

```
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);
```

```

FieldModel model = SampleModels.wrapper();
Create create = SchemaBuilder.createTable("meetup", "sample_model")
    .addPartitionKey(LOGIN.name(), text())
    .addClusteringColumn("snapshot_id", timeuuid());
model.getFieldInfos().stream().filter(f -> f.id() != LOGIN)
    .forEach(f -> create.addColumn(f.id().code(), cqlType(f)));

```

```

-----

Create.Options createWithOptions = create.withOptions()
    .clusteringOrder(LOGIN.name(), DESC);
session.execute(createWithOptions);

```

```

Insert insert = QueryBuilder.insertInto("meetup", "sample_model");
model.stream().forEach(e -> insert.value(e.getKey().code(),
    e.getValue()));
insert.value("snapshot_id", UUIDs.timeBased());
session.execute(insert);

```

```

-----

FieldModel model = SampleModels.wrapper();
model.getFieldInfos().stream().filter(f -> {
    ColumnMetadata column = session.getCluster().getMetadata()
        .getKeyspace("meetup")
        .getTable("sample_model")
        .getColumn(f.id().code());
    return column == null;
}).forEach(f -> {
    session.execute(SchemaBuilder
        .alterTable("meetup", "sample_model")
        .addColumn(f.id().code())
        .type(cqlType(f)));
});

```

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

26. Démarrer un serveur Cassandra
27. Créer un keyspace 'meetup'
28. Créer un cluster et une session en utilisant le Cluster.Builder avec un contact point sur 'localhost' et un codec registry
29. Créer une requête 'Create' avec le SchemaBuilder pour la table 'sample_model' dans le keyspace 'meetup'
30. Exécuter
31. Exécuter 'DESCRIBE meetup ;' dans CQLSH
32. Créer ne requête 'Insert' avec le QueryBuilder pour la table 'sample_model' dans le keyspace 'meetup'
33. Exécuter
34. Exécuter 'USE meetup ;' dans CQLSH
35. Exécuter 'SELECT * FROM sample_model ;' dans CQLSH
36. Ajouter un field 'company' dans la classe Account
37. Ajouter un id COMPANY dans SampleFieldId
38. Ajouter l'annotation @SampleField sur le field 'company'
39. Générer le getter/setter du field 'company'
40. Exécuter la commande './gradlew -p sample doovMapGenSample'
41. Ajouter dans SampleModels.wrapper() l'initialisation de Company
42. Ré exécuter la requête Insert
43. Montrer l'exception lors de l'exécution
44. Créer la requête 'Alter' qui test si chaque colonne existe dans les métadonnées du cluster
45. Exécuter
46. Exécuter 'DESCRIBE meetup ;'
47. Ré exécuter la requête Insert
48. Exécuter 'SELECT * FROM sample_model ;' dans CQLSH