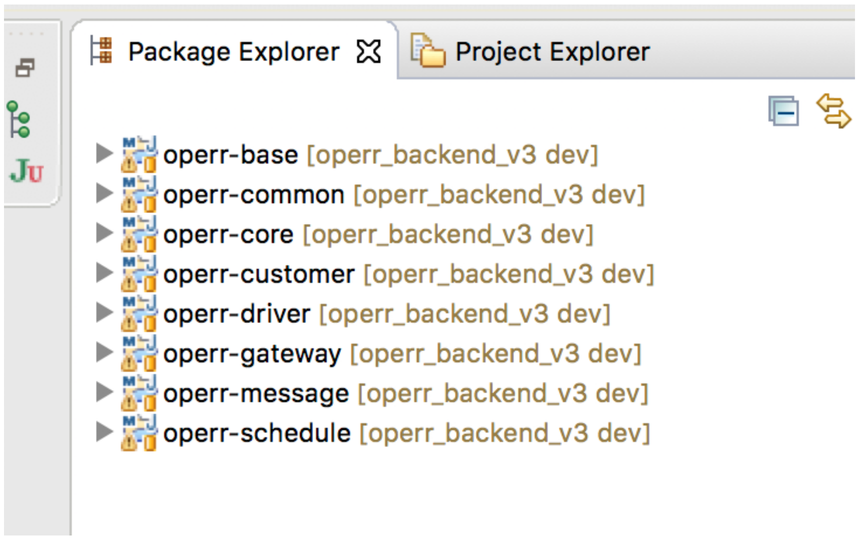
Coding rule of Operr3

09/25/2017

1. Development tool and technologies:
    - Git. I recommend you to use sourcetree.
        - ➢ Please do not touch branch master.
        - ➢ Pls do not directly work on branch dev.
        - ➢ Please create a branch dev_[your name], for example, I created dev_zhao.
        - ➢ Please pull code before you start to work in the morning.
        - ➢ Please push code into repo after you finish work in the evening.
        - ➢ Please do pull request for each submit.
        - ➢ Please add ticket number in push comment, it will establish relationship between ticket and code.
    - I recommend you to use eclipse3, if you use other development tools, please do not push configure file into repository.
        - ➢ Eclipse
        - ➢ NetBeans
        - ➢ Intellij Idea
    - Jenkins. Continuous integration
        - ➢ Daily integration testing based on Junit test.
        - ➢ Each project is using a jar file.
        - ➢ Upload jar files in S3, and do auto deployment with Ansible/Chef OpsWorks.
    - We are using the following technologies.
        - ➢ Java 1.8, standard version, Please don't use lambdas.
        - ➢ Redis 4.2.0
        - ➢ MySQL / MariaDB (latest version)
        - ➢ Springboot 1.5.4
        - ➢ Spring Data
        - ➢ Hibernate
        - ➢ Spring Security
        - ➢ Lombok 1.16.18
        - ➢ Flyway 4.2.0
        - ➢ Druid
        - ➢ AWS
        - ➢ Nginx
        - ➢ AngularJS 4
        - ➢ Jenkins (Ansible/Chef)
        - ➢ Kafka
2. Micro-services. We have separated 6 micro-services.
    - operr-common (all entities)[basic CRUD, Please write it on related project about advanced CRUD, like operr-base, operr-core, operr-customer, operr-driver, operr-message]

- operr-base (base, company, content, agency, ratebook, dashboard, dispatcher,group, language, promo)[CRUD]
- operr-core (insurance trip, phone-call trip, on-demand trip, billing, payment, goods, flight)[CRUD]
- operr-customer (customer,goods-owner)[CRUD]
- operr-driver (tracking,driver,vehicle,car-service)[CRUD]
- operr-message (chat-box,lost and found,emergency-call,SMS,SNS,SES,Kafka,feedback,compliance)[CRUD]
- operr-schedule (check driver license, check TLC license)
- operr-gateway(zuul, eureka, hystrix)

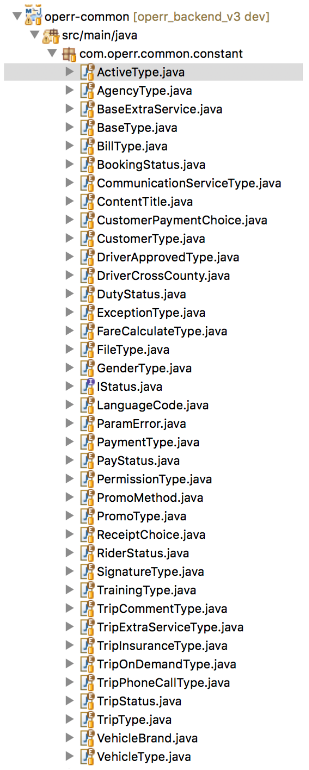


operr-base [operr_backend_v3 dev]
operr-common [operr_backend_v3 dev]
operr-core [operr_backend_v3 dev]
operr-customer [operr_backend_v3 dev]
operr-driver [operr_backend_v3 dev]
operr-gateway [operr_backend_v3 dev]
operr-message [operr_backend_v3 dev]
operr-schedule [operr_backend_v3 dev]

3. Package name format:
   - package name, like feedback
     - ➢ com.operr.message.api.feedback (controller)
     - ➢ com.operr.message.service.feedback (service interface)
     - ➢ com.operr.message.service.feedback.impl (service implementation)
     - ➢ com.operr.common.db.entity.feedback(entity)
     - ➢ com.operr.common.db.repository.feedback(basic repository [CRUD])
     - ➢ com.operr.common.request.feedback(request model)
     - ➢ com.operr.common.response.feedback(response model)
   - package name, like constant, util, exception
     - ➢ com.operr.common.constant (constant)
     - ➢ com.operr.common.util (util)
     - ➢ com.operr.common.exception (exception)
4. Class name and interface name format, like feedback:
   - FeedbackController.java (Controller)
   - FeedbackService.java (Service interface)
   - FeedbackServiceImpl.java (Service implementation)
   - Feedback.java (Entity)
   - RequestFeedbackModel.java (DTO)

