M4206 - Programmation sur appareils mobiles communicants

Android

Quiz Down

Théo PELTIER
Rayan BENLACHEHEB

Sommaire

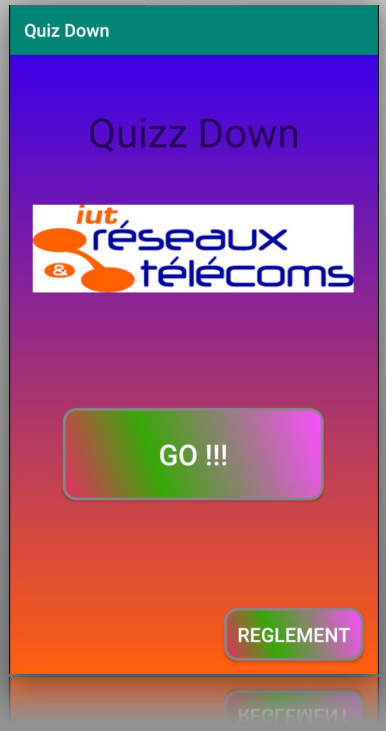
Présentation de l’application

Répartitions des tâches

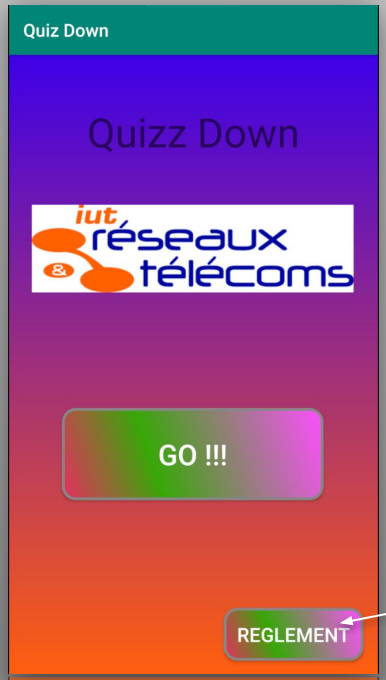
Difficultées rencontrées

Conclusion

Présentation de l'application



Présentation de l'application



Présentation de l'application

Choix du niveau de difficulté 3 modes :

- DUT = Nouveau
- DUT1 = Première année
- DUT2 = Deuxième année

Présentation de l'application

Vie du joueur diminue à chaque mauvaise réponse.

Information sur la question :
- Numéro
- Thème
- Question

Bouton pour passer une question “buguée”

Score du joueur augmente à chaque réponse juste.

Zone de Réponse :
- 1 à 4 cases de réponses
- 1 case s’il n’y aucune bonne réponse

Présentation de l'application

Difficulte.java

```java
package com.example.demourl;

import ...

public class Difficulte extends AppCompatActivity { //activite correspondant au choix du niveau des questions auquel on veut se confronter

    public static ArrayList<String> requeteListe = new ArrayList<>();


    protected void onCreate (Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.dif_page);
    }

    public void difficulte_choix(View v) {

        switch (v.getId()) { //en fonction du bouton de difficulte choisi

            case R.id.DUT:
                requete_id_question_difficulte( niveau: 0);
                startActivity(new Intent( packageContext: this, Quizz.class));
                break;

            case R.id.DUT1:
                requete_id_question_difficulte( niveau: 1);
                startActivity(new Intent( packageContext: this, Quizz.class));
                break;

            case R.id.DUT2:
                requete_id_question_difficulte( niveau: 2);
                startActivity(new Intent( packageContext: this, Quizz.class));
                break;
        }
```

```java
    public void requete_id_question_difficulte(int niveau){ //methode permettant d'effectuer une requete HTTP en vue d'obtenir l'ensemb

        requeteListe.clear(); //vide l'arrayList contenant d'anciens ID de questions valides

        URL recupe_liste_id_question = null;

        try {
            recupe_liste_id_question = new URL( spec: "http://infort.gautero.fr/index.php?action=get&obj=question&niv="+niveau);
        } catch (MalformedURLException e) {
            e.printStackTrace();
        }


        HTTP_Get(recupe_liste_id_question);
    }

    public void HTTP_Get (URL url){ //methode creant une requete HTTP asynchrone et l'executant en lui passant l'URL de destination

        Requete une_tache_de_requete = new Requete( difficulte: this);
        une_tache_de_requete.execute(url);
    }

    public void recup_id_question (String[] id){ //ajoute le tableau des ID des questions dans une ArrayList
        Collections.addAll(requeteListe, id);

    }


}
```

