ETAPE 2 : découverte des classes DAO

L'objectif de cette méthode est de récupérer les informations d'une excursion à partir de son identifiant. Jusqu'ici, on récupérait toutes les excursions disponibles avec la méthode **getAll** qui utilisait une boucle pour afficher toutes les excursions.

Ajout de la méthode getOneById dans DaoMiniExcursion :

Dans **DaoMiniExcursion.java**, on crée la méthode **getOneById** qui se connecte à la base de données et récupère les informations des minis excursions.

```
public static MiniExcursion getOneById(String codeExcursion) throws SQLException {
MiniExcursion excursion = null;
Connection cnx = ConnexionBDD.getConnexion();
PreparedStatement pstmt = cnx.prepareStatement(string: "SELECT * FROM Excursion WHERE Code = ?");
pstmt.setString(i: 1, string: codeExcursion);
ResultSet rs = pstmt.executeQuery();

if (rs.next()) {
    float tarif = rs.getFloat(string: "Tarif");
    if (tarif == 0.0f) {
        excursion = new MiniExcursion(
                code:rs.getString(string: "Code"),
                libelle: rs.getString(string: "Libelle"),
                nbPlaces:rs.getInt(string: "NbPlaces")
        );
    } else {
        excursion = new MiniExcursionPayante(
                code:rs.getString(string: "Code"),
                libelle: rs.getString(string: "Libelle"),
                nbPlaces:rs.getInt(string: "NbPlaces"),
                tarif:rs.getFloat(string: "Tarif")
        );
    }

    // Ajouter les étapes
    ArrayList<Etape> lesEtapes = DaoEtape.getAllByExcursion(codeExcursion);
    excursion.setLesEtapes(lesEtapes);
}
```

Test dans TestDaoMiniExcursion :

Dans TestDaoMiniExcursion, on va ajouter le code nécessaire pour tester notre méthode **getOneById.** On peut tout simplement récupérer le code du test 1 (**getAll**) et modifier la boucle qui parcourt la base de donnée pour trouver toutes les excursions :

```
ArrayList<MiniExcursion> lesExcursions = DaoMiniExcursion.getAll();
for (MiniExcursion exc : lesExcursions) {
    System.out.println(x: exc.toStringEtat());
}
System.out.println(lesExcursions.size()+" excursions trouvées");
```

On remplace la bouche par une condition : si l'identifiant donné existe alors on l'affiche, sinon on affiche qu'il n'existe pas :

```
// Test 2 getOneById
System.out.println(x: "\n Test 2 : TestDaoMiniExcursion.getOneById");
try {
    String codeTest = "E01"; // Code existant dans la BDD
    MiniExcursion excursion = DaoMiniExcursion.getOneById(codeExcursion:codeTest);
    if (excursion != null) {
        System.out.println("Excursion trouvée : " + excursion.toStringEtat());
    } else {
        System.out.println("Aucune excursion trouvée pour le code : " + codeTest);
    }
} catch (SQLException ex) {
    Logger.getLogger(name:TestDaoMiniExcursion.class.getName()).log(level:Level.SEVERE, msg: "TestDaoMiniExcursion - échec getOneById : ", thrown: ex);
}

// Fermeture de la connexion
try {
    ConnexionBDD.getConnexion().close();
    System.out.println(x: "\nConnexion à la BDD fermée");
} catch (SQLException ex) {
    Logger.getLogger(name:TestDaoMiniExcursion.class.getName()).log(level:Level.SEVERE, msg: "TestDaoMiniExcursion - échec de la fermeture de la connexion : ", thrown: ex);
}
```

En utilisant l'identifiant E01, on obtient le résultat de la première excursion :

```
 Test 2 : TestDaoMiniExcursion.getOneById
Excursion trouvée : MiniExcursion{code=E01, libelle=Excursion dans l'île, nbPlaces=8
        les étapes=
        1 : Etape{numEtape=1, description=traversée aller, dureePrevue=30}
        2 : Etape{numEtape=2, description=promenade dans l'île, dureePrevue=60}
        3 : Etape{numEtape=3, description=visite de la chapelle, dureePrevue=20}
        4 : Etape{numEtape=4, description=visite du phare, dureePrevue=30}
        5 : Etape{numEtape=5, description=promenade sur une petite crique, dureePrevue=15}
        6 : Etape{numEtape=6, description=visite du jardin exotique, dureePrevue=45}
        7 : Etape{numEtape=7, description=traversée retour, dureePrevue=30} }
```

En revanche l'identifiant E03 qui n'existe pas, renvoie ce message :

```
 Test 2 : TestDaoMiniExcursion.getOneById
Aucune excursion trouvée pour le code : E03
```