- ResponseFeedbackModel.java (DTO)
- FeedbackRepository.java (Repository interface)
- Feedback(Class name)
- feedback(object name, don't use any shortname), should like this
  Feedback feedback = null;

5. Enum name format, all Enum is defined in operr-common
   - IStatus.java(interface)
   - ActiveType.java(Enum)
   - Please check the following image for other status



```
operr-common [operr_backend_v3 dev]
  src/main/java
    com.operr.common.constant
      ActiveType.java
      AgencyType.java
      BaseExtraService.java
      BaseType.java
      BillType.java
      BookingStatus.java
      CommunicationServiceType.java
      ContentTitle.java
      CustomerPaymentChoice.java
      CustomerType.java
      DriverApprovedType.java
      DriverCrossCounty.java
      DutyStatus.java
      ExceptionType.java
      FareCalculateType.java
      FileType.java
      GenderType.java
      IStatus.java
      LanguageCode.java
      ParamError.java
      PaymentType.java
      PayStatus.java
      PermissionType.java
      PromoMethod.java
      PromoType.java
      ReceiptChoice.java
      RiderStatus.java
      SignatureType.java
      TrainingType.java
      TripCommentType.java
      TripExtraServiceType.java
      TripInsuranceType.java
      TripOnDemandType.java
      TripPhoneCallType.java
      TripStatus.java
      TripType.java
      VehicleBrand.java
      VehicleType.java
```

6. How to use String in Class
   - Please use StringBuffer or StringBuilder.
   - Please use equalsIgnoreCase.
   - Please use null.
   - Please do not hard code string, please set it in file property.
   - Please use block finally

```
//wrong
String fullURL = url + carrier + "/" + flight + "/" + checkMethod +"/" +
+ "?appId=" + appId + "&appKey=" + appKey + (checkMethod.equals("dep") ?

//right
StringBuffer sb = new StringBuffer("abc");
sb.append(true).append("123").append("456");
```

```
//wrong
if(side == "driver") {

//right
if("driver".equalsIgnoreCase(side))) {
```

```
String mobileOs = "";

//better
String mobileOs = null;
```

```
public void method(){
  Base base = (Base)entityMap.get("base");

  line.get("action")
  ...
}
```

```
/**
* method description.
*/
public void methodName(){
  try{
    ...
  }catch(Exception e){
    e.printStackTrace();
    log.err();
  }finally{
    ...
  }
}
```

7. Format all class
    - Format Java code
    - Organize imports
    - Order all methods based on first character of method by ASC
    - The maximum number is 120 for each line in java class
8. Please do not use “System.out”. In the begining and end of method should use
    - logger.info(“---------------- start ------------------------”)
    - logger.info(“---------------- end ------------------------”)
    - logger.err(“---message---”+e.getMessage())as the first line in exception.
    - logger.err(“---e---”+e) as the second line in exception. before e.printStackTrace().
9. Please add comment for each method.
    - It will generate JDK document after releasing.
10. Please don’t write big body of method.
    - Please separate it to two methods If the body of method is bigger than 2/3 & height of screen.
11. Please use format name for CRUD.
    - For example, in FeedbackService.java
        - ➢ createFeedBack();
        - ➢ readFeedBack();
        - ➢ updateFeedBack();
        - ➢ deleteFeedBack();
12. Please write Junit Test method to test each method on service level.
    - For example, in FeedbackServiceTest.java
        - ➢ testCreateFeedBack();
        - ➢ testReadFeedBack();
        - ➢ testUpdateFeedBack();
        - ➢ testDeleteFeedBack();

13. Please don't forget transaction in service level.
    - We should not generate any dirty data on db.
    - If we have any error in the method, just make sure it can be rollback.
14. We use Druid to monitor all SQL time.
    - We should not generate any dirty data on db.
    - If we have any error in the method, just make sure it can be rollback.
15. We use radis to cache data.
    - For example, we want to read data from DB
        - we read data from redis.
        - If data is existed in redis, It will be return directly.
        - If data is not existed in redis, it will read data from db and save data into redis, and then return data from redis.
16. MongoDB calculate distance using spherical geometry.
    - https://docs.mongodb.com/manual/tutorial/calculate-distances-using-spherical-geometry-with-2d-geospatial-indexes/

17. Please don't forget to use swagger2 to generate API description on Controller level.
18. We use flyway to manage db script change.
19. Please don't use any database data to check business logic. We should keep database clean when we do release. If you need some static data, please add it as Enum.
20. You can have self design exception,constant,util and conf based on com.operr.core.exception/com.operr.core.util in operr-common for each micro-service.
    - operr-base
        - com.operr.base.exception
        - com.operr.base.constant
        - com.operr.base.util
        - com.operr.base.conf
    - operr-core
        - com.operr.core.exception
        - com.operr.core.constant
        - com.operr.core.util
        - com.operr.core.conf
    - operr-customer
        - com.operr.customer.exception
        - com.operr.customer.constant
        - com.operr.customer.util
        - com.operr.customer.conf
    - operr-driver
        - com.operr.driver.exception
        - com.operr.driver.constant
        - com.operr.driver.util
        - com.operr.driver.conf

- operr-message
  - com.operr.message.exception
  - com.operr.message.constant
  - com.operr.message.util
  - com.operr.message.conf
- operr-schedule
  - com.operr.schedule.exception
  - com.operr.schedule.constant
  - com.operr.schedule.util
  - com.operr.schedule.conf