Présentation de l'application

```java
package com.example.demourl;

import ...

public class Reglement extends AppCompatActivity { //activite permettant l'affichage du reglement du jeu

    private static TextView reglement;

    protected void onCreate (Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.reglement_page);
        reglement = findViewById(R.id.Reglement);
        reglement.setText("Bonjour et bienvenue sur \"Quizz Down\", le but du jeu ici est d'aller " +
                "le plus loin possible dans les questions sans se tromper plus de 3 fois. \n" +
                "De nombreuses questions concernant la formation R&T vont vous être posées !! ");
    }

}
```

Reglement.java

Présentation de l'application

Requete.java

```java
package com.example.demourl;

import ...

public class Requete extends AsyncTask <URL, Void, JSONObject> { //classe permettant d'effectuer des requetes HTTP vers une

    private Quizz main;
    private Difficulte main2;


    public Requete(Quizz m) { main= m; }

    public Requete(Difficulte difficulte) { main2 = difficulte; }

    @Override
    protected JSONObject doInBackground(URL... urls) { //effectue la requete HTTP vers l'URL specifiee en parametre et passe
        URL u;
        URLConnection c;
        String inputline;
        StringBuilder codeHTML = new StringBuilder("");

        // On récupère l'url passée en paramètre de la méthode exécute
        u= urls[0];

        try {
            c= u.openConnection();  //temps maximun alloué pour se connecter
            c.setConnectTimeout(60000); //temps maximun alloué pour lire
            c.setReadTimeout(60000);    //flux de lecture avec l'encodage des caractères UTF-8
            BufferedReader in = new BufferedReader(
                    new InputStreamReader(c.getInputStream(), charsetName: "UTF-8"));
            while((inputline = in.readLine())!=null){   //concaténation+retour à la ligne avec \n
                codeHTML.append(inputline+"\n");
            }   //il faut bien fermer le flux de lecture
            in.close();
        } catch (IOException e) {
            e.printStackTrace();
            return null;
        }

        return decodeJSON(codeHTML.toString() );
    }
```

```java
public JSONObject decodeJSON (String codeHTML) { //convertit une chaine de caractere (data de requete HTPP) en JSONObject
    String s;
    JSONObject jobj;
    JSONObject tobj = null;
    int vi;
    StringBuilder r;

    r= new StringBuilder("");

    try {

        jobj= new JSONObject( codeHTML );

        if (jobj.length() < 1){
            return new JSONObject();
        }

        try {

            String keys = jobj.keys().next();
            tobj = jobj.getJSONObject(keys);
            tobj.put( name: "num", keys); // si l'object JSON de resultat est une question, alors on lui ajoute un champ pour

        }catch(JSONException e){ //On identifie le cas où un tableau d'ID de réponses est retourne
            return new JSONObject( codeHTML ) ;
        }



    } catch (JSONException e) {
        e.printStackTrace();
        tobj= new JSONObject();
    }
    return tobj ;
}
```

Présentation de l'application

Requete.java

```
protected void onPostExecute (JSONObject question) { //methode executee une fois que DoInBackground est terminee

    String[] resultat = null;

    String num = null;
    String titre = null;
    String domaine = null;
    String intitule = null;
    String niv = null;

    String id_reponse = null;

    try { //Dans le cas d'une requete portant sur les ID de questions associees à une difficultee : on recupere les donnees (
        id_reponse = question.getString( name: "questions");
        String result = id_reponse.substring(id_reponse.indexOf("[") + 1, id_reponse.indexOf("]")); //retire les crochés
        String[] id = result.split( regex: ",");    //split la liste des réponses

        main2.recup_id_question(id);


    } catch (JSONException b) {
```

Présentation de l'application

Requete.java

```
try {   //Dans le cas d'une requete portant sur les ID de reponses associees à une question : on recupere les donnees du JSON
    id_reponse = question.getString( name: "reponses"); //recupere la liste des id des réponses
    String result = id_reponse.substring(id_reponse.indexOf("[") + 1, id_reponse.indexOf("]")); //enleve les crochés
    String[] id = result.split( regex: ",");//split la liste des réponse

    main.affiche_bouton(id);

} catch (JSONException | MalformedURLException e) {

    try { //Dans le cas d'une requete portant sur une question : on recupere les donnees du JSONObject
        num = question.getString( name: "num");
        titre = question.getString( name: "titre");
        domaine = question.getString( name: "domaine");
        intitule = question.getString( name: "enonce");
        niv = question.getString( name: "niveau");

        resultat = new String[]{num, titre, intitule, domaine, niv};

        Quizz.affiche_question(resultat);

    } catch (JSONException c) { //Dans le cas d'une requete portant sur une reponse : on recupere les donnees du JSONObject

        try {
            Quizz.affiche_reponse(question.getString( name: "texte"), question.getInt( name: "juste"));
        } catch (JSONException ex) {
            ex.printStackTrace();
        }
```

