();
    try {
        jsonarray = new JSONArray(contents);
        for (int i = 0; i < jsonarray.length(); i++) {
            JSONObject jsonobject = jsonarray.getJSONObject(i);
            String code = jsonobject.getString("code");
            String nom = jsonobject.getString("nom");
            int population=jsonobject.getInt("population");
            ret.add(new Ville(code,nom,population));
        }
    } catch (JSONException e) {
        e.printStackTrace();
    }
    ListView lv=(ListView)findViewById(R.id.ListView);
    lv.setAdapter(new VilleAdapter(this,ret));
}
```

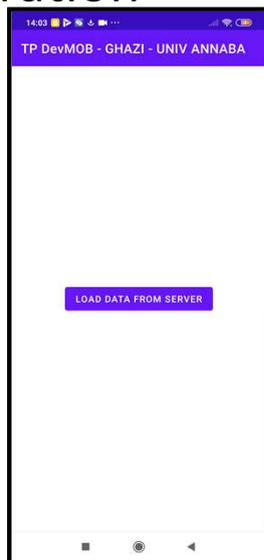
Lire le contenu
JSON
Et Transformer
le en une
collection
d'objet de type
Ville

Afficher la
collection d'objet
dans un
compostant
ListView

sabri.ghazi@univ-annaba.dz

27

Démonstration



sabri.ghazi@univ-annaba.dz

28

D'autres Technologies

- **Retrofit** : <https://square.github.io/retrofit/>
 - Permet de transformer une REST en une API très simple à utiliser.
- **ion** : <https://github.com/koush/ion>
 - Permet de charger les URL en asynchrone.

sabri.ghazi@univ-annaba.dz

29

