



DOMAINMODEL.STREAM()

<https://github.com/lesfurets/model-map>

WELCOME TO THE FURETS !

@DIGUGLI – GILLES DI GUGLIELMO

- Designer of sweet cooked software since 1999
- Software Architect at LesFurets.com

@JUBAUDRY – JULIEN BAUDRY

- Java Developer since 2007
- Software Architect at LesFurets.com



1 website, 5 Insurance Products : Car, Health, Home, Bike, Loan
1 codebase, 450k lines of code, 60k unit tests, 150 selenium tests

22 Developers, 2 DevOps, 4 Architects

19 production servers including Load balancers, Frontend, Backend, Databases,
BI

1 release per day

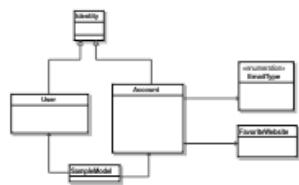
8 years of code history

3M quotes/year, 40% of market share, 4M of customers

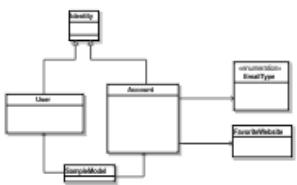
MOVING THE ARCHITECTURE

WHAT YOU HAVE ...

Domain Entities
JSON



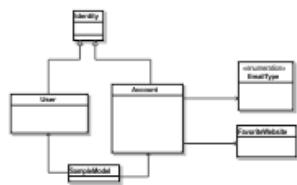
Domain Entities
Java Beans



Domain
Relational
Entities



Analytics Entities
Java Beans

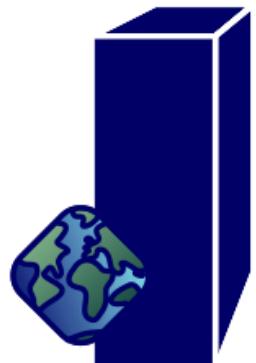


Analytics
Relational
Entities



JavaScript
Front App

JSON
RPC



Front
WebApp

ORM
or
SQL



Runtime
DataBases

ORM
or
SQL



Batch
ETL

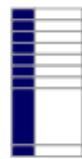
ORM
or
SQL



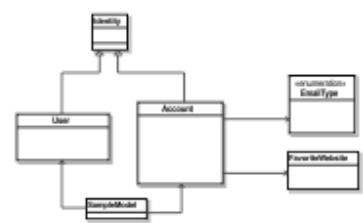
BackOffice
DataBases

WHAT YOU WANT ...

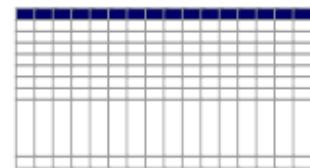
Key/Value
JSON
Dictionary



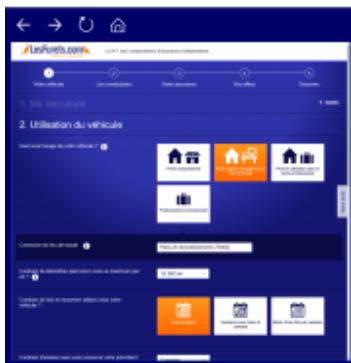
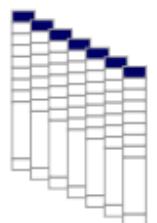
Domain Entities
Java Beans