Présentation de l'application

Quizz.java

```
public void question_suivante (View v){ //methode permettant la requete & l'initialisation de l'affichage d'une nouvelle questi

    bouton_suivant.setText("QUESTION SUIVANTE"); //permet de renommer le bouton "QUESTION SUIVANTE" en "QUESTION SUIVANTE" lors

    View score_affiche = findViewById(R.id.Score); //affiche & met a jour le score
    score_affiche.setVisibility(View.VISIBLE);
    Score_affiche.setText("Score : "+String.valueOf(Score_int));

    View vie_affiche = findViewById(R.id.Vie); // affiche & met a jour le compteur de vie
    vie_affiche.setVisibility(View.VISIBLE);
    Vie_affiche.setText("Vie : "+String.valueOf(vie_int));

    View resultat_affiche = findViewById(R.id.resultat_correction); //permet de cacher le carré qui affiche pour indiquer si la
    resultat_affiche.setVisibility(View.INVISIBLE);

    Nombre_reponse=0; //permet de reinitialiser le compteur du nombre de réponse qui s'incrémente apres l'affichage de chaque r
    Reponse_Juste.clear(); //permet de clear la liste des reponses juste d'une question


    URL recup_question = null;
    URL recupe_liste_id_reponse = null;

    Random rand = new Random();
    String id = requeteListe.get(rand.nextInt(requeteListe.size()));

    try { //url relative a la requete d'une question
        recup_question = new URL( spec: "http://infort.gautero.fr/index.php?action=get&obj=question&id="+id);
    } catch (MalformedURLException e) {
        e.printStackTrace();
    }

    try { //url relative a la requete des id des reponses correspondants a une question
        recupe_liste_id_reponse = new URL( spec: "http://infort.gautero.fr/index.php?action=get&obj=reponse&idQuestion="+id);
    } catch (MalformedURLException e) {
        e.printStackTrace();
    }


    HTTP_Get(recup_question);

    HTTP_Get(recupe_liste_id_reponse);
}
```

```
public void HTTP_Get (URL url){     //methode creant une requete HTTP asynchrone et l'executant en lui pa
    une_tache_de_requete = new Requete( m: this);
    une_tache_de_requete.execute(url);
}


public static void affiche_question(String[] resultat) { //methode permettant de remplir les differentes
    num_q.setText("n°" + resultat[0]);
    num_q.setVisibility(View.VISIBLE);
    niveau_q.setText(resultat[1]);
    intitule_q.setText(resultat[2]);
    titre_q.setText(resultat[3]);
    domaine_q.setText(resultat[4]);
}
```

Présentation de l'application

Quizz.java

```
public void affiche_bouton(String[] id) throws MalformedURLException { //methode permettant de rendre vi

    nombre_id_Reponse = id.length;

    //on rend tout les boutons invisibles au cas ou ceux-ci ne l'etaient pas suite a une precedente ques
    btn1.setVisibility(View.GONE);
    btn2.setVisibility(View.GONE);
    btn3.setVisibility(View.GONE);
    btn4.setVisibility(View.GONE);
    repAucune.setVisibility(View.GONE);


    if(nombre_id_Reponse == 1){
        btn1.setVisibility(View.VISIBLE);
        repAucune.setVisibility(View.VISIBLE);


    }else if(nombre_id_Reponse == 2){
        btn1.setVisibility(View.VISIBLE);
        btn2.setVisibility(View.VISIBLE);
        repAucune.setVisibility(View.VISIBLE);

    }else if(nombre_id_Reponse == 3){
        btn1.setVisibility(View.VISIBLE);
        btn2.setVisibility(View.VISIBLE);
        btn3.setVisibility(View.VISIBLE);
        repAucune.setVisibility(View.VISIBLE);

    }else if(nombre_id_Reponse == 4){

        btn1.setVisibility(View.VISIBLE);
        btn2.setVisibility(View.VISIBLE);
        btn3.setVisibility(View.VISIBLE);
        btn4.setVisibility(View.VISIBLE);
        repAucune.setVisibility(View.VISIBLE);
    }

    for(String id_reponse : id){ //pour chaque ID de reponse possible pour la question
        URL recup_reponse = new URL( spec: "http://infort.gautero.fr/index.php?action=get&obj=reponse&id=
```

```
public static void affiche_reponse(String texte, int validite){ //après chaque affichage d'une
    Nombre_reponse++;
    if (validite == 1){ //si la reponse affichee est une bonne reponse elle est ajoute dans u
        Reponse_Juste.add(texte);
    }

    if(Nombre_reponse==1){
        btn1.setText(texte);
    }else if(Nombre_reponse==2){
        btn2.setText(texte);
    }else if(Nombre_reponse==3){
        btn3.setText(texte);
    }else if(Nombre_reponse==4){
        btn4.setText(texte);
    }
}
```

