SampleModel model = **new** SampleModel();

model.setAccount(**new** Account());

model.getAccount().setEmail("support@lesfurets.com");

System.***out***.println(model.getAccount().getEmail());

FieldModel fieldModel = **new** SampleModelWrapper(model);

System.***out***.println(fieldModel.<String> get(***EMAIL***));

fieldModel.set(***EMAIL***, "gdu@lesfurets.com");

System.***out***.println(fieldModel.<String> get(***EMAIL***));

--------------------------------------------------------------------

FieldModel model = SampleModels.*wrapper*();

System.***out***.println(model.<String> get(***EMAIL***));

model.stream().forEach(System.***out***::println);

Map<FieldId, Object> map = model.stream()

.collect(*toMap*(Entry::getKey, Entry::getValue));

System.***out***.println(map);

SampleModelWrapper newModel = map.entrySet().stream()

.collect(SampleModelWrapper.*toFieldModel*());

newModel.stream().forEach(System.***out***::println);

System.***out***.println(newModel.getModel().getAccount().getEmail());

--------------------------------------------------------------------

FieldModel model = SampleModels.*wrapper*();

Map<FieldId, Object> map = model.stream()

.collect(*toMap*(Entry::getKey, Entry::getValue));

SampleModelWrapper newModel = map.entrySet().stream()

.filter(e -> e.getKey().hasTag(SampleTag.***ACCOUNT***))

// .filter(e -> e.getKey().hasTag(SampleTag.USER))

.collect(SampleModelWrapper.*toFieldModel*());

newModel.stream().forEach(System.***out***::println);

1. Créer une instance de SampleModel
2. Créer une instance de Account
3. Setter l’email
4. Afficher l’email
5. Exécuter
6. Créer un wrapper avec l’instance de SampleModel
7. Afficher l’email en lisant par sa clé SampleFieldId.EMAIL
8. Changer la valeur de l’email par sa clé
9. Afficher l’email
10. Exécuter
11. Créer un FieldModel initialisé avec SampleModels.wrapper()
12. Afficher l’email
13. Exécuter
14. Afficher le stream de toutes les clés/valeurs du modèle
15. Exécuter
16. Collecter le stream de clés/valeurs dans une Map
17. Afficher la Map
18. Exécuter
19. Collecter le stream de clés/valeurs de la Map dans un SampleModelWrapper
20. Afficher le stream de toutes les clés/valeurs du nouveau modèle
21. Exécuter
22. Créer un FieldModel initialisé avec SampleModels.wrapper()
23. Collecter le stream de clés/valeurs dans une Map
24. Collecter le stream de clés/valeurs de la Map dans un SampleModelWrapper en filtrant les clés taguées ACCOUNT
25. Afficher le stream de toutes les clés/valeurs du nouveau modèle

FieldModel model = SampleModels.*wrapper*();

CreateTable createTable = SchemaBuilder.*createTable*("sample", "model")

.withPartitionKey("snapshot\_id", DataTypes.***TIMEUUID***)

.withClusteringColumn(***LOGIN***.name(), DataTypes.***TEXT***);

**for** (FieldInfo info : model.getFieldInfos()) {

createTable = createTable.withColumn(info.id().code(), *cqlType*(info));

}

CreateTableWithOptions createTableWithOptions = createTable.withClusteringOrder(***LOGIN***.name(), ClusteringOrder.***DESC***);

*execute*(createTableWithOptions.build());

--------------------------------------------------------------------

FieldModel model = SampleModels.*wrapper*();

Map<String, Term> values = model.stream().collect(Collectors.*toMap*(

e -> e.getKey().code(),

e -> QueryBuilder.*literal*(e.getValue(),

*codecRegistry*())));

Insert insertInto = QueryBuilder.*insertInto*("sample", "model")

.value("snapshot\_id",

QueryBuilder.*literal*(Uuids.*timeBased*()))

.values(values);

*execute*(insertInto.build());

--------------------------------------------------------------------

FieldModel model = SampleModels.*wrapper*();

**try** (CqlSession session = CqlSession.*builder*().build()) {

model.getFieldInfos().stream()

.filter(f -> !session.getMetadata()

.getKeyspace("sample").get()

.getTable("model").get()

.getColumn(f.id().code()).isPresent())

.forEach(f -> session.execute(SchemaBuilder.*alterTable*("sample",

"model")

.addColumn(f.id().code(), *cqlType*(f)).build()));

}

1. Démarrer un serveur Cassandra
2. Créer un keyspace ‘sample’
3. Créer un cluster et une session en utilisant le Cluster.Builder avec un contact point sur ‘localhost’ et un codec registry
4. Créer une requête ‘Create’ avec le SchemaBuilder pour la table ‘model’ dans le keyspace ‘sample’
5. Exécuter
6. Exécuter ‘DESCRIBE sample ;’ dans CQLSH
7. Créer ne requête ‘Insert’ avec le QueryBuilder pour la table ‘model’ dans le keyspace ‘sample’
8. Exécuter
9. Exécuter ‘USE sample ;’ dans CQLSH
10. Exécuter ‘SELECT \* FROM model ;’ dans CQLSH
11. Ajouter un field ‘company’ dans la classe Account
12. Ajouter un id COMPANY dans SampleFieldId
13. Ajouter l’annotation @SampleField sur le field ‘company’
14. Générer le getter/setter du field ‘company’
15. Exécuter la commande ‘./gradlew –p sample doovMapGenSample’
16. Ajouter dans SampleModels.wrapper() l’initialisation de Company
17. Ré exécuter la requête Insert
18. Montrer l’exception lors de l’exécution
19. Créer la requête ‘Alter’ qui test si chaque colonne existe dans les métadonnées du cluster
20. Exécuter
21. Exécuter ‘DESCRIBE sample ;’
22. Ré exécuter la requête Insert
23. Exécuter ‘SELECT \* FROM model ;’ dans CQLSH