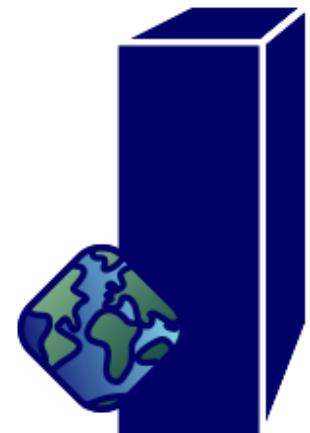
Wide Columns
Model



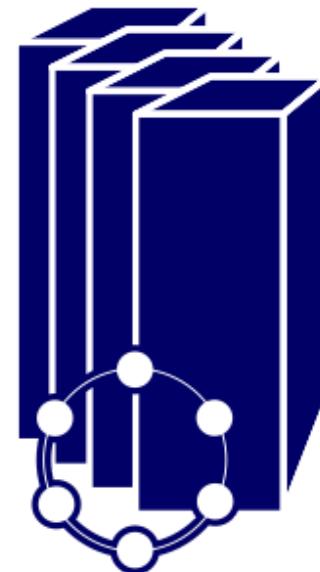
Analytics
Data Vecto



JavaScript
Front App



Front
WebApp



Cassandra
Cluster

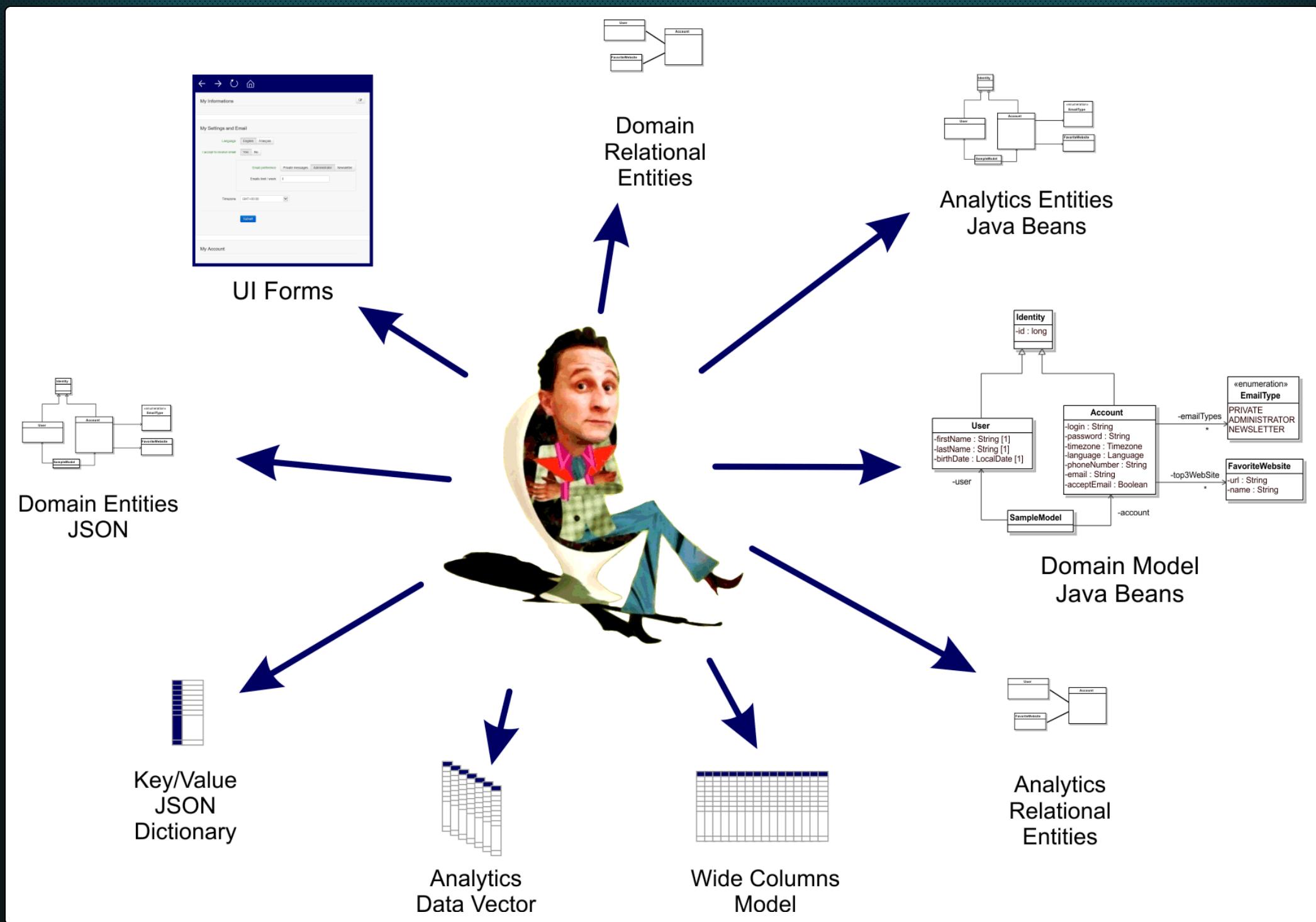


Data
Streaming
μ Batch

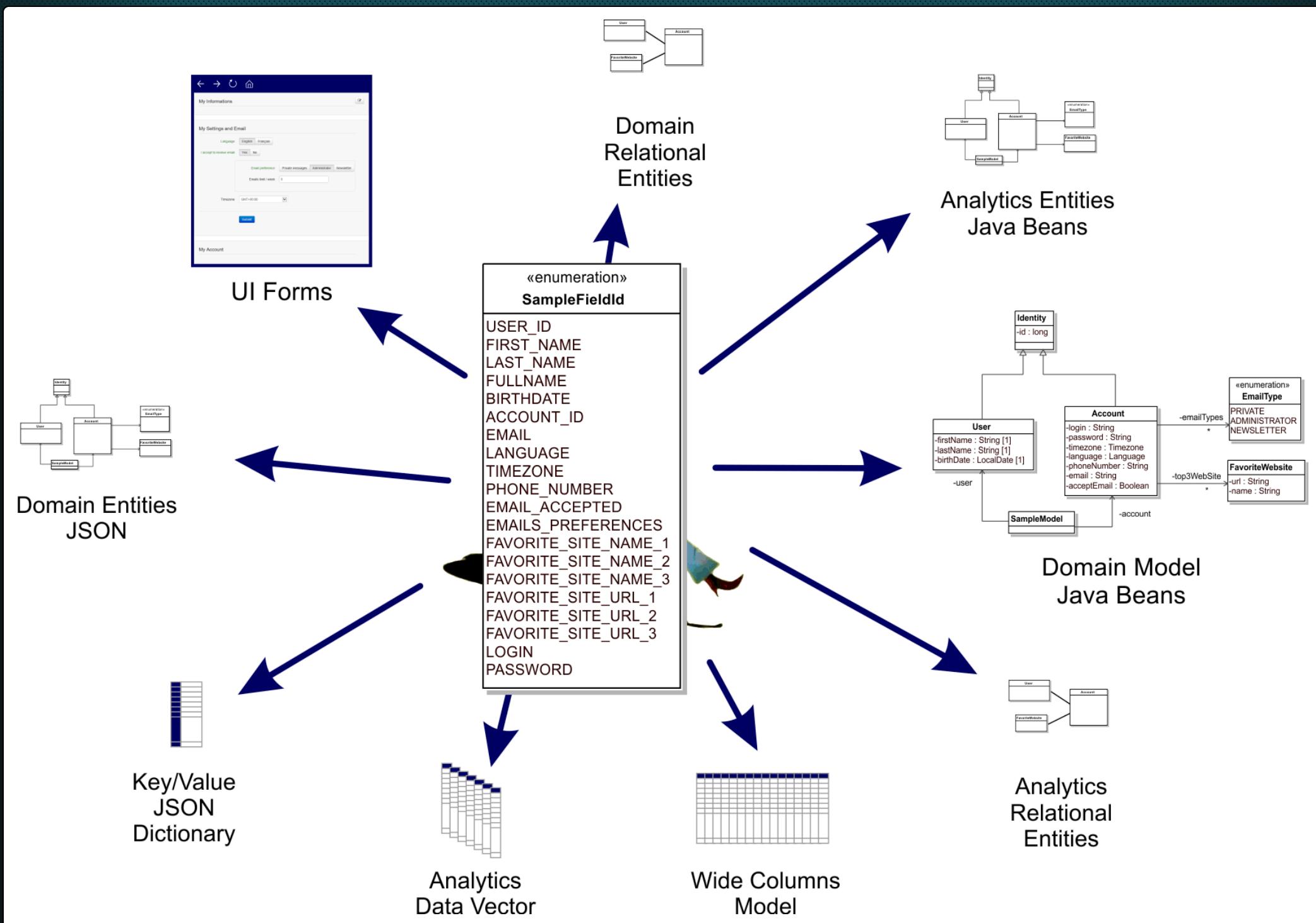
QUE FAIRE ?



