

SERIALIZATION & DESERIALization

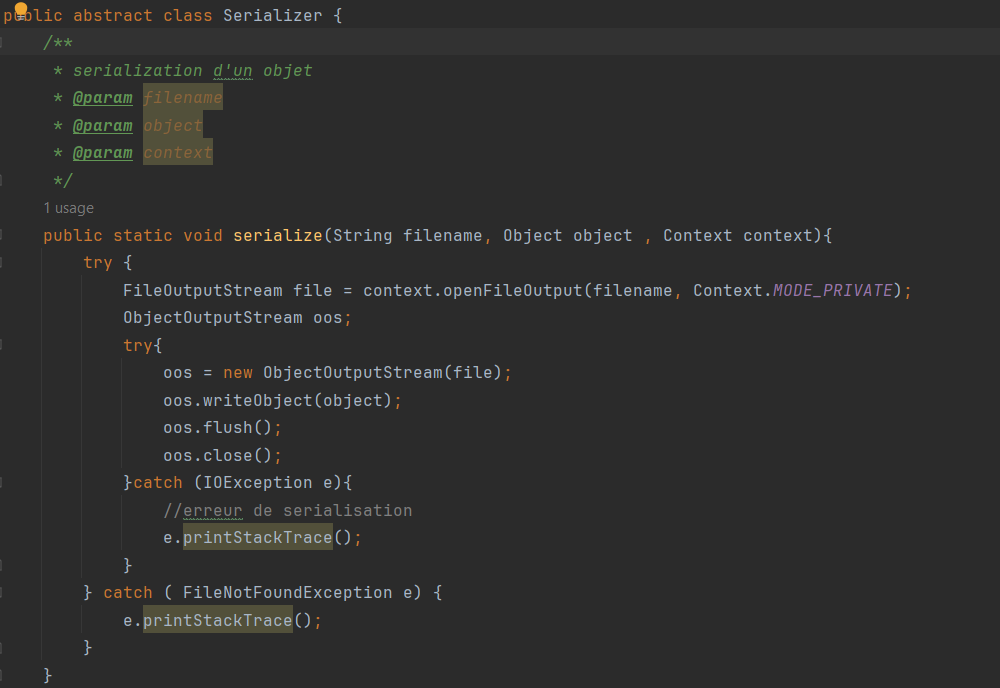
Atelier 5



November 29, 2024

Hafsi Mahdi

ISET Rades



**Classe Serializer**

**Rôle :** Cette classe abstraite est responsable de la sérialisation et de la désérialisation d'objets Java.

* **Méthodes clés :**
  + **serialize :**
    - **Description :** Enregistre un objet dans un fichier.
    - **Processus :**
      1. Ouvre un flux de fichier en mode privé à l'aide de Context.openFileOutput.
      2. Utilise un ObjectOutputStream pour écrire l'objet dans le fichier.
      3. Ferme le flux après l'écriture.
      4. **Gestion des erreurs :** Captures des exceptions comme FileNotFoundException et IOException pour gérer les erreurs de sérialisation.

*/\*\*  
 \* deserialisation d'un objet  
 \** ***@param*** *filename  
 \** ***@param*** *context  
 \** ***@return*** *objet  
 \*/*public static Object deserializer(String filename , Context context){  
 try {  
 FileInputStream file = context.openFileInput(filename);  
 ObjectInputStream ois ;  
 try{  
 ois = new ObjectInputStream(file);  
 try {  
 Object object = ois.readObject();  
 ois.close();  
 return object;  
 } catch (ClassNotFoundException e) {  
 e.printStackTrace();  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 }  
  
 return null;  
}

**deserializer :**

* **Description :** Lit un objet sérialisé depuis un fichier.
* **Processus :**
  1. Ouvre un flux de fichier en lecture à l'aide de Context.openFileInput.
  2. Utilise un ObjectInputStream pour lire l'objet.
  3. Renvoie l'objet lu après le cast.
  4. **Gestion des erreurs :** Gère les exceptions comme FileNotFoundException, IOException et ClassNotFoundException.

Passant au classe Control

*/\*\*  
 \* Creation de profile  
 \** ***@param*** *poids  
 \** ***@param*** *taille en cm  
 \** ***@param*** *age  
 \** ***@param*** *sexe 1 pour homme et 0 pour femme  
 \*/* public void creerProfil(Integer poids , Integer taille , Integer age , Integer sexe , Context context){  
 *profil* = new Profil(new Date(), poids , taille , age , sexe );  
 Serializer.*serialize*(*nomFic*, *profil*, context);  
 }

**2. Classe Control (com.example.coach.controlleur.Control)**

* **Rôle :** Contrôleur du modèle MVC, gère les données utilisateur et leur persistance.
* **Usage de la sérialisation :**
  + **Sérialisation :**
    - **Méthode :** creerProfil
    - **Description :**
      * Après avoir créé un profil via Profil, la méthode utilise Serializer.serialize pour sauvegarder le profil dans un fichier nommé saveprofil.bin.
      * Appelée lors de la création ou mise à jour d'un profil.

*/\*\*  
 \* recuperer l'objet serialiser(le profil)  
 \** ***@param*** *context  
 \*/*private static void recupSerialize(Context context){  
 *profil* = (Profil) Serializer.*deserializer*(*nomFic*, context);  
}

**Désérialisation :**

* **Méthode :** recupSerialize
* **Description :**
  + Récupère un profil existant depuis le fichier saveprofil.bin en appelant Serializer.deserializer.
  + Le profil est ensuite utilisé pour initialiser les attributs de la classe.

Passant au class mainActivity :

*/\*\*  
 \* recuperation du profil s'il été serialisé  
 \*/*private void recupProfil(){  
 if(control.getPoids() != null){  
 txtPoids.setText(control.getPoids().toString());  
 txtTaille.setText(control.getTaille().toString());  
 txtAge.setText(control.getAge().toString());  
 rdFemme.setChecked(true);  
 if(control.getSexe()==1){  
 rdHomme.setChecked(true);  
 }  
 //simule le clic sur le boutton calcul  
 ((Button)findViewById(R.id.*btnCalc*)).performClick();  
 }  
}

**3. Classe MainActivity**

* **Rôle :** Vue principale qui gère l'interface utilisateur.
* **Usage indirect de la sérialisation :**
  + Lors de l'initialisation avec init, la méthode recupProfil appelle Control.recupSerialize pour charger un profil sauvegardé.
  + Les valeurs désérialisées (comme poids, taille, âge, sexe) sont affichées dans l'interface via des champs graphiques (EditText, RadioButton, etc.).
  + Simule un clic automatique sur le bouton "Calcul" pour afficher directement les données désérialisées.

***CONCLUSION***

**Objectifs de la sérialisation :**

* Permettre la persistance des données utilisateur entre les sessions.
* Faciliter le chargement et la sauvegarde des profils sans dépendre de bases de données.

**Gestion des erreurs :**

* Chaque opération de sérialisation/désérialisation est entourée de blocs try-catch, évitant des crashs en cas de fichier manquant ou de problème de lecture/écriture.