Documentation technique de la partie consultation en ligne

Cette partie de l'application nous permet de consulter en ligne les informations des praticiens étant dans le département sélectionné.

Il était tout d'abord fournit une documentation contenant des liens vers des pages web récupérant des informations dans la base de données grâce à des requêtes SQL. Les informations sont encodées en JSON. Ces informations, en JSON sont faites a l'aide de pages en php. Leur liaison avec le projet se fait par le biais de ces lignes.

```
private static final String serveur="192.168.1.19";
private static final String chemin="/gsb_context_android/";
```

```
objAsyncTask.execute("http://"+serveur+chemin+"lesdepartements.php");
```

Pour avoir accès à ces informations , j'ai utilisé la classe AccessHTTP étudiée dans un TP qui permet de demander l'exécution d'une requête HTTP au serveur.

Dans les classes DAO en ligne qui concerne ma partie et celle de mon camarade, je devais écrire des méthodes permettant de récupérer et traiter le résultat des requêtes HTTP exécutées :

DepartementEnLigneDAO :
Il y a la méthode getdepartement() , qui exécute une requête http nous dirigeant vers une page web, qui renvoie tous les départements de la base de données grâce à la page lesdepartements.php.

```
public void getDepartements(){
    AccesHTTP objAsyncTask = new AccesHTTP(){
        @Override
        protected void onPostExecute(Long result) {
            Log.d( tag: "log", msg: "onPostExecute departement dao");
            onTacheTerminee(jsonStringToDepartementArrayList(this.ret));
        }
    };
    objAsyncTask.execute("http://"+serveur+chemin+"lesdepartements.php");
}
```

```php
<?php
include "connexion.php";
header ("content-type: application/json; charset=utf-8");
try {

    $req = $conn->prepare("select * from departement order by departement_id");
    $req->execute();


    while ($ligne = $req->fetch(PDO::FETCH_ASSOC)){
        $resultat[]=$ligne;
    }
    print(json_encode($resultat));

} catch (PDOException $e) {
    print "Erreur !: " . $e->getMessage();
    die();
}

?>
```

On utilise la méthode onTacheTerminée , qui s'exécute après le onPostExecute qui lui-même est exécuté après le doInBackground() (pour ne pas bloquer le mainactivity). Cette méthode a pour paramètre soit un String, soit un département, soit une collection de département, comme précisé dans l'interface dédiée EventAsyncDepartement. Ici, nous utilisons en paramètre la méthode jsonStringToDepartementArrayList(), qui renvoie une collection de département basée sur le résultat de exécution de de la requête HTTP.

la méthode jsonStringToDepartementArraylist() qui a pour paramètre une chaîne de caractère, a été développée dans le but de traiter le résultat de la requête http, qui est encodée en JSON. Elle renvoie une collection de département construite à partir du JSON mis en paramètre.

```
private ArrayList<Departement> jsonStringToDepartementArrayList(String jsonString){
    Log.d( tag: "log", msg: "conversion jsonToDepartementArray : "+jsonString);

    ArrayList<Departement> listeDepartement = new ArrayList<~>();
     String numdepartement;
     String nomdepartement;

    try {
        JSONArray tabJson = new JSONArray(jsonString);
        for(int i=0;i<tabJson.length();i++){
            numdepartement=tabJson.getJSONObject(i).getString( name: "departement_code");
            nomdepartement= tabJson.getJSONObject(i).getString( name: "departement_nom");


            listeDepartement.add(new Departement(numdepartement,nomdepartement));
        }
    }
    catch (JSONException e){
        Log.d( tag: "log", msg: "pb decodage JSON");
    }
    Log.d( tag: "log",Integer.toString(listeDepartement.size()));
    return listeDepartement;
}
```

MedecinEnLigneDAO :
Le principe de la méthode getmedecin() est le même que getdepartement().

La

```
public void getMedecin() {
    AccesHTTP objAsyncTask = onPostExecute(result) → {
            Log.d( tag: "log",  msg: "onPostExecute MedecinEnLigneDAO");
            onTacheTerminee(jsonStringToMedecinArrayList(this.ret));
    };
    objAsyncTask.execute("http://"+serveur+chemin+"lesmedecins.php");
}
```

méthode arraytogetMedecindepartement(departement) a pour paramètre une chaîne de caractère correspondant au nom du département sélectionné dans la première liste déroulante. Son but est de créer une collection de médecin basée sur le résultat de la requête HTTP. Cela permet d'afficher les médecins en fonction du département mis en paramètre (fourni dans le cahier des charges).

```
115    public void arraytogetMedecinpardepartement(String nomdepartement) {
116
117        AccesHTTP objAsyncTask = onPostExecute(result) → {
120                Log.d( tag: "log",  msg: "http://"+serveur+chemin+"lesmedecins_par_depart.php?libelledepartement=Ain");
121                onTacheTerminee(jsonStringToMedecinArrayList(this.ret));
122
123        };
125        objAsyncTask.execute("http://"+serveur+chemin+"lesmedecins_par_depart.php?libelledepartement="+nomdepartement);
126    }
127 }
```

MainActivity :
Les boutons nous dirigeant vers notre partie étaient déjà créés, nous devions faire le lien entre ce bouton et notre activités

ActivityMedecinEnLigne :

Le but de cette activité est de récupérer les informations relatives aux départements et aux praticiens et de les mettre dans des listes déroulantes et des TextView grâce aux méthodes spécifiées précédemment.

Je devais tout d'abord créer des attributs qui me permettait de faire lien avec les objets de mon layout, , et deux attributs medecinSelectionne et departementSelectionne , qui auront pour valeur ce qui est sélectionné dans les listes déroulantes.

Tout d'abord il fallait remplir la première liste déroulante (spinner) qui devait contenir les noms des départements.
Remplirspindepartement() :

J'utilise la méthode onTacheTerminée qui a pour paramètre une collection de départements(alimentée par getdepartement()).

Je crée donc mon spinner (et je fais le lien avec mon layout grâce a l'objet spindepartement de type spinner).

Je crée son adapteur . Il y a une boucle for qui parcourt la collection , et pour chaque objet de la collection, je récupère son nom grâce au getter getnomdepartement() qui retourne l'attribut privé nom de la classe département, et je l'ajoute à l'adapter du spinner, ce qui aliment ma liste déroulante avec les nom des départements de la collection.

L'attribut departementSelectionne fait référence au département actuellement sélectionné dans la première liste déroulante .

Je devais ensuite faire une deuxième liste déroulante dépendante de la première, cela se fait dans le OnItemSelected() de departementAcces(Objet de type DAO), qui représente l'objet actuellement sélectionné dans la première liste déroulante , et de la mème manière je vais chercher les information grâce a arraytogetMedecindepartement(departement) , qui aura pour paramètre departementSelectionne , j'obtiendrais donc les médecins en fonction du département sélectionné dans la première liste déroulante.
Pour remplir les Textview représentant les différentes informations relatives au médecin sélectionné , c'est aussi dans le onItemSelected de medecinAcces , et j'attribue au champ (nom, prénom, etc..) la valeur des attributs du médecin grâce aux accesseurs(getnom(),getprenom(),etc…)

Ce que j’aurai pu améliorer c’est d’ajouter une fonctionnalité permettant de vider les TextView quand on passe d’un département ou il y à un praticien a un département ou il n’y en a pas.