ARCHITECT AT WORK ...



KEYING EVERYTHING !



A BIT OF LIVECODING

LIVE CODE 1 : INTRODUCTION TO KEY VALUE API

```
SampleModel model = new SampleModel();
model.setAccount(new Account());
model.getAccount().setEmail("parisjug@gmail.com");
System.out.println(model.getAccount().getEmail());

FieldModel fieldModel = new SampleModelWrapper(model);
System.out.println(fieldModel.<String> get(EMAIL));

fieldModel.set(EMAIL, "parisjug.gmail.com");
System.out.println(fieldModel.<String> get(EMAIL));
```

LIVE CODE 2 : MIXING WITH JAVA.UTIL.MAP

```
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());
```

LIVE CODE 3 : TAG FILTERING

```
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);
```

LIVE CODE 4 : GENERATE CQL

```
FieldModel model = SampleModels.wrapper();
Create create = SchemaBuilder.createTable("Field").addClusteringColumn(LOGIN.name(), text())
    .addPartitionKey("snapshot_id", timeuuid());

stream(model.getFieldInfos()).filter(f -> f.id() != LOGIN)
    .forEach(f -> create.addColumn(f.id().name(), cqlType(f)));

Create.Options createWithOptions = create.withOptions().clusteringOrder(LOGIN.name(), DESC);
System.out.println(createWithOptions);

Insert insert = QueryBuilder.insertInto("Field");
model.stream().forEach(e -> insert.value(e.getKey().name(), e.getValue()));

System.out.println(insert.getQueryString(codecRegistry()));
```

LIVE CODE 5 : DIFF TWO MODELS

```
FieldModel sample_1 = SampleModels.wrapper();
FieldModel sample_2 = SampleModels.wrapper();

sample_1.set(FAVORITE_SITE_NAME_3, null);
sample_1.set(FAVORITE_SITE_URL_3, null);
sample_2.set(FAVORITE_SITE_NAME_1, "LesFurets.com");
sample_2.set(FAVORITE_SITE_URL_1, "www.lesfurets.com");
sample_2.set(EMAILS_PREFERENCES, Collections.emptyList());

/* stream all key-values pair from both models */
Stream.concat(sample_1.stream().map(buildRight), sample_2.stream().map(buildLeft))

        /* merging key-value pair in a map */
        .collect(Collectors.toMap(Triple::getMiddle, Function.identity(), merge))

        /* filter to keep only key with 2 differents values */
        .values().stream().filter(isNotSame)

        /* print keys with differents values */
        .forEach(System.out::println);
```

USAGES ARE UNLIMITED

YOU CAN USE JAVA 8 STREAM ON YOUR DOMAIN MODEL !

- FieldModel#stream() : a stream of key/value pair (FieldId/Object)
- FieldModel#parallelStream() : parallel stream and better performance
- Easy use a filter/map/reduce operations on a domain model

Could you write a domain model diff tool now ?

WHERE WE USE IT AT LESFURETS.COM

- Simple UI binding by replacing JSON model by a Dictionary (80% less code)
 - Use JSON Dictionary for client/server RPC
 - Batch and ETL with an email provider (90% less code)
 - Persistence in cassandra (from scratch with 200 lines of code for a whole model)
 - Injecting user data in selenium (40% less code)
 - Mocking our domain model in thousand of unit tests
- ... everything in iterative fashion during the past 2 years.

FUTURE AND IDEAS

- Push as OSS our Cassandra serialisation code
- Push some Spark samples
- JEP 276: Dynamic Linking of Language-Defined Object Models

WE NEED A NEW NAME ?

- Model-Map is cool, keep it
- YAMF: Yet Another Mapping Framework
- OWKF: Obi Wan Kenobi Framework
- Submit your ideas

MODEL-MAP AVAILABLE ON GITHUB

- <http://github.com/lesfurets/model-map>
- Framework and examples
- Apache Licence
- Try it and contribute !

THANK YOU!