Présentation de l'application

Quizz.java

```
    case R.id.Rep4:
        if(Reponse_Juste.size() == 0){
            invisible_bouton();
            reponse_fausse();

        }else {

            for (String resultat : Reponse_Juste) {
                if (btn4.getText() == resultat) {
                    invisible_bouton();
                    reponse_juste();

                } else {

                    invisible_bouton();
                    reponse_fausse();
                }
            }
        }

        break;

    case R.id.RepAucune:
            if(Reponse_Juste.size() == 0){
                invisible_bouton();
                reponse_juste();

            }else{
                invisible_bouton();
                reponse_fausse();

            }
        break;
}
```

```
        case R.id.Rep2:
            if(Reponse_Juste.size() == 0){
                invisible_bouton();
                reponse_fausse();

            } else {
                for (String resultat : Reponse_Juste) {
                    if (btn2.getText() == resultat) {
                        invisible_bouton();
                        reponse_juste();

                    } else {
                        invisible_bouton();

                        reponse_fausse();
                    }
                }

            break;

        case R.id.Rep3:
            if(Reponse_Juste.size() == 0){

                invisible_bouton();

                reponse_fausse();

            }else {

                for (String resultat : Reponse_Juste) {
                    if (btn3.getText() == resultat) {
                        invisible_bouton();
                        reponse_juste();

                    } else {
                        invisible_bouton();
                        reponse_fausse();
                    }
                }
            break;
```

```
@RequiresApi(api = Build.VERSION_CODES.O)
public void correction_reponse(View v) { //permet de savoir si la reponse choisie par l

    switch (v.getId()) { //en fonction de l'ID du bouton choisit
        case R.id.Rep1:
            if(Reponse_Juste.size() == 0){

                invisible_bouton();
                reponse_fausse();
            } else {
                for (String resultat : Reponse_Juste) {
                    if (btn1.getText() == resultat) { //si la réponse choisi est conten

                        invisible_bouton();
                        reponse_juste();
                    } else {

                        invisible_bouton();
                        reponse_fausse();
                    }
                }
            }
```

Difficultées rencontrées

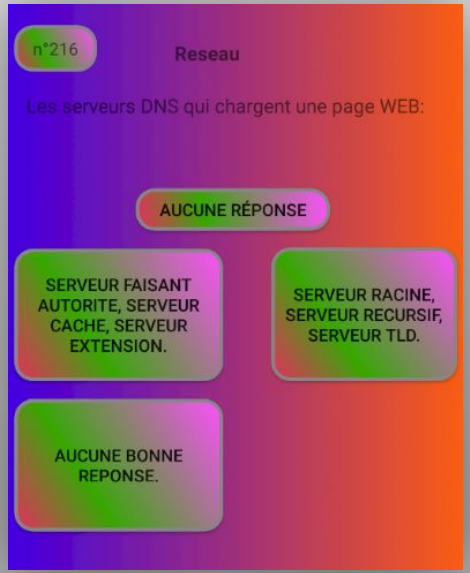
- Mise en place des requêtes asynchrones et de leur fonctionnement :
  Nous avons trouvé une documentation en français claire qui nous a aidé à mieux comprendre, ce qui nous a permis d'exploiter pleinement la classe AsyncTask

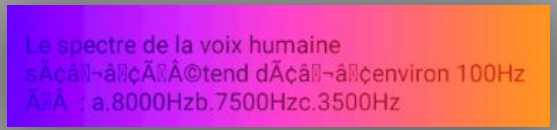
- Organisation minutieuse et longue du layout d'affichage des questions :
  Nous avons dû comprendre le fonctionnement du placement des différent éléments sur une activité, mais nous y sommes parvenus

- Les principales difficultés rencontrées, la méthode de résolution utilisée et les résultats.

Bugs connus

- Certaines questions comportent déjà une réponse “Aucune bonne réponse” (voir l’exemple ci-contre) ce qui induit une contradiction avec notre bouton “Aucune Réponse” prévu pour cela
- 90% des questions ne comporte **aucune** bonne réponse… Il est donc facile de gagner en répondant à chaque fois “Aucune Réponse”
- Certaines questions comportent des caractères accentués qui ne peuvent pas être lu du fait de leur mauvaise insertion dans la base de données (problème d’encodage sans doute)





Conclusion

Nous avons rencontré quelques problèmes mais rien d'insurmontable, cela nous a permis d'approfondir nos connaissance en Java, de découvrir la structure XML et d'apprendre la gestion des différents éléments d'une application. Nous avons également perfectionné nos compétences en travail de groupe, toute les réunions que nous avons effectués se sont faites par Discord (crise du Covid oblige), lorsqu'il s'agissait de travailler à plusieurs sur le code l'utilisation du logiciel TeamViewer nous a été très bénéfique car nous a permis de participer ensemble à la résolution de problèmes & bugs dans le code.